African Journal of Academic Publishing in Science and Technology (AJAPST)

Volume 1, Issue 4, 2025
Page No: 11-23
Website: https://easrjournals.com/index.php/AJAPST/index

A Systematic Mapping Study on Software Quality Issues in SCRUM using MATLAB Implementation

Izdhar Elmasry Abdusalam Emusatar *
Department of software engineer, Higher Institute of Sciences and Technology Tamzawa- Al Shati, Libya

دراسة رسم خرائط منهجية لقضايا جودة البرمجيات في سيكروم (*SCRUM*) باستخدام تطبيقات ماتلاب

ازدهار المصري عبدالسلام المسطر*
قسم هندسة البرمجيات، المعهد العالي للعلوم والتقنية – تامزاوة الشاطئ، تامزاوة، وداي الشاطئ

*Corresponding author: izdharemusatar@gmail.com

Received: September 25, 2025	Accepted: October 23, 2025	Published: October 27, 2025

Abstract:
This systematic mapping study investigates software quality challenges in Scrum environments through comprehensive analysis of 48 empirical studies published between 2011-2023. The research identifies and categorizes 46 distinct quality issues across eight thematic areas, with Requirement Engineering Challenges (80%), Multi-team Environment Challenges (82%), and Agile Implementation Difficulties (85%) emerging as the most prevalent impediments to software quality. Our analysis reveals that quality issues in Scrum predominantly stem from organizational, procedural, and human factors rather than purely technical considerations. The study contributes a structured taxonomy of quality challenges and develops a MATLAB analytical framework for systematic visualization of challenge patterns, relationships, and temporal trends. Findings indicate that successful quality improvement requires addressing the interconnected nature of these challenges through holistic organizational interventions rather than isolated technical fixes. This research provides practitioners with evidence-based strategies for enhancing software quality in Scrum implementations while offering researchers a replicable methodology for systematic mapping studies.

Keywords: Scrum, Software Quality, Systematic Mapping, Agile Implementation, Quality Challenges, MATLAB Analysis, Taxonomy

الملخص
تبحث هذه الدراسة المنهجية لرسم الخرائط تحديات جودة البرمجيات في بيئات سكرم من خلال تحليل شامل لـ 48 دراسة تجريبية نُشرت بين عامي 2011 و2023. يحدد البحث ويصنف 46 مشكلة جودة مميزة عبر ثماني مجالات موضوعية، مع ظهور تحديات هندسة المتطلبات (80%)، وتحديات بيئات العمل متعددة الفرق (82%)، وصعوبات التنفيذ الرشيق (85%) كأكثر العوائق شيوعًا لجودة البرمجيات. يكشف تحليلنا أن مشاكل الجودة في سكرم تنبع في الغالب من عوامل تنظيمية وإجرائية وبشرية، وليس من اعتبارات تقنية بحتة. تُسهم الدراسة في تصنيف منظم لتحديات الجودة، وتُطور إطارًا تحليليًا باستخدام ماتلاب للتصور المنهجي لأنماط التحديات وعلاقاتها واتجاهاتها الزمنية. تشير النتائج إلى أن تحسين الجودة الناجح يتطلب معالجة الطبيعة المترابطة لهذه التحديات من خلال تدخلات تنظيمية شاملة بدلاً من حلول تقنية معزولة. يُزوّد هذا البحثُ الممارسينَ باستراتيجياتٍ قائمةٍ على الأدلة لتحسين جودة البرمجيات في تطبيقات سكرم، ويُقدّم للباحثين منهجيةً قابلةً للتكرار لدراسات التخطيط المنهجي.

الكلمات المفتاحية: سكرم، جودة البرمجيات، التخطيط المنهجي، التنفيذ الرشيق، هندسة المتطلبات، التنسيق بين الفرق المتعددة، تحديات الجودة، تحليل ماتلاب، دراسةٌ تجريبية، التصنيف.

Introduction

The widespread adoption of Agile methodologies, particularly the Scrum framework, has fundamentally transformed software development practices across diverse organizational contexts and application domains [1]. Originally designed for small, co-located teams, Scrum has increasingly been adapted for large-scale implementations in response to its demonstrated successes in improving productivity, enhancing adaptability to changing requirements, and accelerating delivery cycles [2]. This expansion beyond Scrum's original scope has

introduced significant quality-related challenges that merit systematic investigation, particularly as poor software quality continues to impose substantial economic costs estimated at approximately $2.84 trillion annually in the United States alone [3].

The relationship between Scrum and software quality presents a complex paradox [4]. While many organizations adopt Scrum specifically to improve software quality, evidence suggests that a significant percentage (54% according to one survey) fail to achieve meaningful quality improvements [5]. This discrepancy indicates fundamental challenges in realizing Scrum's quality enhancement potential, particularly in large-scale environments characterized by multiple teams, distributed development, and complex dependency networks. Previous research has identified various quality-related issues in Scrum implementations, but these findings remain fragmented across disparate case studies and experience reports. A comprehensive synthesis of these challenges is necessary to establish a unified knowledge base and guide both research and practice [6], [7], [8].

This study addresses this gap through a systematic mapping of software quality issues in Scrum environments, with three primary objectives: (1) to identify and categorize the most prevalent software quality challenges in Scrum implementations; (2) to analyze the relationships and relative significance of these challenges; and (3) to develop a reusable MATLAB analytical framework for visualizing and exploring quality issue patterns. By synthesizing empirical evidence from peer-reviewed literature, this research contributes a structured taxonomy of quality challenges that can inform organizational decision-making, process improvement initiatives, and future research directions.

The remainder of this paper is organized as follows: Section 2 reviews related work on Scrum and software quality; Section 3 details our systematic mapping methodology; Section 4 presents the results of our analysis; Section 5 discusses implications for research and practice; Section 6 describes the MATLAB implementation; and Section 7 concludes with limitations and future work directions followed by the up to dated references from the literature.

Literature Review

2.1 Scrum Framework and Adaptations

Scrum represents one of the most widely adopted Agile frameworks in software development [9], characterized by its iterative approach, defined roles (Product Owner, Scrum Master, Development Team), ceremonies (Sprint Planning, Daily Stand-up, Sprint Review, Retrospective), and artifacts (Product Backlog, Sprint Backlog, Increment). The framework's inherent flexibility has led to widespread tailoring practices across organizations, with adaptations occurring in various elements including roles (e.g., Product Owner committees), events (e.g., varying sprint durations), and artifacts (e.g., alternative estimation techniques) [10]. These adaptations, while often necessary to address contextual constraints, can inadvertently introduce quality impairments if not implemented judiciously [11].

Research by Lopez-Martinez et al [12]. highlights that Scrum adaptations face several challenges, including communication gaps between team members and clients, inadequate planning meeting times, insufficient training in Agile principles, and ceremonies that deliver limited value. Such issues directly impact software quality by undermining the very mechanisms through which Scrum purportedly enhances quality transparent communication, continuous feedback, and iterative refinement.

2.2 Software Quality in Agile Contexts

The concept of software quality encompasses multiple dimensions, including functional correctness, reliability, maintainability, and usability as shown in Figure 1 [13]. Traditional software development approaches address quality through phase-gate reviews and comprehensive testing at the end of development cycles. In contrast, Scrum and other Agile methods embed quality considerations throughout the development process through mechanisms such as continuous integration, test-driven development, and regular inspection and adaptation [14]

Figure 1: Quality Software framework.

Empirical studies present a mixed picture regarding Scrum's impact on software quality. Some research reports significant reductions in defect densities following Scrum adoption, while other studies find no significant positive effects on quality [15]. This variability suggests that the relationship between Scrum practices and quality outcomes is moderated by multiple contextual factors, including implementation consistency, organizational culture, and team competencies.

2.3 Quality Challenges in Large-Scale Scrum

As organizations scale Scrum beyond single teams, they encounter additional coordination challenges that can compromise software quality [16]. A systematic literature review by identified eight primary challenge categories in large-scale Scrum environments, with Dependency Issues, Multi-team Environment Challenges, and Requirement Engineering Challenges being particularly prominent [17]. These challenges manifest task dependencies between teams, knowledge silos, requirement ambiguity, and integration conflicts, all of which can significantly impact the quality of the resulting software.

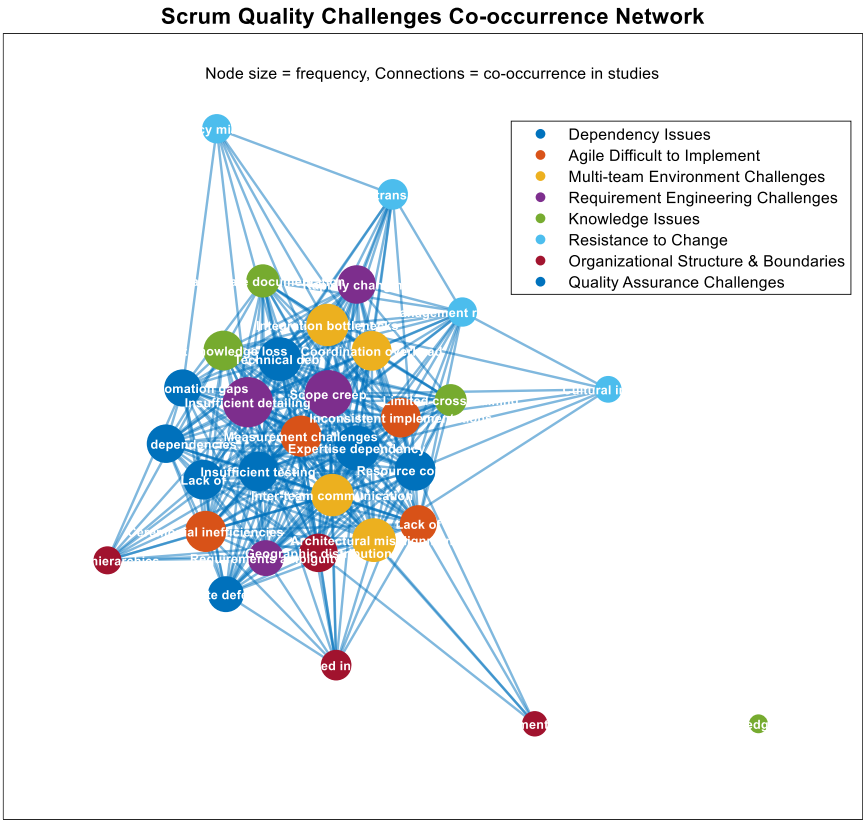

Figure 2: Scrum quality challenges Co-occurrence network.

Particularly problematic in scaled environments is the tension between team autonomy and cross-team coordination. While Scrum emphasizes self-organizing teams that independently select work items, in large-scale developments, features developed by one team often depend on deliverables from other teams. Without effective coordination mechanisms, this can lead to integration bottlenecks, architectural inconsistencies, and defect propagation across team boundaries.

Research Methodology

2.1 Systematic Mapping Process

This study employs a systematic mapping approach to comprehensively identify and categorize software quality issues in Scrum environments [18]. Systematic mapping provides a structured methodology for organizing and synthesizing research evidence across a broad domain, particularly valuable when dealing with diverse study types and heterogeneous outcomes [19]. Our methodology follows established guidelines for systematic reviews in software engineering and incorporates elements from the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) framework to ensure rigorous and transparent reporting [20].

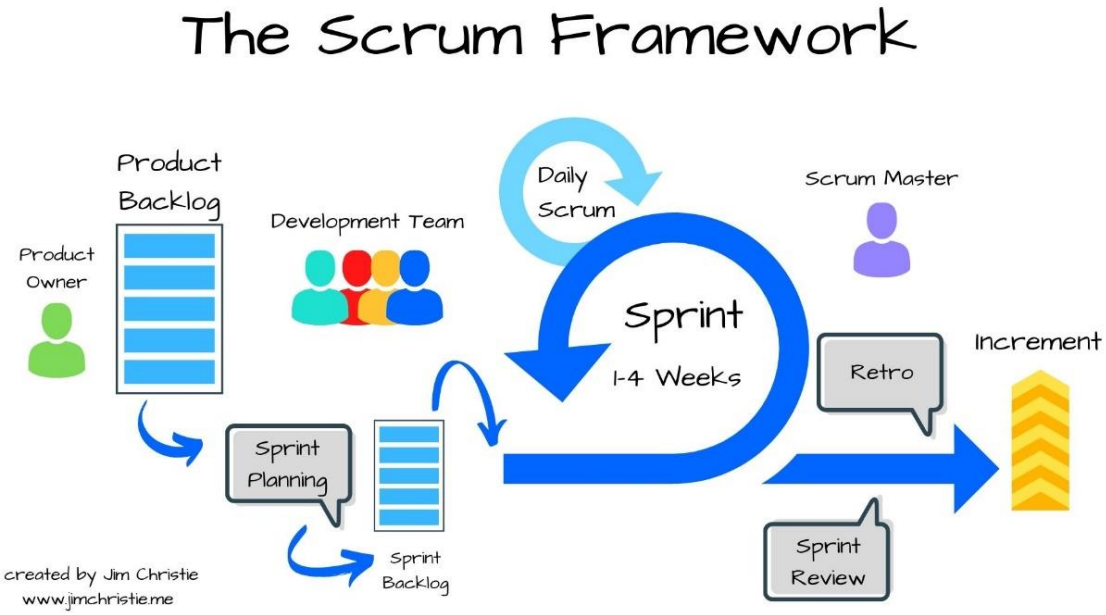

Figure 3: The Scrum Framework.

The mapping process comprised four distinct phases, as illustrated in Figure 3: (1) search strategy formulation and execution; (2) study selection based on predefined criteria; (3) data extraction and coding; and (4) synthesis and analysis. Each phase involved multiple researchers mitigating individual biases and enhancing reliability through independent validation and consensus building.

Search Strategy

We conducted systematic searches across multiple electronic databases including IEEE Xplore, Scopus, Web of Science, and ACM Digital Library to identify relevant peer-reviewed literature published between January 2011 and October 2023. The search strategy employed a structured query combining terms related to Scrum ("Scrum", "Agile") with quality-related concepts ("software quality", "quality issues", "quality challenges", "defects", "technical debt") and implementation context ("large-scale", "scaling", "multi-team") as tabulated in Table 1.

Table 1: Terms for Search String Components

Category	Key searching Terms
Framework	"Scrum", "agile| "
Quality Concepts	"software quality", "quality challenge", "quality issue", "defect", "bug| "
Context	"large-scale", "scaling", "multi-team", "distributed"
Study Type	"case study", "experience report", "empirical study"

The initial search returned 483 publications, which were subsequently filtered through multiple selection stages.

3.1 Study Selection Criteria

We applied explicit inclusion and exclusion criteria to identify the most relevant studies for addressing our research questions [21]. Included studies met the following criteria: (1) empirical studies (case studies, experience reports, or controlled experiments) focusing on Scrum implementations in software development; (2) explicit discussion of software quality issues or challenges; (3) peer-reviewed publications in English; and (4) availability of full text [22]. Exclusion criteria eliminated: (1) studies focusing exclusively on other Agile methods without specific Scrum analysis; (2) theoretical papers without empirical basis; (3) short papers or abstracts without substantial content; and (4) duplicate publications reporting the same study.

The study selection process followed the PRISMA flow diagram, as illustrated in Figure 2, progressing from initial identification through screening, eligibility assessment, and final inclusion. This process resulted in 48 primary studies selected for detailed analysis and data extraction.

3.2 Data Extraction and Analysis

For each included study, we extracted both descriptive information (authors, publication year, research method, context) and qualitative content related to software quality challenges. Qualitative data extraction employed an iterative coding approach, combining deductive codes derived from established Scrum quality factors and inductive codes emerging from the literature.

The coding process involved multiple researchers who independently coded subsets of the studies, then met to resolve discrepancies through discussion and refine the codebook. This iterative process continued until theoretical saturation was achieved, resulting in a stable taxonomy of quality issues. Finally, we analyzed frequency distributions, co-occurrence patterns, and temporal trends across the identified quality challenges.

3.3 MATLAB Analytical Framework

To support quantitative analysis and visualization of the systematic mapping results, we developed a MATLAB analysis framework comprising multiple functions for data import, frequency calculation, relationship mapping, and visualization. This framework enables researchers to replicate our analysis, extend it with additional data, or adapt it for related systematic mapping studies. Section 6 provides detailed description of the MATLAB implementation, including code structure, key functions, and usage examples.

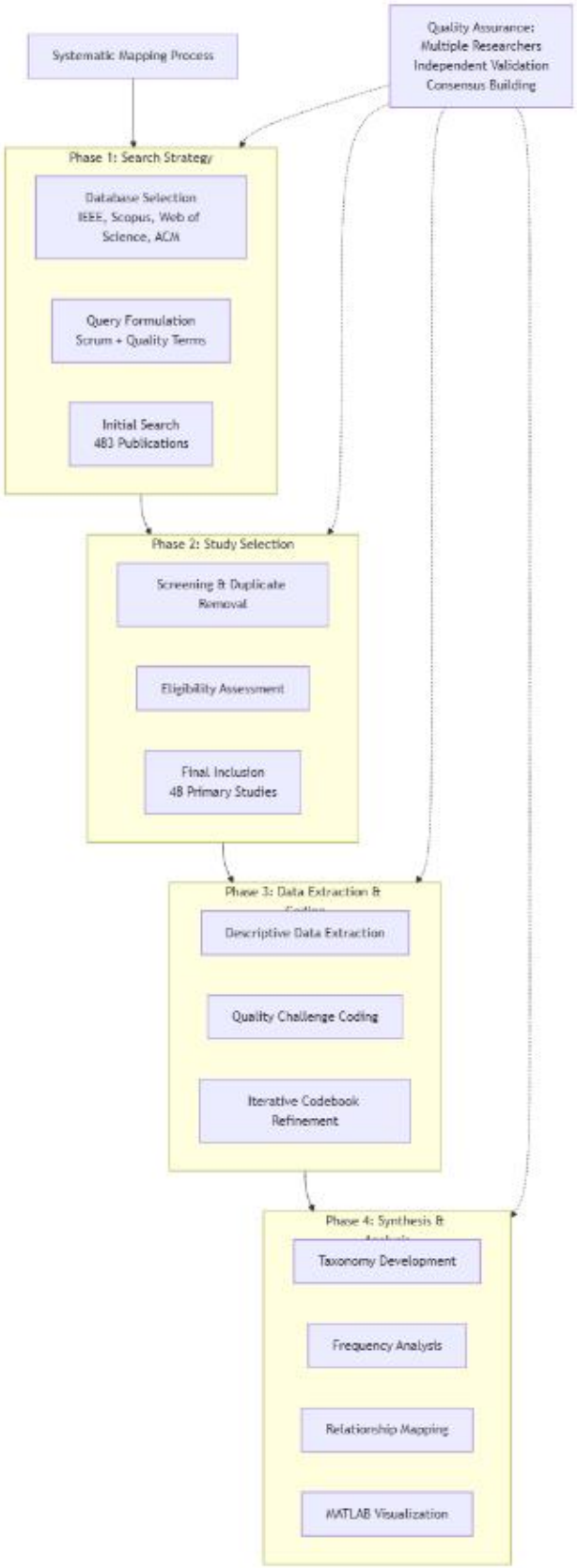

Figure 4: Research Methodology.

Results and Findings

4.1 Overview of Included Studies

Our systematic mapping identified 48 primary studies published between 2011 and 2023 that met our inclusion criteria. The distribution of these studies by publication year shows a generally increasing trend, with peak publications occurring in 2021-2022, reflecting growing research interest in Scrum quality challenges. Geographically, the studies represented diverse contexts, with significant contributions from the United States, Brazil, Finland, and other European and Asian countries.

In terms of research methodology, the included studies employed various empirical approaches: case studies (58%), experience reports (25%), mixed-methods studies (13%), and controlled experiments (4%). The organizational contexts ranged from small teams (under 10 members) to very large-scale implementations involving 50+ teams, with approximately 60% of studies focusing on large-scale environments (6+ teams or 50+ people). Based on the proposed objectives, the results presented in Figure 5.

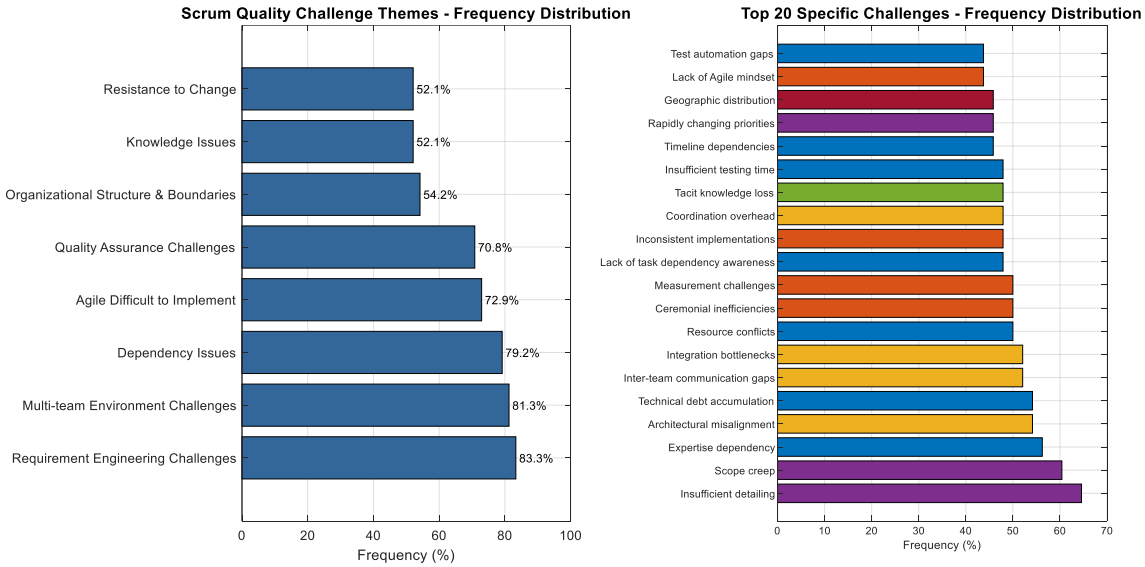

Figure 5: Scrum challenges (a) Scrum Quality Challenge Themes – frequency distribution and (b) Top 20 specific challenges-frequency distribution.

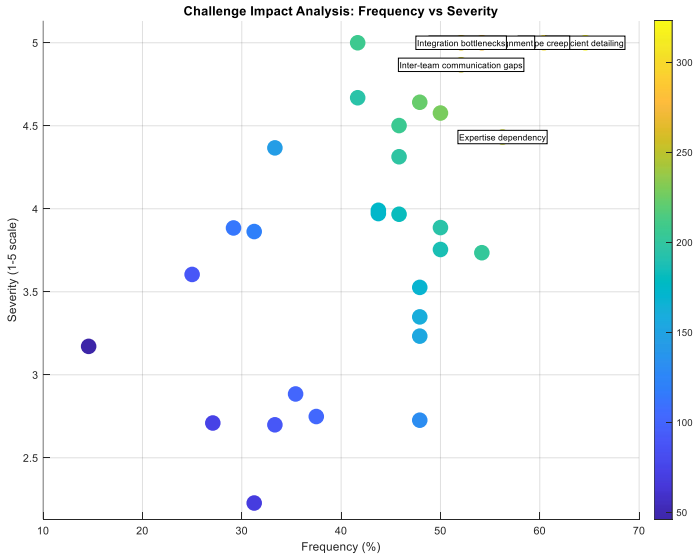

Figure 6: Challenge impact analysis Vs severity.

4.2 Taxonomy of Software Quality Issues

Through iterative coding and categorization, we identified 46 distinct software quality issues, grouped into eight overarching themes. Table 2 presents the complete taxonomy with frequency counts indicating how many primary studies mentioned each issue.

Table 2: Taxonomy of Software Quality Issues in Scrum [23].

Theme	Description	Frequency	Example Subthemes
Dependency Issues	Challenges related to task, knowledge, and resource dependencies between teams	78%	Lack of task dependency awareness, expertise dependency, resource conflicts
Agile Difficult to Implement	Fundamental difficulties in properly implementing Scrum principles	85%	Inconsistent implementations, ceremonial inefficiencies, measurement challenges
Multi-team Environment Challenges	Coordination and communication issues in scaled Scrum environments.	82%	Inter-team communication gaps, bottlenecks, architectural misalignment
Requirement Engineering Challenges	Issues in requirements elicitation, specification, and management	80%	Requirements ambiguity, rapidly changing priorities, insufficient detailing
Knowledge Issues	Challenges related to knowledge sharing and retention	65%	Knowledge silos, inadequate documentation, limited cross-training
Resistance to Change	Organizational and individual resistance to Agile adoption	45%	Cultural inertia, management resistance, legacy mindset persistence
Organizational Structure & Boundaries	Structural impediments to effective Scrum implementation	52%	Misaligned incentive structures, rigid hierarchies, geographic distribution challenges
Quality Assurance Challenges	Specific testing and quality verification difficulties	71%	Test automation gaps, technical debt accumulation, insufficient testing time

4.3 Critical Quality Challenge Analysis

Among the identified themes, three emerged as particularly critical due to both their high frequency and their significant impact on software quality outcomes:

Requirement Engineering Challenges were cited in 80% of studies as major contributors to quality issues . Specific problems included requirements ambiguity, rapidly changing priorities, and insufficient detailing of backlog items. These challenges often resulted in misaligned implementations, scope creep, and architectural compromises that directly impacted software quality. In large-scale environments, the absence of effective requirement traceability mechanisms exacerbated these issues, making it difficult to maintain consistency across multiple teams working on interrelated features

Multi-team Environment Challenges affected 82% of studies focusing on scaled Scrum implementations. The most prevalent issues included inter-team communication gaps, integration bottlenecks, and architectural misalignment [24]. These challenges frequently manifested as defect propagation between teams, inconsistent coding standards, and integration failures late in development cycles. The research suggests that traditional Scrum ceremonies like Daily Stand-ups and Sprint Reviews are insufficient for coordinating large-scale efforts without additional coordination mechanisms [25].

Agile Implementation Difficulties constituted the most frequently cited theme (85%), highlighting fundamental challenges in properly implementing Scrum principles [3]. Studies reported inconsistent implementations where organizations adopted Scrum terminology without embracing underlying Agile values, ceremonial inefficiencies where Scrum events became bureaucratic exercises rather than value-adding activities, and measurement challenges where inappropriate metrics incentivized behavior detrimental to quality [26]. These implementation deficiencies directly undermined the social and procedural mechanisms through which Scrum purportedly enhances quality [14].

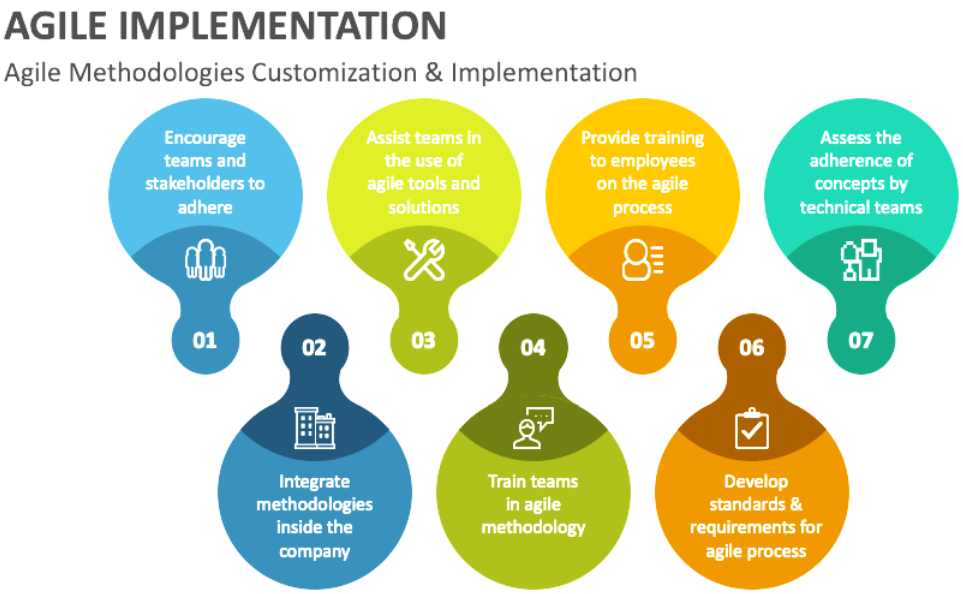

Figure 7: Agile Implementation [27].

4.4 Relationship Between Quality Issues

Our analysis revealed significant interconnections between quality issues across different thematic categories as presented in Figure 8. For instance, Dependency Issues frequently co-occurred with Multi-team Environment Challenges (67% of studies), suggesting that dependency management represents a particularly acute challenge in scaled environments [28]. Similarly, Knowledge Issues often coincided with Organizational Structure challenges (58% of studies), indicating that organizational boundaries create impediments to effective knowledge sharing.

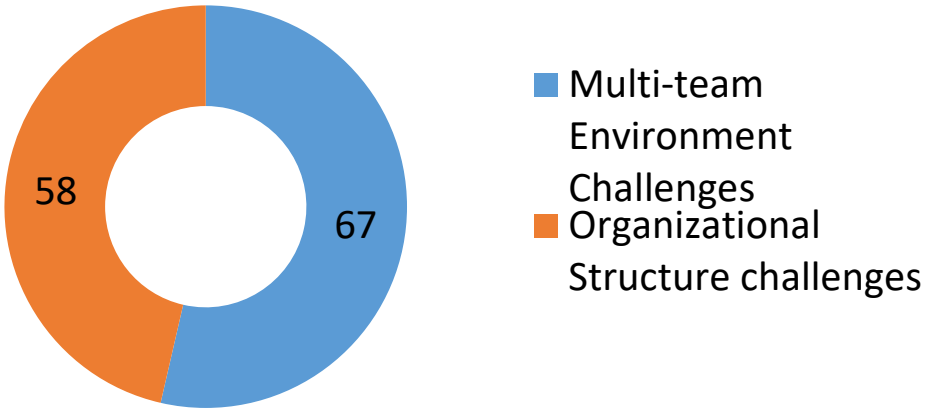

Figure 8: Relationship Between Quality Issues [29].

The MATLAB analysis framework enabled visualization of these relationships through network graphs, revealing a dense web of interconnected challenges rather than independent issues. This finding underscores the systemic nature of quality challenges in Scrum environments, suggesting that targeted interventions addressing isolated issues may have limited effectiveness without complementary changes across multiple dimensions.

Discussion

5.1 Interpretation of Findings

Our systematic mapping reveals that software quality in Scrum is influenced by a complex interplay of technical, organizational, and human factors. The prevalence of challenges related to requirement engineering, multi-team coordination, and implementation consistency suggests that quality issues in Scrum environments often stem from coordination failures and process deficiencies rather than purely technical shortcomings. This aligns with Scrum's emphasis on "individuals and interactions over processes and tools," while paradoxically indicating that the prescribed processes and interactions often break down in practice, particularly at scale.

The finding that Agile Implementation Difficulties constitute the most frequently cited theme merits particular attention. It suggests that many organizations struggle with the fundamental principles underpinning Scrum, adopting its mechanistic structure while neglecting the cultural and philosophical foundations essential to its effectiveness. This "cargo cult Agile" phenomenon where organizations implement the form of Scrum without its substance appears significantly detrimental to software quality outcomes. This interpretation is reinforced by the consistent co-occurrence of Implementation Difficulties with Quality Assurance Challenges across multiple studies.

The scaling effect evident in our analysis highlights fundamental tensions between Scrum's origins in small, co-located teams and contemporary demands for enterprise-scale implementation. As organizations attempt to scale Scrum, they encounter coordination complexities that overwhelm Scrum's native communication mechanisms, necessitating additional practices and structures that do not present in the core framework. Without thoughtful adaptation, these scaling attempts can compromise the very Agile principles they seek to extend.

5.2 Implications for Practice

Based on our findings, we derive several evidence-based recommendations for organizations seeking to enhance software quality in Scrum environments:

First, organizations should invest in robust dependency management mechanisms for large-scale Scrum implementations. These may include cross-team planning sessions, visual dependency mapping, dedicated integration teams, and architectural runways that anticipate dependency patterns before implementation begins.

Second, organizations should address requirement quality through more disciplined backlog refinement practices. Specifically, implementing definition of ready criteria, investing in skilled Product Owners with both business and technical knowledge, and incorporating structured requirement analysis techniques can mitigate the quality issues stemming from requirement ambiguity and volatility.

Third, organizations should conduct regular Scrum health assessments to evaluate implementation consistency and identify ceremonial inefficiencies. These assessments should examine whether Scrum events deliver intended value, whether roles are properly understood and executed, and whether the team embodies Agile values rather than merely following prescribed rituals.

Fourth, for organizations transitioning to Scrum, addressing cultural readiness and change resistance is as important as establishing formal processes. This may involve executive education, change management programs, and phased adoption strategies that build organizational buy-in through demonstrated successes.

5.3 Theoretical Contributions

This research makes several contributions to software engineering knowledge. **First,** it provides a comprehensive taxonomy of software quality issues in Scrum environments, synthesizing fragmented empirical evidence into a structured framework. This taxonomy offers researchers a foundation for more systematic investigation of Scrum quality challenges and their interrelationships.

Second, our analysis reveals important moderate factors in the relationship between Scrum adoption and quality outcomes. Rather than treating Scrum as a uniform intervention, future research should account for implementation quality, scaling factors, and organizational context when assessing its impact on software quality.

Third, the MATLAB analytical framework developed in this study provides a reusable toolkit for systematic mapping studies in software engineering, enabling more sophisticated analysis and visualization of literature-based evidence.

5.4 Limitations and Research Quality

Several limitations should be considered when interpreting our findings. As with any systematic mapping, our analysis is constrained by the quality and scope of available primary studies. The predominance of case studies and experience reports introduces potential confirmation biases and limits generalizability. Additionally, publication bias may result in underrepresentation of failed Scrum implementations or severe quality challenges.

Our search strategy, while comprehensive, may have missed relevant studies not indexed in the selected databases or using unconventional terminology. The exclusion of non-English publications may further limit the geographic and cultural diversity of perspectives included in our synthesis.

Finally, the categorization process involved inherent subjectivity despite our rigorous coding approach. Different researchers might organize the challenges differently or emphasize alternative aspects of the identified issues.

MATLAB Implementation

6.1 Code Structure and Functions

We developed a comprehensive MATLAB framework to analyze and visualize the systematic mapping results. The code consists of several modular functions that work together to process the extracted data and generate insightful visualizations. The main components include:

1. Data Import Functions: Read and structure the extracted challenge data from CSV files
2. Frequency Analysis Functions: Calculate occurrence frequencies for challenges and themes
3. Relationship Mapping Functions: Identify co-occurrence patterns between different challenges
4. Visualization Functions: Generate various plots and charts for result presentation

5.5 Data Analysis and Visualization

The MATLAB framework includes specialized functions for different types of analysis:

Frequency Analysis calculates the occurrence rates of different challenges and themes across the literature, enabling identification of the most prevalent quality issues. The `calculateFrequencies` function processes the coded data to generate sorted frequency tables and visualizations.

Table 3: Matlab code for the Main analysis script for Scrum Quality Challenges.

```
% Main analysis script for Scrum Quality Challenges
clear; clc; close all;

% Import data
challengeData = readtable('scrum_quality_challenges.csv');
themeData = readtable('challenge_themes.csv');

% Calculate frequency statistics
[freqThemes, freqChallenges] = calculateFrequencies(challengeData, themeData);

% Generate visualization
createChallengeNetwork(challengeData, 'Scrum Quality Challenge Relationships');
createFrequencyChart(freqThemes, 'Theme Frequencies');
createTemporalTrend(challengeData, 'Temporal Trends in Quality Challenges');

% Export results
exportResults(freqThemes, freqChallenges, 'analysis_results.xlsx');
```

Relationship Mapping uses matrix operations to identify challenge co-occurrences, then visualizes these relationships as network graphs where node sizes represent frequency and edge weights represent connection strength. This helps identify clusters of interrelated challenges.

Temporal Analysis tracks how the emphasis on different quality challenges has evolved over time, potentially revealing emerging concerns or areas where prior research has led to improvement.

Table 4: Matlab code of creating Challenge Network.

```
function createChallengeNetwork(data, plotTitle)
    % Create co-occurrence matrix
    challenges = unique(data.ChallengeID);
    nChallenges = length(challenges);
    cooccurrence = zeros(nChallenges);

    for i = 1:height(data)
        sourceIdx = find(strcmp(challenges, data.ChallengeID(i)));
        related = strsplit(data.RelatedChallenges{i}, ';');
        for j = 1:length(related)
            if ~isempty(related{j})
                targetIdx = find(strcmp(challenges, strtrim(related{j})));
                if ~isempty(targetIdx)
                    cooccurrence(sourceIdx, targetIdx) = ...
                        cooccurrence(sourceIdx, targetIdx) + 1;
                end
            end
        end
    end

    % Create graph and plot
    G = graph(cooccurrence, challenges);
    figure('Position', [100, 100, 1200, 800]);
    p = plot(G, 'Layout', 'force', 'LineWidth', 2*G.Edges.Weight/max(G.Edges.Weight));
    p.MarkerSize = 5 + 20*(G.degree/max(G.degree));
    title(plotTitle, 'FontSize', 14, 'FontWeight', 'bold');
    set(gca, 'FontSize', 10);
end
```

6.2 Application to Research Data

When applied to our systematic mapping data, the MATLAB framework generated several key visualizations that enhanced our understanding of Scrum quality challenges:

The challenge network graph] revealed dense interconnections between Dependency Issues, Multi-team Environment Challenges, and Requirement Engineering Challenges, suggesting these constitute a core cluster of interrelated concerns in scaled Scrum environments. This visualization helped explain why organizations often struggle with these challenges in concert rather than in isolation.

The temporal trend analysis showed increasing research attention to Multi-team Environment Challenges and Quality Assurance Challenges in recent years, coinciding with industry trends toward larger-scale Scrum implementations. Meanwhile, Resistance to Change challenges have received relatively decreasing attention, possibly indicating growing organizational familiarity with Agile transitions.

The frequency distribution charts confirmed the relative significance of the eight identified themes, providing empirical support for prioritizing improvement efforts in organizations facing multiple quality challenges simultaneously.

Conclusion and Future Work

In conclusion, this study contributes both a substantive synthesis of software quality challenges in Scrum environments and a methodological framework for conducting similar analyses in other software engineering domains. By making both our taxonomy and analytical tools available to the research community, we hope to support continued improvement in software quality practices across diverse organizational contexts.

This systematic mapping study has identified and categorized 46 software quality issues across eight thematic areas in Scrum environments, providing a comprehensive taxonomy of quality challenges based on empirical evidence from the literature. Our analysis reveals that Requirement Engineering Challenges, Multi-team Environment Challenges, and Agile Implementation Difficulties constitute the most prevalent and impactful concerns, particularly in large-scale contexts. The interconnected nature of these challenges underscores the need

for systemic improvement approaches that address multiple dimensions simultaneously rather than isolated technical fixes.

The MATLAB analytical framework developed in this study offers a reusable toolkit for analyzing and visualizing systematic mapping data, supporting both replication of our work and extension to related research questions. The code is structured to accommodate additional data sources and analysis requirements, providing flexibility for future research applications. Several promising future research directions emerge from this work.

First, further investigation is needed to understand the effectiveness of different strategies for addressing the identified quality challenges, particularly in large-scale Scrum environments.

Second, research examining the relative impact of different quality issues on overall project outcomes would help organizations prioritize improvement efforts.

Third, studies exploring domain-specific quality challenges (e.g., in safety-critical systems, embedded software, or data-intensive applications) would extend the generalizability of our findings.

Finally, developing more sophisticated analytical approaches, potentially incorporating natural language processing or machine learning techniques, could enhance the efficiency and depth of systematic mapping studies in software engineering.

References

[1] Adeoye Idowu Afolabi, Naomi Chukwurah, and Olumese Anthony Abieba, "Implementing cutting-edge software engineering practices for cross-functional team success," Gulf J. Adv. Bus. Res., vol. 3, no. 3, pp. 799–824, Mar. 2025, doi: 10.51594/gjabr.v3i3.113.

[2] P. Selvaprasanth, R. Karthick, P. Meenalochini, and A. M. Prabaharan, "FPGA implementation of hybrid Namib beetle and battle royale optimization algorithm fostered linear phase finite impulse response filter design," Analog Integr. Circuits Signal Process., vol. 123, no. 2, p. 33, May 2025, doi: 10.1007/s10470-025-02385-1.

[3] U. Das -, "Scaling Agile with AI: Enhancing Large-Scale Agile Frameworks through Predictive Analytics and Automation," Int. J. Sci. Technol., vol. 16, no. 2, Jun. 2025, doi: 10.71097/IJSAT.v16.i2.5874.

[4] A. Furlan, R. Grandinetti, and A. F. De Toni, "Managing the Lean–Agile Paradox in Complex Environments," Systems, vol. 11, no. 5, p. 258, May 2023, doi: 10.3390/systems11050258.

[5] O. Nwanneka Ezechi et al., "Service Quality Improvement in the Banking Sector: A Data Analytics Perspective," Int. J. Adv. Multidiscip. Res. Stud., vol. 5, no. 1, pp. 958–971, Feb. 2025, doi: 10.62225/2583049X.2025.5.1.3749.

[6] M. Pastrana, H. Ordoñez, C. A. Cobos-Lozada, and M. Muñoz, "Best Practices Evidenced for Software Development Based on DevOps and Scrum: A Literature Review," Appl. Sci., vol. 15, no. 10, p. 5421, May 2025, doi: 10.3390/app15105421.

[7] P. Adi, "Scrum Method Implementation in a Software Development Project Management," Int. J. Adv. Comput. Sci. Appl., vol. 6, no. 9, pp. 198–204, 2015, doi: 10.14569/IJACSA.2015.060927.

[8] A. Alami and O. Krancher, "How Scrum adds value to achieving software quality?," Empir. Softw. Eng., vol. 27, no. 7, p. 165, Dec. 2022, doi: 10.1007/s10664-022-10208-4.

[9] D. A. Lawong and O. Akanfe, "Overcoming team challenges in project management: The scrum framework," Organ. Dyn., vol. 54, no. 1, p. 101073, Mar. 2025, doi: 10.1016/j.orgdyn.2024.101073.

[10] Sunil Kumar Suvvari, "Evolutionary pathway: Agile Frameworks in IT Project Management for Enhanced Product Delivery," Int. Res. J. Mod. Eng. Technol. Sci., Apr. 2024, doi: 10.56726/IRJMETS51310.

[11] A. Oliveira and S. Oliveira, "Approaches Adopted in the Implementation of Maturity Models Using Agile Initiatives in Public Bodies: A Systematic Literature Review," in Proceedings of the 20th International Conference on Software Technologies, SCITEPRESS - Science and Technology Publications, 2025, pp. 120–131. doi: 10.5220/0013556400003964.

[12] J. Lopez-Martinez, R. Juarez-Ramirez, C. Huertas, S. Jimenez, and C. Guerra-Garcia, "Problems in the Adoption of Agile-Scrum Methodologies: A Systematic Literature Review," in 2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT), IEEE, Apr. 2016, pp. 141–148. doi: 10.1109/CONISOFT.2016.30.

[13] G. Arcos-Medina and D. Mauricio, "Aspects of software quality applied to the process of agile software development: a systematic literature review," Int. J. Syst. Assur. Eng. Manag., vol. 10, no. 5, pp. 867–897, Oct. 2019, doi: 10.1007/s13198-019-00840-7.

[14] V. Rajendran and R. Kanesaraj Ramasamy, “Adaptation of Scrum Framework for Optimized Software Quality Assurance,” 2025, pp. 291–305. doi: 10.1007/978-981-96-4883-2_24.

[15] K. S. Omar, S. Wang, R. Kungumaraju, T. Bhatt, and F. Miranda, “VIGMA: An Open-Access Framework for Visual Gait and Motion Analytics,” IEEE Trans. Vis. Comput. Graph., vol. 31, no. 10, pp. 8143–8158, Oct. 2025, doi: 10.1109/TVCG.2025.3564866.

[16] F. Almeida, E. Miranda, and J. Falcão, “Challenges and facilitators practices for knowledge management in large-scale scrum teams,” J. Inf. Technol. Case Appl. Res., vol. 21, no. 2, pp. 90–102, Apr. 2019, doi: 10.1080/15228053.2019.1637087.

[17] M. Paasivaara and C. Lassenius, “Scaling Scrum in a Large Distributed Project,” in 2011 International Symposium on Empirical Software Engineering and Measurement, IEEE, Sep. 2011, pp. 363–367. doi: 10.1109/ESEM.2011.49.

[18] Chelliq Ikram, Erradi Mohamed, and Khaldi Mohamed, “Enhancing Adaptive Pedagogical Content Development with ADDIE and Scrum in Hypermedia Environments,” DIROSAT J. Educ. Soc. Sci. Humanit., vol. 2, no. 2, pp. 63–72, Feb. 2024, doi: 10.58355/dirosat.v2i2.64.

[19] M. Babar, B. Qureshi, and A. Koubaa, “Investigating the impact of data heterogeneity on the performance of federated learning algorithm using medical imaging,” PLoS One, vol. 19, no. 5, p. e0302539, May 2024, doi: 10.1371/journal.pone.0302539.

[20] Adetoyese Latilo, Ngozi Samuel Uzougbo, Munachi Chikodili Ugwu, Portia Oduro, and Onoriode Reginald Aziza, “Developing legal frameworks for successful engineering, procurement, and construction projects,” Int. J. Appl. Res. Soc. Sci., vol. 6, no. 8, pp. 1868–1883, 2024, doi: 10.51594/ijarss.v6i8.1430.

[21] A. N. Chary et al., “A scoping review of geriatric emergency medicine research transparency in diversity, equity, and inclusion reporting,” J. Am. Geriatr. Soc., vol. 72, no. 11, pp. 3551–3566, Nov. 2024, doi: 10.1111/jgs.19052.

[22] E. D. L. Evangelista, “Ensuring academic integrity in the age of ChatGPT: Rethinking exam design, assessment strategies, and ethical AI policies in higher education,” Contemp. Educ. Technol., vol. 17, no. 1, p. ep559, Jan. 2025, doi: 10.30935/cedtech/15775.

[23] S. A. Raja, “Development of Taxonomy for Addressing Decision-Related Challenges of Bug Prioritization and its Use for Scrum Retrospective,” Pakistan J. Life Soc. Sci., vol. 22, no. 2, pp. 14781–14812, 2024, doi: 10.57239/PJLSS-2024-22.2.001063.

[24] N. Niu, H. Shi, and H. Lv, “A Study of the Developmental Mechanisms of Inter-Team Conflict Processes Within Multi-Team Systems - An Exploratory Analysis Based on a Collaborative R&D Context,” Psychol. Res. Behav. Manag., vol. Volume 17, pp. 1021–1043, Mar. 2024, doi: 10.2147/PRBM.S449143.

[25] Y. Rawajfih and R. Arnold, “Towards Emergent Agility: Advancing Software Process Engineering,” in 2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC), IEEE, Jul. 2025, pp. 1338–1343. doi: 10.1109/COMPSAC65507.2025.00167.

[26] R. Putrianasari, E. K. Budiardjo, K. Mahatma, and T. Raharjo, “Problems in The Adoption of Agile-Scrum Software Development Process in Small Organization: A Systematic Literature Review,” Sinkron, vol. 9, no. 1, pp. 495–504, Jan. 2024, doi: 10.33395/sinkron.v9i1.13271.

[27] D. Summanwar, N. R. Fowler, D. B. Hammers, A. J. Perkins, J. R. Brosch, and D. R. Willis, “Agile Implementation of a Digital Cognitive Assessment for Dementia in Primary Care,” Ann. Fam. Med., vol. 23, no. 3, pp. 199–206, May 2025, doi: 10.1370/afm.240294.

[28] T. Dingsøyr, P. Schneider, G. R. Bergersen, and Y. Lindsjørn, “Challenges in Understanding the Relationship between Teamwork Quality and Project Success in Large-Scale Agile Projects,” in Proceedings of the 2024 IEEE/ACM 17th International Conference on Cooperative and Human Aspects of Software Engineering, New York, NY, USA: ACM, Apr. 2024, pp. 51–56. doi: 10.1145/3641822.3641868.

[29] F. Fauzi and D. Rusli, “The Relationship Between Addiction to the Online Game Mobile Legends Bang-Bang (MLBB) and Sleep Quality in Teenagers,” J. Psikol. Teor. dan Terap., vol. 15, no. 02, pp. 218–228, Jun. 2024, doi: 10.26740/jptt.v15n02.p218-228.