Bachelor of Science in Computer Science & Engineering

Analyzing the Impact of Agile Methodologies on Software Development Cycle Time

by

Sanjana Jahan Sifat

ID: 1904109

Prithula Saha Pritha

ID: 1904110

Nabiha Fiza

ID: 1904114

Department of Computer Science & Engineering

Chittagong University of Engineering & Technology (CUET)

Chattogram-4349, Bangladesh.

December, 2023

Chittagong University of Engineering Technology (CUET)

Department of Computer Science Engineering

Chattogram-4349, Bangladesh.

Research Proposal

Application for the Approval of B.Sc. Engineering Research

Student Name	: Sanjana Jahan Sifat
	: Prithula Saha Pritha
	: Nabiha Fiza
Id	: 1904109
	: 1904110
	: 1904114
Session	: 2021-2022
Course Teacher Name	: Dr. Mohammad Shamsul Arefin
Designation	: Professor
	Department of Computer Science and Engineering
Course Teacher Name	: Mir. Md. Saki Kowsar
Designation	: Assistant Professor
	Department of Computer Science and Engineering
Tentative Title	: **Analyzing the Impact of Agile Methodologies on Software Development Cycle Time**

Table of Contents

List of Figures

1 Introduction

Agile Methodologies encompass software development approaches rooted in iterative and incremental progress. Four core characteristics define all agile methodologies: adaptive planning, iterative and evolutionary development, swift and adaptable responses to change, and a focus on fostering communication [1, 2]. The primary emphasis of Agile Methodologies lies in adhering to the principles of being ”Light but sufficient,” prioritizing a people-oriented approach, and fostering communication at its core. Its designation as a lightweight process makes it particularly well-suited for the development of smaller projects [3]. This methodology promotes streamlined processes that aim to deliver sufficient solutions without unnecessary complexity, focusing on human interactions and effective communication to achieve project goals. Its agile nature allows for flexibility and adaptability, which proves advantageous in smaller project settings where quick iterations and responsiveness to changes are crucial for success. In Agile software development, the approach advocates for production teams to commence their work by outlining basic and foreseeable approximations of the final requirements. As the development progresses, these initial requirements are continuously refined and expanded upon, incorporating incremental enhancements throughout the project's lifecycle. This ongoing process of refining requirements serves to continually fine-tune and improve the design, coding, and testing phases at each stage of production. Consequently, the resulting requirements work product becomes highly accurate and valuable, mirroring the precision and utility of the final software product itself [4]. This iterative approach allows for an adaptable and responsive development cycle, ensuring that evolving needs and insights are integrated effectively into the software's development process. The core principle of agile software development, as articulated in [5], advocates for periodic team reflection aimed at enhancing effectiveness, followed by adjustments in behavior for improvement. Alternatively, this principle embodies how the agile methodology aptly confronts the difficulties presented by unpredictable and disorderly business and technology landscapes [6]. Agile methodologies serve as a means to attain superior quality software within condensed timelines, fostering self-organizing teams, active customer collaboration, minimized documentation, and accelerated time-to-market [7, 8].

The Agile methodology constitutes a diverse set of lightweight frameworks and approaches, comprising methodologies such as Scrum, Crystal Clear, Extreme Programming (XP), Adaptive Software Development (ASD), Feature Driven Development (FDD), Dynamic Systems Development Method (DSDM), Crystal, Lean Software Development, and several others [9]. These methodologies under the Agile umbrella encompass a range of principles, practices, and frameworks that share common values but often differ in their specific approaches to software development. Each of these methods emphasizes iterative progress, flexibility, collaboration, and adaptability, catering to various project needs and team dynamics. They offer distinct strategies for managing tasks, organizing teams, handling changes, and delivering high-quality software products within dynamic and evolving environments.

2 Motivation

The research aims to investigate and promote the widespread acceptance of Agile methodologies across industries. Agile emphasizes collaboration, adaptability, and iterative development, offering a departure from traditional, rigid approaches.The research aims to investigate and promote the widespread acceptance of Agile methodologies across industries. Agile emphasizes collaboration, adaptability, and iterative development, offering a departure from traditional, rigid approaches.The research prioritizes understanding and incorporating the voice of the customer in the development process. It aims to ensure that products align closely with customer needs, preferences, and expectations, ultimately enhancing user satisfaction and product success.This motivation centers on enhancing the efficiency of development processes and optimizing resource utilization. The goal is to identify strategies and practices that reduce waste, streamline workflows, and maximize the value derived from available resources.The research focuses on identifying and overcoming challenges associated with traditional development models. It may explore issues like long development cycles, inflexible planning, and difficulties adapting to changing requirements, seeking innovative solutions to improve overall project outcomes.This motivation involves an exploration of the strategic considerations and decision-making processes involved in adopting new development methodologies. The research may examine the factors that influence organizations to transition to Agile or other innovative practices, providing insights for informed decision-making.The research aims to translate theoretical findings into actionable insights for project management. It seeks to provide practical guidance and recommendations that project managers can implement to enhance project success, team collaboration, and overall project management effectiveness. This includes addressing real-world challenges and offering applicable solutions in the context of project management.

3 Background and Present State

Backdoor Agile Software Development is presently an emerging discipline in the field of Software Engineering. It is presently advocated by many software professionals. The Agile software development principles that are followed and advocated emerged from the traditional software development principles and various experiences based on the successes and failures in software projects. According to [9], customers found it difficult to define their needs because of the fast changing technology and the companies using them in products. New methods, now called agile methods are designed to define the changing requirements in software environments. Agile methods focus on the challenges of unpredictability of the real world by relying on people and their creativity rather than processes [10]. The main theme in agile methods is to promote and speed up responses to changing environments, requirements and meeting the deadlines.

There are a number of agile software development methods. Methods for agile software development represent a set of practices for software development that have been created by experienced people [11]. Methods for agile software development represent a set of practices for software development that have been created by experienced people [12].

These methods can be seen as a new idea to plan-based or traditional methods, which emphasize a rationalized, engineering-based approach" [13] in which it is particular that problems are fully specifiable and that best and anticipated solutions exist for every problem. The "traditionalists" are said to advocate extensive planning, codified processes, and rigorous reuse to make development an efficient and predictable activity [13]. By contrast, agile processes address the challenge of an unpredictable world by relying on people and their creativity rather than on processes" [14]. Comparison of traditional and agile development is explained in [15]. Many people have tried to explain the core ideas in agile software development and some of them by examining similar trends in other disciplines. Some of the core ideas in this system were to eliminate waste, achieve quality first time, and focus on problem solving.

The agile ideas have been in Complex Adaptive System by providing a theoretical lens for better understanding and how the agile development is made in volatile business environments by Meso and Jain.Turk et al. [16] have provided comprehensive clarification regarding the underlying assumptions driving the execution of agile development processes. Additionally, they have meticulously identified potential limitations that could emerge from these assumptions. Simultaneously, Nerur and Balijepally's research [17] focused on cultivating and refining design concepts within the domains of architectural design and strategic management. Their work aimed to elevate the maturity level of design ideas, contributing significantly to advancing the understanding and practical application of design principles in these strategic domains.

A definitive conclusion drawn from the literature review on published works regarding agile methods is the imperative need to enhance both the quantity and quality of software development. Specifically, agile project management methodologies, notably Scrum, extensively adopted in the industry, emerge as focal deeper investigation and scrutiny.

The Agile Manifesto serves as the foundational framework for agile software development. In 2001, a group of individuals integral to the manifesto's creation established the Agile Alliance, a nonprofit organization. This collective effort aimed to further articulate and elaborate on the ideologies encapsulated within their manifesto, culminating in the formulation of a set of twelve guiding principles. These principles provide a more detailed and nuanced interpretation of the core values and beliefs embedded in the Agile Manifesto, offering a comprehensive framework for implementing agile methodologies in software development practices.

Based on a study, Pirjo et al [18] shows that agile methods are good for some programming environments, but not for all. Projects that involve large teams, well-defined requirements, clients needing high assurance and large code-bases, the traditional plan-oriented project profile works well. Therefore, Agile methods produces best results in case of when the team is small, the requirements are not yet well defined, the project code base is small and the customer is interested in seeing significant progress. However, as a software project transitions from a small prototype to a large stable system with a large team, with promises to keep and dates to meet, then agile methods alone is not sufficient then some additional mechanism is needed.

4 Research Questions

Creating well-crafted research questions is essential for conducting a thorough investigation into the impact of Agile methodologies on software development cycle time. Below are some research questions that can guide your study:

- How does the adoption of Agile methodologies influence the overall software development cycle time compared to traditional development approaches?
- What are the key factors contributing to the effectiveness or ineffectiveness of Agile methodologies in reducing development cycle time?
- How does the use of specific Agile frameworks (e.g., Scrum, Kanban) impact software development cycle time, and are there significant variations among these frameworks?
- Are certain Agile practices more influential than others in minimizing development cycle time, and if so, which ones?
- In what ways does Agile outperform or fall short in comparison to other development methodologies concerning cycle time?

- To what extent does team collaboration, communication, and autonomy impact the efficiency of Agile practices in software development?
- In what ways does Agile's emphasis on adaptability to changing requirements impact the ability to reduce development cycle time?
- How does the integration of customer feedback throughout the development process impact the overall efficiency of Agile practices?
- Are there trade-offs between speed and software quality in Agile development, and how do these impact cycle time?
- How do different organizational structures and policies impact the ability of Agile methodologies to influence cycle time?

5 Specific Objectives and Possible Outcomes

This research aims to delve into the specific aspects of Agile methodologies and their direct impact on the time taken for various phases of software development. It encompasses evaluating how Agile practices such as iterative development, continuous feedback, adaptive planning, and other Agile principles contribute to shortening or extending development cycle times. Our specific objectives and possible outcomes include:

5.1 Specific Objectives

- To evaluate the influence of Agile methodologies on reducing software development cycle time
- To examine the key factors within Agile practices that contribute to accelerated software development
- To assess the impact of Agile principles on enhancing collaboration and communication among development teams
- To identify challenges and potential drawbacks

5.2 Possible outcomes

- Comprehensive analysis of challenges and drawbacks
- identification of key agile practices for acceleration
- Enhanced collaboration among team members
- Adaptability to project size and complexity

6 Impact Identification

6.1 Time-to-Market Reduction

Agile methodologies aim to deliver functional increments quickly. Investigate how Agile practices contribute to a reduction in the overall time it takes to bring a product or feature to market compared to traditional development approaches.

6.2 Faster Iterative Development

Agile promotes iterative development with short feedback loops. Identify how this iterative approach affects the speed at which features and enhancements are developed, tested, and delivered.

6.3 Adaptability to Changes

Agile is known for its flexibility and ability to adapt to changing requirements. Explore how Agile practices impact the development team's response to changing customer needs or market conditions and how this adaptability influences cycle time.

6.4 Continuous Integration and Delivery

Agile's emphasis on continuous integration and delivery practices impacts the efficiency of the development pipeline. Explore whether these practices lead to a smoother and faster integration of code changes into the main codebase.

6.5 Improved Communication and Collaboration

Agile places a strong emphasis on communication and collaboration among team members. Explore how improved communication, regular stand-up meetings, and collaboration tools contribute to faster decision-making and development cycles.

6.6 Early and Regular Feedback

Agile methodologies prioritize customer feedback throughout the development process. Assess how early and regular feedback from stakeholders, end-users, or customers impacts the development cycle, allowing for quicker adjustments and improvements.

7 Outline of Methodology

The methodology for analyzing the impact of Agile methodologies on software development cycle time involves a structured approach that combines qualitative and quantitative research methods. Here's a proposed methodology:

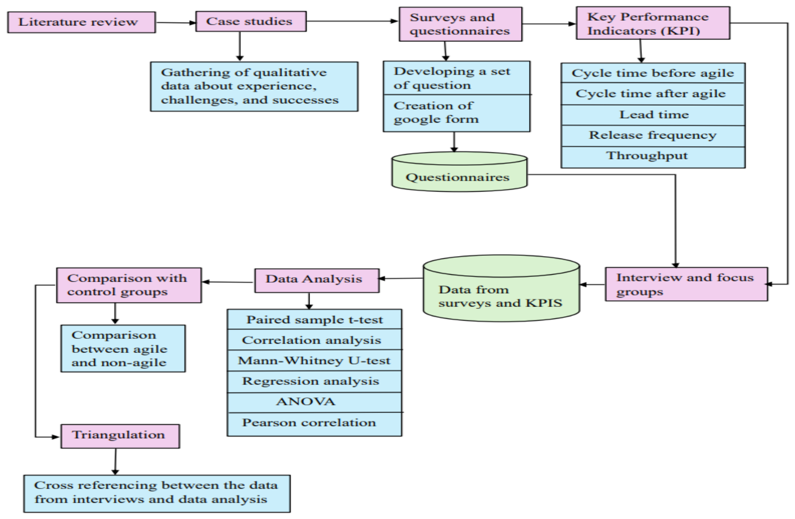

Literature review
Case studies
Surveys and questionnaires
Key Performance Indicators (KPI)
Gathering of qualitative data about experience, challenges, and successes
Developing a set of question
Creation of google form
Questionnaires
Cycle time before agile
Cycle time after agile
Lead time
Release frequency
Throughput
Comparison with control groups
Data Analysis
Data from surveys and KPIS
Interview and focus groups
Comparison between agile and non-agile
Paired sample t-test
Correlation analysis
Mann-Whitney U-test
Regression analysis
ANOVA
Pearson correlation
Triangulation
Cross referencing between the data from interviews and data analysis

Figure 1: Proposed Methodology

7.1 Literature Review

We need to conduct a comprehensive literature review to understand existing research, theories, and findings related to Agile methodologies, software development cycle time, and their interplay. Then we have to identify gaps in the literature that the current study aims to address.

7.2 Case Studies

We should select multiple organizations that have implemented Agile methodologies in their software development processes samples. Then we will conduct in-depth case studies to gather qualitative data on the experiences, challenges, and successes related to cycle time reduction.

7.3 Surveys and Questionnaires

In this stage, develop surveys or questionnaires to collect quantitative data from a broader sample of software development teams. We will include questions about Agile practices, cycle time metrics, collaboration, and perceived impacts.

7.4 Key Performance Indicators (KPIs)

Here, we will define key performance indicators (KPIs) to quantitatively measure cycle time before and after the adoption of Agile. Then we will utilize metrics such as lead time, release frequency, and throughput to assess the impact.

7.5 Interviews and Focus Groups

We will conduct interviews with key stakeholders, including developers, project managers, and Agile coaches, to gain deeper insights into the qualitative aspects of the impact on collaboration and communication.

7.6 Data Analysis

We need to employ statistical analysis tools to analyze quantitative data gathered through surveys and KPIs. Utilize qualitative data analysis techniques, such as thematic analysis, for insights from interviews and case studies.

7.7 Comparison with Control Groups

Establish control groups or compare Agile teams with non-Agile teams to isolate and attribute observed changes in cycle time specifically to Agile methodologies.

7.8 Triangulation

Triangulate findings by cross-referencing results from different data sources, ensuring the reliability and validity of the study.

8 Applications

In our research, we focus on the impact of Agile methodologies on software development life cycle time. In the application section, we will thoroughly state the potential usage of our research topic.

- **Optimizing Project Management:** Enhance project management efficiency by understanding how Agile methodologies impact software development cycle time, leading to better planning and resource allocation
- **Enhancing Team Collaboration:** Foster improved collaboration and communication within development teams through the application of Agile principles, resulting in more cohesive and efficient workflows
- **Mitigating Risks and Challenges:** Identify and address challenges associated with Agile adoption to mitigate potential risks, ensuring a smoother transition and successful implementation
- **Increasing Productivity and Throughput:** Implement best practices derived from the research to increase team productivity and throughput, ultimately leading to faster and more efficient software development

9 Required Resources

- Literature Resources: Academic journals, articles and books
- Access to Agile Teams: Collaboration with organizations or teams that have implemented Agile methodologies for firsthand data collection, including interviews, surveys, and observations

- Research Participants: Access to developers, project managers, Agile coaches,and other stakeholders for interviews, surveys, and focus groups
- Survey and Questionnaire Tools: Google Forms, SurveyMonkey, or Qualtrics
- Case Study Materials: Access to organizational documents, project reports, and other materials for in-depth case studies
- Data Analysis Software: Statistical analysis tools for processing and interpreting quantitative data, such as SPSS, R, or Python for statistical analysis

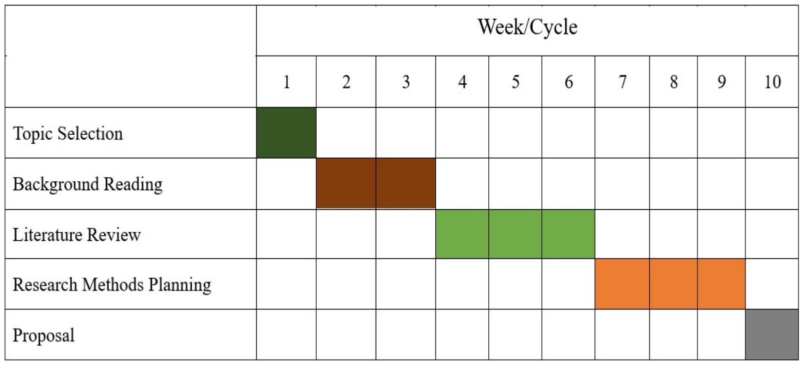
10 Cost Estimation

Table 1: Expense Breakdown

Expense Items	**Cost (Tk.)**
Survey or questionnaire tools	2050
Software/platforms for data collection and analysis	6000
Travel expenses for interviews or on-site observations	1200
Miscellaneous	500
Total	**9750**

10.1 Time Management

Gantt Chart for the entire timeline of the proposal of the research is given below:



	Week/Cycle									
	1	2	3	4	5	6	7	8	9	10
Topic Selection	🟩									
Background Reading		🟫	🟫							
Literature Review				🟩	🟩	🟩				
Research Methods Planning							🟧	🟧	🟧	
Proposal										⬛

Figure 2: The Gantt chart of the timeline of the proposal

11 SWOT Analysis

11.1 Strengths

- Strength lies in the relevance of the research topic to the software development industry, given the widespread adoption of Agile methodologies
- The research can have practical implications for organizations looking to enhance their software development processes by adopting Agile practices
- The opportunity to identify and learn from success stories where Agile methodologies have significantly impacted software development

cycle time

11.2 Weakness

- The complexity of software development environments and the multitude of variables involved might make it challenging to isolate and measure the direct impact of Agile methodologies
- The reliance on self-reported metrics and qualitative data may introduce subjectivity and bias, affecting the accuracy of findings related to cycle time
- The diversity in how organizations implement Agile methodologies could make it challenging to generalize findings across different contexts

11.3 Opportunities

- The opportunity to draw insights from a diverse range of industries and types of software projects, enhancing the applicability of findings
- Identifying opportunities for innovation in Agile practices that could further optimize software development cycle time
- Investigating the impact of Agile on team dynamics may uncover opportunities to enhance collaboration, communication, and overall team performance

11.4 Threats

- Resistance to the adoption of Agile methodologies within organizations may limit the availability of data or affect the validity of research findings
- Potential challenges in collecting accurate and reliable data on software development cycle time, especially if organizations lack robust measurement
- External factors such as economic conditions, industry trends, or global events may impact the relevance and generalizability of findings

References

[1] A. Begel and N. Nagappan, "Usage and perceptions of agile software development in an industrial context: An exploratory study," in *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*, pp. 255–264, IEEE, 2007.

[2] P. Maher, "Weaving agile software development techniques into a traditional computer science curriculum," in *2009 Sixth International Conference on Information Technology: New Generations*, pp. 1687–1688, IEEE, 2009.

[3] A. Zuo, J. Yang, and X. Chen, "Research of agile software development based on formal methods," in *2010 International Conference on*

Multimedia Information Networking and Security, pp. 762–766, IEEE, 2010.

[4] O. Salo and P. Abrahamsson, "Integrating agile software development and software process improvement: a longitudinal case study," in *2005 International Symposium on Empirical Software Engineering, 2005.*, pp. 10–pp, IEEE, 2005.

[5] X. Wang, "The combination of agile and lean in software development: An experience report analysis," in *2011 Agile Conference*, pp. 1–9, IEEE, 2011.

[6] R. Mordinyi, E. Kühn, and A. Schatten, "Towards an architectural framework for agile software development," in *2010 17th IEEE International Conference and Workshops on Engineering of Computer Based Systems*, pp. 276–280, IEEE, 2010.

[7] J. A. Livermore, "Factors that impact implementing an agile software development methodology," in *Proceedings 2007 IEEE SoutheastCon*, pp. 82–86, IEEE, 2007.

[8] A. Ahmed, S. Ahmad, N. Ehsan, E. Mirza, and S. Sarwar, "Agile software development: Impact on productivity and quality," in *2010 IEEE International Conference on Management of Innovation & Technology*, pp. 287–291, IEEE, 2010.

[9] Y. Wang, D. Sang, and W. Xie, "Analysis on agile software development methods from the view of informationalization supply chain management," in *2009 Third International Symposium on Intelligent*

Information Technology Application Workshops, pp. 219–222, IEEE, 2009.

[10] T. Dyba, "Improvisation in small software organizations," *IEEE software*, vol. 17, no. 5, pp. 82–87, 2000.

[11] P. GERFALK and B. Fitzgerald, "Old petunias in new bowls?," *Communications of the ACM*, vol. 49, no. 10, p. 27, 2006.

[12] K. Beck, *Extreme programming explained: embrace change*. addison-wesley professional, 2000.

[13] J. Stapleton, *DSDM, dynamic systems development method: the method in practice*. Cambridge University Press, 1997.

[14] K. Schwaber and M. Beedle, *Agile software development with Scrum*. Prentice Hall PTR, 2001.

[15] M. Aoyama, "Web-based agile software development," *IEEE software*, vol. 15, no. 6, pp. 56–65, 1998.

[16] A. C. Pacagnella Junior and V. R. Da Silva, "20 years of the agile manifesto: A literature review on agile project management," *Management and Production Engineering Review*, vol. 14, 2023.

[17] M. Kajko-Mattsson, G. A. Lewis, D. Siracusa, T. Nelson, N. Chapin, M. Heydt, J. Nocks, and H. Snee, "Long-term life cycle impact of agile methodologies," in *2006 22nd IEEE International Conference on Software Maintenance*, pp. 422–425, IEEE, 2006.

[18] G. Miller, “Want a better software development process’? complement it,” *IT professional*, vol. 5, no. 5, pp. 49–51, 2003.