Operation InVersion at LinkedIn

Bella Apo

Assignment 2.2

CSD380

January 15th, 2024

LinkedIn faced significant challenges in its early days due to its reliance on its monolithic core system, Leo. The system initially supported LinkedIn's growth but quickly became a major

crux as the company scaled. By the time that the company had grown exponentially, Leo was supporting over a hundred different services, but the website was only deployed once every two weeks.

The deployment process became increasingly problematic with each update. Despite adding more memory and CPU to the system, troubleshooting and recovery were inefficient and troublesome. Over time, it became obvious that Leo was not unable to support LinkedIn's growing needs. The system's inefficiencies caused frequent website crashes during deployments, requiring developers to work long hours to restore functionality. These challenges prompted the creation of dedicated teams to address infrastructure issues.

These challenges led to the creation of "Operation Inversion," which was a strategic decision to stop the development of any new features and focus resources on completely overhauling the existing infrastructure. This was a bold move on LinkedIn's part, requiring the company to take a step back and prioritize stability over immediate growth. The decision had some risks, as pausing feature development could have severely impacted user satisfaction and growth. Though, in the end, it was a necessary and crucial step for ensuring the company's long-term success.

Under the leadership of Kevin Scott, the team reimagined LinkedIn's architecture by breaking down the existing system into smaller, functional, and stateless servers. This shift helped to improve stability, simplified troubleshooting and helped to lay the foundation for more efficient and frequent deployments. The changes helped to resolve the existing and reoccurring issues and allowed the team to focus their efforts on innovation rather than fixing constant issues.

This case study helped to highlight some important lessons. For starters, ignoring infrastructure issues for too long can halt growth and create an unsustainable workload for developers. The amount of work required of the development team to fix significant issues at each update can cause excessive burnout, not allowing the team to put their best work forward. Their efforts are spent on extinguishing fire, rather than allowing them to be creative. In addition, it is important to take a step away from a project to address foundational issues. This is essential for future progress, even if it means putting a pause on immediate success. A system redesigned for scalability and resilience ensures that there are smoother operations and helps to support long-term growth. As the company continued to grow over the next decade, each update would have strained the team and the system. The system overhaul was inevitable. Finally, effective

leadership involves making tough decisions that prioritize an organization's future over short-term gains.

Technical debt, which refers to the accumulated extra work and complications that are caused by taking shortcuts in software development, is to blame for the shortcomings of LinkedIn's relationship with Leo. In the initial stages of the company, too many shortcuts were taken to support the rapid development and growth. This had allowed LinkedIn to quickly establish itself within the professional networking space but, these early trade-offs left the development team with increasingly obvious issues.

The lack of modularity in Leo's architecture had made it rigid and increasingly difficult to adapt to the company's rapidly evolving needs. Debugging and troubleshooting became excessive, with minor changes causing a ripple effect across the entire system. As time progressed, these bugs and inefficiencies accumulated, leading to frequent crashes. This underscores an important lesson regarding technical debt: cutting corners might appear to help the process in the short term but it leads to extreme long-term consequences. The team was required to increase the maintenance costs, and it cut the space for innovation. For LinkedIn, the choice between addressing the technical debt and allowing the issues to snowball was obvious, it was a critical and necessary step in ensuring the company's survival.

By addressing LinkedIn's technical debt and transforming its infrastructure, the company ensured its stability, safety, and capacity to scale with its growing user base. The decision to overhaul its existing infrastructure was detrimental to securing the company's position as a leading professional networking platform.

References

Kim, G., Debois, P., Willis, J., & Humble, J. (2016). *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. https://dl.acm.org/citation.cfm?id=3044729