Joseph Kabaso

Professor: Tuck and Bjork

Course: CPS-491

Date: 9/25/17



Chapter 3 starts with planning, it talks about the initial of the project. The developers and the customers identifies user stories, which are the primary development artifacts for project teams. These user stories are the requirements which contains more information, a developer would use to produce a reasonable estimate of the effort to implement the project. The author also talked about how developers underestimate large stories and overestimate small stories. user stories are small, much smaller than other usage requirement artifacts. However, they are simple enough to learn to write for the customers, so that it can make sense to the developers. The customers are responsible for prioritizing requirements, they also have the right to define new requirements. They can change their minds about existing requirements, and even reprioritize requirements which they think may work well. Therefore, customers can also be responsible for making decisions and providing information in a timely manner. When comes to developers they are responsible for estimating the effort required to implement the things which they will be working on and that would include stories.

Chapter 4 the author introduces Test Drive Development approach, which is the approach development that would combine test first development, where developers write a test before they write just enough production of codes to fulfill that test and refactoring. We can think about drive development as the way of thinking through our requirements or design before your write your functional codes. Basically, the author is making us understand that, Test-driven development is a way b where a developer must first write a test that fails before their write new functional code. Test drive development enables developers to take small steps when writing software, which can help to break large problem into smaller. He also discussed about Acceptance Test which he says are written by folks who do not know the internal mechanism of the system. Developers can write a single acceptance test, and the production codes to achieve the test.

Chapter 5 which talks about refactoring which Martin Fowler define as a process of changing a software system in such a way that it does not alter the external behavior of the code yet improves its internal structure. Basically, developers would improve the design of the code after it has been written.