Reliable and Concurrent Systems

Basic Concepts and Terminology

Mário
de Sousa

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Basic Concepts and Terminology

- Basic Definitions
- Faults, Errors and Failures
- Dependability Attributes
- Means of Attaining Dependability
- Computational Systems and Safety

Basic Definitions

- Dependability:
  - "[..] the trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers [..]" (IFIP WG10.4 definition)
  - Ability to avoid service failures that are more frequent or more severe than is acceptable.

A system can, and usually, does fail. When does it become undependable?
For each system/user, we need criteria for deciding
when the system becomes undependable.

Basic Definitions

- Dependability
  - Impairments
    - Faults
    - Errors
    - Failures
  - Methods
    - Obtention
      - Fault Tolerance
      - Fault Prevention
    - Validation
      - Fault Removal
      - Fault Forecasting
  - Attributes
    - Availability
    - Reliability
    - Safety
    - Maintainability
    - Security
      - Integrity
      - Confidentiality

Basic Definitions

- Dependability Impairments
  - The Failure of a system occurs when the service provided is no longer in accordance with the specification;
    - » This definition was later changed in order to include the behaviours which, although satisfying the specification, are unacceptble to the system's users (due to a fault in the specification).
  - Error is a state of the system that may lead to a failure.
  - The hypothetical cause of an error is a Fault.

Basic Definitions

- Methods of obtaining Dependability:
  - fault prevention (do not insert faults);
  - fault tolerance (tolerate existing faults);
  - fault removal (find and remove any faults);
  - fault forecasting(forecast remaining unknown faults).

Basic Definitions

- The Dependability attributes
  - allow us to <u>express the properties</u> that are desired from the system;
  - allow us to quantitatively evaluate the system's qualities

Basic Definitions

- Dependability attributes:
  - readiness to provide the desired service: *availability*;
  - continuity of providing the service *reliability*;
  - capability of avoiding catastrophic failures: *safety*;
  - ease of being corrected: maintanibility;
  - avoid unauthorized information disclosure: confidentiality;
  - avoid unauthorized information alteration: integrity;

Basic Concepts and Terminology

- Basic Definitions
- Faults, Errors and Failures
- Dependability Attributes
- Means of Attaining Dependability
- Computational Systems and Safety

Faults, Errors and Failures

- ***Mission***:
  - desired device function (including correct specification)
- ***Failure***:
  - the non-fulfillment of the mission
  - this may be:
    - » Momentary, Tempory (e.g. due to repairs), or Permanent
    - » there are many other ways to classify these...

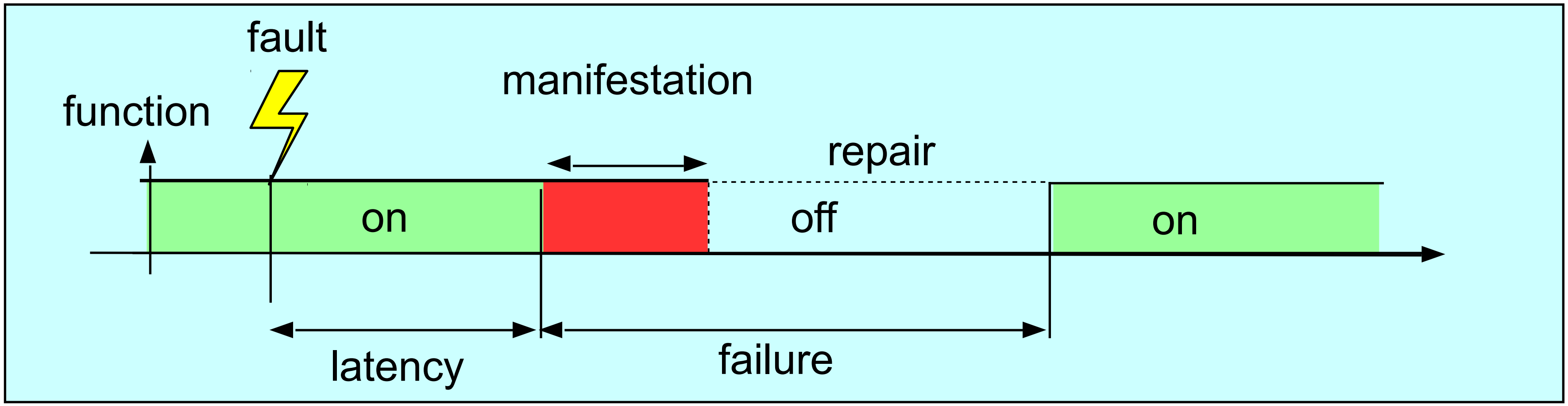
Faults, Errors and Failures

- *Fault*:
  - the cause of an error

    (may occur long before the failure itself)
- *Error*:
  - the manifestation of the fault

FAULT $\xrightarrow[\text{cause}]{\text{may}}$ ERROR $\xrightarrow[\text{cause}]{\text{may}}$ FAILURE

Faults, Errors and Failures



FAULT $\xrightarrow[\text{cause}]{\text{may}}$ ERROR $\xrightarrow[\text{cause}]{\text{may}}$ FAILURE

Faults, Errors and Failures

- Examples of a “Fault → Error → Failure” chain...
  - A software 'bug' introduced by a programmer is a fault;
    - » Once activated (ex.: invocation of function containing the 'bug'), the fault *may* produce an error;
    - » The error is an incorrect internal state;
    - » A failure occurs if/when the incorrect internal state affects the service being provided (either in value or time domain).

Faults, Errors and Failures

- Examples of a “Fault → Error → Failure” chain...
  - An electromagnetic interference / cosmic ray is a fault;
    - » this fault *may* result in an internal error (flipping of bits in communication wire / memory);
    - » these errors may result in failures (when reading the data of the wire / memory);

Faults, Errors and Failures

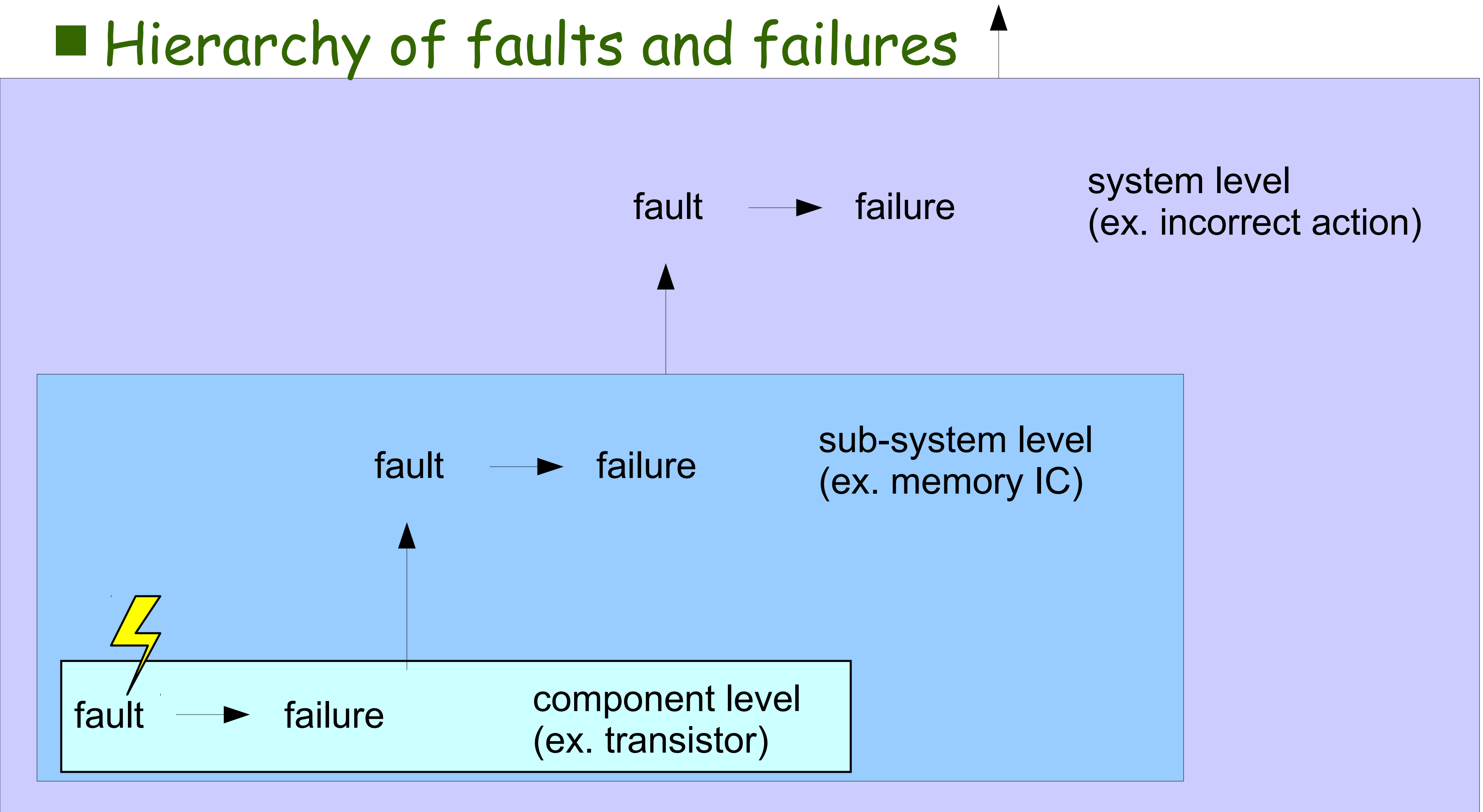
- Examples of a “Fault → Error → Failure” chain...
  - An incorrect operation made by a user is a fault (from the point of view of the overall system);
  - this fault may result in an incorrect internal state (error);
  - etc...

Faults, Errors and Failures

- Examples of a “Fault → Error → Failure” chain...
  - A failure by the editor of a maintenance or user manual may result in a fault becoming present in the manual;
  - This fault will remain inactive until someone follows the incorrect procedures detailed in the manual;
  - etc...

Faults, Errors and Failures

Hierarchy of faults and failures

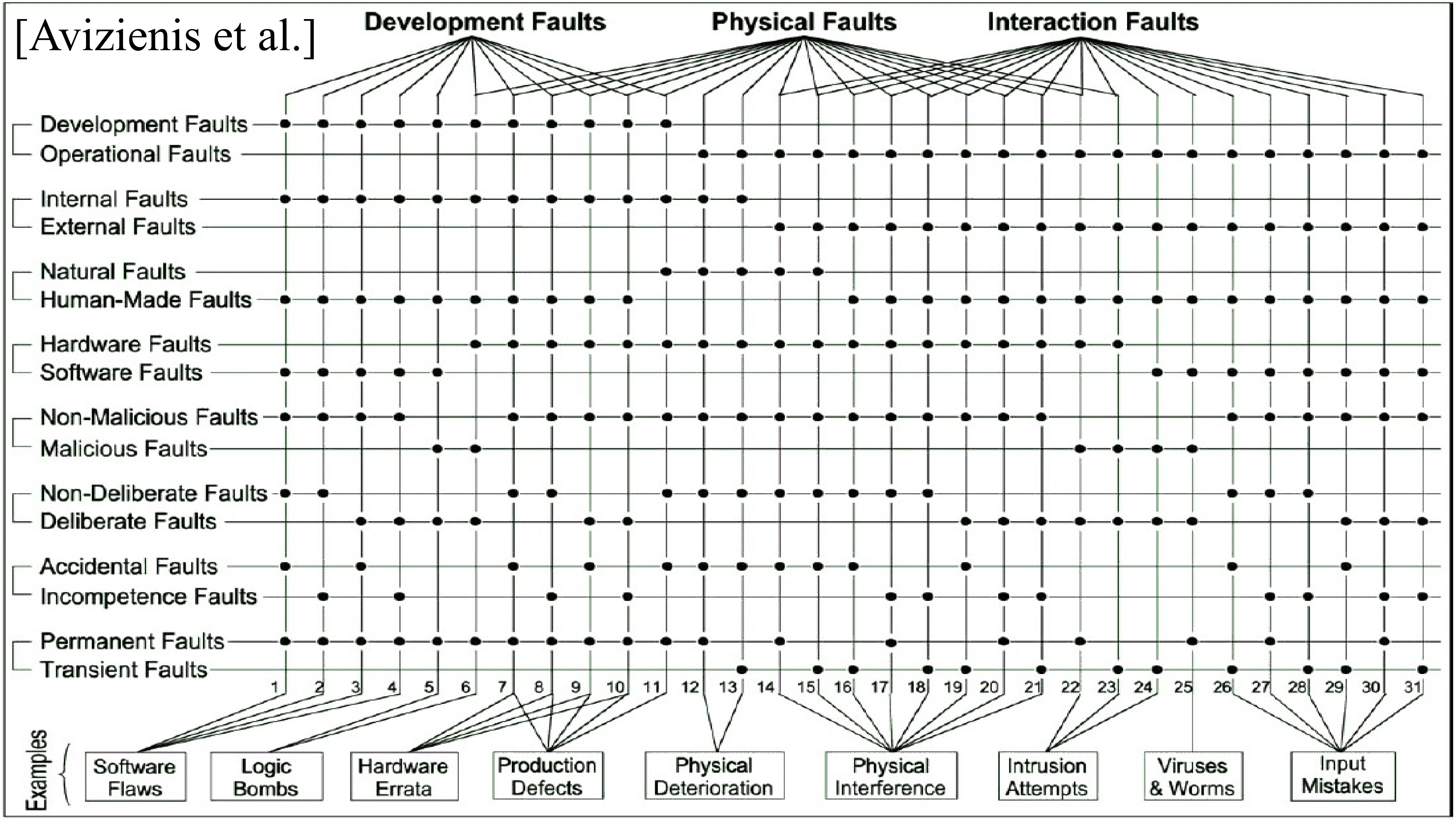


Faults, Errors and Failures

Fault Classification

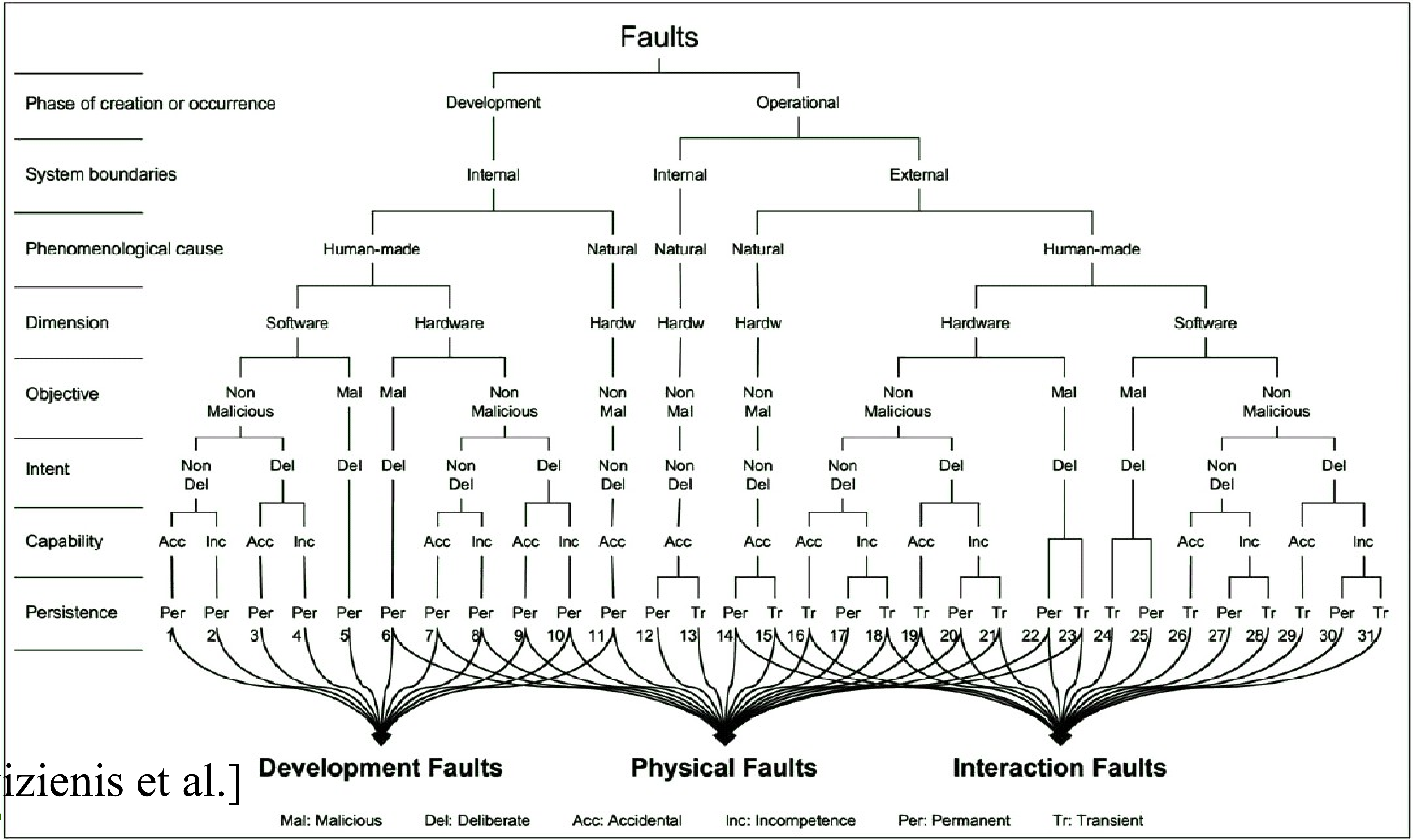
- Faults
  - Phase of occurrence
    - Development Faults
    - Operational Faults
  - System Boundaries
    - Internal Faults
    - External Faults
  - Phenomen. Causes
    - Natural Faults
    - Human-made Faults
  - Dimension
    - Hardware Faults
    - Software Faults
  - Objective
    - Non-malicious Faults
    - Malicious Faults
  - Intent
    - Non-deliberate Faults
    - Deliberate Faults
  - Capacity
    - Accidental Faults
    - Incompetence Faults
  - Persistence
    - Permanent Faults
    - Transient Faults

Faults, Errors and Failures: Fault Classification



[Avizienis et al.]

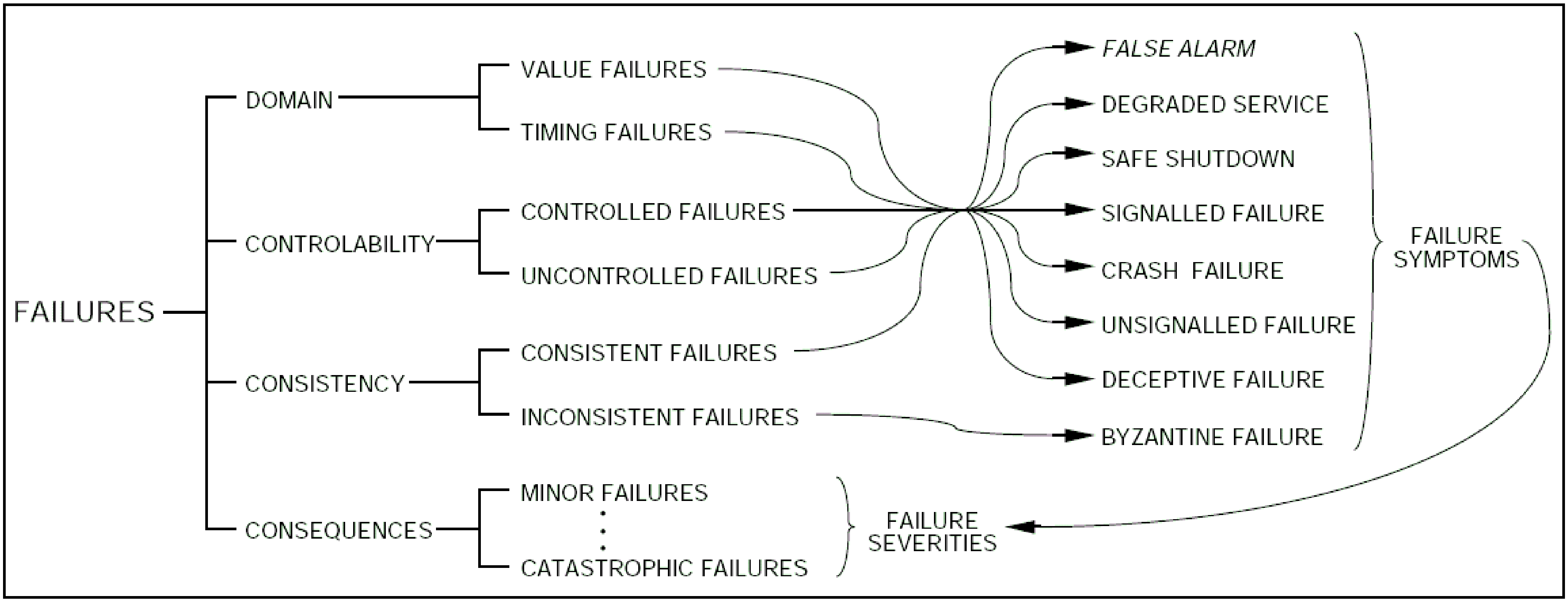
Faults, Errors and Failures: Fault Classification



[Avizienis et al.]

Re

Faults, Errors and Failures: Failures



[Avizienis et al.]

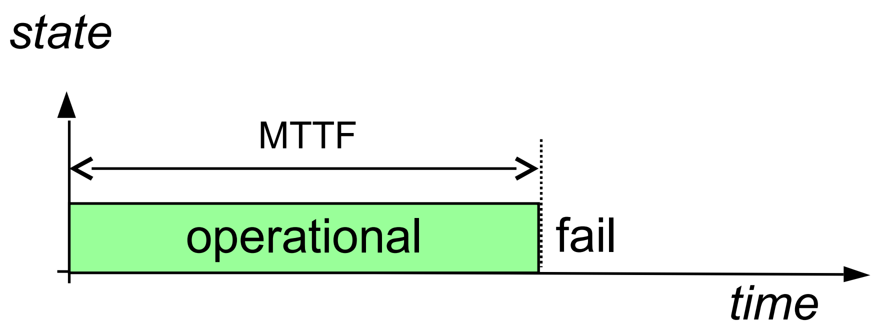
Basic Concepts and Terminology

- Basic Definitions
- Faults, Errors and Failures
- Dependability Attributes
- Means of Attaining Dependability
- Computational Systems and Safety

Dependability Attributes

- Dependability
  - Impairments
    - Faults
    - Errors
    - Failures
  - Methods
    - Obtention
      - Fault Tolerance
      - Fault Prevention
    - Validation
      - Fault Removal
      - Fault Forecasting
  - Attributes
    - Availability
    - Reliability
    - Safety
    - Maintainability
    - Security
      - Integrity
      - Confidentiality

Dependability Attributes: Reliability



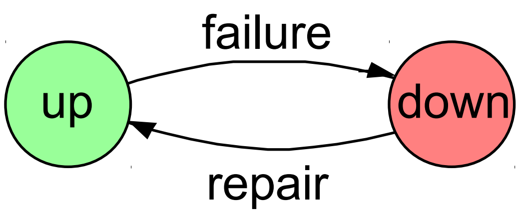
- Definition: R(t)

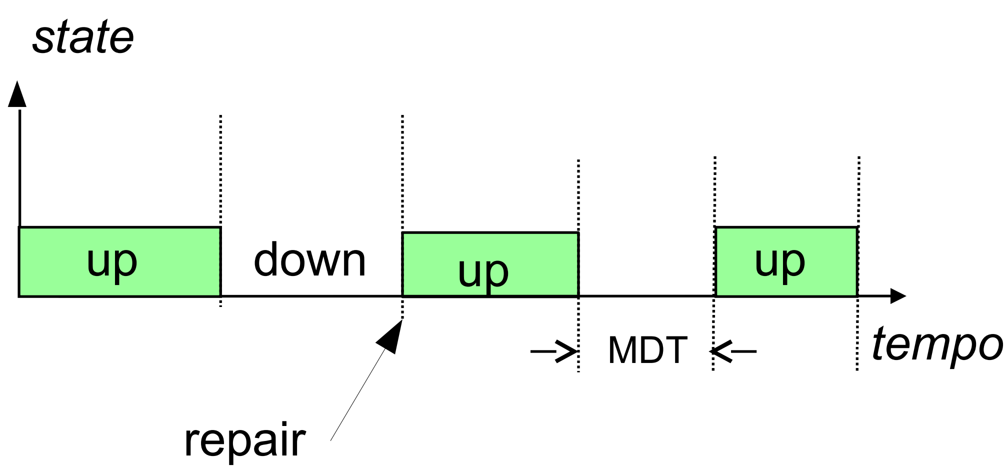
  ”probability that an item will perform its required function in the specified manner and under specified or assumed conditions *over a given time period*”

- Sometime expressed in MTTF:

  Mean Time To Failure

Dependability Attributes: Availability





MDT - Mean Down Time

- **Definition:**

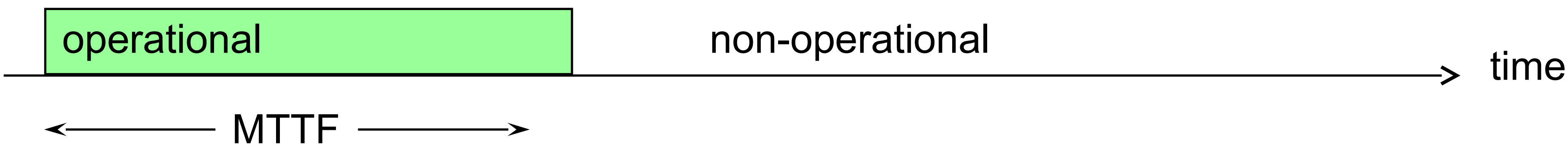
  ”probability that an item will perform its required function in the specified manner and under specified or assumed conditions *<u>at a given time</u>*”

- ***A***

  proportion (eg. in %) of time that an item is 'up'.

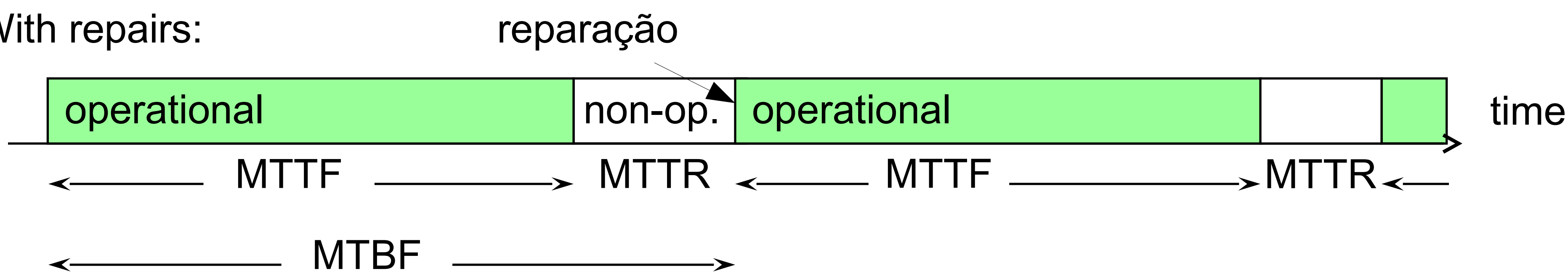
Dependability Attributes: Reliability vs Availability

No repairs:



MTTF: mean time to fail

With repairs:



MTBF – Mean Time Between Failures
MTTF – Mean Time To Fail
MTTR – Mean Time To Repair

MTBF = MTTF + MTTR

if MTTF >> MTTR
then MTBF ≈ MTTF

NOTE: sometimes MTBF – Mean Time Before Failures == MTTF

Dependability Attributes: Reliability vs Availability

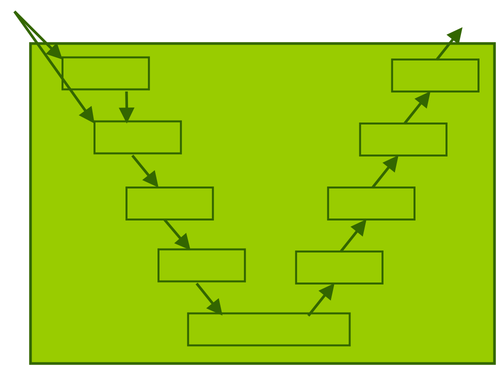
- Example:
  - A system that fails, on average, once an hour, but that reboots automatically in 10 ms, has low reliability, but high availability.
- Example of systems that require high availability, but low reliability:
  - stateless web servers, ...

Dependability Attributes: Safety

- Definition:
  - "safety is defined to be the absence of catastrophic consequences on the environment".
  - "freedom from accidents and loss" [Leveson 95]
- What is a <u>safety critical</u> system?
  - this is a system for which safety (absence of catastrophic failures) is guaranteed to a specific value.

Dependability Attributes: Safety

- Safety concerns
  - The safety guarantee of a system is not only related to the hardware and/or software used, but encompasses all aspects during the system's life-cycle, from its design, implementation, installation, use, maintenance, and decommissioning.



Dependability Attributes: Reliability vs Safety

- Reliability and Safety are not synonyms.
  - » For example, the system may remain safe in case of failure ("*failsafe state / system*"), in which case the system is safe, but not reliable.
- Frequently, the safety has to be reduced in order to allow other requirements to be met!

Dependability Attributes: Reliability vs Safety

- Example:
  - Traffic lights:
    - » all red light ⇨ system is safe, but not reliable
  - Fly-by-wire aircraft
    - » once an aircraft is airborne, there are no fail-safe states, so, in this case, Reliability = Safety

Dependability Attributes: Reliability vs Safety

- Safe operation in case of Failure: “*Failsafe*”
  - If a safe state exists and is reachable from all operational conditions, the system design may use that safe state to obtain safety in case of failure...
    - » ...by changing to that safe state in case of failure, and remaining in that state..

Dependability Attributes: Integrity

- Integrity
  - *System Integrity*:
    "absence of improper system state alterations"

    NOTE: *goes beyond the usual definitions*, that
    - relate to the notion of authorized actions only, and,
    - focus on information
  - *Data Integrity*:
    "property that the data be resistant to unauthorized modification"

Dependability Attributes: Maintainability

- Maintanability
  - "aptitude of a system to undergo repair and evolution"
  - More difficult to quantify, but usually expressed by MTTR – Mean Time To Repair
  - Problem: Some corrective and preventative maintanance may introduce new faults, or activate existing faults.

Basic Concepts and Terminology

- Basic Definitions
- Faults, Errors and Failures
- Dependability Attributes
- Means of Attaining Dependability
- Computational Systems and Safety

Means of Attaining Dependability

- Dependability
  - Impairments
    - Faults
    - Errors
    - Failures
  - Methods
    - Obtention
      - Fault Tolerance
      - Fault Prevention
    - Validation
      - Fault Removal
      - Fault Forecasting
  - Attributes
    - Availability
    - Reliability
    - Safety
    - Maintainability
    - Security
      - Integrity
      - Confidentiality

Means of Attaining Dependability

- Fault Prevention:

  Do whatever it takes to prevent the faults from occurring, or from introducing them during design and development.

  In software Development, this usually means:

  - **Formal**, **Semi-formal**, and **engineering methods** to be used and applied during software development;
  - These methodologies are to be used throughout the software's life-cycle (requirements analysis, specification, design, implementation, verification, validation, ...)

Means of Attaining Dependability

- Fault Tolerance:

No matter how hard we try, we will never be able to prevent all faults from occurring. Therefore, the next step is to tolerate the occurrence of faults, i.e. allow the system to continue operating correctly even in the presence of faults that have caused an error.

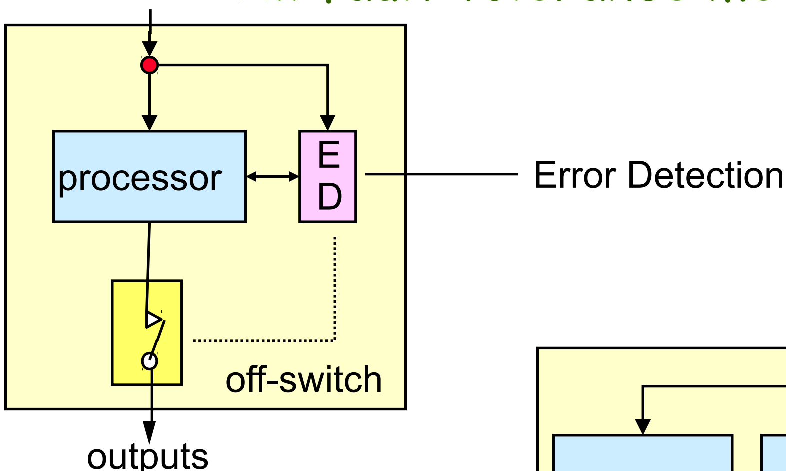
This is achieved through redundancy.

- Hardware redundancy;
- Software redundancy / diversity;
  (time diversity, data diversity, design/implementation diversity)

Means of Attaining Dependability

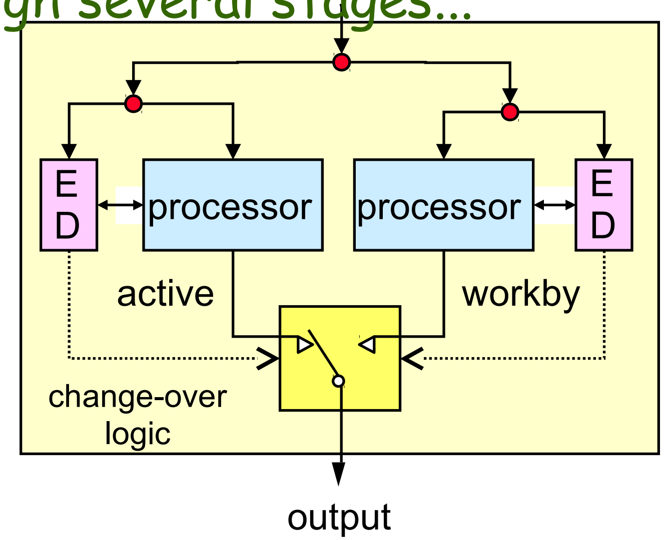
■ Fault Tolerance: the three main architectures

All fault tolerance mechanisms go through several stages...



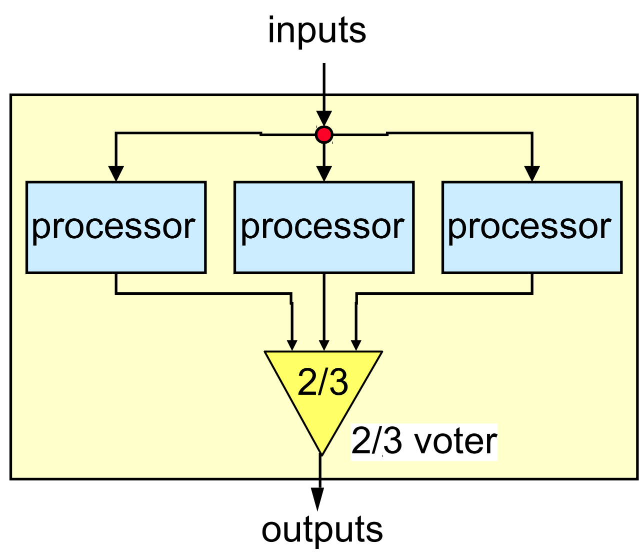
a) Integer

" rather nothing than wrong "
(fail-silent, fail-stop, fail-safe)



b) Persistent

" rather wrong than nothing "
fail-operate



c) Integer & persistent

error masking, massive redundancy

Means of Attaining Dependability

Fault Tolerance:

All fault tolerance mechanisms go through several stages…

- **Fault Tolerance**
  - **Error Detection**
    - Concurrent Detection
    - Preemptive Detection
  - **Recovery**
    - **Error Handling**
      - Roll Back
      - Roll Forward
      - Compensation
    - **Fault Handling**
      - Diagnosis
      - Isolation
      - Reconfiguration
      - Reinitialization

Means of Attaining Dependability

- Fault Removal:

  Reduce the number and severity of faults

  - **During System Use**

    This is essentially Preventative or Corrective Maintenance;

  - **During System Development**

    consists of three steps: verification, diagnosis, and correction.

    verification : the process of checking whether the system adheres to given properties (the term **validation** is often used when checking the specification).

Means of Attaining Dependability

- Fault Removal:
  - » **Static Verification**
    - **On the system itself**

      1) static analysis (*e.g.*, inspections or walk-through, data flow analysis, complexity analysis, abstract interpretation, compiler checks, vulnerability search, etc.)

      2) **theorem proving**;

    - **On models of the system**

      For example, on a state-transition model (Petri nets, finite or infinite state automata), using model checking.

Means of Attaining Dependability

- Fault Removal:

  » **Dynamic Verification**

  Check the system by exercising it...

  - **Symbolic Execution**
    Use symbolic inputs
  - **Testing**
    Use actual inputs
    (The question here is how to create the test vectors so as to achieve the largest coverage in the minimum time and costs)

Means of Attaining Dependability

- Fault Forecasting:

  Estimate the number and consequence of any remaining faults that we were unable to prevent and remove.

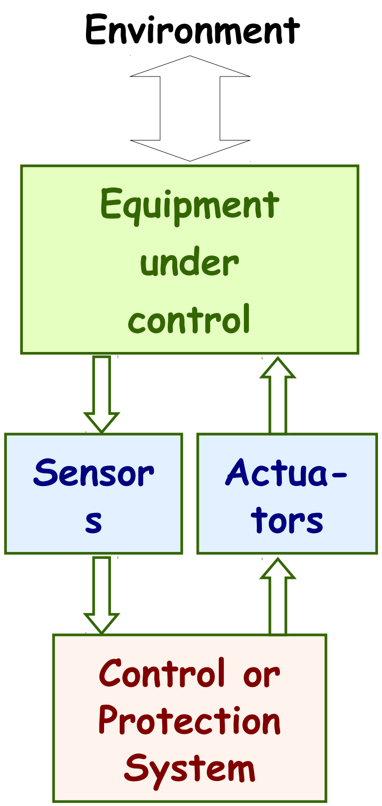
  - How do you do this for software?

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Basic Concepts and Terminology

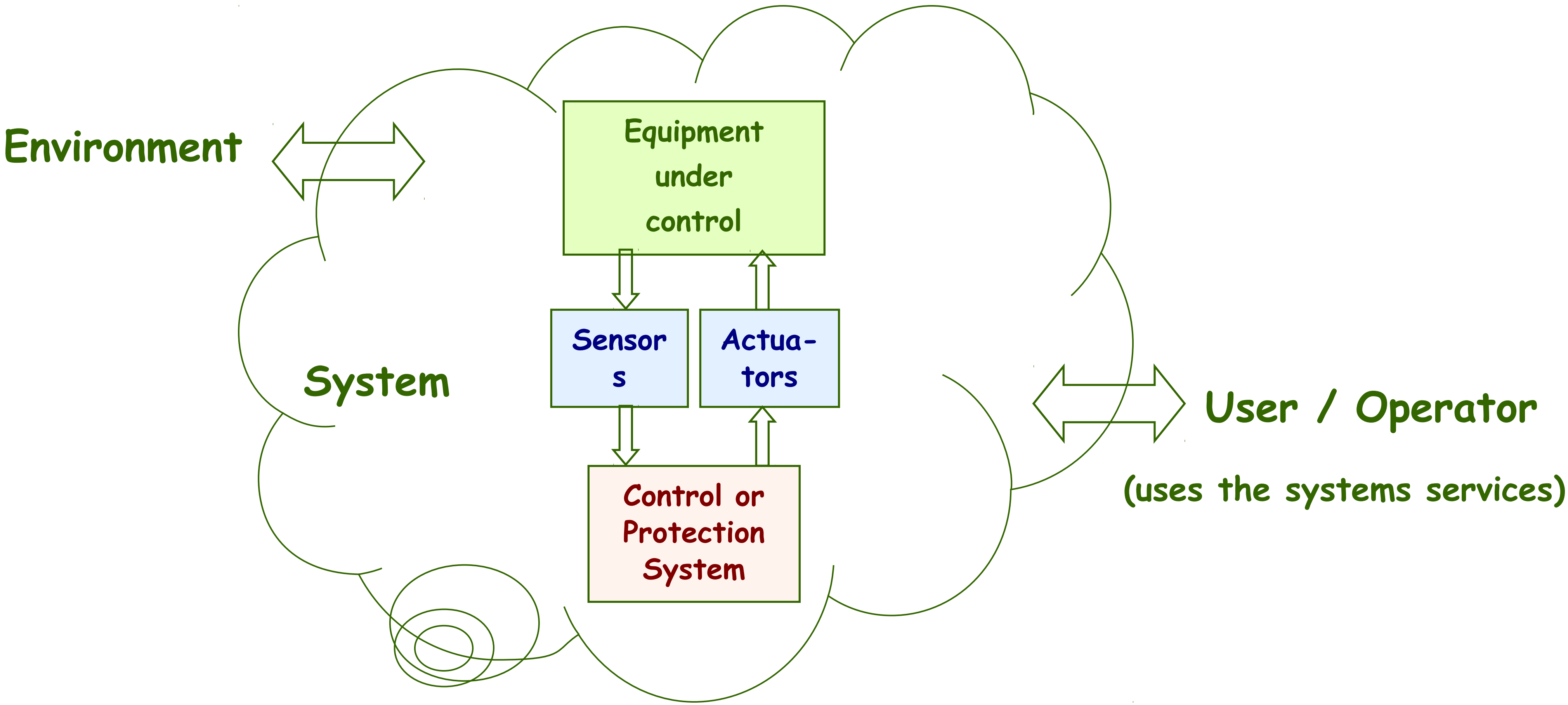
- Basic Definitions
- Faults, Errors and Failures
- Dependability Attributes
- Means of Attaining Dependability
- Computational Systems and Safety

Computational Systems and Safety



- **Control System.**
  - Determnines/controls how a specific system operates (simples <-> complexo).
  - If specified, also provides safety functions (i.e. a safety-critical system);
- **Protecction System.**
  - Uses sensors to detect error conditions, and activates actuators to minimize or cancel their effects;
  - Special case: "*shutdown system*".

Computational Systems and Safety



Computational Systems and Safety

- Tendency:

  "Software is a pervasive Enabling technology"

  - Today, multiple safety-critical functions are already supported by computational systems;

  - Embedded systems will have / already have a dominating role in our interaction with computational systems;

  - The development of embedded systems will soon be one of the largest clients of safety-critical technologies (hardware/software/...)

Computational Systems and Safety

- Advantages of using PES (Programmable and Electronic Systems) to support safety-critical applications
  - High pocessing capacity, which eases the implementation of complex control functions (anc control large number of I/Os) which would be difficult otherwise;
  - High speed / low energy consumption / small dimensions / low cost;
  - High flexibility for system evolution (implies cost reduction);
  - High hardware reliability, due to high integration (reduced number of external interfaces susceptible to fail);

Computational Systems and Safety

- Disadvantages of using PES (Programmable and Electronic Systems) to support safety-critical applications
    - higher design complexity and higher probability of introducing development faults;
    - test operations more complex =>
      => higher probability of undetected faults
    - higher complexity in using and operating the system =>
      => higher probability of human instalation, operation and maintance faults

Basic Concepts and Terminology

- Bibliografy

  - "Safety-Critical Computer Systems", Neil Storey, Chapters 1 and 2
  - "Fundamental Concepts of Dependability", A. Avizienis, J-C. Laprie, B. Randell
  - "Dependability and its Threats: A Taxonomy", A. Avizienis, J-C. Laprie, B. Randell
  - "Quality Attributes", M. Barbacci, T. H. Longstaff, M. H. Klein, C. B. Weinstock, , CMU/SEI-95-TR-021, December 1995.
  - "Quality Attributes and Service-Oriented Architectures", L. O'Brien, L. Bass, P. Merson, CMU/SEI-2005-TN-014, September 2005.