iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

A THRESHOLD-BASED MODEL FOR RISK ESCALATION AND TIMEBOXING EFFICIENCY IN AGILE SPRINTS FOR SMALL-SCALE TEAMS

Jennifer B. Maglinte
https://orcid.org/0009-0000-0759-9771
ORCID: 0009-0000-0759-9771
Iligan Computer Institute, Inc., - Kapatagan, Kapatagan, Lanao del Norte, Philippines

Hex Allain J. Guirigay
Department of Education, Division of Zamboanga del Sur Pagadian City, Philippines

Dr. Paquito G. Fernando Jr
School of Graduate Studies, Northwestern Mindanao State College of Science and Technology
Labuyo, Tangub City, Philippines

ABSTRACT
The study developed a threshold model for risk escalation and time boxing efficiency in Agile sprints of small teams. The model classifies sprint risk as low, medium or high using measurable sprint indicators, including completion rate, non-completion rate, scope change rate, punted issue rate, workload per developer, timebox overrun and timeboxing efficiency. A web-based prototype was developed using React with Vite, Express.js, and MySQL to support sprint creation, work item management, dashboard viewing, risk classification and exporting Excel reports. The model was calibrated with sprint-level data from the Meso Sprint Dataset and was evaluated in six controlled sprint cases representing realistic conditions of software development. Results showed one low-risk sprint, three medium-risk sprints and two high-risk sprints. Low-risk classification occurred when task completion and timeboxing remained within acceptable limits, while medium and high risks were caused by low completion rate, scope change, and timebox overrun. Overall, the findings show that a rule-based threshold model can provide a simple, transparent, and practical approach for monitoring sprint performance, supporting early risk escalation, and improving timeboxing awareness in small Agile teams.

Keywords:
Agile sprint, risk escalation, timeboxing efficiency, threshold-based model, sprint monitoring, rule-based model.

INTRODUCTION

As an increasingly common approach to software development, agile project management offers teams the ability to adapt and develop deliverables quickly while also encouraging flexibility and collaboration. Agile's ability to facilitate quick responses to changing requirements through iterative development and feedback loops has made it a popular choice among developers. Despite their proven effectiveness, Agile environments still struggle with issues related to risk management, resource utilization, and decision-making during the sprint cycle processes. This limitation is supported by Tavares et al. [1], who identified that Agile methods, although increasingly popular, do not offer specific activities for managing risks. They further noted that the lack or inadequate application of risk management is one reason software development projects fail. Their findings suggest that Agile teams may improve effectiveness through risk management tools without necessarily increasing time investment, but many Agile frameworks still lack structured risk monitoring mechanisms. According to recent research on the impact of agile practices, while productivity and speed of delivery are typically improved by the use of Agile methods, project managers frequently encounter challenges due to ineffective monitoring of risks and time constraints [2], [3] , Additionally, the complexity of many of today's development projects requires a more organized approach to managing risks early and efficiently during limited sprint cycles. These challenges become more pronounced in small-scale Agile teams and Small and Medium Enterprises. Choraś et al. [4] found that conducting Agile projects in SMEs is demanding because projects often begin and end quickly while still needing to meet customer quality requirements. Their research pointed to the need for realistic metrics and process improvements targeted at small development teams. Similarly, Mishra [5]

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

found that organizational culture, team structure and management support are critical success factors for Agile adoption and small teams may require different approaches compared to enterprise scale implementations.

The difficulties and the obstacles faced have led to the use of many models, techniques, and methodologies within the Agile methodology during the current technological evolution. This has involved the deployment of Machine Learning, Clustering Methods, and risk management frameworks. The application of machine learning helps to predict the performance of teams and to detect the risks of sprints in the recent paper of Jazm et al. [6]. Moreover, Nacional et al. [7] addresses the identification of issues during the execution of sprints in real-time. Also, a systematic, classification-based approach to managing risk in an Agile environment is described in [8] by Khanna et al. Recent advances in the threshold-based methods also offer a potential approach to these problems. Legnaro et al. [9] proposed a threshold-based framework amenable to optimization via systematic threshold tuning and demonstrated that simple threshold mechanisms can improve performance across a variety of applications. Most of these technologies are complex and not specifically designed to provide a consolidated or simplified method for integrating risk escalation with timeboxing for efficiency.

Innovative ways of handling Agile are leveraging automated processes for risk escalation, structured communication models, and scalable frameworks to enable better project execution. Organizational transformation research also demonstrates the complexity of Agile adoption. Lassenius [10] documented that Agile transformation could also go beyond software development to finance and leadership. Their study also highlighted the difficulty of managing team autonomy, organizational control, and coordination, underscoring the importance of simple, effective monitoring mechanisms in a limited organizational setting. A study by Seppi [11] has shown that structured escalation frameworks improve responsiveness and efficiency of decision making; similarly, the study of Gbabo et al. [12] showed that formal ways of communicating can lead to better coordination. Akhiwu [13] showed that integrated agile practices can be successful at scale, in large organizations. Also, time-boxing has been shown to improve efficiency of digital transformation projects Shaughnessy and Goulding [14]. Nevertheless, many of these implementations have been costly and designed for large organizations. In the Philippines, although agile is becoming prevalent among academic institutions and small businesses, small organizations still primarily rely on manual monitoring and rudimentary solutions that limit their ability to effectively manage risk and to timebox effectively within their sprint cycles.

While there have been advances in Agile practices, the current implementations use the principles of flexibility and fast delivery but mostly rely on experience of team members and manual risks monitoring. Agile tools were primarily developed for large enterprises and overly complex for smaller organizations or departments. Most companies treat risk and time efficiency separately than they do together, and there has been very little research on integrating both of them with simple, threshold-based indicators that would be appropriate for small agile environments. Small agile teams often do not monitor the delays of their tasks, the imbalance of their respective workloads, and/or utilization of their sprint time. Without a structured monitoring system in place, risks are identified too late, and time-boxing methods become less effective, resulting in less productivity and missed sprint goals, as supported by Kalluri [15] and Rehman [16]. Therefore, the purpose of this study will be to develop and evaluate a simple threshold-based model that identifies an organization's risk level and measures a company's time-boxing efficiency through the use of simple indicators that could be applied by small agile companies.

REVIEW OF RELATED LITERATURE

Agile software development has become commonplace in software development companies due to the numerous benefits it provides, as discussed by Choraś et al. [4]. However, conducting Agile projects is particularly demanding in Small and Medium Enterprises (SMEs), because projects start and end quickly but still have to fulfill customers' quality requirements, as emphasized by Choraś et al. [4]. Putrianasari et al. [17] conducted a systematic literature review examining problems in the adoption of the Agile-Scrum software development process in small organizations. They found that some organizations, especially small ones with limited resources, face unforeseen difficulties while implementing Agile-Scrum software development. These issues were categorized into four primary areas: technology, people, process and organization, and Agile techniques. Mishra [5] scientifically validated that organizational culture, team structure and management support are the key success factors in Agile adoption in 52 software businesses in 7 different countries. Their results also demonstrated that the improved control of work is perceived as one of the key benefits of Agile methodologies in small and large enterprises. Much attention has focused on metrics for systematic measurement and improvement of Agile processes. Choraś et al. [4] suggested a set of measures for Agile development processes, which were practically verified in the context of a small-size software

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

development company based on the principles of ActionResearch. Their work reveals that these metrics can be utilized by organizations in their Agile projects and tailored to their own needs and technologies. In this direction De Luca et al. [18] offered a systematic technique to develop context-specific software metric thresholds to be included into fault detection workflows in industrial environments. This type of approach can be used to cross-project defect prediction by obtaining and learning thresholds from one set of projects and applying them to independently developed firmware, allowing reuse across similar software systems without retraining or domain-specific tuning.

Recent research has investigated various techniques to enhance monitoring of Agile projects including risk management frameworks, process metrics and intelligent support tools. The essential limitation identified by several studies is that Agile methods do not provide specific activities for risk management, as stated by Tavares et al. [1]. Tavares et al. [1] proposed and evaluated a tool to manage risks in software development projects using Agile methods, enhancing the effectiveness of risk response planning without increasing the time spent in the process. Nacional et al. [7] acknowledged that Agile can support real-time problem detection during sprint execution; however, the lack of proactive mitigation strategies may still limit the success of Agile projects. Additionally, in the research study Omar et al. [19] practitioners have been encouraged to follow proper structured risk management practices, but this methodology tends to only be used in a reactive fashion rather than predictively. Zahedi et al. [20] presented a risk management framework for projects that use Agile methodology based on ISO31000 standards. Their evaluation in two running Agile software projects showed that the proposed framework increased the average positive risk reaction score by 49%. Moreover, Dugbartey and Kehinde [2] discussed that the benefits of Agile collaboration and faster delivery may also create challenges in balancing stakeholder engagement and project risk control. Singh et al. [21] developed the AGP model specifically for IT projects, which explained up to 76% variability in the potential risks that could arise during IT project deployment. Their study, based on 1868 valid survey responses from European and Asian countries, identified four key factors for dealing with risks in IT projects. These studies show that Agile teams still require practical mechanisms for risk monitoring, escalation support and sprint control.

Related studies on Agile practices depicts that adopting to Agile improves the delivery and collaboration but finds that there are challenges in coordination, sprint monitoring and risk response. This indicates that Agile and timeboxed methods may improve productivity and delivery, but they may not work well when teams do not have a structured monitoring system of delays, workload imbalance and unexpected disruptions in sprints. In addition, Khanna et al. [8] emphasized that managing risks in distributed Agile teams is complicated, particularly due to barriers to communication. Finally, Shaughnessy and Goulding [14] noted that time-boxed approaches could help in delivering efficiently, but they may still be constrained when teams encounter unexpected disruptions without structured monitoring. The findings showed that Agile risk management and sprint monitoring still need simpler and more practical approaches to identify delays, workload imbalance, and escalation needs.

Also, present literature focuses on risk escalation, communication and enterprise-scale Agile adoption. Seppi [11] reports the impact of a structured escalation process on the speed of decision-making but lacks automated predictive analytics capabilities and uses a semi-structured escalation process. This was further supported by Gbabo et al. [12] which highlights how to improve coordination efficiency; however, challenges with coordination were due to manual processes causing delays. Furthermore, Kalluri [15] discusses human behaviours and decision-making, but lacks a technology-enabled means of predicting risks within Agile processes. Finally, Akhiwu [13] demonstrates the scalability of Agile principles/practices, while referring to maintaining scalability across large systems as an obstacle and Rehman [16] highlight the need for embedding risk management frameworks but reports inadequate real-time analytics data availability for measuring risk. The literature reveals that while risk management frameworks and process metrics have been developed separately, there remains limited integration of these approaches using simple threshold-based indicators. Singh et al. [21] mentioned that processes and models are often prioritized over the human element, and advocated for more practical and human-centered approaches. The market has a huge void for integrated solutions to address risk escalation and timeboxing efficiency at the same time with simple indicators, especially for small scale teams that need lightweight yet effective monitoring mechanisms. The studies reviewed suggest that, although Agile methods support flexibility, collaboration and rapid delivery, small-scale Agile teams are still hindered by reactive risk monitoring, fragmented process metrics, inefficient Therefore, the researchers proposed a threshold-based model based on simple indicators can be used for risk level identification, risk escalation support and timeboxing efficiency measurement in small-scale Agile teams.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

METHODOLOGY

The study applies a quantitative developmental research method to design and evaluate a rule-based threshold model for risk escalation and timeboxing efficiency in Agile sprints in small-scale teams. It calculates quantifiable sprint indicators such as completion rate, non-completion rate, scope change rate, punted issue rate, workload per developer, timebox overrun, and timeboxing efficiency to classify sprint risk as low, medium, or high. A web-based prototype was developed to support sprint creation, work item management, sprint analysis, dashboard viewing and export to excel report. The methodology involved five stages: dataset selection, data preparation, indicator computation, threshold-based risk classification, system prototype development and model validation.

3.1 Dataset and Data Preparation

The study used secondary sprint data from the Agile Scrum Sprint Velocity DataSet by Randula Koralage as the basis for model calibration and analysis [22]. The dataset contains Agile Scrum sprint records from open-source projects, and the Meso sprint dataset was selected because it contains sprint-level variables needed for the proposed model. These variables consist of sprint start date, sprint end date, sprint completion date, total number of issues, completed issues, issues not completed, punted issues, issues added during the sprint, number of developers, and sprint length.

Only the fields relevant to sprint performance, risk escalation and timeboxing efficiency were retained to prepare the dataset before analysis. Date fields were converted to MySQL compatible formats, missing completion dates were handled using the sprint end date if needed, and invalid or missing sprint duration values were recomputed based on the difference between the sprint start date and sprint completion date. Records without essential identifiers, such as sprint ID or sprint name, were excluded. The prepared data were then imported into the system as calibration records and separated from user-created sprint records using the record source field.

Variable	Description
Sprint start date	Beginning of the planned sprint timebox
Sprint end date	Planned end of the sprint
Sprint completion date	Actual completion date
Total number of issues	Total sprint workload
Completed issues	Delivered work
Issues not completed	Unfinished work
Punted issues	Deferred or moved work
Issues added during sprint	Scope change indicator
Number of developers	Used to compute workload per developer

Table 1. Variables used in the study

3.2 Dataset and Data Preparation

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

Figure 1. Flowchart of the proposed threshold-based sprint risk model

The proposed threshold-based model is a rule-based sprint analysis framework developed to evaluate sprint performance and determine risk levels in Agile sprints. The model uses numerical sprint data as input and applies calibrated threshold rules to identify whether a sprint is at low, medium, or high risk. Unlike machine learning approaches, the model does not rely on training or prediction algorithms. Instead, it follows a fixed decision process composed of sprint data input, indicator computation, threshold evaluation, risk classification, and result generation. Fig. 1 shows the flowchart of the proposed threshold-based sprint risk model.

3.3 Indicator Computation, Threshold Calibration and Risk Classification

The model computed several sprint indicators to measure delivery performance, unfinished work, scope instability, deferred work, workload pressure, and timeboxing efficiency. The completion rate was computed as

$$CR = \frac{CI}{TI} \quad (1)$$

where CR is the completion rate, CI is the number of completed issues, and TI is the total number of issues. The non-completion rate was computed as

$$NCR = \frac{NCI}{TI} \quad (2)$$

where NCR is the non-completion rate and NCI is the number of issues not completed. The scope change rate was computed as

$$SCR = \frac{AI}{TI} \quad (3)$$

where SCR is the scope change rate and AI is the number of issues added during the sprint. The punted issue rate was computed as

$$PIR = \frac{PI}{TI} \quad (4)$$

where PIR is the punted issue rate and PI is the number of punted issues. The workload per developer was computed as

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

$$WPD = \frac{TI}{ND} \quad (5)$$

where WPD is the workload per developer and ND is the number of developers. Adarsh Anand et al. [23] defines timeboxing in Scrum context: “In Scrum, a sprint is a time-boxed event, and tasks are broken down in such a way that they fit into these time frames.” They also note that “The Scrum requires the sprint length to be fixed” and establish that sprints have “a defined task and a flexible plan to achieve it.” Melnyk et al. [24] reinforces the timeboxing principle, stating that “Sprint duration is a fixed time period of 1-4 weeks. It has the same length until the end of the project.” Based on these principles, the study included planned duration, actual duration, timebox overrun, and timeboxing efficiency as sprint timeboxing indicators. The planned sprint duration was computed as

$$PD = ED - SD \quad (6)$$

where PD is the planned duration, ED is the sprint end date, and SD is the sprint start date. The actual sprint duration was computed as

$$AD = CD - SD \quad (7)$$

where AD is the actual duration and CD is the sprint completion date. The timebox overrun was computed as

$$TO = AD - PD \quad (8)$$

where TO is the timebox overrun. The timeboxing efficiency was computed as

$$TE = \frac{PD}{AD} \quad (9)$$

where TE is the timeboxing efficiency. A value close to 1 indicates that the sprint closely followed the planned timebox. After computation, each indicator was evaluated using calibrated threshold values. Completion rate was classified as medium risk when CR<0.40and high risk when CR<0.20. Non-completion rate was classified as medium risk when NCR>0.50and high risk when NCR>0.70. Scope change rate was classified as medium risk when SCR>0.50and high risk when SCR>0.80. Punted issue rate was classified as medium risk when PIR>0.20and high risk when PIR>0.40. Workload per developer was classified as medium risk when WPD>8.00and high risk when WPD>12.00. Timebox overrun was classified as medium risk when TO>3days and high risk when TO>7days.

The final sprint risk level was determined using a priority-based rule. If at least one indicator was classified as high risk, the final sprint risk level was classified as High. If no high-risk indicator was detected but at least one indicator was classified as medium risk, the final risk level was classified as Medium. Otherwise, the sprint was classified as Low risk.

3.4 System Development

A web-based prototype was developed to implement the proposed threshold-based sprint risk model. The frontend was developed using React with Vite, on the otherhand the backend was implemented using Express.js. MySQL was used as the database for storing sprint records, calibrated threshold values, and computed sprint analysis results. The overall architecture and data flow of the prototype are shown in Fig. 2.

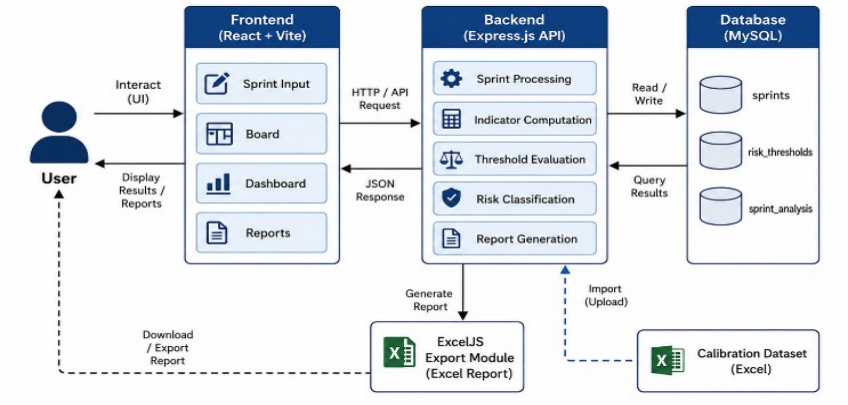

Figure 2. System Architecture of the Prototype

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

The system allows users to create a sprint, manage sprint work items, complete a sprint, calculate sprint indicators, classify sprint risk levels, display dashboard summaries, and export the generated results to an excel report. The database is made up of three main tables: sprints, risk_thresholds and sprint_analysis. The sprints table contains the raw sprint data, the risk_thresholds table contains the calibrated threshold values and the sprint analysis table contains the computed indicators, timeboxing status, risk classifications and generated explanations.

3.4 Model Validation

The model was validated using computational verification and prototype testing. Validation was based on the correctness of the constructed artifact for the implementation of formulas, threshold rules and output generation, as the study developed a new rule-based model and web-based prototype. This approach is consistent with the design science research in which the design, construction and evaluation of artifacts such as systems, applications and methods are the main focus [25]. Validity in design science can also be viewed from the perspective of whether an artifact works correctly and achieves the intended goal [26].

The validation process consisted of verifying the calculated sprint indicators against the defined formulas, confirming that the threshold rules yielded the expected low, medium, and high-risk classifications, and confirming that the final risk-priority rule was correctly applied. Furthermore, the consistency of the output was confirmed across the backend-generated analysis, database records, dashboard display, and exported Excel report. Validation was not primarily focused on predictive accuracy, given the rule-based nature of the proposed model rather than machine learning. Instead, it dealt with correctness, consistency and a proper application of the threshold-based decision rules.

RESULTS AND DISCUSSION

The prototype has been validated in six controlled sprint test cases based on realistic software development activities. The generated test cases were used to validate the ability of the proposed threshold-based model to classify the risk level of the sprint based on measurable sprint indicators. The test cases covered different sprint scenarios such as normal completion, report generation delay, weak dashboard delivery, backend integration problems, urgent change requests and delayed testing and deployment. The prototype outputs include completion rate, non-completion rate, scope change rate, rate of punted issues, workload per developer, timebox overrun, timeboxing efficiency, timeboxing status, and final risk classification.

Sprint Case	Sprint Name	Completed	Un-finished	Added	Deferred	Timebox Overrun	Time boxing Efficiency	Risk Level
Sprint 1	User Account Management Module	5	1	0	0	0 days	100.0%	Low
Sprint 2	Report Generation Module	4	2	0	0	4 days	77.8%	Medium
Sprint 3	Dashboard Enhancement	2	3	0	1	0 days	100.0%	Medium
Sprint 4	Backend API Integration	1	5	0	0	0 days	100.0%	High
Sprint 5	Urgent Change Request Handling	5	3	6	0	0 days	100.0%	Medium
Sprint 6	Testing and Deployment Preparation	3	4	0	1	9 days	60.9%	High

Table 2. Sprint-Level Risk Classification Results

Table 2 presents the generated sprint-level results of the prototype. Of the six sprint cases, the results showed one low-risk sprint, three medium-risk sprints, and two high-risk sprints. The results show that the model classified sprint risk based on different causes, such as low completion rate, scope change, and timebox overrun.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

This suggests that the model is not dependent on a single factor but evaluates a number of sprint indicators before classifying the final risk level.

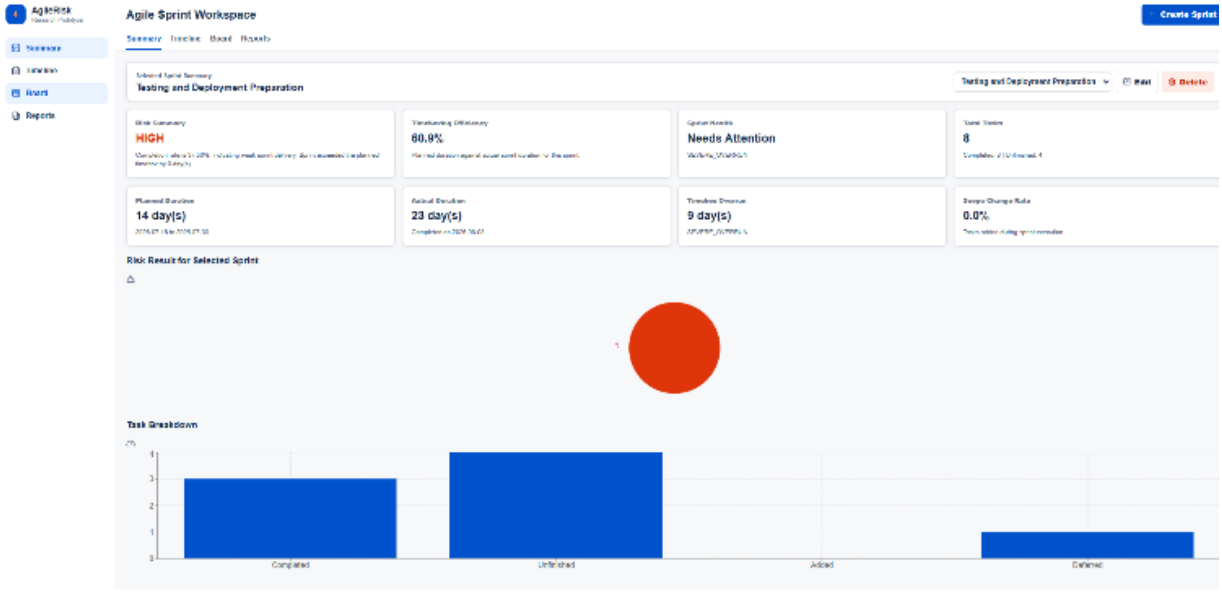

Figure 3. Sprint Summary Dashboard Showing Risk Classification and Timeboxing Efficiency

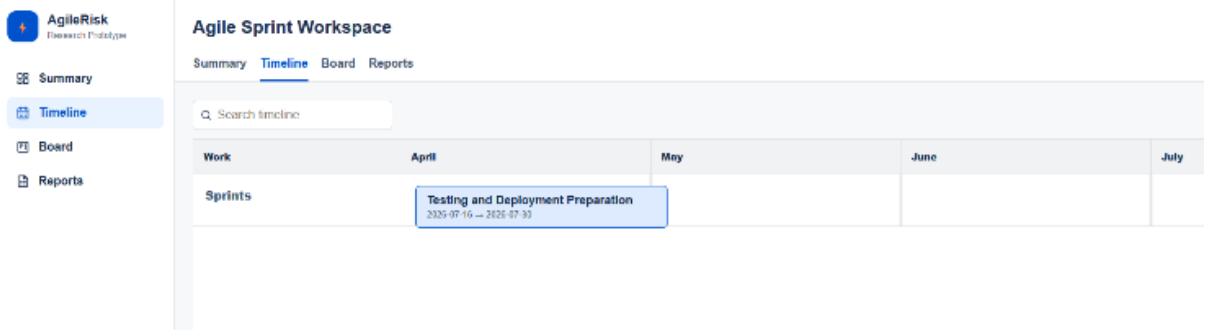

Figure 4. Sprint Timeline View of the Active Sprint

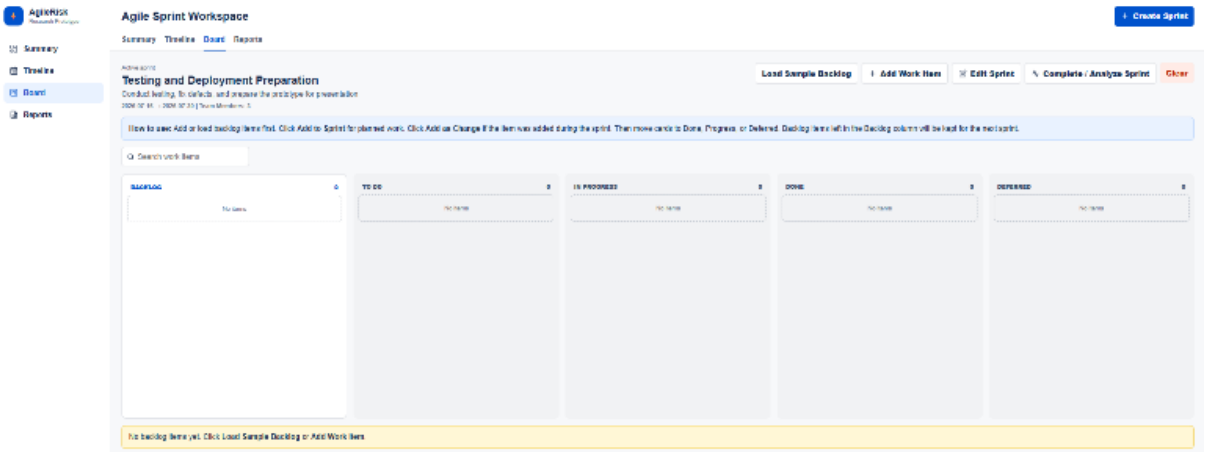

Figure 5. Sprint Board for Work Item Management

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

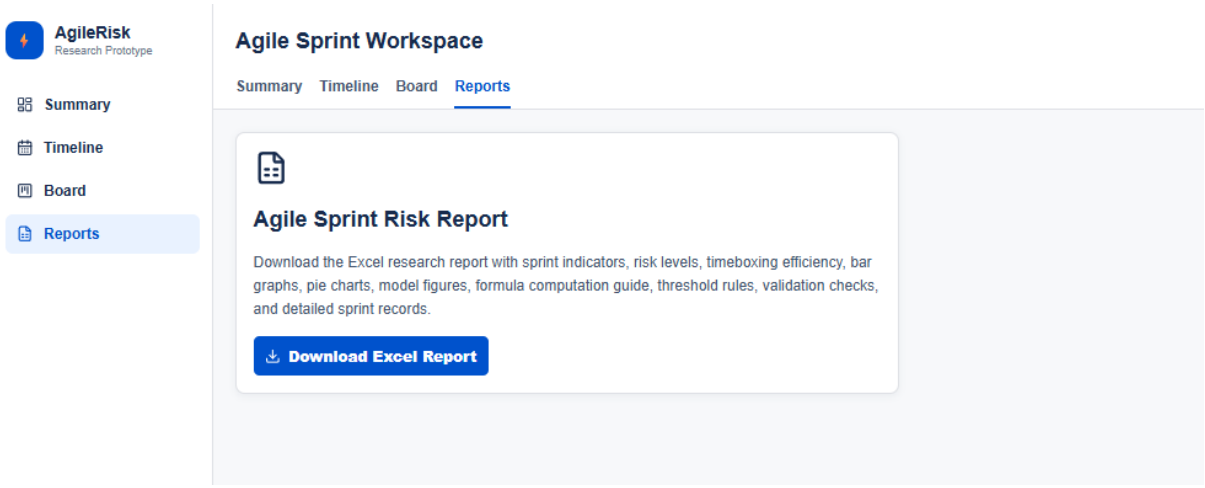

Figure 6. Excel Report Export Interface

The implementation of the web-based prototype of the proposed threshold-based sprint risk model is shown in Figures 3-6. Fig. 3 shows the sprint summary dashboard that shows the calculated sprint indicators, risk level classification, sprint health status, timebox overrun and timeboxing efficiency. Fig. 4 shows the timeline module for visualizing sprint schedules and planned sprint durations. The sprint board for managing the backlog and tracking sprint tasks through the different stages of the workflow is shown in Fig. 5. Fig. 6 shows the report module. The users can export the sprint analysis results and the computed indicators to an Excel report. These outputs show that the proposed model was successfully deployed and could support sprint monitoring, risk escalation analysis and timeboxing evaluation in an Agile sprint environment.

Risk Level	Number of Sprints
Low Risk	1
Medium Risk	3
High Risk	2
Total	6

Table 3. Risk Distribution of Controlled Sprint Test Cases

The overall risk distribution of the controlled sprint test cases is shown in Table 3. The result shows that medium risk sprints were the most frequent, followed by high-risk sprints and then low risk sprints. This was expected as the test cases were purposefully created to represent different sprint scenarios. The existence of all three risk levels indicates the ability of the prototype to classify low, medium and high sprint scenarios based on the calibrated threshold rules.

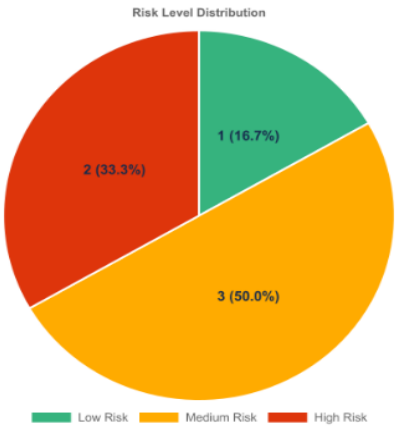

Figure 7. Risk Distribution of Controlled Sprint Test Cases

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

The risk distribution of the six controlled sprint test cases is shown in Fig. 7. The figure shows that the model classified one sprint as low risk, three sprints as medium risk and two sprints as high risk. This is in line with the result in Table 3 and shows that the model was able to distinguish between normal sprint conditions and those sprints that need monitoring or immediate attention.

The first sprint case, User Account Management Module is a normal sprint condition. This sprint included tasks like designing the user account form, creating the user registration API, adding edit profile functionality, implementing role selection, adding password validation, and polishing the user interface. The sprint completed 5 out of 6 tasks and only one task is in progress. There were no other tasks, no late tasks, no timebox overruns. Planned and Actual duration was 14 days; Timeboxing efficiency was 100.0%. All metrics were in the acceptable threshold limits; sprint was categorized as Low Risk.

The second sprint case is the Report Generation Module. This is a sprint with a schedule delay. This sprint includes report filters, report download, report formatting, excel export, summary cards and date format correction. Researchers finished 4 of 6 tasks, but the sprint was 4 days longer than expected. Planned duration was 14 days, actual duration was 18 days, resulting in a timeboxing efficiency of 77.8%. The sprint was rated as Medium Risk due to the timebox overrun that was above the medium-risk threshold.

The third sprint case is Dashboard Enhancement, a sprint with a weak delivery output. This sprint task consisted of dashboard cards, task breakdown charts, risk distribution charts, filtering and layout improvements. Of the 6 tasks, 2 were completed, 3 were unfinished and 1 was deferred. The sprint was completed in 14 days as planned, Scope was unchanged, but completion rate was only 33.33%. The sprint was classified as Medium Risk because the completion rate falls under the range of the medium risk threshold.

The fourth sprint case is the Backend Integration and is regarded as a high-risk sprint based on the very low completion. The sprint included the creation of insert, update, and delete APIs, linking the frontend to backend routes, fixing the database insert problems, and testing backend validation responses. Only 1 of 6 tasks were completed, 5 are not yet finished. This led to a completion rate of 16.67% and a non-completion rate of 83.33%. The sprint had no timeboxed overrun, but the very low completion rate and high non-completion rate exceeded the conditions of the high-risk threshold. Therefore, the sprint was categorized as High Risk.

The fifth sprint case, Urgent Change Request Handling, is a sprint influenced by scope change. This sprint is dedicated to scheduled activity like review of board workflow, producing sprint documentation. However, the sprint had additional work items such as validation adjustments, confirmation modals, risk explanation display, dashboard label changes, export validation checks, and report style changes. There were a total of 8 task of which 6 were identified as added during sprint execution, resulting in a scope change with a rate of 75.0%. The sprint finished on due time in which 5 were completed, but the high number of increased tasks indicated significant scope creep. The sprint was classed as Medium Risk because the scope change rate over the medium-risk threshold but did not exceed the high-risk threshold.

The sixth sprint case, Testing and Deployment Preparation, is an example of a high-risk sprint due to the critical timebox overrun. This sprint consisted of functional testing, board movement testing, excel report testing, dashboard bug fix, date mismatch correction, deployment preparation, user testing checklist preparation and validation screenshot finalization. The sprint was only able to finish 3 out of 8 tasks. There were 4 tasks that were left incomplete and 1 task that was deferred. The anticipated period was 14 days while the actual time reached 23 days. This resulted in a timebox overrun of 9 days and a timeboxing efficiency of 60.9%. Thus, the sprint was classified as High Risk because the overrun was above the high-risk threshold.

Indicator	**Result**
Total analyzed sprint cases	6
Low-risk sprints	1
Medium-risk sprints	3
High-risk sprints	2
Average completion rate	50.0%
Average scope change rate	12.5%
Average timeboxing efficiency	89.8%
Average timebox overrun	2.17 days

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

Table 4. Summary of Computed Sprint Indicators

The computed sprint indicators from the six controlled sprint cases are summarized in Table 4. Average completion rate was 50.0% meaning that half of the sprint tasks were completed across all test cases. The average scope change rate was 12.5% and was mainly driven by the urgent change request sprint. The average timeboxing efficiency was 89.8% and the average timebox overrun was 2.17 days. The values indicate that the controlled test cases contained both stable and unstable sprint conditions.

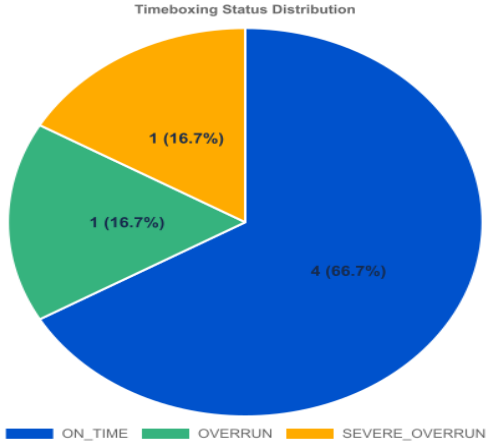

Figure 8. Timeboxing-efficiency per sprint.

Fig. 8 shows the timeboxing efficiency of each case of sprint. The timeboxing efficiency for User Account Management Module, Dashboard Enhancement, Backend API Integration and Urgent Change Request Handling sprints was 100.0%, meaning that the planned and actual durations were equal. The Report Generation Module was 77.8% efficient because of its 4 days overrun. The Testing and Deployment Preparation sprint had the lowest efficiency of 60.9% due to having exceeded the planned duration by 9 days. These results show the decrease of timeboxing efficiency when the real sprint duration is longer than its planned duration.

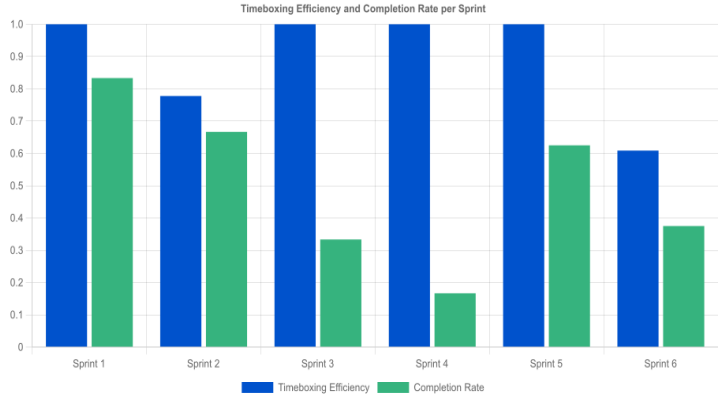

Figure 9. Timebox-overrun per sprint.

The timebox overrun values for each sprint case are shown in Figure 9. The Report Generation Module was rated medium risk as it did a 4 day overrun. The Testing and Deployment Preparation sprint went 9 days over the schedule. Because of this, it was considered High Risk. This confirms that the proposed model elevates the sprint risk when the actual completion date is beyond the planned sprint end date. In summary, the results suggest that the proposed threshold-based model can classify sprint risk using measurable sprint indicators. The model identified low risk when task completion and timeboxing were within acceptable levels. It also identified medium risk for moderate timebox overrun, weak completion or major scope change. The high-risk classifications were found when the sprint delivery was very low or when the sprint duration exceeded the planned timebox to a large extent. The results show that the model can support sprint monitoring by providing transparent and interpretable risk classifications for small Agile teams.

Results are from controlled prototype test cases and are not to be interpreted as actual industry sprint performance. Instead, they show the behavior of the prototype in realistic sprint scenarios. The controlled test cases demonstrate the model's ability to compute sprint indicators, apply calibrated threshold rules, generate risk

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

explanations, display dashboard summaries and export results for reporting. The model can be further tested under actual project conditions by using real-life sprint records in the testing in the future.

The overall results show the proposed threshold-based model was able to classify the sprint risk based on measurable sprint indicators. The controlled sprint cases showed a low-risk classification when task completion and timeboxing performance were within acceptable limits. For moderate issues, like timebox overruns, low completion rate or scope change, medium risk results were obtained. High-risk outcomes were associated with very low sprint delivery or a large difference between actual and planned sprint length. The results indicate that the model is able to offer transparent and interpretable risk classifications for small-scale Agile teams, by identifying the sprint conditions that lead to risk escalation. The results were derived from controlled prototype test cases rather than actual industry sprint records but indicate the prototype's ability to calculate sprint indicators, apply calibrated thresholds, generate risk explanations, and display results through dashboard and report outputs.

CONCLUSION

In the present study, a threshold-based model was developed to assess sprint risk and timeboxing efficiency in small-scale Agile teams. The model classified sprint risk as low, medium or high using measurable sprint metrics composed of completion rate, noncompletion rate, scope change rate, workload per developer, timebox overrun and timeboxing efficiency. The results indicated that the proposed model can distinguish various sprint conditions by explicit and interpretable threshold rules. The low-risk outcomes were achieved when sprint delivery and timebox are still within acceptable limits, and medium and high risks were produced when there is low task completion, scope changes and timebox overruns. The prototype also showed that the model can be implemented through a simple web-based system with dashboard summaries and export to excel reports. In general, the study demonstrates that a rule-based threshold model can offer a practical and understandable approach to monitor sprint performance, support early risk escalation and improve timeboxing awareness in small Agile teams. Future work could validate the model using more real sprint records and adjust the thresholds based on the team's actual performance.

REFERENCES

1) B. G. Tavares, M. Keil, C. Eduardo, and B. G. Tavares, "A Risk Management Tool for Agile Software Development A Risk Management Tool for Agile Software Development," J. Comput. Inf. Syst., vol. 00, no. 00, pp. 1–10, 2020, doi: 10.1080/08874417.2020.1839813.
2) N. Dugbartey and O. Kehinde, "Optimizing project delivery through agile methodologies : Balancing speed , collaboration and stakeholder engagement," 2025.
3) R. Sandstø and C. Reme-ness, "Agile Practices and Impacts on Project Success," vol. 11, no. 3, pp. 255–262, 2021, doi: 10.2478/jeppm-2021-0024.
4) M. Choraś, T. Springer, R. Kozik, and L. López, "Measuring and Improving Agile Processes in a Small-Size Software Development Company," vol. 8, 2020, doi: 10.1109/ACCESS.2020.2990117.
5) A. Mishra, "Organizational issues in embracing Agile methods : an empirical assessment," Int. J. Syst. Assur. Eng. Manag., vol. 12, no. 6, pp. 1420–1433, 2021, doi: 10.1007/s13198-021-01350-1.
6) Y. Jazm, S. Dinora, O. Jim, P. Orlando, and L. Torres, "Sprint Management in Agile Approach : Progress and Velocity Evaluation Applying Machine Learning," no. Ml, 2024.
7) P. Nacional, C. De Investigación, I. P. Nacional, and C. De Investigación, "Futures Perspectives : Risk Analysis in Agile Sprints through Problem Detection," no. Imsci, pp. 68–71, 2024.
8) E. Khanna, R. Popli, and N. Chauhan, "Identification and Classification of Risk Factors in Distributed Agile Software Development," vol. 21, pp. 1831–1852, 2022, doi: 10.13052/jwe1540-9589.2164.
9) E. Legnaro, S. Guastavino, and F. Marchetti, "Multiclass threshold-based classification and model evaluation," pp. 1–18, 2025.
10) Lassenius, Organizational Implications of Agile Adoption : A Case Study from the Public Sector, vol. 1, no. 1. Association for Computing Machinery, 2021. doi: 10.1145/3468264.3473937.
11) R. Seppi, "DESIGNING A RISK ESCALATION PROCESS FOR EXECUTIVE DECISION-MAKING IN DYNAMIC ORGANIZATIONS Case Study of a Fast-Growing International Manufacturing Company," no. May, 2025.
12) E. Y. Gbabo, O. K. Okenwa, and P. E. Chima, "Designing Communication and Escalation Models for Risk Coordination in Infrastructure Programs," pp. 760–766, 2022.
13) Akhiwu, "Operationalizing Innovation at Scale : Embedding Agile and Lean Principles into Enterprise Technology Workflows," 2025.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

14) H. Shaughnessy and F. Goulding, “Sprinting to digital transformation: a time boxed, Agile approach,” Strateg. Leadersh., vol. 49, no. 1, pp. 18–24, 2021, doi: 10.1108/SL-12-2020-0157.
15) R. Kalluri, “A Human Factors Study of Risk Management of Complex Agile Scrum Projects in Large Enterprises,” vol. 03, no. 08, pp. 38–44, 2022, doi: 10.56734/ijbms.v3n8a6.
16) S. Faculty, B. Examiner, M. St, and J. U. November, “INTEGRATING OPERATIONAL RISK MANAGEMENT Master of Science,” no. November, 2025.
17) R. Putrianasari, E. K. Budiardjo, K. Mahatma, and T. Raharjo, “Problems in The Adoption of Agile-Scrum Software Development Process in Small Organization : A Systematic Literature Review,” vol. 9, no. 1, pp. 495–504, 2024.
18) M. De Luca, D. Amalfitano, A. R. Fasolino, and P. Tramontana, “Statistical-Based Metric Threshold Setting Method for Software Fault Prediction in Firmware Projects : An”.
19) M. Omar, M. Ahmad, H. Ibrahim, A. Yasin, H. Awang, and A. Almogahed, “Enhancing agile project success: a comprehensive study of risk management approaches among Malaysian practitioners,” J. Softw. Evol. Process, vol. 36, no. 9, 2024, doi: 10.1002/smr.2681.
20) M. H. Zahedi, A. R. Kashanaki, and E. Farahani, “Risk management framework in Agile software development methodology,” vol. 13, no. 4, pp. 4379–4387, 2023, doi: 10.11591/ijece.v13i4.pp4379-4387.
21) S. Singh, G. Madaan, A. Singh, K. Sood, S. Grima, and R. Rupeika-apoga, “The AGP Model for Risk Management in Agile I . T . Projects,” 2023.
22) R. Koralage, “Agile Scrum Sprint Velocity DataSet.” Accessed: Apr. 25, 2026. [Online]. Available: https://github.com/RandulaKoralage/AgileScrumSprintVelocityDataSet
23) A. Anand, J. Kaur, O. Singh, and O. H. Alhazmi, “Optimal Sprint Length Determination for Agile-Based Software Development,” 2021, doi: 10.32604/cmc.2021.017461.
24) M. Information, S. Architect, and N. Technical, “Decision support technology for sprint planning,” pp. 135–145, 2020, doi: 10.15588/1607-3274-2020-1-14.
25) T. Haryanti, M. Surabaya, and A. Tjahyanto, “The Design Science Research Methodology (DSRM) for Self-Assessing Digital Transformation Maturity Index in Indonesia”.
26) R. Larsen et al., “Validity in design science,” 2025.