RoSES
Robust Self-configuring Embedded Systems

Scalable Graceful Degradation in Distributed Embedded Systems

Charles P. Shelton
Philip Koopman

Specifying Graceful Degradation is Exponentially Complex:

Must rank 2^N system configurations of N software components, sensors, and actuators

Our Model Achieves Scalable Specification for Data Flow-Centric Embedded Systems:

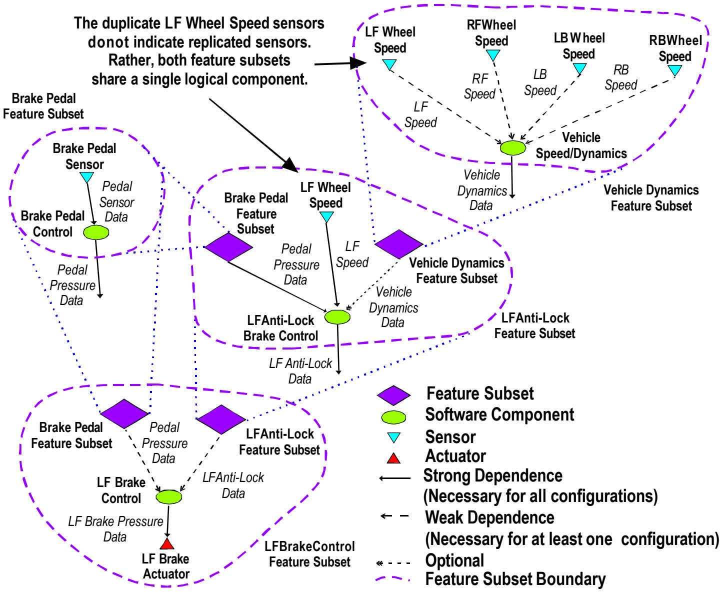
Analysis reduced from $O(2^N)$ to $O(2^k)$; k = number of components per subsystem

1. **Separate system into orthogonal software and hardware views; focus on software configs**
2. **Partition system into *feature subsets* based on component input and output interfaces**
3. **Allow feature subsets to overlap and share components**
4. **Rank relative utility of configurations of subsystems, not all configurations of entire system**
5. **Determine system configuration utility as a composition of working subsystem utilities**

Proof of Concept:

Scalable Specification of an example Brake-By-Wire system

- Real-world electromechanical anti-lock braking system transformed to hypothetical embedded network control system
- 10 Software Components, 5 Sensors, 4 Actuators = 19 System Components
- ***2^{19} = 524,288 possible system configurations; 89,600 valid with positive utility***
- In our model: 10 defined subsystems, max 5 components per subsystem
- ***Specify utility of only 52 subsystem configurations for complete utility function for all system configurations***


The duplicate LF Wheel Speed sensors donot indicate replicated sensors. Rather, both feature subsets share a single logical component.

◆ Feature Subset
● Software Component
▽ Sensor
▲ Actuator
⟵ Strong Dependence (Necessary for all configurations)
← – Weak Dependence (Necessary for at least one configuration)
« - - - Optional
– – Feature Subset Boundary

Application to Real Distributed Embedded Systems:

Elevator Control System:

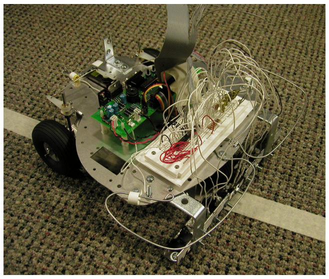
State-centric System

- *Simulate the physical elevator system and build the software control system*
- *Inject component failures to determine how well it performs graceful degradation*
- *Results: system can tolerate loss of up to 75% of system components*

Autonomous Robot Navigation:

Control Loop-centric System

- *Build a robot to traverse a race course*
- *Combine line following sensor system with location tracking and navigation algorithms*
- *Run robot with combination of failed sensors and software to observe graceful degradation*
- *Progress: hardware built, navigation software in development*



GM
Carnegie Mellon
Electrical & Computer ENGINEERING
Institute for Complex Engineered Systems