**Applying Roles**: Each role in a Scrum team has its own essential responsibilities, and effectively applying those roles is important to running an effective team. The first of these roles is the Scrum Master. The Scrum Master is responsible for the overall Scrum process, and coaches the team, encouraging constant improvement. A Scrum master also removes obstacles from the team to remove outside interruptions. During my week studying the Scrum Master role in class, I talked in my journal entry about the importance of balancing the numerous meetings involved in Scrum and respecting people’s times. From my experiences that week I learned the best way to do this is to adhere to the schedule, even if people show up late, you should always start and end meetings on time. Secondly encouraging the teams to be self-sufficient. While, as a Scrum master, you are organizing all the meetings for everyone, it isn’t a “boss” position, and your goal is just to facilitate the process.

The second role I took was the Product Owner. The Product Owner represents the stakeholders and is the voice of the customer. They define the product backlog and work with the team to set priorities. They also get feedback from the stake holders about their view of the project. They do this by taking interviews with stake holders and turning that into user stories that can be developed into trackable metrics for how a project is coming along. During my experience writing from the project manager’s perspective, I gave an example of how developers of a project can grow blind to the views of the ones how will be using the project. The user stories help give reference to the actionable items by helping to describe to the development team the situation they would be used in. After all, the stakeholders are the ones who have commissioned the project, and potentially the ones fronting the money for it.

The third role we played was that of the Tester. The Tester works actively with the Product Manager and Development Team to turn user stories into trackable Unit Tests that can be performed to ensure it is in line with what the user is looking for. During my time writing about them for class, I found it tricky to break the user stories down properly into small chunks, as I was never sure if I was adding too much or too little. By the end of the week though I came to respect how important the role was as it helped when developing smaller work chucks for developers.

The final role was that of the developers, the work horses of the team. The developers turn the user stories into work and utilize the tester’s unit tests to ensure compliance with stakeholders’ demands. My experiences when I was writing about developers during the class were positive because I wrote about the importance of self sufficiency in the team. According to Scrum, a dev team is supposed to self-organize so it is up to them to decide what is and isn’t doable during a time frame. Though the product backlog is managed by the Product Owner it is up to the development team to organize how it gets handled during a sprint to not only be efficient but also ensure that junior programmers are involved to help train them up.

**Completing User Stories**: The Scrum-Agile approach to software development helps user stories by continuing to generate them as needed. When comparing to the waterfall method specifically, agile makes the development process an open dialogue with the stakeholders and can pivot as demands change. In the examples I did for class, I saw how the tester specifically responded back to the product owner to ensure clarification of user stories as well as attain more detailed information. We also saw how the development team was able to use those unit tests as direct rubric to ensure completion of the product. It shows that completing these stories is a multi-layered endeavor but made easier by the proper division of responsibilities across the various Scrum roles.

**Handling Interruptions**: The Scrum-Agile approach supports project completion by being flexible enough to handle changes and interruptions. The cyclical nature of the sprint system provides the ability to stop occasionally from one’s work and take stock of the direction the project is heading in. As stated earlier, Scrum makes it an open conversation between the stake holders and the team so if the stake holders are looking for something that changes slightly, the team can easily acquiesce to the request. Using an example from the work we did, the stakeholders, during one of the assignments, wanted to shift specifically to spa and wellness focused packages. Because the user stories the team has been working on so far kept a modularity to the project, switching from what was already worked on so far could easily be done.

**Communication**: Communication is important in any team but is essential in a Scrum Team. The entire process is built on different team members supporting and communicating with each other. In the work we did one of the most direct examples of inner team communication was in the email between the tester and the email. A short excerpt from the email that I wrote in Assignment 4-3 follows:

To: Christy (SNHU Travel Product Owner)

Subject: User Story Clarifications

Dear Christy,

Having looked over the user stories, I am beginning to develop pass/fail test cases for the various features requested. More clarifying information would be helpful for some of the stories.

**User Story Two**

* How far back would the users like the site to analyze?
* Do they put more weight on type of vacation or location?
* Do they want this feature to be a toggle on the normal tope list instead?

Very Respectfully,

Alex Hitchens

This example shows how effective communication encourages collaboration between team members. It is polite and succinct while also seeking more probing information. Email is just one way to communicate though, and the Scrum-Agile method has multiple ways for team members to communicate with each other. For instance, during a Daily Scrum issues can be communicated across the team and potential obstacles can be easily fixed.

**Organizational Tools**: A Scrum system has many tools both traditional and digital. First is the variety of meetings. Scrum has various meetings which are tools for planning and communicating. Sprint Planning and Spring Retrospectives both help organize what the sprint cycle will entail and how it can be improved in the future. Daily Scrum meetings are short meetings just used to synchronize the team and keep everyone on track. Backlog Refinement meetings prepare and clarify items in the back log for future sprints and make sprint planning easier. Other tools at your disposal are things like information radiators and pair programming. Information radiators provide places where the team can monitor and post progress on a project. This can be a whiteboard or a digital version like Trello or Jiro. Pair Programming was the tool I wrote about frequently as it was something I use to teach new Quartermasters how to operate the navigational software we use in the Navy. While it is not “programming” specifically, and we didn’t shift roles, it was a similar model and allowed me to be the “Navigator” while the junior sailor did the actions. This gave them muscle memory and training while still allowing them to have some freedom to explore the software while I directed them on how to operate it. The effectiveness of these tools is all very circumstantial however as you may have a team that does not respond well to things like pair programming. Another personal example is my wife whose team started utilizing an information radiator and found it was not as effective as an email system they had been using prior to. Because of the flexibility of Scrum/Agile this can be brought up in a forum like a Sprint Retrospective and potentially improved upon in the next sprint.

**Evaluating Agile Process**:

Pros and Cons of a Scrum-Agile Approach to SNHU Travel Project:

*Pros*: Using a Scrum-Agile approach, the work was divided into manageable chunks for each week/team role, and each team role helped in some way to continue the process. Dividing user requests into workable pieces through user stories helped when it came to development, while analyzing each task down to unit tests helped to ensure it was working as intended. Also, because we were working with user stories the process was kept modular, so we were easily able to pivot when the company wanted to go in a slightly different direction.

*Cons*: The cons, in my opinion, were minimized by the fact that this was a simulation of a Scrum environment. My biggest concern about the Scrum-agile process is the overhead when it comes to planning versus actual production. My question is always, are we producing more work for ourselves by implementing it? I think it depends on the size and the effectiveness of the team. Other than that concern there were no real cons to using this process for this project.

*Approach*: I feel the Scrum-Agile approach was the best option for this project. While the project was small it still had several moving parts. From the perspective of a Scrum master at Chada Tech, it served as a good starting point for introduction to a Scrum Environment to the team. From the examples we read about in *The Project Manager’s Guide to Mastering Agile,* some of the biggest initial challenges to setting up an agile-Scrum environment is winning over the people who will be using it. This was a good introduction to this process. Secondly, while the Waterfall method is good if the project needs planning upfront or has a well-defined requirement, the interviews by the Product Manager revealed that the Stakeholders were making up some of the requirements up on the spot. This indicates that the fast and flexible nature of a Scrum-Agile system was the right choice.

**References:**

Charles G. Cobb. 2015. *The Project Manager’s Guide to Mastering Agile : Principles and Practices for an Adaptive Approach*. Hoboken: Wiley. https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,shib&db=nlebk&AN=937009&site=ehost-live.

Ghahrai, A. (n.d.). *Agile testing mindset and the role of the agile tester*. DevQA.io - For Developers and QAs. https://devqa.io/agile-testing-mindset-tester-role-agile-team/

Ravi. (2020, August 2). *Do’s and don’ts for Scrum roles*. DEV Community. https://dev.to/ravimaharjan/do-s-and-don-ts-for-Scrum-roles-2i6o

Sharma, Jatin. 2023. “Pair Programming: Best Practices and Tools.” *DEV Community*. Retrieved October 9, 2024 (https://dev.to/documatic/pair-programming-best-practices-and-tools-154j#myths-about-pair-programming).