CASE STUDY

Operation InVersion at LinkedIn (2011)

LinkedIn's Operation InVersion presents an interesting case study that illustrates the need to pay down technical debt as a part of daily work. Six months after their successful IPO in 2011, LinkedIn continued to struggle with problematic deployments that became so painful that they launched Operation InVersion, where they stopped all feature development for two months in order to overhaul their computing environments, deployments, and architecture.[11]

LinkedIn was created in 2003 to help users "connect to your network for better job opportunities."[12] By the end of their first week of operation, they had 2,700 members; one year later, they had over one million members, and have grown exponentially since then.[13] By November 2015, LinkedIn had over 350 million members, who generate tens of thousands of requests per second, resulting in millions of queries per second on the LinkedIn back-end systems.[14]

From the beginning, LinkedIn primarily ran on their homegrown Leo application, a monolithic Java application that served every page through servlets and managed JDBC connections to various back-end Oracle databases. However, to keep up with growing traffic in their early years, two critical services were decoupled from Leo: the first handled queries around the member connection graph entirely in memory, and the second was member search, which layered over the first.

By 2010, most new development was occurring in new services, with nearly one hundred services running outside of Leo. The problem was that Leo was only being deployed once every two weeks.[15]

Josh Clemm, a senior engineering manager at LinkedIn, explained that by 2010, the company was having significant problems with Leo. Despite vertically scaling Leo by adding

memory and CPUs, "Leo was often going down in production; it was difficult to troubleshoot and recover, and difficult to release new code. . . . It was clear we needed to 'kill Leo' and break it up into many small functional and stateless services."[16]

In 2013, journalist Ashlee Vance of *Bloomberg* described how "when LinkedIn would try to add a bunch of new things at once, the site would crumble into a broken mess, requiring engineers to work long into the night and fix the problems."[17]

By fall 2011, late nights were no longer a rite of passage or a bonding activity, because the problems had become intolerable. Some of LinkedIn's top engineers, including Kevin Scott, who had joined as VP of Engineering three months before their initial public offering, decided to completely stop engineering work on new features and dedicate the whole department to fixing the site's core infrastructure. They called the effort Operation InVersion.

Scott launched Operation InVersion as a way to "inject the beginnings of a cultural manifesto into his team's engineering culture. There would be no new feature development until LinkedIn's computing architecture was revamped—it's what the business and his team needed."[18]

Scott described one downside "You go public, have all the world looking at you, and then we tell management that we're not going to deliver anything new while all of engineering works on this [InVersion] project for the next two months. It was a scary thing."[19]

Vance described the massively positive results of Operation InVersion:

> 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 . . . LinkedIn's engineering corps [now] performs major upgrades to the site three times a day.[20]

By creating a safer system of work, the value they created included fewer late-night cram sessions, with more time to develop new, innovative features.

As Josh Clemm described in his article on scaling at LinkedIn,

> Scaling can be measured across many dimensions, including organizational. . . . [Operation InVersion] allowed the entire engineering organization to focus on improving tooling and deployment, infrastructure, and developer productivity. It was successful in enabling the engineering agility we need to build the scalable new products we have today. . . . [In] 2010, we already had over 150 separate services. Today, we have over 750 services.[21]

Kevin Scott stated,

> Your job as an engineer and your purpose as a technology team is to help your company win. If you lead a team of engineers, it's better to take a CEO's perspective. Your job is to figure out what it is that your company, your business, your marketplace, your competitive environment needs. Apply that to your engineering team in order for your company to win.[22]

By allowing LinkedIn to pay down nearly a decade of technical debt, Operation InVersion enabled stability and safety while setting the next stage of growth for the company. However, it required two months of total focus on non-functional requirements, at the expense of all the promised features made to the public markets during an IPO. By finding and fixing problems as part of our daily work, we manage our technical debt so that we avoid these "near-death" experiences.

This case study is a good example of paying off technical debt, creating a stable and safe environment as a result. The burdens of daily