Sprint Review and Retrospective

Noah Stubbs

Applying Roles:

Throughout the SNHU Travel project, I executed the roles of Scrum Master, Product Owner, Developer, and Tester. All roles are pivotal to a project’s success, but they all have a different lens to look through.

As the Scrum Master, I emphasized team cohesion, flexibility, and organic communication. My proposed strategy for daily scrum was unique in two key ways to support my vision. The first was that scrums began with non-work-related conversation. It strengthens communication by building camaraderie in a way that you can’t do by talking about work alone. It provided the team cohesion and organic communication I was after, while also paying dividends by fostering a more comfortable environment for everyone.

As the Product Owner, my biggest emphasis was getting clear user stories to allow the team to develop in the best way they deem fit. I made this comparable to what we called “Commanders Intent” in the Marine Corps. The customer says what kind of cake they want; we figure out how to bake it. Clear articulation and copious efforts will accomplish any goal.

As a Developer, I again focused on flexibility. The project had a sudden narrowing in scope and pivot towards wellness-focused travel. While some people may think the entire timeline is shot, I believe it better to just focus that energy on requesting clear backlog updates and acceptance criteria. Clear communication with the Product Owner and Tester ensured we could rapidly adapt.

As a Tester, I found acceptance criteria to be the most helpful. It directly guides the structure of what the test cases should be to ensure all features are up to expectations with minimal bugs.

Completing User Stories:

The Scrum-Agile methodology, or “Scragile”, allowed us to complete user stories more efficiently and effectively compared to how it traditionally goes. User stories are extremely similar to the requirements while Waterfalling, but they felt like they had a bit of personality to them. While business executives love to make things sound as fancy as they can, we kept it simple. Someone said “so that I can stay within my budget” and we left it just like that, while the traditional model would’ve had the secretary pull a thesaurus to change it to “ensuring fiscally responsible decision-making in alignment with strategic expenditure frameworks”. The other, and larger reason is that “Scragile” (we’re still testing names) performed well was iterative development. Instead of spending 274 days working on a project for it to be completely opposite of what the customer wanted, we went back and forth with the customer like a game of ping pong, but with features and ideas.

Handling Interruptions:

The pivot was understandably upsetting when it happened, but that discouragement was significantly shorter than usual. We found out that with this approach it was actually a pivot, while the old way requires so much rework it felt more like an about-face. A change in agile required a team huddle to stress eat Doritos and discuss the way forward. In contrast, a pivot in waterfall was a “go home, spend a weekend crying, then come back Monday having a quarter of your brain asleep, a quarter paying attention to the task at hand, and the other half having an existential crisis” type situation.

Communication:

Communication was a struggle over email, a breeze in the office. Emails are so common they end up being background noise and missed. The whole team missed important deadlines, but the project made it in the end.

Organizational Tools:

Agile project-management tools like JIRA and Azure boards are technically phenomenal. There’s a slew of features that you can do, and you can do them all. However, that was my exact problem with it. It was too much data getting thrown at you. A simple share drive is clean and easy to drop things in for the team and have one simple tracker. You can get in and get out without much distraction. The “brand named tools”, while *better*, slowed us down because navigating them was so bright, colorful, full of boards, widgets everywhere, things flashing, things moving, things blinging, it’s too much. I don’t mean this in a “I can’t handle this, it’s overstimulating” type of way, but I noticed it takes significantly more “brain power” to focus on what you’re doing in the flashy boards. That brain power requires not only focus away from the task at hand, but the efficiency losses of task switching. They have their use cases, but I don’t think they should be default.

Evaluating Agile Process:

Pros:

* Fast response to change (e.g., wellness travel pivot)
* High user focus via interviews and user stories
* Regular feedback loops through reviews and retrospectives
* Short development cycles allow for quicker course correction

**Cons:**

* Reliance on team commitment. If participation falters, so does progress
* Vague stories can slow testing and design if not clarified promptly

Overall, Agile was clearly a better approach for the project. While I have complaints and there’s plenty of room for improvement, the continuous iteration, flexibility, and stakeholder involvement far outweigh my gripes. A Waterfall model would’ve introduced rigid phases and delayed feedback. This, in-turn, would compound into wasted effort and a far less efficient and slower final delivery.