Danica Hesemann

CS 250

October 18, 2020

Retrospective

Over the past six weeks or so, our team has adopted a Scrum-agile approach to work on a project for SNHU Travel. Our team consisted of myself (the Scrum Master), the Product Owner, and a developer and a tester. As part of our new approach, the Product Owner met with the customer, SNHU Travel, to get an idea of what requirements we would need to incorporate and how the product should work. The Product Owner conducted interviews with existing SNHU Travel customers to get an idea of how the product could be improved, and created user stories and a product backlog to help our development team understand what our priorities were for the project. The user interviews were a big part of what made the product successful, because we were able to figure out what was in demand. The Product Owner remained in contact with both myself, the development team whenever possible, and the customer to keep everyone up-to-date on progress, issues, and new decisions.

With a working product backlog and user stories to start on, I held a sprint planning meeting and facilitated a conversation with the team to discuss how much we thought we could get done in a sprint, which user stories we should begin with, and what hesitations or questions we had about the work. We came up with a plan, and I implemented daily stand-up meetings and the use of a task board to keep everyone on track. In those meetings, I made sure everyone had an opportunity to update the task board, discuss what they accomplished the day before, what their plan was for the day, and if there was anything they need or anything they were not sure how to do that was holding them back. I mostly allowed the team to run the meetings themselves, but I intervened occasionally to keep the discussions on track and make sure questions or failures were met with helpfulness rather than hostility. I met regularly with the Product Owner to update them on our progress, and tried to help the team with anything holding them back, whether that was offering a different perspective on how to write a program, reaching out to the Product Owner for additional information, or bringing in donuts to boost morale.

The tester looked over the Product Owner’s user stories and created a rough test case for each one. They took into account what they knew about the product and our team’s priorities to make sure the test cases were as clear and thorough as possible. They found that they were missing some information that could help them write more specific and helpful test cases, so they reached out to the Product Owner for clarification. Here is a sample of the tester’s exchange with the Product Owner:

I have been looking over the user stories you sent to develop test cases, and I have a few questions.

Are we working on improving an existing product that we should be looking at to match the format for our additions? If not, did the customer specify any particular format they would prefer? Also, if we are working on an existing product, are there any existing functionalities that I should be aware of while creating test cases?

The tester revised the test cases with the feedback they received and sent them to the developer so they could reference them in their development. This communication between the tester and Product Owner enabled the developer to complete their work as well. As changes were made by both the development team and the customer, the tester updated the test cases to ensure the highest quality product.

The developer reviewed the user stories and contributed greatly to our estimate of how much we could get done in the time we had. They looked over the requirements and test cases to determine what needed to be done, then they created a program that met all of the requirements that could be tested and sent to the customer for feedback. This way, features could be easily added or removed at the customer’s request. The developer reached out to the Product Owner whenever they wanted clarification or had questions about the product, and to the tester when they wanted to make sure the program would work smoothly with the test cases. Here is an example of the developer reaching out to the Product Owner:

You mentioned this week that the product should focus on wellness/detox vacation packages. Did you want to try and find specific wellness/detox attractions to feature in the descriptions of the top destinations we already have, or should we come up with entirely new destinations based on the top locations for wellness/detox vacations? Additionally, you said this change is all we are working on for now, so does that mean we are only changing the destinations/the descriptions of the destinations?

And an example of the developer reaching out to the tester:

With the new changes to the requirements, I am changing the top destinations to feature wellness/detox vacation packages. If you haven’t already, you may want to add a test case to make sure each destination meets this requirement. Are there any test cases you have added that I may want to keep in mind while I work on this week’s changes?

This early communication between the developer, Product Owner, and tester kept the developer from getting stuck with uncertainty about what they should do next. In the meantime, they still may have been able to write up a rough plan for the parts they were confident in.

The agile approach we were working with has proven to be extremely effective and efficient for our team. As soon as our team received a product backlog from the Product Owner, we could begin planning and working on our sprint. It was easier for our team to plan shorter sprints, because we could better estimate how much we were able to get done in a shorter timeframe, and using several sprints allowed us to track our progress and make adjustments to our plan along the way. We worked on fewer user stories at a time, but completed them faster, ensuring that they got done or identifying problems sooner. Our daily communication helped us to eliminate duplicate work, adjust our plan as necessary, and address issues right away. For example, when the developer was working on a specific part of a user story, we would know that nobody else needed to worry about it. However, when they mentioned they were having trouble, the team was able to support them by looking over their work, offering up ideas, or helping them get any information they needed to finish. Our task board also proved to be extremely useful, so that we could look at it throughout the day to see what everyone was working on and plan what to do next. The minimalistic approach to development and frequent testing ensured that the product was working after the completion of each user story, and could easily be adjusted or changed. This greatly helped us when the project changed direction.

The change of direction could have been disastrous if we had been using a different project management model. We may have had to scrap months of work or sort through massive pieces of code to find which part needed to be re-written. We may have needed to come up with a significant new plan that pushed back the deadline. However, the development methods we used helped the transition to go smoothly for everyone. We quickly revised our user stories and communicated a new plan. The tester edited the test cases accordingly and the developer only had to make minimal changes to the program to fit the new product requirements. We were able to change course and still deliver a functional product on time. As the team finished up, they were great at communicating, looking over each other’s progress, and making sure that everything was accounted for. Here an example of an email that the developer sent to the Product Owner and tester before finishing their work:

I have been working on modifying my work based on the new criteria to focus on wellness and detox vacation packages, and I’m making great progress. As I finish up, are there any specific changes to the test cases with the new plan, or any new requests or requirements from the customer I will need to keep in mind?

The communication between the team, the organizational tools, and the principles of agile helped our team to thrive. While agile takes a bit of adjustment and might involve some trial and error with different teams, I think it was the best option for our team and the SNHU Travel project.