Zachariah King

August 24, 2025

Operation InVersion at LinkedIn Case Study Review – Assignment 2.2

CSD 380 – Professor Sue Sampson

Case Study Main Points

A. Technical debt must be paid down as part of daily work.

  A. Even if that means stopping all new feature development and focusing on current computing, deployment, and architectural environment(s) rework.

B. LinkedIn, the exponentially popular networking application, used to primarily run from their own monolithic program named Leo.

  A. As traffic increased, services were being deployed to other applications so much to the point that in 2010, most new service development had nothing to do with Leo.

  B. Leo was hardly being deployed which led to its helpless decline in production, despite scalability efforts like adding memory and CPUs.

C. Leo needed to be laid to rest and replaced with more modular, lighter services.

  A. Kevin Scott is the VP of Engineering who made the call to halt new feature development and dedicate all efforts to fixing LinkedIn's core infrastructure first (Operation InVersion).

    A. Although necessary, this move was a bold one that included risk of backlash from executives and the public.

    B. It proudly produced a new suite of software development tools to aid in developing code for LinkedIn and keeping the value stream within a more confined manageable space, although it did take a couple months.

    C. Scaling is important and more than just adding memory and CPUs. It's also focusing on updated tooling, deployment strategies, architecture, and developer practices. This is how LinkedIn was able to grow from 150+ services in 2010 to 750+ services today.

D. Total redesign should never be taken lightly, but also should never be avoided if needed to create a stable, safe environment that’s staged to accept growth with a strong foundation of non-functional requirements. This leads to time teams can use to actually work on new features.

Lessons Learned

A. Monsters can only stay hidden for so long.

    A. Like the case study said, LinkedIn was already 100+ services away from Leo at the time they noticed it was not as compatible anymore. They were only able to make it work for so long.

    B. This is also a lesson for us to not sweep outdated code/practices under the rug out of laziness.

B. Modular, stateless services are more scalable than monolithic, retentive ones.

C. Thinking ahead will save immense time and pain later on.

D. Don’t be afraid of bold decisions if you know they’re right to make.

E. Work with integrity, always thinking about how to further the team’s progress and stability.

F. Focusing on too many fixes at once leads to more broken things.

G. Even the most popular services have moments of weakness, so don’t get cocky.