EMBEDDED SYSTEM DESIGN

LECTURE VI
TDDI11 Embedded Software

DEPT. COMPUTER AND INFORMATION SCIENCE (IDA)

LINKÖPINGS UNIVERSITET

Design challenge: optimizing design metrics

Common metrics:

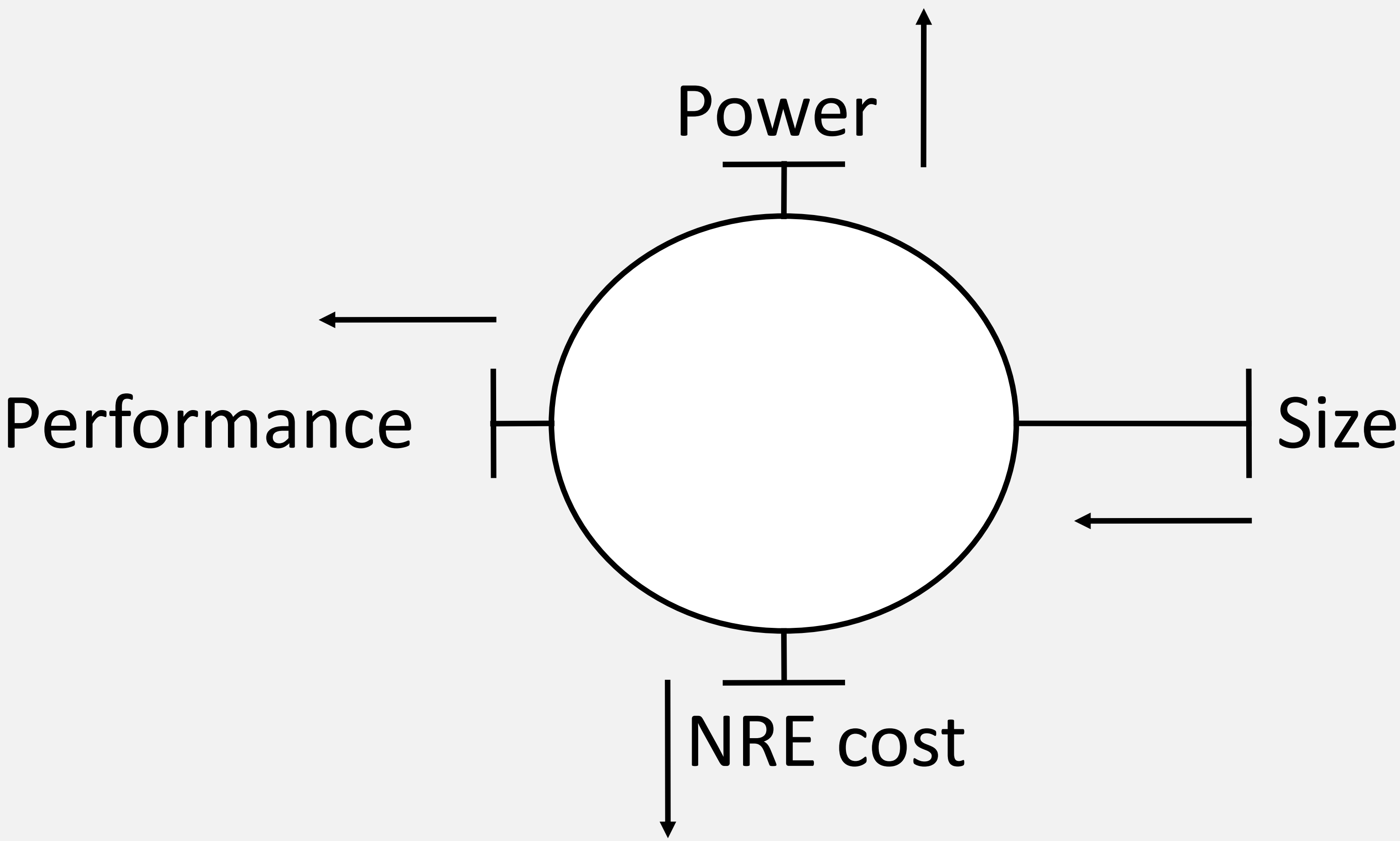
- **NRE cost** (Non-Recurring Engineering cost): The one-time monetary cost of designing the system
- **Unit cost**: the monetary cost of manufacturing each copy of the system, excluding NRE cost
- **Time-to-prototype**: the time needed to build a working version of the system
- **Time-to-market**: the time required to develop a system to the point that it can be released and sold to customers

Design challenge: optimizing design metrics

Common metrics (continued)

- **Size**: the physical space required by the system
- **Performance**: execution time or throughput of the system
- **Power**: amount of power consumed by the system
- **Flexibility**: ability to change the functionality of the system without incurring heavy NRE cost
- **Maintainability**: the ability to modify the system after its initial release
- **Safety:** absence of catastrophic consequences on the user(s) and the environment.

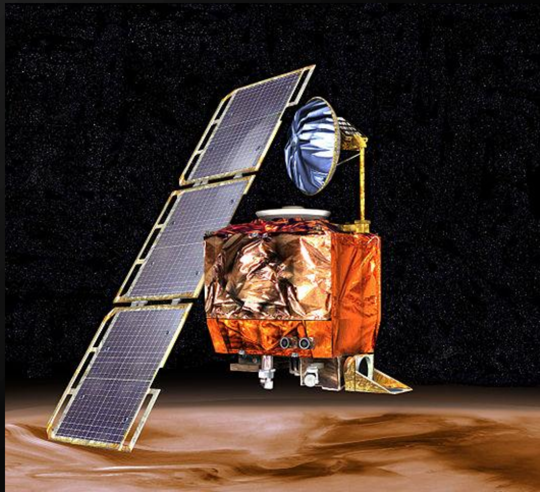
Design metric tradeoffs: improving one may worsen others



Design methodology is important

The loss of the Mars climate observer (1999)

- Most likely approached Mars too closely
- One of the problems: requirement
  - Jet Propulsion Lab expected values in Newton
  - Contractor calculated in pound force
  - Trajectory adjustments unsuccessful
  - Was not caught by configuration process or manual inspection



Design methodology, design flow

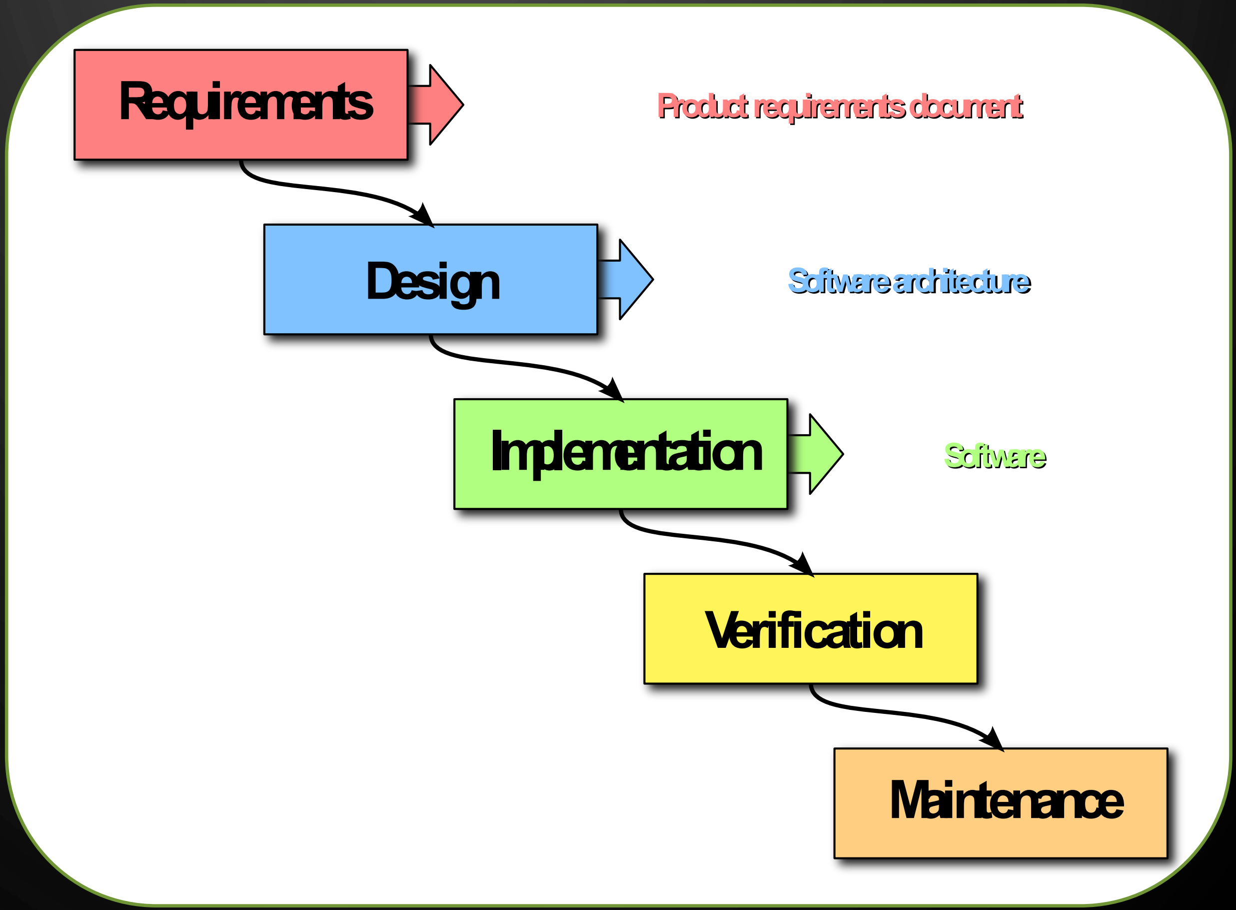
- **Design methodology**: the process of creating a system
  - Goal: optimize competing design metrics
    - Time-to-market
    - Design cost
    - Manufacturing cost
    - Quality, etc.

- **Design flow:** sequence of steps in a design methodology.
  - May be partially or fully automated with compilers and CAD tools.
  - Use tools to transform, verify design.

- Design flow is one component of design methodology. Methodology also includes management, organization, etc.

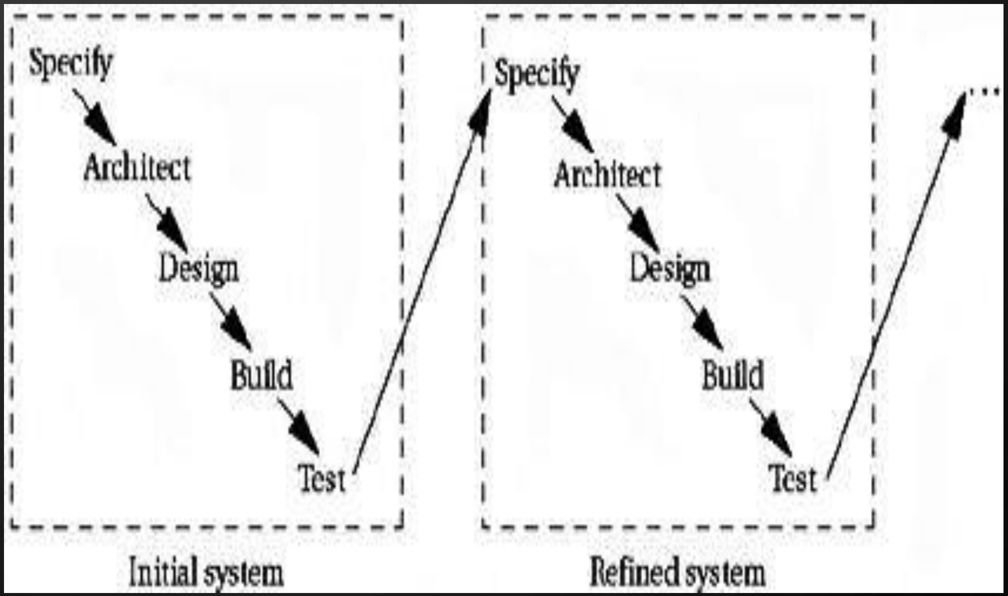
Waterfall model

Early model for software development.



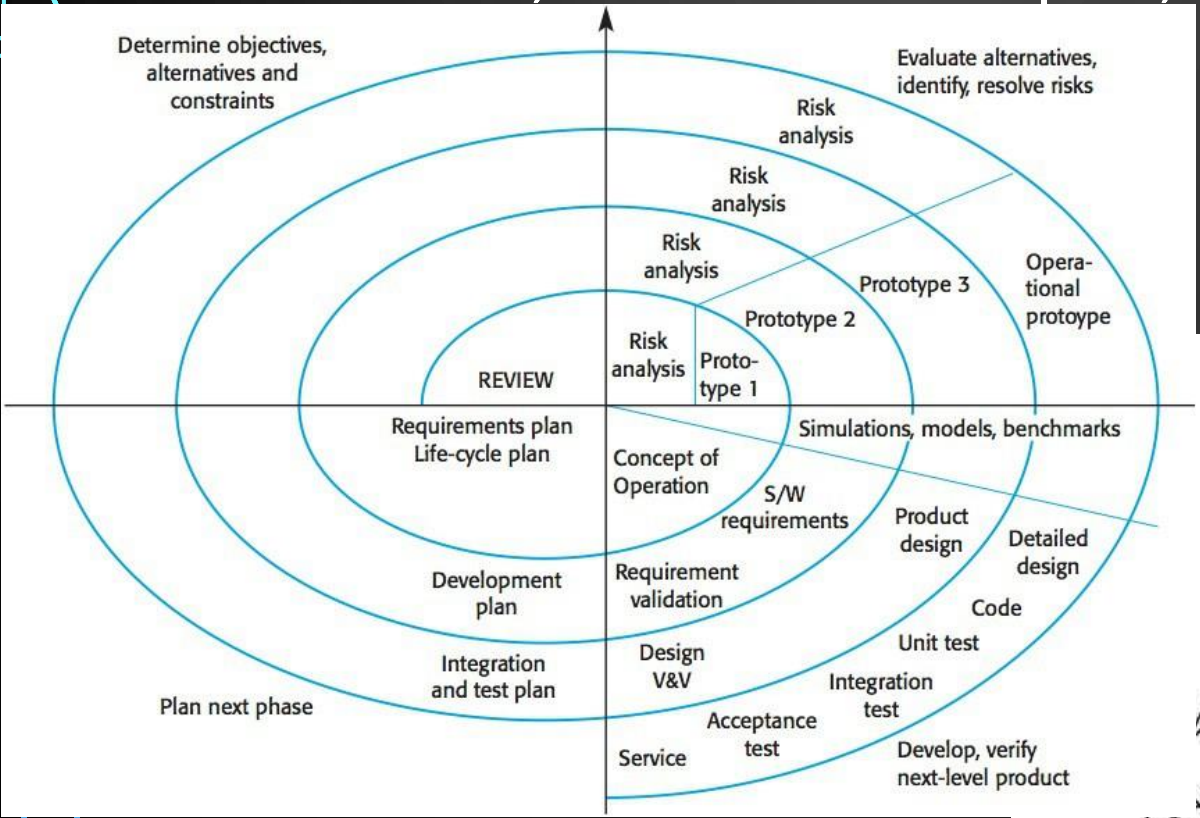
Successive refinement:

- Several iterations, suited for when unfamiliar with domain application

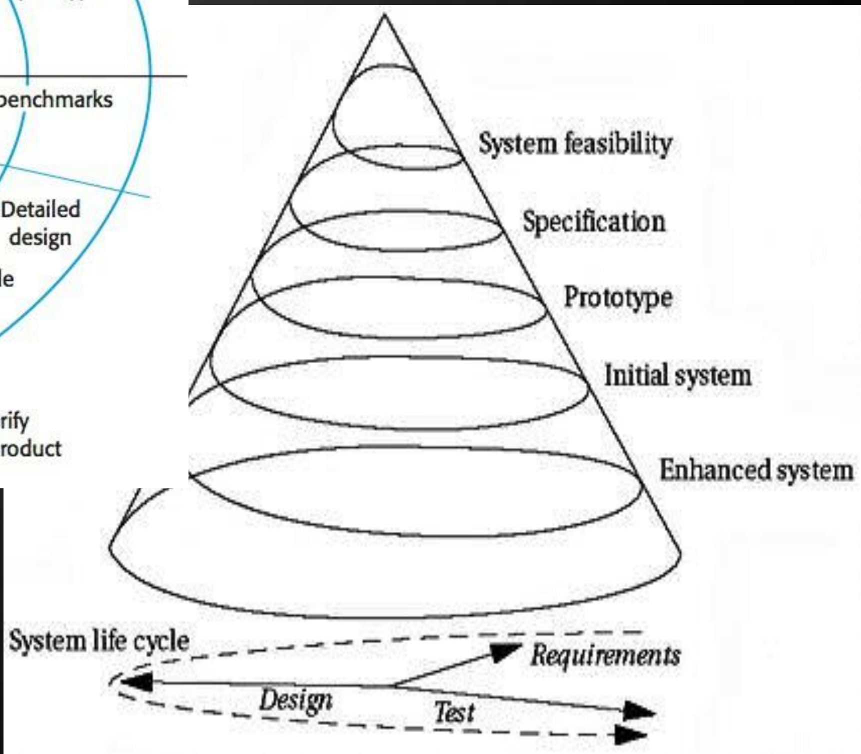


Spiral model

- More realistic, but can be complex, time to market?



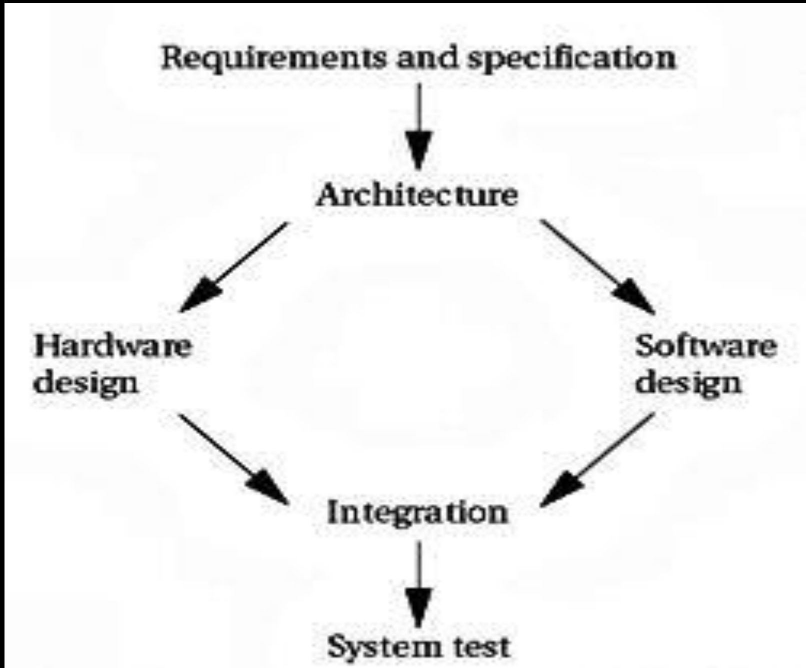
From http://iansommerville.com



Design flows for embedded systems

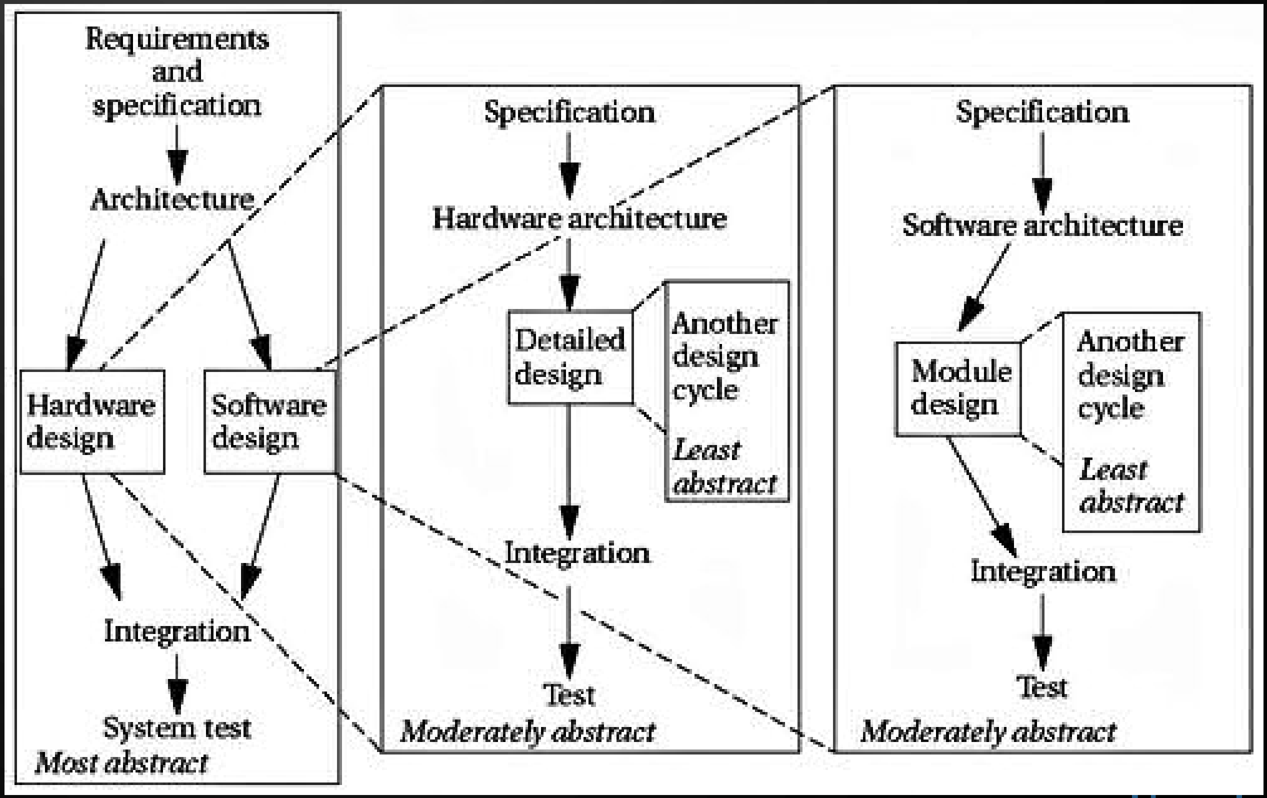
Embedded systems need design of hardware and software

Even if you don't design hardware, you still need select the correct boards, plug together several hardware components, and write code

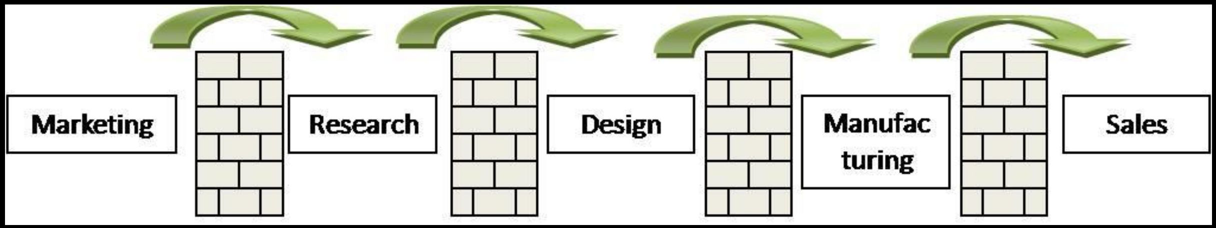


Hierarchical design

- from a most abstract complete system design flow
- to more detailed design flow of components
- many involved teams: requirements, design, testing
- Communication is important!

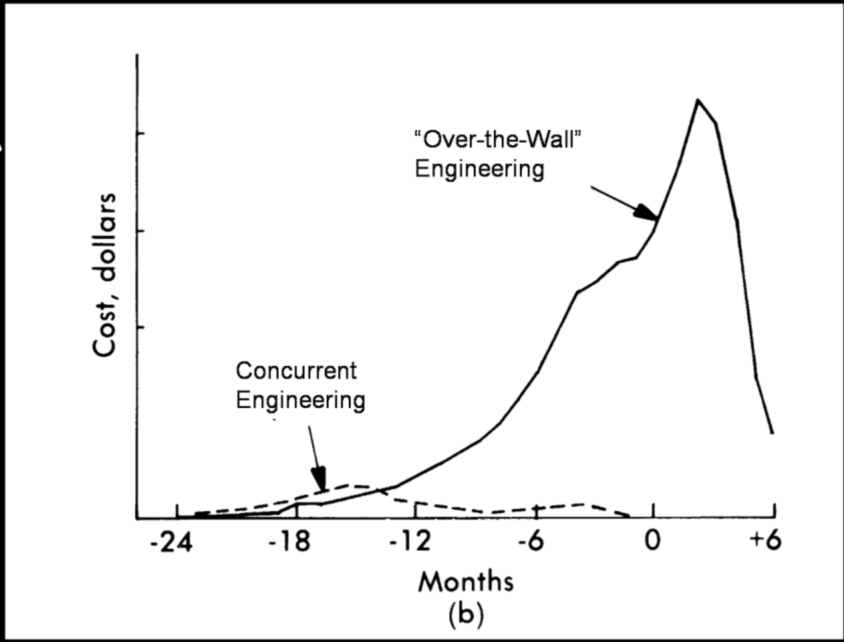


”Concurrent” vs. ”Over-the-wall” Engineering



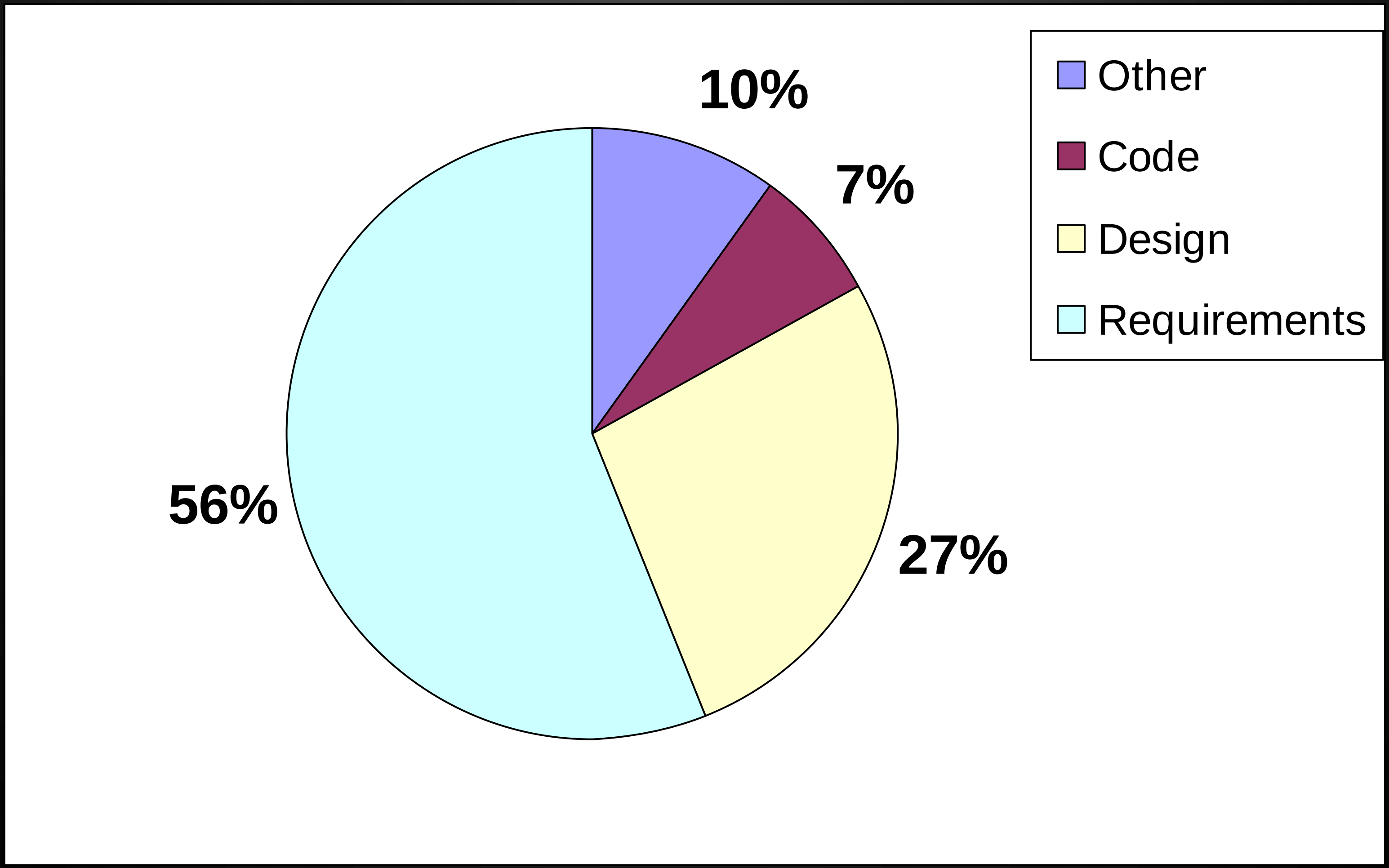
http://npdbook.com

- Large project, several teams
- Good communication between teams: e.g., components and design
- Eliminate “over-the-wall” approa
  - Cross-functional teams
  - Concurrent product realization
  - Information sharing
  - Integrated project management



http://npdbook.com

Frequency of faults



[Jim Cooling 2003]

Requirement and Specification

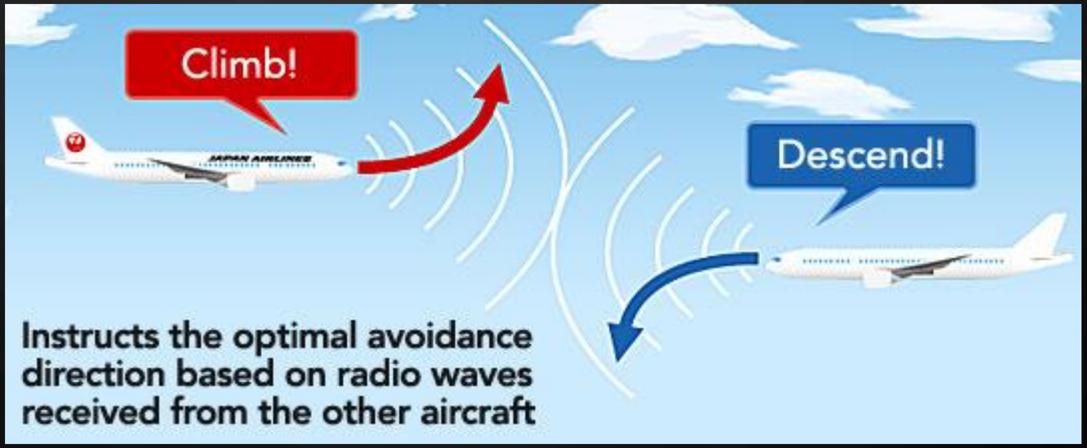
- Requirement: informal descriptions of what customer wants:
  - Correctness, unambiguousness, completeness, verifiability, consistency, modifiability, traceability
- Specification: more detailed, precise descriptions
- Both Requirements and specification describe system behavior from outside

Requirement and Specification

- Describing embedded systems' behavior
  - Can be extremely difficult
    - Complexity increasing with increasing IC capacity
    - Desired behavior often not fully understood in beginning
- English (or other natural language) common starting point
  - Precise description difficult to impossible

The 2002-Überlingen crash

Several factors, including ambiguities about relation between TCAS (Trafic Collision Avoidance System) and ATC (Air Trafic Control)

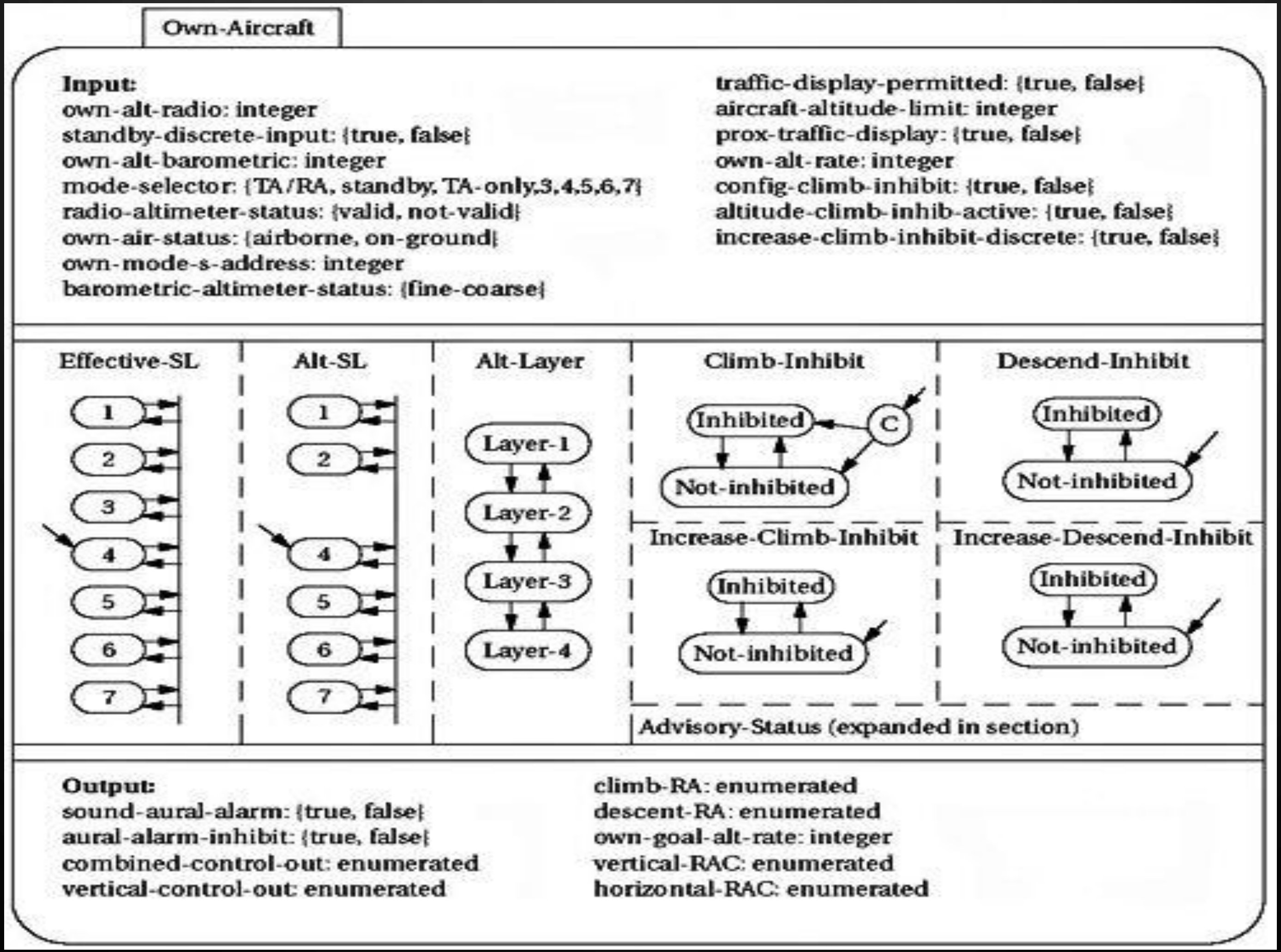


www.jal.com

Models and languages

- How can we (precisely) capture behavior?
  - We may think of languages (C, C++), but *computation model* is the key
- Common computation models:
  - Sequential program model
    - Statements, rules for composing statements, semantics for executing them
  - State machine model
    - For control dominated systems, monitors control inputs, sets control outputs
  - Dataflow model
    - For data dominated systems, transforms input data streams into output streams

Models and languages



From “experience from specifying TCASII requirements using RSML”, Heimfahl and al, 1998.

Quality assurance

- Quality should be built in
- Quality standards (e.g., iso 9000)
- Verify specification and review designs
- Long lived bugs are more expensive
- The Therac-25 medical imaging system
  - Six known accidents, massive radiation overdose
  - Machine software developed in assembly by single programmer over several years
  - Some errors depended on typing speed of operators
  - Limited safety analysis, no mechanical backups, overly complex programs

Why platforms?

- Develop and share key components whether proprietary or open source
- Use them in as many products as possible to provide new functionalities
- Eliminate redundancies, speed development, saves costs

Standards and platforms drive many markets

- Standards drive many markets

ISO 26262

DO-178B/C

DIRECT Wi Fi CERTIFIED

ZigBee

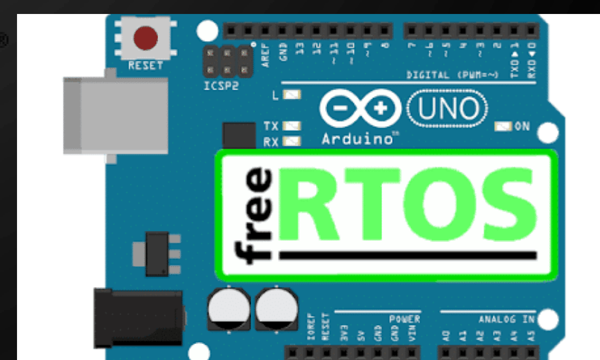
Z WAVE

IEC 62304

MISRA C:2012

QNX

QNX SOFTWARE SYSTEMS



ASIL D, C, B, A