Development of a software quality assessment method

Irakli Basheleishvili, Giorgi Kapanadze and Avtandil Bardavelidze

Akaki Tsereteli State University, Tamar Mepe St #59, Kutaisi, 4600, Georgia

Abstract

The paper is about the development of a software quality assessment method, which is based on the FCM (Factor-Criteria-Metrics) software quality assessment model, ISO / IEC 25010 software quality standard and multi-criteria decision analysis method. The method presented in the paper allows us to obtain a quantitative assessment of quality indicators using the ISO / IEC 25010 standard when assessing software quality.

Keywords

Software, quality, assessment, FCM model

1. Introduction

The rapid development of information technologies is increasing the demand for their use in almost all spheres of human activity, moreover, at present it is inconceivable for them to function effectively without the use of modern information technologies. The rapid development of information technologies and computing processes in recent decades has led to the existence of software in all areas of human activity [1]. Therefore increasing the demand for software quality. Software quality is the combination of the characteristics of a computer software product and their meanings that relate to the ability to use it to meet established or expected requirements [2]. Quality in software means no errors in it [3]. Software errors can cause great material damage, so research into software quality management is very important today.

Based on the above, the aim of our paper is to develop a software quality assessment method based on the FCM (Factor-Criteria-Metrics) software quality assessment model, the ISO / IEC 25010 software quality standard, and the multi-criteria decision analysis method.

2. Standard and model of software quality

There are many standards in software quality evaluation that determine the factors (characteristics) of software quality evaluation. Quality characteristics reflect the different qualities exhibited by self-promotional software, ISO / IEC 25010 is a software quality standard, it refers to the quality of the software application as a "product", as well as the quality of its creation processes. The product quality model specified in ISO / IEC 25010 includes the following quality characteristics[1,3,11]:

- Functional Suitability - Functional Suitability refers to how well a product or system is able to provide functions that meet the stated and implied needs.
- Reliability - Reliability refers to how well a system, product, or component performs specified functions under specified conditions[11].
- Performance Efficiency - This characteristic represents the performance relative to the amount of resources used under stated conditions.

IVUS 2022: 27th International Conference on Information Technology
EMAIL: irakli.basheleishvili@atsu.edu.ge (I. Basheleishvili); kapanadze.giorgi2@atsu.edu.ge (G. Kapanadze); avtandil.bardavelidze@atsu.edu.ge (A. Bardavelidze)
ORCID: 0000-0002-4429-7577 (I. Basheleishvili); 0000-0002-9873-4402 (A. Bardavelidze)

- Useability - Usability refers to how well a product or system can be used to achieve specified goals effectively, efficiently, and satisfactorily.
- Security - Security refers to how well a product or system protects information and data from security vulnerabilities[3,11].
- Compatibility - Compatibility refers to how well a product, system, or component can exchange information as well as perform its required functions while sharing the same hardware or software environment[10,11].
- Maintainability - Maintainability refers to how well a product or system can be modified to improve, correct, or adapt to changes in the environment as well as requirements.
- Portability - Portability refers to how well a system, product, or component can be transferred from one environment to another.

These features can be divided into two categories: functional features which are the main features of software operation and non-functional features that characterize the behavior of the software product during daily use. Non-functional features related to reliability, usability and efficiency should be classified as requirements that a software product, must meet in the course of its operation. To determine that a software product meets different quality characteristics, there are different models and methods of processes [1,2,3,5]. Some models can be attributed (nearest) to the concept of process quality. This means that the high-quality process of creating a product creates a high-quality product. That is why special attention is paid to the processes [3,6,10].

The FCM model is a general model for evaluating software quality, in the model the software termination factor is one factor, one factor is the decisive criterion, several criteria are decided by some metrics.

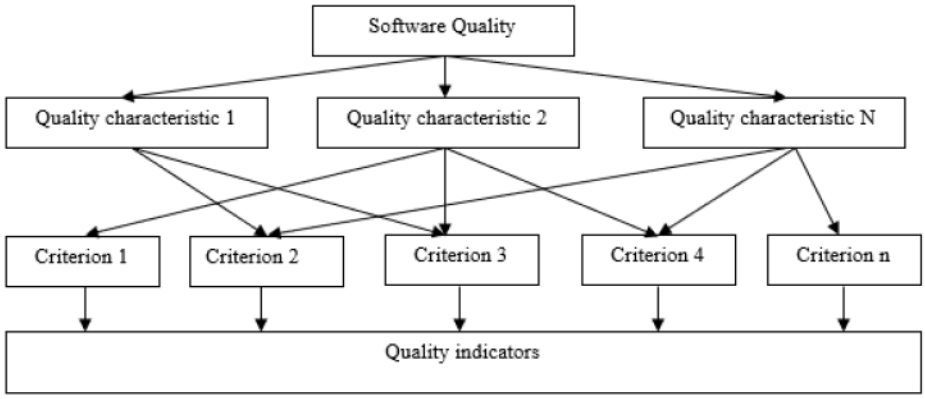

Figure 1: FCM model [3,11]

3. Methodology

If we look at the software quality evaluation model presented above, we will see that its structure is similar to the structure of a multi-criteria decision analysis task. Because individual quality characteristics are characterized by different criteria, compile a decision matrix for a particular quality characteristic (consisting of alternatives (quality characteristic) and evaluation criteria for alternatives). Based on which the multi-criteria decision analysis TOPSIS method [6,7,8] can be used to determine the quantitative evaluation rate for the quality characteristic (in the range 0 -1).

$$\begin{array}{cccccc} & C_1 & C_1 & \cdots & C_n \\ q_1 & x_{11} & x_{12} & \cdots & x_{1n} \\ q_2 & x_{21} & x_{22} & \cdots & x_{2n} \\ q_3 & x_{31} & x_{32} & \cdots & x_{3n} \end{array} \quad (1)$$

In this case, the matrix consists of one real alternative (quality characteristic - q1) and two formal alternatives (q2 - with the maximum possible values according to the criteria, and q3 - with the minimum values). For the real alternative, the x1j values are determined using numbers obtained from metrics and using various types of testing methods (which must be represented on a 100-point scale).

Once we have defined the decision matrix, the following steps need to be performed:

Stage 1. Determine the weight vector $W = [w_1, w_2, \dots, w_n]$ for the evaluation criteria, which must satisfy the following condition:

$$\sum_{i=1}^{n} w_i = 1 \quad (2)$$

Stage 2. Normalize the decision matrix using the following formula:

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}} \quad (3)$$

Stage 3. Compute a weighted normalized decision matrix according to the following formula:

$$V_{ij} = w_j * r_{ij} \quad (4)$$

Stage 4. Define positive ideal and negative ideal solutions:

Positive Ideal Decision:

$$S^+ = \{v_1^+, v_2^+, \dots, v_n^+\} \quad (5)$$

Where:

$$v_j^+ = \max(v_{ij})$$

Negative ideal solution:

$$S^- = \{v_1^-, v_2^-, \dots, v_n^-\} \quad (6)$$

$$v_j^- = \min(v_{ij})$$

Stage 5. Calculate the distance to the ideal positive and ideal negative solution for the q_1 alternative:

$$d_1^+ = \sqrt{\sum_{j=1}^{n} (v_{1j} - v_j^*)^2} \quad (7)$$

$$d_1^- = \sqrt{\sum_{j=1}^{n} (v_{1j} - v_j')^2} \quad (8)$$

Stage 6. Calculate the alternative closest to the ideal solution. Which is calculated by the following formula:

$$R_1 = \frac{d_1^-}{d_1^- + d_1^+} \quad (9)$$

Where:

$$0 \le R_1 \le 1$$

In order to evaluate all quality characteristics according to ISO / IEC 25010 standard, it is necessary to compile a decision matrix for individual characteristics and perform the above steps based on it.

4. Practical example

To illustrate the use of the method presented in the work, consider a practical case, for example, for specific software, we want to evaluate the quality characteristic reliability, the evaluation criteria of which are: accuracy, consistency and completeness. First define the decision matrix represented below:

Table 1
Decision matrix

	accuracy	consistency	completeness
q_1(releability)	87	97	78
q_2	100	100	100
q_3	0	0	0

As in the decided matrix, q_1 is the real characteristic (in our case reliability), while q_2 and q_3 is the formal alternative (q_2 is evaluated according to the criteria with more significant values, and q_3 - with minimal values). q_1 Alternatively use sampling metrics using numbers and other types of testing methods.

Once we have defined the decision matrix, we must weigh the weights for the evaluation criteria, in this case we take equal weights for the three criteria (w_1 = 0.33333333333, w_2 = 0.33333333333, w_3 = 0.33333333333).

Normalize the decision matrix and define a weighted normalized matrix that looks like this:

Table 2
Weighted normalized matrix

	accuracy	consistency	completeness
q_1(releability)	0.22	0.23	0.21
q_2	0.25	0.24	0.26
q_3	0	0	0

Define positive ideal and negative ideal solutions as follows:

positive ideal solution - [0.25, 0.24, 0.26]

negative ideal solution - [0, 0, 0]

Calculate the distance to the ideal positive and ideal negative solution for the q_1 alternative, in our case the distance to the ideal solution is 0.06681, and the distance to the negative ideal solution is 0.37916. Using this to calculate the coefficient of proximity to the ideal solution for the q_1 alternative, which is equal to 0.850187. This is the target value for us by the proximity of which we determine the level of evaluation of the quality characteristic.

5. Conclusion

The work presents a software quality assessment method based on the FCM (Factor-Criteria-Metrics) software quality assessment model, ISO / IEC 25010 software quality standard and multi-criteria solution analysis method, allows us to evaluate software quality features, which is an important tool for the software quality management process, which aims to develop and manage software quality so that the product meets the quality standards required by the user as much as possible.

6. References

[1] Galin, D. (2018). Software quality: concepts and practice. John Wiley & Sons.

[2] Dalla Palma, S., Di Nucci, D., Palomba, F., & Tamburri, D. A. (2020). Toward a catalog of software quality metrics for infrastructure code. Journal of Systems and Software, 170, 110726.

[3] G. Chogovadze, G. Surguladze, M. Gulitashvili and S. Dolidze(2020), Software application quality management: testing and optimization, Georgian Technical University. pp. 366.

[4] Desyatirikova, E. N., Belousov, V. E., Zolotarev, V. N., & Lavlinskaia, O. Y. (2017, September). Design process of software quality management. In 2017 International Conference" Quality Management, Transport and Information Security, Information Technologies"(IT&QM&IS) (pp. 496-499). IEEE.

[5] Hovorushchenko, T. (2018). Methodology of evaluating the sufficiency of information for software quality assessment according to ISO 25010. Journal of information and organizational sciences, 42(1), 63-85.

[6] Pamučar, D. S., Božanić, D., & Randelović, A. (2017). Multi-criteria decision making: An example of sensitivity analysis. Serbian journal of management, 12(1), 1-27.

[7] Basheleishvili, I. (2020). Developing the expert decision-making algorithm using the methods of multi-criteria analysis. Cybernetics and Information Technologies, 20(2), 22-29.

[8] I. Basheleishvili, S. Tsiramua (2019), Development of Method of Multifunctional Personnel Assessment Using a Topsis Method. Journal of Technical Science and Technologies, 7(1), 31-36.

[9] Mädler, J., Viedt, I., & Urbas, L. (2021). Applying quality assurance concepts from software development to simulation model assessment in smart equipment. In Computer Aided Chemical Engineering (Vol. 50, pp. 813-818). Elsevier.

[10] Estdale, J., & Georgiadou, E. (2018, September). Applying the ISO/IEC 25010 quality models to software product. In European Conference on Software Process Improvement (pp. 492-503). Springer, Cham.

[11] https://iso25000.com/index.php/en/iso-25000-standards/iso-25010?start=3

[12] Fraser, G., & Rojas, J. M. (2019). Software testing. In Handbook of Software Engineering (pp. 123-192). Springer, Cham.

[13] Kassab, M., DeFranco, J. F., & Laplante, P. A. (2017). Software testing: The state of the practice. IEEE Software, 34(5), 46-52.

[14] Jamil, M. A., Arif, M., Abubakar, N. S. A., & Ahmad, A. (2016, November). Software testing techniques: A literature review. In 2016 6th international conference on information and communication technology for the Muslim world (ICT4M) (pp. 177-182). IEEE.

[15] Molnar, A. J., Neamtu, A., & Motogna, S. (2019, May). Longitudinal Evaluation of Software Quality Metrics in Open-Source Applications. In ENASE (pp. 80-91).