Bryan Cabrera
11/2/2024
Module 2 Assignment

Operation InVersion at LinkedIn Summary

The case study mentioned in chapter 6 was about how LinkedIn was able to greatly decrease technical debt they had prior to the IPO launch of the company. To be more specific though, LinkedIn initially ran their application called Leo which is a Java application that displayed every page through servlets and managed JDBC connections through different Oracle databases. What happened though is as LinkedIn began to grow, they separated a couple of services from Leo to be on their own. Eventually it got to the point where most new services were running independent of Leo with “nearly one hundred services running outside of Leo”. Because of this separation from Leo, Leo itself would only be deployed once every two weeks which led to issues with Leo itself such as being down in production. Leo being down would then lead to increased difficulty in releasing new code/features therefore, it was clear to engineering managers that this issue had to be fixed moving forward.

Next when Kevin Scott the VP of Engineering joined prior to LinkedIn’s IPO, the company had decided that these issues were becoming too much and Operation InVersion was created and launched internally. This operation consisted of completely halting new feature development until the company’s computing architecture was entirely revamped or in other words until the technical debt was dealt with. Naturally it is a risky and hard decision to make but the engineering team at LinkedIn ultimately decided to not deliver any new features for a total of two months in order to fix all of the prior issues.

Ultimately this tough decision ended up paying off for LinkedIn as the results of Operation InVersion included not only reducing their existing tech debt but also they were able to develop and improve their tools used for infrastructure and deployment. While the outcome for LinkedIn was positive in the end things could have easily gone wrong at such a crucial time for the company potentially leading to its downfall. There are very important lessons to be learned from LinkedIn’s tech debt situation though, for starters it’s crucial to not let large amounts (or any if possible) of technical debt accumulate as this can lead to headaches or even “near-death experiences” for organizations. Furthermore an important lesson to consider is how prior to the tech debt being fixed, it’s mentioned in the case study that it was common for engineers to work late nights simply trying to fix issues and push new code however reducing the tech debt fixed this as well. Therefore reducing tech debt reduces the need to overwork one’s developers which will of course contribute to better attitudes and better work-life balance.