Kody Pope

6-8-24

Assignment 2.2

Case Study: Operation InVersion at LinkedIn (2011)

A Brief Summary (of the main points the author made):

Both (Kim et al.) and (Kim) share with us that in 2011, LinkedIn IPOed, and shortly after the IPO, a senior software engineer at the company named Josh Clemm expressed that their Java server Leo was constantly being overloaded and caused the server to crash. These crashes were so significant that they caused many errors and caused the company to spend long hours, long into the night, to fix these issues. Even though the issue was fixed, it wasn’t really fixed, as more problems continued to arise due to the increased number of users using LinkedIn’s service.

When LinkedIn started, they didn’t have many people using their product. Still, after the IPO, they suddenly had millions upon millions of people using their product every second, and it was clear a change needed to be made. Another senior engineer at the company, Kevin Scott, had decided that all the necessary work had to stop and that the entire company had to reframe its whole software system. For two months, every software engineer with the company had focused on creating smaller and more manageable services instead of one. Once those were completed, normal operations would continue. Now, after these changes, the normal operations consist of creating a small change. Then, many automated systems could check the code for problems and errors and automatically push the new code to the LinkedIn servers. Now, the company could focus on adding new features to its site rather than taking long days, weeks, or even months to fix minor issues. The company would see a reduction in issues and could now add almost three new features a day.

Lessons Learned:

The article (Kim) shares three key lessons learned from the situation. The first lesson learned here is slowification. This is the idea that even though the company had just IPOed and everyone was not watching them, change needed to be made rather than trying to fix current issues while developing a whole new system entirely. Instead, the entire company needs to slow down so staff can reconsider and restructure their product to meet the demands of today and prepare the product for the future.

The second lesson learned here is simplification. Simplification here refers to the fact that they had one web server called Leo that did everything every second of every day, and this caused the server to crash and encounter many errors that were just too hard to fix. Still, they have simplified the process at every point of their product by having many smaller servers. Instead of one server handling everything, they have various servers on hand that only handle more minor tasks.

And the last lesson the article points out is amplification. Because one server did everything, one minor issue became amplified even though it should have been something small and minor that only affected one or two people worldwide. It was an issue that would stop everyone from having success everywhere, even though it should have been a minor issue. Take, for example, we are trying to cook a meal for 20 people. And all the dishes relied on the oven. If we were trying to cook ten different dishes to serve these 20 people, and we only had one oven, and if the oven suddenly stopped heating halfway through cooking the meal, we have now created this massive issue of reducing what can now be served to these 20 people. Still, if we were to have multiple ways to cook the food, meaning a few dishes could be cooked on the stove, a few in the oven, and a few in a pressure cooker, then all of a sudden, if the oven stopped working, instead of having a significant amount of the meal no longer being able to be cooked, we would only have just a few uncooked dishes that required the oven to not be served reducing amplification within our meal/company/product.

Works Cited

Kim, Gene. “Case Study: LinkedIn’s 2011 Operation InVersion through the Lens of Slowify, Simplify, and Amplify.” IT Revolution, 13 Nov. 2023, itrevolution.com/articles/case-study-linkedins-2011-operation-inversion-through-the-lens-of-slowify-simplify-and-amplify/.

---. The DevOps Handbook : How to Create World-Class Agility, Reliability, and Security in Technology Organizations. Portland, Or, It Revolution Press, Llc, 2017.