Sam Alcosser

Professor Juan Arias

CMPT220L: Software Development 1

1 February 2019

Agile Development cycle vs Waterfall Development

There are multiple different versions of the SDLC, and each has its' benefits and its' shortcomings. Most follow the same basic structure of the SDLC of going from analysis, to requirements gathering, to design, to implementation, then testing, and then finally maintenance.

The waterfall development cycle has a very important shortcoming that almost everyone is aware of. The process is called a waterfall as the process seems to flow downward and downward through the process without any looking back. The steps of the SDLC are: analysis, determining requirements, designing, implementing, testing, and maintenance. Much like a waterfall, the constant progressive downward flow of the development cycle makes it very hard to "swim upstream". This causes problems when a client may come in halfway through the testing phase of development asking for more out of the product. For this reason, most companies do not use this development cycle anymore.

Agile on the other hand works in iterations. The cycle, as described on Lucidchart.com, of the process is a bit different. On an agile software development life cycle, the steps are a bit more dynamic. The agile cycle works in 10 day "sprints" where the developers are planning, developing, testing, assessing, and implementing whatever is to be completed by the end of the sprint. Differing from the waterfall process, the first step is to scope out and prioritize projects. This means determining what problems should be done when, and how long it should take. The second step is diagramming the requirements for the initial sprint, or time frame of working. To do this, the development team meets with the client to find out what the software should do. After this, the team makes many different diagrams that are used to get a clearer picture of what needs to be done. Next, the construction begins. The team works to create a working model of the software, knowing that it will not be the final iteration. Because of this, only the bare minimum functionality will be present in this version. Once this is done the team begins the process of releasing the product. This starts with testing the software and addressing any issues. Once the issues that are apparent are taken care of, documentation about the software is produced. This includes a UML diagram of how the software works so that the users can understand the product better. The next step is then to release the product. After this point, it is the production team's job to keep up with the software to make sure that the product is running smoothly until retirement. This last phase is where the system is taken out and replaced by a system that is not obsolete.

Overall, it is clear to see that the second cycle, agile, makes a lot more sense to use as it can be used to make different iterations of the same software within one development "cycle". Because of this, it is clear to see why the waterfall cycle is being phased out, and most bigger and newer companies are implementing agile cycles.

Works Cited

"The Stages of the Agile Software Development Life Cycle." *The Stages of the Agile Software Development Life Cycle*, Lucidchart, www.lucidchart.com/blog/agile-software-development-life-cycle.