Chris Ellis

10/15/2023

CS250

Final project: Sprint Retrospective

**Sprint Retrospective**

Now that we’ve been completing work under the agile process for SNHU Travel, we are going to start having post-sprint reviews and retrospectives in order to determine how things are going, and along with the agile mindset, continuously improve in small increments.

Everyone on the team has made great strides in getting our work done and ensuring the success of the project, from our Scrum Master, to our product manager, down to the developers building the features and the testers ensuring that everything is working to the standards that we set.

Our product owner Christy performed a crucial role in this project by interfacing with the customers, conducting a focus group to determine what features our users would want in the new product, along with translating those requests into user stories for the development team to work on. In addition to this, she beautifully handled changes in the project requirements by being on top her job and bringing the changes to the development team early in the process after repeated meetings with customers.

Our Scrum master acted as the interface between Christy driving the product vision and the actual developers building out the system. Handling conflict is a key trait in a good Scrum master and after some conflict around the new changes requested by Christy (for detox vacation packages), he was able to moderate effectively and find common ground for onboarding the new requirements without having the developers feel like they were having their entire direction changed at a whim.

The contribution of the developers is obvious, as they built the product, but we should point out that they handled the new agile process exceptionally well. Even in the face of the rapidly changing requirements mentioned before, they were able to quickly pivot and adjust to the new requirements, producing an amazing product in only a few short weeks before the vacation season started.

The testers as a group also performed admirably in adjusting to the same requirement changes that developers did. Testers can often end up behind the eight ball as they often need to adjust test requirements behind the work developers are doing, but our testers were able to keep up and make sure that all of the user requirements and acceptance criteria were met.

The Scrum-agile process helped facilitate the rapid development of many of the user stories that were submitted by the product managers. Taking as an example the “Top Five Destination List” user story, where we wanted a feature that would allow a user to click a link to obtain the top five most popular destinations on the platform. This story was initially mentioned as a customer request during the focus group held by the product manager, and it was rapidly developed into a feature idea with a predicted implementation in a planning meeting involving the developers, Scrum master, and product manager. After this the story was sized and prioritized according to agile planning principles like affinity-grouping to vaguely break down stories into small/medium/large in order to spend less time on planning minutiae. In addition this story is an example of a piece that was changed later in the agile process due to new user requirements, and we were able to modify the existing feature to produce a top five list that showed the top five detox vacations instead of just the top five most popular.

The agile process greatly assisted the completion of this project in a timely manner, which is especially evident when looking at the large requirement changes that were handled right in the middle of the process even under a tight deadline. Product had come to the team with some changes in requirements, and the developers were immediately concerned that they were about to throw away all of their work in order to support these new requirements instead. The Scrum master was able to mediate between the product manager and the rest of the team to figure out that most of the existing work could easily be re-used in order to handle the new requirements, which were essentially a more tightly scoped function of the existing requirements. The product manager Christy also facilitated sticking to the deadline by deprioritizing other user stories in favor of getting these new important features implemented first.

Communication is in my opinion the most important aspect of agile that helped our team reach their goals. When communicating as a developer in this project, I was able to have good back and forth discussions with my team about which specific agile processes we wanted to implement and come to a consensus without too much strife. I had suggested the following regarding our ticket/worklog style and suggested using kanban:

“*I believe using a kanban style workflow will assist us in limiting work in process, which will help to remediate our issues with starting many projects/features that never get finished.” - Chris Ellis*

Some team members voiced concerns about the process causing issues, specifically:

“*I'm not sure I agree with going to full on Kanban in the midst of all this. I interned at a company that used Kanban for bug fixes and updates, and for small stuff like that it worked to have that linear "task in, task out" workflow, but I don't see how that will really be very effective when we are coding a new project from the ground up.” - Kinsey Christensen*

*“The way that I have pictured it, the Kanban board would be more useful for large scale visions of the user stories. For example, if we have 5 "large" user stories that are being worked through for this sprint, I think that it'd be helpful for both Scrum Master Ben and I to have a view on the team's progress as it pertains to the customer.” - Brenn Cunningham*

I was able to clarify some of the concern around these issues below:

*“Yes I'm referring to using a kanban style board to break up our tasks. I can see where we might have some issues using it in the initial phases of the project but it should suffice for handling user stories if we have more than the current number of developers.*

*We could try something like a 'swarm' where we gather all of the developers at first to setup the project and the scaffolding we're going to use to add features, and then shift to kanban style ticketing for implementing the features. We can even split up XP style and do pair programming on them if the number of user stories is small enough in order to drive faster development.” - Chris Ellis*

In the end we decided that we could effectively use a kanban style of story management in order to remediate some specific issues that were brought up regarding the old waterfall style of development at the company, namely having too much “work in process” leading to nothing ever reaching completion, and requirements changing a long time down the road when adjusting would be extremely costly.

The Scrum-agile organizational tools that our team implemented were instrumental in the successful completion of the project. One issue with waterfall style development is that there is no defined frequency for planning, and meetings. The agile meetings include a daily standup, along with the sprint-level pre-planning, and retrospectives. The daily standup meetings are incredibly effective at keeping everyone on track, and for team members to share issues early in the process in order to catch problems before they become major issues.

I believe after the previous analysis, that the Scrum-agile approach was the best fit for the project in its initial phase and will likely continue to serve well as we build on the project more in the future. There were both pros and cons with this approach, but I believe the overall outcome to be extremely positive.

On the pros side, we were able to meet our deadlines for development, which were very tight from the beginning, and provide the major features the customers had asked for, along with being able to handle rapidly changing requirements in the middle of development without having to push back the deadlines. Everything was not perfect with our approach though, and there were some possible communication issues that might have arisen had our team not been so stellar in handling things like the requirements changing. Another con was that this being the initial project we were using the Scrum-agile process for, we had to spend a lot of time up front deciding what processes and tools we were going to use, whereas later projects can use the work and experience of our team to help make this phase faster. With such a tight deadline on the project we needed to be very careful about not overloading our developers, and while we were lucky in this case we could deprioritize features to reduce the load after changing requirements, had we had issues we would likely have ended up needing to extend the deadline. Due to the project's scope and short deadline however, I don’t believe waterfall style project management would have performed well, and the changing requirements likely would have either derailed the project, or caused us to run into the same issues we previously had with too much work and very little actually being completed.

All-in-all I believe the project exceeded expectations and I credit most of that to our team and their ability to use this new process as a tool to effectively handle building a great product. We’ve seen many examples of problems solved by the Scrum-agile process that were previously hurting our developer’s and company performance, and I believe that we can continue to refine and improve the process in order to make even more productivity gains in the future.