Phakpoom Chinprutthiwong

Development and Deployment at Facebook

Dror G. Feitelson, Eitan Frachtenberg, Kent L. Beck

This paper talks about the software development at Facebook. Here are the key aspects of working as software developers at Facebook company.

1. Perpetual Development: Unlike typical software development procedure like waterfall, perpetual development supports never ending software development. Facebook grows super linearly and the software changes very often. A new development occurs daily at Facebook while waterfall does not have new development after the goal is achieved.
2. A/B Testing: Because new development occurs in short term basis, users can experience the changes and the developers can get the feed back almost immediately. This helps developers find bugs or performance issues or whether the users like it.
3. Continuous Deployment: This makes developers divide large problem into multiple sub-problems so that if something bad occurs, the scale of the error is not too big to cope with.
4. Pushing New Features: Facebook uses git and subversion to manage codes. There is only one main branch of the code, which reduces complexity of merging. However, before the code can be pushed, it must have gone multiple layers of checking. First, it has to be tested internally by the engineers themselves. Then, some users will get to test it by the Gatekeeper, which controls who can access that part of code. It also helps controlling bad code to spread if there is an issue. There are also some other tools to test including performance test, which is also one of the most important aspect of Facebook. Here in Facebook, they have no Quality Assurance team because they would need to learn the code, while the engineers themselves already know the code and can also do quality test.
5. Personal Responsibility: Each person is responsible for the code he wrote. This does not exist for pin pointing who is at fault, but to make mistake a learning experience.

The software development method they use in Facebook is pretty similar to Agile as it involves short development cycles and rapid interaction with users. I find it more efficient as developers can get feed back sooner as they can make change before they develop too far and it becomes hard to revert back. Therefore, I agree with what the author has presented.