Alexander Ramsey  
SWEN 603  
July 5, 2016

Lean Methodology

To remain competitive, organizations must balance providing quality, cutting costs, and reducing delivery times with achieving the end goal of providing the core values a customer is interested in. This is where the principle of Lean comes into play. Lean is well known to be a manufacturing principle which started at Toyota, where the end goal was to reduce waste while contributing to overall customer value (Poppendieck & Cusumano, 2012). In software, focusing on the end goal is an important aspect of remaining lean and agile so that future market disruptions, customer feedback, and other factors do not cause the product to veer off course and become irrelevant.

The Lean process is summarized by seven guiding principles: 1) Eliminate Waste, 2) Amplify Learning, 3) Decide as Late as Possible, 4) Deliver as Fast as Possible, 5) Empower the team, 6) Build integrity in, and 7) See the whole. Each of these principles provides an important angle for consideration in adopting the lean process (Ebert et al., 2012).

The most important principle that sums up lean well is Eliminating waste. Focusing on the end goal and functionality requested by the client at its most fundamental level is the most important guiding principle. Toyota believed that waste even caused the end product to have less than the desired result, because of the value lost in generating the waste of the product. This may include extra features, processes, and partial work. In order to eliminate waste, processes must be in place to identify and eliminate it iteratively throughout the development cycle.

The secondary principle states that learning should be embraced, with differing perspectives and feedback utilized holistically through the development process. One way to address this principle is with the use of feedback from the customer that focuses on the possible boundaries associated with functionality, rather than focusing on the possible solutions. This provides a facilitated environment for the brainstorming of the solution with the client, thus amplifying the learning process and increasing communication among the parties.

The next two principles relate to each other in that they serve to save time and unnecessary cost over the long run. Deciding as late as possible and delivering as fast as possible serve to reinforce the idea that simple code and simple architecture allow for flexibility to grow and adapt more so than advanced architectures and setup at initial stages. These principles relate well to the just-in-time manufacturing methodology in that the client requirements should be developed upon delivery. This also relates to the Agile methodology in that the teams work together in a common scope to deliver maximum value in a short amount of time. In Agile, it is tough to hide, which ensures speed and productivity among the team of developers.

Empowering the team allows for buy in from the development team. This concept is seen in many other development methodologies as well, where communication with the client triumphs over all other priorities for the team. By communicating with the client, the team can gain a better understanding of what the client feels is most important to them. This focuses the mind’s eye of the team and in turn makes them productive and autonomous, without the need for constant management attention. Similar to the Agile methodology principle of Active user involvement, building integrity into the process allows for the team to ensure that the delivery of results is iterative and open for analysis among all parties; management and client. Seeing the whole is the final principle of Lean. It essentially means to analyze the bigger picture of how interactions may occur between parts of the software.

Lean methodology does not necessarily have a diagram to coincide with it. However, due to its similarity with other “fail fast” methodologies, it may be beneficial to visually compare it with these methodologies in terms of their usefulness when injecting the concept of known vs unknown requirements. Where requirements are known, traditional methods work well to outline predictable outcomes. Whereas, when requirements are unknown or complex, methodologies like Agile and Lean perform best due to their ability to quickly get to the root of the problem.

Lean

Agile

Waterfall

*Known*

*Unknown*

*Unknown*

Solution

Requirements

Lean methodology serves as a good starting point for teams who may be entering the bleeding edge and unknown aspects of development, where real value is delivered in as short amount of time as possible. Lean is an excellent state of mind to be in because it focuses on being competitive while allowing flexibility to explore differing opinions and solutions to advanced problems.

References:

Ebert, C., Abrahamsson, P., & Oza, N. (2012). Lean Software Development. IEEE Software, 29(5), 22-25. doi:10.1109/MS.2012.116

Poppendieck, M., & Cusumano, M. A. (2012). Lean Software Development: A Tutorial. *IEEE Software*, *29*(5), 26-32. doi:10.1109/MS.2012.107. Retrieved at http://eds.b.ebscohost.com.ezproxy.umuc.edu/eds/pdfviewer/pdfviewer?sid=9c13ee78-1903-48c9-852f-66e721f33a82%40sessionmgr103&vid=2&hid=103