Journal of Sustainability Science and Management 2008 Volume 3 (2): 14-29

SOFTWARE CERTIFICATION MODEL BASED ON PRODUCT QUALITY APPROACH

JAMAIAH HAJI YAHAYA[1], AZIZ DERAMAN[2] AND ABDUL RAZAK HAMDAN[3]
[1,2,3] Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), Bangi, 43650 Selangor, MALAYSIA. [2] Graduate Department of Information Technology College of Arts and Sciences, Universiti Utara Malaysia, Sintok, 06010 Kedah, MALAYSIA. [3] Academic and Internationalisation Division, University Malaysia Terengganu, Kuala Terengganu, 21030 Terengganu, MALAYSIA

** Corresponding Author : e-mail: jamaiah@uum.edu.my*

Abstract: In this information age, most businesses are highly dependent on the availability of ICT services, especially in software application components. The interest in the acquisition of a high quality software has increased among various stakeholders. However, some pertaining problems are still being debated such as: (i) determining the quality of software product; (ii) defining mechanism for assessing software product quality; (iii) ensuring and offering software quality guarantee; and (iv) ensuring the continuous improvement of quality of software product. Users are concerned with the quality of software delivered and they expect the software to be in good quality and meet their prescribed standard. Therefore, a practical mechanism for assessment and certification is required to resolve these uncertainties. Findings from previous empirical study conducted in Malaysia agreed that with certification embedded and granted to the software might resolve the uncertainties and suspicion on the status and standing of software product. Based on survey and literature findings, main quality attributes have been identified which are crucial to the proposed certification framework. Two distinct models of certification are developed i.e. SCfM_prod and SPAC based on product and process quality approaches respectively. This paper focuses on software certification model based on product quality approach.

KEYWORDS: Software certification model, product quality approach.

Introduction

In the new global economy and borderless world, computer has become a central issue for business survival. Companies are competing to produce software which are claimed to be good and fulfil user's expectation and requirements. However, 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 products? How do we ensure and guarantee the quality of a particular software product? From our observation we believe users are concerned regarding the quality of software delivered and they expect software to be in good quality and meet their prescribed standard. Furthermore, stakeholders need to put their trust and confidence in the software which are being developed or purchased and used in the organizations.

Many complaints have been reported regarding this quality concern. Users complain that software quality has been degenerating steadily and worsening [Voas., 2000]. Software Engineering Institute's Capability Maturity Model (CMM) (cited in [Slaughter *et al.*, 1998]) reports the following quote from a software manager: "I'd rather have it wrong than have it late. We can fix it later". Therefore, users complaint that software is being delivered with bugs that need to be fixed and are dissatisfied with the product [Denning., 1992][Whittaker and Voas., 2002]. Consistent with our observations in many local organizations, we discover that software practitioners and software quality assurance (SQA) teams within the organizations could not guarantee the quality of the software being developed in-house or purchased from vendors. The SQA teams agree that quality of software is still uncertain if it is done through testing the software alone [Yahaya *et al.*, 2006]. The prevalence of this leads to general perceptions among clients that software industry in the country as a whole lack the standard and mechanism for monitoring or ensuring product quality.

Issues in Software Quality and Certification

General expressions of how quality is realized in software are with "fitness for use" and "conformance to requirements". "Fitness of use" refers to characteristics such as usability, maintainability, and reusability. "Conformance to requirements" means that software has value to the users [Tervonen., 1996]. 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" [Yamada, 1996]. IEEE defines software quality as – a software feature or characteristic used to assess the quality of a system or component [IEEE, 2008]. Of course, quality is in the eye of the beholder. Different people see quality differently. Therefore, "software quality is nothing more than a recipe. Some like it hot, sweet, salty or greasy" [Voas., 2004].

Several software product quality models are available and the more well known are McCall, Boehm, FURPS, ISO 9126, Dromey and Systemic. The main quality characteristics found in majority of the models are: efficiency, reliability, maintainability, portability, usability and functionality, presented in more recent models. These characteristics appear in all models and therefore, are considered as essential and vital. Quality in software domain is critical and requires continuous improvement to retain survival of the companies either in private or public sector. Software quality assurance affects both immediate profitability and long-term retention of customer goodwill. Thus it is important that quality of software to be embedded in the continuous improvement activity.

In addition to issues of quality characteristics in ISO 9126 model, ISO (cited in [Eagles., 2007]) mentions that views of users, developers and managers are different. The manager's view is quoted as:

> *"A manager may be more interested in the overall quality rather than in a specific quality characteristic, and for this reason will need to assign weights, reflecting business requirements, to the individual characteristics"* [Eagles., 2007].

This view is also agreed by Johnsson *et al.* (cited in [Svahnberg *et al.*, 2007]) and Boehm & In (1996) as both believe that different stakeholders tend to have different views of the importance or various quality requirements for a system.

Software certification is still a new concept in Malaysia but is increasingly popular in Europe and the United States. At the same time debates on this issue are reported. Previous survey conducted in Malaysia shows that even though it is a new idea and thought, it is an acceptable concept perceived by the respondents (see also [Yahaya *et al.*, 2005],[Yahaya *et al.*, 2006]).

The term certification by definition is "an official document that says something is of good quality" [Longman, 2003]. Mills *et al.* has mentioned certification of software in early 1980s. In the later year, International Organisation for Standardization defines certification as "a procedure by which a third party gives written assurance that a product, process or service conforms to specified characteristics" [Rae *et al.*]. Recent studies show that software certification can be viewed in three perspectives: product, process and personnel which is also known as software quality certification triangle [Voas., 1998]. The combination of these three will produce a best balance result [Bazzana *et al.*,],[Voas., 1999].

Issues in software quality have lead to the proposal of software certification by independent, third party assessment [Voas., 1999]. Previous survey indicated that involvement of an independent party in the evaluation of software may improve the quality of the assessment and thus guarantees and assures the quality of a product [Yahaya *et al.*, 2006]. Lack of software quality and lack of publisher responsibility are the greatest concerns that the software industry is facing. We believe that certification can bring assurance back to common software. The trust in software certification must come from someone other than the owner of the product. Due to the following reasons: 1) by hiring a third party to grant software certificates, publishers shift the responsible on liability concerning quality onto someone else, 2) unbiased assessment from independent agency may benefit end users, and 3) the cost of performing certification can be reduced if organisation that specialise in software assessment is employed.

One possible approach in implementing certification is through involvement of end user in the process by delivering information regarding the usage of the software [Voas., 1999], developers self-certification [Morris *et al.* 2001] and verification and validation technique [Heck and Eekelen., 2008]. Each of these methods has its advantages and drawbacks. Certification is a possible approach to accomplish a continuous improvement and continuous assured quality of a software product. It is an alternative route to attain quality continuously in its life span [Deraman *et al.*, 2007].

Several attempts have been made to develop certification model but the models created by these studies were based on different perspectives and objectives. Some of these models emphasis on combining product and process quality in the certification process ([Heck & Eekelen., 2008], [Voas., 1999], [S'antana & Guerra, 2003], [PROFES, 1997]). The model proposed in this research offers flexibility in obtaining certification level with a guided procedure of initiating weight factors of quality attributes to meet organisation's business requirements. The proposed model offers practicality and independent from development process model, and meet other criteria mentioned above. In view of the analysed software quality and certification problem and issues, a software certification model is developed that embedded these related requirements.

- A practical model of software quality for software certification process.
- A mechanism for certifying software product based on product priorities and specifications.
- An approach for certifying software product based on collaborative-perspective approach.

Software Certification Framework

Software certification framework (SCF) developed in this research composed of two distinct approaches of certification: by means of development process and end product quality approach. SCF consists of two distinct and separate models for certifying software products.

1. The first model focuses on developing certifying process that concentrates on end product quality approach. The main component in this model is the quality criteria in certification of software. The criteria may be collected through literature study and also may involve industry to contribute the quality criteria in certification.
2. The second model focuses on certifying software via development process approach. The main component in this model is the development processes and other entities that contribute directly and indirectly to the development process.

Figure 1 demonstrates the framework of software certification environment that shows a combined or joint certification criterion to produce a comprehensive certification of candidate software product. The certification level of software product can be obtained through these two distinct approaches of certification viz process and product approach. This paper focuses mainly on the development of software certification model by product quality approach. Another approach for software certification developed in this research was constructed and named as SPAC model. This model consists of several components: development technology, project condition, support activities, management activities, development activities, environment and people [Fauziah *et al.*, 2007]. The detail of this model is not explained in this paper.

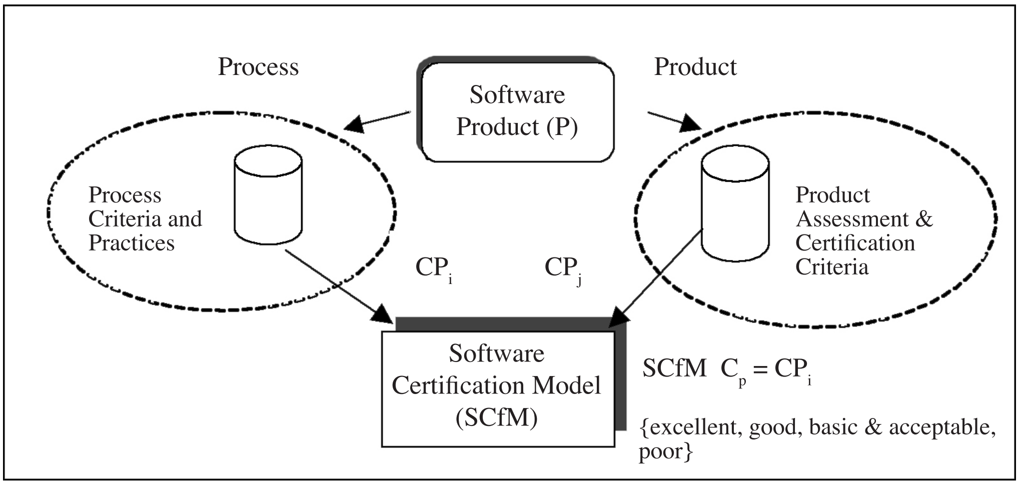
Process
Product
Software Product (P)
Process Criteria and Practices
Product Assessment & Certification Criteria
CP_i
CP_j
Software Certification Model (SCfM)
SCfM $C_p = CP_i$
{excellent, good, basic & acceptable, poor}

Figure 1: Software Certification Framework (SCF)

The approach of conducting certification in this research is through a collaborative perspective approach which involves users, developers and independent assessor. This approach is applied in both certification models defined in the framework. The advantages of this approach compares to other approaches are:- 1) elimination of bias assessment of the product by including independent assessor in the team, 2) removal of unfairness assessment by including the owner or users of the product to participate in the certification process and 3) accelerating the process because the team is familiar with the product and its' environment, and 4) protection of data confidentiality and privacy by only permitting direct users to have access to the software.

SCfM_*Prod*: Software Product Certification Model

The software product certification model developed in this research is named as SCfM_*Prod* (refer to Figure 2). It includes Pragmatic Quality Factor (PQF) as the quality certification guidelines and standard, product criteria, certification specification, certification representation method, repository and certification team.

The first component of this model is the Pragmatic Quality Factor (PQF) which is the quality certification guideline and standard for measuring software product quality. Undertaking quality attributes defined in ISO 9126 model as the based line of the assessment metrics, we define two sets of attributes, by means of the behavioural and the impact attributes. The behavioural attributes consist of high level software quality characteristics which include usability, efficiency, functionality, maintainability, portability, integrity and reliability. Integrity is not included in ISO 9126 model but included in this model because of the requirement from literature and empirical study. In the age of hackers and firewalls, the importance of integrity aspect has increased [Pressman, 2001] and ISO 9126 model is a generic model but requires some customization for particular case [Bertoa *et al.*, 2006]. Integrity measures the ability to withstand attack on its security that comprises of program, data and document. It covers threat and security aspects. Findings from previous empirical study also indicated the importance of integrity in software quality metrics [Yahaya *et al.*, 2006]. Each attributes in PQF is made up of several subattributes and then broken down into several metrics that shows the measurement aspects of the attributes.

Survey done by this research group also indicated that quality attributes can be classified into different levels and weights (see also [Yahaya *et al.*, 2006], [Yahaya *et al.*, 2007]) based on their importance and significance during quality assessment by the respondents. Thus, adopting functional point approach, attributes are classified into three distinct classification layers: high, moderate and low. The attributes are grouped and weight factors are assigned (Table 1).

Table 1: Classification of attributes and its weight factor

Level	Attributes	Weight Factors
Low	Flexibility Intraoperability Interoperability Portability Survivability	1-4
Medium	Safety Efficiency Maintainability Usability	5-7
High	Functionality Reliability Integrity	8-10

The second measurement in PQF is the impact attribute. This attribute indicates the conformance in user requirements, expectation and perception. These attributes include measure of popularity, performance, trustworthiness, satisfaction and user acceptance. These attributes align with the definition of quality. Quality must conform and correspond to requirements and fitness of use. These two groups of attributes are important to balance the assessment between the technical aspects of quality and the human factors [Dekkers & McQuaid, 2002]. Similar to behavioural attributes, the impact attributes are made up of several subattributes and metrics that show the measurement of the attributes.

The product criteria component in this model offers services for weight factors as mentioned above and criteria selection. In this model users may select their interested attributes of quality to meet their organizational requirement and target. This offers flexibility in the certification exercise. The third component is the certification team. As discussed in previous section, the certification process is done collaboratively with three different assessors: the independent assessor, developer and user.

The fourth component is certification specification. This component explains the processes, algorithm, formulas and reporting format in the certification exercise. The algorithm and formulas included in this model will be explained below.

The fifth component is the certification representation method. This component offers certification-mapping process to obtain the associated certification level of software product (see Table 3). It is important to note that the ranking of certification level mentioned here is flexible and does not fixed to the stated figures. They are opened for customisation and tailored to requirement by the organisation. The organisation and the owner of the products may decide to modify and customise the classification levels based on their maturity and the readiness of the organisation itself.

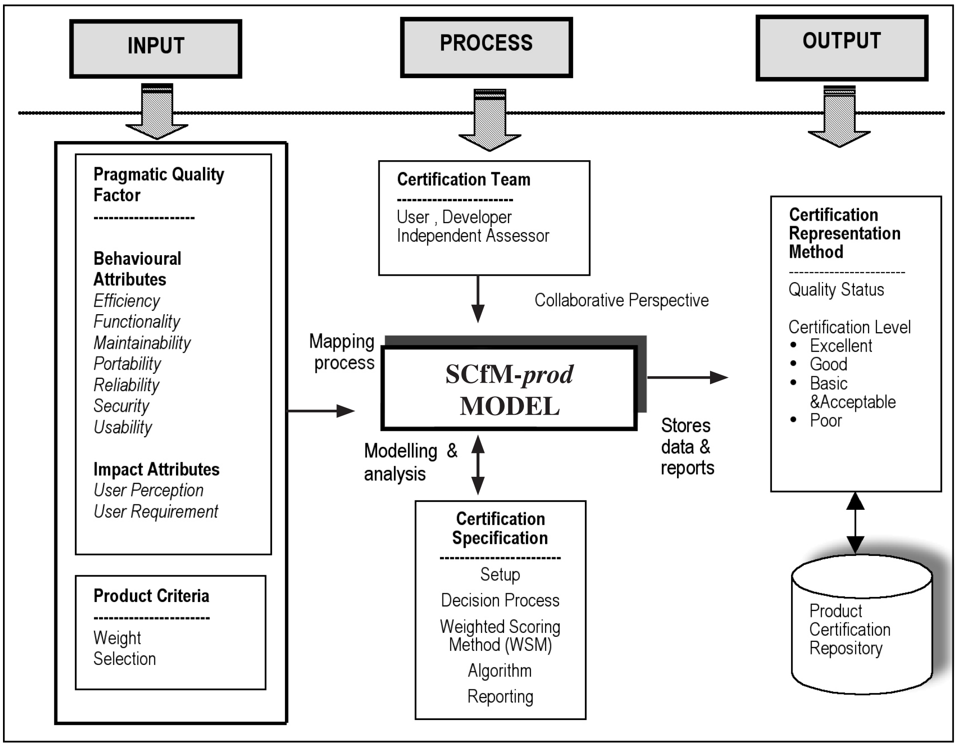

Figure 2: Software Product Certification Model (SCfM_*Prod*)

Certification Specification and Algorithm

This section describes the specification and algorithm associated with certification process as mentioned in Figure 2. The specification includes setting and initialisation, decision process, weighted scoring method (WSM), algorithm and reporting. The steps in certification process can be summarised as follows:-

1. Certification procedure setting and initialisation. In this step, background information on the organisation and product to be assessed are collected. The requirement such as criteria selection and weight will be recorded.
2. Collection and gathering quality data. This step includes collection, gathering, inputting and validating data of quality of software product.
3. Decision-making processes may apply in the certification process. The detail of the processes is explained in [Yahaya *et al.*, 2006].

4. Execution of certification by individual attributes.
5. Execution of certification by product package.
6. Generation of reports.

Execution of certification by individual attribute

Software quality model defined in this model comprises of attributes, sub attributes, and metrics. The architecture of the quality factor is tailored with the software quality metrics framework by IEEE (2008). Quality attribute of software is broken down into subattributes, and further decomposed into metrics. The metrics are the direct metrics that are used to estimate quality of software attributes. The quality score is calculated using the following algorithms. Table 2 represents metrics in individual attribute. M_1, M_2, M_3 and etc represent metrics in specific attributes, A_1, A_2, and etc represent assessor in this model which either user, developer or independent assessor. P_{12}, P_{21}, P_{n1} are the perspective value given by the assessor for each of the metrics.

Table 2: Table of metrics of attribute

Metrics / ***Assessor***	**M_1**	**M_2**	**M_3**	**M_4**	**M_5**	**…**	**M_t**
S_1	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	…	P_{1t}
S_2	P_{21}	P_{22}	P_{23}	P_{24}	P_{25}	…	P_{2t}
•							
•							
•							
S_n	P_{n1}	P_{n2}	P_{n3}	P_{n4}	P_{n5}	…	P_{nt}
Average (T)							

The average score for each of the metric is calculated as follows: -

$$T_k = \left(\sum_{j=1}^{n} p_{ij} \right) / n \quad , \ k=1,2.....t \tag{1}$$

where n represents number of assessor and t represents number of metrics.

The average perspective score (aps) of attribute *a*, is then calculated as follows:-

$$aps_a = \left(\sum_{k=1}^{t} T_k \right) / t \quad , \ k=1,2.....t \tag{2}$$

where t is the number of metrics measured in the attributes.

Each attribute calculated using formula 2 can be used to measure its certification level by:

$$QS_a = (aps_a / 5) * 100 \tag{3}$$

where a, represent specific attribute. The constant 5 represents the maximum possible value of quality score. The QS score is mapped to a certification representation model to obtain its associate level.

Execution of certification by product package

The Weighted Scoring Method or WSM is typically applied in the following fashion: attributes and subattributes (SA) are defined and for each subattributes, several metrics are used to measure attributes. Each attribute holds an average perception scale (aps) given by the assessment team and a weight factor. The weight factors are assigned by the owner of the product based on its organization requirements and expectations and guided in this model. The quality score (QS) and percentage quality score (QSP) are formulated as follows: -

- Calculate total number of weight values given by the owner of the product.

$$TW = (\sum_{a=1}^{n} w_a) \tag{4}$$

where a = number of attributes defined.

- Calculate quality score for each of attributes

$$QS_a = (w_a / TW) * aps_a \quad , a=1,2\ldots..n \tag{5}$$

where a represents each attribute, w = weight, TW = total weight and *aps* represent average perception score.

- Calculate percentage of quality score.

$$TQS_a = (QS_a / 5) * 100 \ , a=1,2\ldots..n \tag{6}$$

where a represents each of the attribute and constant value 5 is the maximum possible score of *aps*.

- Calculate the total quality score of entire software as a product.

$$TQP \ = (\sum_{a=1}^{n} TQS_a) \ , \ a=1,2.....n \tag{7}$$

where a represent attribute and n is the number of attributes.

TQP score is then mapped into a certification representation model to obtain its associate certification level.

The Certification Level

The certification levels are identified and characterised in four distinct levels: excellent, good, basic and acceptable, and poor. The certification level of a product is determined by comparing the score value obtained in equation 7. For TQP value greater than 90% and less than 100%, the product obtains a certification level of *excellent.* This means that the software product satisfies all quality criteria and achieves quality level of excellent and satisfactory. If the TQP score is greater than or equal to 75% and less than 90, the product is classified as *good* which means that it satisfies the quality level of good. If the product gains a TQP score greater and equal to 50 and less than 75, the product is identified as *basic and acceptable* which means that the software satisfies the quality level of basic or average and acceptable. However, if the TQP score is less than 50, the product is graded as poor and unsatisfactory. The classification level is shown in Table 3. The similar classification technique is used in [Ortega *et al.*, 2003].

Table 3: Ranking of certification levels

TQP Score (TQP)	**Certification Level**	**Certification Status**	**Description**
90<= TQP <=100	4	Excellent	Software satisfies all quality criteria and achieves quality level of excellent.
75<= TQP < 90	3	Good	Software satisfies and achieves the quality level of good.
50< = TQP < 75	2	Basic and Acceptable	Software satisfies and achieves the quality level of basic which also means average and acceptable.
0 <= TQP < 50	1	Poor	Software attains quality level of poor and unsatisfactory.

It is important to note that the ranking of certification level mentioned above is flexible and is not fixed to the stated figures. They are opened for customisation and tailored to requirement of the organisation. The organisation and the owner of the products may decide to modify and customise the classification levels based on their maturity and the readiness of the organisation itself.

Testing and Evaluation of the Model

The SCfM_*Prod* model was evaluated in case studies that were launched collaboratively with three large organizations in Malaysia. Results from the study shows that the evaluated aspects in this model are feasible and demonstrates the potentials and practicality of the model in supporting certification methodology. SCfM_*Prod* model facilitates a systematic and repeatable software assessment and certification during its life span. Furthermore, the relationship between attributes, sub-attributes and metrics that measures the quality aspects of software are valuable and provides awareness of improvement in software product quality in the future.

Three software operated in three different environments have been identified and the exercise was implemented in three phases: phase 1- prior to assessment process, phase 2 – during assessment and phase 3 – post assessment. The following results show the testing done of product X, Y and Z using SCfM_*prod* model. Table 4 demonstrates the detail results of each attributes defined in this model for product X, Y and Z. Kiviat graphs in Figure 3 illustrate the results showing the scores obtained by the behavioural attributes in SCfM_*prod* model. The graphs illustrate the weakness and strength of the product based on behavioural attributes. Each attribute is represented by axis and scores are plotted at the limits between 0-100%. Kiviat graph can be used to easily identify attributes that need attention in this process. Attribute that fall on the limit's outer layer is considered of better quality compared to attributes at inner layers of this graph.

Table 4: Comparison of quality score obtained by Case X, Y and Z

	Quality Attribute	**Case X Score/5.00**	**Case Y Score/5.00**	**Case Z Score/5.00**
1	Efficiency	3.73 *(74.6%)*	4.08 *(81.6%)*	4.70 (94.0%)
	Time behaviour	*3.56*	4.33	4.50
	Resource utilization	*4.00*	3.70	5.00
2	Functionality	3.62 *(72.4%)*	3.69 *(73.8%)*	4.96 (99.3%)
	Suitability	*3.83*	3.65	4.88
	Accuracy	*3.33*	3.20	5.00
	Interoperability	*3.63*	4.50	5.00
3	Maintainability	3.34 *(67.8%)*	2.66 *(53.2%)*	3.58 (71.6%)
	Analysability	*3.61*	2.63	3.05
	Changeability	*3.13*	2.20	3.25
	Testability	*2.83*	3.06	2.00
4	Portability	3.20 *(64.0%)*	3.55 *(71.0%)*	3.50 (70.0%)
	Adaptability	*3.56*	5.00	4.75
	Installability	*2.77*	1.80	2.60
	Conformance	*4.00*	4.80	5.00
	Replacebility	*3.33*	4.40	5.00
5	Reliability	3.30 *(66.0%)*	3.36 *(67.2%)*	4.50 (90.0%)
	Maturity	*3.83*	3.80	4.75
	Fault Tolerance	*3.00*	3.20	4.38
	Recoverability	*3.00*	3.00	4.33
6	Integrity	3.67 *(73.4%)*	3.83 *(76.6%)*	4.33 (86.7%)
	Security	*4.00*	3.87	4.33
	Data Protection	*3.33*	3.06	3.00
7	Usability	3.20 *(64.0%)*	2.95 *(59.0%)*	3.41 (68.2%)
	Understandability	*2.56*	3.44	2.72
	Learnability	*2.76*	2.93	3.40
	Operability	*3.70*	3.01	4.61
8	User Conformity	3.53 *(70.6%)*	3.67 *(73.4%)*	4.73 (94.7%)
	User Perception	*3.56*	3.84	4.67
	User Requirement	*3.50*	3.40	4.83

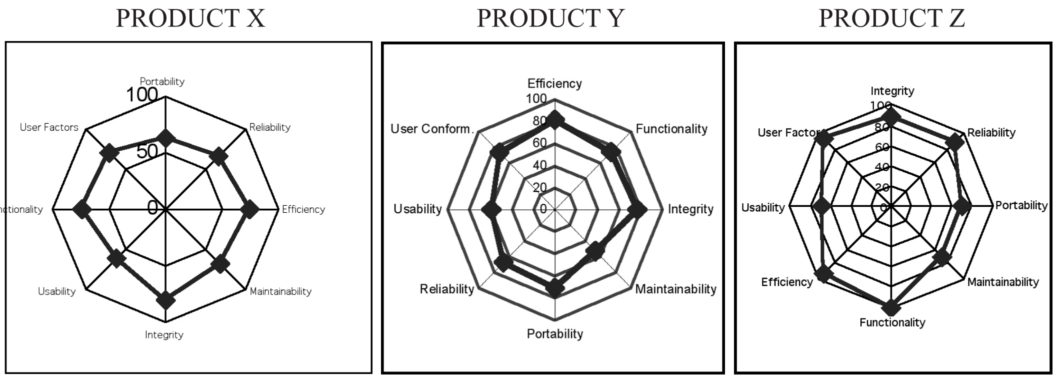

Figure 3: Kiviat graphs show the scores of attributes

In Case X, efficiency, functionality and integrity fall in better quality level compare to maintainability, usability and portability. In Case Y, efficiency, functionality, integrity, portability and user conformity fall in better quality level compared to maintainability, reliability and usability. In Case Z, functionality, efficiency, reliability, portability and the impact attribute, user conformity fall in better quality level compared to maintainability, portability, and usability.

Further analysis is conducted where the certification process involves producing the quality score of software product using SCfM_*prod* model. In this analysis weighting concept is used and applied. It is the organisation's responsibility to identify the appropriate weight for each of behavioural attributes. This is important to reflect their business requirements and constraints in software deployment. Table 5 shows the analysis and result for product Z as one product. In this analysis, an organization is required to assign values of associated attributes that could reflect business requirements and expectations.

Table 5: Assessment analysis of Product Z

Behavioural Attributes	**Max Value (1)**	**Weight (2)**	**Score Score (3)**	**Obtained (4)**	**Quality Score (%) (5)**
Efficiency	5	7	4.7	0.609	12.1
Functionality	5	10	4.96	0.919	18.4
Maintainability	5	7	3.58	0.464	9.3
Portability	5	3	3.5	0.194	3.9
Reliability	5	10	4.5	0.833	16.7
Usability	5	7	3.41	0.442	8.8
Integrity	5	10	4.33	0.817	16.3
TOTAL		54		4.279	85.6
The Impact					
User Conformity					94.7
Total Product (TQP)					**90.1**

As shown in Table 5, 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 and column 3 is the average score obtained by this assessment. Based on the weight assigned, scores are calculated as shown in column 4. Final values (column 5) are the computed values of quality score obtained according to attributes. All the computations involved here are formulated using formula (4) – (7). In this case, the score for behavioural attributes is 85.6%, the score for user conformity which deal with human aspects is 94.7%. Therefore, the total quality score of this product is 90.1%. The score is then mapped into a certification level and software product Z obtains a certification level of 4, which is equivalent to excellent. The certification levels are represented in Table 3.

Discussion

This research acts toward improving software certification process particularly in outlining an approach to certify software product and determine the quality status of specific candidate product operational in certain environment. Moreover, the determined requirements are presented in a practical quality model with the aim to support the certification process. Features and capabilities of the framework are the following:

- Provides an alternative means to certify software product in a collaborative perspective approach. This approach provides confidentiality, security and privacy of the software. It accelerates the process and eliminates biasness of assessment and certification.
- Provides the means to identify the quality status of a product using a pragmatic quality model in a practical environment.
- Provides the means to offer flexibility in obtaining certification level with a guided procedure of initialising weight factors of quality attributes to reflect the organisation's business requirements.

In addition, the model developed in this research is based on certifying software, which has been completed, installed and operated in certain environment. This approach also consistent with Voas and Miller (2006) who believe that certification must cover both software and the environment [Voas & Miller., 2006]. This is to ensure security, reliability and functionality of the system.

The first discussion deals with comparison of results of software products (X, Y and Z). The summary of all the results is shown in Table 5. Product X of Case X was six months old when certification exercise was conducted. Product X was developed through out-sourcing and jointly with another software house. The result shows that product X achieves level 2 of certification with a score of 70.08/100, which is equivalent to basic and is acceptable.

Product Y of Case Y was only one month old during the assessment period. It was developed in house by internal IT professionals with collaboration and support by domain expert users from various departments in the organization. The result shows that product Y achieves level 2 of certification with score of 71.3/100, which equivalent to basic and is acceptable.

Product Z of Case Z has been operated for more than ten years in the environment. Product Z was developed by in-house professionals and experts within the organisation. The result shows that product Z achieves level 4 of certification with a score of 90.1/100, which is equivalent to excellent.

Table 6: Comparisons of Cases X, Y and Z

Criteria	CASE X	CASE Y	CASE Z
Sector	Business	Health	Education/Higher Learning Institution
Software	Human Resource System	Hospital Information System	Staff Information System
Development Approach	Out-Source & Joint Development	In-house	In-house
Duration of Use	6 months	1 month	> 10 years
RESULTS OF ASSESSMENT AND CERTIFICATION			
Quality Score	70.08/100	71.3/100	90.1/100
Certification Level	2	2	4
Certification Status	Basic and Acceptable	Basic and Acceptable	Excellent

At least two factors influence quality and certification level of a software product. The two factors are the operation period in the environment of the software or the maturity of the software and second, the weight factors of attributes assigned by the owner of the product. The studies show that longer operating period of the software gives better result of quality and certification level. This is true because the software has been updated and corrected accordingly and necessarily by the developers. The second aspect that influences the result is the weight factors of the product. Without assigning weights factors to quality attributes, the results may indicate different level of certification. Thus, this model accommodates weight factors for all attributes with different level of importance to reflect individual business requirements [Eagles., 2007]. It is important that the weight factors are identified and assigned accordingly by the owner of the product to reflect the actual quality status of the software based on the organizational requirement. In this circumstance, the maturity of the person who assign the weight in software development and management is an important criteria in validating the certification result.

In order to explore the individual quality attributes for three case studies, the results are tabulated in the summary table as displayed in Table 4. The analysis shows that in these three separate cases, developers are less concerned and concentrated on the aspects of maintainability, portability and usability. These behavioural attributes obtain the lowest scores among the other attributes and are common for these three products.

Research Experience

Several lessons have been learned from the case studies as they were implemented and applied. In general, a few aspects of the model requirements were found to be either inadequate or not feasible. The significant aspect include:

- The issues of data confidentiality and privacy are essential to consider. The owner of the product usually does not permit independent assessor to access the system and data alone. The direct and valid user must do the testing and this is a common phenomenon in any organizations.

- Secondly, some of the metrics defined in the model are not answerable by the users alone thus requiring involvement from developers as well. Therefore, a collaborative perspective approach was introduced in this model.
- The studies observe that in evaluating a software, the assessment does not deal with just extending and refining metrics but also faced with additional challenges in assigning weight for each attributes. This requirement is essential as not all attributes are of equal importance in real situations.
- In handling massive data on quality and involving rules and decisions, a support tool is required to assist in these tasks. Measuring and computing scores on quality for all attributes and sub-attributes using semi-automated system reveal errors and mistakes.
- The studies reveal that the overall model and mechanism of software product certification proposed in this research is feasible and practical to be implemented in the real environment. The model is valid and of integrity through the evaluation by the case studies.

Future Work

This research has demonstrated the practicality of software certification in software industry. This study has noted the inclusion techniques in software certification process by end-product approach. Further works of certification process are therefore recommended. It is necessary to develop a comprehensive model and support tool with intelligent aspects included in the software. The intelligent tool requires a self-learning capability with capturing knowledge from certification processes and experiences. Criteria of software assessment and certification might change and require inclusion of new criteria. Thus the intelligent toolset is capable of detecting the changes and therefore allows recommendations of new modification assessment criteria. Currently we are elaborating and enhancing this model to develop a toolset named **SoCfeS** (Software Certifier System) to support the certification process and this additional requirements.

Conclusion

SCfM_*Prod* model provides the algorithms to measure software quality and software certification level based on identified standard. The first algorithm is to measure the quality status of each attribute based on the average score in an assessment exercise. Second algorithm is to measure the certification level of the software product by applying weights for each behavioural attribute. The results from both algorithms are mapped into a certification representation model to determine the certification level (1, 2, 3, or 4) and its representation either of excellent, good, basic and acceptable or poor. The SCfM_*Prod* model provides procedures and guidelines for certifying software product operational in certain environment. It adopts the collaborative perspective approach in software assessment and thus gives fairer evaluation of the products.

This research shows that certification of software products is feasible and demonstrates the practicality of the implementation in real situation. We continue to extend our model to construct a comprehensive and intelligent model (with intelligent expert system embedded). The expert system is expected to have self-learning capability towards knowledge available in the environment. The system named as SoCfeS is the continuation of SCfM_*Prod* model, a model that has been developed, applied and tested in real industry case studies.

References

Bazzana, G., Andersen, O. & Jokela, T. "ISO 9126and ISO 9000: Friends or Foes?" IEEE Sofware Engineering Standards Symposium.

Boehm, B. & In, H. (1996). "Identifying quality-requirement conflicts" IEEE Software, March 25-35.

Bertoa, M. F., Troya, J. M. & Vallecillo, A. (2006). Measuring the usability of software components. *The Journal of System and Software,* (79), 427-439.

Denning, P.J. (1992). "What is Software Quality?" A Commentary from Communications of ACM, January.

Deraman, A., Yahaya, J.H., Baharom, F., Fadzlah, A.F.A. & Hamdan A.R. (2007). "Continuous quality improvement in software certification environment" Proceedings of the International Conference on Electrical Engineering and Informatics (ICEEI 2007), 11-17.

Dekkers, C.A. & McQuaid, P.A. (2002). The dangers of using software metrics to (Mis) Manage. *IT Pro* (March/April): 24-30.

Eagles. Evaluation of natural language processing systems, Final Report (Sept). http://www.issco.unige.ch/ewg95/ 1995. [18 May 2007].

Fauziah Baharom, Aziz Deraman, Abdul Razak Hamdan. (2007). "Introducing Sofwtare Process Assessment and Certification (SPAC) Model" Proceeding of the 3rd Malaysian Software Engineering Conference (MySec'07), 59-63.

Heck, P & Eekelen, M. v. "LaQuSo Software Product Certification Model (LSPCM)" *http://alexandria.tue.nl/repository/books/633706.pdf* 2008. [16 July 2008].

IEEE (1993). " IEEE standard for a software quality Metrics Methodology", *http://ieeexplore.ieee.org/xpl/standards.jsp.* [23 July 2008].

Jeffrey Voas and Keith Miller, (2006). "Software Certification Services: Encouraging Trust and Reasonable Expectations", IT Professional, vol 8, no. 5, pp. 39-44, Sept/Oct.

Longman. (2003). Dictionary of Contemporary English. Pearson Education Limited, Essex, England.

Mills, H. D., Dyer, M., Linger, R.C. "Cleanroom Software Engineering", IEEE Software, September, 19-24, 1987.

Morris, J., Lee, G., Parker, K., Bundell, G.A., & Lam, C.P. (2001). "Software Component Certification", IEEE Computer, September, 30-36,.

Ortega, M., Perez, M., & Rojas, T. (2003). Construction of a systemic quality model for evaluating a software product. *Software Quality Journal,* (11), 219-242.

Pressman, R. (2001). *Software Engineering A Practitioner's Approach.* McGraw-Hill International Edition.

PROFES 1997. PROFES www-site. http://www.ele.vtt.fi/profes/ [18 May 2007].

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

Slaughter, S.A., Harter, D.E. & Krishnan, M. S. (1998). "Evaluating the Cost of Software Quality", Communications of The ACM, 41(8), 67-73.

Svahnberg, M., Wohlin, C., Lundberg, L. & Mattsson, M. (2002). "A method for understanding quality attributes in software architecture structures" ACM *http://portal.acm.org/citation.cfm?id=568900,*. [22 May 2007].

S'antana, M. L & Guerra, A. C. (2002). Quality of software process or quality of software product. http://citeseer.nj.nec.com/ [29 July 2003].

Tervonen, I. (1996). "Support for Quality-Based Design and Inspection", IEEE Software, January, 44-54.

Voas, J. "Limited Software Warranties, Engineering of Computer Based Systems", (ECBS2000) Proc. of Seventh IEEE International Conference and Workshop, 56-61, 2000.

Voas, J. (1998). "The Software Quality Certification Triangle", Crosstalk, The Journal of Defense Software Engineering, November, 12-14.

Voas, J. (1999). "User participation-based software certification", Proc. of Eurovav, June, 267-276, 1999.

Voas, J. (1999). "Certification: Reducing the hidden cost of poor quality", IEEE Software, July/August, 22-25.

Voas, J. (2004). "Software's secret sauce: The "-ilities"", IEEE Computer, November/December, 14-15.

Whittaker, J. A., Voas, J. M. (2002). "50 years of software: Key principles for Quality", IEEE IT Pro, Nov/Dec, 28-35.

Yahaya, J.H., Deraman, A., Hamdan, A.R. (2006). "Decision Framework for Software Product Certification.

Process" Proc. of The Second Malaysian Software Engineering Conference (MySEC'06), Kuala Lumpur.

Yahaya, J.H., Baharom, F., Deraman, A., Hamdan, A.R. (2005). "A Conceptual Framework for Software Certification", KUTPM Journal of Technology & Management, 3, Num 2.

Yahaya, J.H., Deraman, A., Hamdan, A.R. (2006). "Software Quality and Certification: Perception and Practices in Malaysia" Journal of ICT (JICT), 5, Dec.

Yahaya, J.H., Deraman, A. & Hamdan, A.R. (2007). "Software Product Certification Model: Classification of quality attributes" The First Regional Conference of Computational Science and Technology (RCCST07), Kota Kinabalu, 436-440.

Yamada, A. (1996). " Information Technology – Software quality characteristics and metrics Part 2 – external Metrics", Draft Technical Report ISO/IEC JTC1/SC7.