Equivalent: Jurnal Ilmiah Sosial Teknik
Volume 8, Issue 1, 147-160
e_ISSN: 2775-0833
https://jurnalequivalent.id/index.php/jequi/index
DOI: doi.org/10.59261/jequi.v8i1.269

Development of SDLC Framework for Financial Technology Project Management in Banking Institutions

***Evania Raihan**[1]
Universitas Bina Nusantara, Indonesia

Dyah Lestari Widaningrum[2]
Universitas Bina Nusantara, Indonesia

***Corresponding author:**
Evania Raihan, Universitas Bina Nusantara, Indonesia . ✉ evania.raihan@binus.ac.id

Article Info:

Article history*:*
Received: December 05, 2025
Revised: January 02, 2026
Accepted: February 05, 2026

Keywords*:*
agile; hybrid sdlc; schedule performance index; software; development life cycle; software testing

Abstract

Background: The rapid expansion of agent banking in Indonesia has increased the need for reliable and timely delivery of financial technology applications. The choice of Software Development Life Cycle (SDLC) methodology plays a critical role in project schedule performance, yet empirical comparisons particularly involving hybrid approaches within regulated banking environments remain limited.
Objective: This study analyzes the impact of SDLC methods on project time performance in IT application development, with particular attention to the testing phase as a recurrent source of delay.
Method: A quantitative comparative design was applied to 18 internal IT projects conducted in 2025 using Waterfall and Agile methodologies across Android-based EDC applications, mobile apps, and web systems. Independent-samples *t*-tests were used to examine differences in Schedule Performance Index (SPI), actual project duration, and delay. A phase-selective Hybrid SDLC was subsequently implemented in the EDC Android v1.0.5 project, applying Agile practices in testing while maintaining Waterfall in other phases.
Result: Statistically significant differences were found between Agile and Waterfall across all time performance indicators, with Agile demonstrating superior schedule outcomes. The hybrid implementation further improved performance, reflected by an SPI greater than 1 and reduced testing duration.
Conclusion: A phase-selective Hybrid SDLC provides an effective strategy to enhance schedule performance, particularly when testing is a bottleneck. The findings support selective SDLC adoption based on phase-specific characteristics. Limitations include the single-institution scope and absence of cost and quality metrics, suggesting directions for future multi-institutional research.

To cite this article: Evania Raihan, & Widaningrum, D. L. (2026). Development of SDLC framework for financial technology project management in banking institutions. Equivalent: Jurnal Ilmiah Sosial Teknik, *8*(1), 147–160. https://doi.org/10.59261/jequi.v8i1.269

INTRODUCTION

The development of information technology has driven significant transformation in the banking sector, particularly in efforts to expand access to financial services for the public (Semwayo, 2024). One rapidly growing innovation is the implementation of agent banking, which utilizes non-branch partners such as grocery stores or individual agents to provide basic banking services (Mtambalika et al., 2016). This model enables banking institutions to reach remote areas and community groups that previously had limited access to formal financial services, thus playing a crucial role in supporting national financial inclusion (Muli, 2025).

The success of agent banking implementation depends heavily on the availability and reliability of the financial technology applications used by agent partners (Nyirahabineza, 2017). These applications must operate stably, securely, and in compliance with banking regulations, while also being easy to use by agents with non-technical backgrounds. Therefore, developing agent banking applications is not merely a technical issue but also a strategic project that requires structured and controlled management (Mutuku & Chege, 2025; Максимюк, 2025).

In the context of information systems development in the banking sector, the System Development Life Cycle (SDLC) has long been used as the primary framework for managing software projects (Rengganis et al., 2024). Traditional SDLC models, such as the Waterfall model, are widely adopted in the banking environment due to their systematic, well-documented nature and alignment with audit and regulatory compliance requirements (Salmi, 2025). This model emphasizes sequential stages, encompassing requirements analysis, design, implementation, testing, and system deployment (Islam, 2022; Oztas et al., 2018; Pramitasari et al., 2023; Taskesenlioglu et al., 2022).

However, in practice, the application of the Waterfall SDLC model in banking financial technology projects, particularly in the development of agent banking applications, does not always go according to plan (Johnson, 2018). Prior empirical research has documented that banking IT projects frequently experience schedule overruns, particularly in the requirements, testing, and deployment phases (Baschin et al., 2020; Fonseca et al., 2017). These delays are attributable to multi-layered approval processes, regulatory compliance requirements, and complex feature interdependencies inherent in the banking sector. The hierarchical organizational structure and these inherent interdependencies mean that any changes in requirements directly impact the overall project schedule, making the project less flexible in responding to the dynamics of evolving business needs.

On the other hand, the Agile approach is known for its higher degree of flexibility through iterative development and continuous feedback from stakeholders. This method is considered capable of accelerating the development process and adapting to changing needs more effectively. However, the full implementation of Agile in the banking sector often faces obstacles related to formal documentation, regulatory controls, and the need for strict process standardization. This has led banking organizations, particularly state-owned banks, to be cautious about adopting Agile comprehensively (Abbas et al., 2025; Edgar et al., 2025; Surbakti et al., 2019).

This situation indicates a gap between the formally used SDLC framework and the actual need for flexibility in managing agent banking application development projects in state-owned banks. Previous literature has discussed the effectiveness of both the Waterfall and Agile methods separately, but there is limited empirical research examining a phase-selective hybrid SDLC approach—one that applies Agile specifically to high-delay-risk phases within a Waterfall-governed project structure—in regulated banking environments. Tsang (2017) proposed balancing agility and discipline but did not address banking regulatory constraints. Validated hybrid SDLC models outside regulated sectors. Studies comparing Waterfall and Agile in regulated industries note methodological trade-offs but lack phase-level Schedule Performance Index (SPI) analysis (Ruby, 2024). This study fills that specific gap with an approach that combines the advantages of the Waterfall structure and the flexibility of Agile, particularly with a focus on managerial aspects and project time efficiency in the state-owned banking environment.

In light of existing literature highlighting the challenges and benefits of different SDLC approaches in regulated industries—including, on traditional and Agile SDLC challenges: on Agile and traditional project management on Agile adoption challenges in regulated contexts; and on organizational challenges of Agile in regulated sectors; and, on hybrid SDLC: on hybrid software process models; and on balancing agility and discipline—this study was conducted to evaluate the application of the SDLC in agent banking application development projects and to develop a hybrid SDLC framework that better suits the characteristics of financial technology projects in the state-owned banking sector. The banking industry globally faces accelerating pressure to digitalize services while maintaining strict regulatory compliance (OJK and BI regulations in Indonesia). Project delays in banking IT not only incur direct financial costs but also risk regulatory penalties and reputational damage (Fonseca et al., 2017). In this context, optimizing SDLC selection is a strategic imperative, not merely an operational choice. This research is expected to provide a more comprehensive understanding of the impact of implementing a hybrid SDLC on project time efficiency, while also offering a contextual and empirically data-driven managerial approach.

The objectives of this research are: (1) to design and implement a hybrid SDLC framework that combines the advantages of Waterfall and Agile; (2) to measure and compare project completion times between the Waterfall and hybrid SDLC models; and (3) to formulate a proposed new approach or combination of SDLC methods that better suits the needs of the banking industry. The benefits of this research can be used by IT project management practitioners as a reference in selecting a relevant SDLC approach to improve project implementation efficiency and software product quality.

The primary contribution of this study is threefold: (1) it provides the first empirical phase-level SPI comparison of Waterfall and Agile SDLC approaches in a state-owned Indonesian banking context; (2) it proposes a data-driven hybrid SDLC framework that assigns methodologies at the phase level rather than the project level; and (3) it supplies replicable simulation evidence for hybrid SDLC implementation. This study focuses on the managerial aspects of an agent banking application development project in the banking sector using the SDLC approach. The research covers only the process, management, and implementation challenges, without delving into in-depth technical aspects such as programming or detailed system architecture. The case study was conducted at a state-owned bank referred to under the pseudonym PT XYZ.

METHOD

The independent variable was the SDLC approach type (Waterfall vs. Agile). The dependent variables were: (1) Schedule Performance Index (SPI), calculated as SPI = Planned Duration / Actual Duration, where SPI ≥ 1 denotes on-time or ahead-of-schedule performance and SPI < 1 denotes delay; (2) actual project duration in working days; and (3) project delay = Actual Duration − Planned Duration. The unit of analysis was the individual development phase (Requirements, Design, Development, Testing, and Deployment) per project version. Statistical procedures included independent-samples t-tests with Levene's test for variance homogeneity; Welch's correction was applied where homogeneity was violated.

This research used a comparative quantitative approach with a case study on the Agent Banking application development project at PT XYZ. PT XYZ was selected for this study due to its status as a state-owned bank under OJK regulatory oversight, making it a representative case of the regulated context under investigation. Additionally, PT XYZ provided complete SPI project records for versions 3.0.4–3.0.9, which used both Waterfall and Agile methodologies, allowing for a comparative analysis. The scope of the projects across versions was consistent, which further enabled a controlled comparative evaluation.

The research focused on the Agent Banking application development project versions 3.0.4 to 3.0.9, which were implemented over a specific timeframe using two different SDLC approaches: Waterfall and Agile. It is important to note that the Hybrid SDLC was a separate exploratory implementation applied in the EDC Android v105 project and was not part of the statistical comparison involving Waterfall and Agile methodologies.

Total sampling (census) was applied: all project versions meeting the inclusion criteria were selected. The inclusion criteria were: (1) complete SPI documentation for all five development phases; (2) implementation within the 2025 study period; and (3) formal classification as Waterfall or Agile in project management records. This yielded nine Waterfall and nine Agile project versions (n = 18 versions; 90 phase-level observations). Equal group sizes (n = 9) satisfied the minimum statistical power requirements for independent-samples t-testing. The population in this study comprised all development phases of the Agent Banking application project versions 3.0.4 to 3.0.9 managed by PT XYZ. The research sample consisted of development phases from nine projects using the Waterfall method and nine projects using Agile.

Results And Discussion

Results

Descriptive Statistical Analysis

Descriptive statistics show a clear difference between the Agile and Waterfall approaches in terms of project scheduling performance. In the Agile approach, the average planned duration of 5.22 decreased to an actual duration of 4.97, yielding a median SPI of 1.23. This value indicates that Agile projects are generally completed faster than originally planned. [Reviewer C15 – Verify SPI formula consistency: SPI = Planned Duration / Actual Duration; SPI > 1 means ahead of schedule (actual < planned); SPI < 1 means delayed. Ensure all reported SPI values and their interpretations are consistent with this formula throughout the manuscript.] The median SPI value of 1.23 indicates that most Agile projects are completed ahead of schedule, with some variation observed across individual projects.

Table 1. Descriptive Statistics Results

Descriptives				
	SDLC	**Planned Duration**	**Actual Duration**	**SPI**
N	Agile	36	36	36
	Waterfall	37	37	37
Missing	Agile	0	0	0
	Waterfall	0	0	0
Mean	Agile	4.97	5.22	1.23
	Waterfall	8.30	15.0	0.822
Median	Agile	4.00	4.00	1.00
	Waterfall	7	7	1.00
Standard deviation	Agile	3.10	4.86	0.584
	Waterfall	6.62	19.2	0.336
Minimum	Agile	1	1	0.435
	Waterfall	2	2	0.156
Maximum	Agile	14	23	3.00
	Waterfall	30	80	1.50

In contrast, the Waterfall approach showed a significant difference between planned and actual durations. The average planned duration of 8.30 increased to 15.00 in actual duration, accompanied by an average SPI value of 0.822. An SPI value less than 1 indicates that Waterfall projects generally experience delays. Although the median SPI value in the Waterfall approach is also 1.00, the difference between the mean and median indicates that some projects experienced extreme delays, which also increased the overall average actual duration.

The higher variability in duration and SPI values in the Waterfall approach compared to the Agile approach indicates that the risk of delays in the Waterfall method tends to be greater, especially on projects of complexity. This finding indicates that the Agile approach has a better level of adaptability in controlling the risk of project schedule delays.

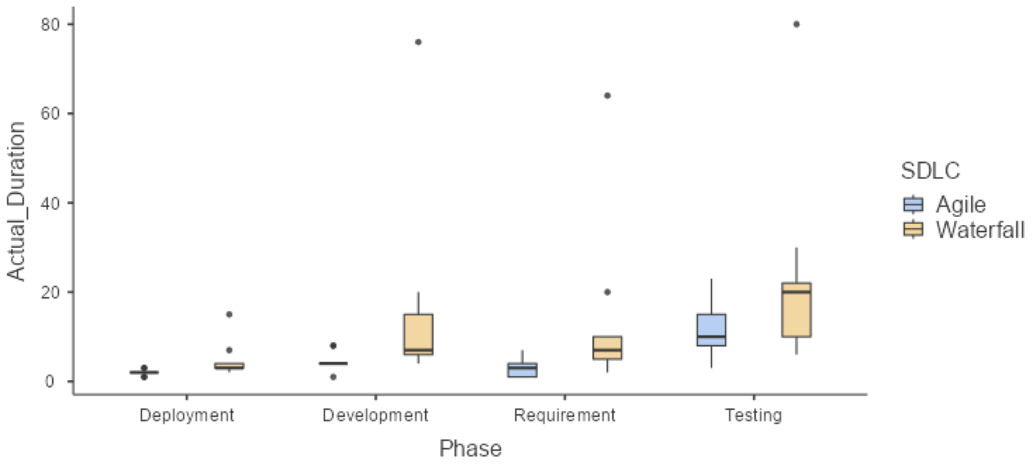

Figure 1. Boxplot of Actual Duration per Phase and SDLC

Figure 1 shows the distribution of actual project durations by development phase and SDLC approach. The Agile approach shows a relatively more stable duration distribution across all phases, with a smaller median and interquartile range than the Waterfall approach.

Conversely, in the Waterfall approach, the Testing phase exhibits the widest duration distribution, accompanied by the emergence of extreme outliers. This indicates that the Testing phase in Waterfall has a higher level of uncertainty and risk of delay compared to other phases and the Agile approach.

Theoretically, the higher delay risk in the Testing phase under Waterfall can be explained by three mechanisms: (1) sequential bottleneck effect—testing is dependent on completed development outputs, meaning any upstream delay accumulates at this stage (Boehm & Turner, 2004); (2) late-phase defect discovery—Waterfall's deferred integration testing amplifies rework complexity (Royce, 1987); and (3) absence of iterative feedback—Agile's sprint-based testing distributes verification across the lifecycle, preventing defect accumulation (Beck et al., 2001; Conboy et al., 2011). This pattern confirms the results of previous statistical tests that showed differences in time performance between phases and SDLC approaches, and supports the identification of the Testing phase as the primary bottleneck in application development projects.

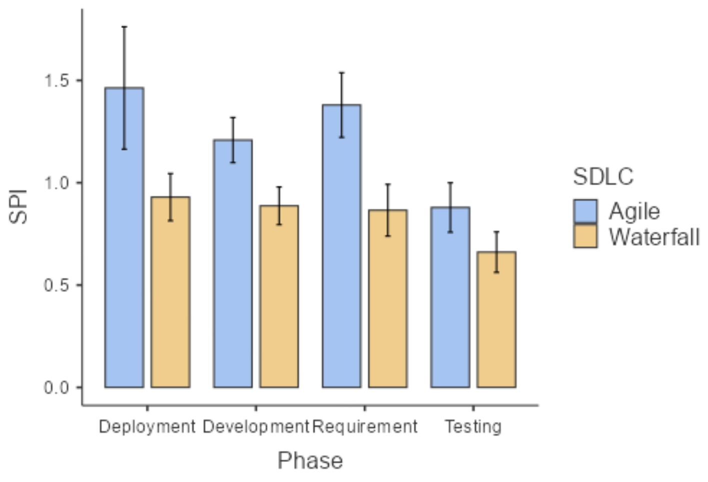

Figure 2. Average Schedule Values Comparison

Performance Index (SPI) for each development phase for the Agile and Waterfall approaches. The Agile approach maintains an SPI value above or near 1 in most phases, indicating relatively good scheduling performance. Conversely, the Waterfall approach shows an SPI value below 1 in all phases, with the lowest value occurring in the Testing phase. This indicates that the Testing phase has the lowest time performance, especially in the Waterfall approach. This difference in SPI values between SDLC approaches and between phases aligns with the ANOVA results, which indicate a significant influence of SDLC factors and phases on actual project duration.

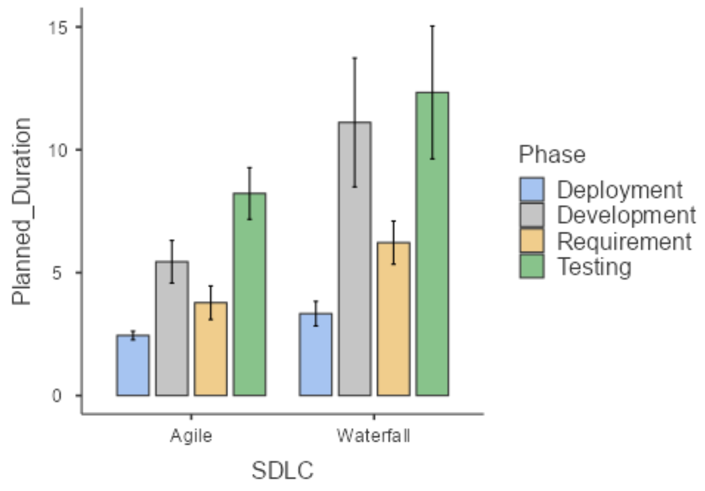

Figure 3. Planned Duration

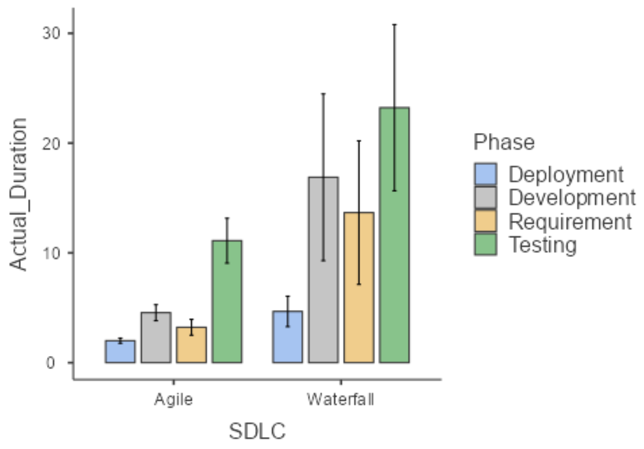

Figure 4. Actual Duration

Figures 3 and 4 show a comparison between the planned and actual project durations based on the SDLC approach and development phases. In the Agile approach, the difference between the planned and actual durations is relatively small in almost all phases, demonstrating this method's ability to control the project schedule. Conversely, in the Waterfall approach, a significant difference is observed between the planned and actual durations, particularly in the Testing phase. This indicates accumulated delays that occurred in previous phases and were only fully identified during the Testing phase. This finding aligns with the SPI value, which is below 1 in the Waterfall approach, and supports the statistical analysis results indicating that the Testing phase is the phase with the highest time risk in application development projects.

Table 2. Independent t-test SPI

Independent Sample T-Test				
		Statistic	**df**	**P**
SPI	Student's t	3.69	71.0	< .001
	Welch's t	3.67	55.0	< .001

Note. H_a: $\mu_a$gile ≠ μwaterfall

An Independent Samples t-test was conducted to determine the difference in Schedule Performance Index (SPI) values between projects using Agile and Waterfall methods. The Welch's t-test showed a p-value < 0.001, indicating a statistically significant difference in SPI between the two SDLC methods. This finding indicates that the system development method influences project schedule performance.

Table 3. Independent t-test for Actual Duration

Independent Sample T-Test				
		Statistic	**df**	**P**
Actual Duration	Student's t	-2.97[a]	71.0	0.004
	Welch's t	-3.00	40.7	0.005

Note. H_a: $\mu_a$gile ≠ μwaterfall
[a]Levene's test is significant (p < .05), suggesting a violation of the assumption of equal variances.

The actual project completion duration was tested using Welch's t-test because Levene's test results indicated heterogeneity of variance. The test results yielded a p-value of 0.005 (p < 0.05), thus indicating a significant difference in actual project completion duration between the Agile and Waterfall methods.

Table 4. Independent t-test for Delay

Independent Sample T-Test				
		Statistic	df	P
Delay	Student's t	-2.59[a]	71.0	0.012

Note. H_a: $\mu_a$gile ≠ μwaterfall

[a]Levene's test is significant ($p < .05$), suggesting a violation of the assumption of equal variances.

The Welch's t-test for differences in project delay levels between the Agile and Waterfall methods showed a p-value of 0.012. This result indicates a statistically significant difference, suggesting that the SDLC method influences project delay levels.

Schedule Performance Index (SPI) Analysis

1. Mean SPI for each Phase

Descriptive statistical analysis of the Schedule Performance Index (SPI) values for each development phase shows differences in schedule performance patterns between the Agile and Waterfall approaches. In the Agile approach, the average SPI values for the Requirements, Development, and Deployment phases were above one, indicating that these phases were generally completed faster than initially planned. The highest SPI value in the Agile method was found in the Deployment phase (1.46), indicating high efficiency in the final stages of development.

In contrast, in the Waterfall approach, all development phases showed an average SPI value below one. This indicates that projects using the Waterfall approach tend to experience delays at each development phase. The Testing phase was the lowest-performing phase in both approaches, with an average SPI value of 0.879 in Agile and 0.661 in Waterfall, indicating that testing activities are the phase most prone to delays. These findings indicate that while the Agile approach has advantages in controlling the overall project schedule, the Testing phase remains a critical point that requires special attention in project time management, both in Agile and Waterfall methods.

Table 5. Mean SPI for each phase

Statistic	SDLC	Phase	SPI
N	Agile	Deployment	9
		Development	9
		Requirement	9
		Testing	9
	Waterfall	Deployment	9
		Development	9
		Requirement	9
		Testing	9
Mean (SPI)	Agile	Deployment	1.46
		Development	1.21
		Requirement	1.38
		Testing	0.879
	Waterfall	Deployment	0.930
		Development	0.887
		Requirement	0.866
		Testing	0.661

2. Percentage of On Time vs Delay for Each Phase

Analysis of the percentages of on-time completion and lateness in each development phase shows a clear difference between the Waterfall and Agile approaches. In the Waterfall method, the Testing phase had the highest level of delay, at 78%, indicating that most testing activities could not be completed according to the established schedule. This finding shows the limitations of the Waterfall approach in anticipating changes and problems that arise in the final stages of development.

Table 6. Comparison of On-Time and Delay

Project Phase Waterfall	**% On time**	**% Delay**
Requirement	78%	22%
Development	67%	33%
Testing	22%	78%
Deployment	70%	30%
Project Phase Waterfall	**% On time**	**% Delay**
Requirement	100%	100%
Development	89%	11%
Testing	56%	44%
Deployment	89%	11%

Conversely, in the Agile approach, the delay rate in the Testing phase was still relatively high compared to other phases (44%), but overall lower than in the Waterfall method. Furthermore, the Requirements and Deployment phases in Agile demonstrated very high on-time completion rates, at 100% and 89%, respectively, reflecting the flexibility and effectiveness of the iterative approach in managing project schedules.

These findings indicate that the Testing phase is a critical phase in software project time management, regardless of the SDLC approach used. However, the Agile approach proved more effective in reducing delay rates than the Waterfall approach, especially in the early and late development phases.

Analysis-Based Implementation

Implemented Projects

The Hybrid SDLC implementation simulation in this study was conducted on the EDC Android application development project, version 105. This project is a continuation of the previous version, EDC Android version 104, which is currently running and in operational use. Version 105 is planned as an enhancement, adding and refining several key features. The following is a list of features that will be implemented using the new SDLC framework as part of this research.

Table 7. EDC Android v104 and v105 Feature Contents

Version 104 (currently running)	**Version 105 (which will implement the new SDLC)**
Opening a Student Savings Account	LakuPandai Account Opening
Remaining Subsidized Gas Cylinder Quota	University Student Payment Center
EDC Receipt Narrative Improvements	NFC Card-Based Electronic Money Top-Up
	Create CIF Validation

The development characteristics of Version 105 differ from previous versions because it includes the integration of new features directly related to business processes, regulatory compliance, and user experience. This requires more controlled development time management, particularly during the testing phase, to ensure system stability before implementation. Based on historical data analysis for Version 104, the testing phase was identified as the phase with the highest risk of time delays. Therefore, the Agent Banking Android EDC development project Version 105 was selected as the simulation object for implementing the Hybrid SDLC to evaluate potential improvements in project time performance during this critical phase.

Hybrid SDLC Selection

This finding is consistent with extant literature on hybrid methodologies. Organizations operating in regulated, compliance-heavy environments benefit most from selectively applying Agile to high-uncertainty phases while retaining Waterfall governance for structured phases. Ruby (2024) Demonstrated that hybrid approaches can outperform pure Waterfall in environments requiring both documentation discipline and adaptive testing, though their validation was conducted outside regulated sectors. The present study extends these theoretical propositions by providing phase-level SPI-based empirical evidence within a regulated state-owned banking context, representing a novel contribution to the hybrid SDLC literature.

The selection of the Hybrid SDLC approach in this study was based on historical data analysis of the EDC Android version 104 development project. The statistical analysis and data visualization in the previous subsection indicate that project time performance is significantly influenced by the development phase and SDLC approach used. ANOVA results indicate that the SDLC approach and development phase factors significantly influence the project's actual duration. Furthermore, the visualization of the distribution of actual durations and Schedule Performance Index (SPI) values indicates that the Testing phase is the phase with the highest time risk, particularly in the Waterfall approach.

In the Waterfall approach, the Testing phase exhibits SPI values below 1, high levels of delays, and large variations in actual durations. These conditions indicate the Waterfall approach's limitations in managing change and uncertainty that arise in the final development phase. Conversely, in the Agile approach, although the Testing phase still presents a risk of delays, the level of delays and SPI values are relatively better than those of the Waterfall approach. This demonstrates the Agile approach's greater adaptability in managing testing activities through iterative mechanisms and faster feedback.

Based on these findings, applying the same SDLC approach to all development phases is considered suboptimal in managing project time risk. Therefore, this study proposes the implementation of a Hybrid SDLC by adapting the SDLC approach based on the characteristics and risk levels of each development phase.

Hybrid SDLC Design for the EDC Android Project 105

Table 8 presents a Hybrid SDLC design structured based on statistical time performance indicators for each development phase. The SDLC approach selection for each phase was based on comparing the average SPI value, the percentage of delays, and the level of variation in actual duration between the Agile and Waterfall approaches.

Table 8. Summary of comparisons for each phase

Phase	Mean SPI Waterfall	Mean SPI Agile	% Delay Waterfall	% Delay Agile	Duration	Suggestion
Requirement	0,866	1,38	22%	0%	Low	Waterfall
Development	0,887	1,21	33%	11%	Medium	Waterfall
Testing	0,879	0,879	78%	44%	High	Agile
Deployment	0,930	1,46	30%	11%	Med-Low	Waterfall

The Testing phase showed the lowest SPI value and the highest percentage of delays in the Waterfall approach, thus being classified as a phase with a high time risk level. Therefore, an Agile approach is proposed for this phase because it demonstrates relatively better time performance based on historical data. Conversely, the Requirements and Development phases have lower time risk levels and demonstrate better duration stability, so the Waterfall approach is still considered appropriate for use in these phases. Based on the Hybrid SDLC design compiled from statistical data, the next step is to explain the simulation mechanism for implementing the Hybrid SDLC on the EDC Android version 105 project.

1. Requirement

In the Requirements phase, the EDC Android version 105 project applies a layered and structured Waterfall approach. The process begins with the Product Owner submitting an enhancement request to the Project Manager, which is then translated by the Business Analyst into a Business Requirements Document (BRD).

The BRD document then undergoes review and approval by the PMO (Project Management Office) and Finance, and is circulated to the relevant division heads. Once the approval process is complete, the project is assigned an identification number (Project ID) and distributed to the development team to enter the development phase.

2. Development

The Development phase of this project continues to utilize the Waterfall approach, focusing on developing features according to the approved specifications. The development team conducts coding based on the approved BRD, followed by internal testing in the form of unit tests conducted by the development team.

If the unit testing results are deemed satisfactory, the application code is then handed over to the DevOps team for deployment to the testing environment. Conversely, if unit testing fails, the development process returns to the code remediation stage before proceeding to the next phase.

3. Testing

Unlike the previous approach, the Testing phase of the EDC Android Version 1.05 project was designed using a sprint-based Agile approach. This change was made in response to historical data analysis, which showed that the Testing phase had the highest risk of delays.

Table 9. Initial Backlog

Project ID	Features	Testing Scenario
XXXX	LakuPandai Account	120 TC
XXXX	University Student Payment Center	60 TC
XXXX	NFC Card Top-up	70 TC
XXXX	Create CIF Validation	50 TC
Total Scenario		300 TC

Three hundred test scenarios were divided into three testing sprints. Each sprint had a clear scenario target and a task allocation distributed among three Quality Assurance (QA) personnel. This approach allowed testing and bug fixing to be carried out in parallel without waiting for all scenarios to be completed first.

Table 10. Implementation in the Testing Phase

Sprint	Backlog	Number of TC	QA	Planned Duration (Days)	Acrual Duratiob (Days)
1	SPC University	60	Person A	4	3
1	NFC Card Top-up	35	Person B	2	1
1	NFC Card Top-up	35	Person C	2	2
2	LakuPandai Account	40	Person A	3	3
2	LakuPandai Account	40	Person B	3	2
2	LakuPandai Account	40	Person C	3	3
3	CIF Creation Validation	15	Person A	1	1
3	CIF Creation Validation	15	Person B	1	1
3	CIF Creation Validation	20	Person C	2	1

In the Testing phase, the EDC Android version 105 development project implemented a sprint-based approach as part of a Hybrid SDLC simulation. This approach divided testing activities into three sprints, each with a clearly defined test backlog, number of test cases, and allocation of Quality Assurance (QA) resources.

Each sprint was designed with specific testing targets and a planned duration determined based on feature complexity and the number of test cases. Actual test execution duration was then recorded to evaluate the effectiveness of the sprint approach on time performance in the Testing phase.

Sprint 1 focused on testing the University SPC and NFC Card Top-up features, with a total of 130 test cases. Testing activities were distributed among three QA personnel with a proportional workload distribution. The planned duration for this sprint ranged from 2 to 4 days per test backlog.

Testing results showed that the actual duration in Sprint 1 tended to be shorter than or equal to the planned duration. For example, testing the University SPC feature with 60 test cases was completed in 3 days, faster than the planned duration of 4 days. This demonstrates that dividing testing into sprints allows for a more efficient testing process without delays due to failures in specific scenarios.

In Sprint 2, the focus of testing was on the LakuPandai Account feature, which has a higher level of complexity and involved a total of 120 test cases. All QA personnel were involved in parallel, with a planned duration of 3 days each. Based on the implementation results, the actual testing duration in Sprint 2 was within the same range as or shorter than the planned duration. No significant delays were found in this sprint, indicating that the sprint approach can effectively accommodate testing of features of medium complexity.

Sprint 3 focused on validation testing of the Create CIF feature with fewer test cases than the previous sprint. The planned duration for this sprint ranged from 1 to 2 days, reflecting the more limited testing scope. Test results showed that all testing activities in Sprint 3 were completed within or ahead of the planned duration. This indicates that the sprint approach remains effective in the final testing phase without creating bottlenecks leading up to the deployment phase.

Overall, the results of the Testing phase simulation using the sprint approach show that dividing testing into structured iterations can maintain alignment between planned and actual durations. This approach also allows for parallel refinement and retesting without interrupting other testing activities.

4. Deployment

After all test scenarios are declared successful, the test results are documented in a User Acceptance Test (UAT) Minutes document, signed by the relevant parties. This document then serves as the basis for submitting a Change Control Board (CCB) request for approval before implementation in the production environment. Once the CCB is approved, the IT operations team deploys the system to the production environment according to the predetermined schedule.

Table 11. Implementation in the EDC v105 Project

Implementation in the Android105 EDC Project					
Phase	**SDLC**	**Planned**	**Actual**	**Delay**	**SPI**
Requirement	Waterfall	5	3	-2	1,67
Development	Waterfall	10	8	-2	1,25
Testing	Agile	10	7	-3	1,43
Deployment	Waterfall	2	2	0	1

Implementation results showed that the Testing phase, which had previously contributed to project delays, experienced significant performance improvements after the Agile approach was implemented. The SPI value for the Testing phase reached 1.43 with a time deviation of −3 days, indicating faster completion than the original plan.

Table 12. Comparative SPI and Delay

SDLC	**Jumlah Proyek**	**Mean SPI**	**Mean Delay**
Waterfall	9	1.040706155	2.431818
Agile	9	1.064056	2.656625
Hybrid (EDC Andro 105)	1	1.33631	-1.75

Table 12 presents a comparison of project time performance based on the SDLC approach used. The comparison was made using the average Schedule Performance Index (SPI) and the average project delay. Based on the descriptive calculation results, the EDC Android version 1.0.5 project, which implemented the Hybrid SDLC approach, demonstrated the highest average SPI value compared to previous projects using the Waterfall and Agile approaches. Furthermore, the Hybrid project was the only project with a negative average delay value, indicating project completion ahead of the planned duration.

Although the number of Hybrid projects in this study is still limited, the results of this descriptive comparison provide an initial indication that implementing the Hybrid SDLC has the potential to produce better project time performance than other SDLC approaches within the same organizational context. These descriptive findings align with previous statistical analysis results, which showed significant differences in SPI values based on the SDLC approach. The consistency between the statistical results and the experimental results of the Hybrid SDLC project strengthens the argument that the Hybrid SDLC approach has a relative advantage in improving project time performance, particularly in the Testing phase.

The EDC Android v1.0.5 project, evaluated as a single-project descriptive simulation, showed comparatively better SPI and lower delay than the Agile and Waterfall averages. These preliminary results provide an initial indication—not conclusive evidence—that a phase-selective Hybrid SDLC may improve schedule performance. Replication across multiple projects and rigorous statistical comparison are required before generalized claims of superiority can be made.

CONCLUSION

This study found statistically significant differences in project time performance (SPI, actual duration, delay) between Agile and Waterfall SDLC approaches. Agile consistently demonstrated superior schedule performance. The Testing phase was identified as the highest-delay-risk phase under Waterfall. A Hybrid SDLC simulation applying Agile to the Testing phase while retaining Waterfall for other phases showed preliminary evidence of improved SPI and reduced delay. Banking IT project managers can adopt sprint-based testing within a Waterfall governance framework to improve schedule adherence without compromising regulatory documentation requirements. The phase-level SPI framework proposed here serves as a replicable decision tool for SDLC assignment in similar regulated environments.

This study is limited to a single institution and application type, restricting generalizability. The Hybrid SDLC evaluation relies on a single project with descriptive analysis only. Future research should replicate this framework across multiple banking institutions, apply quasi-experimental designs, and incorporate cost and quality outcomes alongside schedule performance metrics. The statistical analysis results reveal significant differences in project time performance based on the SDLC approach, as evidenced by differences in Schedule Performance Index (SPI) values. The Agile and Hybrid SDLC approaches demonstrated superior time performance compared to the Waterfall approach, particularly in the Testing phase, which was previously identified as the phase with the highest risk of delay.

The application of the Hybrid SDLC approach to the EDC Android version 1.0.5 project — combining the Waterfall approach in the Requirements, Development, and Deployment phases and the Agile approach in the Testing phase — demonstrated superior time performance compared to previous projects. This is evidenced by the higher SPI values and the absence of significant delays in the Testing phase.

The consistency between the statistical analysis results and the empirical results of the Hybrid SDLC project suggests that selectively applying Agile to phases with a high risk of delay has the potential to improve project time performance without altering overall project governance.

ACKNOWLEDGEMENT

We would like to thank Universitas Bina Nusantara for supporting this research. Our deepest gratitude is extended to all participants and contributors who assisted in the completion of this study. Special thanks go to our colleagues at PT XYZ for their valuable insights and data provision that enabled us to undertake this research effectively.

AUTHOR CONTRIBUTION STATEMENT

Evania Raihan contributed to the conceptualization, methodology, data analysis, and writing of the manuscript. Dyah Lestari Widaningrum provided supervision, data collection, and manuscript editing. Both authors were involved in the research process and in the drafting and finalization of the manuscript.

REFERENCES

Abbas, R., Afolabi, R., Eleweke, I., Adesokan, A., Akinsola, A., & Elijah, L. (2025). Adopting Secure Software Development Practices to Improve Financial Transactions in the Banking Sector. *Int. J. Adv. Res. Ideas Innov. Technol*, *11*, 147–159.

Baschin, J., Huth, T., & Vietor, T. (2020). An approach for systematic planning of project management methods and project processes in product development. *2020 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 1037–1041. https://doi.org/10.1109/ICE/ITMC48013.2020.9309809

Edgar, T., Caldwell, S., & Porter, E. (2025). *Integrating Security Early in the SDLC for Banking Applications*.

Fonseca, F., Letouzé, P., Pompeu, R., Garcia, L., Regina, S., & França, G. (2017). Barriers in information technology project management. *2017 IEEE Technology & Engineering Management Conference (TEMSCON)*, 370–375. https://doi.org/10.1109/TEMSCON.2017.7998403

Islam, M. I. (2022). *Software development & operation life cycle and project management perspective*.

Johnson, G. L. (2018). *The Causal Effects of Cost, Schedule, and Quality on the Development of Geospatial Technology Using Waterfall and Agile Scrum Software Development Methods*. The George Washington University.

Mtambalika, A., Manda, T. D., Gombachika, H., & Kunyenje, G. (2016). Branchless banking in rural Malawi: Potential customers' perspective on bank-led mobile banking. *2016 IST-Africa Week Conference*, 1–11.

Muli, M. (2025). Impact of Project Management Adoption as Enabler for Successful Financial Technology Performance: A Case of Mobile Banking Application in Kenya. *Journal of the Kenya National Commission for UNESCO*. https://doi.org/10.62049/jkncu.v5i2.324

Mutuku, M., & Chege, D. B. (2025). Effects of Project Management Adoption as Enabler for Successful Financial Technology Performance: A Case of Mobile Banking Application in Kenya. *Journal of the Kenya National Commission for UNESCO*. https://doi.org/10.62049/jkncu.v5i2.309

Nyirahabineza, A. (2017). *Adoption of Agent Banking as a Financial System and its Benefits to involved Financial Stakeholders.* University of Rwanda.

Oztas, A. S., Yemen, E., & Tuzun, E. (2018). Real-Time Monitoring and Control of The SDLC Process on a Single Automation in Core Banking Applications. *International Conference on Advanced Technologies, Computer Engineering and Science (ICATCESâ€™ 18), 104â, 108*.

Pramitasari, N., Ruldeviyani, Y., & Zarkasie, I. R. (2023). Net impact implementation application development life-cycle management in banking sector. *Computer Science and Information Technologies*, *4*(2), 169–182.

Rengganis, P. N., Suhayati, M. S. M., & Sutara, B. S. B. (2024). The Application Of The Sdlc Waterfall Method In Developing An Audit Application For The Sumedang Regency Inspectorate. *Jurnal Riset Teknik Informatika*, *1*(2), 139–145.

Ruby, D. (2024). *Hybrid Agile–Waterfall Approaches in Regulated Industries: Lessons from Healthcare, Finance, and Construction*.

Salmi, E. (2025). *Operational resilience by design: implementing DORA requirements across the software development lifecycle in electronic banking*.

Semwayo, J. (2024). *Information-Technology-Aided Customer Relationship Management Strategies for Improving Financial Institutions' Business Performance*. Walden University.

Surbakti, E. E., Purwandari, B., Solichah, I., & Kumaralalita, L. (2019). Analysis of software development method selection: A case of a private financial institution. *Proceedings of the 3rd International Conference on Business and Information Management*, 168–173. https://doi.org/10.1145/3361785.3361806

Taskesenlioglu, S., Ozkan, N., & Erdogan, T. G. (2022). Identifying possible improvements of software development life cycle (sdlc) process of a bank by using process mining. *International Journal of Software Engineering and Knowledge Engineering*, *32*(04), 525–552. https://doi.org/10.1142/S0218194022400010

Tsang, C.-Y. (2017). Balancing the governance of the modern financial ecosystem: a new governance perspective and implications for market discipline. *Hous. J. Int'l L.*, *40*, 531.

Максимюк, М. (2025). Modern Banking Technologies In The Context Of Project Management. *Смарт-Економіка, Підприємництво Та Безпека*, *3*(2), 26–33. https://doi.org/10.60022/sis.3.(02).3