Jason Palmeri
Assignment 2
CSD380
Professor Longely

Case Study: Operation InVersion at LinkedIn

LinkedIn's Operation InVersion presents an interesting case study illustrating that paying down technical debt is a part of the daily work. Months after LinkedIn's successful IPO in 2011, they found themselves struggling with deployment issues that became so painful that they launched "Operation InVersion", which stopped their development for two months. Founded in 2003, LinkedIn found themselves growing massively every year, from 2700 members in the first week to over 350 million in 2015. With so many users there were millions of queries per second happening on LinkedIn's backend. LinkedIn's systems where primarily ran on Leo a Java Backend using JDBC connections to various databases. By 2010 LinkedIn was struggling, adding more memory and CPU was not enough to keep Leo up, and they found themselves often having production go down, being "difficult to troubleshoot and recover, and difficult to release new code…". With late nights ravaging the team at LinkedIn it was time something was done. Kevin Scott launched "Operation InVersion" in 2011, completely stopping all engineering work on new features, and focused the team on fixing the site's core infrastructure. While the downside was having a bit of a PR mess, LinkedIn benefited immensely, Ashlee Vance from bloomberg goes on to say ***"LinkedIn created a whole suite of software and tools to help it develop code for the site. Instead of waiting weeks for their new features to make their way onto LinkedIn's main site, engineers could develop a new service, have a series of automated systems examine the code for any bugs and issues the service might have interacting with existing features, and launch it right to the live LinkedIn site".*** Josh Clemm talks about how scaling can be measured across many dimensions, organizational scaling being one of them, focusing on creating tools for engineers, and improving deployment and infrastructure. There's definitely a big lesson to be learned here, while scaling issues I feel aren't that prevalent now with cloud computing, and buy as you need services, it is still important to be on top of our "technical debt", and it's a good idea not to get too ahead of yourself, be sure to keep everything running, and the consumer happy.