5 ARCHITECTURAL PROBLEMS:

NAME: M. AALYAN MUGHAL

REG #: FA22-BSE-094

SECTION: 5B

DATE: 31 DECEMBER, 2024

Architectural Problems in Software and Their Solutions

Problem 1: Scalability Issues in Twitter (2010)

Problem: Twitter faced scalability challenges due to a monolithic architecture that could not handle the increasing number of users and tweets efficiently. The system frequently crashed during high traffic events, such as major news breaks.

Solution: Twitter transitioned to a microservices architecture, decoupling different components like the timeline service and user service. This change allowed them to scale specific services independently. The release number was updated to reflect these architectural changes, while the build number remained consistent.

Problem 2: Load Management in Amazon (2000s)

Problem: Amazon's early monolithic system struggled with load management during peak shopping seasons, causing slowdowns and occasional downtimes.

Solution: Amazon re-architected their system into microservices, with each service owning its data and functionality. They introduced distributed databases and a service-oriented architecture (SOA), significantly improving performance and reliability. The release number changed to indicate the system overhaul.

Problem 3: Fault Tolerance in Netflix (2012)

Problem: Netflix's traditional data centre-based infrastructure was prone to outages, leading to service interruptions for users.

Solution: Netflix migrated to a cloud-based architecture on AWS and implemented a microservices architecture. They developed tools like Chaos Monkey to test the fault tolerance of their system proactively. These changes resulted in a new release version.

Problem 4: Codebase Maintenance in Microsoft Office (1990s)

Problem: Microsoft Office’s tightly coupled codebase made it challenging to add new features without introducing bugs or affecting existing functionalities.

Solution: Microsoft adopted a component-based architecture, modularizing the software into independent components that could be developed and tested separately. This architectural change improved maintainability and was reflected in a new release number.

Problem 5: Data Handling in Instagram (2015)

Problem: Instagram faced challenges with real-time data synchronization as their user base and interaction rates grew exponentially.

Solution: Instagram adopted a distributed architecture with a focus on asynchronous task queues and optimized their backend database systems. They integrated Apache Kafka for real-time data streaming. These changes led to a release number update.