Alexander Ramsey  
SWEN 603  
June 14, 2016

Extreme Programming Methodology

We live in a time where the market, especially the software/tech market, experiences major disruptions and changes overnight. These disruptions cause consumer demands to be an ever present battle for companies to achieve success and exceed expectations in order to continue to be relevant and competitive in the marketplace. This is an “Extreme” time we live in, where “Extreme” work must be done to keep up! This is where the Extreme Software Development methodology comes into play.

The textbook way of describing Extreme programming is that it is a process where releases of the software occur frequently, boosting productivity, while focusing on the core problems to be solved to meet the end goal. Extreme programming follows the traditional steps outlined in any basic SDLC methodology but speeds the process by introducing a nearly continuous code review process. This process can sometimes even be constant in the pair process of Extreme programming. The pair process entails two people working together to accomplish tasks iteratively, while checking the work done. In this way, Extreme programming is considered a form of agile methodology because it embraces change and the communication driven development of software.

Originally developed in the early 1990s by Kent Beck of the Chrysler car corporation, the Extreme programming methodology was more of a theory that based itself upon the adoption of best practices in the development process. In addition, it was also suggested by Mr. Beck that the development team pursue only what they thought was the most important thing to be accomplished within the span of time, while pushing code reviews and planning meetings to a less rigid state. This method became quite popular in the late 1990s and the early 2000s as the dot com era pushed competition to the brink (Beck, 1999).

The core processes outlined in the XP methodology include Coding, Testing, Feedback, and Design. By jumping into the coding process, the team can quickly determine where their roadblocks may be as well as how they must adjust their logic to best suit the success of the application. My integrating the testing phase with the coding phase, automated unit tests can speed up the process down the road. In addition, periodic Acceptance testing verifies that the product meets the overall requirements of the clientele. The Feedback stage is similar to the Acceptance testing in that the client provides feedback for usability and further expansion of the business logic to meet the needs of the client. The last process involved with successful extreme programming involves designing and architecting the system to ensure that future expansion and code sprints are effectively executed.

In addition to the core processes, Beck outlines some values to enforce among the team involved in the extreme programming realm. These include Communication, Simplicity, Feedback, Courage, and Respect (Williams, 2005). Feedback and courage are the two less apparent values here. Feedback as a value means that, “no fixed direction stays valid for too long.” This is explaining that the group cannot totally rely upon their own self-sustained direction and that sometimes communication is key to continuing in the right direction. Courage means that the team follows through with finding a solution despite the challenges inevitably faced during the software development process.

There are many visual charts that can help guide an XP oriented team including backlog graphing, pair matrix, time box games, and many more. These can help guide teams similarly to the Scrum methodology where projects are outlined in story boards. This helps to frame the mind of the team to orient themselves better to the vision of the project. There are many differing versions of the high level chart used to describe the overall process of XP. The most popular graphic involves a complete circle with lines that rope back to each of the processes at the direct end of the coding step. A differing chart found online looks similar to the following and describes the XP process well (in my opinion). A self-made render of this variant is below:

Execution

Analysis

Consolidation

Requirement

Release/Test

Requirement

Design

Iteration

Extreme Programming is all about embracing the iterative process in a way that encourages the use of quick thinking and determination to accomplish the end goal. The above chart shows that the initial requirements enter this process while design and analysis are a constant process (driven by the code). The ending step of this process is the release and test phase after a brief consolidation of the iterations to ensure code simplicity and readiness. Extreme programming lends itself well to environments where the people are driven to be successful, where the people are invested in accomplishing something. The pair variant of this methodology has been proven to show effectiveness, but only when those involved are well versed in the development environment utilized. Extreme programming will certainly be adopted by many as the future pushes innovation and the development of new ideas to new heights, pushing those who can dream up the next thing to make it a reality.

Resources:

Beck, K. (1999). Extreme Programming Explained: Embrace Change. Addison-Wesley. ISBN 978-0-321-27865-4.

Williams, L. (2005). Case Study Retrospective: Kent Beck’s XP Versions 1 and 2. [Presentation]. Accessed at http://csse.usc.edu/events/2005/arr/proceedings/presentations/OneDayWorkshops/agile/williams.pdf