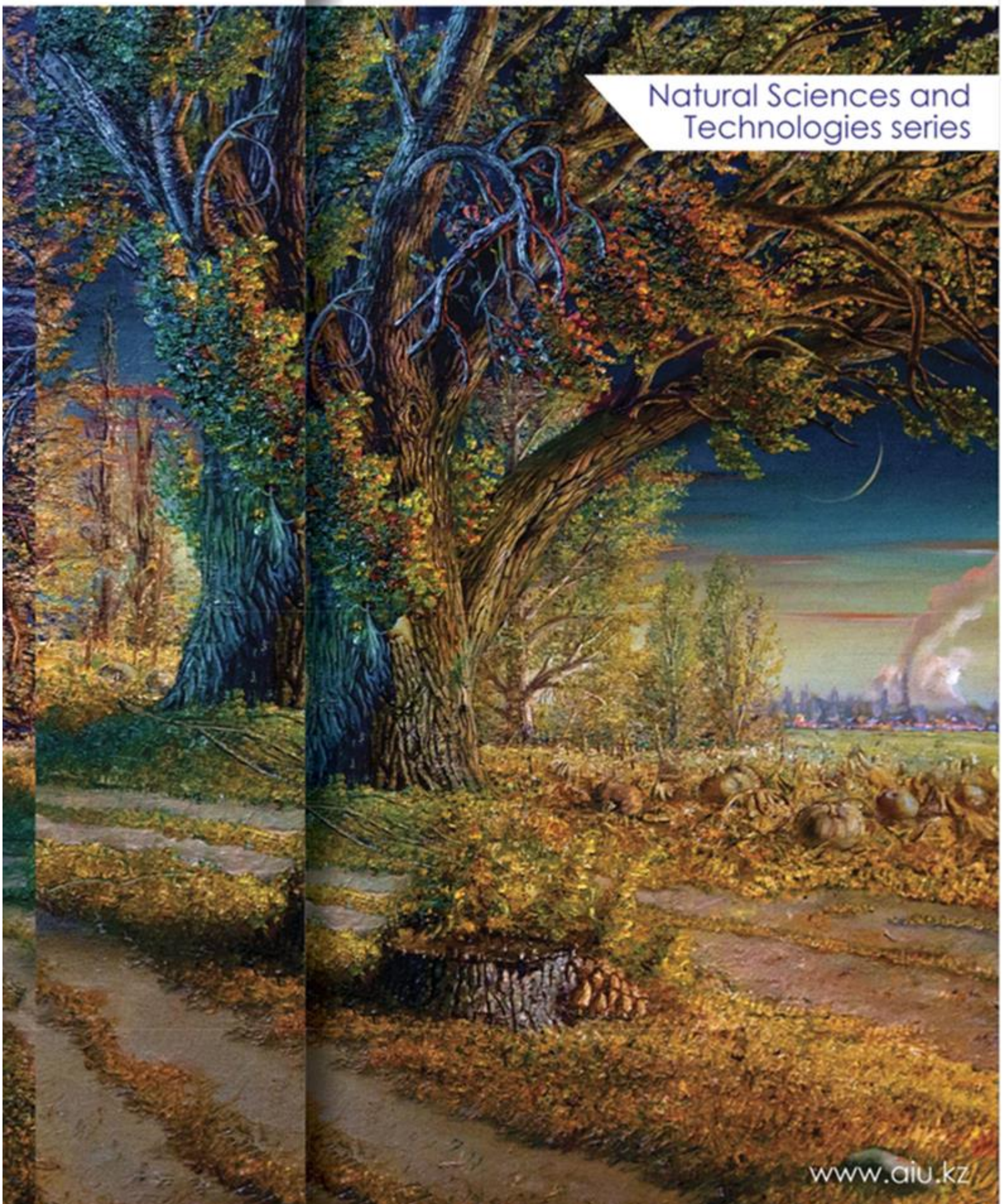
№4 (4) 2023

INTERNATIONAL SCIENCE REVIEWS

AIU

Natural Sciences and Technologies series



www.aiu.kz

ISSN: 2707-4862

ASTANA INTERNATIONAL UNIVERSITY

INTERNATIONAL SCIENCE REVIEWS
Natural Sciences and Technologies series

Has been published since 2020

№4 (4) 2023

Astana

Editorial address: 8, Kabanbay Batyr avenue, of.316, Nur-Sultan,
Kazakhstan, 010000
Tel.: (7172) 24-18-52 (ext. 316)
E-mail: natural-sciences@aiu.kz

International Sciense Reviews NST - 76153
International Science Reviews
Natural Sciences and Technologies series
Owner: Astana International University
Periodicity: quarterly
Circulation: 500 copies

CONTENT

CHARACTERISTICS AND STANDARDS OF SOFTWARE QUALITY

N. Z. Kossym

Astana International University, Astana, Republic of Kazakhstan,
kosym04@gmail.com

Abstract: To attain the envisioned level of software quality, it becomes imperative to possess information concerning the prevailing standards within this domain and adhere to them across all stages of the software life cycle. This research underscores the significance of implementing standards in the realm of software quality. Over the past two decades, the International Organization for Standards has played a pivotal role in shaping quality standards for numerous software products. The commencement of this study involves a succinct overview of the evolution of standards in software quality, followed by an exhaustive examination of the present ISO/IEC standard series known as SQuaRE. This domain holds substantial importance as familiarity with software quality standards and their practical application contributes to the creation of resilient and widely adopted high-quality software. The utilization of the outlined standards and the methodologies for assessing software quality hinge on the available resources for executing the method. These resources encompass factors such as time, financial means, the quantity of evaluators and their proficiency, the volume of users for testing, as well as the accessibility of facilities and testing equipment. Additionally, the chosen level of objectivity and its applicability across diverse stages of application development play a crucial role in determining how these standards and evaluation approaches are implemented.

Keywords: Software Quality, Software Quality Standards, ISO/IEC, SQuaRE.

INTRODUCTION

Software represents an intellectual product and acquiring it involves a substantial investment that demands justification. Justifying such investments necessitates ensuring that the software aligns with users' needs at the anticipated level. The evaluation of investment justification often involves a challenging decision-making process when it comes to choosing or purchasing software. Assessing functionality and other measurable characteristics is part of this mental process, performed in the context of real needs. For intangible assets like software, decision-making mental processes are intricate, as facts are abstract and less convincing. Consequently, there is a need for formalizing these decision-making processes based on clear and unambiguous assessment results.

To safeguard both the contracting authority and the supplier, precise contracts for software purchase and adherence are essential. Contracts in this professional domain should preferably lean on adequate standards to avoid misunderstandings about the subject and its concepts. Beyond legal protection, standards ensure that the quality of the development process and the resulting products meets the required and expected

standards. This becomes particularly crucial in cases where human error in product use could lead to potential harm or significant material damage.

As the number of software applications increases, so does the importance of software quality. To effectively manage software quality, there was a need to establish a methodology for objectively quantifying software products and their development processes. Defining software quality requires measuring and evaluating numerous features that collectively determine its quality. Metrics of software quality play a substantial role in this regard.

Scientists often create models of quality to understand and measure it. Various models suggest how different quality attributes are interconnected. The first quality models were developed in critical application areas. A quality model, such as ISO/IEC 9126 (1991), comprises characteristics and their relationships, forming the basis for specifying quality requirements and conducting quality assessments. Different approaches, ranging from classical models like Boehm (1978) and McCall (McCall et al., 1978) to more recent approaches like ISO/IEC 25010 (2010), define the set of characteristics to be measured for determining software quality or establish the quality model.

SOFTWARE QUALITY AND STANDARDS

The evolving demands and expectations for software in the market have been such that only industrial-scale production could meet them. Organizing industrial production requires standards for development. Given the dynamic nature of software development, official standards couldn't keep pace initially. The first standards in the market were industry standards, representing agreements among producer groups or those of major corporations (e.g., Microsoft) widely accepted in the market. These industry standards form the basis for initiating official standards adopted by international and national standardization bodies.

Globally, there are over 1,000 official software-related standards, with 350+ in software engineering. International standards, though not perfectly systematized, serve as potent tools for improving design methods and ensuring software quality.

Standardization in information technologies contributes to more efficient functions, stability, and smoother transitions. Applying standards in software development creates conditions for efficient, cost-effective, secure, and reliable products. Standardized processes enable planning, quantification, monitoring, documentation, and continuous improvement, laying the groundwork for defined quality.

For complex information systems involving multiple organizations, standards not only ensure quality but also facilitate project exchange, user training, and collaborative work. The rapid evolution of software led to the development of standards and tools. Standardization offers a common framework for effective communication in software development, design, and management.

Despite progress since 1976, there's no widely accepted scheme for evaluating software quality. Various models by McCall, Boehm, Eason, Shackel, Nielsen, and others contributed, but the need for a unified quality model persisted. ISO/IEC 9126 emerged in 1991 to address common questions during software acquisition and implementation, providing a comprehensive model for software quality evaluation.

Standards are crucial in software technology, providing order and enabling progress. ISO standards fall into product-oriented (defining characteristics and requirements) and process-oriented (determining development processes) categories. For software, general quality, documentation, life cycle, and profession-specific standards are commonly recommended. Usability-focused projects may benefit from the CIF standard (Common Industry Format for Usability Test Reports), offering comprehensive requirements, evaluation methods, and reporting for usability testing of software and hardware.

While general standards exist, defining software quality for specific purposes requires "professional" standards tailored to specialized software contracting.

ISO 9241-11 STANDARD

In 1998, the International Organization for Standardization (ISO) embraced the ISO 9241-11 standard to streamline contracting processes and evaluate the fulfillment of specified characteristics. The ISO 9241-11 standard, utilized for assessing usability performance and user satisfaction (Folmer & Bosch, 2004), defines usability as "the extent to which a product can be used by specific users to achieve specific goals with efficiency, effectiveness, and satisfaction in a specified context of use" (ISO 9241-11, 1998).

This definition encompasses two user-centric perspectives. Firstly, it considers user performance, encompassing effectiveness and efficiency, and secondly, it addresses user satisfaction. The definition comprises four key elements: users, user goals, the product, and the context of use. The context involves "users, tasks and equipment, and the physical and social environment in which the product is used." The ISO 9241-11 standard identifies three measurable attributes of usability: effectiveness, efficiency, and satisfaction.

Effectiveness: Users successfully achieve their goals with accuracy and completeness. The system should contain features that ensure users encounter no difficulty in accomplishing their objectives.

Efficiency: The system's resources are utilized accurately and completely to achieve users' goals.

Satisfaction: Users must derive satisfaction from using the system.

ISO 9241-11 underscores the advantages of measuring usability concerning performance and customer satisfaction. It emphasizes that the usability of the visual display is contingent upon the context of use, implying that the achieved level of usability depends on the specific circumstances in which the product is employed.

ISO/IEC 9126 STANDARD

In he early 1990s, there was an initiative in software engineering to consolidate various aspects of quality into a unified model serving as a global standard for assessing software quality. This resulted in the ISO 9126 (ISO 9126, 1991) standard, designed to facilitate understanding in negotiations between client software and manufacturers. The primary goal was to recommend the extent and quality characteristics software must possess. The ISO 9126 model, based on McCall's model, defines product quality through a set of characteristics. Notably, the ISO model is characterized by strict hierarchies, distinguishing it from McCall's and Boehm's models.

The initial version of the standard outlined a quality model with six main characteristics—functionality, reliability, usability, efficiency, maintainability, and portability—comprising 20 subcharacteristics. Each characteristic addresses a specific attribute from the user's perspective, steering away from an internal designer's view. While the standard recommends direct measurement of these characteristics, it doesn't delve into the specifics of how measurements are conducted.

This initial model served as a foundation, adaptable as needed for specific projects. However, ISO/IEC 9126 lacked attributes and metrics within the standard itself, nor did it prescribe measurement methods, rankings, or evaluations. Instead, it offered general guidelines for a software quality evaluation process. Challenges arose in determining completeness and consistency, as models lacked explanations for inclusion or hierarchy structure.

Between 2001 and 2004, the International Organization for Standardization revised the standard, issuing an expanded version. Recognizing the utility of models and metrics beyond evaluation, a new series, ISO/IEC 14598, was introduced to isolate the software quality evaluation process. Over time, ISO/IEC 14598 was split into two standards: ISO/IEC 9126 (ISO/IEC 9126, 2001) and ISO/IEC 14598, a decision that complicated matters and led to a decline in popularity for both standards.

ISO/IEC 9126 comprises four parts, with the first part (ISO/IEC 9126-1) focused on concepts and presenting a dual quality model: internal/external quality and quality in software use. The model categorizes quality attributes into six characteristics, further divided into 27 subcharacteristics measurable by internal or external metrics. The second part (ISO/IEC TR 9126-2) covers external characteristics of software quality metrics, while the third part (ISO/IEC 9126-3) provides internal metrics. The fourth part (ISO/IEC 9126-

4) contains a fundamental set of metrics for each quality characteristic in use, along with instructions for application and examples of their use in the software product lifecycle.

In the context of ISO/IEC 9126-1, quality in use refers to how an end user perceives a complete system, measured by the results of software use. The attributes of internal and external quality serve as causes, while the attributes of quality in use act as effects. These three perspectives are interrelated, where measuring and evaluating quality in use can confirm external software quality, and vice versa. Similarly, considering the attributes of internal quality is crucial for achieving the required external behavior, and attributes of external quality are prerequisites for quality in use.

ISO/IEC 14598 STANDARD

Originally conceptualized by ISO/IEC 14598-1 (ISO/IEC 14598-1, 1999), the process of assessing software quality aims to provide methods for measuring, evaluating, and appraising the quality of software products. Serving as the foundation for this series, ISO/IEC 14598-1 defines fundamental concepts and outlines the utilization of characteristics and quality metrics specified in the ISO/IEC 9126 series. Published under the overarching title of Information technology - Software product evaluation, ISO/IEC 14598-1 comprises multiple parts, covering aspects like the model of quality, evaluation methods, software measurement, and supporting tools. This standard caters to designers, procurement personnel, and independent assessors.

The ISO/IEC 14598 series of international standards offers guidelines and requirements for the evaluation process in three main contexts:

Development of new products or enhancements of existing ones (ISO/IEC 14598-3, 2000): This part addresses the evaluation process concerning the creation or improvement of software products.

Procurement of products or reuse of existing products (ISO/IEC 14598-4, 1999): Focusing on the acquisition of software products or the utilization of existing ones, this part provides guidance for evaluation.

Independent evaluation of requirements by the designer, supplier, or a third party (ISO/IEC 14598-5, 1998): This part is dedicated to independent evaluations carried out by designers, suppliers, or third parties.

Recognizing software testing as the most efficient and effective means of achieving and maintaining software product quality, it operates based on established rules and principles within quality standards. In a way, software testing incorporates all predefined methods and techniques for creating and sustaining high-quality software.

SQUARE – ISO STANDARDS FOR SOFTWARE QUALITY

Recognizing the limitations of the ISO/IEC 9126 standard, as acknowledged by various authors in the literature such as (Azuma, 2001), (AlQutaish, 2010, pp.205-228), (Suryn and Gil, 2005), (Olsina and Molina, 2008), the International Organization for Standardization (ISO) took the initiative to develop a new series of standards for software quality. This new standard, known as SQuaRE (Software Product Quality Requirements and Evaluation), represents the second generation of standards for software quality. The intention is for SQuaRE to eventually replace both the ISO/IEC 9126 and ISO/IEC 14598 series. The development of SQuaRE adheres to several guiding principles:

1. Integration of ISO/IEC 9126 and ISO/IEC 14598: SQuaRE harmonizes these two series into a consolidated standard.

2. Introduction of a New Organization and Standard: A new structure and standard are introduced to enhance organization and clarity.

3. Introduction of a New Reference Model: SQuaRE incorporates a new reference model to guide the evaluation process.

4. Introduction of Detailed Guides: Detailed guides accompany the standards to provide comprehensive understanding and practical guidance.

5. Introduction of Standards on Quality Requirements: SQuaRE introduces standards specifically addressing quality requirements.

6. Introduction of a Manual for Practical Use with Examples: A manual is included to facilitate the practical application of the series, supplemented with examples.

7. Coordination and Harmonization with ISO/IEC 15939: The measurement model within SQuaRE is aligned and harmonized with ISO/IEC 15939 Software engineering - Software measurement process.

SQuaRE encompasses a series of fourteen ISO/IEC standards and technical reports grouped into five thematic sections or parts:

1. Quality Management (ISO/IEC 2500n): Provides an overview of the evaluation process and quality models, establishing common models, terms, and definitions for the entire series.

2. Quality Model (ISO/IEC 2501n): Introduces a detailed model of quality, including features for internal, external, and quality in use, along with instructions for practical use.

3. Measurement of Quality (ISO/IEC 2502n): Encompasses standards with a reference model for measuring software product quality, metrics for internal, external, and quality in use, and practical usage instructions.

4. Requirements for Quality (Series ISO/IEC 2503n): Aids in specifying required quality, mapping the process of defining requirements to technical processes outlined in the ISO/IEC 15288 standard.

5. Evaluation of Quality (Series ISO/IEC 2504n): Provides requirements, recommendations, and guidelines for evaluating software products, involving evaluators, customers, or software developers.

Key distinctions between the SQuaRE series and ISO/IEC 9126 or ISO/IEC 14598 include the introduction of a general reference model, coordinated guidelines for measuring and evaluating software quality, a section on quality requirements, the existence of detailed manuals for each part, guidelines for practical use through examples, and harmonization with terminology related to software measurement (utilized in the ISO 15939 standard). Nigel Bevan, an international expert in software usability, played a significant role in shaping this series, emphasizing the critical importance of quality in use (Bevan et al., 1991).

ISO 25010 STANDARD

The International Organization for Standardization (ISO) has recently introduced an expanded definition of quality in use. This enhanced definition incorporates subcharacteristics such as usability, flexibility, and security, allowing for quantification from the perspectives of various stakeholders, including users, managers, and software maintainers. ISO/IEC 25010 preserves the existing three perspectives on quality—internal, external quality, and quality in use. Notably, it broadens the concept of product quality from the previously established six characteristics in ISO/IEC 9126-1 to eight characteristics (refer to Fig. 1.).

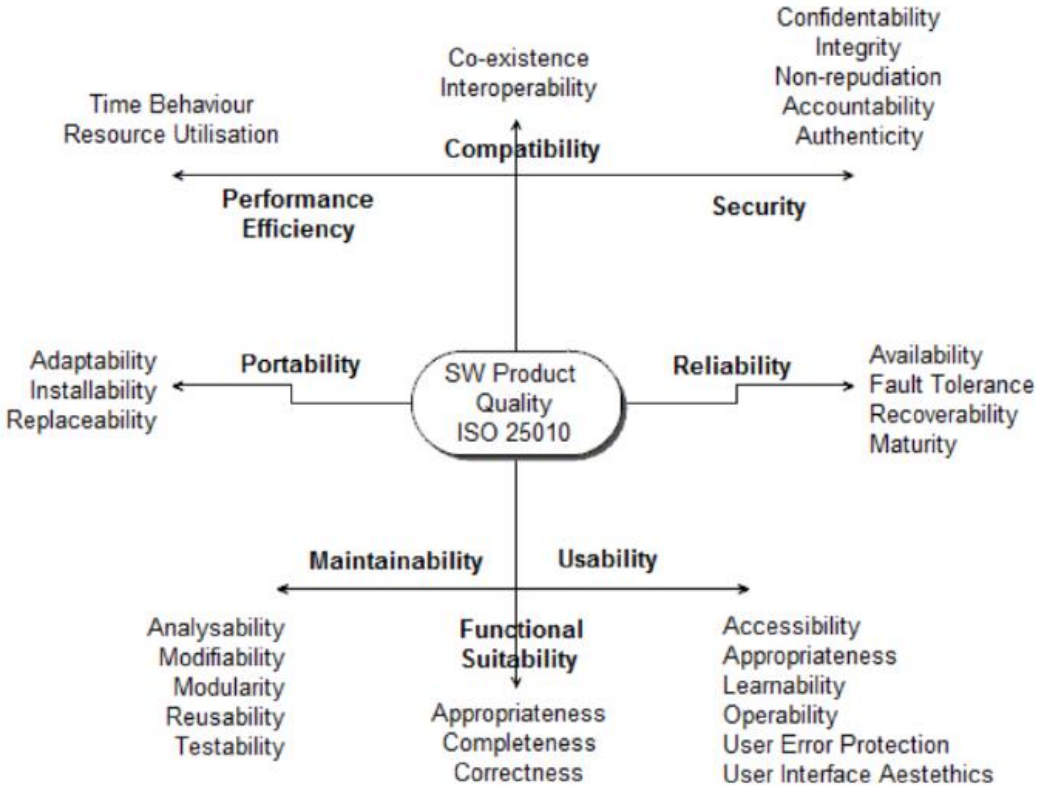


Fig. 1. Software product quality according to ISO/IEC 25010

When compared to ISO/IEC 9126-1, the usability aspect has undergone a name change to "operability," encompassing a more extensive scope. While retaining certain subcharacteristics like suitability for learning, ISO/IEC 25010 introduces new elements such as technical accessibility, compliance, and the capacity of applications to assist users. The addition of compatibility as a new feature is noteworthy, and Safety has been designated as a distinct feature rather than a subcharacteristic of the Functionality trait in the previous standard. Some names have been subtly adjusted to enhance descriptiveness.

The second model within the ISO 25010 standard pertains to quality in use (refer to Figure 2), incorporating the original quality in use characteristics from ISO 9126-1, along with the introduction of some novel elements.

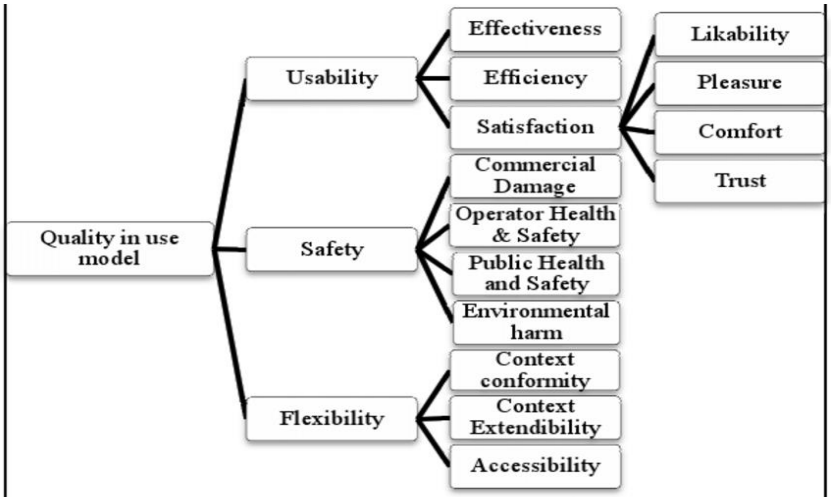


Fig. 2. Model for quality in use in ISO/IEC 25010

In comparison with ISO/IEC 9126-1, the designation for usability has evolved into "operability" in ISO/IEC 25010, reflecting a more inclusive scope. While preserving specific subcharacteristics like suitability for learning, ISO/IEC 25010 introduces fresh elements, including technical accessibility, compliance, and the capability of applications to provide assistance to users. The addition of compatibility as a new feature is notable, and Safety has been identified as a distinct feature rather than a subcharacteristic within the Functionality trait, as was the case in the previous standard. Some names have been subtly adjusted for improved descriptive clarity.

The second model within the ISO 25010 standard pertains to quality in use (Fig. 2.), encompassing the original quality in use characteristics from ISO 9126-1, along with the introduction of some novel elements. This broader interpretation of usability is integrated into the modified ISO 9126-1, 2001, referred to as "quality in use," emphasizing quality from the user's perspective during product usage (Bevan, 1999). As the ISO/IEC 9126-1 quality model became part of the SQuaRE series (such as ISO/IEC 25010), some ISO/IEC national bodies noted a misalignment between the narrow usability definition inherited from ISO/IEC 9126 and the comprehensive definitions of the CIF. To align the SQuaRE usability definition with the CIF, the usability characteristic was rebranded as "operability," encompassing a broader meaning. This adjustment allows usability to be defined as a quality in use characteristic, featuring subcharacteristics like effectiveness, efficiency, and satisfaction. In ISO/IEC CD 25010.3, quality in use encompasses two characteristics: Safety from ISO/IEC 9126-1, along with a new feature—Flexibility.

SOFTWARE QUALITY METRICS

While it may appear that the domain of quality metrics is well-defined and the application of metrics for monitoring and evaluating quality and usability is clear, there remains considerable uncertainty. The formalization of the evaluation area and the measurement of various quality indicators is lacking.

To measure is to assign a value, following specific rules, to an object, usually referred to as a value. Measurement involves determining how many times the measuring quantity (often called a unit of measurement) is contained in the value to be measured. The result of this value measurement is a number, and it is crucial to differentiate between the value and the number obtained through measurement. This number, along with the designation of a value unit, is termed a numerical value of a quantity or a number of a unit of measure.

Measurement is the process of determining the quantity, size, or degree of something using a standardized set of measures and a measurement procedure defined by metrics. Metrics define a measurement system or standard, scales, and units of measurement for monitoring efficiency indicators (Ted, Angelika).

Quality in use pertains to the quality of the software product from the user's perspective when used in a specific environment and within a particular context. It gauges the extent to which users can accomplish their goals in a given environment, focusing on user experience rather than the inherent properties of the software itself. Meeting criteria for external quality is insufficient to meet criteria for internal quality, and often, meeting external quality criteria alone is insufficient for achieving quality in use.

When discussing quality, the emphasis is typically on meeting user demands for quality. User requirements for quality refer to the quality of the product in use. To integrate the fulfillment of quality demands into the software development process, it is crucial to evaluate the software product across all phases of its life cycle.

Quality assessment involves measuring internal (usually static measurement of intermediate products) and external attributes (measuring code behavior during execution) or measuring quality in use attributes. Software quality metrics play a pivotal role in evaluating each characteristic to ensure the required quality. According to the ISO quality model, internal, external, and quality in use metrics are employed to measure software quality.

Internal quality, measured based on internal quality requirements, can be improved during code implementation and testing. However, the fundamental internal quality of the software product remains unchanged, except in cases of redesign. Internal metrics, a quantitative scale and measurement method, are utilized during design or coding in the early stages of the software life cycle. These metrics can be applied to non-executable software products (e.g., specifications or source code) during design and coding.

External quality refers to the quality of software execution, typically measured and evaluated during testing in a simulated environment with simulated data using external metrics. External metrics, a quantitative scale and measurement method, are applicable to software running during testing or in later stages of development, and even during

real-world use. They utilize software product measurements derived from system behavior measurements during testing, running, and monitoring.

Quality in use metrics assess the extent to which a product meets user needs to achieve specified goals effectively, productively, safely, and with satisfaction in a given context. The evaluation of quality in use validates the software product's quality in user-task scenarios.

The relationship between quality in use and other quality characteristics of software products depends on the user type:

- For end users, quality in use is primarily a result of functionality, reliability, usability, and efficiency.
- For those responsible for software implementation, quality in use is mainly a result of maintainability.
- For those handling the software, quality in use is mainly a result of portability.

CONCLUSION

Defining the concept of software quality involves navigating numerous dimensions, making it a complex endeavor. To ensure the desired quality, it is imperative to monitor diverse parameters, formulate plans, and establish standards and quality system documentation applicable to software products. The significance of software quality is evident for both manufacturers and users, with its interpretation varying based on the observer's perspective. Achieving a software product aligned with specifications, meeting customer requirements, and devoid of errors necessitates the measurement of multiple parameters, demanding the selection of suitable measurement parameters and the implementation of relevant testing methodologies and techniques. A pivotal challenge lies in the definition of metrics, correlating the type of measurement to the software system.

Therefore, adherence to prescribed standards during software development is crucial to circumvent later interventions or software modifications in subsequent phases of the software life cycle. Quality, with its diverse aspects and metrics associated with different phases of the product life cycle, is evaluated using a defined quality model during the phase of setting quality goals for products or intermediate products. This evaluation can involve measuring consequences or employing direct measurement. Hierarchical decomposition of the software product aids in generating a list of parameters related to quality, although measuring all characteristics in all cases may not be feasible.

By measuring and evaluating quality in use, external software quality can be validated. Additionally, the measurement and evaluation of external quality serve to verify internal software quality, and an examination of internal quality can lead to conclusions about necessary improvements in the software production process. The attributes of internal and external quality are foundational, while the attributes of quality in use represent the effects. Bevan (1999) succinctly concludes, "Quality in use is the goal, and the quality of the software product is the means by which this goal is achieved."

In accordance with international standards (ISO/IEC 9126-1, ISO/IEC 25010), quality in use is how the end user perceives the complete system on which the software operates, measured by the outcomes of software usage. Existing quality models in current standards describe traditional software products well and are suitable for assessing both the quality and usability issues of traditional graphic user interfaces. However, these models fall short in describing the quality of a broader set of applications based on web technology.

The models of quality in these standards cater to various stakeholders, including developers, system integrators, owners, maintenance teams, contractors, security experts, quality control professionals, and users. Nevertheless, the relevance of the complete set of quality characteristics in these models may vary for different user types. Thus, customization of the model is essential, considering the significance of quality characteristics for each type of user and aligning them with the objectives and tasks. Presently, numerous methods exist for evaluating software quality, with the choice dependent not only on the software product type but also on project objectives and usage context. Factors such as required resources, objectivity levels, and applicability in different development stages play a crucial role in selecting an appropriate evaluation method.

REFERENCES

1. McCall, J., Richards, P., & Walters, G. 1977. Factors in Software Quality. NTIS AD-A049-014.

2. ISO/IEC 25010.3, 2010, Systems and software engineering Software product Quality Requirements and Evaluation (SQuaRE) – Software product quality and system quality in use models.

3. Boehm, B.W., Brown, J.R., & Lipow, M. 1976. Quantitative Evaluation of Software Quality. In: Proceedings of the 2nd International Conference on Software Engineering. , pp.592-605.

4. Eason, 1984. Towards the experimental study of usability. Behaviour and Information Technology, 3(2), pp.133-143.

5. Ted, H., & Angelika, P. 2006. Measuring the effectiveness of your ISMS implementations based on ISO/IEC 2700. BSI.

6. ISO/IEC 14598-4, 1999, Information Technology - Software Product Evaluation. Part 4: Process for Acquirers.

7. ISO/IEC 14598-5, 1998, Information Technology - Software Product Evaluation. Part 5: Process for Evaluators.

8. ISO/IEC 9126-1, 2001, Software Engineering - Product Quality: Part 1: Quality Model.

9. ISO/IEC TR 9126-2, 2003, Software Engineering - Product Quality. Part 2: External Metrics.

10. ISO/IEC TR 9126-3, 2003, Software Engineering - Product Quality. Part 3: Internal Metrics.

11. ISO/IEC TR 9126-4, 2004, Software Engineering - Product Quality. Part 4: Quality in Use Metrics.

12. Azuma, M. 2001. SquaRE: The next generation of the ISO/IEC 9126 and 14598 international standards series on software product quality.

13. J. Nielsen. Usability Engineering. Academic Press, Boston, 1993.