During this course, I took on various roles in an agile Scrum team. I went into this course knowing nothing about Scrum, so it was to my surprise when on the second week of this course I assumed the role of Scrum Master. A Scrum Master is the facilitator of information for the development team. The role also includes clearing any obstacles, creating a productive environment, and addressing team dynamics. In my experience, I found the most useful tool of a Scrum Master was the daily Scrum ceremony. This is a short meeting where everyone addresses how far they’ve come on their responsibilities in the project and what stands in their way. The Scrum Master can then use that information to make work easier and the team more effective.

I also took on the role of product owner. The product owner oversees the entire project and is the only role with any authority on the team. The product owner communicates with the client about their needs and comes up with “user stores” which are basically features of a program. The user stories are usually determined by the users when they ask for features. The product owner can acquire these stories in many ways. For example, I found that one tool for doing this would be holding a focus group.

I learned a lot when taking on the role of tester. The role of tester works directly with the information provided by the product owner, i.e. the user stories. With the stories the tester can make test cases that have a pass or fail metrics attached to them, which is the major role of the tester. With these test cases, we determine if the features work. I found that the more detailed the test cases, the better, as it ensures a quality feature makes it to production. Something interesting about testing on an agile team is that it happens concurrently with development.

As a developer on an agile team, I found that I needed to be particularly adaptable to change. My responsibility, of course, was to program and develop the features needed in the user stories. This meant when a new feature was changed, I had to pivot, even if that meant starting over. I found that a huge part of performing this role effectively meant having open communication to everyone on the team. If a developer did need to start over, for example, it would be crucial that he/she were able to talk to the product owner about possible code that could be recycled, and to the tester about rebuilding test cases to match your new code.

In a waterfall method, these roles would be a bit different. Since development in the waterfall model is linear, I find it less adaptable to change for larger projects. Each step in this model requires the completion of the step before it. This means that, for example, if we needed to change a feature in the middle of development, we would need to start the process over, gathering the requirements and completing the design again before development could take place again. Not only that, but until development is completed, we cannot test, and until testing is completed, we cannot deploy our software.

Both agile and waterfall approaches have their uses, however. Agile is preferred when the customer is needed throughout the process or when large changes are expected. Waterfall is good when the project scope is already known in advance. Waterfall can also keep a team better focused when working on smaller projects. Since it’s an all-or-nothing approach, that can get risky when working on larger projects. Agile is iterative and can release features little by little.

There are several organizational tools that can help facilitate an effective agile team. Communication is by far the most important aspect here. It permeates every part of the team. Face-to-face communication is vital when possible and should be emphasized. The daily stand-ups provide this as well and provide a platform for solving the day’s problems collaboratively. There are also methods such as planning poker, burndown charts, or other types of information radiators that teams can use to provide input or see where they are regarding a completed product. Of course, there are organizational project applications that most teams use to keep track of various tasks. There are several to choose from with features that help with project planning, project backlogs, and sprint planning.

**Communication example:**

**SM:** “Great job on that project team! We met most of the customer expectations on time! How can we improve for next time?”

**Developer:** “I found that for the most part it was smooth sailing. I was able to work well with our tester to ensure my code worked. I did find that the user stories were a little vague, however. It took both of us to decipher it.”

**Tester:** “I agree that the user stories were a bit vague. It would mean less guess work for us if they were more specific next time.”

**Product Owner:** “I didn’t realize they were so unclear! I will ensure that next time they better. Also remember that if there is an issue to bring it up in the daily scrum so we can tackle the issue immediately! Great job team!”

This was an example of a project retrospective. This happens after the deployment of a product and ensures that any final issues are addressed so they can be taken care of in projects going forward.

During this term, we were able to look at the methodology of an agile team under a microscope. Each part of the agile process takes place concurrently with each other. Mainly, analysis, design, development, and testing. Since I took on each role in this project, I produced a charter that gave a basic vision for the project and the expected outcome. I was also able to complete a product backlog that effectively listed the features needed and what each would need to be successful. This helped me develop test cases as a tester, because I was able to look at the user stories and determine whether the code would pass or fail. And finally, as the developer I provided the application itself.

The agile process really shined in this project when we were required to change direction. Because of the structure of this type of team, we were able to start fresh on this new feature with little hesitation. Because of the open communication structure, we were also able to determine that some code could be recycled, which saved a lot of time.

We went over all the tools that I had already mentioned above. I found the most important to be the daily scrum and open communication structure. This ensured that everyone was self-managing and that issues were taken care of immediately when they arose.

**Communication Example:**

**Product Owner:** “Okay team, I have provided the SM with the information that he needed to help us get this project rolling. He has provided a project charter for all of you to look at. I have also provided a list of user stories, so we’ll get started immediately with development and testing. Our first sprint will end one week from now and we’ll commit what we have. Please let me or the SM know if you need anything.”

This is an example of a product owner addressing the team for the first time. This demonstrates how each phase works together concurrently and iteratively. The basis of an agile team.

I do not have any professional experience in the computer science industry currently. I do, however have some personal experience. This class has taught me the importance of having a structured environment in which to develop. I enjoy agile because it has the flexibility that most developers, myself included, need when working on a project, but it still offers the structure to fall back on to not get bogged down on minute details of a project or fall behind entirely. If this class is anything like it is in an actual career, my career choice was confirmed with being a developer as I found that role the most fun and challenging.

I would really like to gain a better understanding of some of the project management applications that are available to teams today. In large organizations, these tools are used daily and in order to be successful at my work, I will need to get familiar with them.

**Communication Example:**

**Developer:** “Hey, Bill, since you’re testing my code I wanted to see if I could sit in with you to see how your process works. I’m hoping this will make me a better developer and maybe make life easier on both of us going forward.”

An agile team encourages collaboration. In my example, I was opening a line of communication with the tester in order to not only learn more about the business and his role, but to become an even better developer and hopefully avoid problems later and maybe have less revisions.

My main goal as a developer is to become a better developer! I do want a job in a professional workspace, but I enjoy programming because to me, it’s like a creative outlet. Like playing music, or maybe woodworking, you come out the other side of all the time you put in with a product that you can hopefully be proud of. I would love to one day use technology to help others in a way that they don’t have today. Maybe one day I would like to own a business doing development or build a product that earned me passive income. Earning my degree and taking classes like these is how I am pursuing my dream currently!