Brian Preston

CSD 380

Module 6

**Chapter 13 Case Study Summary — Strangler Pattern at Blackboard Learn (2011)**

Chapter 13 of The DevOps Handbook explores the importance of evolving architecture to reduce risk and increase developer productivity, especially as systems grow more complex. They introduce the concept of evolutionary architecture as well as the challenges organizations can face when their systems become so tightly coupled that even small changes can cause catastrophic failures. I feel this is a reality many companies face today, as we all strive to deliver faster results without essentially breaking everything along the way.

One of the most potent strategies described in the chapter is the "strangler application pattern," which helps safely migrate from old monolithic systems to newer, more flexible architectures. As Martin Fowler said, “They seed in the upper branches of a fig tree and gradually work their way down... meanwhile strangling and killing the tree that was their host” (Kim et al., 2016, Chapter 13). This metaphor perfectly captures the incremental, non-disruptive nature of this approach.

The case study of Blackboard Learn (2011) really highlights the effectiveness of this pattern. Blackboard faced severe challenges with a legacy J2EE codebase dating back to 1997, including complex builds and extremely long integration cycles. David Ashman, their chief architect, noted that "our build, integration, and testing processes kept getting more and more complex and error-prone. And the larger the product got, the longer our lead times and the worse the outcomes for our customers" (Kim et al., 2016, Chapter 13). To address these issues, Blackboard implemented Building Blocks—modular components accessed through fixed APIs. This allowed developers to work autonomously and significantly improved both speed and safety.

I believe this move was a game changer because it empowered teams to innovate without constantly coordinating with others, reducing bottlenecks and boosting morale. Ashman pointed out that “every developer given a choice would work in the Building Block codebase, where they could work with more autonomy and freedom and safety” (Kim et al., 2016, Chapter 13). The results were clear: more code commits, improved feedback loops, and better code quality overall.

From my own experience within my own company, I see a similarity. We recently had to overhaul our internal inventory tracking system, which had become so outdated that even small updates required multiple approvals and often led to system downtime. Inventory in the past had become an actual nightmare. By introducing a more modular approach where certain features, like promotional price adjustments, were handled in separate services, we dramatically reduced the need for all-hands meetings and improved rollout speed. I feel this mirrors the benefits Blackboard saw using the strangler pattern . In a way this also allows us to adapt quickly while keeping our systems stable.

In conclusion, the I felt the main lessons learned from this chapter and case study are clear: architecture shapes our ability to innovate and respond to change, and patterns like the strangler approach can help us evolve safely. As the authors emphasize, “We must be able to safely migrate from one architecture to another” (Kim et al., 2016, Chapter 13), and doing so can ultimately lead to a more agile, resilient, and productive organization.

**References**

Kim, G., Humble, J., Debois, P., & Willis, J. (2016). The DevOps handbook: How to create world-class agility, reliability, and security in technology organizations (Chapter 13). IT Revolution.