# Towards a Software Architecture for Graceful Degradation



**Charles P. Shelton** Philip Koopman

#### Graceful Degradation:

Component failures should reduce functionality rather than cause system failure

## **Desirable Architectural Objectives:**

- Build a fine-grained distributed system to decouple components
- Partition the system into critical and non-critical components
- Construct well-defined component interface definitions

28.90

28.90

20.00

70.50 145.00

**Carnegie Mellon** 

269.71

399.23 173.47

332.05 723.45

6.98 N/A

33.35 N/A

100.18

289.69

N/A 53.60

1345.70 590.00

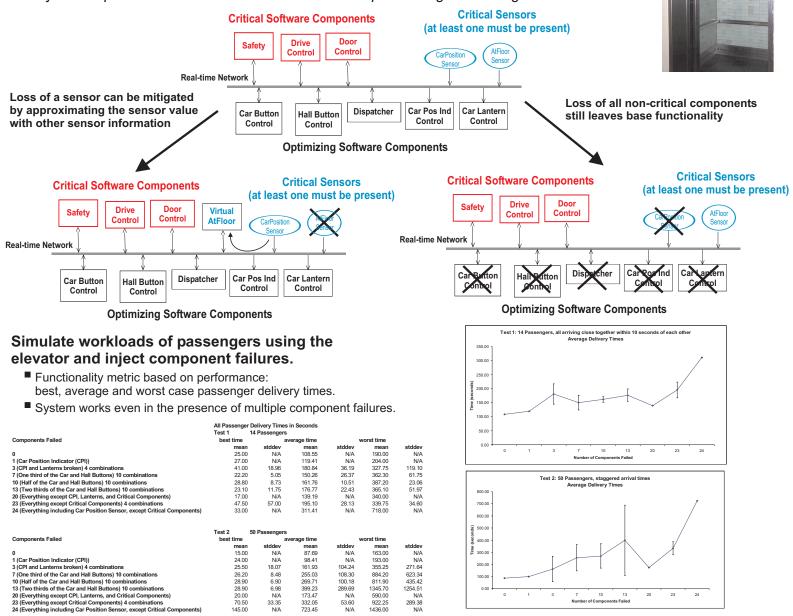
922.25

Design components to be semi-autonomous (provide some functionality when inputs are lost)

### **Proof of Concept:**

#### Model an elevator control system with these architectural objectives

- Simulate the physical elevator system and build the software control system
- Inject component failures to determine how well it performs graceful degradation







435.42 1254.51

N/A 289.38



