Sprint Review and Retrospective

Tariq Mack

Southern New Hampshire University

CS-250-14226-M01: Software Development Lifecycle

Professor Arthur McWain

April 19, 2025

# References

Schwaber, K., & Sutherland, J. (2020). The Scrum Guide. https://scrumguides.org

Pressman, R. S., & Maxim, B. R. (2014). Software Engineering: A Practitioner’s Approach (8th ed.). McGraw-Hill Education.

Sommerville, I. (2016). Software Engineering (10th ed.). Pearson.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Thomas, D. (2001). Manifesto for Agile Software Development. https://agilemanifesto.org

Sprint Review and Retrospective

# Applying Roles

Each role provided specific expertise that contributed to the smooth progress of the sprint. Additionally, our tester worked closely with the Product Owner and developers to ensure test cases aligned with acceptance criteria. The tester’s input helped the team identify edge cases early, reducing rework. Having clear delineation between roles minimized confusion and helped us maintain velocity throughout the sprint cycles.

Each role on the Scrum-Agile Team played a crucial part in the project’s success. As the Scrum Master, I facilitated daily standups, Sprint Planning, Sprint Reviews, and Retrospectives. My job was to remove obstacles and keep the team focused. The Product Owner maintained the Product Backlog and ensured alignment with stakeholder expectations. For example, when the client requested real-time flight notifications, the Product Owner updated the backlog, and our developers prioritized the feature in the next sprint. The development team collaborated to design, code, and test solutions. Their cross-functional nature helped accelerate progress without waiting on external handoffs.

Furthermore, using task boards and daily standups ensured that each user story progressed smoothly. By visualizing our workflow, we avoided bottlenecks and ensured every team member understood their responsibilities. Regular backlog grooming also contributed to cleaner sprint planning, enabling the team to confidently commit to sprint goals.

# Completing User Stories

Moreover, Agile’s iterative structure allowed us to introduce the new feature without restarting the development process. The flexibility in our sprint cycle meant we could adjust priorities without impacting other deliverables. We held an impromptu backlog refinement session to estimate the work involved and adjusted our roadmap accordingly. This capability to embrace change while maintaining momentum showed the strength of Agile methodologies.

The Scrum-Agile approach helped us complete user stories more effectively by encouraging frequent feedback and incremental delivery. We broke larger stories into smaller tasks and estimated them using story points. For instance, the story involving third-party API integration was split across two sprints. This allowed us to test incrementally, reducing the risk of major bugs. The Definition of Done helped ensure that each story was not just complete but also met quality expectations.

Another example of effective communication occurred when we used a shared Google Doc to list technical implementation notes for upcoming features. This living document provided transparency and allowed asynchronous collaboration, especially useful when some team members were remote. This helped maintain alignment across time zones and prevented duplicated efforts.

# Handling Interruptions

Additionally, burndown charts generated in Jira helped us track progress daily. This visual representation of remaining work kept the team accountable and provided stakeholders with quick insights. During retrospectives, we used anonymous surveys to collect honest feedback, leading to more actionable insights for continuous improvement.

When SNHU Travel requested hotel booking support halfway through development, our Scrum framework allowed us to adapt without losing momentum. We reprioritized the Product Backlog, clarified the new requirements, and worked it into the next sprint. Instead of delaying the project, we were able to pivot quickly and deliver a working version of the new feature by the following sprint’s end. This responsiveness is one of Agile’s biggest strengths.

The Agile process also enhanced our team morale. Having a voice in planning, daily scrums, and retrospectives made everyone feel valued and heard. While initial sprint planning sessions took longer due to learning curves, by the third sprint, our estimates improved, and our cycle time decreased. Compared to a Waterfall approach that might have locked us into rigid deliverables, Agile’s adaptive nature helped us deliver value with minimal waste.

# Communication

Communication was a major strength of our Agile team. Through daily Scrum meetings and open Slack channels, we stayed aligned and addressed issues quickly. For example, after Sprint Planning, I summarized our sprint goal and key tasks in a Slack message. This helped everyone stay focused, especially team members in different time zones. Regular Retrospectives also gave everyone space to voice concerns and suggest improvements, strengthening our collaboration.

# Organizational Tools

We used Jira for backlog management and sprint tracking, and Trello to visualize task progress. These tools aligned well with our Scrum events. Sprint Planning allowed us to assign tasks and estimate story points efficiently. During standups, team members updated Jira to reflect progress. After each sprint, the Retrospective helped us identify process bottlenecks and improve task clarity. For example, we decided to shorten our story point range to make estimation simpler and more consistent.

# Evaluating Agile Process

The Scrum-Agile approach proved to be an ideal fit for the SNHU Travel project. Its flexibility allowed us to adapt to changing requirements, such as adding hotel booking support. The pros included faster feedback, stakeholder involvement, and quicker delivery. A con was the learning curve for some team members unfamiliar with Agile, but they adapted after one sprint. Compared to Waterfall, where such a change might have caused delays, Agile allowed us to respond smoothly. In my opinion, this project validated that Agile is the better methodology for dynamic, client-facing software development.