Tej Chudali   
Module-6.2

**Summary of Case Study: Strangler Pattern at Blackboard Learn (2011)**

In 2011, Blackboard Inc., a leading provider of educational technology, faced challenges with their aging Learn product, which had a complex and error-prone J2EE codebase dating back to 1997. Chief architect David Ashman noted the growing difficulties in their development process, with build, integration, and testing taking up to 36 hours, which negatively impacted developer productivity. The increasing complexity led to fewer code commits despite a rising codebase size, signaling the need for change.

In 2012, Ashman initiated a project using the strangler pattern to re-architect the code. They created Building Blocks, which were decoupled modules accessed via fixed APIs. This allowed developers to work more independently and efficiently. As developers shifted to Building Blocks, the size of the monolithic codebase decreased, and productivity improved. The new architecture resulted in smaller, localized failures instead of system-wide issues, enhancing safety and feedback speed.

**Lessons Learned**

1. **Complex Legacy Systems**: Old, monolithic systems can become increasingly difficult to manage and slow down development.
2. **Strangler Pattern**: Incremental re-architecting using the strangler pattern can help transition from a monolithic to a modular system, improving developer productivity and code quality.
3. **Modularity and Autonomy**: Allowing developers to work in decoupled, modular components increases their independence, reduces coordination overhead, and enhances overall productivity.
4. **Faster Feedback**: Improvements in build processes and modularity lead to quicker feedback, which in turn improves code quality and development speed.
5. **Architectural Impact**: The architecture of a system significantly influences how efficiently code can be tested, deployed, and maintained.