© SEP 2025 | IRE Journals | Volume 9 Issue 3 | ISSN: 2456-8880
DOI: doi.org/10.64388/IREV9I3-1710544-450

The Impact of Agile Methodology on It Project Success (A Structural Literature Review)

AGBARAOJO YUSUF ABIODUN[1], AGABA HABEEB TITILOPE[2], BAKARE AKEEM ABAYOMI[3], ANWO RHODA[4]

[1, 4]*Student, Department of Management Technology, Lagos State University*
[2]*Student, Department of Business Informatics, University of Debrecen*
[3] *Doctoral Student, Department of Information Science & Systems, Morgan State University*

Abstract: ***Agile methodology have transformed IT project management by providing a flexible and adaptive framework to address the dynamic nature of software development. The influence of agile approaches was evaluated by this study, particularly Scrum, and the corresponding effect on successful IT projects deliveries. The study employed a structural literature review and thematic analysis to identify key success determinants. Research findings shows that agile strategies markedly enhance project efficiency, adaptability, and customer satisfaction in contrast to conventional methods such as the Waterfall methodology. The research indicates that essential success variables are cross-functional team competence, iterative development, and proactive stakeholder engagement. Nevertheless, obstacles remain, especially in implementing agile methodologies across extensive, dispersed teams and tailoring them to varied business cultures. A comparative analysis employing the Iron Triangle concept underscores Agile's enhanced efficacy in managing time, cost, and quality restrictions, though its efficiency fluctuates across diverse project circumstances. The implementation of a structural literature review guarantees a thorough synthesis of previous research, whereas theme analysis discerns repeating patterns and interconnections among main components. This research provides academic and practical insights, elucidating Agile's role in contemporary IT project management and assisting practitioners in harnessing its potential for project success.***

Index Terms- Agile Methodology, IT Project Success, Scrum, Structural Literature Review, Thematic Analysis.

I. INTRODUCTION

The emergence of Agile methodologies is linked to the challenges faced in software development projects, where traditional models like the Waterfall methodology often failed to adapt to rapidly changing needs and client expectations [1]. Agile methods originated as a solution to these difficulties, highlighting iterative development, ongoing feedback, and collaboration across cross-functional teams. Agile software development (ASD) approaches have garnered widespread popularity among conventional software engineers since the late 1990s. They were originally presented in the forms of Scrum, Crystal, Extreme Programming, and various other approaches. Software development with Agile commenced after a consortium of software practitioners convened to formulate the Agile Manifesto. The manifesto marked the inception of the agile transformation in software engineering. One objective of the Agile Manifesto is to enhance software quality and decrease the incidence of unsuccessful information technology (IT) projects.

Agile practitioners contend that agile methodologies more adeptly respond to the fluid nature of modern business conditions by prioritising short iterations of clearly articulated deliverables and fostering transparent communication with stakeholders throughout the development cycle [2]. Agile methods allow a project team to swiftly adjust to unforeseen and rapidly evolving needs prevalent in most software development projects. Certain firms employ agile software development teams (ASDTs) to create software solutions for clients with enhanced efficiency and effectiveness (Nguyen, 2016). Such ASDTs utilize contemporary agile software

© SEP 2025 | IRE Journals | Volume 9 Issue 3 | ISSN: 2456-8880
DOI: doi.org/10.64388/IREV9I3-1710544-450

development methodologies (ASDMs), technologies, and processes [3]. Throughout the years, developers have created and implemented numerous approaches to address the issues of managing software engineering projects. Nonetheless, numerous projects encounter failure, delays, abandonment, or rejection [1]. The successful projects were completed punctually and within budgetary constraints. The others were either contested or hindered. Challenged indicated that the project was completed and became operational, albeit exceeding the budget, arriving later than anticipated, or delivered with fewer features than originally specified, while hindered signified project cancellation. The Standish Group indicated that 57.7% projects encountered difficulties and 31.1% experienced failure.

Consequently, agile software development approaches have been increasingly employed in recent years to address customer and corporate requirements. Agile software development includes a wide range of frameworks and approaches based on the core concepts and values articulated in the Agile Manifesto and Agile concepts [4]. The agile software development process guarantees frequent release cycles, ongoing enhancement, prompt input to swiftly evolving needs, direct communication, as well as higher quality software [5]. Thus, numerous firms have recently opted for agile software development strategies over traditional plan-based approaches. The latest Standish Group Chaos research reveals that agile project success rates are threefold greater than those of waterfall projects, while the failure rates of waterfall initiatives are double those of agile programs [6]. The Agile concept is anchored in the Agile Manifesto, which articulates four core values and twelve principles that direct Agile project management.

Agile approaches, including Scrum, Kanban, and Extreme Programming, have gained widespread adoption owing to their adaptability and emphasis on producing concrete outcomes in brief cycles, referred to as sprints. Scrum is defined by roles such as the Scrum Master and Product Owner, together with methods like daily stand-up meetings and sprint reviews, which promote continual improvement and flexibility [7].

II. PROBLEM STATEMENT

Although Agile techniques enjoy extensive acceptance, they are not devoid of obstacles. The transfer from traditional to Agile methodologies can be intricate, necessitating a culture transformation inside businesses and a reconfiguration of roles and processes. The scalability of Agile methodologies in extensive, distributed teams continues to be a topic of active research and discussion [8].

Software development projects frequently fail, resulting in elevated cancelation rates. A 2012 study by Dr. Dobbs found that Agile techniques achieved a 72% success rate, whilst traditional methodologies attained a 64% success rate. An 8% improvement is marginal and hardly constitutes a revolution. In the current competitive business landscape, we must enhance our success rate [3]. Research conducted by McKinsey revealed that 50% of IT projects with costs exceeding $15 million are 45% over budget and provide 56% less functionality than expected. In simple terms, Agile is not a panacea. Projects continue to collapse at a rate comparable to that of 2001. It seems that few changes or developments have occurred in this regard [3]. Furthermore, [9] agile study results indicated that, with ASD techniques, what is effective for one team may not be applicable to another.

A fundamental issue in software development is the challenge clients face in articulating their requirements. Customers encounter difficulties in articulating their requirements, as noted in [10]. Numerous IT project failures are caused by unrecognized and unmanaged risks, according to past research [11]. Furthermore, [12] contended that certain software initiatives fail because of the PMO or project managers' insufficient oversight of scheduling, cost, and scope variables.

Although recent studies on agile approaches have demonstrated encouraging outcomes, their practical implications remain unverified [13]. The majority of empirical research in agile development have been on evaluating the advantages of agile approaches or frameworks, such as Scrum, Kanban or Extreme Programming (XP) [14]. Few empirical literatures have examined the elements influencing the success

of IT projects, utilizing agile methodology [1], [14]. This study contributes to the academic literature that will aid agile practitioners in software development decision-making. According to [15], there is a scarcity of literature to aid software development managers in making effective project decisions. This work contributes to the existing knowledge base, offering supplementary insights for both researchers and software project managers.

III. RESEARCH OBJECTIVES

1. To examine how effective the Agile methodology is on the efficiency of information technology project management
2. To analyze the impact of Scrum in response to rapidly changing project requirements in IT environments.
3. To analyze the impact of Waterfall and Agile methodologies on project outcomes using the Iron Triangle framework.

IV. RESEARCH QUESTIONS

1. How effective are agile methodologies in improving the efficiency of IT project management?
2. What is the impact of Scrum on responding to rapidly changing project requirements in IT environments?
3. How do Waterfall and Agile methodologies impact project success when analyzed through the Iron Triangle framework?

V. SIGNIFICANCE OF THE STUDY

This unique study analyses the impact of agile methodologies on project success, specifically in the IT industry. It serves as a comparison between the fundamental project management methodology and the agile project management strategy. The study holds substantial importance for academic studies and real-world use in project management. The study's specific findings are anticipated to significantly enhance the understanding of techniques and provide guidance to project managers on the deployment of Agile, particularly the Scrum framework. It also contributes to academic publications, augmenting known body of knowledge on Agile in the IT industry.

VI. LITERATURE REVIEW

A. Overview of Agile Methodology

The evolution of Agile methodology in project management represents a significant shift from traditional, plan-oriented approaches to a more flexible, iterative framework. Agile methodologies emerged in the early 2000s primarily as a response to the limitations of the Waterfall model in software development. The Waterfall model, defined by its linear and sequential methodology, frequently resulted in considerable difficulties in accommodating evolving requirements and consumer demands [16]. Agile approaches were developed to tackle these difficulties, prioritizing adaptability, customer collaboration, and iterative advancement. Agile methodologies have revolutionised project management by introducing a flexible and iterative framework applicable across all industries. These methodologies, derived from software development, are characterised by their emphasis on adaptability, collaboration, client satisfaction, and incremental progress. Scrum and XP are among the first and most esteemed agile approaches [17]. Scrum seeks to offer an agile approach for managing software projects, enhancing the likelihood of successful software development, while XP emphasizes the project-level tasks involved in software implementation. Both methodologies, however, encapsulate the fundamental tenets of agile software development [5].

Agile includes a wide range of software development philosophies. It is a conceptual structure for the development of software that begins with an initial planning phase and advances to the deployment phase, integrating iterative and incremental interactions throughout the project's lifecycle. The principal aim of agile techniques is to reduce overhead in software development while enabling adaptation to changes without compromising workflow or incurring substantial rework.

The formal initiation of agile techniques may be traced to the release of the Agile Manifesto in 2001. [5], these ideals and principles establish the foundation for guiding the software development process and impart

distinctive qualities to any technique with agility. The Manifesto asserts that in any decision-making scenario, precedence is afforded to the elements on the left side of each fundamental value above those on the right.

These four values are as follows:

1. Individuals and interactions over processes and tools: The manifesto's initial claim posits that emphasising abstract formal processes and their technical background in software development is erroneous; instead, the critical components are communication, interaction, and the expertise of the human software engineers that these elements facilitate [5].

2. Working software over comprehensive documentation: Agile software development processes must have thorough documentation; nevertheless, time and resources must be carefully managed and optimised to ensure that documentation does not impede software development process. The usefulness of extensive documentation is a conventional limitation in development, as the process of writing and maintaining synchronization with the code is time-consuming, particularly when needs change frequently [18]. Furthermore, these documents are infrequently utilized beyond the first implementation stage. Consequently, the pertinent documentation in agile is that which supports the functioning software and enhances the process [5]. The manifesto asserts that genuine advancement in agile methodologies is evaluated by functional software that has been tested, rather than by documentation, as it is less ambiguous and may promptly determine whether requirements are met [19].

3. Customer collaboration over contract negotiation: Agile software development was established to accommodate changes in requirements at any phase; thus, regular client input, negotiation, and engagement with the development team are crucial for effectively fulfilling customer needs, rather than depending exclusively on formal agreements and contracts. Nonetheless, contracts defining the relationship between the development team and clients, along with specifying the software enterprise, remain essential [5].

4. Responding to change over following a plan: Throughout the software development process, both the developer and the clients will gain improved understanding of the system, requiring the modification of specific criteria. The Agile manifesto emphasises adaptability to changes in the development process rather than strict compliance with a predetermined plan, with the ultimate goal of delivering value to the client.

In addition to the four values, the Agile Manifesto specifies twelve guiding principles for methodologies rooted in agile development, the principles include [17]:

1. “Our Highest Priority is to satisfy the Customer through Early and Continuous Delivery of Valuable Software.”
2. “Welcome Changing Requirements, even late in Development. Agile Processes harness change for the Customer’s Competitive Advantage. The corresponding implication of the impact of any embracing any change in requirement must be minimized through practices like ‘refactoring’ that will be discussed in the next sections.”
3. “Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter time scale. This principle put some restriction on the first principle, it recommends the early and often delivery to satisfy the customer needs.”
4. Business people and Developers Must Work Together Daily throughout the Project. These frequent integrations aim to provide feedback and to answer the questions of the development team.”
5. “Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done. The team members are the most vital factor in success, the agile methods trust these individuals the power to make the appropriate decisions to do their job, for example, and they could change the process steps if they thought it will be an obstacle to their team.”
6. “The Most Efficient and Effective Method of Conveying Information to and within a Development Team is face-to-face Communications. This principle emphasizes the direct human communication in the agile team

rather than using written specification, or written plans."

7. "Working Software is the Primary Measure of Progress. Breaking down the product into smaller pieces and the early and often delivery is a better progress measurement method that gives a more honest impact through the running codes."
8. "Agile Processes promote sustainable development: The sponsors, developers, and users should be able to maintain a constant pace indefinitely. Sustainable development means that the team should maintain a constant rhythm, i.e., the ultimate error free deliveries should not be the ultimate goal that must be met early by consuming all the available resources, instead the availability of delivering a high-quality piece and get the correct feedback is better until the final goal is reached eventually."
9. "Continuous Attention to Technical Excellence and Good Design enhances Agility. The highest quality code delivery is a must that should be always achieved to be agile, even if they must refactor the code as a response to any change in requirement."
10. "Simplicity – the art of maximizing the amount of work not done – is essential. The aim is to produce a product that is simple and capable of handling the sudden changes and at the same time fulfill the customer requirements."
11. "The Best Architectures, Requirements, and Designs emerge from Self-Organizing Teams. Any agile team is a self-organized team that shares the responsibilities of a project and determine the best way to handle them through the knowledge they gain throughout the development process to achieve the optimal structure."
12. "At regular Intervals, the Team reflects on how to become more effective, then tunes and adjusts its behavior accordingly."

B. Benefits of Agile methodology

Rapid delivery of software products

Agile software development is a cooperative and iterative process [20]. Agile approaches are increasingly used in the software development industry. [21] In their research publications, they claim that agile methodologies offer an iterative, evolutionary, and incremental framework for software development. The complete application is deployed in incremental segments referred to as iterations. The time of each cycle is concise (few weeks), fixed, and strictly adhered to. Each iteration represents a little enhancement in functionality, constructed upon the preceding iteration. Agile software development, characterized by brief iterative cycles, facilitates swift, transparent, and inspiring enhancements in the software process [5].

Highly tolerant of change requirements

The main difference between heavyweight and agile approaches can be seen is how flexible they are in accepting changes. This ability to frequently adapt to changing requirements determines the success or failure of projects in the IT sector [22]. Rigorous methodologies inhibit product functionality and prevent modification. Agile systems development methodologies arose as a reaction to the inadequacy of earlier plan-driven strategies in managing swiftly evolving settings [6]. Rapid and adaptable responses to change are made possible through agile software development's emphasis on adaptive planning, evolutionary development, and delivery [23].

Customer feedback and active engagement

Customers are more actively involved and hold greater significance in agile techniques than in traditional methodologies. In the agile approach, face-to-face communication and ongoing input from the customer (product owner) consistently occur. Client's value active involvement in projects since it enables them to exert control over the project and enhances the transparency of the development process, while also keeping them informed [24]. The engagement of customers alleviates a prevalent issue in software projects: the discrepancy between their initial expectations and their final acceptance criteria. This contact enables the customer to have a clearer understanding of the evolving product. By visualizing the forthcoming feature based on previous constructions, clients cultivate a deeper comprehension of their own requirements and acquire the terminology to articulate them to the developers [25].

Self-organized team

Teams working within an Agile environment are autonomous, with responsibilities and interactions evolving as necessary to accomplish project objectives [26]. In an agile project, the team is generally cross-functional and self-organising, ignoring any established corporate hierarchy or the roles of people within the organisation [27]. Agile product development methodologies induce changes in team culture to promote mutual loyalty and commitment to the team and its projects. Team members frequently undertake responsibilities for tasks that deliver the essential functionality for an iteration. They independently ascertain how to meet the criteria of each iteration. Teams jointly create applications in a cooperative atmosphere. The Agile Alliance says that for a specific problem size, fewer individuals are required when employing a lighter methodology, but a heavier methodology necessitates more individuals. Furthermore, it contends that there exists a limit to the size of the problem that can be addressed with a fixed number of individuals [26].

Reduce cost and time

The research report by [28] states that the development team deemed the waterfall model an unpleasant experience, but XP was regarded as advantageous and a positive decision by management. The extreme programming project was completed with a 50% time overrun, compared to 60% for the regular project, and incurred a substantially lower cost overrun of 25%, in contrast to 50% for the traditional project. Agile development decreases development costs by decreasing expenditures related to rework, leadership, documentation, and other non-development tasks. Agile development approaches, including XP, Scrum, and ASD, offer enhanced client satisfaction, reduced defect rates, accelerated development timelines, and adaptability to swiftly evolving needs. Plan-driven methodologies, like Cleanroom, the Personal Software Process, and those derived from the Capability Maturity Model, offer predictability, stability, and a high level of assurance [29].

Scrum

Scrum is defined as a framework that encompasses functions executed by team members, documentation artifacts, delivery standards, and events that facilitate vital communication and product delivery [30]. The author in [31] asserts that Scrum is an iterative and incremental project management framework. It is a flexible and complete development strategy, wherein a team collaborates as a unit to attain a shared objective, hence challenging conventional methodologies in product development. Scrum is an agile project management methodology developed by Schwaber and Sutherland in 1990. It is appropriate for organizations engaged in product development within contexts marked by unpredictability, self-organization, moderate oversight, and knowledge transfer [31].

Scrum, renowned for its efficacy in overseeing intricate projects, functions through brief, time-constrained intervals referred to as sprints. In 1986, Hirotaka Takeuchi and Ikujiro Nonaka defined it as "a flexible, holistic product development strategy where a development team collaborates as a unit to achieve a common objective" in the New Product Development Game. This strategy contests the presuppositions of the "conventional, linear methodology" in product development and promotes self-organization among teams by fostering both online collaboration and direct interpersonal connection among all team members and disciplines involved in the project. The framework includes the responsibilities of the Scrum Master, Product Owner, and Development Team, all essential to the project's progress. The Scrum Master facilitates scrum events and supports the team throughout the entire process, the Product Owner represents stakeholders' interests, and take ownership of the product backlog with an ordered list of requirements, and the Development Team focusses on delivering product increments. The framework's flexibility renders it appropriate for projects with changing requirements [32]. The versatility of Scrum renders it especially efficient in projects with changing requirements, facilitating swift adjustments and iterative development [33].

In Scrum, the product is developed through a sequence of fixed-duration iterations known as sprints. Sprints are predetermined time intervals in which the product is developed and presented for evaluation. A Sprint is a development iteration lasting one month or fewer, maintaining a uniform duration throughout the project. It emphasizes providing maximum value in the minimal time frame. It is regarded as one of the most utilized agile methodologies [34]. The inclination to utilise it arises from its simplicity and focus on software administration issues rather than technical software development procedures, making it widely adaptable across several sectors.

The Scrum events

Scrum events transpire during a sprint, which constitutes the essence of scrum, converting concepts into value. A sprint is the fundamental unit of Scrum, including a small team that cooperates on assigned tasks. It persists for a period of 1 to 3 weeks. The tasks of a sprint are defined by the sprint backlog. The sprint backlog is a comprehensive list of all requirements to be fulfilled in the current sprint. The product backlog is a collection of requirements defined by the product owner, known as user stories. The process is segmented into sprint backlogs, followed by sprint planning, which includes plans for executing a sprint.

The Scrum Guide delineates four primary Scrum events; however, in practice, there are five. This research will examine the five scrum events in practice.

Sprint Planning - During sprint planning, tasks and their corresponding weights are selected from the product backlog items. The Sprint backlog is executed, wherein tasks are allocated throughout the entire team, and specific objectives are established for each cycle.

Daily Stand Up - The Daily Scrum seeks to assess advancement towards the Sprint Goal and adjust the Sprint Backlog accordingly, hence modifying the upcoming planned tasks. The Daily Scrum is a 15-minute assembly for the Developers of the Scrum Team. To reduce complications, it is held concurrently at the same time and location on each business day of the Sprint. Should the Product Owner or Scrum Master actively engage with items in the Sprint Backlog, they assume the role of Developers. Developers may select any structure and methodologies they prefer, as long as their Daily Scrum highlights advancement towards the Sprint Goal and produces a feasible plan for the following day's tasks. This improves focus and increases self-control.

Sprint Review - The Sprint Review meeting is a time-constrained session lasting two to four hours for one-month Sprints. The event transpires at the end of a Sprint, wherein the Team presents the finalised functionality to the Product Owner and other stakeholders. Facilitated by the Scrum Master. The Team reviews and discusses the work completed throughout the Sprint. The product owner delivers explicit and forthright feedback. Contributions for the upcoming sprint.

Backlog refinement - Product Backlog refining entails deconstructing and elaborating on Product Backlog items into smaller, more specific components. This is a continuous process to incorporate details, including a description, sequence, and dimensions. Attributes frequently differ according on the field of work. The developers assigned to the task are accountable for the sizing. The Product Owner might impact the Developers by assisting them in comprehending and choosing trade-offs.

Sprint Retrospective - The objective of the Sprint Retrospective is to devise strategies for enhancing effectiveness and quality. The sprint retrospective is conducted before the next sprint, involving the team to facilitate enhancements and optimize outcomes for the next phase [39, 40]. The Sprint Retrospective marks the conclusion of the Sprint. The duration is limited to a maximum of three hours for a one-month Sprint. For shorter Sprints, the event time is typically reduced.

SCRUM	OTHERS
Enhanced prescriptive and formal meeting frameworks, clearly delineated roles, and iterative processes	Kanban: Reduced prescriptiveness, absence of formal meetings, indefinite roles, and iterative processes
Productivity is prioritised above everything, resulting in enhanced customer	XP: Reduced flexibility and insufficient

satisfaction and increased flexibility.	prioritisation of production
Streamlined process, improved team communication	FDD: Reduced communication, increased procedural complexity
Enhanced collaboration and communication among scrum team	DSDM: Reduced communication among scrum team
Simple yet intricate procedures are followed.	ASD: Intricacy in procedural framework
Development and planning are defined by user needs, which improves traceability.	Crystal: Insufficient regard for user requirements, challenging to track completed job

C. Agile Software Development Lifecycle

The Agile Software Development Lifecycle (SDLC) is an iterative and incremental methodology that emphasizes adaptability, collaboration, customer satisfaction, and the swift delivery of functioning software. In contrast to conventional techniques such as the Waterfall model, Agile prioritizes ongoing input, flexibility, and interdisciplinary collaboration to meet the changing requirements of projects. Since its introduction, Agile has emerged as a predominant methodology in software development, mostly because to its capacity to adapt to change and its customer-focused ethos. This is a conceptual framework for software engineering that begins with an initial planning phase and advances to the deployment phase, integrating iterative and incremental interactions throughout the project's lifecycle. The principal aim of agile techniques is to reduce overhead in software development while enabling adaptation to changes without compromising the process or incurring needless rework.

Phase	Processes
Initiate	Formulate Project Vision Identify the Scrum Master and Stakeholders Establish Scrum team Formulate Epic(s)
	Establish a prioritised product backlog Execute release planning
Plan and Estimate	Formulate user narratives Authorise, assess, and allocate user narratives Create tasks Tasks estimation Formulate sprint backlog
Implement	Determine deliverables Conduct daily stand-up meetings. Groomed the prioritised product backlog
Review and Retrospect	Assemble a scrum of scrums Exhibit and authenticate sprint Project retrospective
Release	Deliverables for shipment project reflection

D. Agile Critical Success Measure

The agile success criteria were categorised into three distinct groups: process efficiency, sustainable software product quality, and stakeholder satisfaction.

Constructs	Agile Project Success Measures
Process Efficiency	Completed punctually, adhered to the allocated budget, and delivered the product within the specified scope.
Sustainable Software Product Quality	Software quality, security, usability, efficiency, compatibility, maintainability, and reliability in addition to portability and performance.
Stakeholder satisfaction	Customer satisfaction, employee contentment, senior stakeholders' satisfaction

E. Waterfall method

Royce first presented the waterfall model in 1970 [30]. It is the most prevalent among conventional approaches. The steps include requirements specification, design, implementation/development,

testing, and maintenance [35]. The IT project team allocates significant time to the requirements specification phase, which includes planning and design, to ensure that when the implementation phase begins, there are no additional requirements, uncertainties, or ambiguities concerning the project or its final result. Consequently, long-term projects are appropriate for the waterfall methodology [36], [37], [38], [39]

Methodologies in IT Project Management	Agile Methodology	Traditional Methodology
Methodology	They are employed when just initial requirements and objectives are defined, and when flexibility and adjustments during the project are expected, since they can conserve resources, time, and guarantee prompt product delivery.	They are utilised when project requirements are explicitly delineated, when the project cannot be subdivided into smaller elements, when the client prioritises the end delivery, and when team communication is informal.
Application	“Scrum (Mircea, 2019)”	“Waterfall (Mircea, 2019; [39])”

F. Iron Triangle

The "Iron Triangle" in project management, or the Triple Constraint, denotes the equilibrium that must be preserved among three essential factors: scope, time, and cost. As stated in [37], these three dimensions have conventionally been employed to assess project success. Atkinson stated that time and cost are frequently regarded as estimations during the project planning phase due to insufficient information, whereas quality is contingent upon stakeholders' opinions and perceptions, which may fluctuate during the project lifespan. Research indicates that projects with well-defined objectives are 3.302 times more likely to attain Iron Triangle goals compared to those lacking clarity. They emphasized that incorporating time and financial buffers markedly enhances the likelihood of a project achieving its scope, schedule, and budget objectives. Recent years have seen a growing acknowledgment that rigid adherence to the Iron Triangle may not always represent a project's overall success [34]. For example, [40] observed that projects which did not achieve the initial objectives of the Iron Triangle were not invariably seen as failures, but those that completely complied with these parameters could nevertheless be deemed unsuccessful from a commercial perspective. [34] found multiple critical success factors in project manufacturing environments that substantially influence the attainment of the Iron Triangle's objectives. These encompass team integration, the application of prior technology, and the documentation of lessons learnt. Projects that utilize prior technology are 6.905 times more likely to succeed within the parameters of the Iron Triangle.

The emergence of agile techniques has necessitated a reassessment of the conventional Iron Triangle in IT project management. Agile frameworks, especially Scrum, often "invert the Iron Triangle" by emphasizing adaptability in scope rather than strict compliance with schedule and budget limitations. According to [41], agile approaches facilitate the equilibrium of the Iron Triangle's demands by enabling teams to modify priorities in response to continuous feedback, hence ensuring the provision of customer value despite alterations in initial time and cost estimations.

G. IT Project Management

Information Technology Project management is essential for the proper execution of technology-related projects inside a company. IT project management entails the utilization of diverse strategies, tools, and techniques to guarantee the achievement of project objectives. As stated in [42], the efficacy of IT projects is frequently evaluated based on their adherence to schedule, budget, and scope limitations. Agile approaches, including Scrum, have gained importance as a remedy for the inflexible

frameworks of conventional IT project management. Agile approaches, through iterative development, ongoing stakeholder feedback, and adaptability to change, offer a more flexible framework for IT projects. Agile approaches empower IT teams to effectively manage the uncertainties and complexities inherent in contemporary projects, enhancing both project quality and overall business value.

Empiricism Control Theory

Empiricism is the philosophical doctrine asserting that knowledge is derived from experience and observation, rather than from intrinsic concepts or abstract reasoning. on Agile, particularly under frameworks like as Scrum, empiricism is essential since it directs teams to ground their judgments on observation, experience, and experimentation. This guarantees that agile procedures stay flexible and responsive to evolving circumstances instead of being constrained by rigid pre-established regulations. Empiricists contend that knowledge cannot be validated through sensory experience. For an empiricist, sensory experience or observation constitutes the definitive criterion of truth. Empiricism is sometimes conflated with logical positivism (e.g. [29, 30]). Logical positivism asserts that concepts are validated by indisputable facts recognised by objective observers. Anti-positivism and postmodernism fundamentally reject the existence of objective facts, asserting that subjective observers generate ideas from observations, which they interpret via the framework of pre-existing theories. The thesis of Empiricism posits that observation is the sole source of knowledge. Empiricism does not assert the presence of unequivocal facts, the partiality of observers, or the relationship between facts and hypotheses. The fundamental elements of empiricism in agile are transparency, inspection, and adaptation.

VII. RESEARCH METHOD

Research Approach

This research seeks to examine the application of Agile approaches in IT projects, utilizing an inductive methodology that offers the adaptability to comprehend and evaluate the subtleties of human experiences, team dynamics, and project outcomes in practical contexts. An inductive approach is consistent with the interpretivist mindset, enabling the research to create ideas derived from the themes and patterns detected in the organized literature survey. This technique is not constrained by pre-established hypotheses, which is essential for examining dynamic and developing methodologies like Agile, where outcomes are frequently influenced by contextual elements such as team collaboration, stakeholder relationships, and company culture.

Research design/strategies

Due to the nature of this research, a qualitative research methodology is the most appropriate technique. This approach facilitates a comprehensive examination of the impact of agile approaches on IT project success by concentrating on the experiences and viewpoints of the individuals and teams engaged in these projects. The qualitative approach facilitates a flexible analysis of the interrelations among diverse components, which can be modified or enhanced to optimize project results.

Sources of Data

Google Scholar has served as the principal research instrument for this dissertation regarding the influence of Agile approach on the success of IT projects. The search was broadened to encompass more recently published and highly cited publications within the same field. Principal sources of information comprise:

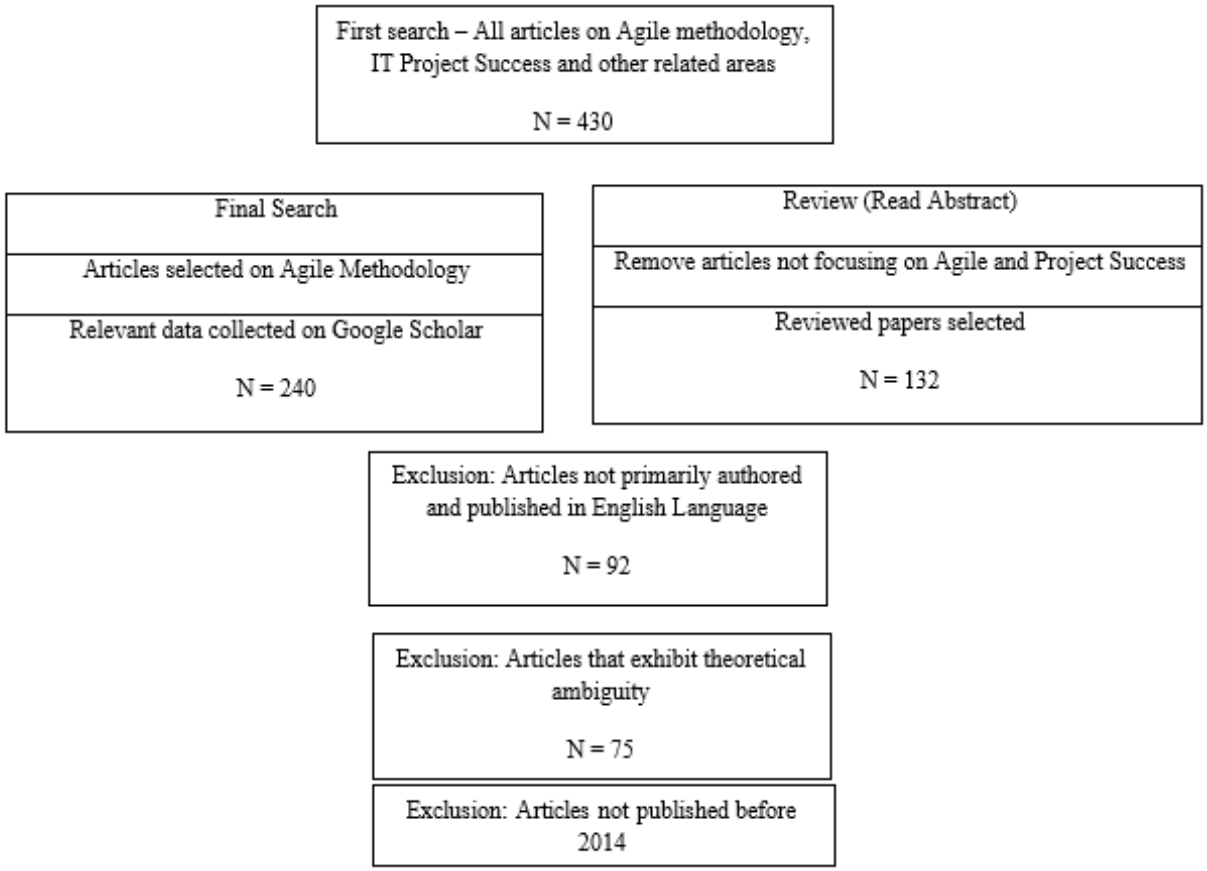
- Research studies on Agile and IT project management
- Journal of Software Project Management
- Journal of Agile Methodologies
- Research articles on [44] research onion focusing on interpretivism and qualitative research approaches

Data Analysis

The research employs a qualitative inductive technique, directing the data analysis procedures and customizing them to the particulars of the collected data. The outcomes will be examined through theme analysis, a methodical technique for data collecting and interpretation. This strategy will enable the discovery and examination of recurring themes in the collected journals, ensuring a systematic analysis of the study's results.

© SEP 2025 | IRE Journals | Volume 9 Issue 3 | ISSN: 2456-8880
DOI: doi.org/10.64388/IREV9I3-1710544-450

Systematic Literature Review Process



VIII. ANALYSIS AND FINDINGS

A. Systematic Literature Review (SLR)

SLR is a systematic and comprehensive methodology for collecting, assessing, and synthesising research studies relevant to a specific research topic. This methodology guarantees a comprehensive review, encompassing all pertinent literature on the topic in a clear and replicable fashion. SLRs are especially effective in delivering evidence-based insights by minimizing researcher bias and assuring the inclusion of only high-quality, peer-reviewed literature [45]. This systematic literature review seeks to assess the efficacy of Agile approaches, with particular emphasis on IT project success rates, the function of Scrum in handling changing requirements, and the juxtaposition of Agile and Waterfall methodologies. This study encompasses studies ranging from the inception of Agile, exemplified by [46], to contemporary analyses of its implementation in intricate IT projects [2].

B. Thematic Findings

Agile's Impact on Project Efficiency

A significant trend identified in the investigation is the beneficial effect of agile approaches on project efficiency. The iterative methodology of Agile, which entails dividing projects into smaller, manageable steps, facilitates ongoing feedback and modification. This iterative procedure guarantees that difficulties are resolved promptly, averting their escalation into substantial obstacles. According to [47], agile techniques are transforming project management by emphasizing adaptability and prioritizing the delivery of high-value features, hence enhancing productivity in IT projects.

The focus of Agile on client collaboration during the project lifecycle is an additional element that enhances

project efficiency. By engaging the customer throughout the development process, Agile teams may guarantee that the final product aligns with the customer's requirements, minimizing the need for extensive revisions. This ongoing contact enables teams to swiftly adjust to changes, minimizing the probability of delays or budget excesses [8]. This ongoing collaboration enables teams to swiftly adjust to changes, hence minimizing the risk of delays or budget excesses.

Recent studies by [48] indicate that agile projects are more likely to be completed on schedule and under budget than traditional techniques, owing to their adaptability. Moreover, agile's emphasis on waste reduction via iterative cycles guarantees optimal resource utilization, resulting in improved project outcomes. Furthermore, studies indicate that Agile projects attain elevated success rates owing to their iterative cycles, which permit teams to modify the project's scope and produce value incrementally. This strategy differs from the inflexible, phase-oriented approach of conventional techniques like as Waterfall, which frequently encounters difficulties with scope alterations, resulting in project delays or failures.

Scrum's Role in Managing Dynamic Requirements

The second theme emphasizes the function of Scrum in overseeing fluctuating project requirements. Scrum, a prominent Agile paradigm, is especially effective in settings where project requirements change over time. The iterative process, defined by brief development cycles termed sprints, enables teams to adapt to evolving needs without compromising the overall project timeframe [49], [50].

An important benefit of Scrum is its implementation of daily standup meetings and sprint reviews, which facilitate ongoing communication among team members and stakeholders. These tools promote transparency and provide prompt modifications to the project scope in response to client feedback [51], [49], [52]. [54] highlight that Scrum teams are more adept at managing changing priorities and unforeseen obstacles due to iterative reviews that allow for course correction.

Moreover, [55] highlight that the positions of Product Owner and Scrum Master are essential in overseeing evolving needs. The Product Owner prioritizes essential features for the team, while the Scrum Master assures the efficient execution of sprints by eliminating impediments to development.

The Iron Triangle and Agile Flexibility

The third theme examines the interaction between agile approaches, specifically Scrum, and the conventional limitations of the Iron Triangle (scope, time, money). The Iron Triangle, or Triple Constraint, is a project management concept that evaluates project success by three critical variables: scope, time, and cost. In conventional project management methodologies such as Waterfall, these restrictions are typically established at the project's inception. Agile approaches provide a more adaptable framework, enabling scope to develop while maintaining oversight of time and expenses [56].

In Agile, the emphasis transitions from strict compliance with a predetermined scope to providing value through ongoing iterations. This adaptability allows agile teams to meet evolving customer demands without compromising the project's budget or schedule. A recent study by [8] indicates that Agile projects are more prone to success under the Iron Triangle framework due to its focus on iterative delivery and scope control informed by real-time feedback.

The adaptability of Agile enables project managers to reassign resources or modify priorities as necessary, so guaranteeing that projects stay on course. As stated in [57], agile's iterative methodology enables teams to prioritize features and produce incremental outcomes, facilitating the management of time and cost limitations while accommodating evolving customer requirements. This adaptability is especially beneficial in IT projects, where the swift evolution of technology frequently requires modifications to project scope.

C. Discussion and Summation of the Research Findings

Agile's Contribution to Project Efficiency

This research reveals a substantial favorable effect of Agile techniques on project efficiency. Agile frameworks inherently prioritize iterative

development, enabling teams to produce functional components of a project frequently and promptly. This early delivery guarantees ongoing feedback and allows teams to detect possible issues earlier, thereby averting significant obstacles later in the project [58]. Agile emphasizes dividing projects into smaller, manageable increments, termed sprints, enabling teams to target high-value features and sustain momentum despite obstacles [2].

As opposed to traditional methods such as Waterfall, which can experience significant disruptions and project delays due to scope changes, Agile excels in adaptability. The literature endorses the notion that Agile's ongoing communication with stakeholders enables teams to implement real-time modifications, hence improving project efficiency and customer satisfaction [59]. Agile teams develop functional software early and consistently, so providing value at each iteration, regardless of the project's overall completion status [16].

Moreover, studies indicate that Agile's flexibility reduces the risks linked to scope creep, a prevalent challenge in conventional project management approaches [59]. The capacity to adapt to changes without much interruption enables Agile teams to exert greater control over project schedules and finances, thus enhancing the probability of timely and budget-compliant delivery [60].

Scrum's Role in Managing Dynamic Project Requirements

Scrum's iterative development cycles and continuous feedback mechanisms proved to be exceptionally successful in handling evolving project needs. This is especially significant in the IT sector, as project requirements frequently evolve due to the emergence of new technologies or shifts in client expectations [55]. The literature regularly indicates Scrum's capacity to manage such changes without incurring substantial delays or budget excesses [61].

Daily standup meetings and sprint reviews, integral to Scrum methodologies, foster a collaborative atmosphere in which teams can consistently reevaluate project priorities informed by real-time feedback [62]. These methods enable teams to detect and resolve issues promptly, ensuring that alterations in requirements do not impede the project [32]. The position of the Product Owner in Scrum also helps the team's capacity to manage dynamic requirements by ensuring that the team remains focused on high-priority items that fit with customer expectation [55].

The research substantiates that Scrum's framework and methodologies, including brief sprints and specified roles, render it exceptionally appropriate for IT projects where adaptability and responsiveness are essential. By promoting consistent feedback and permitting swift adjustments, Scrum empowers teams to accommodate changing requirements while preserving project momentum [59].

Agile's Interaction with the Iron Triangle

This research reveals that agile techniques provide flexibility within the limitations of the Iron Triangle, which assesses project performance based on scope, time, and cost. Conventional techniques like as Waterfall frequently see these limitations as immutable, resulting in considerable difficulties when project requirements evolve during execution. Agile, in contrast, permits flexibility in scope while enforcing rigorous oversight of time and money [63].

Agile frameworks, especially Scrum, prioritize delivering value through iterative cycles rather than strict compliance with a predetermined scope. This adaptability allows teams to modify the project scope in response to customer feedback or new requirements without compromising the overall project timeline or budget [7]. The iterative characteristic of Agile allows teams to frequently reevaluate priorities, guaranteeing that the most essential features are prioritized for delivery, even if the entire scope is not fulfilled [59].

Nevertheless, the literature emphasizes that although Agile provides considerable flexibility, sustaining control over time and cost necessitates disciplined resource management and efficient prioritization [64]. Agile's capacity to reconcile these constraints is especially advantageous in IT projects, where frequent scope alterations are prevalent. The research validates that Agile methodology enable scope evolution without compromising schedule and cost, offering a more dynamic and responsive project management approach than traditional methods [65].

CONCLUSION

The outcomes of this research provide clear proof of the major impact that agile techniques have on IT project success. Agile frameworks like Scrum not only increase project productivity but also offer better flexibility in managing dynamic requirements and overcoming the limits of the Iron Triangle. However, the success of agile projects rely largely on numerous essential success elements, including team autonomy, customer involvement, and a supportive company culture. These variables are critical to ensure that agile approaches are applied properly and produce the desired objectives.

As agile techniques increasingly proliferate across diverse industries, it is essential for organizations to comprehend both the technical dimensions of these frameworks and the cultural and organizational elements that facilitate their success. Organizations can utilize agile approaches to effectively manage the complexities of IT projects and attain successful outcomes by promoting a collaborative and adaptable environment.

REFERENCES

[1] Stankovic, D., Nikolic, V., Djordjevic, M., & Cao, D. B. (2013). A survey study of critical success factors in agile software projects in former Yugoslavia IT companies. *Journal of systems and software*, *86*(6), 1663-1678.

[2] Cervone, D. (2021). The KAPA model of personality structure and dynamics. In *The handbook of personality dynamics and processes* (pp. 601-620). Academic Press.

[3] Nguyen, D. S. (2016). Success factors for building and managing high performance agile software development teams. *International Journal of Computer*, *20*(1), 51-82.

[4] Cobb, P. J. (2023). Large Language Models and Generative AI, Oh My!: Archaeology in the Time of ChatGPT, Midjourney, and Beyond. *Advances in Archaeological Practice*, *11*(3), 363-369.

[5] Rodríguez, P., Mäntylä, M., Oivo, M., Lwakatare, L. E., Seppänen, P., & Kuvaja, P. (2019). Advances in using agile and lean processes for software development. In *Advances in computers* (Vol. 113, pp. 135-224). Elsevier.

[6] Litchmore, K. A. (2016). *A comparative study of agile methods, people factors, and process factors in relation to project success*. Capella University.

[7] Berbegal-Mirabent, J., Gil-Doménech, D., & Berbegal-Mirabent, N. (2017, June). Teaching agile methodologies in a project management course. In *Proceedings of the 3rd International Conference on Higher Education Advances* (pp. 312-320). Editorial Universitat Politècnica de València.

[8] Iqbal, J. (2021). The effects of agile methodologies on software project management in pakistani software companies. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, *12*(3), 1717-1727.

[9] Kropp, M., Meier, A., Anslow, C., & Biddle, R. (2020). Satisfaction and its correlates in agile software development. *Journal of Systems and Software*, *164*, 110544.

[10] Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. *Computers in human behavior*, *51*, 915-929.

[11] Paul, R., Turzo, A. K., & Bosu, A. (2021, May). Why security defects go unnoticed during code reviews? a case-control study of the chromium os project. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)* (pp. 1373-1385). IEEE.

[12] Kerzner, H. (2023). *PROJECT RECOVERY Case Studies and Techniques for Overcoming Project Failure*. wiley.

[13] Tripp, J. F. (2012). *The Impacts of Agile Development Methodology Use on Project Success: A Contingency View*. ProQuest LLC. 789 East Eisenhower Parkway, PO Box 1346, Ann Arbor, MI 48106.

[14] Lee, O. K. D. (2012). IT-enabled organizational transformations to achieve business agility. *The Review of Business Information Systems (Online)*, *16*(2), 43.

[15] Sheffield, J., & Lemétayer, J. (2013). Factors associated with the software development agility of successful projects. *International Journal of Project Management*, *31*(3), 459-472.

[16] Hummel, M. (2014, January). State-of-the-art: A systematic literature review on agile information systems development. In *2014 47th Hawaii International Conference on System Sciences* (pp. 4712-4721). IEEE.

[17] Fernández-Diego, M., Méndez, E. R., González-Ladrón-De-Guevara, F., Abrahão, S., & Insfran, E. (2020). An update on effort estimation in agile software development: A systematic literature review. *IEEE Access*, *8*, 166768-166800.

[18] Larson, D., & Chang, V. (2016). A review and future direction of agile, business intelligence, analytics and data science. *International Journal of Information Management*, *36*(5), 700-710.

[19] Gupta, S., & Gouttam, D. (2017, August). Towards changing the paradigm of software development in software industries: An emergence of agile software development. In *2017 IEEE International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)* (pp. 18-21). IEEE.

[20] Hoda, R., Salleh, N., & Grundy, J. (2018). The rise and evolution of agile software development. *IEEE software*, *35*(5), 58-63.

[21] Flora, H. K., & Chande, S. V. (2014). A systematic study on agile software development methodologies and practices. *International Journal of Computer Science and Information Technologies*, *5*(3), 3626-3637.

[22] Reiff, J., & Schlegel, D. (2022). Hybrid project management–a systematic literature review. *International journal of information systems and project management*, *10*(2), 45-63.

[23] Amajuoyi, P., Benjamin, L. B., & Adeus, K. B. (2024). Agile methodologies: Adapting product management to rapidly changing market conditions. *GSC Advanced Research and Reviews*, *19*(2), 249-267.

[24] Meckenstock, J. N. (2024). Shedding Light on the Dark Side–A Systematic Literature Review of the Issues in Agile Software Development Methodology Use. *Journal of Systems and Software*, 111966.

[25] Seitsamo-Räsänen, S. (2021). Building an Agile Approach to Individual Feedback.

[26] Leffingwell, D. (2010). *Agile software requirements: lean requirements practices for teams, programs, and the enterprise*. Addison-Wesley Professional.

[27] Reinotas, P. (2024). *Self-organising practices in project management* (Doctoral dissertation, Vilniaus universitetas.).

[28] Neto, F. S. (2016). *Impact of agile practices on organization learning: a model for knowledge creating and sharing in agile teams*.

[29] Turner, R., Abba, W., Elliott, M. R., Tolentino, G., Tian, J., Stracener, J., ... & O'Neill, D. (2017). Modern Process Trends. *CrossTalk*.

[30] Schwaber, K. (1997). Scrum development process. In *Business Object Design and Implementation: OOPSLA'95 Workshop Proceedings 16 October 1995, Austin, Texas* (pp. 117-134). Springer London.

[31] Esteki, M., Gandomani, T. J., & Farsani, H. K. (2020). A risk management framework for distributed scrum using PRINCE2 methodology. *Bulletin of Electrical Engineering and Informatics*, *9*(3), 1299-1310.

[32] Bogacheva, A. (2023). *Innovative dynamics of development of the modern organization management process and innovative approaches in management* (Doctoral dissertation, Sumy State University).

[33] Kalyani, D., & Mehta, D. (2019). Study of agile scrum and A likeness of scrum tools. *International Journal of Computer Applications*, *178*(43), 21-28.

[34] Daraojimba, E. C., Nwasike, C. N., Adegbite, A. O., Ezeigweneme, C. A., & Gidiagba, J. O. (2024). Comprehensive review of agile methodologies in project management. *Computer Science & IT Research Journal*, *5*(1), 190-218.

[35] Thummadi, B. V., & Lyytinen, K. (2020). How much method-in-use matters? A case study of agile and waterfall software projects and their

design routine variation. *Journal of the Association for Information Systems*, *21*(4), 7.

[36] McCormick, M. (2012). Waterfall vs. Agile methodology. *MPCS, N/A*, *3*, 18-19.

[37] Vresk, A., Pihir, I., & Furjan, M. T. (2020). Agile vs. Traditional Methods for Managing IT Projects-A Case Study. In *Central European Conference on Information and Intelligent Systems* (pp. 183-191). Faculty of Organization and Informatics Varazdin.

[38] Bhavsar, K., Shah, V., & Gopalan, S. (2020). Scrum: An agile process reengineering in software engineering. *International Journal of Innovative Technology and Exploring Engineering*, *9*(3), 840-848.

[39] Kassawat, M. (2024). Agile working and job satisfaction for localization language agents. *Translation Spaces*.

[40] Mircea, E. (2019). Project management using agile frameworks. *Academy of Economic Studies. Economy Informatics*, *19*(1), 34-44.

[41] Sljivar, I., & Gunasekaran, A. (2018, April). Agile-Scrum for facility design project management. In *SPE Western Regional Meeting* (p. D041S007R007). SPE.

[42] Peslak, A. R. (2012). Information technology project management and project success. *International Journal of Information Technology Project Management (IJITPM)*, *3*(3), 31-44.

[43] Rajan, E. R., & Santhosh, V. A. (2021). Adoption of Agile Methodology for iMproving it project perforMAnce. *Serbian Journal of Management*, *16*(2).

[44] Saunders, L. J., Zhu, H., Bunce, C. V., Doré, C. J., Freemantle, N., Crabb, D. P., & Ophthalmic Statistics Group. *(2015)* (Vol. 24, No. 76, pp. 1168-1170). Ophthalmic statistics note 5: diagnostic tests-sensitivity and specificity. British Journal of Ophthalmology, 99.

[45] Kitchenham, B., Budgen, D., Brereton, P., Turner, M., Charters, S., & Linkman, S. (2007). Large-scale software engineering questions–expert opinion or empirical evidence?. *IET software*, *1*(5), 161-171.

[46] Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., & Thomas, D. (2001). Manifesto for agile software development.

[47] Serrador, P., & Pinto, J. K. (2015). Does Agile work?—A quantitative analysis of agile project success. *International journal of project management*, *33*(5), 1040-1051.

[48] Annamalah, S., Paraman, P., Ahmed, S., Pertheban, T. R., Marimuthu, A., Venkatachalam, K. R., & Ramayah, T. (2023). Exploitation, exploration and ambidextrous strategies of SMES in accelerating organisational effectiveness. *Journal of Global Operations and Strategic Sourcing*, (ahead-of-print).

[49] Ahmed, Z., Mansor, Z., & Ahmad, K. (2017). An analysis of knowledge management challenges in agile global software development. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, *9*(3-4), 63-66.

[50] Arias Gonzáles, J. L., Flores Turpo, G. A., Paricoto, I. M., Molina, S. U., Maldonado Farfan, A. R., Velásquez Velásquez, W. L., ... & Chuyma, R. C. (2022). Building Environment for a Good Business: The Integration of Scrum Project Management Method to find and develop innovative Business Solutions in Peru. *Turkish Online Journal of Qualitative Inquiry*, *13*(1).

[51] Al-Baik, O., & Miller, J. (2015). The kanban approach, between agility and leanness: a systematic review. *Empirical Software Engineering*, *20*, 1861-1897.

[52] Shafiq, S., & Inayat, I. (2017, September). Towards studying the communication patterns of Kanban teams: A research design. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)* (pp. 303-306). IEEE.

[53] Kalyani, D., & Mehta, D. (2019). Study of agile scrum and Alikeness of scrum tools. *International Journal of Computer Applications*, *178*(43), 21-28.

[54] Schwaber, K., & Sutherland, J. (2020). Panduan Scrum. *Diakses tanggal*, *2*.

[55] Munteanu, V. P., & Dragos, P. (2021). The case for agile methodologies against traditional ones in financial software projects. *European Journal*

of Business and Management Research, *6*(1), 134-141.

[56] Stellman, A., & Greene, J. (2017). *Head First Agile: A Brain-Friendly Guide to Agile Principles, Ideas, and Real-World Practices*. " O'Reilly Media, Inc.".

[57] da Silva Estácio, B. J., Prikladnicki, R., & Grechenig, T. (2018). Systematic literature review on agile practices in global software development. *Information and Software Technology*, *96*, 161-180.

[58] Sudhakar, G. P. (2012). A model of critical success factors for software projects. *Journal of Enterprise Information Management*, *25*(6), 537-558.

[59] Davis, K., & Pinto, J. K. (2023). Corporate innovation and agile project management. In *Handbook on Innovation and Project Management* (pp. 375-392). Edward Elgar Publishing.

[60] Ramesh, B., Cao, L., Kim, J., Mohan, K., & James, T. L. (2017). Conflicts and complements between eastern cultures and agile methods: an empirical investigation. *European Journal of Information Systems*, *26*(2), 206-235.

[61] Hossain, E., Ali Babar, M., & Verner, J. (2009). Towards a framework for using agile approaches in global software development. In *Product-Focused Software Process Improvement: 10th International Conference, PROFES 2009, Oulu, Finland, June 15-17, 2009. Proceedings 10* (pp. 126-140). Springer Berlin Heidelberg.

[62] Maruping, L. M., Venkatesh, V., & Agarwal, R. (2009). A control theory perspective on agile methodology uses and changing user requirements. *Information systems research*, *20*(3), 377-399.

[63] C., & Da Silva, V. R. (2023). 20 Years of the Agile Manifesto: A Literature Review on Agile Project Management. *Management and Production Engineering Review*, *14*.

[64] Pikkarainen, M., Huhtala, T., Kemppainen, L., & Häikiö, J. (2019). Success factors for data–driven service delivery networks. *Journal of Innovation Management*, *7*(4), 14-46.

[65] Sljivar, I., & Gunasekaran, A. (2018, April). Agile-Scrum for facility design project management. In *SPE Western Regional Meeting* (p. D041S007R007). SPE.