Sprint Review

Roles

In our team, we had a Product Owner, a Scrum Master, several Developers, and several Testers. All the team members were active during all steps of development in one way or another, and each played a key part in the success of the project.

## Product Owner

The Product Owner was responsible for many of the non-labor parts of the project, and it was their work that made sure the project had meaning. Their communication with stakeholders and project backers ensured the project was on track to meet their vision. They also worked with marketers to make sure the product would reach a target audience. Once a backlog of features was defined, it was the Product Owner who organized and prioritized its contents, ensuring features critical for the project’s success criteria were completed before less essential features.

In this project, we saw the Product Owner talk directly to clients and collect details about their vision. While other team members were present for these meetings, it was the Product Owner leading the conversation and sharing their vision.

## Scrum Master

The Scrum Master was responsible for organizing much of the labor with the project, and their mastery of logistics and direction ensured the team stayed focused. They organized Scrum Events every day so that the team members could regularly touch base and coordinate. They created and enforced an information radiator at the center of the meetings and work environment so that no detail was forgotten or neglected. They worked directly with the Product Owner to ensure that the vision of the product was maintained, and they worked directly with the Developers and Testers to pass the vision on and coach them through their sprints.

We saw the Scrum Master at work with their communication to the Product Owner after a change in focus. Their efforts minimized lost time and labor on the team’s behalf and ensured the team could change to the new goal with little friction.

Developer

The Developers were responsible for most of the labor in making the product, they were the ones coding the product. Their knowledge of programming is of course crucial to creating a working product. Also, they created the backlog of features by itemizing what the product vision was into smaller, more workable parts. They attended each Scrum Event and stayed in regular communication with the whole group to update everyone on their progress. They maintained a reliable working pace to ensure the team’s velocity was consistent and their own health wasn’t neglected.

Of course, the Developer’s work is most visible in the fact that the project saw completion. No roles were assigned to individual developers, just tasks. At early Scrum Events, Ryleigh took the task of building the UI for what was initially a list and Patrick took the task of developing the backend where the vacation packages were stored. After the change in focus, Ryleigh took the task of designing the slide show and Patrick took the task of researching new content.[1]

Tester

The Testers were our assurance that the product was not only working, but exactly what was supposed to be made. They, of course, took the usual quality assurance of testing all features of the product to make sure they worked correctly. Even more importantly, though, is their user stories. Their work emulated customers using the software with specific goals the Product Owner wanted them to achieve. They worked with the Developers to define the scope of their backlog items, and they worked with the Product Owner to provide insight into their prioritization decisions.

The Testers worked to create a spreadsheet of user stories complete with a level of importance and a predicted scale of the features the user story claimed to need.

# How Scrum Helped User Stories

User stories are a huge part in Scrum development. In any development environment, you are designing a product for a user, and understanding how the user will use your product is integral to design. Scrum, as a system, encourages direct and ongoing collaboration between developers and their clients. With most traditional “Waterfall” systems, it could be expected to have an initial meeting with a client, they set the expectations for the project and the time/money budget, and you might not see them again until delivery day. Because Scrum encourages building that relationship between developers and clients, you get a more detailed insight into their desires, and can even help foster the vision of the project by understanding why they want things a certain way.

In this project, we can use User Stories as an example of understanding the client’s wishes. We were given the initial stories from the client about users wanting personalized slide shows and built that out to potentially include budget restraints and base the presentation on how the client has previously interacted with the service.

# How Scrum Helped During a Focus Change

As previously mentioned, Scrum encourages a relationship between developers and clients. It is expected that the Product Owner will be in regular contact with the client to ensure their expectations are being met. Not only does this ensure that the project never deviates or wastes effort on the wrong focus, but it also ensures that the development team is following the clients' wishes even when they should change.

We saw our clients change their focus from resort destinations to detox and wellness programs. Thankfully, this focus change was relatively small and only required extra content research, and most of our code base was still usable. Because our Product Owner was communicating with the client and the change was communicated to our team as soon as they knew about it, we got these changes made as fast as possible with minimal wasted effort.

# Communication Samples

An email sample from the Scrum Master to the Product Owner after the focus change to detox programs:[2]

Product Owner Jon,

I understand that we’ve changed focus in our project. I have a few questions about the product to ensure that we can meet the new success criteria:

1. Do we need to change our marketing approach? Is this still advertised to regular vacation-goers, or a new market of people who want to enjoy a tech-free retreat?
2. Our team has already conducted research on several vacations that don’t meet the new “detox” goal, but might still be deemed as attractive vacation destinations. What can we do with this work?
3. Our testers would like new success criteria to make sure we are on the right track. Can we schedule a meeting with them to discuss new testing criteria?

# Tools That Assisted in Scrum

Our team functioned as well as it did thanks in part to our information radiator. Our team utilized Microsoft’s Azure Boards to maintain a digital information radiator that was used as the central part of our meetings and updated as needed to reflect task completion. The documentation this software allowed ensured an equal sharing of burdens and credit given where credit was due. Azure Boards’ dashboards also helped the Scrum Master easily determine the team’s velocity, allowing them to accurately forecast how the team was performing. This software also aided in maintaining and prioritizing our backlog, which made sure no task was left behind.

# Scrum Pros and Cons

Pros to using a Scrum development model:

* Self-managing teams can allocate tasks to the person who can best complete them.
* Well-organized distribution of effort.
* Continual contact with clients ensures:
  + Shifting client goals can be met,
  + Existing goals are as close to their expectations as possible, and
  + Discussions can maintain expectations regarding delivery date.
* Denial of specialized roles maintains every team member on the same tier
* Regular in-team communication keeps everyone up to date
* Information radiator maintains visualization of progress.
* Modular development ensures functionality at early stages and constant QA testing.

Cons to using Scrum, compared to a Waterfall model:

* Reliance on communication and cooperation. One or two uncooperative team members can sink the project.
* Not newbie friendly. A new developer will have difficulty understanding what tasks they can claim, and a new tester may not properly understand how to set success criteria.
* Initial adopting difficulties:
  + Cost of adoption. Initial investment in new facilities and software for meetings and information radiators.
  + Risk of adoption. Team may have difficulty adhering to new meetings and maintaining information radiators.
  + Old ways. Team may just perform better in a Waterfall model, making a change unnecessary.
* Client difficulties:
  + Some clients may be unreachable or impossibly busy to maintain relationships with, or even just unwilling to do so.
  + Clients may misunderstand the development process, making such a relationship difficult for the Product Owner and risk adding hardship to the team through meddling.
* As development progresses, additional features may be envisioned and added to the backlog, potentially leading to a positive feedback loop of ever-more development.

[1] Names of actual students in the group who claimed the role of Developer, but tasks fabricated for the purpose of the document.

[2] Email draft existed as is, but was not really sent to the person it is addressed to. The name of the student is the one who claimed the role of Product Owner during the next week’s discussion.