Jarred Santos 2/9/2025

The case study presented in chapter 13 revolves around the idea of adapting a legacy monolith code base into smaller more manageable “building block” micro services. Blackboard was presented with an ever growing problem, It was becoming to slow to respond to feed back and the overall amount of code commits was shrinking. Before implementing “building blocks” they were using a legacy J2EE codebase. The amount of code was growing yet the company was seeing less commits.

From here Blackboard began to use the strangler fig pattern to help re architecture the codebase. Before teams would have to communicate and were dependent on each other when trying to add new features, but this shift would enable internal teams to work independent of one another. The code base overall began to shrink, code was being moved into these new modules.

Since these modules are more independent than the old monolithic codebase, teams were able to be faster, more quickly adapt to feedback and were able to produce all around better code.

It can be impractical to rework a legacy piece of software from the ground up and that is were utilizing the strangler fig pattern can prove beneficial. Code could be moved separate individually functional modules that can be tested and worked on independently. The reason this can be so effective is that it provides no disruption to the older code as there is a period where new and old coexist before the new modules supplant the old monolith. Shrinking the size of your code base creates a more agile development team.