sciendo International Conference KNOWLEDGE-BASED ORGANIZATION
Vol. XXX No 1 2024

SYNERGIZING QUALITY AND LEGISLATIVE SOFTWARE REQUIREMENTS: A HOLISTIC APPROACH ACROSS THE SOFTWARE PRODUCT LIFECYCLE

Nicoleta-Mădălina NIȚĂ*, Sebastian-Emanuel STAN, Aurel Mihail ȚÎȚU*****

*** University POLITEHNICA of Bucharest, Romania**
****"Nicolae Bălcescu" Land Forces Academy, Sibiu, Romania**
*****"Lucian Blaga" University of Sibiu, Romania**
madalina.nita12@yahoo.com, sebastian.stan@armyacademy.ro
mihail.titu@ulbsibiu.ro

Abstract: *This scientific research investigates the intricate interplay between quality assurance and legislative software requirements, focusing on the holistic integration of these critical components across the software product lifecycle within the realm of Information Technology (IT). As software development and deployment continue to be shaped by an evolving regulatory landscape and an increasing emphasis on quality, the need to align these two pillars becomes fundamental to the success and compliance of software solutions. This study aims to provide a comprehensive examination of the challenges, opportunities, and strategies associated with synergizing software quality and legislative requirements, thereby offering valuable insights for IT practitioners, organizations, and policymakers. The research delves into the diverse stages of the software product lifecycle, encompassing requirements gathering, design, implementation, testing, deployment, and maintenance, and explores the implications of legislative compliance on each phase. By synthesizing industry standards, regulatory frameworks, and best practices, this study seeks to offer a nuanced understanding of the multifaceted landscape of legislative software requirements and their integration with quality imperatives. Moreover, the research endeavours to provide actionable recommendations and key strategies to support IT practitioners and decision-makers in enhancing their software development practices to meet the dynamic demands of regulatory compliance and quality assurance.*

Keywords: Quality Assurance, Legislative Compliance, Software Product Lifecycle, Regulatory Standards, Project Management

1. Introduction

In the dynamic landscape of Information Technology (IT), the development and deployment of software products are intricately intertwined with the complex web of legislative requirements and the relentless pursuit of high-quality software solutions. This essay endeavors to explore the critical intersection of quality assurance and legislative software requirements, presenting a holistic approach that spans the entire software product lifecycle. By examining the challenges, opportunities, and strategies associated with aligning these vital components, this essay seeks to provide valuable insights for IT practitioners, organizations, and policymakers. The software product lifecycle encompasses a continuum of stages, from initial requirements gathering and design to implementation, testing, deployment, and ongoing maintenance. At each of these stages, the pursuit of software quality must be intricately woven with an

DOI: 10.2478/kbo-2024-0036

unwavering commitment to legislative compliance, encompassing a wide spectrum of regulatory standards, industry guidelines, and legal mandates. The holistic approach requires organizations to navigate this complex terrain by harmonizing quality assurance practices with legislative requirements throughout the software product lifecycle. Ensuring software quality is fundamental to the success and competitiveness of software solutions. Quality assurance practices encompass a range of activities, including requirements validation, design reviews, code inspections, testing, and continuous quality improvement. These practices are essential for delivering software that meets user expectations, functions reliably, and is maintainable over time. Moreover, the pursuit of high-quality software is central to building customer trust, enhancing organizational reputation, and minimizing costly errors and rework. Concurrently, legislative software requirements impose a layer of regulatory compliance that organizations must navigate to ensure that their software solutions meet legal and industry-specific standards. These requirements can span data privacy regulations, security standards, accessibility guidelines, industry-specific mandates, and international standards. Navigating this intricate web of legislative requirements is essential for organizations to avoid legal ramifications, protect user data, and ensure that their software products meet the expectations of the regulatory landscape. The challenges of synergizing quality and legislative software requirements across the software product lifecycle are multifaceted. Furthermore, the complexity of this intersection is compounded by the rapid pace of technological advancement, which continuously introduces new challenges and opportunities for software development and compliance. In response to these challenges, organizations must adopt a holistic approach that integrates quality assurance and legislative compliance across the software product lifecycle. This holistic approach requires a thorough understanding of regulatory frameworks, industry best practices, and the implications of legislative requirements on various facets of the software development process. It also demands the adoption of robust quality assurance methodologies and the incorporation of compliance considerations into every phase of the software product lifecycle. By adopting a holistic approach, organizations can strive to cultivate software solutions that not only meet the highest quality standards but also adhere to the ever-evolving legislative landscape. This integrated approach offers numerous benefits, including enhanced risk management, improved software reliability, increased customer satisfaction, and a competitive edge in the market. Moreover, by aligning quality and legislative software requirements, organizations can streamline their development processes, reduce the likelihood of legal issues, and demonstrate a commitment to ethical and compliant software practices. In conclusion, the holistic integration of quality assurance and legislative software requirements across the software product lifecycle is fundamental to the success, compliance, and competitiveness of software solutions in the contemporary IT landscape. By understanding the challenges, opportunities, and strategies associated with this critical intersection, organizations can strive to navigate the complex terrain of software development while meeting the dynamic demands of regulatory compliance and quality assurance. This integrated approach not only fosters a deeper understanding of the multifaceted landscape of legislative software requirements but also empowers organizations to cultivate software solutions that are both compliant and of the highest quality.

2. Software Requirements and Recommendations

The performance, dependability, and general efficacy of a software product are all determined by its software quality

criteria [1]. In the current digital era, software applications are essential in many different fields, thus it's critical to comprehend and meet quality criteria to provide software solutions that are competitive and successful. A complex set of characteristics that should ensure the quality of the software product must be met in order to achieve the expected results and those categories can be split into:

- Essential Functions
  The meeting of functional specifications is one of the core components of software quality criteria. This entails making certain the program reliably and effectively completes the desired functions. In order to satisfy user expectations and business objectives, it is essential to have thorough and well-documented functional requirements, covering input/output behavior, data processing, and system interactions.
- Dependability and Efficiency
  User happiness and organizational success are directly impacted by reliability and performance, which are essential quality standards. A dependable software program should constantly provide the desired functionality without glitches or malfunctions in the system [2]. Furthermore, the software's speed, scalability, and resource usage are determined by performance requirements, which provide the best possible user experience and operational effectiveness.
- User-friendliness and Experience
  Considerations for usability and user experience are included in user-centric quality criteria. People should be able to engage with a software product easily and intuitively, making it accessible and user-friendly [3]. Positive user experiences are influenced by elements including responsiveness, accessibility, visual design, and navigation, which in turn affect consumer adoption and retention.
- Safety and Observance
  Software quality is inextricably linked to security and compliance requirements in an era of growing cyber threats and strict laws. Safeguarding sensitive data and upholding user and stakeholder confidence require strong security protocols, data protection procedures, and compliance with industry norms and legislation [4].
- Maintainability and Scalability
  Requirements for scalability and maintainability deal with the software's capacity to adapt to changes and growth over time [5]. Software that is scalable need to be able to adjust to growing user numbers and changing business requirements without sacrificing functionality. Requirements for maintainability also highlight how simple it is to update, maintain, and modify software, which lowers total cost of ownership and promotes long-term sustainability.

ISO/IEC 25030 is a standard that focuses on software product quality requirements and evaluation [6]. It provides a framework for defining quality requirements and assessing the quality of software products, aiming to enhance the overall quality and reliability of software systems [7]. This standard offers guidelines and best practices for identifying, specifying, and evaluating quality characteristics, such as functionality, reliability, usability, efficiency, maintainability, and portability.

ISO/IEC 25030 plays a crucial role in guiding organizations and software developers in understanding and meeting the diverse quality requirements of software products. By leveraging this standard, stakeholders can gain a comprehensive understanding of the essential quality attributes that contribute to the success and effectiveness of software solutions.

Additionally, ISO/IEC 25030 provides a structured approach to evaluating and measuring the attainment of quality characteristics, enabling organizations to make informed decisions and improvements throughout the software development lifecycle [8].

Figure 1 depicts a personal perspective about the processes that are conducted in the quality assurance branch, starting with the requirements coming directly from the customer and finalizing with some quality requirements in software development.

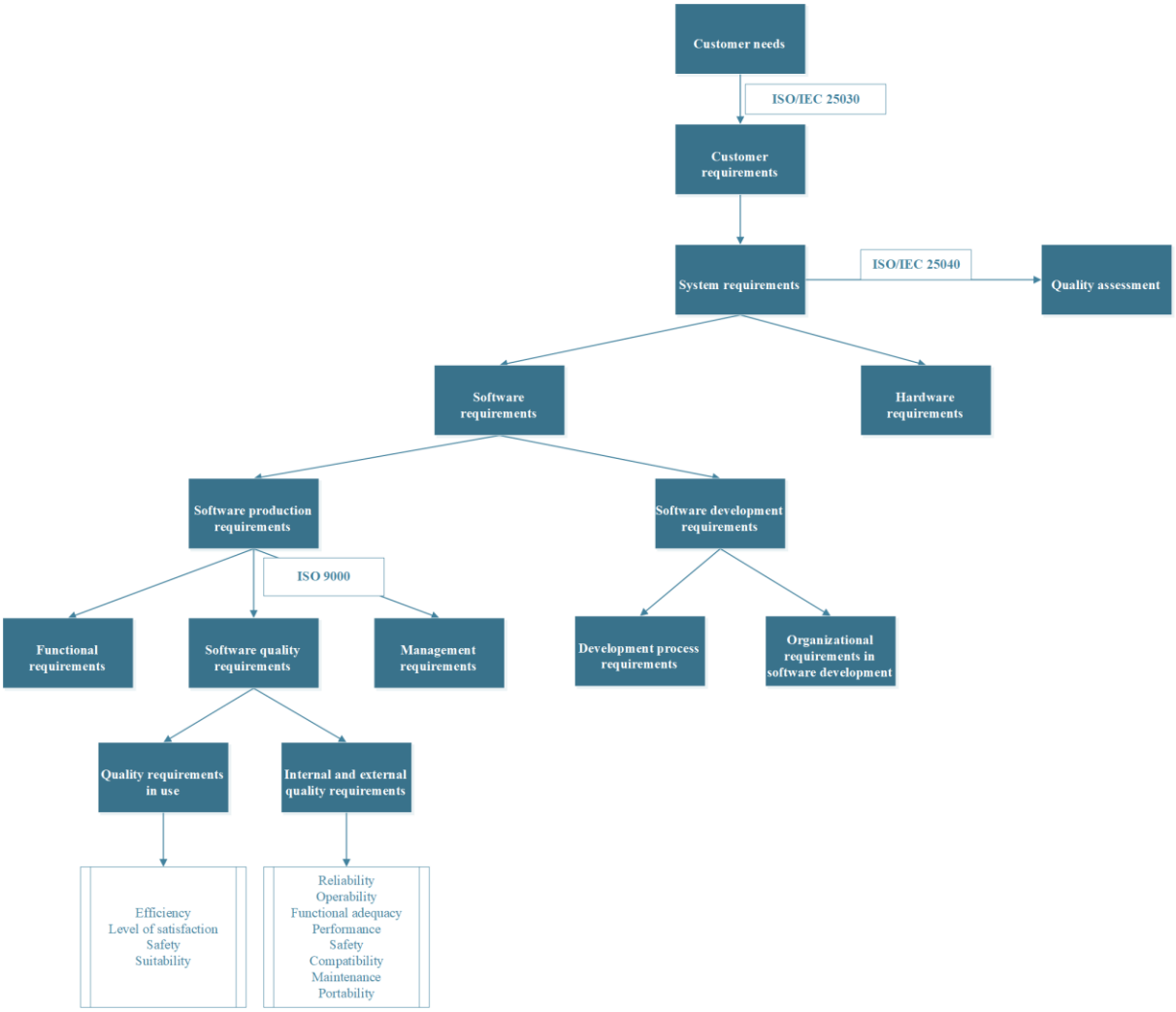

Figure 1 Hierarchy of processes that make up Quality Assurance and Regulatory Compliance in IT (Source: own contribution)

These requirements in Figure 1 are further broken down, analysed and transformed into general categories such as those in Table 1:

Table 1 Description of the processes that make up Quality Assurance and Regulatory Compliance in IT branch

Process	Subprocess			
P1. System requirements *(represents the totality of the needs of the software system at a holistic level)*	**S1. Software requirements**	**S1.1. Software production requirements** *(is done using ISO9000)*	**S1.1.1. Functional requirements** *(describe the behaviour of the software as it was designed)*	
			S1.1.2. Software quality requirements *(to be achieved using ISO/IEC 25010)*	**S1.1.2.1. Quality in use requirements**
				S1.1.2.2. Internal and external quality requirements
			S1.1.3. Management requirements *(such as budget, delivery, development time constraints)*	
		S1.2. Software development requirements	**S1.2.1. Development process requirements** *(refers to methodologies that apply to this area)*	
			S1.2.2. Organization's requirements in software development *(is achieved through specific procedures created and applied by the organization)*	
	S2. Hardware requirements *(such as restrictions or limitations of the equipment on which the software product will be implemented)*			
P2. Quality assessment (is carried out using ISO/IEC 25040 and aims to prepare the strategy for refining and improving requirements)				

3. Life Cycle of a Software Product Regarding Quality

The lifecycle of a software product is a complex process that involves various stages from development to deployment and maintenance. Quality is a crucial aspect of this lifecycle, as it directly impacts the performance, reliability, and user satisfaction of the software product. In the following sentences, the lifecycle of a software product from a quality perspective will be explored, highlighting the key stages and considerations involved.

1. Requirements Gathering and Analysis:

The lifecycle of a software product begins with the gathering and analysis of requirements. Quality is established at this stage by ensuring that the requirements are clear, comprehensive, and aligned with the needs and expectations of the users. Proper requirements gathering and analysis help in setting the foundation for a high-quality software product by defining the scope, functionality, and performance expectations.

2. Design and Development:

The design and development phase of the software lifecycle is where the actual creation of the software product takes place. Quality is embedded in the design and development process through various practices such as adherence to coding standards, use of design patterns, and implementation of best practices. Quality assurance techniques such as code reviews, unit testing, and integration testing play a crucial role in ensuring that the software product is being developed to meet the specified requirements and quality standards.

3. Testing and Quality Assurance:

The testing and quality assurance phase is a critical stage in the software lifecycle where the focus is on validating the functionality, performance, and reliability of the software product. Various types of testing such as functional testing, performance testing, security testing, and user acceptance testing are conducted to identify and address defects and ensure that the software product meets the desired quality criteria. Quality assurance practices such as test automation, continuous integration, and defect tracking are essential in maintaining and improving the overall quality of the software product.

4. Deployment and Release:

Once the software product has been developed and thoroughly tested, it is deployed and released to the users. Quality in deployment and release involves ensuring that the software product is packaged, installed, and configured correctly to minimize the risk of errors and disruptions. Continuous monitoring and feedback mechanisms are put in place to gather insights into the performance and usability of the software product in a real-world environment, which helps in identifying areas for improvement and maintaining high quality.

5. Maintenance and Support:

The lifecycle of a software product extends beyond its initial release, as ongoing maintenance and support are essential for ensuring its continued quality and relevance. Maintenance activities such as bug fixes, updates, and enhancements are carried out to address issues and improve the overall quality of the software product. Support services are provided to assist users in resolving issues and maximizing the value derived from the software product. In software systems engineering, projects most often involve the creation of a website, software or mobile application [9]. Depending on the knowledge management strategies the organisation adopts, the lifecycle of the software product created can be greatly influenced, as a very good knowledge transfer based on well-structured documentation results in a shorter project planning and sometimes even execution time, due to the IT bases, which can be adapted instead of being created from scratch [10].

Keeping in mind the quality criteria that a software product must meet, the lifecycle of a software product regarding quality is presented in the Figure 2:

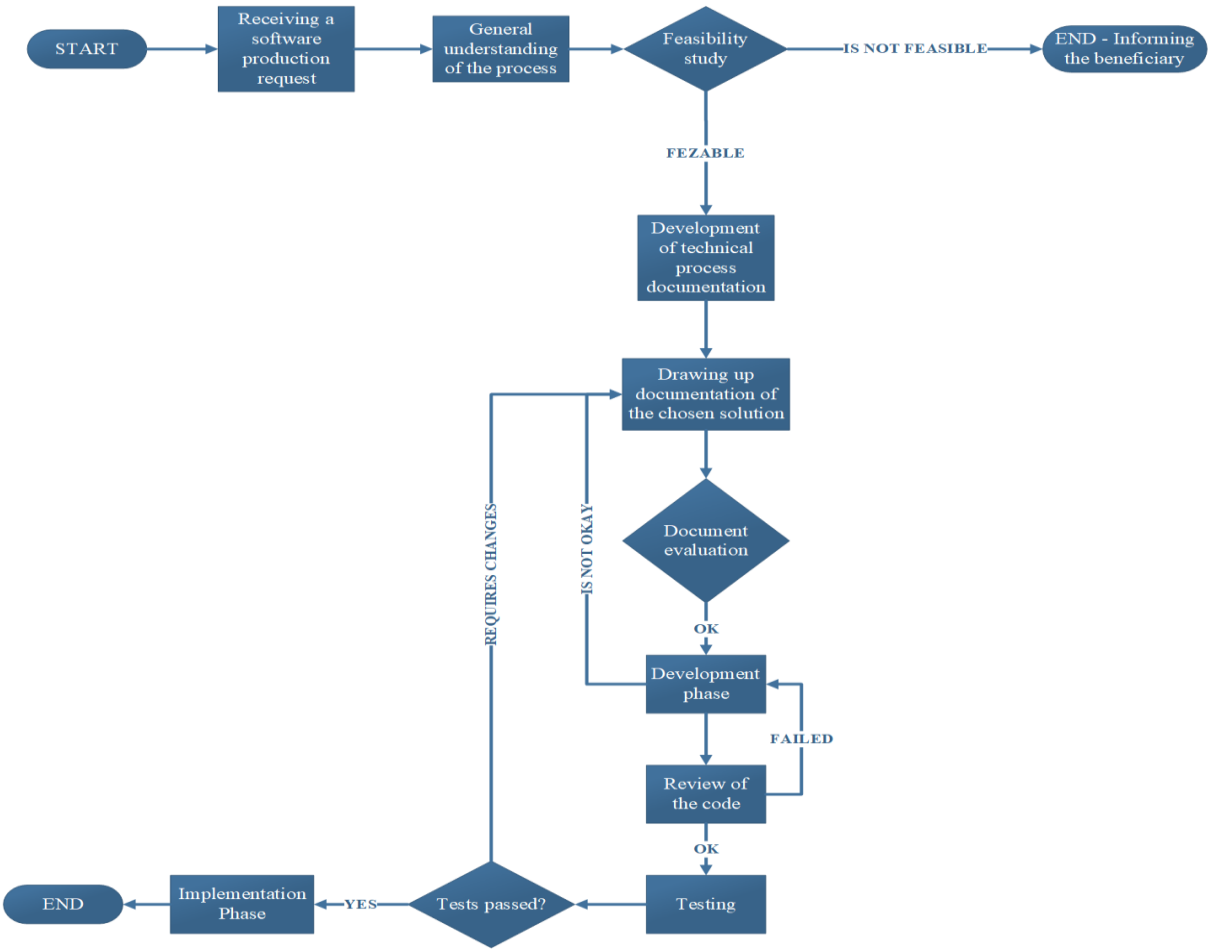

Figure 2 The lifecycle of a software product regarding quality
(Source: own contribution)

This diagram in Figure 2 reflects the processes present in software systems engineering projects. This illustration is also valid when a new product is created from scratch, but also when changes are made to an existing product and materialised in the form of new functionality or a new version.

4. Conclusions

In conclusion, the holistic integration of quality assurance and legislative software requirements across the software product lifecycle is essential for organizations to navigate the complex terrain of software development effectively. By understanding the multifaceted landscape of legislative requirements and their integration with quality imperatives, organizations can strive to cultivate software solutions that not only meet the highest quality standards but also adhere to the ever-evolving legislative landscape.

The challenges associated with aligning quality and legislative software requirements are significant, but the adoption of a holistic approach offers numerous benefits. This integrated approach enhances risk management, improves software reliability, increases customer satisfaction, and provides a competitive edge in the market. Moreover, by aligning quality and legislative software requirements, organizations can streamline their development processes, reduce the likelihood of legal issues, and demonstrate a commitment to ethical and compliant software practices.

Furthermore, the holistic approach enables organizations to harmonize quality assurance practices with legislative requirements throughout the software product lifecycle. By adopting robust

quality assurance methodologies and incorporating compliance considerations into every phase of the software development process, organizations can ensure that their software solutions not only meet the highest quality standards but also adhere to the ever-evolving legislative landscape.

Ultimately, by embracing a holistic approach to synergize quality and legislative software requirements, organizations can navigate the intricate intersection of quality and compliance effectively. This approach fosters a deeper understanding of the critical components across the software product lifecycle and empowers organizations to develop software solutions that are both compliant and of the highest quality, thereby contributing to their success, compliance, and competitiveness in the contemporary IT landscape.

References List

[1] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice (SEI Series in Software Engineering) 3rd Edition*. 2012.

[2] B. W. Boehm, "Software risk management: principles and practices", *IEEE Software*, vol. 8, no. 1, p. 32, 1991.

[3] B. Singh and S. Gautam, "The Impact of Software Development Process on Software Quality: A Review", Dec. 2016, pp. 666–672. doi: 10.1109/CICN.2016.137.

[4] R. S. Pressman and B. Maxim, *Software Engineering: A Practitioner's Approach 8th Edition*. 2014.

[5] E. Kamsties, J. Horkoff, and F. Dalpiaz, Eds., *Requirements Engineering: Foundation for Software Quality*, vol. 10753. in Lecture Notes in Computer Science, vol. 10753. Cham: Springer International Publishing, 2018. doi: 10.1007/978-3-319-77243-1.

[6] "Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Quality model overview and usage." 2024.

[7] "Why Software Fails", IEEE Spectrum. Accessed: Aug. 26, 2022. [Online]. Available: https://spectrum.ieee.org/why-software-fails

[8] I. Sommerville, *SOFTWARE ENGINEERING*. 2011.

[9] C. Y. Laporte and A. April, *Software Quality Requirements*. IEEE, 2018. doi: 10.1002/9781119312451.ch3.

[10] K. Esaki, "Verification of Quality Requirement Method Based on the SQuaRE System Quality Model," vol. 2013, Jan. 2013, doi: 10.4236/ajor.2013.31006.