User Centric Software Quality Model For Sustainability: A Review

Nur Zuria Haryani Zakaria, Abdul Razak Hamdan, Jamaiah Yahaya, and Aziz Deraman

***Abstract*—Software quality is a tacit and multifaceted concept of desired combination of software attributes. Quality is commonly considered as product property, thus the product view of quality seeks to identify those properties which can be built into a product and assessed to certify quality. The goal of assessment is to satisfy the stakeholders of the software. However there is no single correct quality model may be accepted by researchers and experts in modelling and measuring the software quality. The goal of quality assessment should move towards sustainable software quality. This paper presents part of work done in assessing software quality for sustainability. Among the objectives is making concerns of user's perspective is crucial among the principles of assessment. Software quality metrics can be employed to asses and quantify the software quality value and helps in reducing the ambiguity of the view towards sustainable software.**

***Index Terms*—Software quality, software quality model, software quality assessment, sustainability, quality factors, quality metrics.**

I. Introduction

Nowadays, the software role and position continuously demanded in many information systems. Thus, any software defects may lead to serious damage and even physical harm [1]-[3]. Software developers also compete to produce software products quicker and in simple approach. However software project is considered failed if it is over schedule, over budget, suffer to achieve the business objective and does not meet user requirements [1], [4]. From the social and economic aspects, customers or users will lose their confidence. In term of economy; maintenance cost will multiply if the project fails. The software applications are more transparent and much closer to the users due to the current development and technologies. Software development cycle becomes shorter which require dynamic user commitment. The scenarios reveal that users are more analytical towards diverse functional and non-functional features of the software [5]-[8]. This confirm that producing a quality software is very important in order to sustain the software and able to last longer over a period of time. Sustainability in this context refers to environmental, social, and economic aspects of software development and the usage of software system [9]-[12].

Manuscript received December 5, 2015; revised February 23, 2016. This work was supported in part by the UKM Grant AP2013-007 and Kolej Poly-Tech MARA.

Nur Zuria Haryani Zakaria is with Kolej Poly-Tech MARA, Bandar Baru Bangi, 43650, Malaysia (e-mail: nzharyani@yahoo.com).

Abdul Razak Hamdanr and Jamaiah Yahaya are with National University Of Malaysia (UKM), Bandar Baru Bangi, 43650, Malaysia (e-mail: arh@ukm.edu.my, jhy@ukm.edu.my).

Aziz Deraman is with University Malaysia Terengganu, 21030 Kuala Terengganu, Malaysia (e-mail: a.d@umt.edu.my).

II. Software Quality

Software quality is the degree to which software possesses a desired combination of attributes. This desired combination of attributes must be well defined; otherwise the quality assessment is left to intuition [13]. For that matter, defining software quality is equivalent to defining a list of software quality attributes required for one system. However quality is tacit and is not easy to be measured [14]. Quality is commonly considered as a property of a product, thus the product view of quality seeks to identify those attributes which can be designed into a product or to be evaluated to ensure quality [15], [16]. Ref [5] reframe "What is software quality?" to "How do we satisfy the customers of our software?". It is based on reasoning that by making matters of customer satisfaction is central among the criteria for assessing software, actions will materialize towards making reliable and trustworthy software. Quality comprises all characteristics and substantial features of a product or an activity related to the satisfying of given requirements [3], [17].

The misunderstanding and ambiguity of the popular opinion about software quality however will not benefit the quality improvement effort in the industries. Thus quality must be expressed in a workable description [4], [17]. Software quality also become the key competition for software product market. Threfore, software quality control and assessment is a continous responsibility in delivering high quality software[3]. Software quality cannot be specified in an unambiguous way because it is impossible to measure certain quality characteristics directly.

A. Quality

Quality is a dynamic concept and the definitions are numerous and at variance [18]. It is a complex multifaceted concept of quality described from five different perspectives:

- The *transcendental perspective* defines quality as something that can be recognized but not defined in advance.
- The *user perspective* defines quality as fit for purpose.
- The *manufacturing perspective* defines quality as conformance to specification.
- The *product view* defines quality in terms of essential characteristics of the product in question.
- The *value-based* view defines quality in terms of the amount a customer is willing to pay for it.

Other conceptions of quality are:

IEEE Standard (IEEE Std 729-1983):

doi: 10.18178/lnse.2016.4.3.250

1) "The totality of features and characteristics of a software product that bear on its ability to satisfy given needs: for example, conform to specifications.
2) The degree to which software possesses a desired combination of attributes.
3) The degree to which a customer or user perceives that software meets his or her composite expectations
4) The composite characteristics of software that determine the degree to which the software in use will meet the expectations of the customer".

ANSI Standard (ANSI/ASQC A3/1978):

"Quality is the totality of features and characteristics of a product or a service that bears on its ability to satisfy the given needs".

B. Software Quality Assessment

Different stakeholders assess software products differently [1], [3], [19], [20]. For example, users concern about the whole product while it is operational, while the developers may be interested in developing quality software [1], [19]. Software quality engineering needs to utilize a quality model throughout the software lifecycle which incorporates all the perspectives of quality model and different stakeholders [14]. The growth of number of research on developing the new software measures is due to the increasing significance of software measurement [21]. The quality model helps to provide a base for assessing and measuring characteristics like size, complexity, performance and quality. Measurement is a mechanism for answering a variety of questions associated with the performance of any software process [22]. It applies also in evaluating software product. Software measurement is concerned with deriving a numeric value or profile for an attribute of a software component, system or process to draw conclusion about the software quality, or asses the effectiveness of software processes, tools, and methods [23].

Measuring software quality has been investigated for years in software engineering (SE). Software assessment and measurement must obey the science of measurement. To assess and measure is an identifying attribute of entities of interest in software, which are processes, products, or resources. Attributes are either internal or external [4]. A software product is also assessed by the degree of satisfaction to required quality [3]. The development of software with goal to acquire quality can avoid the waste of time and effort. Therefore it is crucial to clearly define quality requirement, and to evaluate the product at the early stage of life-cycle concretely. Software quality assessment is expected to assist developer to identify and correct the defect thus to avoid the negative assessment by users [3], [14], [24], [25].

Software measurement can be categorized into direct measurement and indirect measurement. Direct measurement includes lines of code produced, execution speed, memory size, and defect reported over some period of time. Indirect measurement of products may include functionality, complexity, efficiency, reliability, and many others. These characteristics are unmeasurable software characteristics decomposed into several subcharacteristics and metrics of quality characteristics. The unmeasurable characteristics are the base to generate measureable metrics [4]-[7], [26]. An applicable set of software metrics shall be established to enable the measurement of software quality attributes (IEEE std 1061-1992).

C. Software Quality Metrics

Software quality metric is a tool of measurement whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality [4], [27]. This area of software metric claimed to be under research in SE [28]. Nevertheless it has a major function in software engineering [23], [27]. Metric can be defined as a quantitative measures of software or processes for a given attributes to assess quality [6], [7], [15], [16]. The principle of software metrics is to make assessments throughout the software life cycle, to measure whether the software quality needs are being met. The advantages of software metrics is it provides a quantitative basis in the assessment of software quality. Thus it reduces the subjectivity and make the software quality more visible. On the other hand the for human judgement in software assessment is still demanded [6], [7], [15], [16], [27], [28].

III. Software Quality Model

A software quality model, set of characteristics and the relationships between them provides basis for specifying quality requirements and assessing quality of a software [4], [26]. A quality model is a reflection of quality from a precise view. Engineers have long recognized that in order for something to find its way in a product, it should be properly defined and specified [28]. A solid foundation in the form of an agreement upon quality model is very crucial in the industry [29], [30]. A software product is assesed by the degree of satisfaction to the required quality [3]. The development of software with goal to acquire quality can avoid the waste of time and effort and to avoid the negative assessment by users towards the application [4], [8], [23]. Therefore it is necessary to clearly define quality requirement [3]. Software quality assessment is expected to contribute and assist developer and tester to identify and correct the defect. Since 1970s to 2000s, the development of software product and software quality and assessment methods has progress [4]. The assessment method grow from measurement of complexity, estimation, internal measurements to the development of software quality model such as McCall and Boehm model [6], [31].

At current trend, software quality models are still in the scope of technology, and behavioral views of assessment. Thus Ref [6], [7] focus on development of software quality factors and metrics that based on user's perspective and views. Software certification model by user centric approach is proposed to improve the existing software certification model to meet user's needs and demands. Previous research has develop a number of software quality models to support software quality. Among the popular models are McCall's quality model, Boehm's quality model, Dromey quality model, ISO 9126,ISO 25010, and UcSoftC [6], [7], [28], [31]. The earliest models of quality such as McCall, Boehm, FURPS and ISO 9126 are limited to measure of external software characteristics which consider less other needs such as

conformance of user requirements and expectation [6], [32], [33].

A. Mc Call Quality Model (1977)

This model has been introduced in 1977 by Jim McCall, is the first of the modern software product quality models [23]. It is addressed to the system developer, to be used during the system development process. It is to match and reflect the user's opinion and system developers concern [33]. In categorizing the software quality attributes, McCall identify three main perspectives: Product Revision, Product Operation and Product Transition [34]. The model uses a hierarchy of factors, criteria and metrics to address internal and external product quality [17]. Further metrics are associated with the factors allowing quality to be measured and managed [4], [8], [32]. Among the major contributions is the relationship formed between metrics and quality factors. However one view not considered directly is the software functionality.

TABLE I: MC CALL QUALITY MODEL

Perspectives	**Factor**
Product Revision	Maintanability, Flexibility, Testability
Product Transition	Portability, Reusability, Interoperability
Product Operation	Correctness, Reliability, Efficiency, Intergrity, Usability

B. Boehm's Quality Model (1978)

Boehm's quality model follows the McCall quality model [8], [23]. It begins with the software general utility from various dimensions Like McCall's model, Boehm's model presents product quality in a hierarchy with three high level characteristics linked to seven intermediate factors, which are in turn linked to 15 primitive characteristics [17]. Therefore Boehm's model gives more emphasis on the cost-effectiveness of maintenance. It has a wider scope than McCall. This model attempts to define and express the quality of software by a predefined set of attributes and metrics as is Table II [32].

TABLE II: BOEHM QUALITY MODEL

Product Perspective	**Factors**	**Criteria**
Portabilitty		Device Independence, Completeness
As-Is-Utility	Reliability	Self containedness, Intergrity, Accuracy
	Efficiency	Accountability, Accessibility
	Human Engineering	Accessibility, Communicativeness
Maintainability	Testability	Structuredness, Accountability, Accessibility
	Understandibility	Legibility, Conciseness, Structuredness, Self-descriptiveness
	Modifiability	Structuredness, Augmentability

C. Dromey Quality Model (1995)

Ref [30] proposed a model consisting of eight high-level quality attributes, namely the same six from ISO 9126 adding Reusability and Process Maturity. The model level quality attributes, namely the same six from ISO 9126 plus Reusability and Process Maturity. It suggested a more dynamic idea for modeling the process on three prototypes concerning quality [30], [35]. Dromey claims that quality process only exist if it is based on a product quality model [33].

D. FURPS Quality Model

FURPS model proposed by Grady B. R. and Hewlett Packard Co. categorized characteristics into two different requirements such as Functional Requirements (F) which is defined by expected input & output and Non Functional Requirements in which U sands for Usability (includes human factors, aesthetic, documentation of user and material of training), R stands for Reliability (includes frequency and severity of failure, recovery to failure, time among failure), P stands for Performance (includes functional requirements) and S stands for Supportability (includes backup, requisite of design, implementation, interface) [17], [36].

TABLE III: FURPS QUALITY MODEL

Characteristics	Sub Characteristics
Functionality	Joint of characteristics, Capacities, Security
Usability	Human Factors, Aesthetics, Documentation of the user, Material of Training
Reliability	Frequency and severity of failures, Recovery to failures, Time among failures
Performance	Velocity, Efficiency, Availability, Time of answers, Time of recover, Utilization of resources
Supportability	Testability, Extensibility,Adaptability, Maintainability, Compatability, Configurability, Serviceability, Installability, Localizability

E. Systemic Quality model

The systemic quality model is developed by identifying the relationship between product-process, efficiency-effectiveness and user-customer to obtain global systemic quality [36]. Process Effectiveness and Process Efficiency are essential elements of the model but are not present in the Dromey or ISO 9126. It includes Process and Product dimension. In order for the quality evaluation to be systemic, The Process dimension is incorporated [33]. The model serves as a benchmark that allows their products to evolve and be competitive. The disadvantages of this model is the absence of the user requirements and conformation aspects [6].

F. ISO 25010:2011

The international standard most directly applicable to software quality control is SQuaRE series of standard of the *Internatinal Organization for Standardization (ISO)*. This Standard is derived from revised ISO/IEC 9126:1991. It defined quality characteristics and described a software product evaluation process model and incorporates the same software quality characteristics with some amendments [5], [37]. ISO 25010 consists of quality in use and product quality models as summarized below. Product Quality is the static properties of the model concerns of computer software and dynamic properties systems. The Quality in Use model relates to the interaction outcome when a product is used in a particular context of use. These characteristics subdivided into the respective sub characteristics. Each characteristic is composed of a set of related sub characteristics [32].

G. UcSoftC Quality Model

The user centric software certification (UcSoftC) model is a new model [6] that is claimed to fulfill the requirement of

organization according to demands and constraints in software product quality and assessment because it supports the user centric approach in assessing and certifying the software [5]-[7]. The model proposed the improvement in the certification process. It enables software users to assess and certify their own products in their own environment with tailored and chosen attributes based on the organization's requirements and expectations.

TABLE IV: ISO 25010:2011: PRODUCT QUALITY MODEL

Characteristics	Sub Characteristics
Functional suitability	Functional completeness, Functional Appropriateness , Functional correctness
Performance efficiency	Time Bahavior, Resource Utilization, Capacity
Compatibility	Co-existence, Interoperability
Usability	Appropriateness recognizability, Learnability, Operability, User error protection, User interface aesthetics, Accessibility
Reliability,	Maturity, Availability, Fault Tolerance, Recoverability
Maintainability	Modularity, Reusability, Analysability Modifiability, Testability
Portability	Adaptability, Installability, Replaceability
Security	Confidentiality, Integrity, Non-repudiation, Accountability, Authenticity
Operability	Appropriateness, Recognizability

TABLE V: ISO 25010:2011 : QUALITY IN USE QUALITY MODEL

Characteristics	Sub Characteristics
Satisfaction	Usefulness, Trust, Pleasure, Comfort
Effectiveness	
Efficiency	
Freedom from Risk	Economic risk mitigation, Health and safety risk mitigation, Environmental risk mitigation
Context coverage	Context completeness, Flexibility

H. PQF

Pragmatic quality factor (PQF) is practical software quality model which can be used in assessment of software operating in certain environment involving the user. PQF consists of four main components: I) behavioural attributes, 2) impact attribute, 3) responsibility and measurement of metrics and 4) classification of attributes and weight factors. This model is beneficial and valuable to the organizations because it applies Weighted Scoring Method [6], [7]. It also fill the void of the earliest models of quality such as McCall, Boehm, FURPS, which limit the measurement of external software characteristics consider less other needs such as conformance of user requirements and expectation. This model exhibit that the unmeasurable characteristics can be measured indirectly using measures and metrics approach. It has been validated involving assessment and certification applications in real case studies in Malaysia [6], [7].

IV. USER CENTRICITY

User involvement in assessing software is vital as they involve in many stages of software lifecycle [3], [12]. Commonly, user centric approach focuses in emphasizing user perspective in assessing software product operating in their environments [5]-[7]. The significance of software is not only for business excellence and sustainability, but also include the user and society [6]. Thus embracing user centricity and providing the paradigm where people and user are considered as stakeholder is crucial [5], [38]. The convention of social network application like *Twitter*, *WhatsApp*, *Facebook*, and many others signify the immense pressure of software and computers in human life [5]. Ref [38] offered his conception on customer satisfaction; (1) when the basic promises is fulfilled, (2) no negative consequences is produced and (3) the customer is happy with the product and service. Software quality is evolving beyond static measurement to an expansive area of quality description, the importance of human aspect in assessment must be incorporated in the process. Previous studies had suggested the human aspect in software quality, however did not include comprehensively this aspect together with the behavioural aspect of software quality [1], [5]-[7], [29]. Improvements in user involvement may lead to an improved quality as perceived by the end users [1], [5]-[7]. The available software quality models such as McCall, Boehm, Dromey and FURP'S [27], [39] only target the software product or process characteristics and does not fit to measure software quality from user point of view [4]. Therefore those models need to be revised and extended to include users in the process [1], [38].

V. THE CONCEPT OF SUSTAINABILITY

There still is no concrete and definite direction for the different aspects of sustainability that are observable from the point of view of software engineering (SE) [9], [10], [36]. This can due to the fact that the concept of sustainability does not become adequately tangible from the definition [11], [12]. Currently, there are little research on the different aspects of sustainability in SE while other disciplines are already more active [9]-[12], [36]. The general definition of sustainability is the "capacity to endure" [11] and sustainable development as "meeting the needs of the present without compromising the ability of future generations to meet their own needs" [16]. Sustainable software also can be defined as *software whose direct and indirect negative impacts on economy, society, human beings, and the environment resulting from development, deployment, and usage of the software is minimal and/or has a positive effect on sustainable development* [40]. Sustainability has not been fully supported as a significant, first-class interest by traditional SE [12], [40]. Software engineers approach specific topics that have to do with sustainability in this discipline. As example, green IT, efficient algorithms, smart grids, agile practices and knowledge management, but it is still deficient of a common understanding consensus of the sustainability concept in SE and if and how it can be applied to SE [11], [12], [40].

While sustainability is a standardized practice in a number of engineering disciplines there is currently no such awareness within the SE community [9], [10] , [18], [28], [36], [40]. Thus the sustainability assessment can be considered as another quality aspect. In [13], [14] a quality model (25010+S) based on ISO/IEC 25010 that considers sustainability as a new factor that affects quality was presented. Ref [35] propose a Generic Sustainable Software Star Model (GS3M) that forms the basis towards a "complete" view of sustainable software. The model covers different sustainability

dimensions: environmental, technical, social, individual and economic. Corresponding software sustainability values, attributes and metrics are defined for each dimension. The measurement of software sustainability is expected to provide basis for measured software improvement [35], [41].

Since many years sustainability is becoming a challenging issue in software engineering domain. However no clear nor exhaustive characterization was proposed to the concept of "sustainable software" [11], [12], [28], [40], [42]. Therefore sustainability remains an intangible idea for software systems and consequently can't be assessed nor controlled nor enhanced.

VI. Conclusion and Future Works

Software quality engineering needs a quality assessment model throughout the software lifecycle and includes all the perspectives of quality model. The quality model should be the basis for measuring software sustainability. The quality model must involve the users as the significant stakeholders because users demand a sustainable software that gives positive impact on economy, society, human beings, and the environment resulting from development and deployment of the software. Therefore software quality should be considered in assessing the software sustainability. As the concept of sustainability in software engineering is still in infancy, this research is to explore the inclusion of software quality in the assessment of software sustainability from the user's perspective.

Acknowledgment

This research is partly funded by National University of Malaysia under Advanced Nature Inspired Computing For Spatio-Temporal Climate Change Predictive Analysis Grant Scheme (AP2013-007) and Kolej Poly-Tech MARA, Malaysia.

References

[1] I. Atoum and C. H. Bong, "A framework to predict software 'quality in use' from software reviews," in *Proc. the First International Conference on Advanced Data and Information Engineering*, 2014, pp. 429–436.

[2] M. Azuma, "Systems engineeringapplying ISO / IEC 9126-1 quality model to quality requirements engineering on critical software department of industrial and management," in *Proc. the 3rd International Workshop on Requirements Engineering for High Assurance Systems*, Kyoto, Japan, 2004.

[3] D. Garvin, "What does 'product quality' really mean?" *Sloan Management Review*, vol. 26, pp. 25-43, 1984.

[4] N. Fenton, "Software measurement : A necessary scientific basis," *IEEE Transaction on Software Engineering*, vol. 20, no. 3, pp. 199–206, 1994.

[5] A. Deraman, J. Yahaya, F, Baharom, and A. R. Hamdan, "User-centred software product certification : Theory and practices," *International Journal Of Digital Society (IJDS)*, vol. 1, no. 4, pp. 281–288, 2010.

[6] J. Yahaya, A. Deraman, A. R. Hamdan, and Y. Jusoh, "User-perceived quality factors for certification model of web-based system," *International Journal of Computer, Information, Systems and Control Engineering* ,vol. 8, no. 5, pp. 640–646, 2014.

[7] J. Yahaya, A. Deraman, S. R. Ibrahim, and Y. Y. Yusoh, "Software certification modeling: From technical to user centric approach," *Australian Journal of Basic and Applied Sciences*, vol. 7, no. 8, pp. 9-18, 2013.

[8] J. Patton, "Understanding user centricity," *IEEE Software*, vol. 24, no. 6, pp. 9–11, 2007.

[9] C. Calero, M. Bertoa, and M. Moraga, "A systematic literature review for software sustainability measures," in *Proc Green and Sustainable Software (GREENS) 2nd International Workshop*, 2013, pp. 46–53.

[10] C. Calero, M. Bertoa, and M. Moraga, "Sustainability and quality: icing on the cake," in *Proc 2nd International Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy)*, 2013.

[11] B. Penzenstadler, "Towards a definition of sustainability in and for software engineering," in *Proc. 28th Annual ACM Symposium on Applied Computing* 2013, pp. 1183–1185.

[12] B. Penzenstadler and H. Femmer, "A generic model for sustainability," Technical Report, p. 6, 2012.

[13] *IEEE Standard Glossary of Software Engineering Terminology*, 1990.

[14] S. Barney and C. Wohlin, "Software product quality: Ensuring a common goal," *Lecture Notes in Computer Science*, Springer, vol. 5543, pp. 256–267, 2009.

[15] N. Bevan, "Measuring usability as quality of use," *Software Quality Journal*, vol. 4, pp. 115–130, 1995.

[16] N. Bevan, "Quality in use: Meeting user needs for quality," *Journal of Systems and Software*, vol 49, pp. 89–96, 1999.

[17] R. A. Khan, K. Mustafa, and S. I. Ahson, *Software Quality Concepts and Practices*, 2006.

[18] M. Al Hinai, "Quantification of social sustainability in software," in *Proc. IEEE 22nd International Requirements Engineering Conference*, 2014, pp. 456–460.

[19] I. Somerville, *Software Engineering*, 9th ed. Addison-Wesley, 2011, p. 668

[20] S. K. Dubey, S. Ghosh, and P. A. Rana, "Comparison of software quality models : An analytical approach," *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, no. 2, pp. 2250-2259, 2012.

[21] S. F. Ahmad, M. R. Beg, and M. Halem, "A Comparative Study of Software Quality Models," *International Journal of Science, Enginering and Technology Research*, vol. 2, no. 1, pp. 172–176, 2013.

[22] V. Caldiera and H. Rombach, "The goal question metric approach," *Encyclopedia of Software Engineering*, vol. 2, pp. 1–10, 1994.

[23] B. Kitchenham and S. L. Pfleeger, "Software quality," *IEEE Software*, 1996.

[24] H. Subramaniam and H. Zulzalil, "Software quality assessment using flexibility: A systematic literature review," *International Review on Computers & Software*, vol. 7, no. 5, 2012.

[25] D. Jamwal, "Analysis of software quality models for organizations," *International Journal of Latest Trends in Computing*, vo1, no. 2, pp. 19–23, 2010.

[26] M. Azuma, "Software products evaluation system: Quality models, metrics and processes — International Standards and Japanese practice," *Information and Software Technology*, vol. 38, no. 3, pp. 145–154, 1996.

[27] K. Akingbehin, "Taguchi-based metrics for software quality," in *Proc. Fourth Annual ACIS International Conference on Computer and Information Science*, 2005, pp. 713–716.

[28] Marc-alexis and M. Ing., "Software quality model requirements for software quality engineering," *Quality Engineering*, pp. 38–54, 2005.

[29] N. Bevan and M. Azuma, "Quality in use: Incorporating human factors into the software engineering lifecycle," in *Proc. Software Engineering Standards Symposium and Forum*, 1997, pp. 169-179.

[30] R. G. Dromey, "A model for software product quality," *IEEE Transactions on Software Engineering*, vol 21, pp. 146–162, 1995.

[31] Suman and M. Wadhwa," A comparative study of software quality models," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 4, pp. 5634–5638, 2014.

[32] S. Wagner, *Software Product Quality Control*, 2013.

[33] O. Maryoly, P. Mar á, and R. Teresita, "Construction of a systemic quality model for evaluating a software product," *Software Quality Journal*, vol. 11, no. 3, pp. 219-242, 2003.

[34] *IEEE Standard for a Software Quality Metrics Methodology*, 1993.

[35] A. Rawashdeh and B. Matalkah, "A new software quality model for evaluating COTS components," *Journal of Computer Science*, vol. 2, no. 4, pp. 373–381, 2006.

[36] C. Calero, M. Moraga, M. Bertoa, and Duboc, "Quality in use and software greenability," *CEUR Workshop Pcoceeding*, vol. 1216, pp. 28-36, 2014.

[37] H. Abu Bakar and R. Razali, "A preliminary review of legacy system evaluation models," *Journal of Software*, vol. 9l, no. 1, pp. 314–318, January 2014.

[38] P. J. Denning, "What is software quality," *Communications of the ACM*, vol 35, no. 1, pp. 13–15, 1992.

[39] R. Amri and N. B. B. Saoud, "Towards a generic sustainable software model," in *Proc. Fourth International Conference on Advances in Computing and Communications*, 2014, pp. 231–234.
[40] M. Dick, S. Naumann, and N. Kuhn, "A model and selected instances of green and sustainable software," *What Kind of Information Society? Governance, Virtuality, Surveillance, Sustainability, Resilience*, vol. 238, pp. 248-259, 2010.
[41] R. C. Seacord, J. Elm, W. Goethert, G. A. Lewis, D. Plakosh, J. Robert, and L. Wrage, "Measuring software sustainability," in *Proc. the International Conference on Software Maintenance*, September 2003, p. 450.
[42] M. Al Hinai and R. Chitchyan, "Social sustainability indicators for software: Initial review," in *Proc. the Third International Workshop on Requirements Engineering for Sustainable Systems*, 2014, vol. 1216, pp. 21–27.

Nur Zuria Haryani Zakaria is an information technology lecturer at Kolej Poly-Tech Mara (KPTM). She obtained her master degree from National University of Malaysia (UKM) in 2007, and is a PhD candidate at the National University of Malaysia (UKM). Her research interests are software quality, software sustainability, software assessment, data mining and artificial intelligence.

Abdul Razak Hamdan is a professor at the Faculty of Technology and Information Science and Technology, National University of Malaysia (UKM). His research interest are combinatorial optimization data mining and impact study and strategic planning. He is an active researcher with several postgraduate students. He is the chairman of the Content Base Informatics Niche in UKM and is the Head of Data Mining And Optimization Group at this Faculty

Jamaiah H. Yahaya is an associate professor at the National University of Malaysia (UKM). She received her masters from University of Leeds, UK in 1998, and PhD from The National University of Malaysia (UKM) in 2007. Her research interests are software certification, software quality, software maintenance, and software ageing/anti-ageing.

Aziz Deraman is the dean of the School of Informatics and Applied Mathematics, University Malaysia Terengganu. He received his masters from Glasgow University in 1984 and PhD from the University of Manchester Institute of Science and Technology (UMIST) in 1992. He is presently a senior professor of software engineering specializing in software process, management and certification.