Test management based on quality characteristics

Daiju Kato
Nihon Knowledge Co., Ltd.
Tokyo, Japan
d-kato@know-net.co.jp

Ayako Okuyama
Nihon Knowledge Co., Ltd.
Tokyo, Japan
a-okuyama@know-net.co.jp

Hiroshi Ishikawa
Tokyo Metropolitan University
Tokyo, Japan
ishikawa-hiroshi@tmu.ac.jp

***Abstract*—By using SQuaRE in product development projects, one can achieve quality traceability based on quality characteristics. Conveying the quality of the software product to stakeholders can be done objectively. This paper introduces quality characteristics and to the use of SQuaRE, with explanation of quality requirement definitions and evaluation methods using quality characteristics.**

Keywords—Evaluation process, Quality characteristics, SQuaRE, Test management, Traceability

I. Reason for using Quality Characteristics

We have investigated ways for stakeholders or our products' customers to understand software quality objectively. To date, many companies have used their own quality data to explain the quality of deliverables. For example, the IT Development White Paper published by the Information Technology Promotion Agency (IPA) of Japan provides survey results elucidating the bug density of projects in various industries [1]. However, bug density metrics often vary greatly depending on development languages, coding rules, and development processes. It is difficult to ascertain whether the quality of products is good or bad simply using numerical values alone.

Therefore, as a method to conveying the software product quality to anyone using an easy-to-understand methodology, we have examined the use of Systems and Software Quality Requirements and Evaluation (SQuaRE) [2] to represent understandable quality objectively.

II. Combination of International Standards in the Development Process

Using SQuaRE in a software development project is insufficient for simple description of the quality requirements as quality characteristics. An evaluation result must indicate that the quality requirements classified by the quality characteristics are satisfied. To take advantage of the quality characteristics defined in the ISO/IEC 25010 quality model [3], relevant international standards were identified. The related standards were identified during the development project, as shown in Fig. 1.

By assigning a quality characteristic that can be guaranteed by a corresponding test types to a test type that satisfies each quality requirement, the same quality characteristic can be assigned to test viewpoints constituting the test types. At this time, in the case of a functional test, several quality characteristics such as functional suitability and functional accuracy are assigned to the test type, but a single quality sub-characteristic is assigned to the test viewpoint. Test cases derived from the test perspective will also be mapped to a single-quality sub-characteristic. When the created test case is executed and an incident is found, the incident has the quality sub-characteristic assigned to the executed test case as an attribute. If quality characteristics are

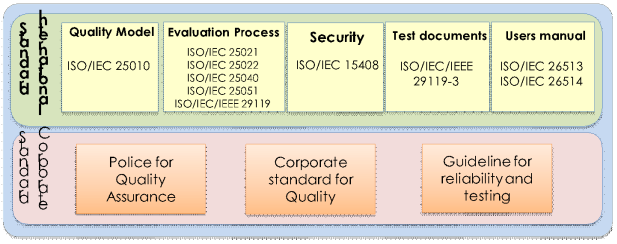

Fig. 1. International standards used in the development process

used in this way as traceability of test cases and incidents from quality requirements, the number of test viewpoints, the number of test cases, and the number of bugs can be aggregated for each quality characteristic to ascertain which quality characteristic is problematic. In other words, quality characteristics are useful as an indicator of traceability, Fig. 2.

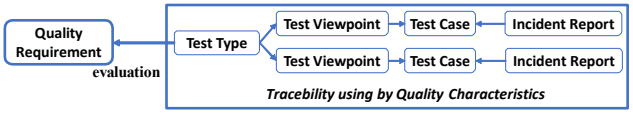

Fig. 2. Quality characteristics as an indicator of traceability.

III. Quality Requirements Categorized by Quality Characteristics

To use quality characteristics as an indicator for the quality of software under development project, one must define quality requirements for each quality characteristic. Instead of simply writing quality requirement definitions for each quality sub-characteristic, it is better to refer to quality requirements included in ISO/IEC 25051 [4].

According to our developed rules, when writing quality requirements for product quality, the function or operation of the software is written as the subject, whereas, when writing quality requirements for quality in use, the definition is written with the users as the subject. Product quality is the quality possessed by the product. Quality in use is the quality that people feel. By dividing the subject clearly, one can be aware of differences in quality. Only the quality required for the target software needs to be defined for both product quality and quality in use, and quality requirements are not defined for all quality sub-characteristics.

Once the definition of the quality requirement is presented, the test type which can evaluate the quality requirement is mapped. If possible, we recommend identification of the required quality characteristics at each test level or sprint and recommend mapping of test types that meet the test level or the sprint quality.

TABLE I. QUALITY REQUIREMENTS CLASSIFIED BY QUALITY CHARACTERISTIC

Characteristic	Sub-Characteristics	Target Quality
Functional suitability	Functional completeness	• Development requirements of new features have been verified. • Migration from past versions is possible; compatibility is collateral.
	Functional correctness	• Results of new functional requirements are correct. They have been verified. • Each product definition file developed by the past version functioned properly. • All functions under English and Chinese environments have the same operation quality as in the Japanese environment. • Scenario test results present no difficulty under the estimated users' operations.
Reliability	Maturity	• Each test level has an analysis action for test coverage and turn-around time to fix bugs. • Each test level has a quality improvement action derived from quality analysis of the previous test level. • All bugs verified at the RC stage.
	Fault tolerance	• Operations have perfect continuity even though one node is stopped under a clustering environment. • Migration programs continue running when some definition files have some errors.
	Recoverability	• Prior versions of definition files can be saved even though the files include some errors.
Performance efficiency	Time behavior	• Performance is less than or equal to a maximum 3% of the prior version. • Concurrent multi-access testing is done with no error. • There is no high CPU load or I/O load condition under some functional operations.
	Resource utilization	• No memory leak occurs under usual operations. • Memory and I/O resources are used effectively.

IV. EVALUATION PROCESS

In the test design of each test type, the test design is done considering the assigned quality characteristics. As described above, the test viewpoint allows a single quality sub-characteristic to be assigned. We aggregate all test viewpoints for each quality characteristic so that the test viewpoint regarding functional compatibility is less than 70%. This is true because the test is not only a functional test; it also assesses the completeness of quality.

Because the created test case also has an inherent quality sub-characteristic as an attribute, the bug found in the execution of the test case is mapped to the same quality sub-characteristic as the test case. Therefore, our bug management system prepares quality sub-characteristic items for each incident, and uses the quality characteristics in bug analysis. If many bugs exist for a particular quality trait, increased testing related to that quality trait can be done to enhance the review issued by the development team.

V. CONCLUSION

Using quality characteristics as an indicator of software quality of the final product of a project not only enables traceability by quality characteristics. It also makes it easier to analyze quality objectively and to balance the software quality with other quality characteristics.

By classifying bugs found after a product release and inquiries received from the support center based on quality characteristics, one can analyze the evaluation work validity during development and can build a feedback process.

To realize software development that uses quality characteristics, all project members must have knowledge of quality characteristics. However, considering that international standards are useful and that objective quality information can be provided to stakeholders and customers, quality characteristics are used. We believe that using quality characteristics can provide excellent results. We are investigating more effective use of SQuaRE.

REFERENCES

[1] "Software Development Data White Paper 2018-2019", Information-Technology Promotion Agency (IPA), Japan, ISBN 978-4-905318-64-4, 2018, pp. 283-285.

[2] ISO/IEC 25000:2014, "Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE", March, 2014.

[3] ISO/IEC 25010, Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models, March 2011.

[4] ISO/IEC 25051, Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, February 2014.

[5] D. Kato, H. Ishikawa, "Develop Quality Characteristics based quaity evaluation process for ready to use software products", JSE-2016, February, 2016.

[6] D. Kato, A. Okuyama, H. Ishikawa, "Use proactive evaluation process for 'Quality in Use'", Seventh World Congress for Software Quality", Seventh World Congress for Software Quality, March, 2017.

[7] D. Kato, H. Ishikawa, "Development of quality assurance process for Ready to Use Software Product with using JIS X 25051:2016 and benefit of the process", Software Quality Symposium, September 2016, in Japanese.

[8] D. Kato, H. Ishikawa, "Feedback process of inquiry analysis conscious of recurring business", Software Quality Symposium, September 2016, in Japanese.