Software Quality from Behavioural and Human Perspectives

Jamaiah Haji Yahaya[†], Aziz Deraman[††] and Abdul Razak Hamdan[†††]

[†,††, †††]Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), Bangi, 43650 Selangor, MALAYSIA

[††]Academic and Internationalisation Division, University Malaysia Terengganu, Kuala Terengganu, 21030, Terengganu, MALAYSIA

[†]Graduate Department of Information Technology College of Arts and Sciences, Universiti Utara Malaysia, Sintok, 06010 Kedah, MALAYSIA

Summary

Software quality is evolving beyond static assessment to include behavioural attributes and human aspects. These two groups of attributes are vital and necessary to balance between technical and non-technical (human) aspects in software assessment. PQM or Pragmatic Quality Model is a proposed model of quality, which composes of behavioural and human perspectives in assessment. This model provides opportunity to give priority or contribution of quality attributes to reflect the business requirement. Therefore, it is more practical that can suit different users and purposes. As for our research, PQM is used for assessment of software for certification process. This paper explains in detail this model of PQM.

Key words:
Software quality model, behavioural perspective, human perspective and software assessment.

1. Introduction

In the new global economy and borderless world, computer has become a central issue for surviving in business. Companies are competing to produce software which are claimed to be good and fulfil user's expectation and requirements. There are questions arise regarding the status of software being developed either in-house or off-the-shelf product. Some of the questions are: How do we determine the quality of the software being developed? What are the mechanisms to assess software product? How do we ensure and guarantee the quality of a particular software product? From our observation we believe users are concerned regarding quality of software delivered to them and they expect software are in good quality that meet certain standard. Furthermore, stakeholders need to put their trust and confidence over the software being developed or purchased and used in the organizations.

In the earlier part of software development, software quality is measured through static assessment of code's structure. Fortunately, a new generation realizes that software quality is more than just static features. People who involves in software as users, stakeholders, developers or practitioners are becoming more concern on the other aspects or views of quality. It should also comprise non-functional such as behavioral and human aspects.

2. Software Quality And It's Problem

Software has become very important in everyday life thus quality of the software is a great concern, vital and critical. It requires continuous improvement to retain survival of a software company either in private or public sector. Software quality assurance affects both immediate profitability and long-term retention of customer goodwill. In January 2002, Bill Gates demanded Microsoft to think of quality of their product and to produce less defects in its products [3]. He seems to have recognized the importance and emergence of this new definition of quality. He sent the following e-mail to all employees reminding them the necessities and higher priorities of Trustworthy Computing [4].

The past decade has seen rapid development and diffusion of software and ICT related technologies not only in Malaysia but also worldwide. In Malaysia, statistic produced by MSC (2007) states that 51% from 1372 operational MSC status companies are functioning on software development, and 13% are working on support services, 9% are running on creative multimedia, 9% are dealing with hardware designs, while 11% are functioning on Internet Based Business (IBB) and 7% on Shared Services/Outsourcing (SSO) (refer to Figure 1). It shows that software development industry has a significant contribution and impact to the development and success of the MSC. Thus, an appropriate attention

Manuscript received August 5, 2008.
Manuscript revised August 20, 2008.

is necessary to monitor the quality of software product delivered by these companies as well as other non-MSC companies, organizations and public sectors.

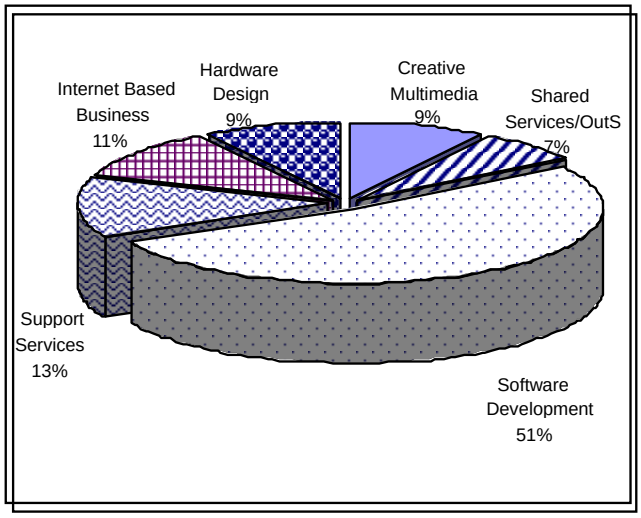

Figure 1. MSC Status Companies By Technologies Cluster: Operational At January 31, 2007, Source: [5]

Companies could not provide any justification on the quality of their products to the users and users are left with uncertainties on the standard and quality of the software [1],[6]. This raises legal and moral questions: To what extent is an organization that develops and/or uses software responsible for its result? How to monitor software quality and correctness? And to what lengths should such organizations go to assess software quality and correctness?

ISO defines software as "all or part of the programs, procedures, rules, and associated documentation of information processing system". Software product is defined as "the set of computer programs, procedures, and possibly associated documentation and data designated for delivery to a user" [7]. The term product from the view of software engineer covers the programs, documents, and data. While from the view of user's product is the resulted information that somehow makes the user's world better.

General expressions of how quality is realized in software dealing with "fitness for use" and "conformance to requirements". The term "fitness of use" usually means characteristics such as usability, maintainability, and reusability. On the other hand, "conformance to requirements" means that software has value to the users [8]. International Organization for Standardization (ISO) defines quality as "the totality of features and characteristics of a product or services that bear on its ability to satisfy stated or implied needs" [7],[9]. IEEE defines software quality as – a software feature or characteristic used to assess the quality of a system or component [10]. Software quality is also defined as the fitness for use of the software product and to conform to software requirements and to provide useful services [38]. Later, software quality is defined as "conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software" [11].

In many organizations, software is considered as one of the main assets with which the organization can enhance its competitive global positioning in current global economic era. To remain competitive, software firms must deliver high quality products on time and within budget. Software Engineering Institute's Capability Maturity Model (CMM) (cited in Slunghter, Harter and Krishnan [12]) reports the following quote from a software manager: "I'd rather have it wrong than have it late. We can always fix it later". Thus, many complaints have been reported regarding quality of the software. These complaints claimed that software quality is not improving but rather deteriorates steadily and worsening. Therefore, users report and claim that software is being delivered with bugs that need to be fixed and dissatisfied with the product [2],[1].

Peter J. Denning [2] presented his argument that "software quality is more likely to be attained by giving much greater emphasis to customer satisfaction. Program correctness is essential but is not sufficient to earn the assessment that the software is of quality and is dependable". Software quality and evaluation not only deal with technical aspects but also in dimensions of economic (managers' viewpoint), social (users' viewpoint) and as well as technical (developers' viewpoint) [13].

Dromey [26] stated that an ultimate theory of software quality is like "the chimera of the ancient Greeks, is a mythical beast of hybrid character and fanciful conception. We obliged, however, to strive to make progress, even though we realize that progress often brings a new set of problems". He also suggested that software quality usually referred to high-level attributes like functionality, reliability and maintainability and the important thing to focus was on the priority needs for the software. Dromey stated that priorities vary from product to product and project to project.

Software product quality can be evaluated via three categories of evaluations: internal measures, external measures and quality in use measures [14]. Internal measuring is the evaluation based on internal attributes

typically static measures of intermediate products and external measuring is based on external attributes typically measuring the behaviour of the code when executed. While the quality in use measures include the basic set of quality in use characteristic that effect the software. This characteristic includes effectiveness, productivity, safety and satisfaction. This measurement is an on-going research of SQuaRE which is the next generation of ISO 9126 but not fully published and accepted currently. SQuare quality model consists of internal and external measures that include quality in use aspects. It presents similar concept of characteristics and subcharacteristics as in ISO 9126 approach [15].

3. Software Quality Models

Software and quality are among the most common topic of discussions on computers. Fenton and Pfleeger [16] suggest, "Without an accompanying assessment of product quality, speed of production is meaningless". This observation has led to the development of software quality model that are measured and combined with productivity models.

Thus, several quality models are available from literature and the most well known models are McCall, Boehm, FURPS, ISO 9126, Dromey and Systemic. The following sections will discuss briefly on each of these models.

3.1 The McCall model (1977)

The McCall quality model is one of the earliest models and commonly called the FCM (Factor Criteria Metric) model. The model is usually constructed in a tree-like fashion. The upper branches hold important high-level quality attributes, such as reliability and usability, which will be quantified. Each quality attribute is composed of lower-level criteria ([17],[18]).

This model puts emphasis on grouping quality factors into several working areas viz product operation, product revision and product transition. The factors associated with the working areas are: Correctness, reliability, efficiency, integrity, usability, maintainability, testability, flexibility, portability, reusability and interoperability.

In this model factors are not directly measured and therefore a set of metrics are defined to develop the relationship. McCall defines metrics in a form of checklist that is used to grade attributes of the software. It is interesting to notice that some of the factors are still relevant and as fresh today as they were in the 1970's. However McCall model does not include functionality.

3.2 The Boehm model (1978)

The Boehm model ([19], [18], [17]) is similar to McCall model in that it represents a hierarchical structure of characteristics, each of which contributes to total quality. Boehm model views software with general utility. It looks at utility from various dimensions, considering the type of users expected to work with the software once it is delivered. General utility is broken down into portability, utility and maintainability. Utility is further broken down into reliability, efficiency and human engineering. Maintainability is in turn broken down into testability, understandability and modifiability. This model is presented in levels called primary uses, intermediate constructs and primitive constructs.

3.3 The FURPS model (1987)

Hewlett-Packard developed a set of software quality factors that make up its name FURPS. The FURPS model takes five characteristics of quality attributes - Functionality, Usability, Reliability, Performance and Supportability. When the FURPS model is used, two steps are considered: setting priorities and defining quality attributes that can be measured [18]. One disadvantage of this model is that it does not take into account the software product's portability [20].

3.4 ISO 9126 (1991)

ISO 9126 defines product quality as a set of product characteristics. The characteristics that govern how the product works in its environment are called external quality characteristics. The characteristics relating to how the product is developed are called internal quality characteristics. ISO 9126 indicates six main quality characteristics which are associated with several subcharacteristics [7]. Many researches done investigated software assessment and quality using the ISO/IEC 9126 model as their guidelines in the assessment. Examples are PROFES[28], Torchiano, Sorensen and Wang [29], Cote et al. [25] and Adnan and Bassem [24]. These studies showed that ISO 9126 model is an appealing model, irrespective of some limitations. The characteristics defined in this model are functionality, reliability, efficiency, usability, maintainability and portability.

One advantage of ISO 9126 model is that it is intended to be exhaustive and identifies the internal characteristics and external quality characteristics of a software product. ISO 9126 model can be used as a practical approach for defining quality and the questionnaire based method [23]. It has been invented

since 1991 and today, it is still being accepted and used in researches that deal with software quality [24],[25].

However, at the same time it has the disadvantage of not showing clearly how these aspects can be measured. Pfleeger reports some important problems associated with ISO 9126. The problems are: there are no guidelines on how to provide an overall assessment of quality and rather than focusing on the user view of software, the model's characteristics reflect a developer view of software [39].

There are another issues that relate to quality attributes, among them are priorities and different views of quality among users, stakeholders and managers. It is recognized that views of users, the developers and managers are different. A manager is more interested in the overall quality rather than a specific quality characteristic thus requires assigning weight to reflect the business requirements [21],[22]. In ISO 9126 all attributes are equally important.

3.5 The Dromey model (1996)

Dromey [26] proposes a working framework for building and using a practical quality model to evaluate requirement determination, design and implementation phases. Dromey points out that high level quality attributes, such as maintainability, functionality and reliability, cannot be built into the system. The alternative way to input quality into software is by identifying a set of properties and build them up consistently, harmoniously and fully to provide high level quality. Links must be established between tangible product properties and intangible quality attributes.

Five steps of quality model were constructed and refined. Dromey includes high-level quality attributes: functionality, reliability, efficiency, usability, maintainability, portability, reusability and process-mature. In comparing to ISO 9126, additional characteristics like process maturity and reusability are noticeable. Subattributes associated with reusability are machine-independent, separable and configurable. While process maturity include client-oriented, well-defined, assured and effective attributes. Process maturity is an attribute which has not been considered in the previous models.

3.6 The Systemic Quality Model (2003)

The systemic model is differed from the previous model mentioned above. This model is developed by identifying the relationship between product-process, efficiency-effectiveness and user-customer to obtain global systemic quality [20], [27. The model proposed focuses on product quality, which includes efficiency and effectiveness and the concept of systemic global quality. This model enables the systemic quality model to obtain specifically the product's efficiency and effectiveness dimension. The elements of the systemic quality model for software products include functionality, reliability, usability, efficiency, maintainability and portability. This part of the model is for measuring effectiveness of a product. While to measure the efficiency of a product, this model takes into account the systemic quality model by Callaos & Callaos and Dromey model. Lastly this model considers process effectiveness and efficiency. The disadvantages of this model are that it does not cover the user requirements and conformant aspects.

Analysis done on the different models demonstrates that different quality characteristics associated with these different models. It shows that the main quality characteristics found in majority of the models are: efficiency, reliability, maintainability, portability, usability and functionality, which are presented in more recent models. These characteristics appear in all models and therefore, are considered as essential and vital. Table 1 summarises the quality characteristics identified in different models from McCall, Boehm, FURPS, ISO9126, Dromey and Systemic. This table also illustrates our proposed Pragmatic Quality Model (PQM), which is used in certification process conducted in our research (see also [36] for detail).

Fenton presents a framework for software measurement which classifies software attributes (characteristics) [39]. Fenton makes a distinction between attributes which are internal and external. Internal attributes refer to the measurement of merely terms of product. Examples are size, structure and modularity. While external attributes are measurements of how the product relates to its environment. This framework suggests the use of internal attributes to infer and predict the levels of external attributes. Fenton framework is beneficial for the developer to estimate and predict the future quality of the software product but it is considered as unimportant for the users. Users dismiss the importance of internal attributes because they are unable to access the internal attributes of the software themselves.

Even though there are several models of quality available from literature, it still believed that quality is a complex concept. Quality is in the eye of the beholder and it means different things to different people and highly context dependent [30] [31]. Therefore, "software quality is nothing more than a recipe. Some like it hot, sweet, salty or greasy" [32]. Thus , there can be no single simple measure of software quality acceptable to everyone.

Literature on software quality shows that there are a number of characteristics that contribute to the behavioral aspects of quality. A classification of characteristics might be necessary to group characteristics according to their importance.

Our research proposes an attribute weight and classification to be included in the quality model. This will discuss in the following section.

4. PQM: Pragmatic Quality Model From Behavioural and Human Perspectives

In this paper we invite the software community to view quality in more practical terms and propose several modification and enhancement to measure software quality. The PQM consists of four main components: behavioural attributes, impact attributes, responsibility, and weight. The features of the components are presented next.

4.1 The Behavioural Attributes

The behavioural attribute is defined as the external quality characteristic of specific software and how it behaves in the environment. These attributes include efficiency (*E*), functionality (*F*), maintainability (*M*), portability (*P*), reliability (*R*), integrity (*I*) and usability (*U*). Each attribute is made up of several subattributes and then broken down into several metrics that shows the measurement aspects of the attributes. The behavioural attributes are derived from ISO 9126 attributes with the integrity aspect included. ISO 9126 model is a generic quality model for any software product but requires some customization and enhancement for particular case [19],[40]. In the age of hackers and firewalls, the importance of integrity aspect has increased. This attribute measure the ability to with-stand attack on its security that comprises of program, data and document. It covers threat and security aspects. Findings from previous survey indicated the importance of integrity in software quality attributes.

In PQM, attributes are decomposed into several subattributes and then a further level of decompositions to associate with directs measurable metrics. Each of the subattributes and metrics comprises of information on interviewees.

The quality of product based on behavioural aspects is formulated as:

$$Q_B = f(R, E, F, M, P, I, U) + \varepsilon \quad (1)$$

Software quality (Q) is thus a function of these combined attributes plus an error term (ε) that represents a quality aspect that these seven attributes can't define.

4.2 The Impact Attributes

The impact attribute defined in PQM refers to the human aspect of quality toward the product. It illustrates the impact of the software in term of quality to the users and also measures the conformity of software to the user requirement. This attribute is important to balance the quality model between technical measurement of software and human factor [34]. Similar to behavioural attributes, the impact attribute is made up of several subattributes and metrics that show the measurement of the attributes. The impact attribute is decomposed into two distinct subattributes, which by means of user perceptions and user requirements. The metrics include measures of popularity, performance, trustworthiness, law and regulation, recommendation, environmental adaptability, satisfaction and user acceptance. Two categories of attributes are defined :-

User perception (*Up*)= {popularity, performance, law & Regulation, Recommendation, Trsutworthiness, Requirement & Expectation, Environmental adaptability}

User Requirement (*Ur*) = {User acceptance, satisfaction}

Therefore,

$$Q_H = f(Up, Ur) + \varepsilon \quad (2)$$

4.3 Responsibility and Measurement of Metrics

The third component in PQM is the responsibility. It is defined as the responsibility person to answer the questions related to metrics. It is also named as the interviewee in this model. The PQM has identified specific interviewee to responsible in giving the assessment score of each metrics.

The measurements used are Likert scale of 1 to 5 based on collaborative perspective among assessment team members. Likert scale is defined as something that is the satisfaction measured based on perception. The Likert technique presents a set of attitude statements. Subjects are asked to express agreement or disagreement of a five-point scale. Each degree of agreement is given a numerical value from one to five. Thus a total numerical value can be calculated from all the responses. The scale used in this approach is recommended as 1 = unacceptable, 2 = below average, 3 = average and 4 = good, 5= excellent.

4.4 Classification of Attributes and Weight Factors

Several researchers have applied weights in their calculation of different domains. Kontio [37] applied weights in calculations of measuring COTS selection technique while Hampton and Quinn [41] applied weights concept in project management measurement. In this method, score are computed by multiplying weight and score of each criterion.

The weighting factors defined in PQM is based on findings from previous survey [35]. In the survey we asked respondents to indicate levels of consideration which are by means of 1=not considered, 2=low consideration, 3=average, 4=high consideration and 5=very high consideration of all the quality attributes. These criteria are taken into account during assessment exercise of software product in their organizations.

For the purpose of this classification, we are interested to analyse the two modes of considerations that are *Very High Consideration* and *High Consideration* only. Data management and analysis was performed using SPSS and the weight of each attributes is calculated using the following formula:-

$$\text{TotalVH} = \sum_{a=1}^{n} \text{VH}_a \quad (3)$$

where n = number of attributes defined in the analysis and VH is the score for Very High Consideration. Then,

$$\text{Weight}_a = (\text{VH}_a / \text{TotalVH}), \quad (4)$$

and

$$\%\ \text{Weight}_a = (\text{VH}_a / \text{TotalVH}) * 100 \quad (5)$$

where subscript *a* represents an attribute.

From the analysis, function point approach is used to group and classify attributes into three distinct classifications namely low, moderate and high. Then, the attributes are sorted into these classifications according to the calculated weight score (4) and (5). The analysis shows that functionality is 14.29% more important compares to other quality attributes defined in this model. It obtains the highest weight in this analysis. Reliability is considered 12.34% more important and integrity is considered 11.69% important. These three attributes (functionality, reliability and integrity) are classified in the classification group of high. Second group of classification defined as moderate includes safety (8.44), efficiency (9.09%), maintainability (7.79%) and usability (7.79%). On the other hand, the third group of classification defined as low includes flexibility (5.84%), Interoperability (6.49%), Intraoperability (5.84%), portability (5.19%) and survivability (5.19%). The classification analysis and method are discussed in detail in [35].

For the purpose of assessment and certification applied in this research we therefore assign weight factor for each group accordingly. This is consistent with the requirements of having different weights for attributes [4, 21]. Table 2 demonstrates the classification of attributes and its weight factor.

5. Formulating Quality of Product

Underlying our equation is that particular software can achieve a specific quality level by combining each attribute's priorities or contributions. In this particular case, equation will describe the degree to which the software contains a particular attributes. We measure contributions or priorities in weight units i.e. 1,2,3,4,5,...9,10. Each of these attributes is classified into three main classifications, which by means are high, moderate and low (see Table 2). Therefore, each attribute is not necessarily equal contributions in quality function. Refer to [35] for detail on this classification.

The formulation of quality of software product based on behavioural attributes is defined as follows:-

$$Q_B = w_R R + w_P P + w_F F + w_U U + w_I I + w_M M + w_E E \quad (6)$$

where the sum of weights in the equation is 1.0, w is the weight factor of attribute.

(4.2)

Table 2 Classifications Of Attributes And Its Weight Factor

Levels	Attributes	Weight Factor
Low	Flexibility Intraoperability Interoperability Portability Survivability	1-4
Moderate	Safety Efficiency Maintainability Usability	5-7
High	Functionality Reliability Integrity	8-10

The second aspect of quality which deals which human aspect is shown in the following equation:-

$$Q_H = w_j Up + w_k Ur \ . \quad (7)$$

This attributes of quality is vital and important to balance between technical and human aspects [34]. We assume that in this function weight of these attributes are equal to 1. This assumption is made because these attributes have no influence on the behaviour of the software in the environment. The total formulation of quality of a software product is characterized as follows:-

$$Q_P = Q_B + Q_H \quad (8)$$

Voas & Agresti suggest in chemical analogy, quality of software is more a compound than a mixture [4]. Thus this linear equation of quality is an acceptable model to represent quality of software based on behavioural and human perspective.

6. Model Testing and Evaluation

PQM has been applied in software certification model developed by our research group. The whole process has been implemented and tested in real case study. We have tested the model collaboratively with three large organizations in Malaysia. The certification process developed by this research group requires a software quality model as a benchmark and standard of the assessment. The quality model must suit with the certification specifications and requirements. Thus, PQM is designed to fulfill certification requirements.

Three software operated in their environment have been identified and tested applying the PQM for assessment. Table 4 shows an example of the result showing the scores obtained by the behavioural attributes and the impact attributes (human aspects) defined in this model.

Column 1 refers to the maximum value of each score by respondents. Column 2 refers to the weight values given by the owner of the software or any appointed individual, Column 3 is the average score obtained by this assessment. Based on the weight assigned, scores are calculated (see [36] for detail) as shown in column 4. Final values (column 5) are the computed values of quality score obtained according to attributes. In this case, the score for behavioural attributes is 69.4% while the score of the impact attributes is 73.3%. The total quality score of this product is computed by averaging the behavioural and impact attributes scores. The total score of this product is 71.3%. In our case, this score is mapped into a certification level to obtain the relevant certification status of this product.

Table 3 illustrates the different quality attributes and characteristics defined in previous models. It also shows our model, which consists of the current and vital characteristics of behavioural attributes and human aspects of quality. If assessment of software product is done without using any weights or priorities as in previous model, it shows slightly different result (refer to Table 5). In this case the result obtained is 68.9% while if using PQM in assessment, the product obtained 71.3% in term of quality.

7. Conclusion

PQM is a quality model that provides flexibility in identifying quality of software product based on individual and organization requirements. It composes of two main components: the behavioural and the impact attributes. Literature suggests that lack of mechanism and techniques of software assessment that cover both aspects of software quality. The behavioural attributes exhibit how the software behaves in the environment. While the impact attributes cover how human views and perceive ness toward the software. These two components of quality produce a balance between technical requirements and human factor. Table 3 shows the summarization of quality characteristics in different models. PQM is being used as the quality model for certification of software based on product quality approach. PQM has been implemented and tested in real case studies involving three large organizations in Malaysia. These illustrate the practicality and feasibility of the PQM in real world.

References

[1] J.A. Whittaker, & J.M. Voas,, “50 years of software: Key principles for quality,” *IEEE IT Pro* (Nov/Dec), pp.28-35, 2002.

[2] P.J. Denning, “What is software quality?” *A Commentary from Communications of ACM* (January), 1992.

[3] C.C. Mann, “Why software is so bad?” *MIT Technology Review* 105(July/August), pp. 33-38, 2002.

[4] J. Voas & W.W. Agresti, “Software quality from a behavioral perspective,” *IT Proffesional* (July/August), pp. 46-50, 2004.

[5] MSC. MSC Malaysia Progress Updates. http://www.msc.com.my/xtras/fact_figures/msc.asp , 2007. [May 22, 2007].

[6] Compuware,. “Application quality and its business impact- a view from the top,” http://www.compuware.com/whitepapers/ok.asp. [13 January 2004].

[7] ISO/IEC 9126, "Software quality characteristics and metrics-Part2: External metrics", Technical Report, ISO/IEC JTC1/SC7/WG6, 1996.
[8] I. Tervonen, "Support for quality-based design and inspection," *IEEE Software* (January), pp. 44-54, 1996.
[9] M.G. Jenner, *Software Quality Management and ISO 9001*. New York: A Wiley/QED publication, 1995.
[10] IEEE. "IEEE standard for a software quality Metrics Methodology", http://ieeexplore.ieee.org/xpl/standards.jsp, 1993. [August 20, 2005].
[11] Galin, Daniel, *Software Quality Assurance: From Theory to Implementation*. Harlow: Pearson Addison Wesley, 2004.
[12] Slaughter, S.A., Harter, D.E. & Krishnan, M. S., "Evaluating the cost of software quality," *Communications of The ACM* 41(8): 67-73, 1998.
[13] L. Buglione & A. Abran, 1999. A quality factor for software. Proceeding of QUALITA99, 3rd International Conference on Quality and Reliability, pp. 335-344.
[14] W. Suryn, A. Abran, P. Bourque & C. Laporte, "Software product quality practices: Quality measurement and evaluation using TL9000 and ISO/IEC9126," *Proceeding of the 10th International Workshop, Software Technology and Engineering Practice (STEP)* 2002.
[15] W. Suryn, A. Abran & A. April, "ISO/IEC SQuaRE: The second generation of standards for software product quality", http://www.lrgl.uqam.ca/publications/pdf/799.pdf, 2003. [20 Sept 2004].
[16] N.E. Fenton & S.L. Pfleeger, *Software Metric: A rigorous & practical approach*. London: Thompson Computer Press, 1996.
[17] F.E. Norman & S.L. Pfleeger, *Software Metrics: A Rigorous and Practical Approach*, Second Edition. Boston: PWS Publishing, 1997.
[18] K. Khosravi, & Y.G. Gueheneuc, "A quality model for design patterns", http://www.yann_gael.gueheneuc.net/work/Tutoring/Documents/041021+Khosravi+Technical+Report.doc.pdf, 2004. [26 October 2005].
[19] M.F. Bertoa, J.M. Troya & A. Vallecillo, "A survey on the quality information provided by software component vendors," *Proceedings of 7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2003)*, pp. 25-30, 2003.
[20] M. Ortega, M. Perez, & T. Rojas, "Construction of a systemic quality model for evaluating a software product", *Software Quality Journal* 11: 219-242, 2003.
[21] Eagles, "Evaluation of natural language processing systems", Final Report (Sept). http://www.issco.unige.ch/ewg95/, 1995 [18 May 2007].
[22] M. Svahnberg, C. Wohlin, L. Lundberg & M. Mattsson, "A method for understanding quality attributes in software architecture structures," *ACM* http://portal.acm.org/citation.cfm?id=568900, 2002. [22 May 2007].
[23] R. Hendriks, R.v. Vonderen & E.v. Veenendaal, "Measuring software product quality during testing," *Proceedings of Conference European Software Quality Week*, 2000.
[24] R. Adnan & M. Bassem, "A new software quality model for evaluating COTS components," *Journal of Computer Science* 2(4): 373-381, 2006.
[25] M.A. Cote, W. Suryn & C.Y. Laporte, "Evolving a corporate software quality assessment exercise: A migration path to ISO/IEC 9126," *SQP* 6(3): 4-17. http://www.mitre-corporation.com/work/best_papers/best_papers_04/martin_software/martin_software.pdf , 2004. [10 July 2007].
[26] G.R. Dromey, "Software product quality: Theory, model and practice. Software Quality Institute," Griffith University, Brisbane, Technical Report. http://www.sqi.gu.edu.au, 1998 [23 August 2006].
[27] G. Rincon, M. Alvarez, M. Perez & S. Hernandez, "A discrete-event simulation and continuous software evaluation on a systemic quality model: An air industry case," *Information & Management Journal* 42: 1051-1066. http://www.sciencedirect.com, 2004. [14 April 2007].
[28] PROFES, "PROFES www-site", http://www.ele.vtt.fi/profes/, 1997. [18 May 2007]
[29] M. Torchiano, L. Jaccheri, C.F. Sorensen & A.L. Wang, "COTS product characterization," *14th International Conference on Software Engineering and Knowledge Engineering, SEKE '02*, pp. 335-338, 2002
[30] J. Voas & P. Laplante, "Standards confusion and harmonization," *Computer* 40(7): 94-96, 2007.
[31] B. Kitchenham & S.L. Pfleeger, "Software quality: The elusive target," *IEEE Software* (January): 12-21, 1996.
[32] J. Voas, "Software's secret sauce: The "-ilities"," *IEEE Computer* (November/December): 14-15, 2004.

[33] D. Longstreet, "Fundamentals of function point analysis," http://www.SoftwareMetrics.com , 2005. [23 August 2006].
[34] C.A. Dekkers & P.A. McQuaid, "The dangers of using software metrics to (Mis)Manage.," *IT Pro* (March/April): 24-30, 2002.
[35] J.H.Yahaya, A. Deraman & A.R. Hamdan, "Software product certification model: Classification of quality attributes," *The First Regional Conference of Computational Science and Technology (RCCST 07), Kota Kinabalu,* pp. 436-440, 2007.
[36] J.H. Yahaya, A. Deraman & A.R. Hamdan, "A model and methodology for software product certification," *Proceedings of the National Conference Software Engineering and Computer System (Nacses07), Cherating, Kuantan, 2007.*
[37] J. Kontio, " A Case Study in Applying a Systematic Method for COTS Selection," 201-209. *Proceedings of the 18th International Conference on Software Engineering.* IEEE *Computer Society Press*, pp. 201-209, 1996.
[38] G.G. Schulmeyer & J.I. McManus, 1998. *Handbook of Software Quality Assurance*, 3rd Edition. New Jersey: Prentice Hall.
[39] S.L. Pfleeger, 2001. *Software Engineering: Theory and Practice*, 2nd ed. Upper Saddle River, N.J: Prentice Hall.
[40] J. Boegh, "Certifying software component attributes," *IEEE Software* (May/June): 74-81, 2006.
[41] I.M. Hampton & B.W.T. Quinn, Software project measurement criteria. http://www.ieeexplore.ieee.org, 2000.[20 Dec, 2006].

Table 3: Quality characteristics present in Pragmatic Quality Model and Previous Models

Quality characteristics	McCall (1976)	Boehm (1978)	FURPS (1987)	ISO 9126 (1991)	Dromey (1996)	Systemic (2003)	PQM (2007)
Testability	x	x					
Correctness	x						
Efficiency	x	x	x	x	x	x	x
Understandability		x			x		
Reliability	x	x	x	x	x	x	x
Flexibility	x						
Functionality			x	x	x	x	x
Human engineering		x					
Integrity	x						x
Interoperability	x						
Process Maturity					x		
Maintainability	x	x	x	x	x	x	x
Changeability		x					
Portability	x	x		x	x	x	x
Reusability	x				x		
Usability			x	x		x	x
Performance	x		x				
User Conformity							x

Table 4: Assessment Analysis of Product Y: An Example

Behavioural Attributes	Max Value	Weight	Score Obtained	Score	Quality Score (%)
	(1)		(3)	(4)	(5)
Efficiency	5	7	4.08	0.539	10.8
Functionality	5	9	3.69	0.627	12.5
Maintainability	5	7	2.66	0.351	7.0
Portability	5	4	3.55	0.268	5.4
Reliability	5	9	3.36	0.571	11.4
Usability	5	7	2.95	0.390	7.8
Integrity	5	10	3.83	0.723	14.5
TOTAL		53		3.469	69.4
The Impact					
User Conformity					73.3
Total Product (TQP)					**71.3**

Table 5: Comparing Assessment Analysis of Product Y

Behavioural Attributes	Max Value	Weight	Score Obtained	Quality Score (%)
	(1)	(2)	(3)	(4)
Efficiency	5	1	4.08	81.6
Functionality	5	1	3.69	73.8
Maintainability	5	1	2.66	53.2
Portability	5	1	3.55	71.0
Reliability	5	1	3.36	67.2
Usability	5	1	2.95	59.0
Integrity	5	1	3.83	76.6
TOTAL		53		68.9
The Impact				
User Conformity				0
Total Product (TQP)				**68.0**

Jamaiah Haji Yahaya is a post doctoral fellow in Department of Computer Science, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM). She is also a lecturer in Information Technology at Northern University of Malaysia. She received her BSc (Computer Science) from University of Wisconsin – La Crosse, USA (1986), MSc (Information System) from University of Leeds, UK (1998) and PhD in Computer Science from Universiti Kebangsaan Malaysia in 2007. Her research interests includes software certification, software quality and software management. Her email addresses are <jamaiah@uum.edu.my> and <jhyahaya@yahoo.com>.

Aziz Deraman is a professor in Software Engineering in Department of Computer Science, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia. Currently he is the deputy vice chancellor at University Malaysia Terengganu, Malaysia. He received BSc degree from UKM (1982), MSc degree from Glasgow University (1984) and PhD from University of Manchester Institute of Science and Technology (UMIST) in 1992. His research interests include IT strategic planning, software management and certification, medical computing and community computing. His email addresses are <a.d@umt.edu.my> and <ad@ftsm.ukm.my>.

Abdul Razak Hamdan is a professor in System Management and Science, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM). Currently he is the dean of this faculty. He received his BSc degree from UKM (1975), MSc degree from University of Newcastle Upon Tyne, UK (1977) and PhD in Artificial Intelligence from Loughborough University of Technology, United Kingdom in 1987. His research interests include ICT & Strategic Policy, Intelligent Decision Support, Medical Data Mining and Jawi Pattern Recognition. His email address is <arh@ftsm.ukm.my>.