Cerdika: Jurnal Ilmiah Indonesia, 6 (3), 2026
p-ISSN: 2774-6291 e-ISSN: 2774-6534
Available online at http://cerdika.publikasiindonesia.id/index.php/cerdika/index

Optimization of Software Testing Processes Through a Quality Management System

Maya Munaiseche[1*], Radea Aditya Putra Mohamad[2] , Immanuel Naharia[3] , Stive Immanuel Ante[4] , Dave Immanuel Lilonsili[5]
Politeknik Negeri Manado, Indonesia
Email: Maya.munaiseche@polimdo.ac.id* , Radeamohamad0@gmail.com, immanuelnaharia520@gmail.com, stifante@gmail.com, davelilonsili04@gmail.com

Keywords:
Quality Management System; Software Quality Assurance; Quality Control; ISO 9001:2015; Software Testing.

Abstract

Background: The development of technology, both hardware and software, is rapidly advancing due to the growth of Artificial Intelligence (AI), which simplifies many human activities, including the software testing process. This research journal discusses how software testing is conducted through a Quality Management System (QMS). Objective: The journal examines whether software testing processes are related to QMS, how significantly QMS supports software testing, and the differences between testing with QMS and without QMS. Methods: This study outlines the stages of software testing through QMS, such as Software Quality Assurance (SQA), Quality Control (QC), Test Maturity Model Integration (TMMI), and performance indicators. According to the American Society for Quality (ASQ), QMS is a formal and documented system consisting of policies, processes, and procedures designed to achieve an organization's quality policies and objectives. Results: Several experts also define QMS as a system that focuses on improving customer satisfaction, process consistency, efficiency, and the overall quality of products or services. Software, according to Roger S. Pressman, is a set of instructions (programs), data structures that allow information manipulation, and related documentation describing operations and functions. Conclusion: Overall, QMS not only improves software product quality but also strengthens the testing process, making it more optimal, efficient, and aligned with organizational goals. The integration of QMS frameworks such as ISO 9001:2015, CMMI, and TMMI provides organizations with a competitive advantage by ensuring consistent delivery of high-quality software products that meet or exceed customer expectations.

INTRODUCTION

According to the American Society for Quality (ASQ), QMS is a formal, documented system consisting of policies, processes, and procedures designed to achieve the quality policy and objectives of an organization. Several experts also define QMS as focusing on improving customer satisfaction, process consistency, operational efficiency, and overall product or service quality (Gomaa 2025; Schmeleva 2017; Usman et al. 2019).

Software is systematically defined as a set of instructions, programs, or electronic data that cannot be physically touched and serves to direct the computer in performing specific tasks (Englander et al. 2021; Storey 2022). In today's digital era, the role of software has become crucial in almost every aspect of human life, ranging from communication applications and health systems to financial management and industrial automation (Aceto et al. 2019; Gopal et al. 2019; Marques et al. 2019).

The software testing process involves a series of procedures to ensure that a product functions properly and is free from defects before release. The process begins with product

planning, requirement analysis, design, execution, and ends with reporting test results (Jaskó et al. 2020; Seider et al. 2016). The main objective is to identify bugs and ensure the software's quality and reliability meet user expectations (Abelson et al. 2022; Englander et al. 2021; Storey 2022).

Software testing through QMS integrates structured quality management processes—from planning to corrective action (CAPA)—ensuring compliance with applicable standards (Sommer et al. 2015). QMS automates and centralizes the management of testing workflows, from identifying testing needs and validation planning to tracking corrective actions (Owen et al. 2025; Srinivas et al. 2024).

Despite this substantial body of research, significant gaps remain in understanding how QMS principles can be optimally integrated into software testing processes to achieve measurable improvements in efficiency, effectiveness, and quality outcomes. Most existing studies either focus on general quality management principles without specific application to software testing contexts or examine testing practices in isolation without systematic connection to broader QMS frameworks (Agmon et al. 2024; Pargaonkar 2023; Tsolakis 2026). Furthermore, limited research has addressed the practical implementation challenges organizations face when attempting to integrate QMS with testing processes, including issues of documentation consistency, training adequacy, and alignment with agile development methodologies that have become prevalent in contemporary software development.

The urgency of addressing this research gap is underscored by several converging factors. First, software systems have become increasingly critical to economic activity, public safety, and daily life, making software quality failures potentially catastrophic in terms of financial loss, safety risks, and reputational damage. Second, regulatory requirements in many industries mandate documented quality processes and traceable testing evidence, requiring organizations to demonstrate compliance through systematic QMS implementation. Third, the cost of defect correction escalates exponentially as defects progress through development phases, with research indicating that fixing defects during testing costs approximately 10 times more than during design, and up to 100 times more if discovered after release (Pressman, 2019). Fourth, customer expectations for software quality and reliability continue to rise, making consistent quality delivery a competitive necessity rather than an optional differentiator.

The background of this research lies in the growing complexity of software systems, which demands more structured and systematic testing approaches. Many organizations face challenges in maintaining consistency in their testing processes, documenting results, and responding to defects efficiently. The absence of a proper quality management framework often leads to repeated errors, delayed releases, and unsatisfied end-users.

This study aims to analyze the effectiveness of QMS integration in software testing by examining its impact on testing efficiency, documentation quality, defect reduction, and overall software product reliability. It also identifies common obstacles in QMS implementation and proposes strategic recommendations to overcome these challenges.

RESEARCH METHOD

This educational research utilizes a qualitative descriptive method, aiming to thoroughly describe the processes, conditions, and application of the Quality Management System (QMS)

in optimizing software testing effectiveness. This approach does not rely on statistical data but collects narrative data, observations, and interviews to gain a holistic understanding.

Sugiyono (2019) states that qualitative research explores natural conditions where the researcher acts as the primary instrument, emphasizing meaning over generalization. Moleong (2017) describes qualitative research as a method intended to understand phenomena experienced by research subjects holistically through descriptive narratives within natural contexts.

This research aims to describe and analyze the application of QMS in optimizing software testing, beginning from planning to execution and evaluation. The software testing process is carried out using QMS principles such as ISO 9001, CMMI, or organizational internal standards.

Data Collection Methods

Data collection for this study employs multiple qualitative methods to ensure triangulation and validity of findings:

1. **Literature Study:** Review of academic journals, industry standards (ISO 9001:2015, CMMI, TMMI), textbooks, and technical documentation related to QMS and software testing.
2. **Observation:** Direct observation of software testing workflows in organizations that have implemented QMS, noting adherence to documented procedures, testing consistency, and defect tracking practices.
3. **Interviews:** Semi-structured interviews with QA engineers, software developers, and project managers to gather firsthand insights on QMS implementation challenges and benefits.
4. **Document Analysis:** Analysis of testing documentation, audit reports, CAPA records, and quality metrics produced under QMS frameworks.

Data Analysis

Data analysis in this study uses qualitative descriptive analysis consisting of:

1. **Data Reduction:** Sorting and simplifying observation data by selecting relevant information, focusing on QMS implementation aspects most pertinent to software testing effectiveness.
2. **Data Presentation:** Organizing findings in narrative form, tables, or testing process flow diagrams to facilitate comprehension and comparison.
3. **Conclusion Drawing:** Identifying how QMS influences the effectiveness and efficiency of the testing process, supported by evidence from literature and empirical observation.

RESULTS AND DISCUSSION

Based on literature studies and analysis of QMS implementation in software testing, several key findings were obtained:

Improved Testing Efficiency

QMS reduces testing time by clarifying workflows, roles, and leveraging automated testing tools, enabling teams to troubleshoot defects more quickly. Organizations that implement QMS in their testing processes report a significant reduction in the time spent on

regression testing, as standardized test cases and automated tools help maintain consistency across testing cycles.

The introduction of automated testing pipelines within a QMS framework allows testing teams to execute thousands of test cases within hours—a task that would require days if performed manually. This efficiency gain translates directly to shorter development cycles and faster product releases without compromising quality.

Greater Consistency and Documentation Quality

Standards such as ISO 9001:2015 encourage systematic documentation of every testing phase, simplifying audits and traceability. Consistent documentation ensures that testing activities are reproducible, knowledge is retained within the organization, and regulatory compliance can be demonstrated effectively.

The following table summarizes key QMS documentation requirements and their benefits in software testing:

Table 1. QMS Documentation Requirements and Testing Benefits

Documentation Type	QMS Requirement	Testing Benefit
Test Plan	ISO 9001 Clause 8.1	Defines scope, objectives, and resources
Test Cases	ISO 9001 Clause 8.3	Ensures complete requirements coverage
Defect Reports	CAPA Procedure	Tracks and resolves non-conformities
Audit Reports	ISO 9001 Clause 9.2	Verifies adherence to quality standards
Test Summary Report	IEEE Std 829	Documents test results for stakeholders

Higher Software Product Quality

By applying QMS principles, the number of defects significantly decreases because Software Quality Assurance (SQA) activities are performed continuously throughout the development cycle. Early defect detection—facilitated by systematic reviews, inspections, and testing at each stage—reduces the cost of fixes exponentially compared to defects discovered post-release.

Research indicates that the cost of fixing a defect during the testing phase is approximately 10 times higher than during the design phase, and up to 100 times higher if discovered after product release (Pressman, 2019). QMS-driven early testing therefore offers substantial cost savings in addition to quality improvements.

User Satisfaction and Product Reliability

Software tested under QMS frameworks demonstrates higher stability and performance levels. Customer satisfaction surveys conducted across multiple technology companies show a positive correlation between QMS maturity levels and end-user satisfaction scores. Products developed with structured QMS testing processes experience fewer post-release defects and require less corrective patching.

Testing Stages within QMS

The stages of software testing within QMS provide a structured lifecycle that ensures quality at each phase:

1. **Planning and Definition:** Identification of testing needs and objectives aligned with organizational quality goals. Risk analysis is performed to identify potential issues in

the software under development. A software validation plan is prepared as part of QMS documentation.

2. **Implementation and Development:** This stage includes the design and coding of software, continued risk analysis, and traceability matrix creation. The traceability matrix links each requirement to specific test cases, ensuring complete coverage.
3. **Testing (Verification):** The testing process focuses on verifying the software using QMS tools and ensuring that it functions consistently within predefined parameters. This includes unit testing, integration testing, system testing, and user acceptance testing.
4. **Corrective Actions and Improvements (CAPA):** CAPA addresses non-conformities found during testing in a structured manner using QMS tools to ensure consistent documentation. Each defect is traced, analyzed for root cause, and resolved with preventive measures.
5. **Reporting and Continuous Improvement:** This phase evaluates testing and CAPA trends, ensuring that production processes are documented and controlled effectively through internal audits and inspections to ensure continuous compliance.

Discussion

Based on observations, the implementation of QMS in software testing significantly enhances structure and systematic workflow in software development. QMS principles such as Quality Planning, Quality Control, and Quality Assurance are applied from the planning phase to test result reporting. This aligns with Crosby's (1979) idea of 'Doing it right the first time,' which emphasizes preventing errors early rather than detecting them late in the process.

However, findings reveal inconsistencies in documentation and reporting practices among team members, indicating incomplete QMS implementation, particularly regarding adherence to quality standards. These inconsistencies point to a gap between QMS policy and actual practice—a challenge identified across multiple organizations, especially those in the early stages of QMS adoption.

The positive impact of QMS on testing effectiveness is evident: QA engineers acknowledge that standardized procedures accelerate testing and reduce errors. This supports Deming's (1986) principle of continuous improvement, which stresses the need for ongoing process refinement. The evidence consistently shows that teams with well-defined testing standards and documented workflows outperform those without formal quality systems.

Nonetheless, several obstacles remain, including insufficient QMS training, tight project deadlines, and lack of periodic evaluations. These challenges indicate that QMS success depends not only on the system but also on human resource commitment. Juran (1998) reinforces this by highlighting the role of quality culture in organizational success—the idea that quality must be internalized at every level of the organization, not just enforced through policies.

Key Roles in QMS-Driven Testing

Effective QMS implementation in software testing requires well-defined roles and responsibilities:

- **Software Quality Assurance (SQA) Engineer:** Responsible for defining quality standards, developing testing strategies, and ensuring all team members adhere to

established procedures. SQA engineers serve as the guardians of the QMS framework within development teams.

- **Quality Control (QC) Analyst:** Focuses on product-level inspection and defect identification. QC activities are reactive in nature, identifying deviations from quality standards after they occur and initiating corrective actions.
- **Test Manager:** Oversees the overall testing process, allocates resources, manages timelines, and ensures that testing activities align with organizational quality goals and QMS requirements.
- **Development Team:** Responsible for implementing quality at the source through code reviews, unit testing, and adherence to coding standards defined in the QMS.

Recommended Optimization Strategies

Strategies recommended for optimizing QMS in software testing are aligned with the PDCA cycle (Plan-Do-Check-Act), which forms the foundation of QMS. These strategies address both systemic and human dimensions of quality management:

- **Establish Clear and Measurable SOPs:** Standard Operating Procedures should be documented, regularly reviewed, and accessible to all team members. SOPs must include specific acceptance criteria and step-by-step testing procedures.
- **Provide QMS Training and Certification:** Regular training programs ensure that team members are proficient in QMS principles and testing tools. Certifications such as ISO 9001 Lead Auditor or ISTQB (International Software Testing Qualifications Board) enhance team competence.
- **Increase Automation Testing Tool Usage:** Automation tools such as Selenium, JUnit, Jenkins CI/CD pipelines, and SonarQube integrate seamlessly within QMS frameworks and significantly reduce manual testing effort while improving coverage and repeatability.
- **Conduct Regular Internal Audits:** Periodic internal audits identify deviations from QMS standards early, allowing corrective actions to be implemented before they escalate into major quality failures.

Comparison: Testing with and without QMS

A comparative analysis between software testing conducted with and without QMS integration reveals significant differences in outcomes:

Table 2. Comparison of Software Testing with and without QMS

Aspect	Without QMS	With QMS
Process Structure	Ad hoc, inconsistent	Standardized, systematic
Documentation	Minimal, informal	Comprehensive, traceable
Defect Detection Rate	Lower, often late	Higher, early-stage detection
Testing Time	Longer, repetitive	Reduced with automation
Team Coordination	Fragmented	Aligned with defined roles
Compliance	Difficult to demonstrate	Built into the process
Continuous Improvement	Reactive	Proactive through PDCA

CONCLUSION

Software testing using a Quality Management System (QMS) significantly enhances effectiveness and efficiency in ensuring software product quality. QMS provides a structured framework—from planning and standardized procedures to execution and evaluation—allowing each testing phase to be conducted consistently and measurably.

Through the use of ISO/IEC standards, systematic documentation, and software quality metrics, organizations can reduce errors, minimize repetitive testing, and detect defects earlier in development. QMS also enhances communication across teams and ensures continuous quality control throughout the software development lifecycle.

Overall, QMS not only improves software product quality but also strengthens the testing process, making it more optimal, efficient, and aligned with organizational goals. The integration of QMS frameworks such as ISO 9001:2015, CMMI, and TMMI provides organizations with a competitive advantage by ensuring consistent delivery of high-quality software products that meet or exceed customer expectations.

Future research should explore the quantitative impact of specific QMS tools on defect rates across different software development methodologies such as Agile, Scrum, and DevOps. Additionally, a longitudinal study examining the long-term effects of TMMI maturity progression on organizational software quality outcomes would provide valuable insights for industry practitioners.

REFERENCES

Abelson, H., & Sussman, G. J. (2022). *Structure and interpretation of computer programs: JavaScript edition*. MIT Press.

Aceto, G., Persico, V., & Pescapé, A. (2019). A survey on information and communication technologies for Industry 4.0: State-of-the-art, taxonomies, perspectives, and challenges. *IEEE Communications Surveys & Tutorials, 21*(4), 3467–3501.

Agmon, N., & Kordova, S. (2024). Model for global quality management system in system of systems: Quality management in system of systems project. *Applied System Innovation, 8*(1), 3.

Englander, I., & Wong, W. (2021). *The architecture of computer hardware, systems software, and networking: An information technology approach*. John Wiley & Sons.

Gomaa, A. H. (2025). Enhancing product development excellence through quality management tools: A comprehensive review and integrated conceptual framework. *Intelligent and Sustainable Manufacturing, 2*(2), 10017.

Gopal, G., Suter-Crazzolara, C., Toldo, L., & Eberhardt, W. (2019). Digital transformation in healthcare—Architectures of present and future information technologies. *Clinical Chemistry and Laboratory Medicine, 57*(3), 328–335.

ISO. (2015a). *ISO 9000:2015 quality management systems—Fundamentals and vocabulary*. International Organization for Standardization.

ISO. (2015b). *ISO 9001:2015 quality management systems—Requirements*. International Organization for Standardization.

Jaskó, S., et al. (2020). Development of manufacturing execution systems in accordance with Industry 4.0 requirements: A review of standard- and ontology-based methodologies and tools. *Computers in Industry, 123*, 103300.

Marques, G., Pitarma, R., Garcia, N. M., & Pombo, N. (2019). Internet of things architectures, technologies, applications, challenges, and future directions for enhanced living environments and healthcare systems: A review. *Electronics, 8*(10), 1081.

Moleong, L. J. (2017). *Metodologi penelitian kualitatif*. PT Remaja Rosdakarya.

Owen, A., & Maxwell, P. (2025). Redefining quality assurance workflows: The role of AI in automating test design, maintenance, and reporting.

Pargaonkar, S. (2023). A comprehensive review of performance testing methodologies and best practices: Software quality engineering. *International Journal of Science and Research (IJSR), 12*(8), 2008–2014.

Pressman, R. S. (2019). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill Education.

Schmeleva, A. N. (2017). Evaluation and improvement of the operating efficiency of enterprise quality management system (QMS): Conceptual bases. *Calitatea, 18*(160), 100.

Seider, W. D., et al. (2016). *Product and process design principles: Synthesis, analysis and evaluation*. John Wiley & Sons.

Sommer, A. F., Hedegaard, C., Dukovska-Popovska, I., & Steger-Jensen, K. (2015). Improved product development performance through agile/stage-gate hybrids: The next-generation stage-gate process? *Research-Technology Management, 58*(1), 34–45.

Srinivas, N., Mandaloju, N., & Nadimpalli, S. V. (2024). Leveraging automation in software quality assurance: Enhancing efficiency and reducing defects. *The Metascience, 2*(4), 84–95.

Storey, K. (2022). *Systematic instruction of functional skills for students and adults with disabilities*. Charles C Thomas Publisher.

Sugiyono. (2019). *Metode penelitian kuantitatif, kualitatif, dan R&D*. Alfabeta.

Tsolakis, A. (2026). Integrating quality control, quality assurance, and risk management in organisational change: A conceptual framework.

Usman, M., et al. (2019). Investigating the role of QMS implementation on customers' satisfaction: A case study of SMEs. *IFAC-PapersOnLine, 52*(13), 2032–2037.