Software quality models: Exploratory review

Lloy Pinedo¹, Miguel Valles-Coral¹, Jorge Raul Navarro-Cabrera¹, Richard Injante¹, Ángel Cárdenas-García¹, Fernando Ruiz-Saavedra¹, Cecilia García-Rivas-Plata² and César Augusto Flores-Tananta³

¹Universidad Nacional de San Martín, Tarapoto, Peru
²Universidad Nacional Jose María Arguedas, Andahuaylas, Peru
³Universidad Cesar Vallejo, Tarapoto, Peru

Abstract

INTRODUCTION: The emerging industrialization of software promotes the continuous development of new evaluation models that adapt to the product's characteristics and the organizations' needs.

OBJECTIVES: This article aims to identify the software quality models proposed between 2016 and 2020.

METHODS: We conducted an exploratory systematic review in the SciELO, IEEE Xplore, and Scopus databases, resulting in 14 research papers proposing new models for assessing software quality.

RESULTS: 79% of the research papers were extracted from IEEE Xplore, and 50% of them were authored by individuals affiliated with European institutions. We presented various software quality models focused on assessing both product quality and usability for specific purposes. Additionally, we reported that the proposed models are based on common software quality metrics standards such as CMMI, ISO/IEC 9126, and others.

CONCLUSION: The review serves to update the state of the art regarding the software quality models proposed in recent years, which will be valuable for researchers and software developers when seeking evaluation models that align with their specific needs.

Keywords: product quality, quality of use, quality standards, software development, systematic review

Received on 12 March 2023, accepted on 15 September 2023, published on 26 September 2023

Copyright © 2023 Pinedo et al., licensed to EAI. This is an open access article distributed under the terms of the CC BY-NC-SA 4.0, which permits copying, redistributing, remixing, transformation, and building upon the material in any medium so long as the original work is properly cited.

doi: 10.4108/eetsis.3982

1. Introduction

Callejas-Cuervo et al. [1] point out that the term “Software Quality” refers to the performance that a computer system must uphold throughout its lifecycle to ensure the efficiency of its functionalities. In a similar vein, Toro et al. [2] mention that software quality determines the success of companies that are currently striving for innovation and continuous improvement. Hence, the significance of the attention given to this subject.

This is why assessing the quality of software products has become a strategic element for organizations due to the impact it has on the competitiveness of developing companies. As a result, various models have been proposed to accommodate the size of companies and provide attributes for the evaluation and selection of software based on compliance with quality metrics [3].

According to Akbar et al. [4], developing high-quality software not only satisfies customer expectations but also benefits the organization by meeting production goals within the planned timeframe and cost estimates. To achieve this, as suggested by Quiroz-Martínez et al. [5], it is essential to prioritize the identification of requirements that determine the functionalities to be included in the product, considering key factors to meet software quality criteria.

On the other hand, certification processes issued by institutions based on quality standards for software products ensure efficiency, effectiveness, and add value to the products. They also instill confidence in customers and provide predictability in work. Therefore, their
consideration is valuable for companies engaged in software development that aim to showcase the quality of their production through certifications with international impact [6].

According to Huda et al. [7], inherently large and complex software systems with numerous correlated metrics developed under different components make the construction of new software quality evaluation models very complicated. As a result, as indicated by Martinez-Fernandez et al. [8], the evaluation and improvement of software quality are primary objectives for the community of systems engineers, computer scientists, and computing professionals, leading to new proposals and the formulation of quality standards and models.

It is worth noting that globally, in the field of software production, multiple standards, methodologies, and models have been developed to improve their management and development, aiming for high-quality products [2,9]. However, Saini et al. [10] point out that these quality assurance approaches or traditional metrics have limitations or shortcomings because some of them solely focus on evaluating source code and cannot be used for quality prediction or fault detection.

Several systematic review studies have been conducted on software quality models [1,10–16] where researchers compare, classify, evaluate, and determine models that are in higher demand or provide greater reliability and effectiveness in their applications for assessing software quality. However, there is still a need to continue exploring new software quality models, as they are frequently emerging in scientific literature.

For this reason, this article sets out to identify software quality models proposed between the years 2016 and 2020. The goal is to compile and encompass models aimed at software quality evaluation (QS). To achieve this, we adapted an analysis matrix for models based on the studies of Villalta et al. [11] and Yan et al. [14], classifying them by years of development or launch, incorporation of standards, and the authors who designed them. The primary purpose of this review is to update and enrich the state of the art in terms of research conducted on QS models, providing researchers and developers with the ease of finding current models.

2. Methodology

In recent decades, the software industry has undergone a revolution. Initially, determining the quality of a product was solely based on its functionality. However, as the years have passed and the complexity of building robust systems has increased, new characteristics and sub-characteristics have been incorporated to assess quality in a broader and more rigorous manner. As a result, there has been a growing development of models that can adapt to the needs of a project [12]. In light of this perspective, we formulate the research question: What software quality models have been proposed between the years 2016 and 2020?

To answer this question, we applied the methodology of Exploratory Systematic Review (ESR), which is a valuable tool for bibliographic reviews due to its systematic, transparent, and applicable nature in various scientific fields [17]. Taking into account the phases of ESR, we proceeded to conduct a literature search in electronic databases during the month of March 2021. We used SciELO, IEEE Xplore and Scopus as resources because they have a broad regional and international scope and also facilitate advanced searching through filters.

We used the terms "model", "quality" and "software" which we concatenated using the boolean operator AND, and we searched in the fields of title, abstract, and keywords. For SciELO, we conducted the search in both spanish and english.

Regarding the criteria for document selection, we included scientific articles, conference papers, and book chapters in both Spanish and English published between 2016 and 2020 that focused on new proposals for software quality evaluation models. We excluded review articles, opinion pieces, short communications, duplicates, and items classified as gray literature (technical reports, theses, monographs, etc.).

The results of the initial search without filters yielded 325, 17,935, and 65,430 documents according to SciELO, IEEE Xplore, and Scopus, respectively. After applying filters for the year range, document type, and language, we obtained 105, 4,051, and 16,198 documents. Subsequently, we reviewed the titles and abstracts of each article, excluding those that did not propose new models for evaluating software quality. This process was exhaustive to eliminate duplicates and avoid any bias. It is worth noting that a significant portion of the filtered literature focused on the evaluation of software product quality rather than on proposals or model development.

At the conclusion of the review process, you’ve gathered 14 papers focused on the development of software quality measurement models. Storing them in Mendeley for subsequent in-depth review is an excellent approach, as it allows you to organize the metadata such as title, authors, journal, year, volume, number, pages, URL, or DOI. Properly managing your bibliographic sources will be crucial as you progress in your research [18].

3. Results and discussion

3.1. Publications by database

After extracting the information, we identified that a large percentage of studies proposing the development of new software quality (QS) models are available on IEEE Xplore (79%), followed by SciELO (14%) and Scopus (7%) (Figure 1). This does not mean that scientific literature is limited to a single database but rather reflects that research is disseminated across different sources and may even be included in one another.
3.2. Regional distribution

We consider it important to identify the distribution of proposed software quality models by region. To do so, we took into account the institutional affiliation of the authors, obtaining a higher proportion in the European region (50%), followed by Asia (29%), and Latin America (21%) (Figure 2). These results reveal that a significant portion of the software quality models proposed between 2016 and 2020 were formulated by European researchers, although there is also scientific production evident from authors in other regions.

3.3. Software quality models (2016-2020)

To characterize the software quality models published between 2016 and 2020, we designed a data collection matrix from which we extracted information regarding the authors, model name, objective, standards used, and impact or results generated. These details are presented in Table 1 (next page).

Among the identified software quality models, we highlight the use of widely adopted industry standards, which served as a foundation or were expanded upon for new software quality evaluation proposals. Some of these standards include.

ISO/IEC 9126

ISO/IEC 9126 is an international standard developed for assessing and measuring the quality of software products. This standard focuses on six primary quality characteristics: functionality, reliability, usability, efficiency, maintainability, and portability. Each of these characteristics is further divided into subcharacteristics that provide specific criteria for evaluating quality in various aspects of the software [19].

ISO/IEC 9126 is widely used in the software industry to define quality requirements and establish metrics for measuring compliance with these requirements. For example, in the realm of functionality, it assesses the software's ability to meet user needs, while in efficiency, it measures performance and resource utilization.

ISO/IEC 25010

ISO/IEC 25010 is a revision and expansion of the ISO/IEC 9126 standard. This standard introduces a more comprehensive and up-to-date model for software quality. It divides software quality into eight quality characteristics, which are further broken down into subcharacteristics and quality attributes. In addition to the characteristics mentioned in ISO/IEC 9126, ISO/IEC 25010 includes aspects such as security, compatibility, and interoperability [20].

This more detailed model provides a more comprehensive view of software quality and helps organizations better understand user needs. It facilitates informed decision-making to enhance software quality, as it allows for a more precise assessment of the strengths and weaknesses of the product.

CMMI

CMMI (Capability Maturity Model Integration) is a maturity and capability model used to assess and enhance an organization's ability to consistently develop high-quality software. It focuses on key areas such as project management, software engineering, and process management [21]. CMMI provides a set of best practices and a gradual path for improving the quality and efficiency of software development processes [22].

Organizations that adopt CMMI can assess their maturity in terms of processes and receive guidance on how to improve. This helps them standardize and optimize their software development practices, which, in turn, leads to higher product quality and greater operational efficiency.
<table>
<thead>
<tr>
<th>Autores</th>
<th>Denominación</th>
<th>Objetivo</th>
<th>Estándar</th>
<th>Impacto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Espejo-Chavarría et al. [23]</td>
<td>High-Level Quality Assurance Model</td>
<td>Develop a quality assurance model with a focus on human talent</td>
<td>CMMI</td>
<td>Quality cost reduction from 19.85% to 7.41%.</td>
</tr>
<tr>
<td>Gitto et al. [24]</td>
<td>First-tier Construction Quality Assurance Model</td>
<td>Establish a complex Quality Assurance (QA) model that integrates the</td>
<td>ISO/IEC 9126</td>
<td>According to a survey, it provides reliability, security, and other</td>
</tr>
<tr>
<td></td>
<td></td>
<td>client's perspective</td>
<td></td>
<td>aspects (38%)</td>
</tr>
<tr>
<td>González-Reyes et al. [12]</td>
<td>External Quality Assurance Evaluation Model</td>
<td>Propose an external quality model for evaluating software in early-stage</td>
<td>ISO / IEC 25010</td>
<td>Applicable to any type of organization initiating their work in</td>
</tr>
<tr>
<td></td>
<td></td>
<td>organizations</td>
<td></td>
<td>quality or the quality area</td>
</tr>
<tr>
<td>Akbar et al. [4]</td>
<td>AZ Model</td>
<td>Develop the Software Development Life Cycle (SDL) model, known as</td>
<td>Not specified</td>
<td>The model combines the most important features of traditional</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the “AZ Model”</td>
<td></td>
<td>methodologies, yielding optimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>results in Quality Assurance (QA)</td>
</tr>
<tr>
<td>Kumar et al. [25]</td>
<td>Innovative Software Defect Prediction Model</td>
<td>Design and implement a prediction model using probabilistic</td>
<td>Not specified</td>
<td>The proposed model has a high</td>
</tr>
<tr>
<td></td>
<td>utilizing Probabilistic ABC</td>
<td>classification based on ABC</td>
<td></td>
<td>detection rate compared to traditional models</td>
</tr>
<tr>
<td>Yuanxu et al. [26]</td>
<td>HMM-FNN</td>
<td>Introduce a software analysis model based on fuzzy data and</td>
<td>Not specified</td>
<td>The model not only increased</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Feedforward Neural Network (FNN), as well as Hidden Markov Model (HMM)</td>
<td></td>
<td>prediction accuracy but also</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>improved learning efficiency by modifying the error function</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>algorithm during the learning phase</td>
</tr>
<tr>
<td>Mendonca et al. [27]</td>
<td>ELECTRE TRI</td>
<td>Present a multi-criteria analysis model</td>
<td>ISO 9126</td>
<td>It is observed as a highly useful model for analyzing service quality</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>in terms of user satisfaction but still</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>underutilized for analyzing software</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>quality</td>
</tr>
<tr>
<td>He et al. [28]</td>
<td>Method of Evaluation Based on Characteristic</td>
<td>Demonstrate the reliability assessment of a software component using</td>
<td>Not specified</td>
<td>Preliminary results show that this method</td>
</tr>
<tr>
<td></td>
<td>Parameters</td>
<td>a characteristic-parameter-based evaluation method</td>
<td></td>
<td>can be effectively applied in</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>evaluating the reliability of software</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>components</td>
</tr>
<tr>
<td>Utsunomiya et al. [29]</td>
<td>Not specified</td>
<td>Group classes based on influencing factors such as the number of roles,</td>
<td>Not specified</td>
<td>Empirically, the failure density</td>
</tr>
<tr>
<td></td>
<td></td>
<td>type of design pattern, and static associations between design patterns</td>
<td></td>
<td>distributions among class groups influence the design pattern for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and anti-patterns</td>
<td></td>
<td>classifying the classes</td>
</tr>
<tr>
<td>Legowo et al. [30]</td>
<td>QAS (Quality Assurance System)</td>
<td>Develop a quality assurance system using the agile development</td>
<td>BAN-PT / ISO</td>
<td>The accuracy of the QAS method in Quality</td>
</tr>
<tr>
<td></td>
<td></td>
<td>methodology with Scrum</td>
<td>9001:2008</td>
<td>Determination (QD) is verified</td>
</tr>
<tr>
<td>Cevallos- Lopez et al. [31]</td>
<td>JS (Joint System)</td>
<td>Create a JS (Joint System) model that serves as a specific guide in</td>
<td>CMMI-Dev 1.3</td>
<td>The model extends the CMMI model by incorporating quality criteria and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the domain of educational JS to ensure project quality</td>
<td></td>
<td>best practices obtained from JS studies</td>
</tr>
<tr>
<td>Moyo et al. [32]</td>
<td>Secure-SSDM (Secure Software Development Model)</td>
<td>Propose a software development methodology that promotes quality and</td>
<td>ISO / IEC 25010</td>
<td>The theoretical contribution is the addition of security practices to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>security in software products</td>
<td></td>
<td>the individual software development knowledge base</td>
</tr>
<tr>
<td>Yan et al. [33]</td>
<td>SCT Model</td>
<td>Build a comprehensive evaluation system that assesses the scientific</td>
<td>CMMI</td>
<td>The quality of the methodology, using Backpropagation Neural</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nature and accuracy of the proposal</td>
<td></td>
<td>Network (BP), significantly aligns with the comprehensive software</td>
</tr>
<tr>
<td>Ahmad et al. [34]</td>
<td>SOQEMM (Software Quality Engineering and Management</td>
<td>Develop a software subcontracting model (SOQEMM) from the provider's</td>
<td>Not specified</td>
<td>evaluation in terms of quality</td>
</tr>
<tr>
<td></td>
<td>Model)</td>
<td>perspective</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The matrix provides a panoramic view of research in the field of software quality between 2016 and 2020. The results reveal a diversity of approaches and objectives adopted by researchers, reflecting the breadth and complexity of this field.

One noteworthy aspect is the multiplicity of approaches in the models. These models are designed to address various objectives in software quality according to the specific software product [35]. Some focus on quality assurance, while others are geared towards defect prediction, integrating security into development [36], or assessing educational projects [37]. This array of approaches underscores the need for adaptive and specific strategies to tackle different aspects of software quality.

Furthermore, several of these models are based on widely recognized quality standards such as CMMI, ISO/IEC 9126, and ISO/IEC 25010. This trend of using established standards suggests that there is significant value in following well-established frameworks to enhance software quality [1]. The implementation of these standards provides a solid and globally recognized foundation for optimizing software quality.

In terms of impact, some models have achieved favorable outcomes. For example, Espejo-Chavarría et al.‘s high-level quality assurance model [23] reduced quality costs, demonstrating the practical value of implementing an approach based on recognized standards. Similarly, Kumar et al.‘s software defect prediction model [25] stands out for its ability to detect defects with a higher success rate than traditional models. These results indicate that applying specific approaches can positively influence software quality and development process efficiency.

However, it is relevant to note that some models do not specify concrete standards, which raises questions about the replicability and applicability of their results in broader contexts. Additionally, the matrix highlights that while many models have focused on technical aspects of software quality, such as functionality and security, there are still opportunities to research approaches that address quality from more holistic perspectives, such as user satisfaction and project management.

This matrix of software quality models emphasizes the importance of ongoing research in this ever-evolving field. Technology advances, and user expectations constantly change, necessitating an agile and adaptable approach to address quality challenges in software development. Each model and approach presented in the matrix contributes to the knowledge in this field and provides valuable insights for future research and improvements in software quality.

3.4. Limitations of the review

As limitations of the present review, we highlight that the search was conducted in only three databases, and we restricted the year range from 2016 to 2020. For future research, we suggest expanding the search to include new software quality models developed between 2021 and 2023. Additionally, it would be beneficial to consider the inclusion of additional databases such as Web of Science and Google Scholar to ensure a comprehensive collection of relevant models.

Furthermore, our review primarily focused on identifying software quality models proposed within a 5-year period, emphasizing their objectives and generated results. In future research, it could be explored whether these models have been adopted and used in the software industry, as well as whether they have undergone evolutions or new versions. This analysis would provide a more comprehensive understanding of the applicability and effectiveness of these models in practical environments.

Conclusions

In this article, we have presented an exploratory systematic reviews that comprehensively addresses the software quality models developed or proposed up to the present. The results reflect the dynamic progress in creating new models that respond to the changing demands of software production projects and have the primary objective of ensuring customer satisfaction.

A noteworthy finding of this review is the prevalence of adopting established standards such as CMMI and ISO regulations as fundamental bases for the development of new proposals in the field of software quality. This highlights the importance of using internationally recognized frameworks to establish solid foundations in software quality management. The ability of these standards to serve as a starting point in creating customized models demonstrates their versatility and relevance in today's software industrialization.

Finally, this article aims to provide access to relevant and useful information for academics, students, and professionals interested in software quality. Quality models play a crucial role in improving software development processes and end-user satisfaction, and this review provides a solid foundation for future research and practical applications in the ever-evolving field of software quality.

References

[L. Pinedo et al.]


