Project Overview

Software Upgrades in Large Scale Distributed Systems

- Upgrades must propagate automatically
- Operator requires control of scheduling
- Upgrades cannot occur instantaneously
- System must continue to provide service

Practical Byzantine Fault Tolerance

- Replication algorithms that tolerate SW errors and malicious attacks
- Efficient and designed for asynchronous environments
MIT2002-02: Software Upgrades in Distributed Systems

Barbara Liskov

Progress Through December 2002

Software Upgrades:

• Definition of a new architecture for upgrades
• Scheduling functions provide operator control
• Simulation objects allow continuous operation
• Transform functions preserve state across upgrades

Byzantine Fault Tolerance:

• Byzantine fault tolerant large scale services
• Use peer-to-peer design principles to scale service
• Hybrid architecture: servers act as peer-to-peer nodes and a separate configuration service determines current membership
Research Plan for the Next Six Months

Software Upgrades:
- Begin implementation of upgrade infrastructure
- Develop correctness conditions for simulation objects and transform functions

Byzantine Fault Tolerance:
- Finish large scale service implementation
- Deploy it on a wide area testbed (PlanetLab)