SNHU Travel Project: Sprint Review and Retrospective

Eric Foster

Southern New Hampshire University

CS-250: Software Development Lifecycle

Professor Anna Sandifer

08/25/2024

SNHU Travel Project: Sprint Review and Retrospective

Throughout this course, I have been challenged to take on the individual roles found within a scrum team. While operating each role, I was tasked to work with a client, SNHU Travel, assisting them in designing and building features for a web application. They wanted to move away from a traditional project management style, utilizing the waterfall method, and transition over to utilizing an agile approach for this project. The team consisted of a Product Owner, a Scrum Master, and the Development Team, consisting of developers and testers. For the remainder of this paper, I will be analyzing the various aspects of the Scrum team and agile methodologies that were utilized throughout this project.

Applying Roles

First, we must describe each role and how they apply to the agile/scrum team in making this methodology work so well. Each member of an agile/scrum team has an important role they play in ensuring that the agile approach is executed well.

Product Owner

Starting with the Product Owner, this role plays a crucial role in ensuring that communication between the client/stakeholders and the development team remains strong. They oversee a few different aspects. First, they oversee ensuring that the product backlog is up to date with the latest requests from clients. By focusing on the product backlog and interpreting user stories and their priorities, this can assist the development team in making informed decisions for each sprint on what to work on and how to approach that specific user story. Second, they oversee receiving feedback from users and clients and translating that into user stories. As stated just before, user stories help the team stay focused on what the user is requesting and what that priority is for each story. By effectively implementing and utilizing user stories, this helps keep the team organized at a high level and can help the team make more informed decisions about time boxing.

Scrum Master

Moving forward, I later took on the role of Scrum Master. There is quite a bit that is entailed with the role of a Scrum Master. They are essentially a bridge between the product owner, stakeholders/clients, and the development team. When I was in this role, it was my responsibility to assist the product owner in keeping the product backlog up to date with current prioritization of user stories. I was also tasked with setting up meetings both on the client to product owner side, as well as on the development side. When it came to the development team, my job as a Scrum Master was to be a mentor to each member, host scrum meetings, including but not limited to, Sprint Planning, Daily Standups, and other Scrum Events that ultimately helped the team stay organized and in constant communication with each other. When it came to daily meetings, my job mainly was to act as a mediator, ensuring that the meeting stayed on track and that all thoughts were shared and organized appropriately. The end goal for that was to soon instill a sense of self initiative for the team so that I as the Scrum Master was not always needed at these meetings, allowing me to focus more on development or helping the product owner. Ultimately, I was a resource for the team and was there to assist them in any way possible and coach them on agile methodologies as needed.

Development Team

Moving forward, I then stepped into the roles within the development team, in this case, a tester and later, a developer. Starting with the tester. My focus as a tester was to design and develop detailed test cases, ensuring that each user story had an in-depth testing scenario and acceptance criteria, ensuring that bugs would be minimized. As the saying goes, “Test early, Test often.” This is especially true in an iterative development environment. The more we test and the earlier we test, the less likely we are to let bugs slip into our codebase on the production end. Next, I stepped into the developer role. There isn’t much to this as it is self-explanatory. My job was to take the backlog and work with the team in planning out what user stories were going to be taken in for the sprint that we

were planning. Once we understood what user stories we were going to work on for that sprint, a Sprint Backlog would be utilized to help keep that sprints user stories organized and prioritized correctly. From there, as a team, we would have daily meetings, talking about what we completed, what was impeding/distracting us, and what we will be working on next for that day. Finally after that, we would break off and start our coding/work for the day. Within a scrum team/agile setting, the development team is given quite some creative leeway on how to approach a user story implementation. We are also heavily encouraged to give our inputs during the daily sprint meeting.

Completing User Stories

When it comes to the Agile approach of the SDLC, there are many aspects to this approach that help isolate critical functionality within a project. Planning out software can be a very complex concept if not executed properly. Having the ability and skills to break down complex tasks into smaller chunks of problems helps drive a project more successfully into deployment. There is one factor though that plays a heavy role in being able to decipher and interpret user problems so that we can apply the best solutions to those problems. That being the utilization of User Stories. For the SNHU Travel Project, it was my job as the Product Owner (When acting in that role) to meet with users and stakeholders, receive feedback about any features/changes wanted to be added into the new iteration, then create user stories of those user needs. User Stories play a critical role in helping the team make informed decisions about what to work on and is often utilized to keep a team busy due to the agile approach of always iterating through versions. User Stories consist of a few factors. Best practice is to state the requirement and isolate functionality as well as the purpose. User Stories commonly follow a structure of “Who?”, “What?”, and “Why?”, where who focuses on who the user story is intended to serve, the what focuses on outlining the requirements and tasks a user takes to complete a specific functionality, and the why focuses on the reasoning behind the functionality, ultimately adding value to the requirement.

Handling Interruptions

When it comes to handling interruptions, it is quite unique within an Agile environment. Within an Agile environment, Agile methodology is the idea of planning in heavy detail as you go and focuses heavily on the ability to be flexible and responsive to changes that occur. Because Agile does not focus on detailed expectations up front, but rather throughout the project, there is a level of uncertainty that will inevitably cause sudden changes to happen throughout the lifecycle of a project. For example, the change that happened within the SNHU Travel Project, where the clients/stakeholders decided to shift a feature towards focusing on detox/wellness vacation options, caused a mild interruption in the development process. However, because agile methodology is flexible, we were able to shift quickly with the new change and simply alter the existing code to include the change, thus saving time, and potentially saving costs in labor as a result.

Communication

As stated in our previous section, changes within an agile environment happen all the time. For our SNHU Travel Project, there was a change that appeared during the sprint where they wanted to shift a feature over to focusing on detox/wellness centered vacations. When changes like this occur, communication is key to ensuring that the change(s) are implemented and executed successfully and properly. In our case, questions arise regarding this change as it felt a little vague on how exactly the user wanted this change implemented. As a developer, our goal is to ensure we aren't redundant while also ensuring that we do not introduce new bugs into the existing codebase. So, utilizing email communication with different members of the team is essential to ensuring that as a developer, we know exactly what the expectations are for that feature. We can utilize an email format such as the following:

To: Christy – Product Owner; Brian - Tester

CC: Ron – Scrum Master

From: Nicole – Developer

Subject: New Requirements Expectation Clarification

Hey Christy and Brian,

As stated earlier today in our meeting this morning, I am starting my work on the revisions of the codebase to implement the new user requirements that were brought in this morning about adding a focus on detox/wellness type vacations.

If the expectation for how this feature will be implemented is as I think it will be, then the implementation should not take long and will be quite simple to implement. However, Christy, can you confirm with our clients if they would like the detox/wellness type vacations to be selected and shown by default or if they would like all packages to be shown by default and they can select the detox/wellness option on their profile preferences?

Also, Brian, can you provide me with some new test cases with this new requirement so that I may approach this requirement with the best solution in my code?

Thank you,

Nicole

This email example above does well in being concise and to the point while still getting the point across. In this example, I follow a few specific constructs to help format the email, First, I state the requirement(s) I am focusing on, then I direct the communication to the product owner and ask for clarification from the clients about how they would like to see this feature implemented, and finally, I direct my communication to the tester, asking for more test cases so that I can better understand and best approach the requirement with a fitting solution. By communicating and clarifying with each other, this helps positively steward an environment of collaboration and transparency within the project, and ultimately will help the team ensure that the development process is as smooth as possible.

Organizational Tools

For this project, there were a few tools utilized to help facilitate different aspects of the lifecycle. In our SNHU Travel Project, we utilized tools such as Azure DevOps and JIRA. Azure Devops helped the team transition into an agile approach by implementing the use of a Product Backlog, User Stories, and Sprints. JIRA is utilized to help keep the team organized by providing an online centralized area for individual tasks and bugs. Other tools that were utilized to help the team stay in communication due to a remote work environment were tools such as Skype. By utilizing these tools, it allowed the team to organize and provide information on tasks and bugs not only for them but for a distributed team environment as well.

Evaluating Agile Process

When evaluating the agile process for this project, there are many benefits that came with using an agile approach for this project, but also with that, came some mild drawbacks. It was quite difficult to predict the scope of the SNHU Travel project. This lack of scope certainty can harm the project negatively and can cause a project to derail and go well over the budget if it is not carefully managed. With that said, in this project, the benefit that an agile approach provided was the flexibility to implement a change that occurred during development. This flexibility is what allowed the team to provide value to the project and not miss a potentially critical need by the user. In my opinion, the quality of the project in this case far outweighs the uncertainty that comes with utilizing the agile approach.

In conclusion, the agile approach for this project was implemented and managed well and left both the development team and clients happy with the outcome of the project. Of course, each project is different and careful consideration of each projects requirements should be top priority before deciding whether or not to utilize an agile approach for that project.

References

Atlassian. (2024). *Waterfall Methodology for Project Management*. Atlassian. https://www.atlassian.com/agile/project-management/waterfall-methodology#:~:text=What%20is%20the%20Waterfall%20methodology

Cobb, C. G. (2015). *The project manager's guide to mastering agile : principles and practices for an adaptive approach*. John Wiley. https://ebookcentral.proquest.com/lib/think/reader.action?docID=1895876&ppg=131

DATAVALLEY.AI. (2023, January 31). *Waterfall Vs Agile: Which is better for You and Why?* Www.linkedin.com. https://www.linkedin.com/pulse/waterfall-vs-agile-which-better-you-why-datacademy-cloud/

Harris, R. (2021, May 20). *Software Development Life Cycle (SDLC)*. Www.linkedin.com. https://www.linkedin.com/pulse/software-development-life-cycle-sdlc-tutorial-richard-harris/

Scrum.org. (2019). *Home*. Scrum.org. https://www.scrum.org/