

Class 3 Foundations and Perspectives

David Garlan
Carnegie Mellon University

www.cs.cmu.edu/~garlan/17811/

Today's Class

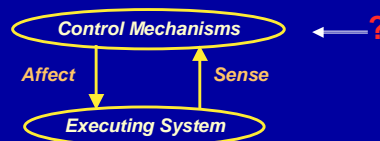
- Discuss readings
 - > what are they arguing?
 - > what are their biases?
 - > how are they related to each other?
 - A framework for understanding SHS engineering issues
 - > key ingredients of a SHS
 - > dimensions of variability
 - Topic taxonomy
 - > what should we study in this course?
-

Towards a Framework: Four Unifying Ideas

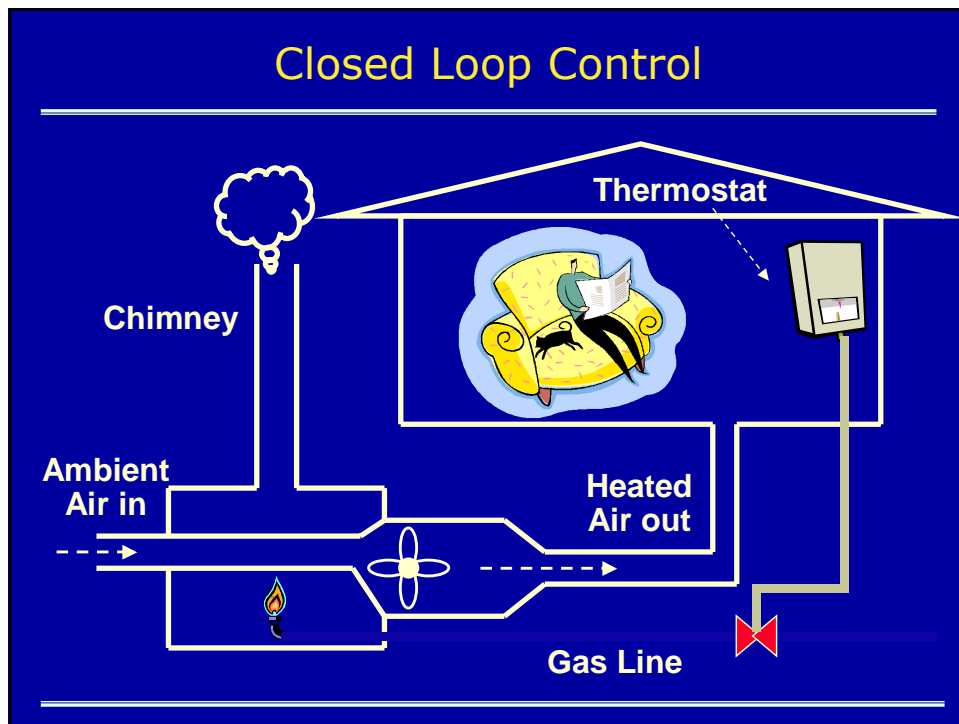
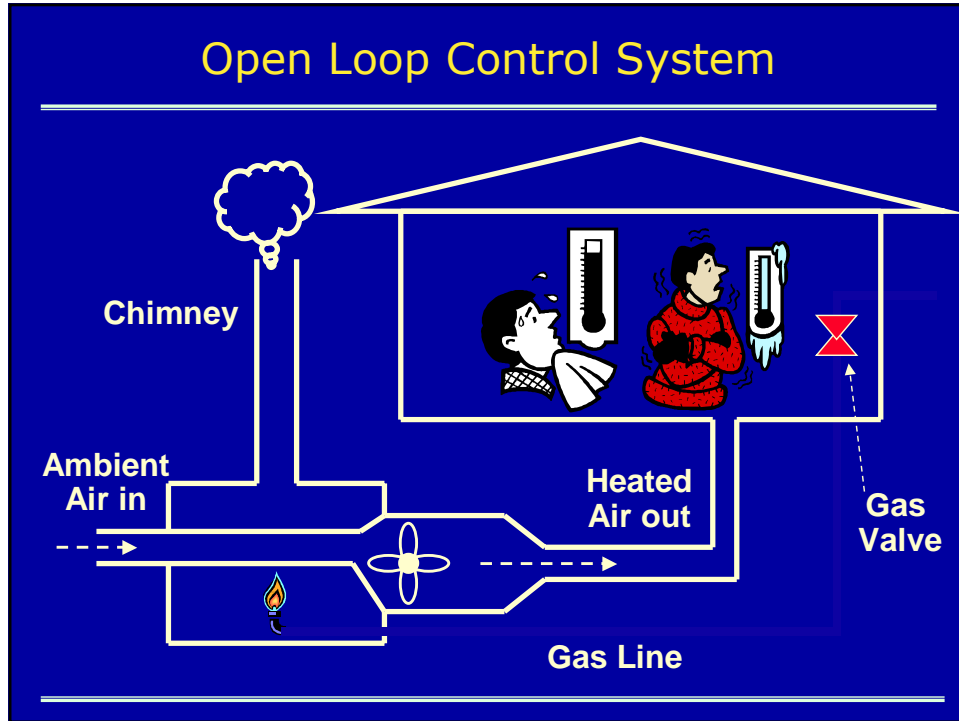
- Closed loop control systems
 - Extension of classical dependability
 - Translucency
 - A new view of system layers
-

Idea 1: Closed Loop Control

- Move from open-loop to closed-loop systems



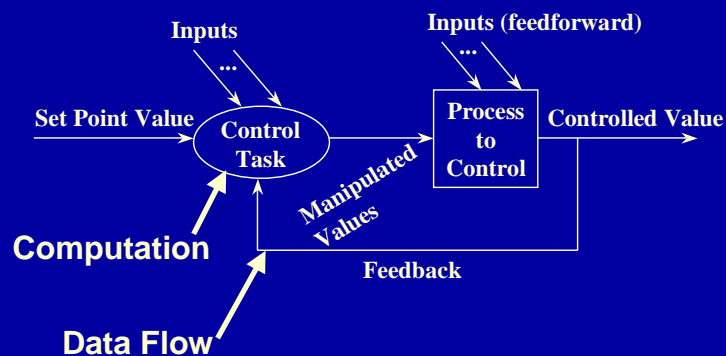
- Systems monitor their own behavior, and have externalized control mechanisms to detect problems and effect repairs.
-



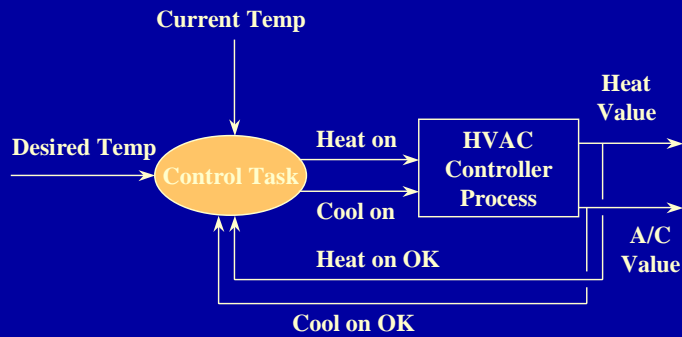
Process Control Vocabulary

- Open Loop System: Information about process variables not used to control the system
- Closed Loop System: Information about process variables used to control the system
 - > Process Variable: things that can be measured
 - > Controlled Variable: air temperature
 - > Input Variable: room temperature
 - > Manipulated Variables: resultant (changed) values
 - > Set Point: value for the controlled variable
 - > Feedback Control: value of controlled variable changes the process
 - > Feed-forward Control: value of other process variables change the process

Closed Loop Process Control Architecture

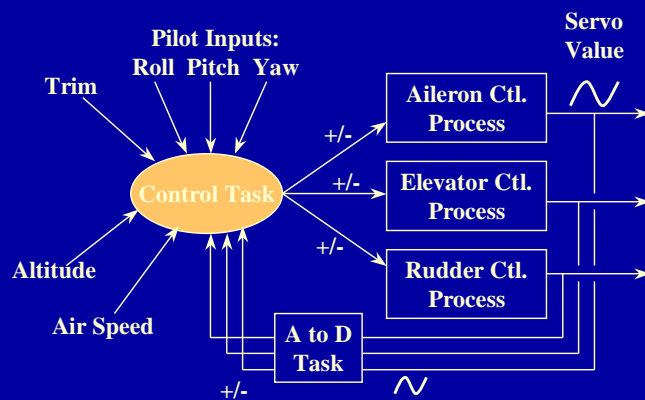


Example - Thermostatic Control Architecture



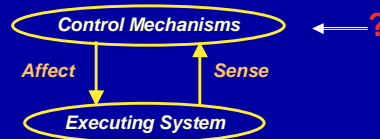
Example - Simplified (overly) Avionics

- +/- Digital positive or negative control surface deflection
- ~ Analog signal



For Software?

- Move from open-loop to closed-loop systems



- Systems monitor their own behavior, and have externalized control mechanisms to detect problems and effect repairs.

Advantages of Approach

- Externalized control mechanisms support separation of concerns
- Reusable infrastructure for monitoring and reflection reduces cost of development
- Repair policies can be formally verified in terms of abstract models at the control layer
- Permits multiple views, each view encapsulating repair for some dimension of quality
 - > E.g., performance, reliability, security

Four Technical Requirements

- **Monitor:** Observe the running system and abstract observed behavior.
 - **Detection:** Continuously check design constraints via explicit run-time models.
 - **Resolution:** Determine the cause of constraint violation and choose a repair strategy.
 - **Adaptation:** Adapt the system using verified change strategies.
-

Research Issues

- **Monitor:**
 - » How can we get information out of a running system?
 - » How can we abstract it to make sense of observations?
 - **Detection:**
 - » What kind of models are useful? What kinds of constraints?
 - **Resolution:**
 - » What kind of repair engines are useful, efficient, flexible?
 - **Adaptation:**
 - » How can we verify repair strategies?
-

Idea 2: Dependability

- See Phil Koopman slides

Definitions

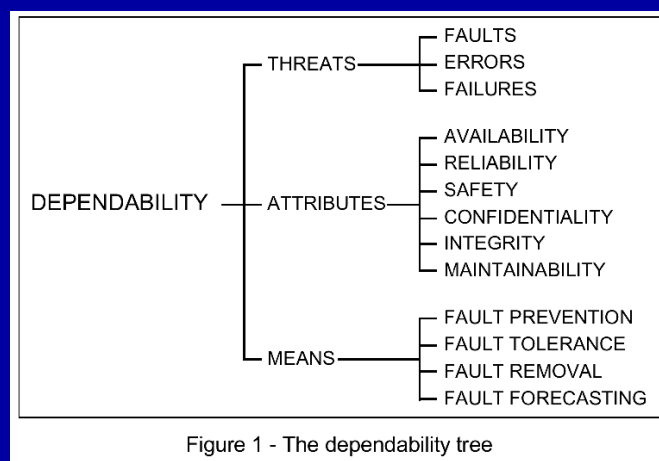


Figure 1 - The dependability tree

[Avizienis/Laprie/Randell 01]

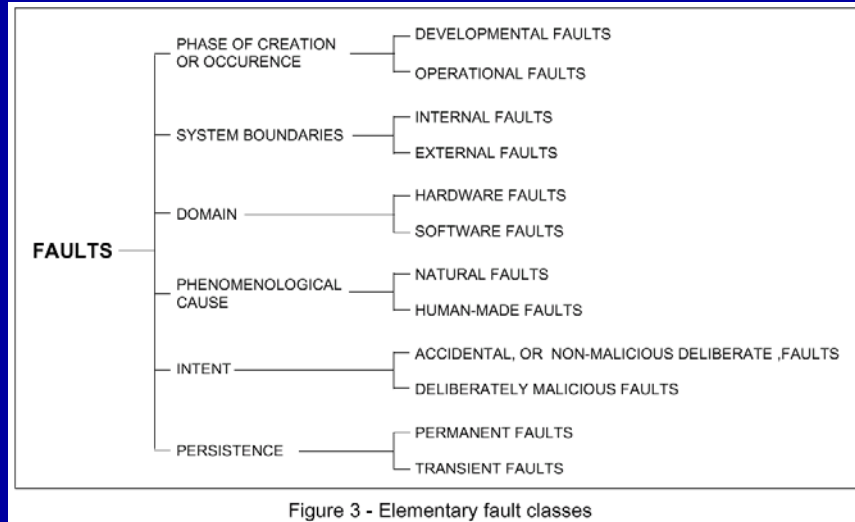
Basic Steps in Fault Handling

- Fault Confinement -- **contain it before it can spread**
 - Fault Detection -- **find out about it to prevent acting on bad data**
 - Fault Masking -- **mask effects**
 - Retry -- **since most problems are transient, just try again**
 - Diagnosis -- **figure out what went wrong as prelude to correction**
 - Reconfiguration -- **work around a defective component**
 - Recovery -- **resume operation after reconfiguration in degraded mode**
 - Restart -- **re-initialize (warm restart; cold restart)**
 - Repair -- **repair defective component**
 - Reintegration -- **after repair, go from degraded to full operation**
-

Always State Your Fault Model

- > Failure = system does not deliver service
 - > Error = system state incorrect (may or may not cause failure)
 - > Fault = defect; incorrect data; etc. (*potential cause of error if activated*)
-

Class 3 Foundations and Perspectives

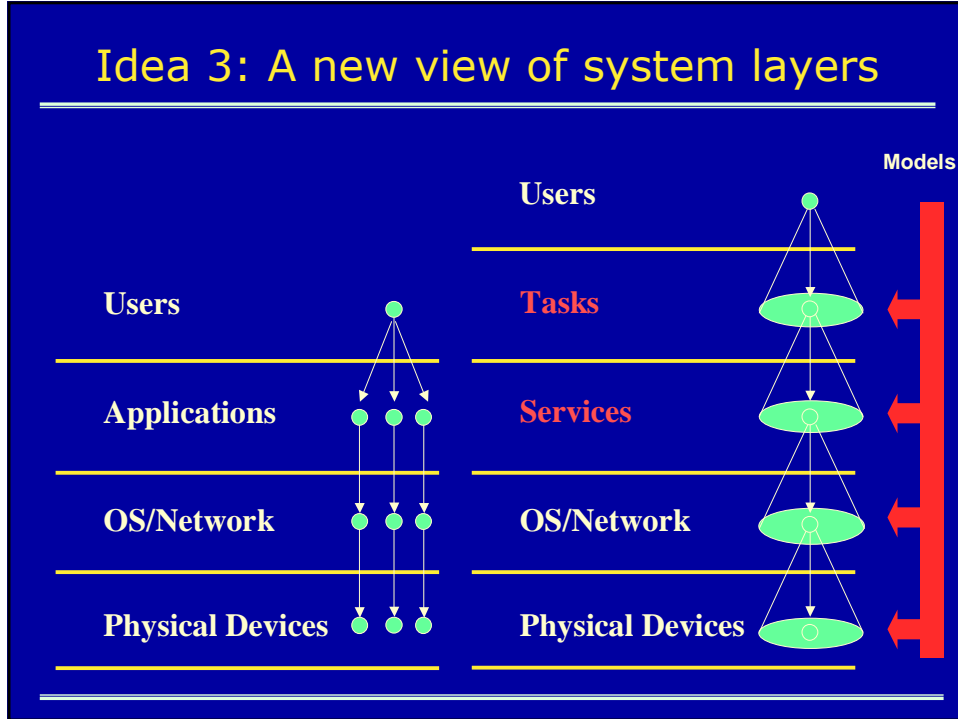


[Avizienis/Laprie/Randell 01]

Idea 3: Translucency

- See slides from Chroma meetings.

Idea 3: A new view of system layers



Framework for Comparison?

Class 3
Foundations and Perspectives

Topics Revisited

- User Interfaces
 - Model-based Approaches –
 - » Owen (arch, esp), Bhuricha (arch, reflection)
 - Mobility, Ubiquitous Computing, OS Support, Resource-awareness
 - » Joao, Vahe
 - Alternative Models of Computation
 - » Kevin (bio & evolutionary, esp)
 - Agent-based
 - » Justin (games)
 - Algorithms and Code
 - » Vahe (esp self-stabilizing)
 - Networks, Distributed Systems, and Middleware
 - » Sukanya? (esp. fault-tolerance, etc.)
 - Fault Tolerance, Dependability, Reliability, etc.
 - » Paul?
 - Formal Models
 - » Jung Soo + Model-based
-