# Safety-Critical Real-Time Systems

## ARTES PhD course

### Lecture 2: Safety-related Processes

**Simin Nadjm-Tehrani ©**

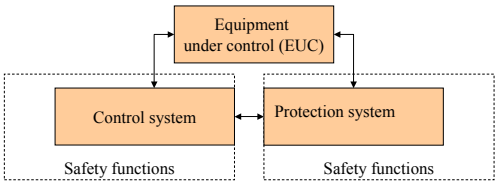
Real-time Systems Laboratory

simin@ida.liu.se



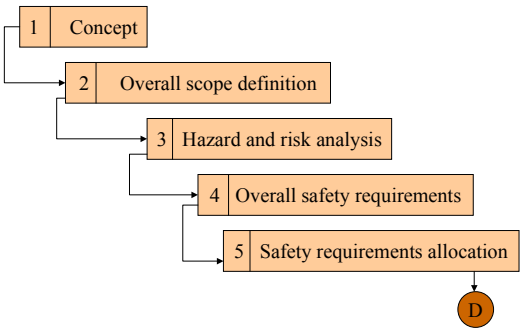
## Structure of SC systems

**IEC 61508**

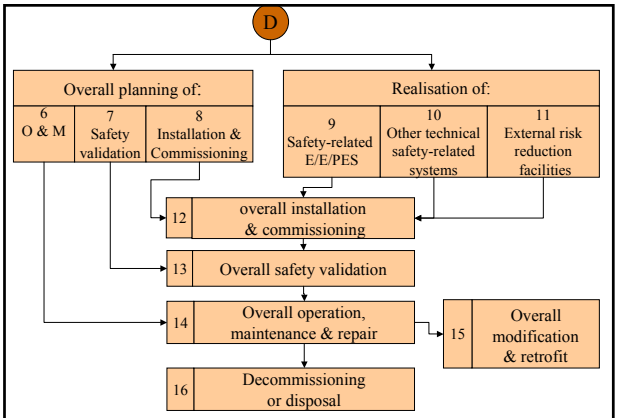





## Overall safety lifecycle





But how does this relate to our classical (computer) systems development activities?



## What is safety analysis?

- Understanding the risks
- Associating assurance levels with subsystems
  - E.g. Safety Integrity Levels - SILs
  - can be seen as targets for risk reduction
- Implementing appropriate risk reduction schemes
- Constructing safety arguments



## Recall from earlier...

- Faults may lead to failures
- Failures may cause hazards
- Hazards may jeopardise safety

Thus:

- Removing/containing certain faults enhances safety



## More on dependability

Four approaches [IFIP 10.4]:

1. Fault avoidance
2. Fault tolerance
3. Fault removal
4. Fault forecasting

## Fault avoidance

- Proponents of **formal methods** present this as the complementary approach to current practices (that can impossibly eliminate faults).
- More on this in Lecture 4.



## Fault tolerance (FT)

- Is only possible by incorporating some form of **redundancy**
- This lecture: Basic introduction to FT
- Lecture 5: Application of FT analysis techniques.



## Fault Removal

- Consists of two phases:
  - Identification of the fault: Testing/Debugging
  - Repair or correction of the fault: Domain dependent

*In practice many safety engineers aim towards error removal.*



## Fault Forecasting

- Also sometimes referred to as ”fault evasion”
- To use the knowledge from
  - current behaviour during operation
  - current and potential future load
- Forecast potential failure scenarios and avoid them under operation (e.g. by load balancing, patching).

## Fault ⇒ Error ⇒ Failure

- Goal of system verification and validation is to remove faults
- Goal of hazard analysis, Fault Tree Analysis (FTA), and Failure Modes and Effects analysis (FMEA) is to focus on important faults, those which might lead to failures
- Goal of fault-tolerance is to reduce effects of errors if they appear - *eliminate or delay failures*



## Fault-tolerance

- Means that a system provides a minimal acceptable function
  - Even in presence of (a class of) faults
  - During a period defined by certain model assumptions
- Foreseen or unforeseen?



## External factors

Filmsnutten...



## Examples

- Year 2000 bug
- Bit flips in hardware due to cosmic radiation in space
- Loose wire
- Air craft retracting its landing gear while on ground

Effects in time:
Permanent/ transient/ intermittent

## Fault management

- On-line fault-detection
  - by program or its environment
- Fault-tolerance using redundancy
  - software
  - hardware
  - data
  - time



## Redundancy

From D. Lardner: Edinburgh Review, year 1824:

*"The most certain and effectual check upon errors which arise in the process of computation is to cause the same computations to be made by separate and independent computers*; and this check is rendered still more decisive if their computations are carried out by different methods."*

* *people who compute*



## Static Redundancy

Used in all cases (whether an error has appeared or not), just in case…

- SW: N-version programming
- HW: voting systems
- Data: parity bits, checksums



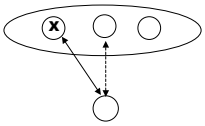
## Dynamic Redundancy

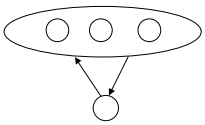
Used when error appears and has to be treated

- SW: recovery methods
- HW: switching to back-up module
- Data: error-correcting codes

## Basic replication models

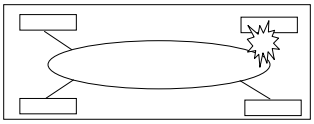
- Primary backup
- Active replication







## Brake-by-wire





## Sources of failure

From Computers & Risk (P. Neumann):

1. Approach: Using immature technology, or not using technology when it would avoid the failure
2. Requirements Specifications: Invalid assumptions, incomplete or inconsistent requirements



## Sources of failure

3. System Design: Fundamental shortcomings in the specification or design of hardware or a program

4. Support systems: Bad programming languages, erronuous compilers and debuggers, misguiding/confusing tools

## Sources of failure

5. HW/SW implementation: Faults in production process, faulty circuits, program bugs, harmful code (viruses)

6. Analysis of design: Incorrect assumptions about physical world, operating environment, human behaviour



## Sources of failure

7. Analysis of implementation: incomplete testing, errors in debugging or verification process

8. Evolution & decommissioning: insufficient maintenance, too early removal of a backup system, hidden dependencies on an old subsystem, ”last straw”



## Complementary Reading

- Hazard analysis
- Fault Tree Analysis
- Failure Modes and Effects analysis
- Safety Cases, Safety Arguments