International Journal of Advance Research in Science and Engineering
Vol. No.6, Issue No. 05, May 2017
www.ijarse.com
IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

An Overview of Challenges in Software Testing

Anish Patel

Assistant Professor, Department of Computer, Seva Sadan Mahavidyalaya, Burhanpur (M.P.)

Abstract

The issue of software quality remains a significant challenge within the software industry, contributing to substantial setbacks and, in some cases, company closures. Addressing and resolving the root causes of these quality problems is crucial, especially given the pressure on timely product deployment. This paper explores key challenges in software quality assurance and testing faced by small and medium-sized software companies globally. It identifies various categories of challenges and the stakeholders involved. Despite advancements in testing tools, elements, and processes, significant testing challenges persist. This research highlights these challenges and suggests areas where software engineers can focus their efforts to improve and overcome these issues.

Keywords: *Software Quality Assurance (SQA), Testing, Resources, Challenges, Software Companies, Developers.*

I. Introduction

Software Quality Assurance (SQA) plays a crucial role in ensuring the quality of software by integrating a range of activities to establish, maintain, and verify processes, procedures, and standards suited to a project. It operates concurrently with the software development process, aiming to enhance the development methodology to prevent significant issues before they arise. SQA functions as a comprehensive activity that spans the entire software lifecycle. Given the pervasive nature of software in daily life, failures can have serious repercussions, particularly in safety-critical applications and business operations. For instance, Ashton-Tate, once a leading software company, ceased to exist due to quality issues with its software.

Testing is a fundamental aspect of ensuring quality and is a significant part of quality management. It is a costly endeavour, with estimates indicating that up to 80% of software development costs are allocated to identifying and fixing defects (ETH Zurich, 2010). To address these costs, various tools and techniques have been developed to enhance the testing process. This research examines the challenges faced in software quality assurance and testing, focusing on procedural, programming, and

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

managerial issues. It traces these challenges from the initial stages where defects first appear to the final stages of development.

The study emphasizes the importance of improving the software engineering knowledge base to address SQA challenges and highlights the financial implications of late defect detection and resolution. A detailed analysis of the cost associated with testing is provided, including a table of data to illustrate these expenses. The aim is to evaluate whether the costs associated with testing reflect challenges that need to be addressed to enhance the cost-effectiveness of testing practices within software development firms.

A recent report by the National Institute of Standards and Technology estimated that software failures cost the U.S. economy approximately $59.5 billion annually, which equates to about 0.6% of the Gross Domestic Product. The report indicated that more than a third of these costs, around $22.2 billion, could potentially be avoided with improved testing infrastructure that allows for earlier and more effective defect detection. Currently, over half of software errors are detected late in the development process or during post-sale usage.

The increasing complexity of software systems has exacerbated the challenge of testing, making it increasingly expensive and resource-intensive. Software testing, which typically involves generating test inputs, executing tests, and verifying outcomes, becomes more challenging as software systems grow in size and complexity. The cost of testing accounts for roughly 50% of software development expenses. Effective test case design is crucial but increasingly difficult to achieve manually for complex systems. Although automated testing can reduce manual effort, existing techniques often require specific inputs that may not always be available.

This study explores the challenges faced by software companies in quality assurance and testing. It emphasizes that while testing is a critical part of the development process, it must be integrated into all phases of development rather than being treated as a separate activity. The research presents updated strategies for addressing software testing challenges and offers recommendations for best practices. It underscores that modern software testing is an integral part of the entire development process, not merely a final-stage activity. Effective testing requires optimal resource utilization to ensure a high-quality end product, taking into account the fixed parameters of resources, time, and scope.

This paper is based on a study conducted in the Information Technology industry from February 2016 to December 2016, aiming to provide valuable insights and recommendations for improving software testing practices.

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

II. Key Challenges in Software Testing

2.1 **Testing the Entire Application**

Testing a complete application in its entirety is often impractical due to the vast number of possible test combinations. Attempting to cover every potential scenario can be time-consuming and may delay the product's release. As a result, prioritizing test cases based on risk and impact becomes essential.

2.2 **Interaction with Developers**

Managing relationships between testers and developers can be challenging. Conflicts may arise when testers and developers disagree on issues or test results. Effective communication, problem-solving skills, and the ability to analyze and address disagreements constructively are crucial for resolving such conflicts and maintaining a positive working environment.

2.3 **Regression Testing**

As a project evolves, regression testing can become overwhelming. Ensuring that new changes do not adversely affect existing functionalities requires extensive effort. The challenge lies in balancing the management of changes with the need to track and address bugs in previously tested features.

2.4 **Time Constraints in Testing**

When faced with tight deadlines, there is a tendency for testers to focus more on completing tasks rather than on the quality of the product. Meeting deadlines can sometimes compromise the thoroughness of testing. It is crucial to strike a balance between speed and quality to ensure that critical quality tasks are not overlooked.

2.5 **Understanding Requirements**

Effective testing relies heavily on a clear understanding of the requirements. Testers who do not fully grasp the requirements may struggle to conduct comprehensive tests. Good listening skills and the ability to interpret requirements accurately are essential for successful testing.

2.6 **Deciding When to Stop Testing**

Determining the right point to stop testing can be challenging. Skilled testers need to evaluate the testing processes and their importance to make informed decisions about when to conclude testing. This decision involves balancing the need for thoroughness with practical constraints.

III. Categories of Challenges in Software Quality

In my experience working with various local and international software companies, I have identified several categories of challenges impacting software development. These challenges are grouped into

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

three main categories, each encompassing multiple specific issues. These challenges can directly or indirectly affect a software company's overall productivity and growth:

1. **Procedural Challenges**: These involve difficulties in implementing and adhering to testing procedures and methodologies. Inadequate processes or inconsistent application of procedures can hinder effective quality assurance.
2. **Programming Challenges**: These include issues related to coding practices, such as defects in code, integration problems, and difficulties in managing code changes. Such challenges can impact the stability and performance of the software.
3. **Managerial Challenges**: These pertain to issues in managing testing activities, including resource allocation, prioritization of testing tasks, and balancing quality with deadlines. Effective management is essential to address these challenges and ensure successful software quality assurance.

3.1 Software Requirement Challenges

Clients are increasingly focused on software quality, but quality issues often begin with the collection of software requirements. Problems such as improper collection methods, incomplete or unclear requirements, insufficient time allocated for gathering requirements, and a lack of commitment can all contribute to quality failures. A primary challenge is ensuring that requirements are gathered correctly and in a timely manner (Teodoro, 2009).

Sometimes, those responsible for gathering requirements may neglect the crucial processes of elicitation and validation. In some organizations, there might not be dedicated personnel for requirement collection; instead, programmers or other staff members may handle this task, often leading to unstructured and incomplete requirements. This oversight can adversely affect the entire development lifecycle.

It is perplexing why such critical aspects of requirements collection are often mishandled, despite their importance. Observations over the past 15 years reveal several issues that are not always documented in literature or reports. For instance, in-house software development frequently encounters challenges with requirements that change unexpectedly, leading to logical and functional errors due to missed or incorrectly implemented requirements. These issues contribute to diminished software quality.

3.2 The Requirements Collection Period

The period during which requirements are collected is crucial. Incomplete requirements collection can adversely affect subsequent project phases. Projects can fail if the requirements are not clearly defined from the outset. Often, requirements collectors may bypass certain requirements if they are deemed

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

out of scope, too complex, or time-consuming. This avoidance can occur unconsciously, where Software Requirements Engineers (SREs) might inadvertently neglect to capture all client needs.

Clients may also contribute to this issue by providing requirements incrementally, starting with a prototype and then supplying additional requirements or changes as needed. This approach can lead to gaps in requirements and, consequently, impact software quality. Both the requirements collectors and the clients share responsibility for these quality issues.

IV. Stakeholder Perspectives on Challenges

In this study, I aimed to identify the reasons behind software quality issues and the key stakeholders responsible for these problems (Jeff, 2005). I found that various stakeholders, including software requirement engineers and clients, significantly impact software quality. For example, some users may resist adopting new software or abandoning legacy systems, leading to non-cooperation and provision of incorrect requirements, which ultimately affects software quality and project success. Based on my observations, I have categorized these stakeholder-related challenges into four main types:

4.1 Developer/Programmer Challenges

Developers often face several obstacles related to software quality assurance and testing. My analysis reveals that developers frequently prioritize completing coding tasks over performing basic quality checks, including unit testing. This tendency, coupled with overconfidence, can result in a lack of thorough testing before release. Developers working under pressure may only conduct minimal functional testing and may not allocate sufficient time for comprehensive quality assurance. Additionally, some programmers may lack a foundational understanding of Software Quality Assurance (SQA) and the Software Development Life Cycle (SDLC), particularly if their background is in non-technical fields. They might have received only brief training in specific programming languages, which can lead to poor coding practices and subsequent software quality issues.

4.2 Company-Level Risks

Software companies often face pressures to meet tight deadlines, leading to insufficient time allocated for thorough testing. Some companies may opt to defer comprehensive testing until later stages, such as during the support and maintenance phase or the next release cycle. Furthermore, smaller firms or private companies with in-house development teams might not have dedicated SQA departments or quality assurance engineers, reflecting a lack of emphasis on quality assurance.

4.3 Client-Side Threats

Clients frequently focus on minimizing costs, often perceiving that higher-quality software will come with a higher price tag. In many cases, clients request software that meets only their basic needs and

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

are less concerned with achieving high quality. This mind-set is especially common in developing countries, where clients may intentionally downplay the importance of quality assurance (Ricardo, 2007).

4.4 Vendor Negligence

Vendor negligence can also adversely affect software quality. Third-party implementation vendors might rush to complete projects, opting for quick fixes that can introduce future problems and require substantial rework. Discrepancies between software and current business processes can lead to operational issues. For example, an Oracle E-Business Suite implementation project faced difficulties in meeting deadlines and budget constraints due to vendor-related challenges.

V. Where Does Software Quality Suffer?

Determining why and where software quality suffers is essential for improving software development practices. Effective software quality assurance should be integrated from the project's inception through to its completion. Key phases where quality issues may arise include project planning, budgeting, and requirements gathering. According to a survey by Testplant's Application Crisis Research, 70% of businesses report feeling pressured to innovate, with about half releasing software without adequate testing and 40% releasing software with no testing at all (John, 2017).

Key areas where software quality may suffer include:

- **Objective and Scope**: Misalignment in objectives and scope can lead to quality issues.
- **Business Value and Cost/Benefit Analysis**: Inadequate analysis can impact the project's success.
- **Estimation**: Poor estimation practices can affect project timelines and resources.
- **Requirements**: Incomplete or incorrect requirements can cause significant quality problems.
- **Design and Coding**: Deficiencies in design and coding can propagate defects.
- **Testing**: Insufficient or ineffective testing can fail to identify critical issues.
- **Integration**: Integration issues can affect overall system functionality.
- **Implementation**: Problems during implementation can impact the final product's performance.
- **Training**: Inadequate training can affect the effective use of the software.
- **Change Management**: Poor change management can lead to instability and quality issues.

VI. Barriers to Effective Testing

Despite diligent efforts and the benefits of an agile approach, Quality Assurance (QA) teams often receive tasks for testing only at the last moment or at the end of a sprint. To ensure a high-quality

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

application, there are strategies that can be employed even when time is tight. Ideally, while waiting for a testable version of a feature, QA should use this period to thoroughly understand the feature, review any available documentation, and consult with relevant members of the product and development teams. This preparation allows QA to develop comprehensive test cases and to be ready for execution once the feature is delivered. Effective use of the waiting period involves formulating a detailed testing strategy and documentation.

Regarding supporting documentation, the quality and completeness of such materials can vary significantly. While most organizations require functional, non-functional, and technical specifications, these documents might not always provide all the necessary information for thorough testing. A proactive tester will seek additional details beyond what is provided in the documentation to fully comprehend the feature and ensure that all aspects are tested effectively.

VII. Conclusion

This paper aims to highlight the challenges associated with Software Quality Assurance (SQA), the financial impacts of quality failures, and their effects on company growth, while also proposing solutions to address these issues. The research identifies key challenges faced by software companies and offers solutions to mitigate these problems. Despite the software industry's extensive history of over 70 years, many companies struggle to maintain consistent success. While some firms have achieved significant milestones with large projects, they often face difficulties in sustaining success and profitability over time. Some companies experience fluctuations in project volume and revenue, with occasional periods of financial loss or even closure.

The reasons for these challenges are multifaceted and represent significant concerns for the software industry. Understanding why companies fail to achieve consistent returns on investment (ROI) is crucial. Although this research has identified root causes and suggested solutions based on established software engineering practices, the scope was limited due to constraints in time and resources.

To improve productivity, quality, and reliability in software products, it is essential for companies to focus on quality assurance. This paper aims to assist the software industry in its ongoing efforts to enhance its business processes and address quality issues. The ultimate goal is to increase stakeholder awareness and ensure that software quality is given the highest priority, as this is critical for both company profitability and stakeholder benefits. Addressing these quality issues and promoting awareness among all stakeholders is vital for the continued success and growth of the software industry.

IJARSE
ISSN (O) 2319 - 8354
ISSN (P) 2319 - 8346

References

- Beizer, B. (1995). *Software Testing Techniques*. Van Nostrand Reinhold.
- Black, R. (2009). *Managing the Testing Process*. Wiley.
- Boehm, B. W. (2006). *A Spiral Model of Software Development and Enhancement*. ACM SIGSOFT Software Engineering Notes.
- Boehm, B. W., & Turner, R. (2003). *Balancing Agility and Discipline: A Guide for the Perplexed*. Addison-Wesley.
- Dorfman, M. & Thayer, R. H. (1990). *Software Engineering: Principles and Practice*. Wiley.
- ETH Zurich (2010). *Software Engineering Cost Analysis*.
- IEEE (2010). *IEEE Standard for Software and Systems Test Documentation*. IEEE Std 829-2008.
- John, S. (2017). *Application Crisis Research: The Cost of Testing*. Testplant.