The Development of Pragmatic Quality Model for Software Product Certification Process

Jamaiah H. Yahaya
Faculty of Technology and Information Science, National University of Malaysia
43600 Bangi, Selangor, Malaysia
Tel No: 0060389216088
Fax No:0060389256732

Aziz Deraman
Faculty of Technology and Information Science, National University of Malaysia
43600 Bangi, Selangor, Malaysia
Tel No: 0060389216725
Fax No:0060389256732
ad@ftsm.ukm.my

Abdul Razak Hamdan
Faculty of Technology and Information Science, National University of Malaysia
43600 Bangi, Selangor, Malaysia
Tel No: 0060389216725
Fax No:0060389256732
arh@ftsm.ukm.my

Faculty of Information Technology,
Northern University of Malaysia,
Sintok, 06010, Kedah
Malaysia
jamaiah@uum.edu.my

Abstract- **The emergence of Multimedia Super Corridor (MSC) in Malaysia in 1996 was the starting point for the blooming of software and ICT related companies. Despite that, not much attention was given to the quality of the software product that was being developed by different categories of companies. These companies could not justify the quality of their products to the users and the users are left with uncertainties on the quality of the software. Our previous survey identified the need for independent software assessment and certification. Therefore, we propose a conceptual model of software certification process by product quality approach. It is designed to be applied by an independent certification body or any appointed institution. The main focus of this paper is the development of pragmatic quality model that will be applied in the certification process.**

I. INTRODUCTION

Software quality issue is recognized as an important issue in the last few years as we see a tremendous growth of software companies all over the world. In Malaysia, the emergence of the Multimedia Super Corridor (MSC) was the most exciting initiative for the global information and communication technology (ICT) industry. It has been conceptualized in 1996 and later grows dynamically hosting more than 900 multinationals, foreign-owned and home-grown Malaysian companies. These companies focused on multimedia and communications products, solutions, services and research and development. There are a number of services and software being developed by these MSC companies and as well as many other companies in Malaysia. Many innovative applications has been developed by the MSC focusing on the development of Smart Schools, Telehealth, e-business, smartcard technology, R&D Cluster, electronic government and technopreneurship. There are many flagship applications being developed with different categories in different respective industries.

The questions that need to be considered here is the quality of this software and how we determine the quality of the software being developed. In addition, the production of the software is still costly; too error prone takes too much time and involves too much risks. Therefore, users are becoming more aware of the importance of software quality in the various software types especially for safety-critical functions. Users expect the software to be developed to a certain standard to meet their requirements and expectations.

If we look at different scenarios such as medical and drugs, there is an approved source to endorse drugs and medicine available in the market. Even though a consumer may not be able to assess accurately if a particular drug is safe, they can be reasonably confident that drugs obtained from approved sources have an endorsement of the U.S Food and Drug Administration (FDA) which confers important safety information. Therefore, similar approach is anticipated to work with software products. With the development of software certification, users may be able to choose the correct software that meets their requirement even though the users do not understand the processes and program underlying the completed software development. The Capability Maturity Model (CMM) that was developed by Software Engineering Institute is an example of a mechanism for software quality improvement. It is based on software development process and defines in term of maturity level. There is also another mechanism for improvement and assessment but involves people and skill, examples are People Capability Maturity Model (PCMM) and British Computer Society.

This paper focuses on related issues of software certification process by product quality approach. First, it discusses the software quality and its problem

1-4244-0220-4/06/$20.00©2006 IEEE

and follows by issues in software certification. Section four of this paper discusses the pragmatic quality model and section five presents a conceptual model for software product certification process. The last section concludes our discussion in this paper.

II. SOFTWARE QUALITY

Quality in the software context means involving variety of quality attributes, for example performance, security, reliability and other attributes. International Organizaton 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"[1]. IEEE defines software quality as – a software feature or characteristic used to assess the quality of a system or component. The features or characteristics are broken down into several subcharacteristics and metrics. Software quality metrics is defined as "a function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affect its quality" [2]. In other words, quality here means satisfying the customer based on certain value or degree that is interpreted from some data. Generally, people think of quality as conformance and compliance to specification continuously and consistently.

Users from various backgrounds are exposed to different types of software products and they are left with uncertainties in selecting the right product for the right requirements. This is due to lack of proper mechanism and standard in conforming quality of the software product against the user's requirements.

Many complaints are reported concerning the issues of software quality. Some of the complaints and comments are from the US Department of Defense's CIO, publisher of CIO Magazine and the President of ACM [3]. Both of them agreed that software today is getting worse and no good quality compared to previous years. Vendors are claimed to be delivering softwares with bugs that needed to be fixed. The discussion on this topic leads to the lacking of standard and recognised authority to conduct the assessment.

A literature on this subject covers several software quality models: McCall, Boehm, FURPS, ISO9126, Dromey and Systemic. Main quality characteristics found in majority of the models are: efficiency, reliability, maintainability, portability, usability and functionality, which were presented in more recent models [4]. These characteristics appear in all models and therefore, are considered as essential and vital.

The McCall quality model [5,6,7] 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, that need to be quantified. Each quality attribute is composed of lower-level criteria. Three working areas that McCall model highlighted are: 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.

The Boehm model [5,7,8] is similar to McCall model in that it represents a hierarchical structure of characteristics, each of which contributes to total quality. Boehm model view the software with general utility.

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 [1]. The characteristics relating to how the product is developed are called internal quality characteristics. ISO 9126 indicates six main quality characteristics which associated with several subcharacteristics: functionality, efficiency, maintainability, reliability, usability and portability (see Table 1). One advantage of this model is that it identifies the internal characteristics and external quality characteristics of a software product. However, at the same time it has the disadvantage of not showing clearly how these aspects can be measured [6].

Other software quality models from literature are Dromey, FURPS and Systemic. The discussion and comparison of these quality models can be obtain in Yahaya et al [4].

III. ISSUES IN SOFTWARE CERTIFICATION

Software certification particularly the product quality approach is still a new concept in software industry in Malaysia. This idea is becoming increasingly popular in Europe and United States, but at the same time many debates on this issue are reported.

The term certification in general is the process of verifying a property value associated with something, and providing a certificate to be used as proof of validity. International Organisation for Standardization (ISO) defines certification as "a procedure by which a third party gives written assurance that a product, process or service conforms to specified characteristics"[6]. Certification of software can be viewed in three aspects: product, process and personnel. It is also known as the software quality certification triangle [9]. Survey by

1-4244-0220-4/06/$20.00©2006 IEEE

Aldrich, Goulde and Wong [10] indicates the importance of certification in term of cost reduction.

TABLE 1
IS0 9126 QUALITY CHARACTERISTICS [11]

Characteristic	Subcharacteristics
EFFICIENCY	
The capability of the software to provide the required performance, relative to the amount of resources used, under stated conditions.	• Time behaviour • Resource utilisation
FUNCTIONALITY	
The capability of the software to provide functions which meet stated and implied needs when the software is used under specified condition	• Suitability • Accuracy • Security • interoperability
MAINTAINABILITY	
The capability of the software to be modified.	• Analyzability • Changeability • Stability • Testability
RELIABILITY	
The capability of the software to maintain the level of performance of the system when used under specified conditions.	• Maturity • Fault tolerance • Recoverability
USABILITY	
The capability of the software to be understood, learned, used and liked by the user, when used under specified conditions.	• Understandability • Learnability • Operability • Attractiveness
PORTABILITY	
The capability of the software to be transferred from one environment to another.	• Adaptabality • Installability • Co-existence • Replaceability

Evaluation method and procedure are needed in order to apply the certification process. The procedure of evaluation may be applied in one of the following approach:

- First party evaluation, is related to internal product and process assessment
- Second party evaluation, is associated with acceptance evaluation on product before delivery
- Third party evaluation, is the independent evaluation by an independent body or a testing laboratory. [12]

Some issues in software quality have lead to the proposal of software certification by independent, third party assessment [9, 13, 14, 15, 16]. Lack of software quality and lack of publisher responsibility are the greatest concerns that the software industry is facing now and in the future. It is certification that can bring assurance back to generic software. Additional to that numerous certification methodologies are needed and each must be designed according to: (1) what the software does, and (2) what level of integrity is guaranteed by a particular "seal of approval" [3, 17]. According to Voas it is sensible to certify specific software applications according to their platforms, domains, environments, profiles, "ilities", etc. Then, software certificates and fact sheets can be published that are unambiguous and limited in scope.

Jeffrey Voas [14] discusses the importance of software certification in the software engineering industry. He claimed that in 1997 an estimated 25.5 percent of a typical corporation's software portfolio was commercial off-the-shelf software. Forecasts had that this figure is rising in 1998 to around 28.5 percent and to exceed 40 percent in the next four years. Although the time for the development of a system can be shortened if we use the COT components available, the possibility that the bad COT will be used in the application cannot be ignored. The problems may appear during systems maintenance. This is where the independent software certification comes in.

Two reasons that trust in software certification must come from someone other than the software publisher. The reasons are: 1) by hiring a third party to grant software certificates, publishers shift the responsible on liability concerning quality onto someone else, and 2) unbiased assessment from independent agency may benefit end users. The approach on certifying software through independent third party organization has a potential to change the manner in which software is evaluated, graded and sold. One way to conduct this approach is through involvement of end users in the process [16]. In this approach the independent certification body collects valuable information from user's environment and collects on how the product is used. The respective agency then produces the report based on significant operational experience created by the users.

Software Certification Laboratory (SCL) is the third party independent agency for certifying product proposed by Jeffrey Voas from Reliable Software Technologies [16]. The SCL uses the approach of end-user participation to provide information in the process of certifying a software product. In this approach the products must already have some number of valid and qualified users to participate.

IV. PRAGMATIC QUALITY MODEL

Our previous survey indicated that functionality, efficiency, integrity, maintainability and reliability were the main characteristics with high and very high consideration in assessing software products by respondents in Malaysia [4,18].

1-4244-0220-4/06/$20.00©2006 IEEE

Table 2 compares the result from the survey and the software quality characteristics according to ISO 9126 model. There are four characteristics resulted from the survey that are equivalent to the ISO 9126 characteristics. The characteristics are efficiency, reliability, functionality and maintainability. Integrity is not included in the ISO model but is considered as high consideration by participants in the survey. However portability and usability are not among the favorite high consideration characteristics but included in the ISO model. Even though usability is not considered as high consideration by the respondents in this survey, the mean score is high and almost achieving level high consideration.

TABLE 2
QUALITY CHARACTERISTICS: COMPARISON OF ISO9126 MODEL AND SURVEY

Quality Characteristics	ISO9126 Model	Survey
Efficiency	x	x
Reliability	x	x
Functionality	x	x
Maintainability	x	x
Portability	x	
Usability	x	
Integrity		x

The proposed pragmatic quality model considers the analysis above. The model is to be anticipated by combining and filtering these two sources as well as other quality models available from literature. The software quality framework shown in figure 1 is to be adopted in the development.

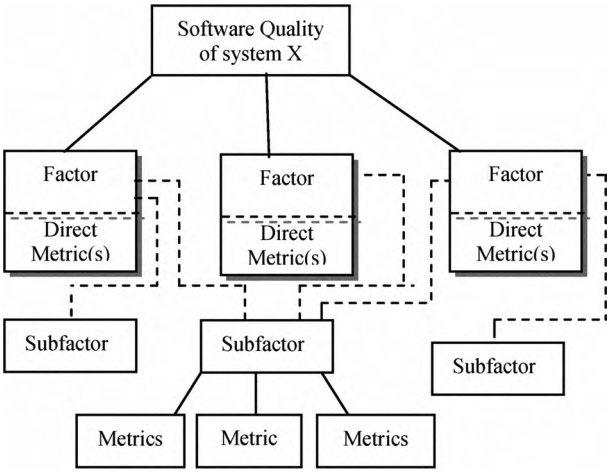

Fig. 1. Software quality metrics framework (IEEE 1992)

The first level of the software quality metrics framework begins with the identification of relevant quality attributes (or also called by some author as characteristics). At the second level of the hierarchy are the quality subfactors (or some authors use sub characteristics). At the third level of the hierarchy the subfactors are decomposed into metrics used to measured software products.

The proposed pragmatic model is different from other models as we identify factors, subfactors and metrics relevant to requirements for certification process. The metrics are considered relevant if they are measureable before the product being used by any users. Thus no additional information regarding that particular product is available as an input to the process. Therefore, there will be some quality metrics that irrelevant for this purpose.

Table 3 shows an example of the approach of pragmatic model. The factor of this example is functionality. Functionality is characterized into several subfactors. The subfactors are suitability, accuracy, interoperability, compliance and security. For this example and discussion we only include suitability. Subfactor *suitability* can be evaluated in two ways: - (1) Functional implementation coverage and (2) Functional specification stability [1]. In this approach the first metric for suitability is considered applicable because it can be measured while the second metric is not applicable because it cannot be measured during testing and assessing for certification (refer to Table 3) . Therefore, the second metrics will not be included in the pragmatic model.

V. A CONCEPTUAL MODEL FOR SOFTWARE CERTIFICATION PROCESS

The proposed certification model is designed based on the following foundation:

a. The independent certification body (ICB) may solve the problem raised by the SQA team in the survey regarding uncertainties of software quality by testing alone [4]. The independent certification is believed to be the only approach that user should trust and the demands for it are being heard from both publishers and users [8, 16, 19].
b. The quality model is based on ISO 9126 model and attributes from the survey.
c. The quality attributes may change from time to time as required.
d. ICB needs to assess a new software product even though no additional information available from users or developers.

The goal is to provide certification process that users can easily confidence with the quality and certification level the software hold. This may help the users in choosing the right product with the right requirements. The product can be the commercial Off-The-Shelf product or any completed development

1-4244-0220-4/06/$20.00©2006 IEEE

product. There are two situations in our proposed model. Firstly is the software publisher or software developer itself who wants to assess, evaluate and certify his product after development completion. This is beneficial to the publishers to promote their products. Secondly is the user who wants to know the status of any particular software available in the market.

The proposed certification process model is shown in Figure 2. The involvement of independent certification body (ICB) in the certification process and the issues of uncertainties in software quality by SQA team in testing are discussed in [4].

In this model a software publisher or user requests for a product certification to the ICB. Our approach is capable of analysing any new product available even when there isn't any user using the product yet. Once a request is received the process goes through the procedures of analyzing, implementing, assessing and reporting. This process requires specific quality model to be set as a benchmark to the system. A new pragmatic software quality model is to be anticipated by combining and filtering these sources. The certification level identifies the level of quality and thus giving the level of certification of that particular product. The tool for certifying is accessible by the independent certification body and the agency will grant the certification level to the publisher or requester.

VI. CONCLUSION

Demands for software certification based on product quality approach are being reported to overcome some problems regarding software qualities. Issues such as testing quality by SQA team, liability risks and uncertainties of software status are the reasons for implementing ICB approach. The ICB approach is not only beneficial to the user but also to the publisher as well. There are several ICBs or known software certification laboratories in existence today. For example, KeyLabs and SCL proposed by Voas. KeyLabs "certifies language purity for Java but makes no promise about software quality" [16]. The second SCL is conducted differently from our approach as it captures user information on the product.

We propose a first draft of the conceptual model for certifying software product. This model is useful for end user to obtain the status of a particular commercial Off-The-Shelf product as well as to the publisher to obtain certification on their product. The proposed conceptual model includes a pragmatic quality model and certification levels. The pragmatic quality model is based on the practicality of metrics for certification process. Our approach does support new product assessment when no user information is available. The conceptual model is to be enhanced in future works.

REFERENCES

[1] A. Yamada. "Information Technology – Software quality characteristics and metrics Part 2 – external Metrics," D*raft Technical Report ISO/IEC JTC1/SC7*, Nov 1996.

[2] IEEE. IEEE standard for a software quality Metrics Methodology. http://ieeexplore.ieee.org/xpl/standards.jsp. 1993.

[3] J. Voas. "Limited Software Warranties", Engineering of Computer Based Systems, (ECBS2000) *Proceedings. Seventh IEEE International Conference and Workshop*, 2000, pp 56-61.

[4] J.H. Yahaya, A. Deraman, and A.R. Hamdan, 2005. "The Development of A Conceptual Model For Software Product Certification Process", *Technical Report Series, Faculty of Technology and Information Science, UKM*, FTSM/Julai 2005/LT9.

[5] K. Khosravi and Y.G. Gueheneuc, "A Quality Model for Design Patterns", Technical report 1249, University de Montreal. Citing Internet sources URL http://www.yann_gael.gueheneuc.net/work/Tutoring/Documents/041021+Khosravi+Technical+Report.doc.pdf, 2004.

[6] Rae, A., P. Robert and H. Hausen, *Software Evaluation for Certification: Principle, Practice and Legal Liability*, Middlesex:McGraw-Hill International, 1995.

[7] F. E. Norman and S. L. Pfleeger, *Software Metrics: A Rigorous and Practical Approach*, Second Edition. PWS Publishing, Boston, 1997.

[8] M.F. Bertoa, J.M. Troya and A. Vallecillo, "A Survey on the Quality Information Provided by Software Component Vendors", *Proc. 7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2003)*, Darmstadt, Germany, July 21-25 2003, pp. 25-30.

[9] J. Voas, "The Software Quality Certification Triangle", Crosstalk, *The Journal of Defense Software Engineering*, November 1998:12-14.

[10] S.E. Aldrich, M. A. Goulde, and F. Wong, "Customer Want Windows 2000 Certified Applications: Certification Delivers Greater Reliability and Lower Cost", Citing Internet Source URL http://www.etestinglabs.com/downloads/certification/cert_benefits.pdf. 2000.

[11] Punter, T., Solingen, R. v., & Trienekens, J., 1997. Software Product Evaluation - Current status and future needs for customers and industry.Proceeding of 4th Conference on Evaluation of Information Technology.

[12] D. Welzel and H.L. Hausen. "Practical Concurrent Software Evaluation for Certification", *Journal Systems Software*; 38, 1997, pp. 71-83.

[13] J.Voas, "Developing a Usage-Based Software Certification Process", *IEEE Computer*, August 2000, pp32-37.

[14] J. Voas (a), "Certification: Reducing the hidden cost of Poor Quality", *IEEE Software Journal,* July/August 1999

1-4244-0220-4/06/$20.00©2006 IEEE

[15] J.H. Yahaya, A. Deraman, A.R. Hamdan, and A.S. Yahaya, "Software Product Certification Process: The Empirical Study and Conceptual Framework", *Proceedings of International Conference on Robotics, Vision, Information and Signal Processing, Penang, Malaysia*, 20-22 July 2005, pp. 809-813.

[16] J. Voas, "User participation-based Software Certification". *Proceedings of Eurovav 1999*, pages 267-276, June 1999.

[17] J.Voas (b), "Certifying Software for High-Assurance Environments". *IEEE Software Journal,* July/August 1999

[18] J.H. Yahaya, A. Deraman, A.R. Hamdan, and A.S. Yahaya, "Software Quality Metrics: The Empirical Study", *Proceedings of Brunei International Conference on Engineering and Technology 2005*, Brunei, 15-18 August 2005, pp.87-97.

[19] J. Voas. "Software Certification Laboratories?", Crosstalk, April 1998.

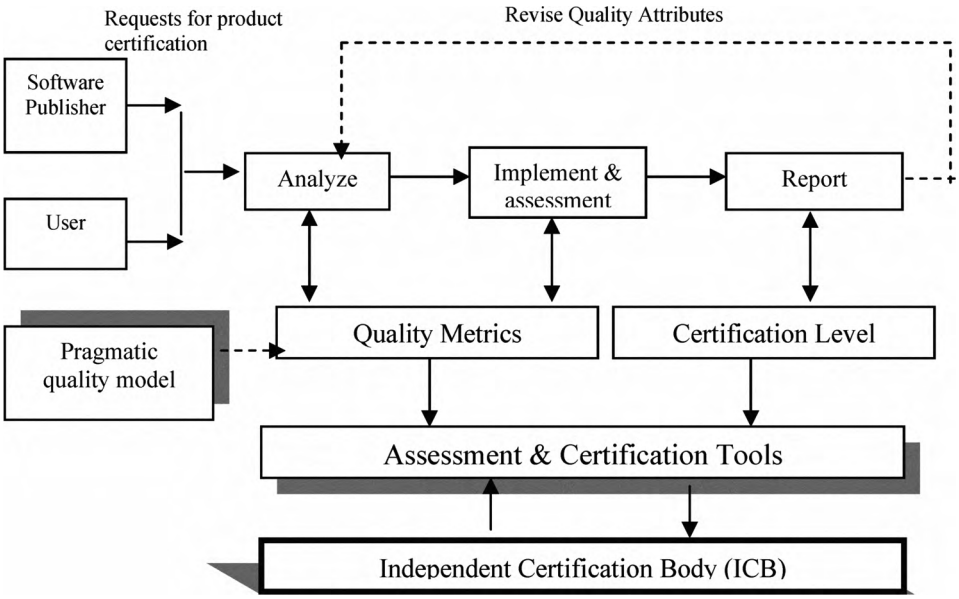

Fig. 2. A Conceptual model for software product certification process

TABLE 3
FUNCTIONALITY : METRICS AND MEASUREMENTS (EXAMPLE)

Sub factors	Indicator	Metrics	Status
Suitability	Functional implementation coverage	Function implementation ratio in testing X = (A/B) A = Number of implemented functions confirmed in executing testing B = Number of functions described in specification	applicable
	Functional specification stability	Functional specification change free ratio after entering operation X = 1 – (A/B) A= Number of functions obliged to be changed after entering operation during observed period of operation B = Number of specified functions (or any other size measures)	Not applicable