Risk Identification in Scrum and Kanban: A Comparative Analysis

Author: William Brian Richmond

Institution: Champlain College

Course: SDEV-550-85: Agile Development

Instructor: Willard Widmar

Date Due: 10 August 2025

Date Submitted: 10 August 2025

Introduction

Risk management in Agile methodologies must be proactive, adaptive, and tightly integrated into the development process. Scrum and Kanban, while both Agile approaches, take different paths toward identifying, assessing, and mitigating risks. This paper explores these differences through targeted 10-point checklists for each methodology, supported by real-world case studies, and highlights additional strategies — including pre-mortem analysis — to strengthen risk identification. It also examines a pervasive industry practice that introduces avoidable risks: the delegation of software documentation entirely to non-developer roles.

The accompanying checklist document supplements this analysis by providing tailored risk identification points for Scrum and Kanban, along with a Pre-Mortem section that enhances both methods. These checklists translate the concepts discussed here into practical, repeatable steps for Agile teams to apply in real-world settings.

Scrum Risk Identification

Scrum's structured, iterative nature offers multiple points for systematic risk identification. Sprint 0 can be used for a comprehensive risk review, mapping technical, organizational, and external threats before the first increment is delivered. Product backlog refinement allows high-risk items to be flagged early, while sprint planning provides an opportunity to detect skill gaps or capacity issues. The definition of done can embed

quality, compliance, and security criteria to mitigate the downstream problems (Schwaber, 2004).

Other Scrum checkpoints include tracking velocity changes across sprints to spot delivery instability, maintaining an impediment log to uncover systemic blockers, and engaging stakeholders at sprint reviews to detect misaligned expectations. Retrospectives provide a feedback loop to assess whether risk responses worked, while mid-release reassessments capture emerging risks before they escalate (Cohn, 2005; Ribeiro et al., 2009).

Kanban Risk Identification

Kanban's continuous flow approach relies on real-time visibility and data trends for risk detection. Pre-change workflow reviews assess potential risks before altering WIP limits or process stages. Cumulative flow diagrams reveal bottlenecks or throughput irregularities, while work item aging highlights tasks exceeding normal cycle times (Middleton & Joyce, 2011).

Frequent expedited items, WIP limit violations, and blocked tasks are red flags for process instability or overcommitment. Continuous stakeholder feedback mitigates misalignment risks, while throughput and lead time trends reveal performance declines. Defect tracking by workflow stage can pinpoint recurring quality risks, and prioritizing high-risk items early in the flow allows more time for resolution.

Case Study Insights

Ribeiro et al. (2009) describe implementing Agile risk management in a multi-project environment, where formalized tracking of impediments and cross-project dependencies significantly reduced delivery uncertainty. In contrast, the BBC Worldwide case study (Middleton & Joyce, 2011) highlights Kanban's strength in visualizing workflow risks but also reveals a recurring gap in process documentation and knowledge transfer.

The BBC case demonstrated that while Kanban improved predictability through data visualization, the absence of structured documentation steps left critical "why" decisions undocumented, creating onboarding challenges and increasing the likelihood of reintroducing resolved issues. This mirrors a widespread industry practice in which documentation is written by individuals who did not write the code, leading to omissions or misinterpretations of technical intent.

The Documentation Risk

Although technical writers excel at producing clear, accessible documentation, their distance from the coding process often results in a loss of critical decision-making context. This disconnect is a form of institutional knowledge risk, particularly in Agile environments where teams iterate rapidly and make frequent trade-offs.

In Scrum, this risk can be mitigated by embedding "developer-authored documentation" into the definition of done. The initial documentation — written immediately after coding — captures the rationale

behind design decisions while still fresh in the developer's mind. Technical writers can then act as proofreaders and editors, ensuring readability without altering meaning.

In Kanban, the absence of natural iteration boundaries makes documentation more vulnerable to neglect. Risk identification checklists can counteract this by incorporating explicit steps for documenting workflow changes, rationale for expedited items, and resolution paths for recurring defects.

Enhancing Risk Detection with Pre-Mortems

Pre-mortem analysis, introduced by Klein (2007), asks teams to imagine the project has already failed and identify the reasons why. This approach surfaces risks that might otherwise remain unspoken due to political sensitivity or cognitive bias. It is equally valuable in Scrum and Kanban, serving as a psychologically safe icebreaker for new teams and fostering creativity when exploring edge cases.

In Scrum, a pre-mortem can be integrated into Sprint 0 or early backlog refinement to anticipate high-risk scenarios. In Kanban, it can be applied before significant process changes to identify potential failure points in the flow. In both contexts, pre-mortems complement ongoing risk monitoring by broadening the scope of possibilities considered.

Conclusion

Scrum and Kanban offer distinct pathways to risk identification: Scrum leverages structured events and artifacts, while Kanban relies on continuous monitoring and visual management. Both benefit from targeted checklists, case study lessons, and enhancements like pre-mortems. However, standard industry practices — such as outsourcing documentation entirely to non-developers — can undermine risk management efforts in both systems.

By treating developer-authored documentation as a first-line risk control, supported by technical writer refinement, Agile teams can reduce institutional knowledge loss, improve maintainability, and strengthen overall resilience. The combination of tailored risk identification practices, adaptive enhancements, and conscious avoidance of unnecessary risks positions both Scrum and Kanban to deliver not just speed, but sustainable value.

References

Cobb, C. G. (2023). The project manager's guide to mastering Agile: Principles and practices for an adaptive approach. John Wiley & Sons.

Cohn, M. (2005). Agile estimating and planning. Addison-Wesley Professional.

Klein, G. (2007, September). Performing a project pre-mortem. Harvard Business Review. https://hbr.org/2007/09/performing-a-project-premortem

Middleton, P., & Joyce, D. (2011). Lean software management: BBC Worldwide case study. IEEE Transactions on Engineering Management, 59(1), 20–32. https://doi.org/10.1109/TEM.2011.2136431

Ribeiro, P., Domingues, C., & Gomes, R. (2009, September). A case study for the implementation of an agile risk management process in multiple projects environments. In 2009 4th IEEE International Conference on Management of Innovation and Technology (pp. 703–708). IEEE. https://doi.org/10.1109/ICMIT.2009.5215906

Schwaber, K. (2004). Agile project management with Scrum. Microsoft Press.