CMPS 115

RAC 2

25.4.2017

ПР**Scrum + Engineering Practices: Experiences of Three Microsoft Teams**

*Scrum + Engineering Practices: Experiences of Three Microsoft Teams* is an article written by Laurie Williams, Gabe Brown, Adam Meltzer and Nachiappan Nagappan. I wasn’t able to find the year it was published in the article, but from googling the information I learned it was published in 2011 by ESEM '11 Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement. This article discusses important points of the Scrum methodology; it also discussed case study research and comparing the three different Microsoft teams and the way each team used Scrum.

According to the article, *Flaccid Scrum* “is a term coined by Martin Fowler to refer to teams that utilize only Scrum’s project management practices.” (Williams 1). In other words, Flaccid Scrum is when a Scrum team tries to adopt Scrum practices and Scrum principles. However, after some time, the team ends up not paying enough attention so the progress is slow and the code/product is a mess. This tends to happen because Scrum is a management oriented rather than engineering oriented, so management is very important. Good news, Flaccid Scrum can be prevented if thought through. One of the criteria to prevent Flaccid Scrum is Quality Gates (aka Done Criteria). Rather than monitoring and measuring what percentage of a feature has been completed, majority of agile teams user a binary “all or nothing” feature to complete tracking. However, this feature isn’t complete until it passes all the “done criteria” that was in advance established by the team. This “done criteria” can prevent the team from rushing through the design and implementation process which can lead to other problems.

The three Microsoft teams used a total of nine engineering practices along the Scrum framework. The four practices of these nine are: **(1)** Planning Poker, **(2)** Continuous Integration, **(3)** Done Criteria (Quality Gates) and **(4)** Peer Review.

* All teams used Planning Poker to estimate the amount of time to complete an iteration (user story 😊). Usually, Planning Poker is played by the whole team during the Sprint Planning meeting. The team discusses a specific iteration and the key and features to accomplish it. Then each member independently estimates the iteration and reveals the time; this is repeated until same times are revealed. Basically, Planning Poker enables the teams to have relatively low # if errors from the first try. One of the potential problems could be that “Planning Poker voting results in a deadlock when no consensus can be reached.” (Williams 5)
* The teams used Continuous Integration as one of the practices. It is a software development practice in which the team merges their work into the main build system on a regular basis. One of the potential problem could be that the builds and tests are unsuccessful; this causes the team to deal with a bunch of system and integration problems.
* Done Criteria (Quality Gates) was another process used by the teams; it’s an exit criteria for the team’s further work; it focuses on the quality instead of the quantity. The conditions state; all tests must pass, test code coverage at least 80%, must have documentation, non-test code can’t have any static analysis errors or warnings and build must compile without and errors or warnings. Potential problem – “impose overhead on their process due to the need for monitoring.” (Williams 6)
* Peer reviewing is when the team conducted the peer reviews of architecture diagrams and code. These reviews helped the team and improved the quality of the project by carefully analyzing each feature.

The authors provide evidence on which they build their conclusion that **combination of Scrum and engineering practices leads to improved productivity and product quality.** Basically, the authors contrast and compare the quality of software systems produced; they compared against prior published defect rates of non-Scrum managed project @ IBM. The results showed that the Microsoft Scrum-based projects had lower defect density rates than the non-Scrum project. “These results further back up our assertion on the importance of the engineering practices followed with Scrum (in this case more extensive testing) rather than the Scrum process itself.” (Williams 7)

Probably the most important point I took from reading this paper was that Scrum is an effective framework to use in developing any kind of project. I now know about the nine engineering practices of which I didn’t know before. This article was also helpful for me to understand the Scrum process in full detail.