Bryan Cabrera
11/21/2024
Module 6.2 Assignment

Chapter 13 Case Study: Strangler Pattern at Blackboard Learn

The case study in the reading mentions how Blackboard Learn in 2011, the company behind widely used software for education, was dealing with a codebase that went as far back as 1997. This being said though in 2010 David Ashman who was their chief architect was mainly focused on how Blackboard's build, integration, and testing were consistently getting more complex as well as error prone. In short what ended up happening is that the larger that the project got over time, the longer it took for the development team to implement any changes and truthfully the worse the product/product updates would get for the customers. It was even mentioned in the case study that it would take 24-36 hours to receive feedback from their integration process.

Naturally this had a negative impact on the development team. As it was found that the number of code commits was decreasing, basically showing how it had gotten more difficult to implement any code changes to the project even though the overall amount of lines of code was steadily increasing. As a result of this negative impact, in 2012 David Ashman was primarily focused on a re-coding project of sorts that utilized the strangler fig pattern. Specifically this was accomplished by using what was internally at Blackboard called *Building Blocks*. This practice gave the developers the ability to work in separate modules that ended up being decoupled from the original monolithic code base. As a result, when building blocks were made available to the development team, the overall size of the code base started to decrease as programmers were able to write updates or changes in a more individual manner without having to worry about disrupting the original code repository. David Ashman even pointed out that if presented with a choice, that every developer for Blackboard Learn prefers to work in the building block strategy of code base and code deployment.

The building block strategy ended up being very important for Blackboard Learn as was mentioned the development team was able to decrease the overall number of lines of code in the project as they were able to make efficient updates to the code base. Due to these positive changes, the development team was in turn more productive as they were also able to increase the number of total code commits for their building block repositories as the overall number of commits had gone down originally due to the monolithic structure of the project. Lastly but potentially most importantly, the development team at Blackboard Learn as a result was also able to work in safer and less stressful manner since if any mistakes were made, they would only affect the local building block repository instead of impacting the global system which could otherwise have greater effects (the difference between a local failure and a major system-wide issue).