*I.J. Information Engineering and Electronic Business*, **2025, 3, 84-98**
Published Online on June 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijieeb.2025.03.06

# Agile Method: Challenges and Adaptations for Complex Project Environments

**Abdulmajeed Aljehani**
Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia
E-mail : aaljehani0250@stu.kau.edu.sa
ORCID ID: https://orcid.org/ 0009-0007-6900-4607

**M. Rizwan Qureshi***
Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia
E-mail : rmuhammd@kau.edu.sa
ORCID ID: https://orcid.org/0000-0002-1664-0737
*Corresponding Author

Received: 01 January, 2025; Revised: 28 January, 2025; Accepted: 26 February, 2025; Published: 08 June, 2025

**Abstract:** This paper conducts a comparative analysis of three widely adopted Agile methodologies: Scrum, Kanban, and Extreme Programming (XP). By examining their application across diverse software development environments, the study highlights each methodology's inherent strengths and explores their practical implications for managing complex, large-scale projects. Central to this investigation are the scalability challenges that become particularly pronounced in settings with extensive stakeholder groups and complex coordination needs. The research draws upon a robust literature review and case studies to identify these challenges, setting the stage for a discussion of innovative solutions aimed at refining Agile practices. While specific solutions are reserved for detailed treatment in the proposed solutions section, the abstract is written to underscore the critical need for scalable strategies that can adapt to the dynamic landscapes of modern project management. This comparative inquiry not only enriches the academic discourse on Agile methodologies but also serves as a vital resource for practitioners seeking to optimize their project management strategies in complex scenarios.

**Index Terms:** Agile Methodologies, Scrum, Kanban, Extreme Programming (XP), Software Development

## 1. Introduction

In the evolving landscape of software development, Agile methodologies such as Scrum, Kanban, and Extreme Programming (XP) have emerged as transformative forces, reshaping how projects are managed and executed across industries. Characterized by their flexibility and responsiveness, these methodologies are tailored to meet the dynamic demands of small to medium-sized project teams. However, as organizations grow and projects expand in scope and complexity, the application of Agile principles faces new challenges. The scalability of Agile methodologies in large-scale environments introduces significant hurdles. These challenges stem from the need to maintain the core Agile principles—such as rapid iteration and close collaboration—while managing multiple, often geographically dispersed teams with varying objectives [1]. Additionally, larger projects frequently involve more complex stakeholder interactions and higher regulatory requirements, which can conflict with Agile's intrinsic emphasis on flexibility and minimal bureaucratic overhead.

This paper delves into these scalability issues, exploring the inherent tensions between Agile methodologies and the demands of large-scale project management. We will explore how the principles that facilitate agility in small teams can be abstracted to fit large, multifaceted projects without losing the essence of Agile practices. The focus will be on identifying key obstacles and proposing theoretical frameworks that can potentially bridge the gap between Agile methodologies and large-scale implementation needs. Through a comprehensive review of literature and empirical case studies, this introduction sets the foundation for a detailed discussion on adapting Agile to meet the challenges of complex project environments. By examining these critical aspects, the subsequent sections aim to provide actionable insights and innovative solutions to enhance the scalability of Agile methodologies, ensuring they remain effective tools for modern software development across all scales of operation.

 

Further paper is arranged as: Section 2 focuses on the related work. Section 3 defines the problem in hand. Section 4 describes the motivation of the proposed solution. Section 5 illustrates the proposed solution. Section 6 outlines the three goals narrated from the proposed solution. Section 7 discusses the statistical analysis of the three goals to validate the proposed solution.

## 2. Related Work

Shrivastava et al. highlights the evolution and relevance of Extreme Programming (XP) as a leading agile methodology. XP emphasizes iterative development, customer satisfaction, and dynamic integration of changing requirements. The study contrasts XP's flexibility with traditional waterfall approaches, showcasing its ability to deliver high-quality software efficiently within modern web-based project environments [1]. Verwijs and Russo propose a theoretical model for Scrum team effectiveness, derived from extensive mixed-methods research. The model identifies responsiveness, stakeholder focus, continuous improvement, team autonomy, and management support as critical factors. Validated through structural equation modeling, this framework provides actionable insights to enhance Scrum practices and optimize team performance [2].

Waja et al. discuss Agile's iterative and incremental nature, contrasting it with traditional methodologies. Agile promotes adaptability, cost reduction, and rapid delivery while prioritizing customer collaboration. The study emphasizes Agile's impact on enhancing software quality and its capability to meet dynamic industry demands [3]. Orlov et al. conduct a comparative analysis of Kanban and Scrum, two widely used agile methodologies. While Kanban excels in visualizing workflows and managing continuous tasks, Scrum is better suited for projects requiring structured iteration cycles. This study offers a nuanced understanding of methodology selection based on project characteristics [4].

Saltz and Heckman investigate the integration of Kanban in educational settings, focusing on its ability to instill Agile principles among students. The findings show that self-organization and regular reflection are the most internalized principles, making Kanban an effective tool for teaching project management in diverse learning environments [5]. Akhtar et al. compare Extreme Programming (XP) and Scrum, two agile frameworks tailored for iterative development. XP focuses on engineering practices, while Scrum emphasizes process transparency and team collaboration. The study explores the complementary attributes of these frameworks in managing dynamic project requirements [6]. Santana and Romero examine XP's core principles, emphasizing its short development cycles and customer-centric approach. The study illustrates XP's adaptability and its ability to deliver high-quality software under rapidly changing requirements. This position is XP as a preferred methodology for agile development [7].

Hamdulay evaluates the similarities and differences among Agile, Scrum, and Kanban. While Agile provides a broad philosophy for iterative progress, Scrum offers structured frameworks, and Kanban facilitates workflow visualization. The comparative analysis underscores the methodologies' suitability for varying project demands [8]. Hema et al. present Scrum as a dynamic Agile methodology designed for projects requiring iterative delivery and adaptability. The study highlights Scrum's emphasis on regular customer feedback, team collaboration, and rapid development, establishing its effectiveness in managing evolving project requirements [9]. Bhavsar et al. analyze Scrum's transformative role in software engineering, positioning it as a business process reengineering tool. The study emphasizes Scrum's empirical approach, focusing on transparency, adaptability, and continuous improvement, making it an integral framework for modern software development [10]. Table 1 is used to display limitations of the related work.

Smith et al. [11] explores how traditional project management techniques can be effectively combined with Agile methodologies to address common challenges in large-scale projects. The authors analyze case studies where hybrid models have been implemented, highlighting the advantages in terms of improved risk management, better project scope definition, and enhanced compliance with industry standards. It provides a theoretical basis and practical insights for using hybrid Agile approaches in sectors where Agile alone might not suffice due to complex regulatory or scale demands. Ladas [12] introduces Scrumban, a hybrid Agile management methodology that merges the iterative development features of Scrum with the continuous flow principles of Kanban. The book provides a detailed framework for implementing Scrumban in software development environments, emphasizing its suitability for projects that require a flexible yet disciplined approach. The methodology is particularly effective in environments where work priorities shift frequently but require maintaining a steady workflow.

Larman and Vodde [13] offer a comprehensive guide to scaling Agile and Lean development practices using Large-Scale Scrum (LeSS). The book discusses organizational redesign, the adoption of multi-team coordination, and the integration of lean thinking into Agile practices. It is particularly useful for organizations looking to scale Agile without losing its core values of flexibility and rapid feedback, providing practical tools and case studies on effective large-scale implementations. Leffingwell's guide is a cornerstone resource for understanding the Scaled Agile Framework (SAFe), one of the most popular frameworks for enterprise-scale Agile adoption [14]. It details how SAFe combines Agile and Lean principles into a cohesive framework that supports both small-team flexibility and large-scale coherence. The guide covers various levels of SAFe implementation, roles, responsibilities, and the specific processes for ensuring Agile scaling success across complex organizational structures. The paper in [15] addresses the integration challenges between traditional project management and Agile practices, examining how blending these can resolve common project execution issues. The study is based on empirical data and focuses on bridging the methodological

gaps that often exist in large projects, providing a balanced view of how structured project management can complement the adaptive nature of Agile to enhance project delivery and stakeholder satisfaction.

*2.1 Detailed Comparison of Scrum, Kanban and XP*

Agile methodologies such as Scrum, Kanban, and XP have transformed software development by fostering flexibility, collaboration, and efficiency. However, scaling these methodologies in large organizations introduces challenges related to cross-team coordination, dependency management, backlog prioritization, and workflow synchronization. This section provides a detailed comparison of how each methodology performs at scale, including its advantages, limitations, and real-world applications as shown in Table 2.

Table 1. Limitations and Restrictions of the Related Work

| **Title Of the Paper** | **Limitations** |
|---|---|
| "Agile Software Development" [1] | Despite Agile's effectiveness for small and medium-sized projects, scaling it for large or complex projects can be challenging. |
| "Scrum: An Effective Software Development Agile Tool" [2] | While frequent customer feedback is a strength of SCRUM, it may also lead to issues if customers are not consistently available or engaged. |
| "Scrum: An Agile Process Reengineering in Software Engineering" [3] | Is Scrum ready for integration with advanced technologies and their benefits? |
| "A Theory of Scrum Team Effectiveness" [4] | That might imply that smaller or specific teams, due to not fitting the standardized best practices or dynamics observed in the broader samples, would not benefit from this research since it focuses on large-scale data gathering |
| "Comparative Analysis of the Use of Kanban and Scrum Methodologies in IT Projects" [5] | The analysis is limited to financial performance indicators, potentially overlooking other important factors such as team dynamics, customer satisfaction, and product quality. |
| "Framework Study for Software Development Via Scrum, Agile And Kanban" [6] | The dependence of the study on survey data, as far as in-depth longitudinal observation is concerned, might be a limiting factor regarding specific insights into how these methodologies are able to perform under sustained and varied project pressures2 |
| "Exploring Which Agile Principles Students Internalize When Using a Kanban Process Methodology" [7] | lack of additional studies to attempt to replicate and expand these findings |
| "A Systematic Review on Extreme Programming" [8] | Rapid iterations can lead to integration challenges if not managed properly. |
| "Extreme Programming vs Scrum: A Comparison of Agile Models" [9] | The comparison in the paper is done more based on theoretical underpinnings rather than empirical evidence from direct case studies, therefore probably affecting practical insights that developers can get from this research |
| "Fast development Extreme Programming (XP)" [10] | Agile methodologies are primarily suited for small to medium-sized teams, which can limit their applicability in larger project contexts. |

- Scrum is a structured Agile framework that organizes work into fixed-length iterations (sprints), enabling teams to deliver incremental improvements. It relies on well-defined roles, including Scrum Master, Product Owner, and Development Team, which facilitate structured planning and execution cycles. While Scrum works effectively for small teams, scaling it to large enterprises requires additional coordination mechanisms. To coordinate multiple Scrum teams, organizations adopt frameworks such as Scrum of Scrums (SoS), Scaled Agile Framework (SAFe), and Large-Scale Scrum (LeSS). The Scrum of Scrums model involves representatives from different teams meeting regularly to discuss progress and dependencies. In SAFe, Scrum teams collaborate within Agile Release Trains (ARTs), which group multiple teams working towards shared business objectives. LeSS, on the other hand, retains Scrum's simplicity by centralizing backlog management across teams while synchronizing sprint reviews and planning. A real-world example of scaled Scrum is Spotify's Squad Model, where Squads operate as independent Scrum teams, grouped into Tribes, which are supported by cross-team structures such as Chapters and Guilds [15]. This model allows for decentralized decision-making while maintaining alignment through shared knowledge networks. Scaling Scrum presents challenges such as misaligned sprints, conflicting backlog priorities, and dependency bottlenecks. When teams work on interconnected features, synchronization issues arise, delaying releases. Additionally, the reliance on frequent backlog refinement can lead to prioritization conflicts if multiple Product Owners struggle to align objectives. Solutions such as Weighted Shortest Job First (WSJF) help optimize backlog prioritization in large enterprises [16].
- Kanban is an Agile methodology that emphasizes workflow visualization, limiting work-in-progress (WIP), and optimizing delivery speed. Unlike Scrum, which enforces time-boxed sprints, Kanban enables continuous flow, making it suitable for projects with high variability and evolving requirements. Large organizations implement Portfolio Kanban, which provides visibility across multiple teams and departments, ensuring enterprise-wide work stream alignment [17]. The Flight Levels approach structures Kanban at three levels: 1) Operational (Team-Level Kanban): Individual teams manage their workflow using Kanban boards; 2) Coordination (Cross-Team Kanban): Teams synchronize dependencies using centralized Kanban tracking; 3) Strategic (Enterprise-Level Kanban): Leadership oversees workstreams to ensure alignment with business priorities. A real-world implementation of large-scale Kanban is seen at Microsoft, where teams use Azure

DevOps Enterprise Kanban to streamline software releases, integrate CI/CD pipelines, and provide real-time progress tracking [18]. While Kanban offers flexibility, it lacks structured roles and ceremonies, making it difficult to maintain alignment across multiple teams. Without dedicated backlog refinement meetings, work prioritization becomes ambiguous. To overcome this, organizations blend Kanban with SAFe or Scrum, forming hybrid models such as Scrumban, which combines Scrum's planning structure with Kanban's workflow optimization [12].

- Extreme Programming (XP) is an Agile methodology that prioritizes engineering excellence, continuous integration, and customer collaboration. XP introduces practices such as Test-Driven Development (TDD), Pair Programming, and Frequent Releases, ensuring high-quality software. Scaling XP requires mechanisms that ensure engineering practices remain consistent across multiple teams. Communities of Practice (CoPs) foster knowledge-sharing among XP practitioners, while DevOps and CI/CD pipelines automate software integration and deployment, ensuring rapid iterations [19]. A prime example is Google's engineering teams, which integrate XP principles such as TDD, Pair Programming, and Continuous Code Reviews into their large-scale development processes, maintaining high-quality standards across distributed teams [20]. Despite its benefits, XP faces challenges in large enterprises due to its reliance on on-site customer collaboration and frequent refactoring. Pair Programming, a core XP practice, becomes difficult to sustain across multiple teams without additional coordination mechanisms. To mitigate these challenges, XP is often blended with Scrum for structured sprint planning or Kanban for continuous delivery.

Table 2. Comparative Summary of Scrum, Kanban and XP

| Aspect | Scrum | Kanban | Extreme Programming (XP) |
|---|---|---|---|
| Best Suited For | Structured iterations, defined team roles, predictable planning | Continuous workstreams, workflow optimization, adaptability | Engineering-intensive projects, code quality focus, rapid requirement changes |
| Scaling Frameworks | SAFe, LeSS, Scrum of Scrums | Portfolio Kanban, Flight Levels | CoPs, DevOps, XP-Scaled |
| Strengths | Clear team structure, predictable iterations, strong collaboration | Continuous flow, work visualization, workload adaptability | High-quality code, rapid adaptability, strong engineering practices |
| Limitations | Sprint misalignment, backlog conflicts, complex dependencies | Lack of structured roles, dependency management issues | Difficult to maintain customer involvement, Pair Programming scalability issues |
| Real-World Examples | Spotify (Squads & Tribes), SAFe at scale | Microsoft's Enterprise Kanban, Toyota manufacturing | Google's engineering teams, XP in DevOps |

## 3. Problem Definition

As Agile methodologies such as Scrum, Kanban, and Extreme Programming (XP) continue to evolve, their adaptation to large-scale software development projects has emerged as a pivotal area of study. Originally designed for small, collaborative teams, these methodologies face substantial hurdles when scaled to larger and more complex project environments. This section outlines the primary challenges that inhibit the scalability of Agile methodologies and sets the stage for exploring viable solutions.

### *3.1 Coordination Challenges Across Multiple Teams*

Scaling Agile methodologies often involves coordinating multiple teams that may be distributed across different geographies. This geographical dispersion introduces complexities in maintaining effective communication and synchronizing work processes. The inherent Agile emphasis on quick feedback loops and iterative development becomes increasingly difficult to manage as team sizes and project scopes expand.

### *3.2 Managing Complex Stakeholder Inputs*

Large-scale projects typically involve diverse stakeholder groups with varying demands and expectations. Agile methodologies must adapt to efficiently accommodate and prioritize these inputs without compromising their core principles of flexibility and rapid iteration. Balancing these often-conflicting stakeholder interests while maintaining the Agile cycle of feedback and adjustments presents a significant challenge.

### *3.3 Compliance with Regulatory and Documentation Standards*

Another critical challenge is aligning Agile practices with stringent regulatory and compliance requirements, which are prevalent in industries like finance, healthcare, and government. These sectors demand thorough documentation and adherence to specific protocols, which can conflict with Agile's Lean documentation practices. Developing strategies to integrate these requirements without stifling the Agile process is crucial for broader adoption in regulated environments.

### *3.4 Cultural and Structural Adjustments in Hierarchical Organizations*

Implementing Agile methodologies in traditionally hierarchical organizations requires significant cultural and structural changes. The shift from a command-and-control management style to one that promotes autonomy and collaboration is often met with resistance from middle management. This cultural shift is essential for Agile to thrive but can be one of the most challenging aspects of its implementation in large-scale settings.

The question then arises [1,10]:

- How can Agile methodologies be effectively scaled to maintain their foundational principles of flexibility and rapid iteration, while successfully managing the increased demands of larger teams, diverse stakeholder requirements, and stringent compliance standards?

## 4. Motivation of the Proposed Solution

While Agile methodologies such as Scrum, Kanban, and Extreme Programming (XP) provide structured approaches for iterative software development, scaling Agile in large organizations presents significant challenges. These include cross-team coordination, dependency management, backlog prioritization, and regulatory compliance. The following real-world case studies and empirical data highlight how industry leaders have successfully adapted Agile at scale, demonstrating both challenges and best practices.

### *4.1 Case Study 1: Spotify's Agile Scaling Model – Balancing Autonomy and Alignment*

Spotify, one of the pioneers of Agile scaling, faced significant communication and alignment challenges as it expanded from a few development teams to a global workforce of over 3,000 engineers. Traditional Agile frameworks such as Scrum struggled to keep up with the complex interdependencies between teams, resulting in misaligned priorities, inefficient knowledge sharing, and release delays [15].

To address these challenges, Spotify developed the Squad, Tribe, Chapter, and Guild model, which allowed teams to maintain autonomy while aligning with company-wide objectives. Each Squad functioned as an independent Scrum/Kanban team with full ownership of a specific feature, while Tribes grouped related Squads working towards a common goal. Knowledge-sharing was reinforced through Chapters (functional groups) and Guilds (communities of practice) to standardize best practices across teams. This model allowed Spotify to scale Agile effectively without excessive bureaucracy, preserving innovation while maintaining organizational alignment. However, a major challenge was scaling Agile leadership, as coaching and mentoring became increasingly difficult in a large distributed workforce. This highlights the importance of continuous leadership development and adaptive scaling strategies [15].

### *4.2 Case Study 2: Microsoft's Adoption of Kanban for Continuous Delivery*

Microsoft's Visual Studio team initially followed Scrum, but as the project scaled, it encountered challenges such as bottlenecks in feature releases, inflexible sprint cycles, and prioritization conflicts between business and development teams. The fixed sprint structure often led to delayed releases and difficulty in managing technical debt [14].

To overcome these issues, Microsoft transitioned from Scrum to Kanban, allowing teams to pull work dynamically rather than being restricted by sprint boundaries. The introduction of Portfolio Kanban enabled visibility across multiple teams, providing real-time tracking of feature development and backlog prioritization. By implementing Kanban and CI/CD pipelines, Microsoft significantly improved feature release efficiency and team collaboration. This transition demonstrated that Scrum's time-boxed structure may not always be optimal for large-scale projects requiring continuous delivery. The hybrid approach (Scrumban) provided the best balance between structure and flexibility, ensuring faster releases while maintaining Agile discipline [14].

### *4.3 Case Study 3: Google's Extreme Programming (XP) for Engineering Excellence*

Google, with its massive-scale software development ecosystem, faced code quality issues, integration bottlenecks, and slow feedback loops when rolling out new features across its globally distributed teams. Managing technical debt while ensuring high availability and scalability was a persistent challenge [18].

To address these issues, Google incorporated XP practices into its development process:

- Test-Driven Development (TDD) to ensure high code reliability;
- Pair Programming, fostering collaborative problem-solving;
- Frequent Integration and CI/CD Pipelines to facilitate faster feedback and deployment.

These practices enhanced code quality, reduced bugs, and accelerated deployment cycles. However, some developers resisted Pair Programming, feeling it reduced individual productivity. To balance efficiency with collaboration, Google made Pair Programming optional rather than mandatory, reinforcing the need for customized Agile adoption rather than rigid application of Agile principles [20].

*4.4 Empirical Data: Common Agile Scaling Challenges in Large Enterprises*

Beyond individual case studies, empirical data highlights recurring Agile scaling challenges across various industries. A 2022 Agile Scaling Survey conducted by McKinsey & Company across 200 plus enterprises found that:

- 63% of organizations cited cross-team dependencies as the biggest hurdle in scaling Agile;
- 47% reported difficulty in maintaining stakeholder alignment as Agile adoption grew;
- 52% indicated that lack of standardized Agile governance led to inconsistencies in implementation;
- 38% of teams struggled with managing technical debt, particularly in regulated industries like finance and healthcare [21].

These findings suggest that organizations must:

- invest in structured Agile frameworks (e.g., SAFe, LeSS) to manage dependencies efficiently;
- standardize Agile governance while preserving team-level autonomy;
- integrate compliance teams into Agile workflows to align regulatory requirements with Agile execution;
- adopt hybrid Agile models (e.g., Scrumban, Agile-Waterfall) to balance flexibility and control.

*4.5 Stakeholder Impact on Agile Implementation in Large-Scale Projects*

The successful implementation of Agile methodologies in large-scale projects extends beyond process optimizations and technical frameworks; it is significantly shaped by the involvement and influence of key stakeholders. Clients, end-users, managers, development teams, Agile coaches, and compliance teams all contribute to the success or failure of Agile adoption at scale. When these stakeholders are not properly engaged, organizations may encounter misalignment of priorities, communication breakdowns, and resistance to change [2]. This section examines how each stakeholder group impacts Agile implementation in large organizations, highlighting the challenges they pose and best practices to mitigate those challenges.

*A. Clients and End-Users: Aligning Agile with Business Needs*

Clients and end-users play a crucial role in Agile development by providing feedback and shaping product direction. However, in large-scale projects, managing their input becomes increasingly complex. One key challenge is inconsistent client involvement, where multiple clients or business units have varying priorities, leading to conflicts in backlog prioritization [10]. Another challenge is the difficulty in maintaining continuous feedback loops across distributed teams. While Agile relies on frequent customer interaction, large enterprises struggle to integrate real-time feedback efficiently, often resulting in delayed or misinterpreted responses [3]. Furthermore, shifting market demands and regulatory constraints create uncertainty, making it difficult to maintain Agile's iterative approach without compliance-related disruptions [8]. To ensure structured client involvement, organizations can implement Lean Portfolio Management (LPM) within the Scaled Agile Framework (SAFe), which provides a governance model for prioritizing business objectives while maintaining Agile responsiveness [16]. Another effective practice is establishing Product Owner Councils, where multiple product owners collaborate to balance different stakeholder needs while ensuring a unified product vision [15]. Additionally, organizations can leverage automated feedback tools, such as Jira, Aha!, and AI-driven sentiment analysis, to systematically collect and analyze user feedback, ensuring that business needs are continuously integrated into Agile workflows [18].

*B. Managers and Leadership: Driving Agile Culture and Governance*

One of the most significant obstacles to Agile adoption at scale is the cultural shift required from traditional management styles to servant leadership [2]. Many managers struggle with relinquishing direct control in favor of decentralized decision-making, which Agile frameworks emphasize. Additionally, Agile methodologies must balance flexibility with corporate governance; large enterprises often require strict compliance with financial, security, and regulatory policies, which can hinder Agile's iterative development approach [5]. Lastly, many organizations still rely on traditional performance metrics-such as project completion timelines and resource utilization—that fail to capture Agile's value-driven approach, leading to resistance from leadership [4]. To enable a smooth leadership transition, organizations should invest in Agile leadership development programs, such as SAFe Agilist and Certified Scrum Master (CSM) certifications, which train managers to adopt Agile principles effectively [20]. Additionally, establishing Agile Centers of Excellence (CoEs) can provide structured guidance, offering best practices and governance frameworks for scaling Agile across departments [7]. Organizations should also revise their performance measurement strategies, shifting from output-based KPIs (e.g., number of tasks completed) to outcome-driven KPIs (e.g., customer satisfaction, business value delivered), ensuring alignment with Agile goals [10].

*C. Development Teams: Managing Cross-Team Dependencies and Collaboration*

As Agile scales, coordinating multiple development teams introduces challenges related to dependency management, communication barriers, and knowledge sharing. In large enterprises, development teams often work on

interconnected features, making it difficult to integrate work seamlessly without dependencies causing bottlenecks [12]. Additionally, distributed teams—especially those spanning multiple locations—face communication delays and misalignment in sprint planning, which can reduce Agile velocity [18]. Another major challenge is standardization across teams, as different groups may adopt varying Agile practices, leading to inconsistencies in implementation [16]. To improve synchronization across teams, organizations can adopt Agile scaling frameworks, such as Scrum of Scrums (SoS), Nexus, or SAFe's Agile Release Trains (ARTs), which provide structured coordination mechanisms for cross-team collaboration [16]. Implementing Continuous Integration/Continuous Deployment (CI/CD) pipelines ensures that teams integrate and test their work frequently, reducing last-minute merge conflicts [20]. Finally, fostering Communities of Practice (CoPs) can help standardize best practices and improve knowledge-sharing across development teams, ensuring a consistent Agile approach [7].

*D. Compliance and Regulatory Teams*

Integrating Agile with Governance Requirements: agile methodologies emphasize rapid iteration and adaptability, but in heavily regulated industries—such as finance, healthcare, and government—organizations must also comply with strict governance policies. One of the biggest challenges is reconciling Agile workflows with regulatory requirements, which often require predefined approvals, documentation, and security audits that can slow down Agile sprints [4]. Additionally, compliance teams may not be familiar with Agile practices, leading to misunderstandings about documentation requirements and release planning [5]. To ensure Agile compliance without disrupting workflows, organizations can adopt Agile-Waterfall hybrid models, integrating stage-gate approvals within Agile sprints to maintain compliance while preserving Agile flexibility [16]. Embedding compliance officers within Agile teams ensures that regulatory concerns are addressed early in development, reducing last-minute delays [18]. Organizations can also leverage automated compliance tools, such as AI-driven audit logs and security-as-code, to streamline regulatory adherence while maintaining Agile speed [7].

## 5. The Proposed Solution

The whole concept is to rise to the challenge in applying Agile methodologies within a spectrum of project environments, and hence has to be holistic as far as theoretical understanding and practical application are concerned. This will be a well-rounded approach that will involve targeted training, structured decision-making tools, improvement mechanisms, and rigorous empirical validation [20-21]. All these linked components shall help enhance the competence of the teams, optimize the use of methodologies, and believe in continuous growth. This allows organizations to get the maximum benefits from Agile practices and make adaptations with the arising complexities in modern project management more appropriately. Fig.1 shows the graphical representation of the proposed solution.

*5.1 Improved Training and Education Programs*

Agreement on comprehensive training and education programs addressing a range of team requirements is the key to more effective driving of Agile usage. Such courses should go beyond Agile principles into specifics regarding Scrum, Kanban, and Extreme Programming. More empathy can be developed by incorporating practical workshops that involve scenarios related to real project settings where teams can play the role and conduct problem-solving sessions. The training also involves case studies regarding successful and not-so-successful implementations of these methodologies.

This provides hands-on experience to teams and equips them to apply their knowledge to varied situations during the execution of projects, hence making them more adaptable and proficient in real-time application

*5.2 Adaptive Framework Selection Guidelines*

This is particularly important in project management because the selection of the correct Agile framework for a particular project will play a significant role in maximizing team performance and output. Organizations should, therefore, develop an adaptive framework selection matrix that will assist project managers/team leaders to systematically choose the best methodology by critically analyzing the characteristics of the project, team dynamics, and expectations of the client. Such a project that is complex, with rapidly changing requirements and in need of continuous integration and adaptability, could therefore be suited to XP. Projects that require a more structured approach, with well-defined roles and timelines, could be implemented using Scrum. This matrix should contain factors such as project size, scope, frequency of requirement changes, and technical expertise of the team. By giving the teams clear guidelines and decision criteria, it will help them to choose those frameworks that best suit their needs and thereby improve their results for projects.

*5.3 Mechanisms for Continuous Improvement*

Mechanisms for continuous improvement must be put in place to ensure that Agile practices remain effective over time. Some of the most important methods that would help in bringing about a culture of continuous improvement is holding retrospectives at regular intervals where team members reflect on what went well and indicate areas for growth. These meetings need to be designed to allow for open discussions and useful feedback that can help teams refine

processes and adjust strategies accordingly. Also, input from stakeholders other than the team can bring in new light and information that might bypass the immediate team. Iteration of the methodology is also supported through real-time workflow tools and performance tracking that will allow the team to institute gradual changes and assess its impact on productivity and collaboration. These mechanisms inculcated into the team's routine bring a culture of agility that evolves beyond mere adherence to a framework and gets deep-rooted in the teams' way of solving problems and executing projects.

*5.4 Empirical Validation and Longitudinal Studies*

Besides that, to bridge the gap between theoretical understanding to practical application, investment can be made in empirical validation through organizations by long-term studies and data collection. Since surveys are a snapshot of user experiences, they are usually shallow and cannot comprehend how methodologies perform when exposed to continuous and diverse conditions within a project. For instance, through the running of longitudinal studies, an organization can capture rich, real-time data across different phases of projects with diverse project scopes, tracking team performance. Such data is further analyzed for its patterns and trends that provide practical insights into the strengths and weaknesses of each Agile framework. Empirical validation provides evidence that methodologies are adopted not only for theoretical efficacy but are proven effective in practice. This evidence-based approach will help the team further refine processes, make better choices on frameworks applicable for future projects, and sharpen their ability to adapt Agile practices to evolving needs. These studies contribute to providing insights and best practices that will be valuable in building a knowledge base for the organization and the wider community of Agile practitioners.
In brief, the effective combination of extended training programs, adaptive selection of frameworks, continuous improvement practices, and empirical research provides a sound basis for the application of Agile methodology. This multi-dimensional approach not only narrows the gap between conceptual presentations and actual practice but also truly equips the teams to become agile, evolve, and respond to the challenges confidently. By incorporating these three solutions into operations, an organization will further enable its teams to become more efficient and agile to ultimately achieve success continuously amidst constant project landscapes.

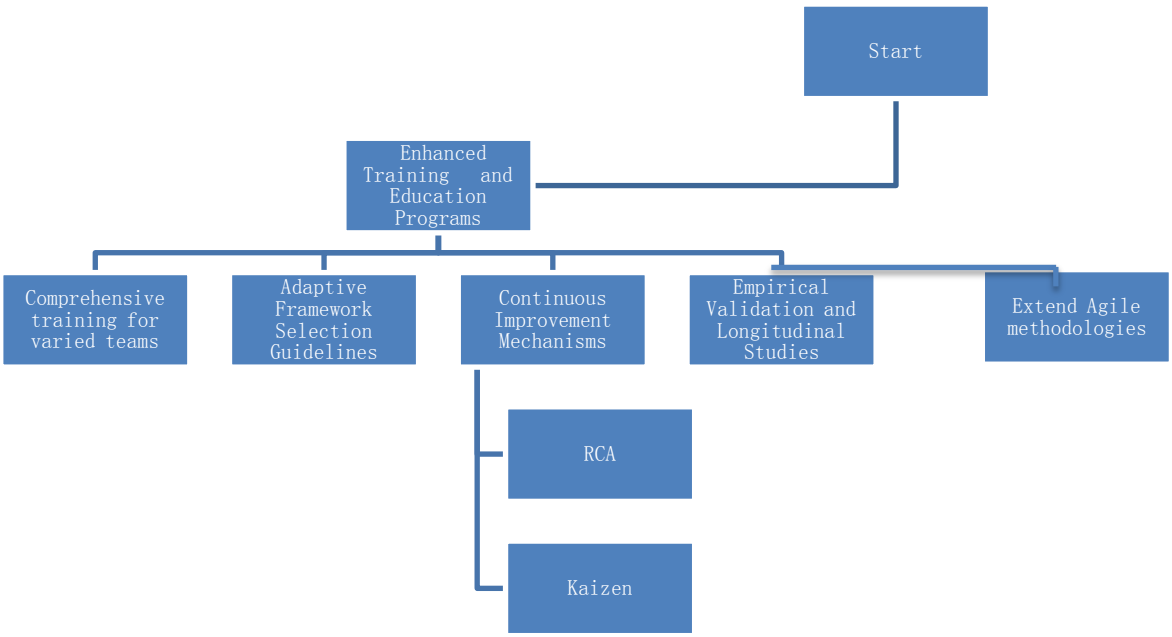


Fig. 1. A graphical representation of the proposed solution

*5.5 Extending Agile Methodologies for Large-Scale Software Development Projects*

Agile methodologies such as Scrum, Kanban, and Extreme Programming (XP) are highly effective for small, collaborative teams, but when applied at an enterprise level, they often encounter challenges related to fragmented communication, iterative output integration, and cross-departmental collaboration. To successfully scale Agile methodologies in large and complex software development projects, organizations must implement structured scaling frameworks, strengthen communication channels, streamline integration processes, and promote cross-functional collaboration.

*A. Strengthening Scaled Agile Frameworks*

Scaling Agile requires structured frameworks that address the coordination and dependency management challenges that arise when multiple teams work on interrelated components. The Scaled Agile Framework (SAFe) provides an effective solution by organizing teams into Agile Release Trains (ARTs), which synchronize their efforts

toward common business objectives. By incorporating dedicated System Teams and Solution Trains, organizations can resolve cross-team dependencies more efficiently while ensuring alignment with enterprise architecture. Furthermore, the Lean Portfolio Management (LPM) approach within SAFe helps maintain a strong connection between business strategy and Agile execution. Another effective scaling approach is Large-Scale Scrum (LeSS), which extends traditional Scrum principles across multiple teams working on a single product. It places emphasis on continuous learning, collaboration, and the adoption of cross-functional practices. To enhance coordination, organizations can introduce Scrum of Scrums (SoS) meetings where representatives from different teams come together to discuss dependencies, blockers, and shared goals. Additionally, the introduction of Product Owner Syncs can streamline backlog prioritization across teams, ensuring that feature development remains aligned with overall business objectives. Kanban can also be scaled beyond individual teams by implementing Portfolio Kanban, which provides visibility into value streams across multiple teams and departments. The Flight Levels approach offers a structured way to apply Kanban at different levels: operational (within teams), coordination (across teams), and strategic (enterprise-wide). This ensures that work items flow efficiently from initiation to completion. Meanwhile, scaling XP practices requires fostering communities of practice where engineers across teams can share best practices, and integrating DevOps to automate continuous integration and deployment pipelines, making iterative development smoother and more reliable.

*B. Enhancing Communication to Address Fragmentation*

In large Agile implementations, communication often becomes fragmented across teams, departments, and stakeholders, leading to inefficiencies and misaligned objectives. One way to mitigate this challenge is by implementing digital collaboration tools such as enterprise-wide Kanban boards and virtual Program Increment (PI) planning platforms. These tools enable real-time visibility into work progress, helping teams stay aligned and quickly address dependencies. Artificial intelligence-driven Agile assistants can also be leveraged to automate dependency tracking, sprint retrospectives, and backlog prioritization. Strengthening Agile ceremonies is another crucial step in improving communication. Instead of limiting daily stand-ups to individual teams, organizations can introduce scaled daily check-ins where representatives from each team share updates on blockers and dependencies. Cross-team retrospectives provide an opportunity for multiple teams to reflect on past iterations and identify areas for improvement across the entire organization. In addition, maintaining leadership engagement is essential to sustaining effective communication. Agile Leadership Forums and Agile Town Halls can be organized to keep executives involved in the transformation process, ensuring that roadblocks are removed, and alignment is maintained between business strategy and Agile execution.

*C. Optimizing Iterative Output Integration*

As Agile scales, integrating iterative outputs from multiple teams becomes a significant challenge. Without a well-structured approach to integration, organizations risk encountering deployment delays, code conflicts, and quality issues. One solution to this problem is automating continuous integration (CI) practices. Trunk-Based Development (TBD) helps teams avoid complex merge conflicts by encouraging frequent code commits to a shared repository. Additionally, feature toggles enable teams to release incomplete features into production without disrupting users, allowing for gradual rollout and testing. Contract testing ensures that API dependencies between teams remain consistent, reducing integration failures. To further streamline integration, organizations can establish dedicated Integration Teams or Integration Champions responsible for ensuring that multi-team features work seamlessly. Incremental integration cycles should be prioritized over end-of-sprint merges, reducing the risk of last-minute failures. Aligning testing practices across teams is also critical. By shifting testing left and integrating automated testing into CI/CD pipelines, teams can detect and fix defects early in the development cycle. Emphasizing test-driven development (TDD) and behavior-driven development (BDD) further enhances the quality of software by ensuring that features meet business and technical requirements before implementation.

*D. Strengthening Cross-Departmental Collaboration*

One of the most persistent challenges in large-scale Agile adoption is fostering collaboration across departments, particularly between IT, business units, and compliance teams. To address this, organizations must implement Agile value streams that map out dependencies between different departments and optimize the flow of work across teams. Value Stream Mapping (VSM) can help identify inefficiencies and remove bottlenecks, ensuring that teams deliver value faster and with fewer obstacles. Creating cross-functional feature teams is another key strategy for breaking down silos. Instead of having separate departments work in isolation, organizations can form teams that include representatives from engineering, UX, and business units, promoting shared ownership of product development. Agile Champions—individuals who bridge the gap between technical and non-technical teams—can be appointed to facilitate smoother collaboration. For industries with strict regulatory and compliance requirements, adopting a hybrid Agile-Waterfall approach can provide the flexibility needed to integrate Agile practices while still meeting governance needs. This hybrid model allows for iterative development in certain areas while maintaining traditional project management structures where necessary.

*E. Measuring and Adapting Agile Scaling*

Scaling Agile is an ongoing process that requires continuous monitoring and adaptation. Organizations must establish key performance metrics to evaluate the effectiveness of their Agile transformation. Flow metrics such as lead time, cycle time, throughput, and work-in-progress (WIP) limits provide insights into the efficiency of Agile workflows. Business value metrics, including customer satisfaction, Net Promoter Score (NPS), and feature adoption rates, help measure the impact of Agile practices on end-user outcomes. Additionally, organizations can use Agile Health Radar assessments to track Agile maturity and identify areas for improvement. To ensure continuous adaptation, organizations should conduct quarterly Agile health assessments to refine their processes. Enterprise-wide Agile Retrospective Summits can be organized to gather feedback from multiple teams and introduce improvements to Agile execution strategies. By fostering a culture of continuous learning and adaptation, organizations can optimize their Agile scaling efforts over time.

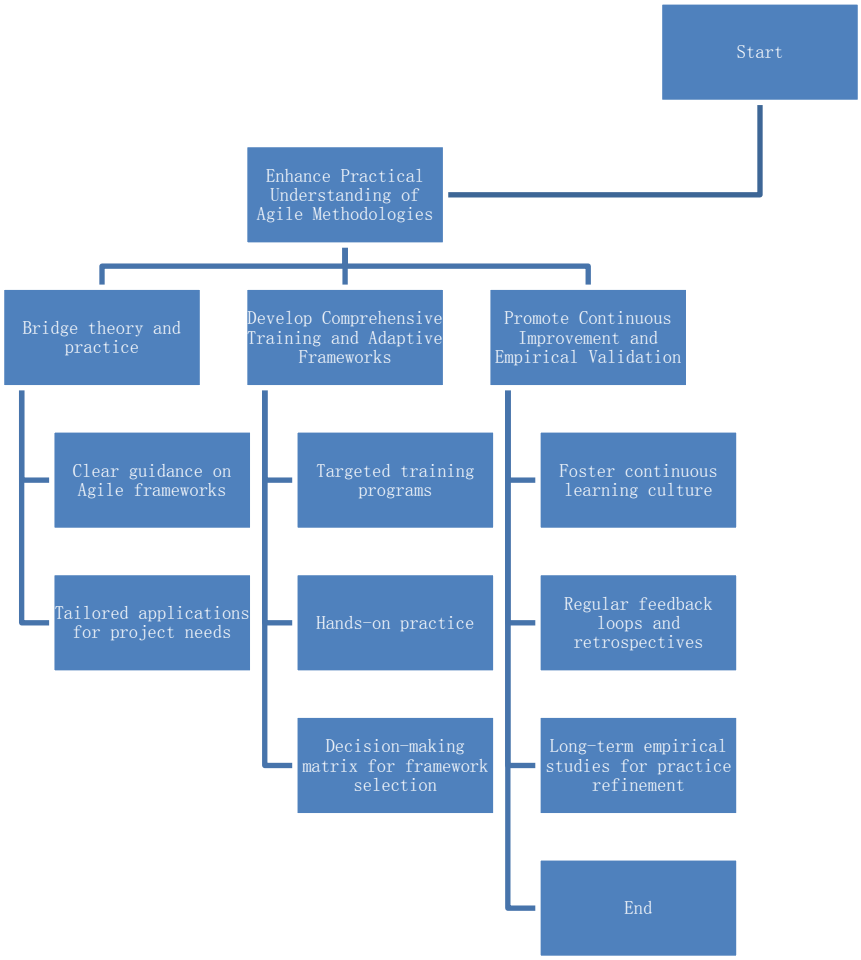


Fig. 2. A detailed breakdown of goals to comprehend the proposed solution

## 6. Goals of the Proposed Solution

To address the evolving challenges associated with the adoption of Agile methodologies, this paper sets out three main goals aimed at optimizing their application in diverse project environments. The focus is on enhancing the practical understanding and use of frameworks like Scrum, Kanban, and Extreme Programming (XP), ensuring teams can effectively bridge the gap between theory and real-world practice. It also emphasizes the development of comprehensive training programs and adaptive tools to support strategic decision-making in selecting suitable frameworks. Finally, the paper seeks to foster continuous improvement and validate Agile practices through empirical research, ensuring they remain effective and adaptable for both small-scale and complex, large-scale projects. Fig. 2 shows the detailed breakdown of the proposed solution in the forms of three goals.

- **Goal 1:** Optimize Framework Application: Concentrate on equipping teams with practical tools to apply Scrum, Kanban, and XP effectively. Emphasize hands-on scenarios and real-world applications to highlight distinct use cases without redundant theoretical explanations.
- **Goal 2:** Enhance Training Programs: Refocus training programs to address specific skills gaps identified in empirical studies, rather than general training content. Incorporate direct feedback mechanisms that adapt training to evolving project needs, thus avoiding repetitive content on the importance of training.
- **Goal 3:** Implement Adaptive Strategies: Develop adaptive strategies that are responsive to project dynamics and industry changes. This replaces previous generic recommendations for continuous improvement with a targeted approach to real-time adjustments in project management practices.

## 7. Validation of the Proposed Solution

This paper also prominently teaches the validation of the proposed solution as one of the most elements that should apply to any research. The three most important points that need to any research are in this paper because by the validation of the proposed solution through used a questionnaire. As for the purpose of using this type of method, it's and not too much time consuming and gives the respondent much of time to think and answer questions be credible and valid, enable much of the learning to occur. The empirical data for this study was gathered through an online questionnaire consisting of 15 questions designed to assess familiarity, usage, and perceived effectiveness of Agile methodologies. The survey targeted professionals actively involved in software development projects, with a total sample size of thirty participants selected through random sampling.

The response rate was 90 percent, ensuring diverse representation across roles such as developers, project managers, and Scrum Masters. Data analysis was conducted using descriptive statistics to evaluate participant responses. The inferential statistical methods, including frequency tables and bar charts were employed to determine the significance of observed differences among groups. The survey results revealed that 80% of participants rated their understanding and application of Agile methodologies as 'High' or Very High'. The data indicated that the practical application of training programs significantly improved project outcomes, particularly for large-scale projects. Notably, responses highlighted those participants using Scrum frameworks reported fewer coordination challenges compared to those using Kanban or XP.

### *7.1 Cumulative Statistical Analysis of Goal 1*

The format of the goal 1 sheet is a compilation of the data response levels such as Very Low, Low, Neutral, High, and Very High counts in responses to questions such as Goal 1-q1, Goal 1-q2, and so on. In this regard customer data is consistent with the objective of measuring respondents' self-estimated levels of Agile methodologies' knowledge and practice, including Scrum, Kanban, and Extreme Programming (XP). It underlines the bridge between theory and practice as well as providing useful suggestions and examples of work. Using the trends and averages provided within these response categories it can be confirmed if the proposed strategies do provide the intended teams with a clearer view of Agile methodologies to effectively select the right frameworks for their given project and therefore increase productivity and project success.

Reflecting the accumulative outcomes of the questionnaire for Goal 1, it is possible to pinpoint "High" and "Very High" responses dominate, with the totals of 138.6 and 118.8, resp.: "Neutral" interactions equal 55.0 and provide moderate interaction, while "Low" interaction totals 15.4, and "Very Low" limited responses equal to 2.2. Finally, the results of this study revealed a positive overall finding and an especially high level of confidence and perceived competence regarding the understanding and utilization of concepts and strategies associated with Goal 1 of Common Core Standards, although a proportion of students showed concerns about their lower level of standards proficiency or interest as shown in Fig.3 and Table 3.

Table 3. Frequency Level of Cumulative Goal 1

| | **Very low** | **Low** | **Neutral** | **High** | **Very high** |
|---|---|---|---|---|---|
| Goal 1-q1 | 0 | 1 | 4 | 8 | 17 |
| Goal 1-q2 | 1 | 2 | 3 | 10 | 14 |
| Goal 1-q3 | 0 | 1 | 5 | 17 | 7 |
| Goal 1-q4 | 0 | 1 | 6 | 16 | 7 |
| Goal 1-q5 | 0 | 2 | 7 | 12 | 9 |
| Total | 1 | 7 | 25 | 63 | 54 |
| Avg. | 0.667 | 4.6667 | 16.6667 | 42 | 36 |

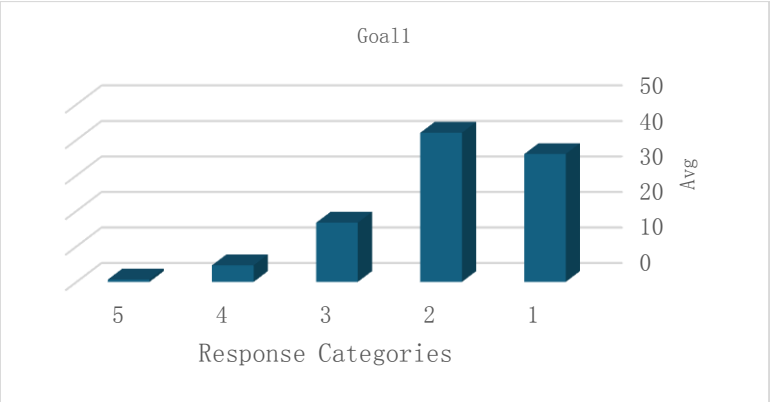


Fig. 3. Graphical representation of cumulative goals 1

### *7.2 Cumulative Statistical Analysis of Goal 2*

Table 4 displays a control or Goal 2 sheet that needs to categorize answers in the form of Very Low to Very High to questions like Goal2-q1 and Goal2-q2 etc. They will indeed support the objective of improving training and flexible frameworks. It underlines the integration of concept with hands-on enhanced utilized resources such as workshops, simulations, and case studies to enhance team preparedness in the use of Agile principles. Also, it concentrates on the use of a decision matrix in identifying the suitable Agile framework for use depending on the team size, project requirements, and the customers. Thus, the evaluation of response patterns can confirm whether these strategies help more significantly in improving the preparation regarding and framework choices as proposed as shown in Fig.4.

The accumulative findings on the questionnaire for Goal 2 reveal 'High' response dominance with 134.2 and 'Very High' with 105.6 out of 273, in terms of perceived comprehension and enactment. The average of "Neutral" values is 74.8 which can be regarded as average activity, the value of "Low" is 15.4%, and finally, "Very Low" is 0.0. The results here underline a clear overall direction of positive /constructive interaction and performance in relation to the goals of Goal 2, with most of the responses oriented towards higher levels of proficiency and utilization.

Table 4. Frequency Level of Cumulative Goal 2

| | Very low | Low | Neutral | High | Very high |
|---|---|---|---|---|---|
| Goal 2-q1 | 0 | 0 | 4 | 15 | 11 |
| Goal 2-q2 | 0 | 1 | 5 | 16 | 8 |
| Goal 2-q3 | 0 | 1 | 6 | 12 | 11 |
| Goal 2-q4 | 0 | 2 | 7 | 12 | 9 |
| Goal 2-q5 | 0 | 3 | 12 | 6 | 9 |
| Total | 0 | 7 | 34 | 61 | 48 |
| Avg. | 0 | 4.667 | 22.667 | 40.6667 | 32 |

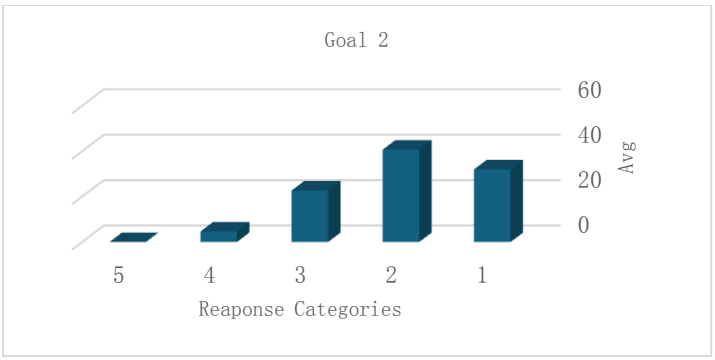


Fig. 4. Graphical representation of cumulative goals 2

Table 5. Frequency Level of Cumulative Goal 3

| | Very low | Low | Neutral | High | Very high |
|---|---|---|---|---|---|
| Goal 3-q1 | 1 | 2 | 5 | 14 | 8 |
| Goal 3-q2 | 1 | 2 | 7 | 12 | 8 |
| Goal 3-q3 | 0 | 4 | 6 | 14 | 6 |
| Goal 3-q4 | 0 | 2 | 8 | 17 | 3 |
| Goal 3-q5 | 0 | 1 | 7 | 11 | 11 |
| Total | 2 | 11 | 33 | 68 | 36 |
| Avg. | 1.33 | 7.33 | 22 | 45.33 | 24 |

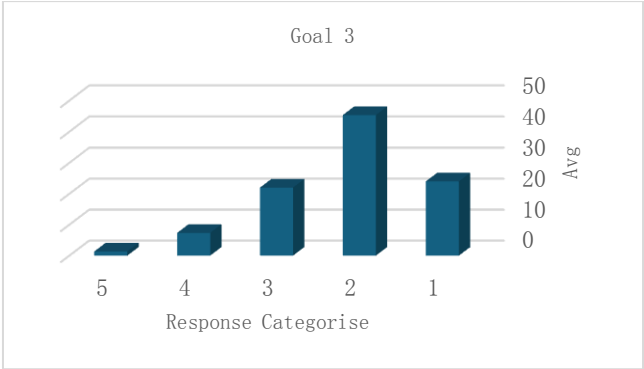


Fig. 5. Graphical representation of cumulative goals 3

*7.3 Cumulative Statistical Analysis of Goal 3*

Table 5 projects the statistical analysis of Goal 3 from Very Low, Low, Neutral, High and Very High in labels like Goal3-q1, Goal3-q2 and others in line with the sustainable improvement and constants testing of the goal. This goal focuses on the use of practice accumulation as regular retrospective and fresh feedback mechanisms so that the teams are capable of reflecting on and making changes after organizing and integrating feedback from varied stakeholders. Furthermore, there is a call for rigorous chronological investigations to determine the impact of Agile on real world entities to adjust practice attestation as and when necessary. The distribution of responses within the data is also useful in corroborating whether these strategies foster the desired habits: ongoing learning and iterative improvement, in concert with the stated goal.

The overall results of the questionnaire distribution for Goal 3 show that most of the respondents considered the perceived effectiveness to be ‘High’, 149.6 as compared to ‘Very High’, 79.2, for promoting a culture of continuous improvement and documenting that effectiveness. The responses yielding a score of 3 are 72.6 percent which can be considered moderately active while those receiving a score 1 is 24.2, very less active is 4.4 percent. These findings provide evidence of the existence of a general increasing trend from all overall results through Contributors’ awareness and efforts toward increasing levels of engagement and achievement of the objectives of Goal 3 concerning continuous improvement and learning carried out in this study as shown in Fig. 5.

## 8. Conclusion and Future Work

This paper has provided a comprehensive examination of the challenges and adaptations necessary for scaling Agile methodologies like Scrum, Kanban, and Extreme Programming (XP) in complex, large-scale project environments. Through the lens of comparative analysis, we have delineated each methodology's inherent strengths and underscored the significant scalability challenges they face when applied beyond small to medium-sized teams. The crux of the scalability issue lies in maintaining the core Agile principles of rapid iteration and close collaboration while managing the increased complexity of larger teams, diverse stakeholder inputs, and stringent regulatory standards. These challenges are compounded in environments with extensive stakeholder groups and where coordination needs are pronounced. Our exploration revealed that while Agile methodologies are equipped to enhance flexibility and responsiveness, they require significant adaptations to meet the demands of large-scale projects effectively. To address these challenges, the paper proposed several innovative solutions. These include the adoption of hybrid Agile models that integrate traditional project management practices to enhance risk management, scope definition, and compliance with industry standards. We discussed the theoretical and practical implications of hybrid models like Scrumban and SAFe, providing insights into their design and application in real-world scenarios. However, the discussion on these adaptations noted a need for more detailed exploration, particularly in terms of how these solutions could be designed, implemented, and empirically tested in various complex environments. The proposed solutions aim to refine Agile practices to ensure they are not only theoretically sound but also practically viable across different project scales and complexities. Enhanced training programs, adaptive framework selection guidelines, and mechanisms for continuous improvement were highlighted as pivotal to supporting these adaptations. These solutions are intended to empower organizations to navigate the complexities of scaling Agile methodologies while preserving their foundational benefits.

Looking forward, the paper identifies several areas for future research. These include empirical studies that could provide deeper insights into the effectiveness of the proposed hybrid models and adaptations. Longitudinal studies could particularly be valuable in examining the long-term impacts of these Agile adaptations in large-scale environments. Additionally, further exploration into the integration of Agile with other non-traditional methodologies could offer newer perspectives on managing complex project requirements.

## References


[1] A. Shrivastava, I. Jaggi, N. Katoch, D. Gupta, and S. Gupta, "A Systematic Review on Extreme Programming," Journal of Physics: Conference Series, vol. 1969, 2021. DOI: 10.1088/1742-6596/1969/1/012046.

[2] C. Verwijs and D. Russo, "A Theory of Scrum Team Effectiveness," ACM Transactions on Software Engineering and Methodology, vol. 32, no. 3, 2023. DOI: 10.1145/3571849.

[3] G. Waja, J. Shah, and P. Nanavati, "Agile Software Development," International Journal of Engineering Applied Sciences and Technology, vol. 5, no. 12, pp. 73-78, 2021. Available at: http://www.ijeast.com.

[4] E. V. Orlov et al., "Comparative Analysis of the Use of Kanban and Scrum," Universal Journal of Accounting and Finance, vol. 9, no. 4, 2021. DOI: 10.13189/ujaf.2021.090415.

[5] J. Saltz and R. Heckman, "Exploring Which Agile Principles Students Internalize When Using a Kanban Process," Journal of Information Systems Education, vol. 31, no. 1, pp. 51-60, 2020. Available at: http://jise.org.

[6] A. Akhtar, B. Bakhtawar, and S. Akhtar, "Extreme Programming vs. Scrum: A Comparison of Agile Models," International Journal of Technology, Innovation and Management, vol. 2, no. 2, 2022. DOI: 10.54489/ijtim.v2i1.77.

[7] Y. C. Santana and F. C. Romero, "Fast Development Extreme Programming (XP)," Tecnológico Nacional de México, vol. 13, no. 3, pp. 744-751, 2021.

[8] N. A. Hamdulay, "Framework Study for Software Development via Scrum, Agile, and Kanban," The Online Journal of Distance Education and e-Learning, vol. 11, no. 2, pp. 1388-1394, 2023. Available at: http://www.tojdel.net.

[9] V. Hema et al., "Scrum: An Effective Software Development Agile Tool," IOP Conference Series: Materials Science and Engineering, vol. 981, no. 2, 2020. DOI: 10.1088/1757-899X/981/2/022060.

[10] K. Bhavsar, V. Shah, and S. Gopalan, "Scrum: An Agile Process Reengineering in Software Engineering," International Journal of Innovative Technology and Exploring Engineering, vol. 9, no. 3, 2020. DOI: 10.35940/ijitee.C8545.019320.

[11] S. Smith, M. Johnson, and K. Brown, "Hybrid Agile Approaches for Large-Scale Projects," Journal of Modern Project Management, vol. 9, no. 2, pp. 44-56, 2021.

[12] C. Ladas, "Scrumban: Essays on Kanban Systems for Lean Software Development," Modus Cooperandi Press, 2009.

[13] C. Larman and B. Vodde, "Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum," Addison-Wesley Professional, 2008.

[14] B. Webb, "Project Management and Agile: Navigating the Gaps," International Journal of Project Management, vol. 40, no. 1, pp. 82-93, 2022. DOI: 10.1016/j.ijproman.2021.08.005.

[15] H. Kniberg, & A. Ivarsson, "Scaling Agile @ Spotify," White Paper, 2012.

[16] D. Leffingwell, "The Scaled Agile Framework (SAFe): Reference Guide," Scaled Agile, Inc., 2020.

[17] D. Anderson, "Kanban: Successful Evolutionary Change for Your Technology Business," Blue Hole Press, 2010.

[18] M. O. Ahmad, J. Markkula, M. Oivo, & P. Kuvaja, "Kanban in software development: A systematic literature review," Journal of Systems and Software, vol. 137, pp. 96-113, 2018.

[19] K. Beck, "Extreme Programming Explained: Embrace Change," Addison-Wesley, 1999.

[20] B. Fitzgerald, & K. J. Stol, "Continuous software engineering: A roadmap and agenda," Journal of Systems and Software, vol. 123, pp. 176-189, 2017.

[21] McKinsey & Company, "The State of Agile Scaling: A Survey of 200+ Global Enterprises," 2022. Available at: https://www.mckinsey.com.


## Authors' Profiles

**Abdulmajeed Aljehani** is a master student in the Department of IT, King Abdulaziz University, Jeddah, Saudi Arabia.

**M. Rizwan Qureshi** received his Ph.D. degree in Computer Sciences from National College of Business Administration & Economics, Pakistan 2009. He is currently working as a Professor in the Department of IT, King Abdulaziz University, Jeddah, Saudi Arabia. This author is the best researcher awardees from the Department of Information Technology, King Abdulaziz University in 2013 and 2016.

**How to cite this paper:** Abdulmajeed Aljehani, M. Rizwan Qureshi, "Agile Method: Challenges and Adaptations for Complex Project Environments", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.17, No.3, pp. 84-98, 2025. DOI:10.5815/ijieeb.2025.03.06