Retrofitting Quality

Raul Martinez
Subcomité de Calidad en Tecnología de la Información

IRAM

Buenos Aires, Argentina
rmartinez582@gmail.com

***Abstract*—This short paper proposes a simple and practical way to improve quality while system/software product is in production, combining theoretical points of view of quality models and information extracted from the operation, change requests logs and user feedback like incidents, required/requested modifications, and detected reactions from users. Such information can be used to enhance the existing models, or to create a new one, and therefore to improve the quality of the system/software products.**

SQuaRE [1][2] framework, which models quality as a set of quality characteristics valuable for users, is a widely accepted technique and will be used as reference in this paper.

Keywords— SQuaRE, Quality Models, Quality in Production Environments, Quality Maintenance

I. Introduction

The goal of SQuaRE quality modelling process is to represent quality before the system/product exists, as a set of characteristics and sub characteristics to which users can assign importance and value, providing helpful clues for development prioritization, requirements trade-off and monitoring.

In this proposal, modelling, originally a mental process, is enhanced with the actual data obtained from the execution of the system/product in the production environment.

Citing Lehman [6], E-type software systems that solve a problem or implement a computer application in the real world will be perceived as of declining quality unless rigorously maintained and adapted to a changing operational environment.

The aim of the proposed basic process is the application of SQuaRE quality models as an ordered reference framework to organize and optimize this job of preserving the system/product good quality perception of the user.

II. Potential Sources of Data for Elicitation of Quality Problems

There are many different situations during maintenance of a system/software product where the quality could be compromised. Two common sources of data for quality maintenance opportunities are proposed, but the idea could be easily extended to other sources.

A. *Corrective maintenance*

During normal execution of the system/software product incidents are reported by user community. Following, some common instances of this problem are enumerated.

- Errors classified as functional by the maintenance organization; these errors could hide suitability, correctness, appropriateness problems.
- Security problems.
- Poor system/software product performance.
- User operation errors reported to helpdesks could be signals of usability problems or learnability problems.

Error statistics of specific software modules could be used as a source of internal/external quality problems. Products components with more defects usually spot quality problems.

B. *Enhancements*

- Pure functional enhancements containing new quality characteristics or modification to existent products.
- New requirements related with improvements of the quality of the product.
- Communities' boards, helpdesk, and other user-developer communicational and organizational tools that can be used for quality enhancement discovery.

III. Classification Process

The following basic process is proposed to utilize the SQuaRE model as a categorization framework for quality maintenance incidents.

A. *Classification*

Most of maintenance requirements are classified into two categories: incidents and enhancements, but others could exist.

Each incident / enhancement can be mapped to a characteristic / sub-characteristic / measures of the SQuaRE model, allowing a clear classification of the quality problem.

- Incidents

Incidents imply that some user expectations are not being met and should be included, or result in unexpected values and should be corrected.

Incidents will be analysed to detect quality events and characteristics affected. These characteristics could be related to Data Quality, Product Quality or Quality in Use models.

A detected quality incident could affect a characteristic, sub-characteristic, or measure. Response should be analysed, a trade-off with other needs solved and, if necessary, the model or models updated, and changes reflected on the system/software product.

- Enhancements

Enhancements and new requirements will be analysed looking for quality improvements.

Existing characteristics and/or sub-characteristics / measures could be affected, or new ones added. These characteristics could be related to Data Quality, Product Quality or Quality in Use models.

This situation implies that users believe that certain capacities will be useful if included.

The required quality should be analysed and a trade-off with other needs solved and, if necessary, the model or models updated.

B. Correction and improvement of the model

The previous classification could uncover failures of the system/software quality model if an explicit model exists.

Some common situations can be:

- The incident / enhancement cannot be categorized in the model because the characteristic was not considered in the original model of the project but exist in SQuaRE. This is an important case; it means that certain characteristic that user expects is not offered or is offered with insufficient or less than the expected performance in the product. Characteristic is evaluated and added to the model.
- The incident/enhancement can be categorized within a characteristic, but no sub-characteristic covers the incident/enhancement. In this case, characteristics have been detected during modelling process but certain sub-characteristics, valuable to the user, are missing.
  Example: Interaction Capability detected but Learnability not considered. Sub-characteristic is evaluated and added to the model.
- The incident/enhancement can be fully classified because the characteristic / sub-characteristic exists, but actual measures are out of range from the proposed on the model. Values obtained for the measures differ from users' expected values.
  Time behaviour values are classical examples of this situation. Target values are evaluated and changed in the model.

IV. Expected Outputs

A. Primary result

An updated model with new and/or revised characteristics / sub-characteristics / measures, illustrating user requirements for quality.
This updated model and its correspondent implementation on the system/software product will be useful for diminishing the perception of declining quality.

V. Special situations

A. Reflections of requirements on the existent models

Not necessarily incidents and enhancements reflect directly on a unique characteristic / sub-characteristic / measure of the existent or generic SQuaRE model.
New enhancements could include different characteristic / sub-characteristic of the standard model and an incident or enhancement might be reflected in more than one characteristic / sub-characteristic in Data Quality, Product Quality or Quality in Use models.

B. Explicit model does not exist

In this case generic models proposed by SQuaRE [1] [2] [3] [4] [5], Data Quality, Product Quality, Quality in Use could be used as a metamodel to create the specific instances of quality models for system/software product project, enabling a more precise communication about quality between user and developer organization.

VI. Conclusion

Returning to Lehman's reference, system/software product evolution is intrinsic to software, not necessarily a developer's fault.
SQuaRE models can be used as a practical tool to reflect those evolving quality needs of users in an ordered and visible way.
These models can be created or updated/enhanced along the whole system/product life cycle, maintaining visibility of quality enclosed in the system/software product and preserving as previously mentioned the system/product good quality perception of the user.

References

[1] ISO/IEC 25000:2014 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE

[2] ISO/IEC 25010:2011. ISO/IEC 25010:2011 Systems and Software engineering System and software quality models

[3] SO/IEC 25023:2016 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality

[4] ISO/IEC 25022:2016 Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Measurement of quality in use

[5] ISO/IEC 25024:2015 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of data quality

[6] M. M. Lehman, Laws of Software Evolution Revisited, EWSPT '96, 1996