Lindsey Yin

CSD 380 Module 6

Assignment 6.2

24 June 2024

**Summary and Lessons Learned: Strangler Fig Pattern at Blackboard Learn (2011)**

In 2011, Blackboard Inc., a leading provider of educational technology, faced significant challenges with their legacy J2EE codebase for their flagship product, Blackboard Learn. The complexity and lead times associated with their old system were hindering productivity and impacting customer satisfaction. Chief architect David Ashman noted that the build, integration, and testing processes were increasingly error-prone and time-consuming, requiring 24 to 36 hours to get feedback from their integration process.

An analysis of their source code repository revealed a troubling trend: while the number of lines of code in the monolithic repository was increasing, the number of code commits was decreasing, indicating difficulty in introducing code changes. To address these issues, Ashman implemented the Strangler Fig Pattern in 2012. This involved creating Building Blocks—modular components decoupled from the monolithic codebase and accessed through fixed APIs. This re-architecting project allowed developers to work more autonomously and with greater efficiency.

The introduction of Building Blocks led to a significant decrease in the size of the monolith repository and an exponential increase in the number of lines of code and commits in the Building Blocks repositories. Developers preferred working in the new modular architecture due to the increased autonomy, freedom, and safety it offered. Mistakes were localized, reducing the risk of major system-wide failures.

Ashman concluded that the modular architecture and improvements in the build process provided faster and better feedback on development work, leading to improved code quality and developer productivity.

Lessons Learned:

1. Modular Architecture Enhances Productivity:
   1. Transitioning from a monolithic architecture to a modular one, where components are decoupled and accessed through fixed APIs, can significantly enhance developer productivity and autonomy. This shift allows developers to work independently on different modules without extensive coordination with other teams.
2. Early Issue Detection and Faster Feedback:
   1. Implementing a modular architecture and improving the build process can lead to faster and more reliable feedback on development work. This enables early detection of issues and contributes to higher code quality and more efficient development cycles.
3. Reduced Complexity and Improved Maintenance:
   1. The Strangler Fig Pattern helps manage and gradually replace legacy systems without the need for a complete shutdown. By incrementally moving functionality from the old system to the new modular components, organizations can reduce complexity and improve system maintainability over time.
4. Risk Mitigation through Localized Failures:
   1. Modular architectures localize the impact of errors, reducing the risk of catastrophic failures that affect the entire system. This approach ensures that mistakes in one module do not propagate throughout the system, enhancing overall system stability and reliability.
5. Autonomy and Developer Satisfaction:
   1. Allowing developers to work in a modular codebase fosters a sense of autonomy and freedom, leading to higher job satisfaction and motivation. Developers can make changes and improvements more confidently, knowing that their work is less likely to disrupt the entire system.

In summary, Blackboard Learn's implementation of the Strangler Fig Pattern and the transition to a modular architecture provided significant benefits in terms of developer productivity, code quality, and system maintainability. These improvements demonstrate the value of modular architectures and the importance of strategic re-architecting projects in managing legacy systems.

References

Kim, G., Humble, J., Debois, P., & Willis, J. (2021). *The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations* (2nd ed., pp. 153-156). IT Revolution.