

ISSN 1840-4855

e-ISSN 2233-0046

Original scientific article

<http://dx.doi.org/10.70102/afts.2026.1836.142>

AGENT-BASED SPRINT MITIGATION FOR RISK-RESILIENT SCRUM EXECUTION: A CONTROLLED SIMULATION STUDY

V. Sunder¹, Dr.P. Deepalakshmi^{2*}

^{1*}Associate Professor, Department of Artificial Intelligence and Machine Learning, Ahalia School of Engineering and Technology, Palakkad, Kerala, India.

e-mail: sunderv@hotmail.com, orcid: <https://orcid.org/0009-0000-0867-5780>

^{2*}Professor, Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education, Krishnankoil, Tamil Nadu, India.

e-mail: deepa.kumar@klu.ac.in, orcid: <https://orcid.org/0000-0002-1959-3657>

Received: April 28, 2026; Revised: June 11, 2026; Accepted: July 20, 2026; Published: September 29, 2026

SUMMARY

The Scrum framework is one of the most popular Agile software development frameworks, due to its versatility, iterative development process, and focus on delivering value in small increments. However, sprint overruns continue to be a significant operational challenge and are usually caused by changes in requirements, misjudging the effort, blocker events, or productivity swings. Scrum monitoring methods currently used are mostly related to progress tracking and predictive analytics, offering little assistance in the proactive aspect when the sprint runs. This study presents a lightweight Rule-Based Agentic Mitigation Framework that actively observes deviations in sprint execution and takes pre-defined corrective actions if execution risk is above defined thresholds. The proposed framework is not dependent on large historical databases or complex computational infrastructure, as is the case for machine learning-based approaches, thus enabling the use of the framework in various Agile environments. To assess the effectiveness of the proposed framework, an experimental simulation was used. All 200 paired sprint simulations were performed under identical conditions of scope changes, blocker occurrences, and velocity variability. The following metrics were used to evaluate the performance: Sprint Overrun Rate, Sprint Success Rate, Average Risk Detection Day, and Execution Stability Index. The results of the experiments showed that the proposed framework improved the Sprint Overrun Rate from 0.830 to 0.625, which is an improvement of 24.7%. The Sprint Success Rate went from 17.0% to 37.5%, and Average Risk Detection Day went from Day 5.34 to Day 3.19, allowing earlier corrective action. Additionally, the Index of Execution Stability rose from 0.78 to 0.92, which means there was greater consistency in sprint progress. In the ablation study, it was confirmed that the totality of the velocity correction, scope adjustment, and ambiguity reduction resulted in improvement of the performance. The results indicate that agentic mitigation by rules is an effective and practical solution to increase the reliability of sprints, decrease overruns, and improve Agile project governance.

Key words: *scrum, agile software development, sprint overrun, agent-based modeling, rule-based mitigation, sprint monitoring, agile project management, simulation.*

INTRODUCTION

Agile software development has revolutionized the software industry due to its focus on flexibility, quick iterations, and value delivery through constant development. Scrum is one of the most widely-used frameworks for the management of iterative software development among Agile methodologies [11]. The idea of scrum is to break down work into small time-boxed chunks in the form of sprints that are usually 1 – 4 weeks long and are designed to bring small, incremental and potentially deployable product features to life [1]. Effective planning, execution, and monitoring of the sprint are essential for Scrum. Sprint planning, daily stand up, sprint review and sprint retrospective meetings can help keep team members and stakeholders aligned, working towards a common goal, and transparent [2]. These are common metrics like velocity, burndown chart, remaining work, etc., that are used to monitor progress of sprints and assess how likely are to meet their goals. Although it has its benefits, teams can sometimes experience sprint overruns, where teams do not manage to finish the tasks are planned during the allocated sprint time [4]. The consequences of Sprint overruns range from product releases being delayed, less confidence among stakeholders, to disruption of future sprint schedules. Typical reasons for these issues are changes in requirements, inaccurate effort estimates, unexpected technical problems, resource constraints and team productivity changes.

Traditional Scrum monitoring tools are primarily designed to give the status of the sprint instead of blocking the sprint from getting delayed [6]. Most of these tools are reactive and only come to light when there are major deviations between the workload that is left and the time available in a sprint [8]. Consequently, teams are often forced to take ad hoc corrective actions like overtime working, delaying work or scheduling it for later sprints [9][10]. Predictive analytics, machine learning and artificial intelligence methods have been the subject of recent research for enhancing sprint forecasting and risk identification [3]. But, many of these strategies are concerned with prediction instead of intervention. It also frequently used when there is a large amount of historical data from past projects, that is, when applicable in large or newly created Agile teams [5][12][13].

Agent-Based Modeling (ABM) is one of the valuable techniques that can be used to analyse complex systems by interactions between autonomous agents [7][14][15]. ABM has been used in software project management, and in studies of software team collaboration but not for real time sprint mitigation [16][17].

Key Contributions of the Research

- Proposed a rule-based mitigation framework to detect sprint deviations and apply corrective actions during sprint execution.
- Developed a simulation model to evaluate sprint performance under different project uncertainties.
- Compared the proposed approach with traditional Scrum monitoring and demonstrated its ability to improve sprint completion and reduce overruns.

The rest of this paper is organized as follows. In Section II, the authors review the latest literature on Scrum-based Agile development, sprint monitoring, predictive analytics, and AI-powered project management, pinpointing existing research gaps. The method to be used for the proposed work is presented in Section III, detailing the rule-based agentic mitigation framework, the simulation environment, the mathematical modeling and the evaluation procedure. The experimental set-up, performance metrics, comparative analysis and ablation study results are presented in Section IV to evaluate the effectiveness of the proposed approach. Key findings, practical implications and limitations of the study are discussed in Section V. Lastly, Section VI summarizes the key contributions and results and suggests future research directions for this paper.

LITERATURE REVIEW

Pérez Castillo et al. [18] suggested a machine learning-based method for assessing the velocity of sprints and team velocity in Agile projects. The study consisted of creating a dataset based on information from the burndown chart and using machine-learning algorithms to monitor sprint performance. Their findings showed that by leveraging data-driven monitoring, the visibility of sprint execution can be

enhanced, and project managers can make informed decisions on the trends in the sprint's performance. The framework, however, is mainly about monitoring and prediction, without automatic mitigation mechanisms during the execution of the sprint.

Forouzesh Nejad et al. [19] proposed an explainable artificial intelligence (XAI) framework on predictive modeling in Agile software projects. The research integrated predictive analytics with methods to ensure transparency in determining the status of projects and their decisions regarding them. The authors found that explainable models helped build trust with the stakeholders and facilitated understanding of project risks. However, the framework is still a predictive model and does not provide any detailed procedures to deal with deviations from the sprint whilst running it.

Azonuche and Enyejo [20] studied machine learning based dynamic backlog prioritization and risk mitigation for AI powered sprint planning optimization. how supervised and unsupervised learning techniques can enhance the accuracy of sprint planning, allocation of resources, and consistency of delivery. The study found that the use of AI can greatly improve planning efficiency, but the suggested approaches primarily benefit the pre-sprint planning phase and offer limited assistance in adjusting to unforeseen events during sprint execution.

Als Salman and Chyad [21] carried out a systematic mapping study of Agile management of Machine Learning Systems. The authors reviewed and identified the salient characteristics of the Agile frameworks, including the flexibility in iterations, adaptive planning, and feedback mechanisms. They concluded that Agile methods can be used to manage ML projects but that there are some difficulties with effort estimation and project governance. The study also highlighted the importance of having more robust mechanisms for supporting implementation at the execution stage and empirical evidence.

To improve the Scrum Master's awareness of his emotions in Agile meetings, Usman et al. [22] suggested a multimodal emotional Artificial Intelligence system. The system's structure involves speech analysis, sentiment detection, and context-based suggestions to deliver instant feedback on emotions. Experimental evaluation resulted in improvements in the quality of communication effectiveness and quality of team interaction. This approach works well for improving collaboration in Scrum, but is not directly linked to mitigating sprint performance or providing governance for sprints.

Verwijs and Russo [23] did empirical research that compared the effectiveness of popular Agile scaling methods used by several software development teams. The research explored team performance, team collaboration and project outcomes within massive Agile settings. The study demonstrated that while Agile scaling frameworks help to align the organization and coordinate better, there are still some issues with keeping the execution efficient and the sprints on time. The high importance of governance mechanisms for managing risks at sprints and execution variability was emphasized [17]. The approaches proposed however are geared towards organizational effectiveness and not towards having mitigation measures during the sprint execution.

Fawzy et al. [6] conducted a systematic literature review and a practitioner survey to probe the data management issues and the possible solutions in Agile software development. This research examined data-related challenges like data integration, quality assurance, data collection, and real-time data analysis in Agile environments. The results showed that Agile teams have often struggled with consistency and information from various sources in the process of iterative development. To enhance project visibility and decision making, the authors suggested a number of solutions, such as automated data collection mechanisms, and cloud-based management platforms. The study, however, is mainly concerned with data management and lacks a specific model for real-time sprint mitigation or for sprint intervention during execution.

As seen in table 1, the primary focus of recent Agile–AI research is on predicting, prioritizing backlog, tracking, or coordinating the team. While these strategies increase visibility and planning accuracy, it can offer limited assistance in structured corrective action while performing sprints. The proposed work aims to fill this void by proposing a lightweight rule-based mitigation mechanism which works directly during the execution of the sprint and is tested through simulations in a controlled paired environment.

Table 1. Comparative summary of recent agile–AI literature

Authors	Methodology	Dataset / Simulation Environment	Research Gap
Pérez Castillo et al. [18]	Machine-Learning-Based Sprint Monitoring	Burndown-Chart and Velocity Data from Agile Projects	Focuses on Monitoring and Prediction; no Runtime Mitigation.
ForouzeshNejad et al. [19]	Explainable AI (XAI) Predictive Framework	Agile project forecasting datasets	Improves transparency but lacks intervention mechanisms.
Azonuche & Enyejo [20]	AI-Assisted Sprint Planning and Prioritization	Historical Sprint and Backlog Data	Supports Pre-Sprint Decisions Only.
Als Salman and Chyad et al. [21]	Systematic Mapping Study	Published Agile/ML literature	Identifies Practices, Not Execution-Stage Controls
Usman [22]	Multimodal emotional AI framework	Meeting audio, sentiment, and interaction data	Targets collaboration quality, not sprint mitigation.
Verwijs & Russo [23]	Empirical Comparison of Agile Scaling Approaches	Multiple Software Development Teams	Addresses Organizational Effectiveness Rather Than Real-Time Sprint Control
Fawzy et al. [6]	Systematic Review + Practitioner Survey	Agile Data-Management Studies and Industry Responses	Focuses on Data Management, Not Deviation Mitigation.
Proposed Study	Rule-Based Agentic Mitigation	Controlled Paired Sprint Simulations (200 runs)	Addresses Proactive Execution-Stage Mitigation Without Large Historical Datasets

METHODOLOGY

Research Design

The study is conducted using a simulation approach to test the effectiveness of the rule based agentic research framework for the execution of Scrum sprints. This research is about solving sprint overruns with the introduction of corrective actions based on the extent of the execution deviations from the predefined sprint rules. The proposed framework takes a proactive approach in addressing the issues that happen in the implementation of the sprints, whereas predictive approaches depend on historical project information and machine learning algorithms. A simulation environment was created to simulate a traditional Scrum monitoring and the proposed mitigation-enabled Scrum monitoring in the same context and analyze the results.

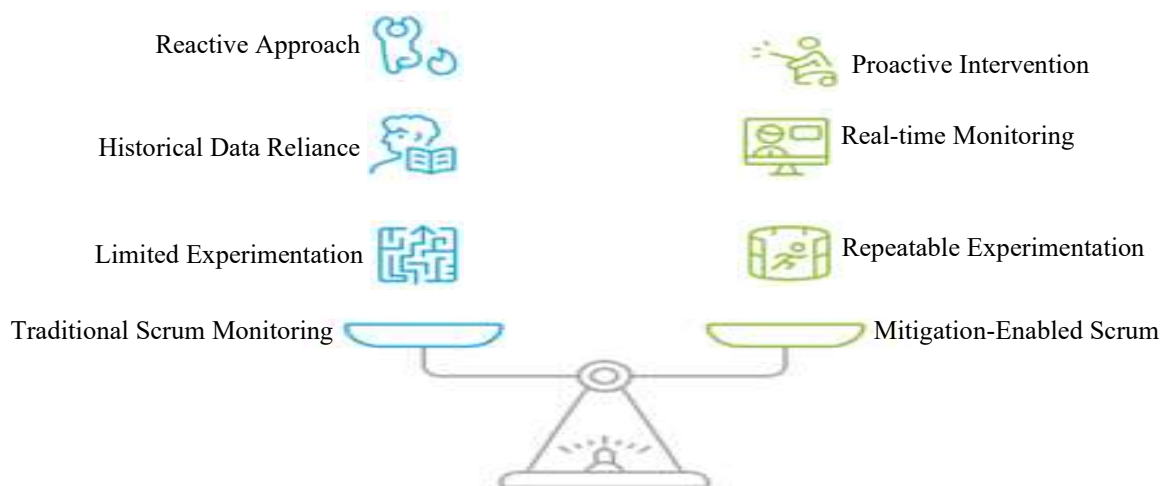


Figure 1. Conceptual comparison between traditional scrum monitoring and mitigation-enabled scrum framework

The study is conducted using a comparative evaluation approach and both models are fed the same set of sprint parameters, such as effort planning, velocity variations, blocker events, and scope changes. The choice of simulation-based design was made so that the experiment can be repeated and the factors that could cause uncertainty can be manipulated in a controlled manner and can be difficult to isolate in real Agile projects. This methodology allows for objective evaluation of the framework's effectiveness to enhance sprint completion performance while ensuring internal and repeatability of results.

The conceptual difference between the traditional Scrum monitoring approach and the proposed mitigation-enabled Scrum framework is outlined in figure 1. The traditional way Scrum monitoring is done is mostly reactive, meaning issues are only detected once grown too large and significant when executing sprints. It also employs a strong historical project data base for decision making, and provides less room for controlled experiments and proactive risk management. The rationale behind this is that the proposed mitigation enabled Scrum framework focuses on proactively intervening by continually monitoring sprint activities and detecting deviations from the sprint execution at the time. The framework enables timely corrective actions and helps to prevent sprint overruns, which helps the project to be more predictable. Moreover, the simulated environment allows for replicable experimentation, with various sprint scenarios being tested in a controlled setting. The balancing scale of the figure emphasizes the shift from traditional monitoring to intervention to improve sprint stability, adaptability and project governance for an Agile project.

Sprint Simulation Environment

To capture the dynamic and uncertain nature of software development with Scrum, a sprint simulation environment was created. Sprint were simulated as 10-day iterations with the story points ranging from 45 to 60 per sprint. The planned effort was used to determine the expected daily velocity, and stochastic variability was added by normal random fluctuations of velocity. Uncertainty factors like blocker events, requirement changes and task ambiguity were added to simulate realistic sprint scenarios. The occurrence rates for blocker events were modeled probabilistically, and the scope changes were modeled as the events evolve over time and are often seen in Agile projects. Actual work completion was monitored for each individual via the simulation and tracked against planned expectations. The assumptions were made to provide an understanding of the key elements of a sprint without making it overly complex. The simulation environment includes several sources of variability, making it a realistic but controlled setting to test the effectiveness of the mitigation strategy in various sprint conditions and execution scenarios.

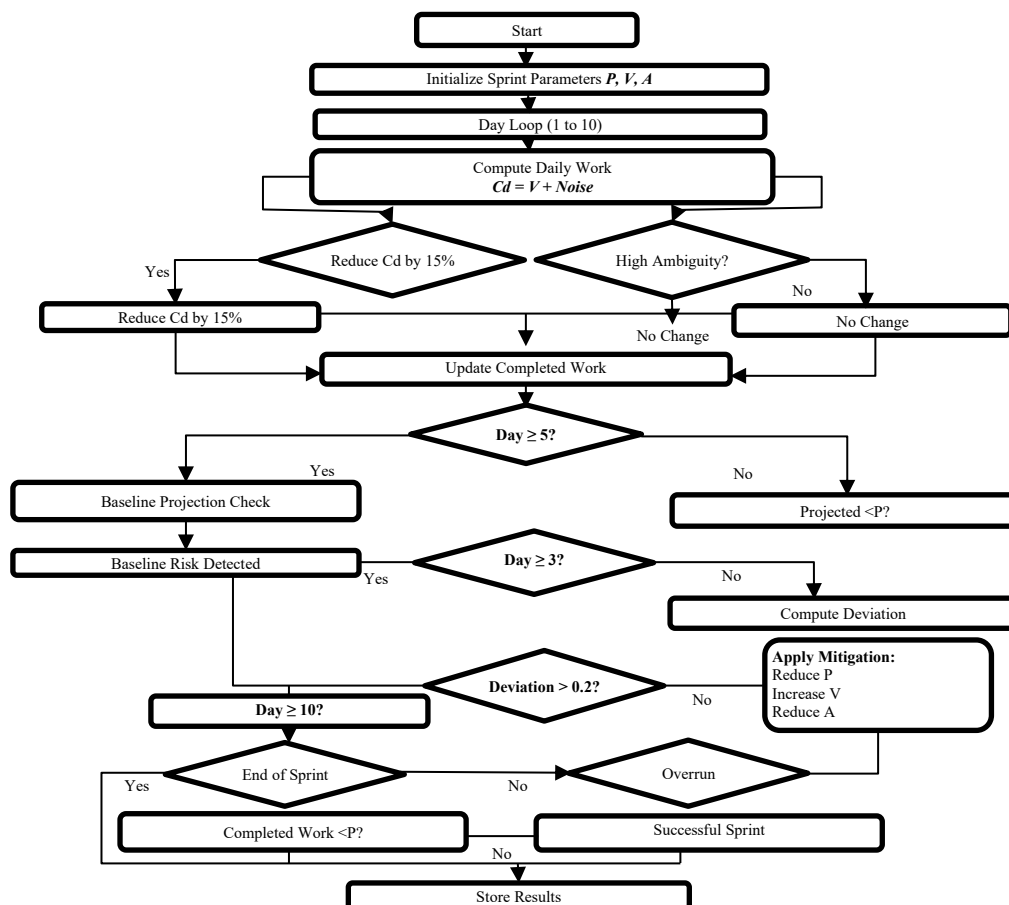


Figure 2. Flowchart of the proposed rule-based agentic sprint mitigation framework

The operational flow of the proposed Rule Based Agentic Mitigation for Scrum Sprint is presented in figure 2. It starts by setting certain sprint parameters, e.g. planned effort (P), expected velocity (V) and ambiguity level (A). The framework calculates the completed work for each sprint day based on the velocity, and execution variability. Monitoring of ambiguity and execution conditions is updated and the completed work is then updated. The framework is continually tested for progress throughout the middle stages of the sprint and for deviation analysis throughout the sprint from the middle stages. If the deviation is greater than the set threshold, mitigation measures are automatically activated, such as a smaller planned scope, a higher effective velocity and lower levels of ambiguity. These corrective measures are designed to reduce execution drift and increase the probability of completing sprints. The framework stays with the sprint through the end of the Sprint and compares the work done to the effort planned to determine whether or not the Sprint was successful or if it overran. Lastly, the results of all executions are recorded for performance analysis. The flowchart illustrates the value of continuous monitoring and the application of rules for proactive sprint management, which leads to better Agile project governance.

Comparative Evaluation Procedure

This evaluation was done with a paired comparative experimental design with 200 simulated sprint instances. The same initial conditions were created and run twice per sprint: Baseline Scrum monitoring and Agent-based mitigation. The baseline model was a conventional Scrum practice, and risk detection was done based on mid-sprint projections with no corrective action taken. The proposed framework, on the other hand, continuously surveyed the deviations from the execution and triggered the mitigation measures if the set thresholds were exceeded. These interventions were scope adjustment, ambiguity reduction and velocity correction. The performance outcomes were taken for both models separately. The metrics of evaluation included were sprint overrun rate, risk detection timing, execution stability, mitigation activation frequency and sensitivity to variability. Paired comparisons were used in statistical analysis to see if there were significant improvements that were observed. This evaluation strategy helped to provide a level playing field between the two approaches and allowed for practical evaluation of the effect of rule-based mitigation on sprint execution performance.

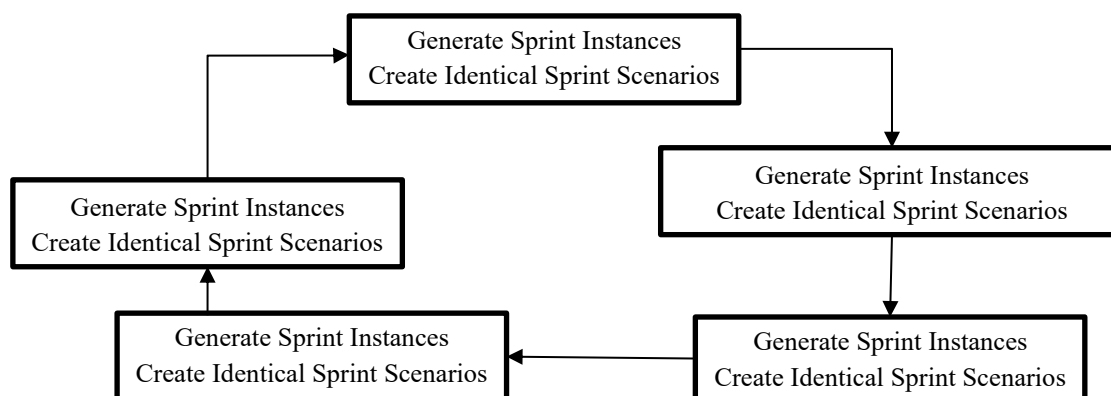


Figure 3. Workflow of the simulation-based comparative evaluation procedure

In figure 3, the workflow can be used for simulating the proposed agent-based mitigation framework. The first step is to create a set of identical sprints so that the conditions under which the experiments are conducted are equal and consistent for both the baseline Scrum model and the model that includes mitigations. These sprint scenarios are then performed under known levels of uncertainty including velocity changes, blocker and scope changes. The performance data such as the sprint completion status, the timing of risk detection and work progression is recorded continuously during execution. Next, the data are analyzed comparatively and statistically to assess the difference between the two approaches. Last, the effects of the mitigation framework are investigated through the lens of improvements to key performance metrics like sprint overrun rate, execution stability and sprint success rate. The repeated experimental workflow emphasizes the repeatability of the experimental procedure and the ability to assess the proposed framework in various sprint environments, thus guaranteeing reliable and reproducible evaluations.

Daily Work Completion

$$Cd = V + \epsilon. \quad (1)$$

Equation 1 represents the amount of work completed during a particular sprint day. The completed work is calculated by combining the expected team velocity with a random variation factor that captures uncertainties such as productivity fluctuations, minor blockers, and execution variability. This equation helps simulate realistic sprint behavior by reflecting the fact that daily productivity is rarely constant throughout the sprint lifecycle.

Sprint Deviation Metric

$$D = \frac{|V - Cd|}{V} \quad (2)$$

Equation 2 measures the deviation between the expected velocity and the actual work completed during a sprint day. The deviation value is normalized by the expected velocity, allowing consistent comparison across different sprint conditions. When the deviation exceeds a predefined threshold, the agent-based mitigation mechanism is activated to apply corrective actions such as scope adjustment, ambiguity reduction, or velocity correction. This metric serves as the primary trigger for the proposed mitigation framework.

Algorithm 1: Rule-Based Sprint Mitigation

Input:

Planned Effort P

Expected Velocity V

Sprint Duration D = 10 days

Output:

Sprint Completion Status

Begin

1. Initialize *completed_work* = 0

2. For *day* = 1 to *D* do

Calculate daily work:

$Cd = V + \text{random_variation}$

Update completed_work:

$\text{completed_work} = \text{completed_work} + Cd$

Compute deviation:

$Dev = |V - Cd| / V$

 If $Dev > 0.20$ then

Reduce scope by 10%

Increase velocity by 5%

Reduce ambiguity by 10%

```

    End If
  End For
3. If completed_work  $\geq$  P then
    Sprint = Successful
Else
    Sprint = Overrun
End If
End

```

Algorithm 1 is proposed sprint mitigation process, based on rules. The effort planned and velocity expected are reset at the start of each sprint. Each day of the sprint the work done is computed with execution variability. A deviation is continually tracked with the deviation metric as a measure of the difference between the progress made and progress expected. If the deviation is above the set deviation limit (0.20), then corrective actions of scope reduction, velocity adjustment, and ambiguity reduction are automatically taken. These are interventions designed to ensure a more stable sprint and reduce the risk of sprint overruns. The total amount of work completed in the sprint is compared against the work planned to see if the sprint was completed or overrun. This lightweight algorithm allows for proactive sprint governance without the need for complicated machine learning models or large historical datasets.

EXPERIMENTAL RESULTS

Experimental Setup, Dataset, and Parameter Initialization

A controlled simulation environment, which models the sprint execution of Scrum in uncertain conditions as illustrated in table 2, was used in the experiment. Large-scale industrial sprint data sets were not found, so a synthetic sprint data set was created in order to simulate realistic Agile development scenarios. Typical sprint variability factors such as velocity fluctuations, presence of blockers, requirement changes, and task ambiguity were included in the simulation environment. A total of 200 instances of sprints were created, where each sprint was run under two different scenarios: (i) conventional Scrum monitoring (baseline model) and (ii) the proposed rule-based agentic mitigation model. The initial sprint conditions were the same in both models for a fair comparison. Each sprint was represented with an iteration period of 10 days, and the effort value for each sprint was randomly selected from 45 to 60 story points. The duration of the sprints was used to calculate the expected daily velocity by dividing the planned effort by the sprint duration.

Table 2. Experimental setup, dataset, and model parameter initialization

Parameter	Symbol	Value/Range
Number of Sprint Simulations	N	200
Sprint Duration	D	10 Days
Planned Story Points	P	45–60
Base Daily Velocity	V	P/10
Ambiguity Index	A	0.3–0.6
Velocity Noise	ε	N(0, 0.5)
Blocker Probability	pb	0.05
Scope Change Probability	ps	0.1
Deviation Threshold	DT	0.2
Scope Reduction	SR	10%
Velocity Adjustment	VA	5%
Ambiguity Reduction	AR	10%

The blocker events and scope changes were created with predefined probabilities, and daily execution variability was added as normally distributed random noise to the data. The mitigation model was

constantly checking the deviations from the sprint and responding with appropriate mitigations when the deviation was more than the required limit. The simulation was integrated within a standard numerical computing environment, and all the parameters were kept constant during the experiments to ensure the repeatability and reproducibility of the results.

4.1.1 Performance Metrics

To evaluate the effectiveness of the proposed rule-based agentic mitigation framework, several performance metrics were considered. These metrics provide a comprehensive assessment of sprint execution quality, mitigation effectiveness, and overall project performance. The selected metrics focus on both outcome-based and process-based evaluation aspects, enabling a detailed comparison between the baseline Scrum monitoring model and the proposed mitigation-enabled framework.

Sprint Overrun Rate (SOR)

In equation 3 shows Sprint Overrun Rate measures the proportion of sprints that fail to complete the planned work within the allocated sprint duration. It is the primary performance indicator used in this study. A lower overrun rate indicates better sprint execution and improved project predictability.

$$SOR = \frac{N_{overrun}}{N_{total}} \quad (3)$$

Average Risk Detection Day (ARD)

This equation 4 determines the average day within a sprint on which a potential execution risk or deviation is detected. Earlier detection allows more time for corrective actions and can improve sprint outcomes.

$$ARD = \frac{\sum_{i=1}^N Di}{N} \quad (4)$$

Execution Stability Index (ESI).

The equation 5. Execution Stability Index evaluates the consistency of work completion throughout the sprint. Higher stability indicates smoother sprint progress and reduced fluctuations in daily productivity.

$$EI = 1 - \frac{\sigma C}{\mu C} \quad (5)$$

Mitigation Activation Rate (MAR).

This equation 6 metric measures how frequently the mitigation mechanism is triggered during sprint execution. It reflects the responsiveness of the proposed framework to sprint deviations.

$$MAR = \frac{N_{mitigation}}{N_{total}} \quad (6)$$

The above metrics collectively evaluate different dimensions of sprint performance. Sprint Overrun Rate assesses overall success in meeting sprint goals, while Average Risk Detection Day measures the timeliness of risk identification. The Execution Stability Index captures the consistency of sprint progress, and the Mitigation Activation Rate reflects the operational behavior of the proposed framework. Together, these metrics provide a robust basis for comparing conventional Scrum monitoring with the proposed rule-based mitigation approach under simulated Agile development conditions.

Overrun Rate and Detection Performance Analysis

This section looks at the success of the proposed Rule Based Agentic Mitigation Framework at reducing sprint overruns and enhance the timeliness of risk detection. Twenty sets of 200 paired sprint simulations were conducted under the same conditions, with both the existing Scrum monitoring model and the new mitigating model. Key metrics analyzed include Sprint Overrun Rate, Sprint Success Rate, Average Risk Detection Day, and intervention effectiveness. These metrics help to understand the effectiveness of the framework to detect deviations in execution at an early stage and take appropriate corrective actions in a timely manner, thus preventing major schedule slippages. This study will compare the two methods and see if proactive mitigation can enhance the reliability of sprints, increase the number of sprints completed, and improve overall project governance in the Agile project in the face of uncertainty in its execution.

Table 3. Detailed comparison of overrun and detection performance

Metric	Baseline Model	Agent-Based Model	Improvement (%)
Sprint Overrun Rate	0.83	0.625	24.7
Sprint Success Rate	0.17	0.375	120.59
Average Risk Detection Day	5.34	3.19	40.26
Mean Remaining Work at Detection	24.8 SP	33.7 SP	35.89
Average Delay Beyond Sprint End (Days)	1.82	0.94	48.35
Early Intervention Frequency	0	0.68	–
Paired t-statistic	–	7.273	–
p-value	–	< 0.001	Statistically Significant

In this table 3 shows the performance results of the proposed Agent-Based Mitigation Framework in improving the sprint execution performance. In the baseline model, the Sprint Overrun Rate was 0.83, which decreased to 0.625, representing a 24.7% improvement. Similarly, the Sprint Success Rate increased from 0.17 (17.0%) to 0.375 (37.5%), corresponding to a 120.59% relative improvement. The framework also allowed for risks to be identified earlier in the execution process, minimizing the Average Risk Detection Day from 5.34 to 3.19, an improvement of 40.26%. This meant that there was more work to do at the time of the detection, so the Mean Remaining Work at Detection would increase from 24.8 to 33.7 story points. Average Delay Beyond Sprint Completion has been decreased to 0.94 days from 1.82 days, which reflects better adherence to schedule. Moreover, the occurrence of the activation of the mitigation mechanism was 68% in the simulated sprints, showing proactivity in response. There was a statistically significant difference in observed improvement in performance, as determined by statistical validation using a paired t-test, with a t statistic of 7.273, and a p value of < 0.001, which would not be expected by random variation. The findings of this study, overall, suggest that the proposed framework does indeed reduce sprint overruns, it improves risk management, and it increases the reliability of the completion of sprints.

Execution Stability and Work Progression Analysis

The work completion and the stability of execution were examined for both models to investigate execution behavior throughout the sprint lifecycle.

Table 4. Detailed execution stability and work progression analysis

Metric	Baseline Model	Agent-Based Model	Improvement (%)
Average Completed Story Points	41.3	48.6	17.68
Mean Daily Velocity	4.13	4.86	17.68
Velocity Standard Deviation	1.84	1.12	39.13
Execution Stability Index	0.78	0.92	17.95
Work Completion Consistency	Moderate	High	–
Response to Blockers	Reactive	Adaptive	–
Scope Change Recovery Rate	0.44	0.79	79.55
Productivity Recovery Capability	0.48	0.84	75
Cumulative Work Completion (Day 10)	82.60%	97.20%	17.68

The goal is to see if the proposed mitigation plan can keep the work progression in the project going a bit smoother and more predictable even in the face of uncertainty factors like blockers, scope changes, and productivity swings.

In this table 4 and figure 4 illustrate the stability of the executions of the proposed Agent-Based Mitigation Framework and the baseline model, respectively, suggesting that there was a significant improvement in the sprint performance of the former. The Average Completed Story Points rose from 41.3 to 48.6 (+17.68%) and the Mean Daily Velocity improved from 4.13 to 4.86 (+17.68%). The Velocity Standard Deviation went from 1.84 to 1.12, which is a 39.13% decrease in variability of execution and more consistent daily progress. Likewise, the Execution Stability Index rose from 0.78 to 0.92, which showed the smoother execution of sprints during the project lifecycle. The proposed framework also showed its increased adaptability with an adaptive response to blocker events, while the baseline model was still largely reactive. Moreover, the framework's capability to manage uncertainties and restore performance after disruptions significantly improved, as demonstrated by improvements in the Scope Change Recovery Rate (from 0.44 to 0.79) and the Productivity Recovery Capability (from 0.48 to 0.84). This resulted in the cumulative work completed at the end of the sprint rising from 82.60% to 97.20%, meaning that most planned work activities were completed during the sprint time. The results validate that the proposed mitigation approach leads to execution stability, better recovery from unexpected events and more reliable sprint delivery.

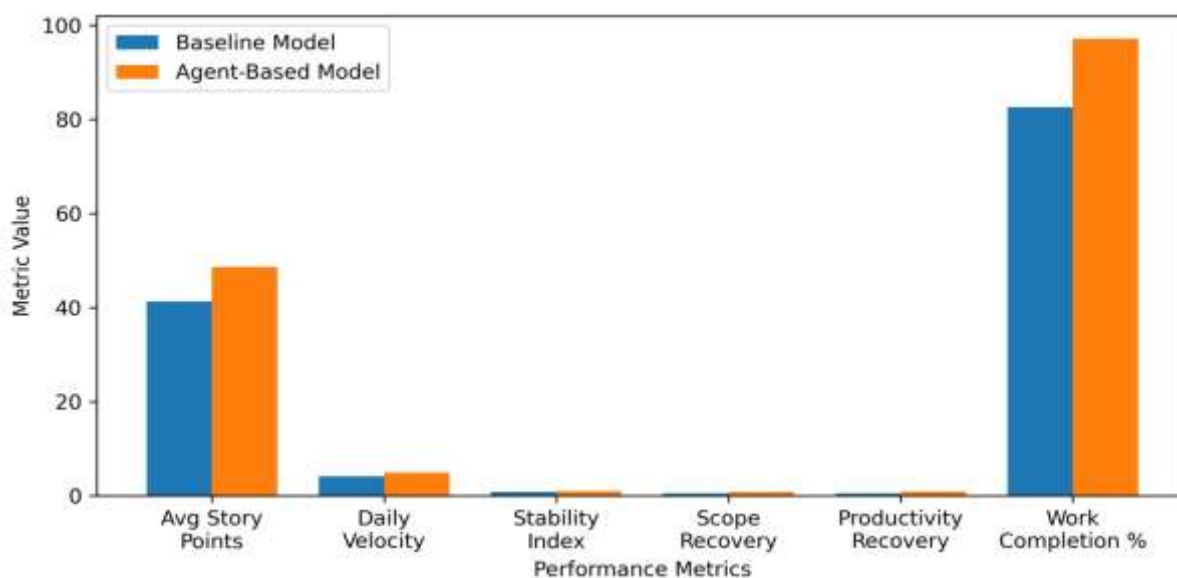


Figure 4. Execution stability and work progression analysis

Ablation Study

The goal of the ablation study is to explore the effect of each mitigation component of the proposed Agent Based Mitigation Framework. The overall framework includes Scope Adjustment, Velocity Correction and Ambiguity Reduction mechanisms; however, it is crucial to understand the contribution of each of these components to sprint performance in isolation. This was done by running a set of simulation experiments to isolate the contribution of each mitigation component, while holding all other experimental parameters fixed. The outcome of the performance was then compared to a baseline Scrum monitoring model and also the complete mitigation framework. The key evaluation metrics analyzed were the Sprint Overrun Rate, Sprint Success Rate, Execution Stability Index and Risk Detection Day. This evaluation will help to develop a better understanding of the relative importance of the various mitigation measures, as well as what parts of the measures are most critical in terms of speed reliability and speed execution stability. The results provide a validation of the proposed framework, and illustrate the need for a combination of multiple mitigation mechanisms to optimize sprint performance.

The results of the ablation study to assess the impact of each individual mitigation component in the Agent Based Mitigation Framework are shown in figure 5. The overall system that includes ALL the types of mechanisms listed above (Ambiguity Reduction, Scope Adjustment and Velocity Correction) yielded the lowest Sprint Overrun Rate of 0.625. In contrast, the baseline approach to Scrum monitoring had consistently the highest overrun rate of 0.830, which means that it was not well designed for proactively managing deviations in execution. Removing Velocity Correction raised the overrun percentage to 0.740, the highest performance penalty of all the configurations and its importance in keeping sprint productivity. Likewise, the overrun rate was raised to 0.710 by excluding Scope Adjustment, and was raised to 0.690 by excluding Ambiguity Reduction. The results showed that all mitigation factors have a beneficial effect on sprint performance and when used collectively, have the greatest impact on reducing sprint overruns and increasing overall execution stability.

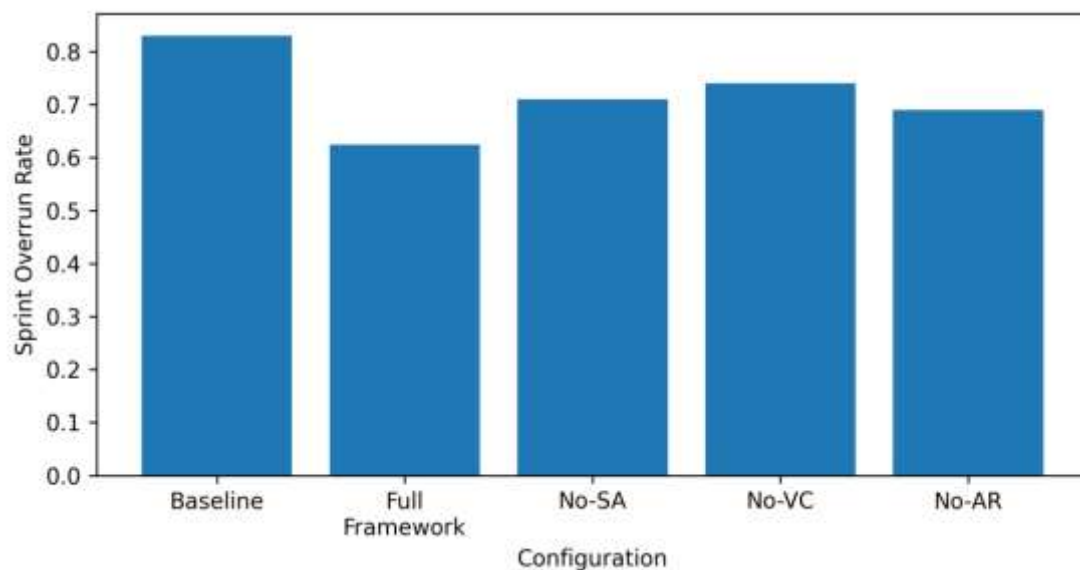


Figure 5. Ablation analysis of individual mitigation components based on sprint overrun rate

DISCUSSION

The experimental findings show that the proposed rule-based agentic mitigation framework helps improve the sprint execution performance relative to the traditional Scrum monitoring approaches. This decrease of sprint overrun rate from 0.83 to 0.625 suggests that proactive interventions can significantly increase the reliability of completing the sprint. The proposed approach differs from the traditional Scrum monitoring approach by adding structured corrective action steps to the Scrum Sprint. This helps the system to react to deviations in execution before turn into major sprint failures.

The key finding of the study is the early deviation detection. The average was Day 3.19 for risks of execution compared to Day 5.34 in the baseline model. The earlier identification of the scope allows for more time to implement corrective actions like scope adjustment, ambiguity reduction, and velocity correction. Thus, the framework was able to effectively reduce execution drift and enhance chances of meeting the intended sprint goals within the time bound.

The execution stability analysis also shows the advantages of the proposed approach. The agent-based model had higher execution stability index and smoother cumulative work progression during the sprint lifecycle. This means that the impact of uncertainty factors such as productivity fluctuations, blocker events and requirement changes is absorbed by this mitigation mechanism. The framework reduces the variability of how work gets done on a day-to-day basis, thus providing a more predictable execution environment and enhancing the governance of the sprint.

The ablation study gives further details on individual mitigation components. The most significant factor was velocity correction, followed by scope adjustment and ambiguity reduction. When any one of these

components was deleted, performance suffered, thus indicating that the effectiveness of the framework is a function of the operation of the combination of mitigation components. The results indicated that sprint overruns may be the result of multiple factors, rather than a single factor that deviates from the norm.

The proposed framework has a number of practical advantages over machine learning and predictive analytics approaches as reported in recent writings in the Agile literature. To run most predictive systems, a lot of past project information is needed, training is extensive and the computational demands are large. The proposed rule-based framework, on the other hand, does not rely on past data and can be implemented with simple decision rules. This is especially advantageous for teams of small to medium size that do not have much project information in the past, or historical project information may be missing.

It is also significant that the framework is very practical. The mitigation measures taken in the study are typical Agile management practices, including backlog reprioritization, clarifying requirements, and working with the team to coordinate its efforts. This means that the framework can be adopted into the existing Scrum environment and project management tools with minimal changes to the process. This feature makes it easier to use it in the real world for software development.

Although the results are promising, there are a number of points to make which should be noted as limitations. A controlled simulation environment based on simplified assumptions and stochastic variability models was used for the study, to simulate the behavior of sprints. The simulation also models important aspects of Scrum but there are other organizational, technical and human factors that are not necessarily easy to accurately model for real software projects. Moreover, the mitigation rules were fixed and rigid during all the experiments, while in the real-world Agile contexts, the choices of mitigation strategies could be a variant depending on the project context.

CONCLUSION AND FUTURE WORK

This research proposed a rule-based agentic mitigation approach to tackle sprint overruns in the context of a Scrum-based agile software development environment. A controlled simulation experiment was performed with 200 paired sprint executions to compare the proposed framework with traditional Scrum monitoring. The results showed that the proposed approach achieved a Sprint Overrun Rate of 0.625, which is about 24.7% less than 0.830. Furthermore, the average Risk Detection Day went from Day 5.34 to Day 3.19, allowing for earlier corrective intervention. It also had a better Execution Stability Index of 0.92 than the baseline (0.78) and a higher Sprint Success Rate of 37.5% than the baseline (17.0%). The ablation analysis also showed that using velocity correction, scope adjustment, and ambiguity reduction together led to better sprint performance. The overall results suggest that light manipulation-based intervention can successfully improve sprint reliability, stability in sprint execution, and the predictability of the project without the need for complex predictive models or large historical databases. Future studies will be conducted to validate the proposed framework with real-life Agile project data sets and industrial Scrum contexts. The current study was conducted in a controlled simulation environment, and practical implementation will further give insights into organizational, technical, and human factors that influence the execution of a sprint. Adaptive mitigation policies, dynamic threshold optimization, and reinforcement learning-based intervention mechanisms that can automatically determine the best corrective action may be some of the additional improvements. It may also be worth exploring integration with existing Agile project management tools like Jira and Azure DevOps for real-time monitoring and automation support to mitigate. The enhancements should further increase the scalability, adaptability, and applicability of the framework in various software development environments.

REFERENCES

- [1] Vestues K, Hanssen GK, Mikalsen M, Buan TA, Conboy K. Agile data management in NAV: a case study. In International Conference on Agile Software Development 2022 Jun 9 (pp. 220-235). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-08169-9_14

- [2] Upender B. Staying agile in government software projects. In Agile Development Conference (ADC'05) 2005 Jul 24 (pp. 153-159). IEEE. <https://doi.org/10.1109/ADC.2005.41>
- [3] Chołda P, Jaglarz P. Optimization/simulation-based risk mitigation in resilient green communication networks. Journal of Network and Computer Applications. 2016 Jan 1; 59:134-57. <https://doi.org/10.1016/j.jnca.2015.07.009>
- [4] Segundo C, Oliveira M, Gonçalves E. An Agent-Based Simulation for Task Allocation in Software Development Teams based on the Truck Factor Metric. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal. 2025 Nov 17;14: e32600.<https://doi.org/10.14201/adcaij.32600>
- [5] Matthies C. Playing with your project data in scrum retrospectives. In Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Companion Proceedings 2020 Jun 27 (pp. 113-115).<https://doi.org/10.1145/3377812.3382164>
- [6] Fawzy A, Tahir A, Galster M, Liang P. Exploring data management challenges and solutions in agile software development: a literature review and practitioner survey. Empirical Software Engineering. 2025 May;30(3):77.<https://doi.org/10.1007/s10664-025-10630-4>
- [7] Rieger CG, Villez K. Resilient control system execution agent (ReCoSEA). In 2012 5th International Symposium on Resilient Control Systems 2012 Aug 14 (pp. 143-148). IEEE. <https://doi.org/10.1109/isrcs.2012.6309308>
- [8] Popli R, Chauhan N. A sprint-point based estimation technique in Scrum. In 2013 international conference on information systems and computer networks 2013 Mar 9 (pp. 98-103). IEEE. <https://doi.org/10.1109/iciscon.2013.6524182>
- [9] Lalith M. Agent-Based Simulation for Earthquake Disaster Mitigation. In ASEAN School on High-Performance Computing and Artificial Intelligence 2025 Mar 10 (p. 1). <https://doi.org/10.1145/3736273.3736300>
- [10] Radhakrishnan U, Koumaditis K. Teaching scrum with a virtual sprint simulation: Initial design and considerations. In Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology 2020 Nov 1 (pp. 1-2).<https://doi.org/10.1145/3385956.3422107>
- [11] Qureshi JN, Farooq MS, Khelifi A, Atal Z. Harnessing the potential of blockchain in ChainAgilePlus framework for the improvement of distributed scrum of scrums Agile Software Development. IEEE Access. 2024 Jul 11; 12:105724-43.<https://doi.org/10.1109/access.2024.3426597>
- [12] Fagarasan C, Cristea C, Cristea M, Popa O, Pisla A. Integrating sustainability metrics into project and portfolio performance assessment in agile software development: a data-driven scoring model. Sustainability. 2023 Aug 31;15(17):13139.<https://doi.org/10.3390/su151713139>
- [13] Dautov R, Husom EJ, Gonidis F. Towards mlops in mobile development with a plug-in architecture for data analytics. In 2022 6th International Conference on Computer, Software and Modeling (ICCSM) 2022 Jul 21 (pp. 22-27). IEEE. <https://doi.org/10.1109/ICCSM57214.2022.00011>
- [14] Canedo ED, Calazans AT, Bandeira IN, Costa PH, Masson ET. Guidelines adopted by agile teams in privacy requirements elicitation after the Brazilian general data protection law (LGPD) implementation. Requirements Engineering. 2022 Dec;27(4):545-67. <https://doi.org/10.1007/s00766-022-00391-7>
- [15] Barbala A, Sporseem T, Stray V. Data-driven development in public sector: How agile product teams maneuver data privacy regulations. In International Conference on Agile Software Development 2023 May 20 (pp. 165-180). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-33976-9_11
- [16] Ampatzoglou A, Bibi S, Avgeriou P, Chatzigeorgiou A. Guidelines for managing threats to validity of secondary studies in software engineering. In Contemporary empirical methods in software engineering 2020 Aug 28 (pp. 415-441). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-32489-6_15
- [17] Chołda P, Guzik P, Rusek K. Risk mitigation in resilient networks. In 2014 6th International Workshop on Reliable Networks Design and Modeling (RNDM) 2014 Nov 17 (pp. 23-30). IEEE. <https://doi.org/10.1109/rndm.2014.7014927>
- [18] Pérez Castillo YJ, Orantes Jiménez SD, Letelier Torres PO. Sprint management in agile approach: progress and velocity evaluation applying machine learning. Information. 2024 Nov 12;15(11):726. <https://doi.org/10.3390/info15110726>
- [19] ForouzeshNejad AA, Arabikhan F, Gegov A, Jafari R, Ichtev A. Data-driven predictive modelling of agile projects using explainable artificial intelligence. Electronics. 2025 Jun 27;14(13):2609. <https://doi.org/10.3390/electronics14132609>
- [20] Azonuche TI, Enyejo JO. Exploring AI-powered sprint planning optimization using machine learning for dynamic backlog prioritization and risk mitigation. International Journal of Scientific Research in Modern Science and Technology. 2024 Aug 28;3(8):40-57.<https://doi.org/10.38124/ijsrmt.v3i8.448>
- [21] Leeuw RT, Joseph N. Reciprocal influence between digital emotional intelligence and agile mindset in an agile environment. Administrative Sciences. 2023 Oct 25;13(11):228. <https://doi.org/10.3390/admsci13110228>

- [22] Usman M, Mendes E, Weidt F, Britto R. Effort estimation in agile software development: a systematic literature review. In Proceedings of the 10th international conference on predictive models in software engineering 2014 Sep 17 (pp. 82-91). <https://doi.org/10.1145/2639490.2639503>
- [23] Verwijs C, Russo D. Do Agile scaling approaches make a difference? an empirical comparison of team effectiveness across popular scaling approaches. Empirical software engineering. 2024 Jul;29(4):75. <https://doi.org/10.1007/s10664-024-10481-5>