Software Quality Evaluation: User's View

Anas Bassam AL-Badareen,
Mohd Hasan Selamat, Jamilah Din, Marzanah A. Jabar, Sherzod Turaev

Abstract: - Nowadays, software products rapidly increased and it is usage not limited to specific people or corporation. It is now used in the most of human life activities. Therefore, the quality of the software product is increasingly being important and the users demanding higher quality than ever before. However, most of this research is focused on the internal/development view of quality. Hence, in software development, strong attention must be given to the user's satisfaction. Recently, the studies intend to understand the users' perspective of the software quality. In this study, we intend to discuss the characteristics of the software products that influence the users' satisfaction on software quality. Based on the well-known software quality models and the emotion of the software users, a model of software quality evaluation based on users' views is proposed.

Key-Words: - Software Quality, Quality Model, Software Evaluation, User Perspective.

I. Introduction

In last decade, the cost of software products is lower, that caused grow in the software market contention and software products are being used by individuals in addition to the corporations. Therefore, research in software engineering increasingly grew and focused on software quality evaluation and enhancement, whereas most of these researches concentrate on the internal/ development perspective [1].

Since, the software market interest on the user's satisfaction, and their quality expectations are not typically based on size and complexity [2]. More attention to the users' perspective in software quality is required. Software users from different education background and culture are considered in developing software products. Hence, without considering these factors, the software will be less used [3], which means the software product failed in the market.

According to Bevan [4], Garvin [5] distinguish between five different approaches of defining a quality of software product, one of these approaches is a user perceived quality. That it means a combination of product attributes which provide the greatest satisfaction to a specified user.

According to Bevan [6], the expectation of software quality in both consumer and professional market is rapidly increased. That is, matching the software product with the real users needs is increasingly demanded. Moreover, he state that it is not enough sufficient to deliver a software product has a technical excellence. Also, it has to be easy to use and suitable for the end users in the work practices and activates, either the users are normal consumer or professionals.

However, Chulani [2, 7] presented software product quality from customer's perspective, and the companies can increase the revenue by increase customer's satisfaction and improve the quality of the software product. That is in order to develop any software product, the user's requirements have to be addressed either functional and non-functional (quality of services) [8].

II. Software Quality Views

ISO 9126 defined three main different views of software quality. As shows in figure 1, the quality of the software product is required for three different people for different purposes: Manager, Developer, and User.

The manager is interested in the overall quality characteristics. That he has to balance the quality with management criteria, achieve the user's requirements with a certain level of quality within specific time, limited resources (human, tools), and limited cost.

The users are mainly interested in the software usage without knowing it is internal aspects. Therefore, they considered the reliability and the ability of the software to

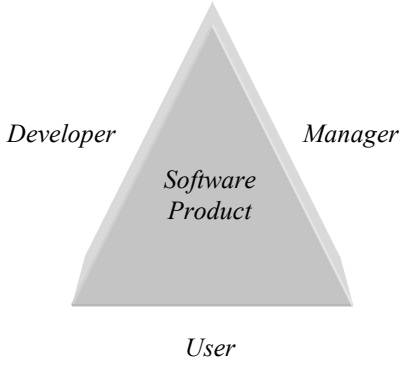

Figure 1: Software Quality Views

Anas Bassam AL-Badareen is with the University Putra Malaysia, 43400 UPM, Serdang, Selangor, Malaysia. Phone: 601-72301530; e-mail: anas_badareen@hotmail.com.

Mohd Hasan Selamat is a professor of software engineering with University Putra Malaysia. Phone: 603-89471720; Fax: 603-8946 6577; e-mail: hasan@fsktm.upm.edu.my.

Jamilah Din PhD is with University Putra Malaysia, 43400 UPM Serdang Selangor, Malaysia; e-mail: jamilah@fsktm.upm.edu.my.

Marzanah A. Jabar PhD is with University Putra Malaysia, 43400 UPM Serdang Selangor, Malaysia; e-mail: marzanah@fsktm.upm.edu.my.

Sherzod Turaev is a Postdoctoral Researcher with University Putra Malaysia, 43400 UPM, Serdang Selangor, Malaysia; e-mail: sherzod@fsktm.upm.edu.my.

perform the required functions easily and efficiently in different environments.

The developers are mainly required to develop software products within certain level of quality as users' needs. On the other hand, the developers are interested on the internal quality characteristics. That affects their tasks of software development process. Therefore, the differentiation between these two sides is very important.

According to ISO 9126, the main consideration of the users is the software usability, performance, and its effects without knowing what inside it, how it is work, or how it was developed.

Since, the software users does not care about all of software characteristics that are required to identify the quality of the software product, it is seems to be inaccurate to show them the quality that they are looking for. Therefore, the quality of the software as users need is very important in the market.

The current software quality models combine the different points of views: Manager, Developer, and user. Therefore, the models did not show the quality of the software product as a user need, it shows a combination of users' and developers quality factors.

III. Software Quality Models

Since 1978, when McCall proposed first software quality model, several models were proposed to evaluate the characteristics of the software products. In the following, five well-known software quality models are discussed.

A. McCall

McCall [9] is the first software quality model was proposed in 1978 by the US air-force electronic systems division (ESD), the Rome Air Development Center (RADC), and General Electrics (GE). The model was developed to enhance the quality of the software product and to consider the relationships between the external and internal characteristics of the software product. McCall defined the layers of quality model as:

- Factors: identified 11 factors to describe the external view of the system (from user point of view)
- Criteria: identified 23 criteria to describe the internal view of the system (from developer point of view);
- Metrics: used to provide a scale and method for measurement.

Moreover, the model classified the quality characteristics around three main categories, product operations, product revisions, and product transitions. However, the model lacks of measuring the functionality of the software product.

B. Boehm

Boehm [10] added new factors to McCall model, and hierarchical relationships were defined in order to contribute to overall of software quality. The model aims to address the contemporary of shortcomings of models that automatically and quantitatively evaluate the quality of the software product. Moreover, the model emphasize on the maintenance process with respect to the software utility. However, the models lacks of measuring the functionality, reusability, and usability of software products. Boehm defined the layers of quality model:

- High-level characteristics;
- Primitive characteristics;
- Metrics.

C. FURPS

FURPS [11] (Functionality, Usability, Reliability, Performance, Supportability). Robert Gradly and the Hewlett-Packard Co. classified the characteristics into two main categories according to the user's requirements, functional requirements, and non-functional requirements. IBM rational software extended the model into FURPS+, which added new category, called constraints.

- Functional requirements (F): Defined by input and the expected output.
- Non-functional requirements (URPS): Usability, reliability, performance, and supportability.
- Constraints (+): Design requirements, Implementation requirements, Interface requirements, and Physical requirements.

D. Dromey

Whereas, the evaluation process is different for each software product, a dynamic idea of software evaluation is required [12]. The model intends to increase the understanding of the relationships between the attributes and the sub-attributes. Two main layers are defined in this model, high level attributes and subordinate attributes. The model lacks of criteria for software quality measurement.

E. ISO 9126

ISO 9126 is a part of ISO 9000 standard, which is the main standard of quality assurance. The model consists of 21 sub-characteristics distributed on six main characteristics. These characteristics are able to be used for different types of software products and data.

However, these models combined the different points of views: Manager, Developer, and user. Therefore, there is no a clear picture of the software quality shows to the intended users.

For example, if such software product has a high maintainability and low usability may be same as software has a high usability low maintainability. Table 1 shows two different values results same total quality.

IV. User's Perspective Quality Factors

Whereas, different characteristics of software product were considered and measured, a number of these characteristics are considered by the end user's. In the following, the list of software characteristics, that considered by the end users and affect their emotions.

Table 1: Different Quality Values

Factor	Product A	Product B
Maintainability	8/10	6/10
Usability	6/10	8/80
Total Quality	7/10	7/10

A. Functionality

The main idea of any software product is to perform specific business function. Therefore, the functionality of the software product considered as crucial factor in the software quality, which identify whether the software is usable or useless, regardless the values of other software quality factors. The functionality of the software presents whether the software product is suitable, the result is accurate, and whether certain standard is followed in order to perform intended functions. Figure 2, shows the characteristics of the software functionality.

The suitability of the software presents how the system fit the developer's requirements [13]. Therefore, the suitability evaluates the ability of the software product to produce desired result and appropriate for a specified environment [14].

Software accuracy is defined as the ability of the software products to achieve its requirements [9], by producing accurate result as required by the system developer [13, 15]. Software accuracy affects the system safety, process continuity, maintainability, and totally the cost of the system [16].

The compliance presents whether the system has followed any standard or certificates to achieve the user requirements.

B. Reliability

The reliability of the software represents the ability to perform the intended function properly without any failures [9]. That is maintaining a level of services under specific condition within specific period of time during system operation [15, 17-18]. Therefore, the reliability measures the failures occurred in the software product within defined period of time [13, 15].

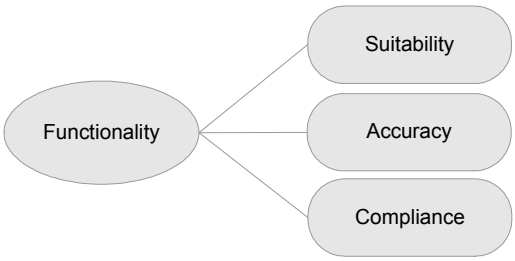

Figure 2: Software functionality

Moreover, the reliability considered the information and the system function safety harms that may be caused by unauthorized people. Hence, the reliability of the software product consists of several characteristics: integrity, fault recovery, and maturity. Figure 3, shows the characteristics of the software reliability.

i. Software Integrity

High attention must be given to the system security [19], whereas, data and information are key factors for any establishment. Software products provide a support to recognize a people who might use a system, which identify the system users into authorized and unauthorized users [17, 20].

The software security is deal with the privileges that are given to the system users, in order to access specific functions such as (view, add, update, or delete data) [21]. Generally, it can be defined as the ability of the system to defend itself against unauthorized use [19] and to protect it components (data, information and functions) from [15].

Whereas software integrity intends to protect the system from any harm, the errors and faults need to be considered. Hart [22] defined the software integrity as a probability that a given system will operate within its specified limits without the occurrence of a software incident, which is directly related to the number of errors remaining in the software.

Moreover, the quantitative and qualitative risk analyses are used to reduce the level of threats risk to acceptable level. They also mention about the security preserving that can apply during software development and after produce the software.

In terms of user's access, two layers of access are defined, access audit and access control. The access audit allows the authorized users to enter and use a system. This level of access defined the user to authorized and unauthorized. Access control allows specific authorized users to access specific function within the system. At this level, the authorized users classified into several groups according to the privileges that are given to them.

Hence, the levels of security doesn't covered the responsibility of each action occurred within the system, which represents the level of the software security. The software accountability aimed to ensure that every action occurred in the system has been traced back to some entity [23].

ii. Fault Recovery

Indeed, it is very difficult to expect all of failure that may occur during system operation. Hence, the ability of the system to perform it is function during failure is a crucial during software using. The failure characteristics and types are calculated, the frequency and severity of failures and the time among failures. Fenton [24] presents the failure types and the reports of the failures collection. Moreover, the behavior of the system and the affection of the failures on the system are considered.

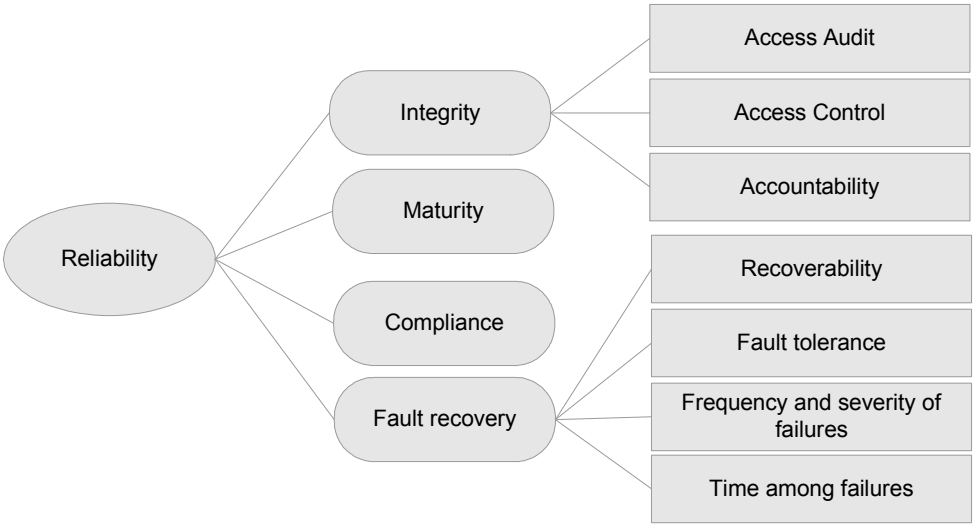

Figure 3: Software Reliability Characteristics

The fault tolerance is used to understand the context of the system which describes whether the system is able to maintain specific level of performance during faults that may occur during system using [13, 15]. At the same time, the system elegance during failure recovery is considered. The software recoverability represents the ability of the system to reestablish its level of performance and restore the data, which are directly affected from an unexpected failure [13, 15].

iii. Maturity

Practically, the history of the software and the certificates are like stamp that verify the software features and characteristics. Software history describes the maturity story of the product [25], how much work has been done using this technology or number of versions released of the technology [15].

iv. Compliance

The compliance presents whether the software has followed any standard or certificate in order to achieve certain level of reliability.

C. Performance

Software performance is a most affected software characteristic, which is affected by everything in the system product, from a software characteristics to the system environment such as operating system, middleware, hardware, and communication networks [26]. System performance is a make-or-break quality for software [27], which is an important nonfunctional attribute of software systems for producing quality software [28-30], that consider the run time property [31].

System performance is characterized by the amount of useful work accomplished by a system compared to the time and resources used. The performance factor is destined to evaluate whether the software application running efficiently on the computing resources available.

The performance factor represents the degree of the system efficiency to produce desired result during system operation. This degree is represented by combination of software and hardware attributes which influence on the time of answer and the range of the software services coverage. Figure 4, shows the characteristics of the performance factor.

i. Software Coverage

In terms of coverage, software performance concerned about the availability, accessibility, and the velocity. The availability of the software is the proportion of time a system is in a functioning condition [32]. This time up of the system represents the duration time of the system responding. The velocity of the system is the average rate of successful massages that are delivered through communication channels.

Software accessibility presents the degree to which a product, device, service, or environment is accessible by as many people as possible. This factor concerned about the area covered by the system. Beside the coverage area, the average of the success services that offered on this area is covered.

ii. Software Efficiency

In terms of speed, the performance factor represents the efficiency to perform software functions and the time to recover a system from failures. The efficiency is the ability of the system to use its hardware and software resources to perform its functions in order to achieve required requirements [9, 20].

Thus, this factor concerned about the amount of resources used [13] under specific conditions [17] during required functions performing [15].

Therefore, it is dealing with processor speed and storages capacity.

The time behavior represents the ability of the system to

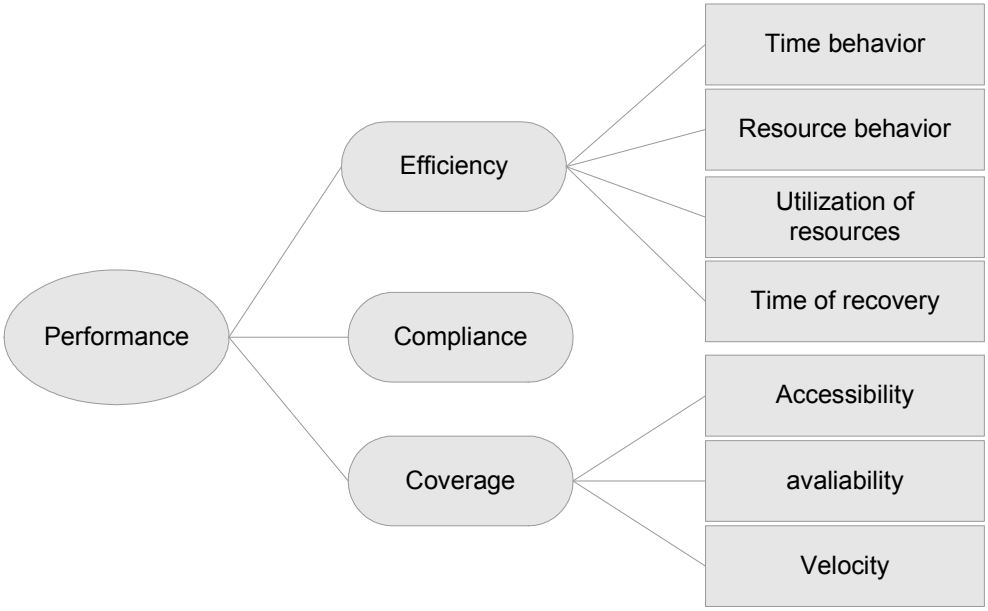

Figure 4: Software Performance Characteristics

perform specific function under stated condition within appropriate response time and throughput rate.

Resource behavior is the capacity of the storages that used under specific condition in order to perform specific task. Besides, the efficient use of the resources is considered. The utilization of resources represents the ratio of the available resources used by the software application.

The time of recovery represents the time required by a system to reestablish it is level of performance and recover it is data that affected from unexpected failures [32].

iii. Compliance

The compliance presents whether the system has followed any international standard or certificate in order to achieve a level of performance.

D. Usability

According to ACM the usability engineering (called human-computer interaction engineering) is defined as "a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and the study of major phenomena surrounding them".

The characteristics of usable software as shows in figure 5, were discussed in [33]. According to the software life cycle phases, the usability characteristics were classified into three main categories: Interface characteristics, training, and operation supportability.

i. Interface Characteristics

The interface factor considered the user interface characteristics, which are aesthetic, consistency of the user interface, and communicativeness. The interface aesthetic presents the beautifulness of the interface and how much it is liked by the end users. The consistency of the user interface presents whether the user interface has any contradictions in term of words, situation, or actions. The interface communicativeness presents how software well communicates with the end user.

ii. Training

The training factor considered the material and the process of produce a motivated user who has basic skills to operate the system. This factor consists of material of training and human factor. The material of training is the documents are used to train the end users.

The material of training factor presents the quality of the materials that used to teach the end users the basic skills of how to use a system. This factor consists of completeness, clarity, consistency, and suitability.

iii. Operation Supportability

Operation supportability is the facilities that are used to support the end users during using the system.

The user assistance is a general term for guided assistance to software product users. Assistance can automatically perform procedures or step users through the procedure, depending on the question that the user asked.

Online help (aid in line) is a form of users' assistance. That is designed to give assistance in the use of a software application or operating system by present information on a broad range of subjects through computer software. The online help version of the installation instructions meets the users' usability requirements by allowing users to access information directly from the interface itself [34].

User document is an electronic or printed body of material that provides information to users of software. The user

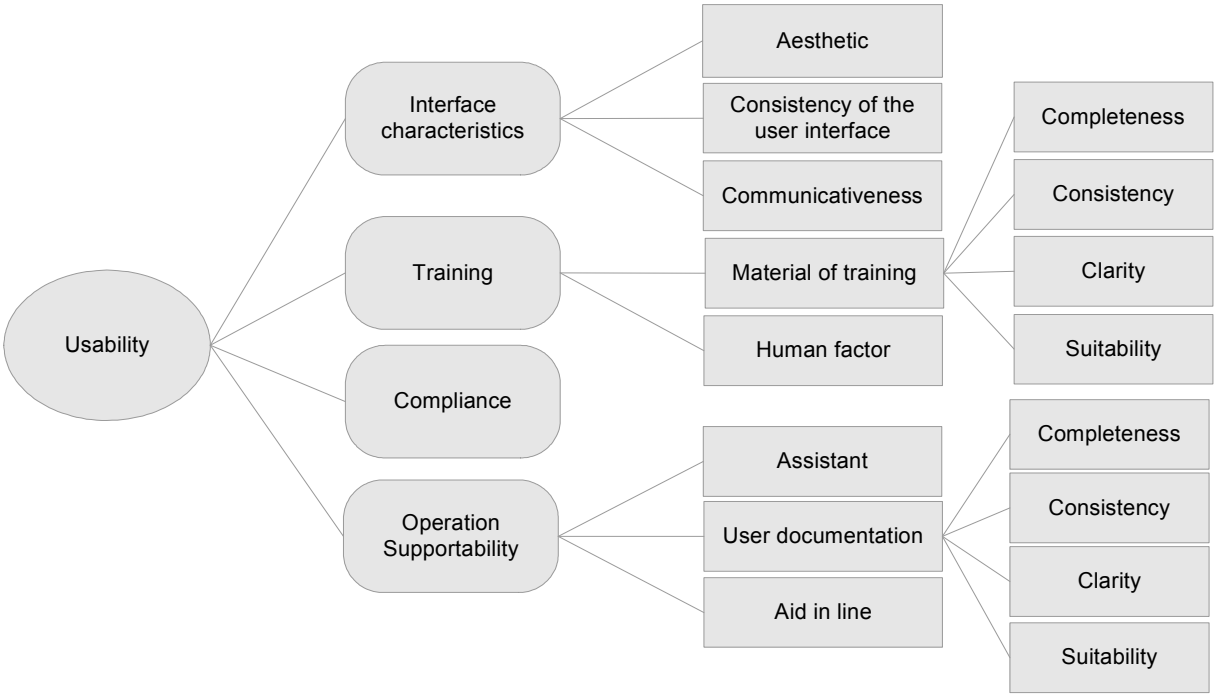

Figure 5: Usability Characteristics

documentation and the help system should be complete; the help should be context sensitive and explain how to achieve intended tasks [13]. Therefore, the document should be clear, complete, consistent, and suitable.

iv. Compliance

The usability compliance shows whether the software product has followed any international standard or certificate in order to achieve the level of usability.

E. Transferability/Portability

Software transferability expresses the ability of the software to work properly in different type of platforms [9]. It is deal with effort required to transfer a program from one hardware configuration and/or software system environment to another [18, 20] with little modification [13]. This characteristic refers to how the software can be adopted to changes its environment or with its requirements [17]. Figure 6, shows the characteristics of the portable software.

i. Coexistence

Coexistence (integrated) is a state in which two or more systems are working together while respecting their differences and resolving their conflicts nonviolently. System integration is combination of several functions of several productivity software programs into one application. In order to evaluate whether the system able to integrate with others, two characteristics have to be considered, software system independence and machine independence. Software system independence is represent the degree to which program is independent of nonstandard programming language features, operating system characteristics and other environment constraints. Machine independence (hardware independence) is the degree to which the software is de-coupled from its operating hardware.

ii. Adaptability

Software adaptability/interoperability/interface facility is the ability of the system to provide the users with tools to make them able to change the system characteristics [35]. In order to be able to adapt the system to different specified platforms [13]. It is evaluated by the degree of ease with which the system is adapted to new environments. Hence, adaptability factor concerned about software modularity, communication communality, and data communality.

Decomposable (modular) system which is combines several independent manageable parts. These parts are developed to be communicative with other parts and to be independent from any out affect. Communication commonality is represents the degree to which standard interfaces, protocols and bandwidth are used. Data commonality is explicit the use of standard data structures and types throughout the program.

iii. Setup Facility

The setup ability of the software is concerned, which is the ability to install the software and to replace the previous versions. Software install-ability is the ability to install the software product easily in different platform [13, 15, 35]. Replace-ability represents the ability of the software to replace others specified in the environment of that software [15]. It evaluate whether the software compatible with its previous

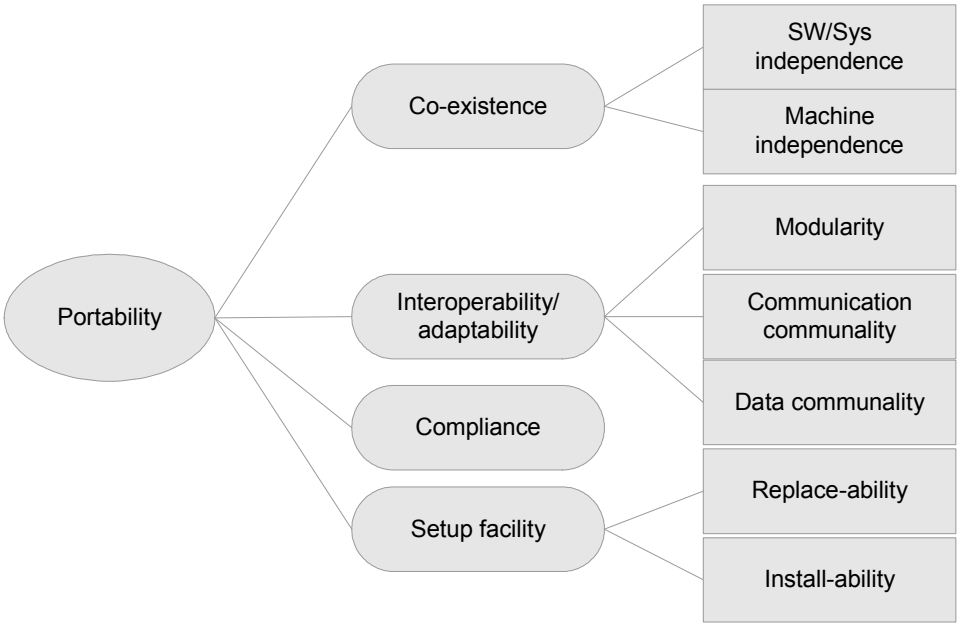

Figure 6: Portability Factor

versions, which means that the software product can easily replace the previous version without any major efforts [13].

3.5.4 Compliance

The compliance of the portability factor is also considered. It represent whether the system has followed any standard or international certificates, in order to verify the portability characteristic.

V. Discussion and Implication

User's satisfaction is the essential key of success any product in the market. That is the software market interests on the user's satisfaction, and their quality expectations are not typically based on size and complexity. Recently, software quality research concentrates on user satisfaction through using software products. This helps software developers to produce software products likely and accepted by end users.

The current software quality models considered the different views in the total quality of the software product, whereas the end users are not interested in the internal quality, such as reusability and maintainability. The end users only interested on the ability of the software product to perform their requirements as they need efficiently without any problems.

The hierarchical structure of the software characteristics is mainly considered in this model in order to contribute in overall of the total quality.

The structural relationships have been defined based on the definition, objectives, and the requirements of every characteristic in the software product. Moreover, the relationships between quality characteristics in the existing models are considered in order to provide reliance to the relationships we defined in the new model.

For example, in software usability, we defined the characteristics that intend to make the software product easy to use as we discussed previously [33]. The reliability of the software product is the ability of the software product to be dependable by the end users.

The relationship between the security and the fault recovery is that both of them are required to make the software trustworthiness by the users. Moreover, the security intends to save the software functions and data from external threats (non-authorized used). Whereas the fault recovery intends to save the software functions and data from any internal threats (errors that may occurred in the system). Moreover, the external threats caused a failure in the system and damaged in the data in addition to use them. Whereas, the failures occurred in the system may weaken a system in order to be defended against any external threats.

The comprehensiveness of the quality evaluation is one of the main issues were considered in this model. This is used in order to increase the accuracy of the quality value for every factor and finally in the total value of the software quality [36].

VI. Conclusion and Future Work

Since last decade, the expectation of software quality in both consumer and professional market rapidly increased. That is, matching the software product with the real users needs is increasingly demanded. Moreover, it is not enough sufficient to deliver a software product has a technical excellence. Also, it has to be easy to use and suitable for the end users in the work practices and activates, either the users are normal consumers or professionals.

In this study, the model of software quality evaluation based on user's view is developed. The model basically defined based on the well-known models in software products

evaluation. We identified the main characteristics of the software product that interested by the end users in order to be likely and usable. Five main quality factors are defined, functionality, reliability, performance, usability, and portability. These factors are analyzed and discussed in details in order to identify their sub characteristics. The process of defined the relationships between the characteristics was based on the aims of these characteristics and their contribution in the total quality of the software product.

However, the proposed model broadly considered the characteristics of the software product and is not taking into account specific scope of software products. That is every type of software product has it is singularity. Therefore, we intend to consider specific scope of software products in order to identify the method

Moreover, in this model, the user's view is considered only, whereas no any mentions about the others, such as developers. In the future, the quality evaluation based on developer view is managed to be considered.

REFERENCES

[1] B. Boehm, S. Chulani, J. Verner, and B. Wong, "Fifth Workshop on Software Quality," in *29th International Conference on Software Engineering - Companion.*, 2007, pp. 131-132.

[2] S. Chulani, P. Santhanam, D. Moore, and G. Davidson, "Deriving a software quality view from customer satisfaction and service data," 2001.

[3] R. Holcomb and A. L. Tharp, "An amalgamated model of software usability," in *Proceedings of the 13th Annual International Computer Software and Applications Conference.*, 1989, pp. 559-566.

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

[5] D. A. Garvin, "What does" product quality" really mean?," *Sloan management review,* vol. 26, pp. 25-43, 1984.

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

[7] S. Chulani, B. Ray, P. Santhanam, and R. Leszkowicz, "Metrics for managing customer view of software quality," 2003.

[8] C. Rohleder, "Quality product derivation: a case study for quality control at Siemens," 2009, pp. 51-56.

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

[10] B. Boehm, "Characteristics of software quality," North-Holland, Amsterdam, Holland1978.

[11] K. Khosravi and Y. G. Guéhéneuc, "A quality model for design patterns," 2004.

[12] G. Dromey, "A Model for Software Product Quality," *IEEE Transactions on Software Engineering,* vol. 146, p. 21, 1995.

[13] A. Sharma, R. Kumar, and P. S. Grover, "Estimation of quality for software components: an empirical approach," *SIGSOFT Softw. Eng. Notes,* vol. 33, pp. 1-10, 2008.

[14] T. Kroeger and N. Davidson, "A Perspective-Based Model of Quality for Software Engineering Processes," in *Australian Software Engineering Conference.*, 2009, pp. 152-161.

[15] A. Kumar, P. S. Grover, and R. Kumar, "A quantitative evaluation of aspect-oriented software quality model (AOSQUAMO)," *SIGSOFT Softw. Eng. Notes,* vol. 34, pp. 1-9, 2009.

[16] C. E. Davis, "Evaluating computer programs for analyzing industrial power systems: what users need to know," in *Industry Applications Society 37th Annual Petroleum and Chemical Industry Conference.*, 1990, pp. 77-85.

[17] M. Torchiano, L. Jaccheri, C.-F. Sorensen, and A. I. Wang, "COTS products characterization," presented at the Proceedings of the 14th international conference on Software engineering and knowledge engineering, Ischia, Italy, 2002.

[18] F. Haiguang, "Modeling and Analysis for Educational Software Quality Hierarchy Triangle," in *Seventh International Conference on Web-based Learning.*, 2008, pp. 14-18.

[19] S. Khaddaj and G. Horgan, "A Proposed Adaptable Quality Model for Software Quality Assurance," *Journal of Computer Sciences,* vol. 1, pp. 482-487, 2005.

[20] J. J. E. Gaffney, "Metrics in software quality assurance," presented at the Proceedings of the ACM '81 conference, 1981.

[21] F. Haiguang, "Modeling and Analysis for Educational Software Quality Hierarchy Triangle," in *Web-based Learning, 2008. ICWL 2008. Seventh International Conference on*, 2008, pp. 14-18.

[22] G. Hart, "The software integrity of a computer system installed in a royal naval frigate," *Microelectronics Reliability,* vol. 22, pp. 1061-1066, 1982.

[23] E. Bertino, W. Lee, A. C. Squicciarini, and B. Thuraisingham, "End-to-end accountability in grid computing systems for coalition information sharing," presented at the Proceedings of the 4th annual workshop on Cyber security and information intelligence research: developing strategies to meet the cyber security and information intelligence challenges ahead, Oak Ridge, Tennessee, 2008.

[24] N. E. Fenton and S. L. Pfleeger, *Software Metrics: A Rigorous and Practical Approach*: PWS Publishing Co., 1998.

[25] M. Torchiano, L. Jaccheri, C.-F. S\, \#248, rensen, and A. I. Wang, "COTS products characterization," presented at the Proceedings of the 14th international conference on Software engineering and knowledge engineering, Ischia, Italy, 2002.

[26] M. Woodside, G. Franks, and D. C. Petriu, "The Future of Software Performance Engineering," in *FOSE '07 Future of Software Engineering.*, 2007, pp. 171-187.

[27] K. S. Jasmine and R. Vasantha, "Identification Of Software Performance Bottleneck Components In Reuse based Software Products With The Application Of Acquaintanceship Graphs," in *International Conference on Software Engineering Advances.*, 2007, pp. 34-34.

[28] D. E. Geetha, T. V. S. Kumar, and K. R. Kanth, "Predicting performance of software systems during feasibility study of software project management," in *6th International Conference on Information, Communications & Signal Processing.*, 2007, pp. 1-5.

[29] C. E. de Barros Paes and C. M. Hirata, "RUP Extension For the Software Performance," in *32nd Annual IEEE International Computer Software and Applications.*, 2008, pp. 732-738.

[30] C. Del Rosso, "The process of and the lessons learned from performance tuning of a product family software architecture for mobile phones," in *Eighth European Conference on Software Maintenance and Reengineering.*, 2004, pp. 270-275.

[31] K. Tokuno and S. Yamada, "Dynamic Performance Analysis for Software System Considering Real-Time Property in Case of NHPP Task Arrival," in *Second International Conference on Secure System Integration and Reliability Improvement.*, 2008, pp. 73-80.

[32] P. Eeles. (2005, 3/13/2010). *Capturing Architectural Requirements*. Available: http://www.ibm.com/developerworks/rational/library/4706.html

[33] A. AL-Badareen, B., M. Selamat, H., M. A.Jabar, J. Din, and S. Turaev, "Software Usability Factor within Software Life Cycle," in *The 2010 International Conference on Intelligent Network and Computing*, Kuala Lumpur, Malaysia, 2010, pp. 424-427.

[34] D. Yeats, "Revising documentation deliverables based on usability evaluation findings: a case study," presented at the Proceedings of the 22nd annual international conference on Design of communication: The engineering of quality documentation, Memphis, Tennessee, USA, 2004.

[35] N. J. C. Primus, "A generic framework for evaluating Adaptive Educational Hypermedia authoring systems," MSc, Business Information Systems University of Twente, Enschede, 2005.

[36] B. Behkamal, M. Kahani, and M. K. Akbari, "Customizing ISO 9126 quality model for evaluation of B2B applications," *Information and software technology,* vol. 51, pp. 599-609, 2009.