Joy Kalinowsky

CS 250

Sprint Review and Retrospective

The Product Owner held meetings with stockholders such as the SNHU Travel representative client, created tasks out of the clients' user stories, and placed these tasks on a Scrum board that they created. This enabled workers to later on assign themselves the responsibilities of completing certain tasks. The developers developed, debugged, and refined the code for the website itself as well as provided documentation for the code in the source files. This documentation took the form of comments above every code segment in the SNHU Travel website source code that described what each segment did to build the SNHU website. The testers reviewed the developer's current code product for the website near the end of each Sprint to test its functionality, determine what features and functionality still needed to be implemented, and catch any bugs or unexpected behavior that the developers missed. The Scrum Master helps to define Sprint goals for developers and testers to accomplish as well as held meetings such as the Daily Scrum that provided a space for team members to communicate their progress to other team members and discuss any issues or concerns they may be currently having.   
 The Scrum-agile approach to the SDLC helped each of the user stories come to completion through the use of compartmentalizing. In particular, the user stories were compartmentalized so their details and requirements were organized and could be clearly understood and user stories were distinct from each other. Each user story was listed in a separate row of an Excel spreadsheet which listed the user story itself and what the client was communicating through the user story. Specifically, each section for each user story in the Excel spreadsheet was to extract information about what exactly was the client's requested feature, what kind of role the client wanted to play with their requested feature, and what the client wanted to accomplish with their requested feature. With Scrum-agile, every user story was distinct and clearly defined, and the purpose of each user story's requested feature and its function was well known and could communicated to every team member through a simple Excel spreadsheet. Compartmentalizing with an Excel spreadsheet made each user story easy to understand.  
 The Scrum-agile approach supported project completion when the project was interrupted and changed direction by adapting the newest Sprint cycle to incorporate the client's new request. When the SNHU Travel client requested a change to the SNHU Travel website by requesting the travel packages recommended to the user have a wellness/detox theme, instead of rebuilding the entire project's vision to revolve around wellness/detox, only the travel package recommendation feature was focused on. This focus took the form of a modified user story where the feature to be developed was still one whose purpose was to recommend travel packages, but this time, the feature would be modified to only recommend packages with a focus on wellness/detox. This gave developers very specific instructions on how exactly to fulfill the client's new requests without having to get rid of all of the code of the previous package recommendation feature. Simply put, all the developers needed to do was shift the focus of travel package recommendations onto wellness/detox package recommendations and modify nothing else. This is an example of the agility of agile because by putting user stories into small production cycles, the team can turn a change in direction into a small modification by turning the requested change into a user story to determine what needs to be changed and what can stay the way it is to meet the client's new request.  
 One example of my communication with my team was when we were discussing our contributions and questions for other team members regarding the roles we played in developing the SNHU Travel product. A sample of our communication can seen in my response to one of the points that the Product Owner made in my discussion post regarding whether strictly adhering to timeboxes is necessary.   
The Product Owner had made this point regarding timeboxes:  
"I believe that those timeboxes break up the flow of the team and create more meetings than is necessary for a single project when the communication and feedback that could be in a sprint review or sprint retrospective could just be included in the daily standup, or meetings called as-needed, such as if the customer's changed something vital that the team should know right away."   
I responded with:  
"Now that you mention it, I do realize that timeboxes sprints and the meetings that are required at the end of each sprint really do have the potential to mess up the workflow of the team and waste time with unnecessary meetings."   
 This was effective in its context because, in this instance, the Product Owner and I had agreed about the unnecessary nature of rigid timeboxes that disrupt workflow. The Product Owner was successfully able to show me how putting so much emphasis on working within timeboxes can decrease productivity, and I was able to easily communicate that I now agreed with her.   
This encouraged collaboration among team members by helping me see timeboxes in a new light. I had not considered the possibility that timeboxes can disrupt workflow by forcing team members to stop their work at specific intervals. However, the Product Owner's comment helped me to see how other team members may be negatively affected by strict adherence to timeboxes.  
 Some of the organization tools that helped my team be successful were spreadsheets, forums, and Scrum boards. Spreadsheets helped my team organize groups of similar yet distinct items such as user stories and their details. Forums were useful as a way to communicate remotely between team members. One feature of the forums used to communicate that was particularly useful was the notification functionality, which informed team members in real-time of responses to their posts, so they could quickly clear up questions and concerns other team members had for them. Scrum boards were useful during Scrum events such as Daily Scrums to give the entire team a visual representation of the work that needed to get done to complete the product. Scrum boards organized work into three columns. The first column consisted of tasks that did not have work completed on them yet. They still needed to be assigned to one or multiple people. The second column consisted of tasks that had work completed on them but were still incomplete. These were In-Progress tasks, and they informed the team which tasks were already taken by other team members, so no two team members would waste time working on the same task separately. The third column consisted of tasks that were already completed. The team did not have to worry anymore about these tasks. Some Scrum-agile principles that helped my team be successful were "Working software is the primary measure of progress" and "Our highest priority is to satisfy the customer through the early and continuous delivery of valuable software.". The first principle helped my team to be successful because it gave us a way to concretely measure how much progress we made on our product. With this principle in mind, the more working features we had, the more visible progress we were making on our product. Being able to see what progress looked like assured team members that their efforts amounted to something. The second principle helped my team be successful because following this principle not only kept the SNHU Travel clients in the loop but also gave them chances to provide feedback if they wanted to change anything about the current working version of the SNHU Travel website software. This way, the team had a better understanding of what the client wanted so there was less risk of the client ending up disappointed.   
 The Scrum-agile approach was very effective for the SNHU Travel project. The pros of this approach were it made work more manageable and offered more flexibility. This approach made work for the SNHU Travel website more manageable by breaking it up into pieces. The development process was split into short production cycles called Sprints. Each sprint had a small sprint goal that had to be met within the sprint's timebox. The breaking up of the overall goal of a fully functional SNHU Travel website into smaller goals that one or two team members could work on each turned what would normally be an overwhelming user epic into production cycles of user stories that were much more manageable. Another pro of this approach was that the Scrum-agile approach offered more flexibility. SNHU Travel website specifications could change and these changes would simply become new user stories that could be worked on in small groups instead of forcing the entire development team to change their vision of what the website was. The main con of this approach is that the big picture of the final product may become unclear to individual team members. When progress on the product is broken up into Sprint goals, it can be easy for one particular team member to lose sight of the big picture. If they are only completing the work for user stories assigned to them, they may forget that their work is supposed to be part of a whole, which in this case is a functioning SNHU Travel website. This could lead to them completing user stories on a technical level, but in losing the big picture of needing their features to work with the features of others, end up having their features break or become incompatible with the features that other team members have developed.   
 The Scrum-agile approach was the best approach for the SNHU travel development project. This is because the Scrum-agile approach allowed for flexibility made on the SNHU Travel product that would not have been possible following a Waterfall methodology. This was seen particularly with the flexibility that the team had to demonstrate when their client slightly changed their expectations for the product. Near the end of development, the main client, SNHU Travel, discovered new information that changed their vision for the product. They wanted the travel packages that the website would recommend to the user as "hot deals" to have a particular wellness/detox theme. This kind of change, although not as drastic as it might've been, would've required going back to rewrite sections of the documentation if it occurred during development under Waterfall methodology since documentation must be very precise and up to date in that methodology. Developers would've also had a larger amount of pressure put on them to make the changes to the code that the client requested since they would've been nearing the end of the development cycle with most of the code already developed yet still haven't sent their code out for testing. This would have likely put even more undue stress on both developers and testers since they only have one opportunity to get debugging and product testing right under the Waterfall methodology and some of their precious time has been used up making the client's requested changes.