### Sprint Review and Retrospective

### Veronika Skliarova

### Southern New Hampshire University

### CS250: Software Development Lifecycle

### Instructor Joe Rangitsch

### December 12, 2021

Throughout this course, I tried on many hats, such as Scrum Master, Product Owner, Developer, and Tester, to learn how each role contributes to the process of software development. The Product Owner is the first line of defense between the development team and a client. Of course, they’re not there to defend anyone, as they are responsible for gathering data and requirements from the client and then conveying them to the team in the form of user stories. If something changes in the project, it is on the Product Owner to make appropriate changes in user stories and update the team. Scrum Master is another leadership role, but Scrum Master is a servant leader. They ensure the development team has everything they need and guide the team to become as efficient as possible. Scrum Master can also serve as a barrier between the team and the Product Owner when the PO tries to push the team beyond their limits. Developers, in their turn, are the muscle of the operation: their job is to deliver the application. And testers are helping them to assure the end-to-end quality of the product.

The Scrum-agile approach intends each little feature to go through the lifecycle individually, unlike the Waterfall, which sends a whole project down the pipeline. In terms of user stories, it means that each one starts in the hands of the Product Owner, then moves along to the developer, and, finally, gets to QA for testing. QA and development stages don’t have to be strictly consecutive. In fact, for our project, we had developers and testers work together so that the potential bugs could be identified early and therefore prevented. Traditionally, a good agile practice is to keep the user stories independent so that it’s easier to work on the project and adapt the changes as needed, and that’s the practice we stood by.

The beauty of the Scrum-agile approach is that it always expects change - it is not rigid. So, when we heard the news that the client changed their mind and wanted to revisit some of the design and functionality, we were able to swiftly incorporate the feedback into the project. Sometimes you have to scratch a feature, add a new one, or even go in a whole new direction. Unlike the Waterfall approach, Scrum-agile allows for sudden changes at any point of the project.

Communication is the key when it comes to teamwork. My team made sure to take advantage of all the tools available to us, including chat app, meeting calls, and, of course, emails, to have a written record of the project development. There was a moment where a tester needed requirements clarified, so they wrote an email to the Product Owner with all the questions and bullet points that needed attention and/or details. Another good example was when the interruption occurred, and the requirements needed to be changed. A developer took initiative to write an email to the Product Owner to inform them of the work that was done already, how they can treat the change in the feature, and what information they needed for the highest level of efficiency.

Organizational tools for me personally stand very close to the communication in any project, basically going hand in hand. The Scrum-agile approach accounts for the self-organizing teams, and we followed those principles by implementing various Scrum events with the help of our Scrum Master. Every day, in the morning, we would hold a short stand-up meeting where each team member talked about what they did yesterday, what they were going to do today, and whether there were any obstacles. Daily Scrum was also a proper place to find opportunities for collaboration because quite often, when one person is blocked by something, someone else will jump in to help, moving the project along and improving everyone’s professional skills. During that short meeting, we also updated the information radiator tool, or Scrum Board, which is a visual representation of all work in progress. Besides that, there was a Backlog Refinement meeting. That is where the team, together with the Product Owner, went through the backlog to trim any excess and add details to the tickets, which helped us prepare for the next sprint, and make sure that our backlog is up to date instead of being a black hole where tickets disappear. One more thing worth mentioning is the Sprint Planning meeting. After we groomed the backlog, it was easy to see the prioritized items, and everyone was on the same page about the work that needed to get done. So, during Sprint Planning, we had to account for the team’s velocity and select the number of user stories for the team to accomplish in the upcoming sprint.

One of the biggest pros of the Scrum-agile approach is, well, agility. Unlike in the Waterfall approach, in Scrum you start with a loose plan, and, as you go on, you generally plan for one, maybe two sprints ahead. That allows for unexpected turns and changes, just as happened in the SNHU Travel project when the client decided to go in a different direction in the middle of the course, and we were able to adapt to the changes. On the other hand, a lack of planning can be a disadvantage, as it creates a lot of uncertainty, and it can be difficult for inexperienced teams to handle. Another important pro of the Scrum-agile approach is that the development and testing are happening hand in hand, which cuts down on work and improves the overall quality of the end product. Finally, an agile framework offers a high level of interaction between the client and the Scrum team. This is beneficial for the product because the team can receive and incorporate feedback on the go, and the client knows exactly what they’re getting, even if they weren’t a hundred percent sure what they wanted, thus the product is as relevant as it can be. I think the Scrum-agile approach was a great decision for this project.