Evaluating Software Product Quality based on SQuaRE Series

Hidenori Nakai, Naohiko Tsuda, Kiyoshi Honda, Hironori Washizaki, and Yoshiaki Fukazawa
Dept. of Computer Science and Engineering, Waseda University, Tokyo, Japan
Email: {hide-and-seek, 821821}@toki.waseda.jp, khonda@ruri.waseda.jp, {washizaki, fukazawa}@waseda.jp

***Abstract*—Although the high quality of software is important for software stakeholders, quality of software products is often not effectively defined. Some quality models have been proposed, but these conventional models cannot measure and evaluate software product quality comprehensively. Moreover, quality measured and evaluated based on organization-specific quality models cannot be compared to the quality of other software products. To alleviate this problem, ISO/IEC defined international standards called the SQuaRE (Systems and software Quality Requirements and Evaluation) series for comprehensive quality measurement and evaluation; however, these standards include ambiguous measurements, making them difficult to apply. In this paper, we propose a comprehensive quality measurement framework, which includes a clear measurement plan based on ISO/IEC 25022 and 25023[1]. We confirmed the usefulness of our framework by conducting a case study of applying our framework to a commercial software product. As future work, we will introduce our framework to various domains. And then, we revise and refine measurements and evaluation plans to improve feasibility and usefulness.**

***Index Terms*—Software Quality Management, Quality Assurance, SQuaRE series**

I. Introduction

Software stakeholders (e.g. developers, managers, end users) require high quality of software products. Although several works have aimed to identify software quality such as [15] and [14], the quality of software products is not comprehensively, specifically, and effectively defined because previous approaches focused only on certain quality aspects. Therefore, software project stakeholders can not identify and understand other aspects of software quality.

Moreover, issues with definitions are obstacles to control and understand the quality of software products [18]. Since, the software product quality has subjective component [3], the evaluation results of quality metrics depend on software stakeholders.

On the other hand, the ISO/IEC has tried to define the evaluation methods for the quality of software products and has provided common standards, called the SQuaRE (Systems and software Quality Requirements and Evaluation) series. This series includes a comprehensive quality model, software product quality characteristics, and quality in use characteristics. Additionally, this series includes several metrics and measurements for each quality characteristic.

Measurement issues and ambiguities about the understanding limit the evaluation methods [6]. In particular, software quality managers struggle to define the quality of software products due to ambiguities in the evaluation methods. According to S. Wagner et al.[19], only 28% of companies apply the ISO/IEC standard to their software products. This is because the ISO/IEC standard has too general and ambiguous metrics, measurements, inputs, and outputs to apply practically to software development project and products [6] [2].

On the other hand, more than 70% of companies apply their own quality models [19]. Moreover there are various frameworks such as [21], [22], [23], [24] proposed for quality evaluation. However, non-standard organization-specific quality models and frameworks cannot be compared to others because they are often constructed with different standards and focus on only the quality characteristics of interest.

Therefore, we propose a comprehensive quality measurement framework that includes clear metrics and a measurement based on the latest standards ISO/IEC 25022:2016[8] and 25023:2016[9] in the SQuaRE series.

Contributions of this paper include:

- A comprehensive framework for quality measurements and evaluations based on ISO/IEC 25022 and ISO/IEC 25023
- An evaluation experiment of our framework using a case study

The remainder of this paper is organized as follows. Section II details our proposed framework. Section III shows a case study using our framework. Section IV describes related works. Finally, Section V concludes the paper.

II. Proposed Framework

The purposes of our framework are to develop a framework based on an international standard by establishing a comprehensive framework for all quality (sub-)characteristics of ISO/IEC 25022 and ISO/IEC 25023, reduce ambiguous metrics and measurements, and define the inputs and outputs for quality measurements clearly.

Using our framework, which was developed with input of the ISO working group members, project stakeholders can recognize how to measure their own software product quality, evaluate whether their software product has high/low quality based on an international standard, identify sufficient/insufficient quality (sub-)characteristics, determine weak qualities compared to other software products, and develop

[1]This paper is an extended version of a poster "Initial Framework for Software Quality Evaluation based on ISO/IEC 25022 and ISO/IEC 25023"[16] presented at The 2016 IEEE International Conference on Software Quality, Reliability & Security (QRS 2016).

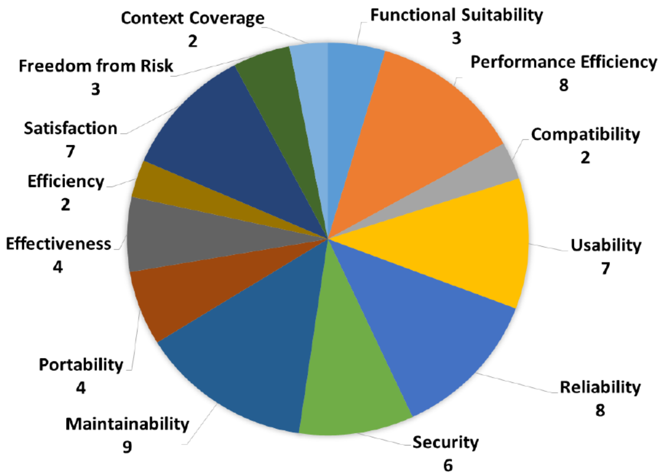

Fig. 1. Number of metrics for each quality characteristic

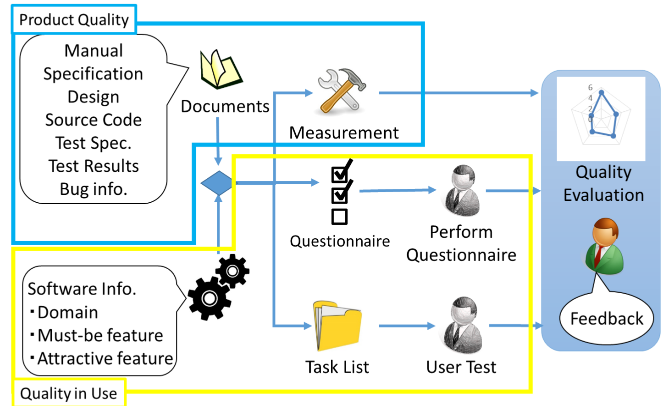

Fig. 2. Framework overview

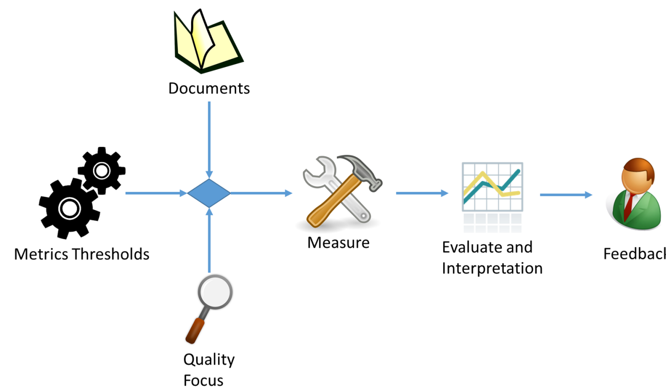

Fig. 3. Procedure for product quality

an objective interpretation. The results of a quality evaluation based on our framework help project stakeholders identify areas for improvement.

Our framework consists of two parts: ”Product Quality” and ”Quality in Use”. The former contains internal and external product quality characteristics, metrics, and measurements based on ISO/IEC 25023, whereas the latter has quality characteristics, metrics, and measurement of quality in use based on ISO/IEC 25022. Product quality influences quality in use. That is, quality in use depends on the product quality. Thus, ”Product Quality” measures and ”Quality in Use” measures are connected. Therefore, if either product quality or quality in use is absent, the software quality is insufficient.

In our framework, there are 47 product metrics and 18 metrics of quality in use. Figure 1 shows the number of metrics of each quality characteristic. These metrics cover 51% of the ISO/IEC metrics.

The overview of procedure to use our framework is shown in Figure 2. To measure and evaluate the product quality, our framework requires some information such as manual, specifications, test specifications, and bug information. To measure and evaluate quality in use metrics, information should be collected and evaluated using a questionnaire and a user test. Based on the results, software quality is assessed, identifying what quality characteristics are sufficient/insufficient from the viewpoint from the international standards.

A. *Product Quality*

The product quality indicates the degree of how the required needs (e.g. software purpose, performance, usability of product, and easy maintainability) are satisfied. If this quality is insufficient, the software product may include incidents, high costs for development or maintenance, and violations of user needs. Therefore, product quality should be identified.

The product quality part involves internal/external quality characteristics and sub-characteristics (e.g., the quality characteristic is Functional Suitability, and one of its sub-characteristic is functional completeness), metrics, and measurements based on ISO/IEC 25023. There are 47 product quality metrics. A part of these metrics focuses on the main functions of the software. The main function means that must-have functions and the functions described in catalogs.

To measure and evaluate the product quality, our framework requires information from some elements: manuals, specifications, design, source code, violations of the coding standard, test specifications, test results, and bug information. In addition, thresholds are needed for an objective quality interpretation. These thresholds are defined based on metrics information from many domain software products. However, in this research, the initial thresholds are defined based on the prediction approach [7] and conventional work. The overview of procedure to measure product quality is shown in Figure 3.

The following steps are used to measure the product quality.

1) Define the quality to be considered by the project stakeholders.
2) Define the thresholds for the quality metrics in my framework.
3) List the information for the measurement based on select documents (e.g., manual, specifications, etc.).
4) Measure and evaluate the quality metrics based on

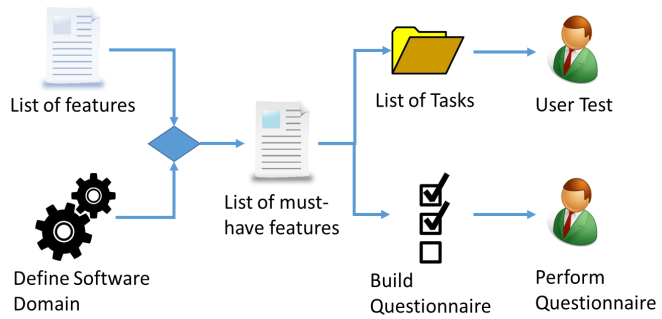

Fig. 4. Procedure for quality in use

thresholds.

For example, "Functional Suitability" indicates whether the software functions satisfy user needs. The lack of functional suitability means that the software does not perform as the user intends (i.e., this software cannot satisfy the user needs). One measure of the characteristic is as follows:

$$
\begin{aligned}
X &= 1 - A/B \\
A &= Number\ of\ functional\ requirements \\
  &\quad not\ implemented \\
B &= Number\ of\ functional\ requirements
\end{aligned}
$$

B. Quality in Use

Quality in use indicates the degree that a software product can satisfy a specific user needs, effectively, efficiently, and satisfactorily to achieve a user's goals and mitigate risks in the context of use. If this quality is insufficient, users tend to be dissatisfied with a software product, and the number of users may decrease. Therefore, quality in use should be identified.

Quality in use involves some quality characteristics and their sub-characteristics (e.g., the quality characteristic is Satisfaction, and one of its sub-characteristic is usefulness), and a measurement plan based on ISO/ IEC 25022. There are 18 metrics for quality in use.

To measure and evaluate the product quality, our framework requires some information from experiments: a user test and a questionnaire. The user test evaluates the effectiveness, efficiency, and satisfaction. The questionnaire is related to satisfaction, freedom from risk, and context coverage. It should be noted that the questionnaire is developed according to [10] [25], and popular usability measurement scales, SUS [4], SUMI [11], and NPS [17].

The overview of procedure to measure quality in use is shown in Figure 4.

The following steps are used to measure the quality in use.

1) Define the software product domain by vendor or third organization to evaluate the software.
2) Define must-have features in the software domain.
3) Create normal/abnormal tasks based on must-have features for user test. In addition, build the questionnaire considering must-have features.
4) Prepare a test environment based on desired system requirements. In addition, distribute the questionnaires to the actual users.
5) Perform a user test and questionnaire.

III. Case Study

To confirm the usefulness of our framework, we applied our framework to a commercial software product.

A. Design and Result

In the case study, 30 product metrics and six quality in use metrics were measured. All of these metrics can be measured, suggesting that project stakeholders can adapt these metrics and measurements to their own projects.

Additionally, we performed a user test. The list of user test tasks was developed by its vendor based on their scenario test. In addition, we developed abnormal tasks based on the task list. The subjects of the user test are several students belonging to our laboratory. Because these subjects are not the target of the software, developers in the vendor helped them perform user test tasks.

It took 4-6 hours to measure the product metrics and another 2-4 hours to complete the user test. Regarding the metrics, measurement results of most metrics reached 100%, indicating that the quality of the target product is fairly good. In the user test, all tasks were completed; the results reveal several problems, such as "There may be some potential bugs.".

B. Assessment

The vendor of the target product assessed the evaluation results objectively. The details of the vendor assessment include:

Internal Quality The way that the target of the number of bugs is defined should be revised and refined.

External Quality Measurement results indicate useful suggestions for improving quality, such as possible refinement of product testing process.

Quality in Use The result of the user test can help improve the quality of software products and user satisfaction.

C. Discussion

Our framework collects metrics information based on documents such as specifications, test designs, and manuals. However, the format of these documents depends on the organization or project. Thus, in the case study, some metrics information is not collected because there is no information for the metric.

Our framework may be time consuming for project stakeholders to implement. It is difficult for project stakeholders to introduce all the metrics and measurements defined in our framework due to time limitations. Therefore, some metrics and measurements should be revised to improve the feasibility of evaluating quality.

IV. Related Work

AENOR provides ISO 25000 Software Product Quality Certification [1] based on the SQuaRE series. The certification evaluates maintainability and functional suitability based on the results of functional tests, source code, and a third party library. On the other hand, our research not only examine maintainability and functional suitability but also investigate other quality characteristics of the SQuaRE series.

In the project Quamoco, a quality meta model was developed for specific operationalized quality models [20]. Because Quamoco is used to create an appropriate and introduce-able quality model based on meta model, it cannot be used to compare to other software product's quality.

[13] presented a scheme to identify a suitable quality model based on the existing quality model's purpose (e.g., quality specification, quality measure, monitoring quality, and quality improvement), object (e.g., product, process, resource), and quality focus (general or specific). However, unlike our framework, this scheme cannot be used for a quality measurement.

To identify the software quality, some quality models, quality measurement methods, and metrics have been defined, such as COQUALMO [5] and HDCE [12]. However, these models/approaches have only rely on a specific quality focus and require subjective expert judgments.

V. Conclusions and Future Works

This research strives to develop a framework based on an international standard, propose a comprehensive framework for all quality characteristics. To evaluate the product quality, we defined 47 quality metrics and 18 quality in use metrics, and their clear measurements based on documents, user test, and questionnaire. Our contributions are (1) defining a framework for quality measurements and evaluations based on ISO/IEC 25022 and ISO/IEC 25023, (2) establishing a procedure of using our framework to evaluate the software quality, (3) incorporating feasible metrics and measurements into the proposed framework, and (4) demonstrating the effectiveness of our framework for project stakeholders through the case study.

The case study is performed for document management systems with an emphasis on a part of quality characteristics. To measure and evaluate these quality, metrics information is collected through documents, vendor input, and a user test. Although the vendor indicated that the evaluation results based on our framework is very useful, some metrics and measurements may be unnecessary for other software domains. Moreover, a lot of time is needed to measure and evaluate metrics and quality. Thus, our framework might not have enough feasibility from the viewpoint of time cost.

As future work, we will introduce our framework to various domains. And then, we revise and refine measurements and evaluation plans to improve feasibility and usefulness. Additionally, we will build the GQM model to combine clearly the quality characteristics and metrics to clear interpretation of software quality. Then, we define relationships between metrics and characteristics obviously, and verify the validation of these relationships thorough some case studies.

Acknowledgments

This work has been conducted as a part of "Research Initiative on Advanced Software Engineering in 2015" supported by Software Reliability Enhancement Center (SEC), Information Technology Promotion Agency Japan (IPA).

References

[1] Aenor. ISO 25000 software product quality certification. http://www.en.aenor.es/aenor/certificacion/calidad/calidad_software_25000.asp.

[2] A. Abran et al. Usability meanings and interpretations in ISO standards. *Software Quality Journal*, 11(4):325–338, Nov. 2003.

[3] H. Al-Kilidar et al. The use and usefulness of the ISO/IEC 9126 quality standard. In *Empirical Software Engineering, 2005. 2005 International Symposium on*, pages 7 pp.–, Nov 2005.

[4] J. Brooke. SUS-A quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[5] S. Chulani et al. Modeling software defect introduction and removal: COQUALMO. Technical report, USC-CSSE, 1999.

[6] J. Heidrich et al. Model-based quality management of software development projects. In *Software Project Management in a Changing World*, pages 125–156. Springer, 2014.

[7] K. Honda et al. A generalized software reliability model considering uncertainty and dynamics in development. In *Product-Focused Software Process Improvement*, pages 342–346. Springer, 2013.

[8] ISO/IEC. *ISO/IEC 25022:2016 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of quality in use*. 2015.

[9] ISO/IEC. *ISO/IEC 25023:2016 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of system and software product quality*. 2015.

[10] J.-Y. Jian et al. Foundations for an empirically determined scale of trust in automated systems. *International Journal of Cognitive Ergonomics*, 4(1):53–71, 2000.

[11] J. Kirakowski et al. SUMI: the software usability measurement inventory. *British Journal of Educational Technology*, 24(3):210–212, 1993.

[12] M. Kläs et al. Managing software quality through a hybrid defect content and effectiveness model. In *ESEM '08*, pages 321–323. ACM, 2008.

[13] M. Klas et al. CQML scheme: A classification scheme for comprehensive quality model landscapes. In *Software Engineering and Advanced Applications, 2009. SEAA '09. 35th Euromicro Conference on*, pages 243–250, Aug 2009.

[14] J. Münch et al. Software project control centers: concepts and approaches. *Journal of Systems and Software*, 70(1):3–19, 2004.

[15] H. Nakai et al. Continuous product-focused project monitoring with trend patterns and GQM. In *APSEC '14*, volume 2, pages 69–74, 2014.

[16] H. Nakai et al. Initial framework for software quality evaluation based on iso/iec 25022 and iso/iec 25023. In *QRS, Poster*, 2016.

[17] F. F. Reichheld. The one number you need to grow. *Harvard business review*, 81(12):46–55, 2003.

[18] A. Trendowicz et al. Model-based product quality evaluation with multi-criteria decision analysis. *CoRR*, abs/1401.1913, 2014.

[19] S. Wagner et al. Software quality models in practice - survey results-. https://mediatum.ub.tum.de/doc/1110601/1110601.pdf, 2010.

[20] S. Wagner et al. The quamoco product quality modelling and assessment approach. In *ICSE '12*, pages 1133–1142, 2012.

[21] H. Washizaki et al. Experiments on quality evaluation of embedded software in japan robot software design contest. In *ICSE*, pages 551–560, 2006.

[22] H. Washizaki et al. A framework for measuring and evaluating program source code quality. In *PROFES*, pages 284–299, 2007.

[23] H. Washizaki et al. A metrics suite for measuring quality characteristics of javabeans components. In *PROFES*, pages 45–60, 2008.

[24] H. Washizaki et al. Reusability metrics for program source code written in C language and their evaluation. In *PROFES*, pages 89–103, 2012.

[25] D. Watson et al. Development and validation of brief measures of positive and negative affect: the PANAS scales. *Journal of personality and social psychology*, 54(6):1063, 1988.