POLITECNICO MILANO 1863

Computing Infrastructures
-
System Dependability

Reliabiltity and Availability

The topics of the course: what are we going to see today?

HW Infrastructures:

System-level: Computing Infrastructures and Data Center Architectures, Rack/Structure;

Node-level: Server (computation, HW accelerators), Storage (Type, technology), Networking (architecture and technology);

Building-level: Cooling systems, power supply, failure recovery

SW Infrastructures:

Virtualization: Process/System VM, Virtualization Mechanisms (Hypervisor, Para/Full virtualization)

Computing Architectures: Cloud Computing (types, characteristics), Edge/Fog Computing, X-as-a service

Methods:

Reliability and availability of datacenters (**definition**, **fundamental laws**, RBDs)

Disk performance (Type, Performance, RAID)

Scalability and performance of datacenters (definitions, fundamental laws, queuing network theory)

What dependability is? (SLIDE Form the OVERVIEW SET)

- A measure of how much we trust a system…

  …from a microwave oven up to an airplane and a large datacenter!
- The ability of a system to perform its functionality while exposing:
  - **Reliability**
  - **Availability**
  - Maintainability
  - Safety
  - Security

Continuity of correct service

Readiness for correct service

Ability for easy maintainance

Absence of catastrophic consequences

Confidentiality and integrity of data

Reliability

The ability of a system or component to perform its required functions under stated conditions for a specified period of time
[IEEE610]

[IEEE610]: IEEE Standard Glossary of Software Engineering Terminology, IEEE Std 610.12-1990 (R2002).

definition

R(t): probability that the system will operate correctly in a specified operating environment until time *t*

$$R(t) = P(\text{not failed during } [0, t])$$

assuming it was operating at time t = 0

- *t* is important!
- If a system needs to work for slots of ten hours at a time, then ten hours is the reliability target
- Often used to characterize systems in which even small periods of incorrect behavior are unacceptable (e.g. Impossibility to repair)

Characteristics

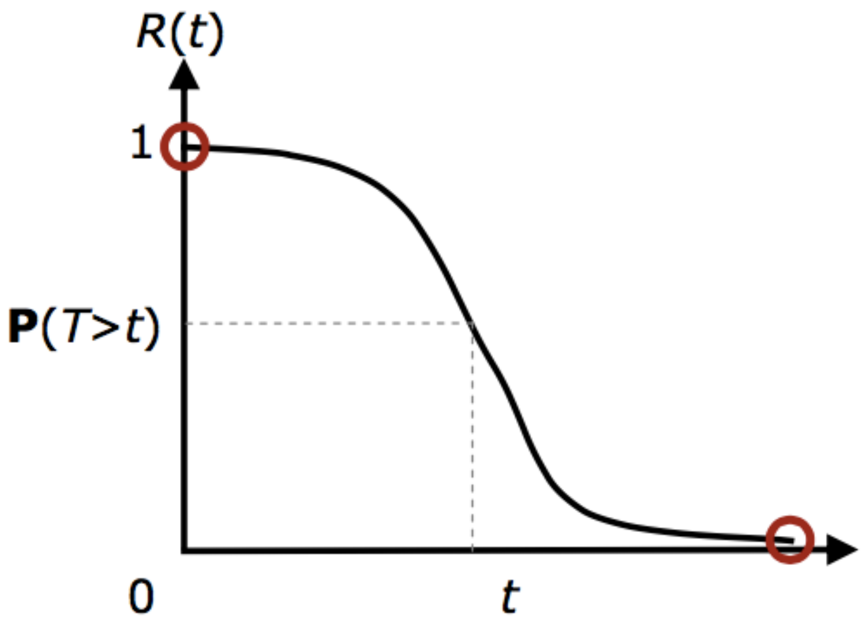
1 – R(*t*): unreliability, also denoted Q(*t*)

R(*t*) is a non-increasing function varying from 1 to 0 over [0,+∞)

$$R(0) = 1$$

$$\lim_{t \to +\infty} R(t) = 0$$

$$f(x) = -\frac{\mathrm{d}R(t)}{\mathrm{d}t}$$



probability density function of the failure

Availability

The degree to which a system or component is operational and accessible when required for use [IEEE610]

- [IEEE610]: IEEE Standard Glossary of Software Engineering Terminology, IEEE Std 610.12-1990 (R2002).

Availability = Uptime / (Uptime + Downtime)

- Fundamentally different from reliability
  - "reliability: does not break down …"
  - "availability: even if it breaks down, it is working when needed…
- Any examples of designs requiring high reliability/availability?

definition

$A(t)$: probability that the system will be operational at time t

$$A(t) = P(\text{not failed at time } t)$$

- Literally, readiness for service
- Admits the possibility of brief outages

- When the system is not repairable: $A(t) = R(t)$
- In general (repairable systems): $A(t) \geq R(t)$

- $1 - A(t)$: unavailability

Some numbers

Availability

Availability as a function of the "number of 9's"

Number of 9's	Availability	Downtime (mins/year)	Practical meaning
1	90%	52596.00	~5 weeks per year
2	99%	5259.60	~4 days per year
3	99.9%	525.96	~9 hours per year
4	99.99%	52.60	~1 hour per year
5	99.999%	5.26	~5 minutes per year
6	99.9999%	0.53	~30 secs per year
7	99.99999%	0.05	~3 secs per year

Some example

Number of 9's	Availability	Downtime/year	System
2	99%	~4 days	Generic web site
3	99.9%	~9 hours	Amazon.com
4	99.99%	~1 hour	Enterprise server
5	99.999%	~5 minutes	Telephone system
6	99.9999%	~30 seconds	Network switches

R(t) & A(t): Two points of view

Of course, they are related:

- if a system is unavailable it is not delivering the specified system services

It is possible to have systems with low reliability that must be available

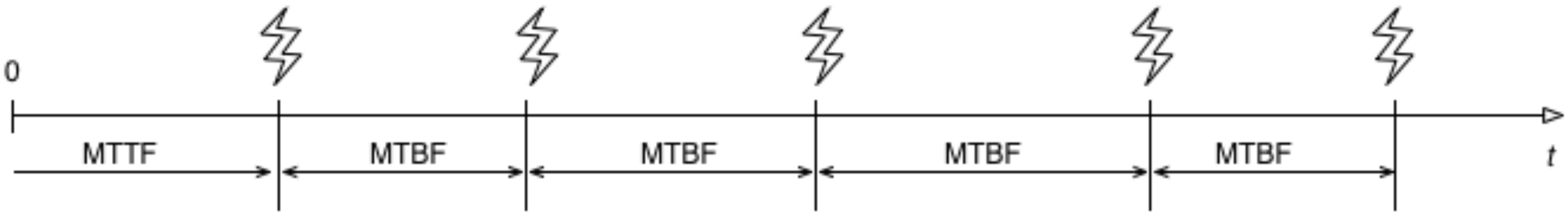
- system failures can be repaired quickly and do not damage data, low reliability may not be a problem (for example a database management system)

The opposite is generally more difficult…

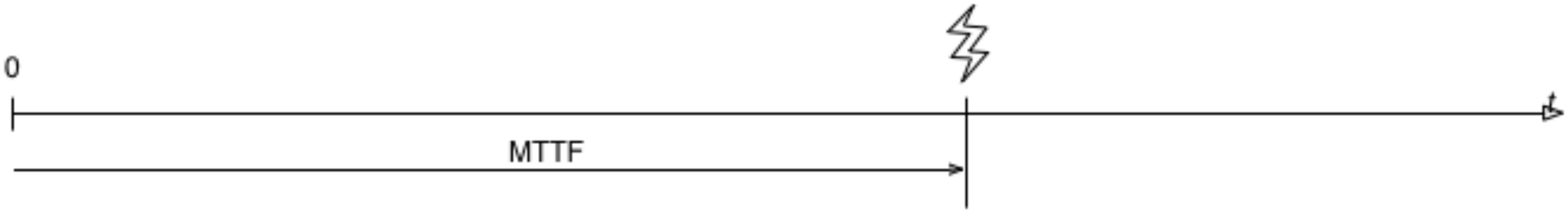
R(t) & A(t) related indices

MTTF (Mean Time To Failure): mean time before *any* failure will occur

MTBF (Mean Time Between Failures): mean time between two failures



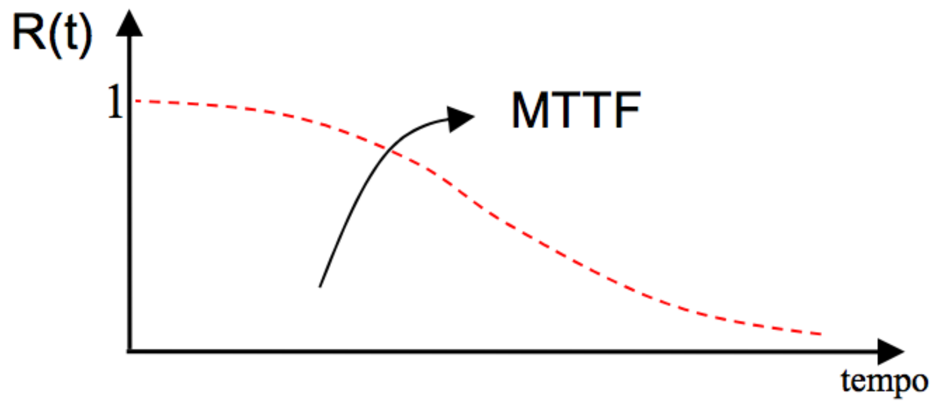
hypothesis: negligible repair time



R(t) & A(t) related indices

MTTF (Mean Time To Failure): mean time before *any* failure will occur

$$MTTF = \int_0^{\infty} R(t)\,dt$$



R(t) & A(t) related indices

Other indices

MTTF: mean time to (first) failure, the up time before the first failure

MTBF: mean time between failures

$$\text{MTBF} = \frac{\text{total operating time}}{\text{number of failures}}$$

FIT: failures in time

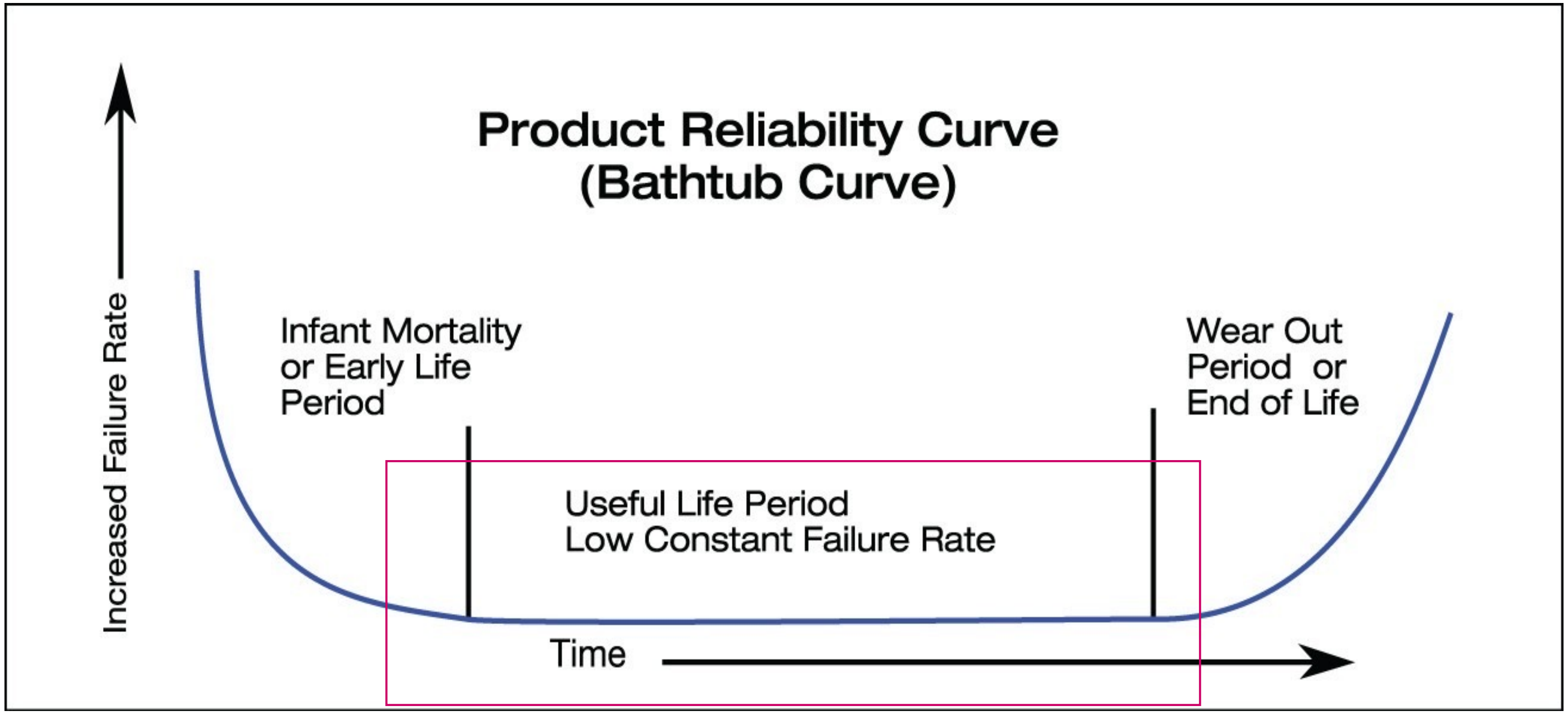
- another way of reporting MTBF
- the number of expected failures per one billion hours (10^9) of operation for a device
- MTBF (in h) $=10^9/\text{FIT}$

$$\text{Failure Rate } \lambda = \frac{\text{number of failures}}{\text{total operating time}}$$

$$\text{MTBF} = \frac{1}{\lambda}$$

R(t) & A(t) related indices

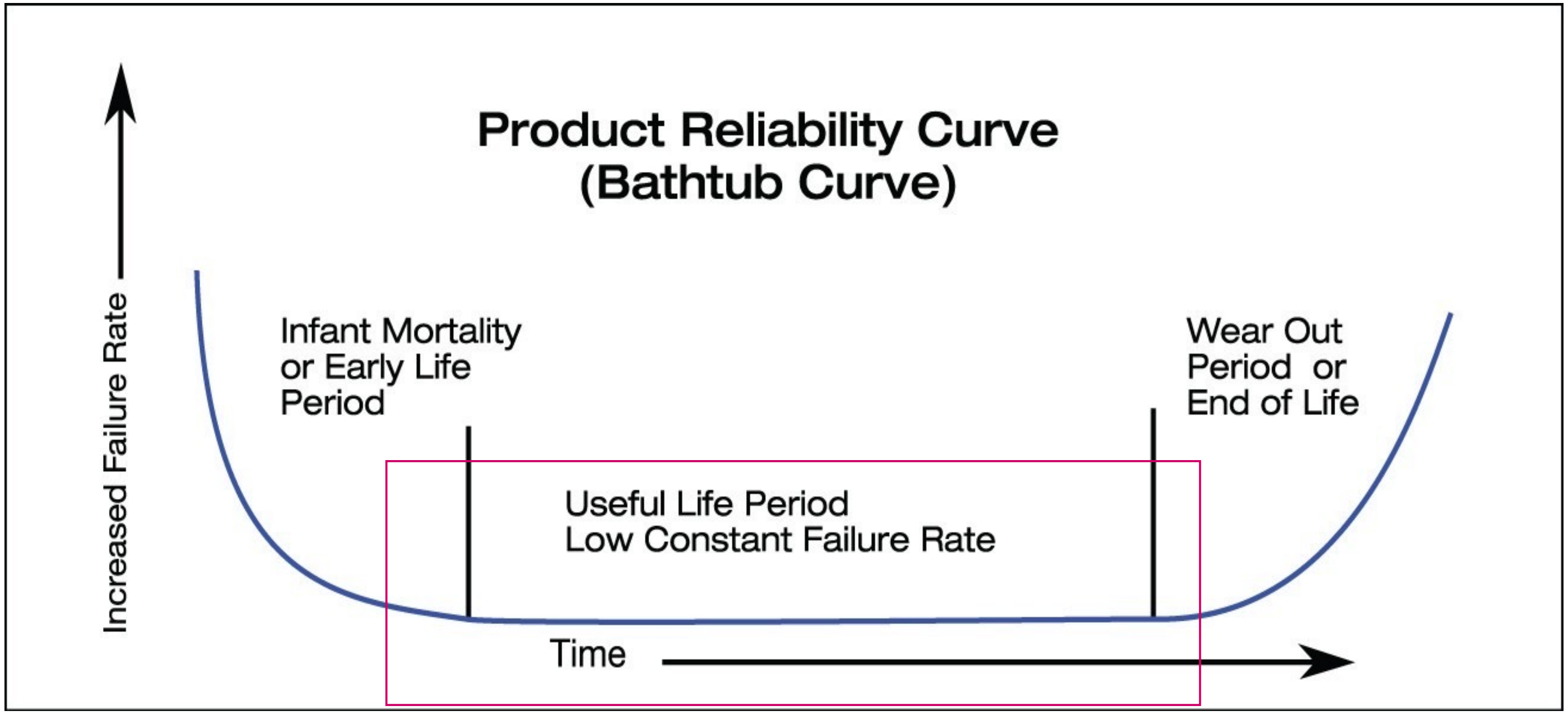
Other indices



- *Infant Mortality*: failures showing up in new systems. Usually this category is present during the testing phases, and not during production phases.
- *Random Failures*: showing up randomly during the entire life of a system.
  - **Our main focus**
- *Wear Out*: at the end of its life, some components can cause the failure of a system. **Predictive mainteinance** can reduce the number of this type of failures.

R(t) & A(t) related indices

Other indices

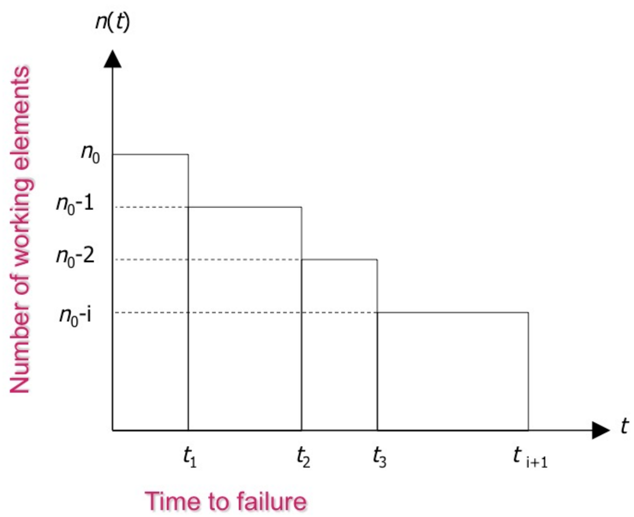


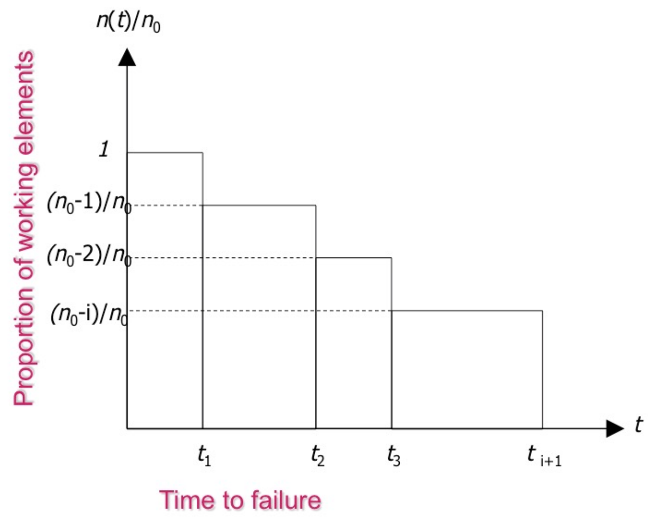
How to identify defective products and calculate MTTF?

Burn-in test: *stress* the system with excessive temperature, voltage, current, humidity so to accelerate wear out.

How to compute reliability? Empirical Evaluation

- Let's consider n_0 independent and statistically identical elements deployed at time $t = 0$ in identical conditions $n(0) = n_0$
- At time t, $n(t)$ elements are not failed
- $t_1, t_2, \ldots t_{n0}$ are the times to failure of the n_0 elements
  - times to failure are independent occurrences of the random quantity τ





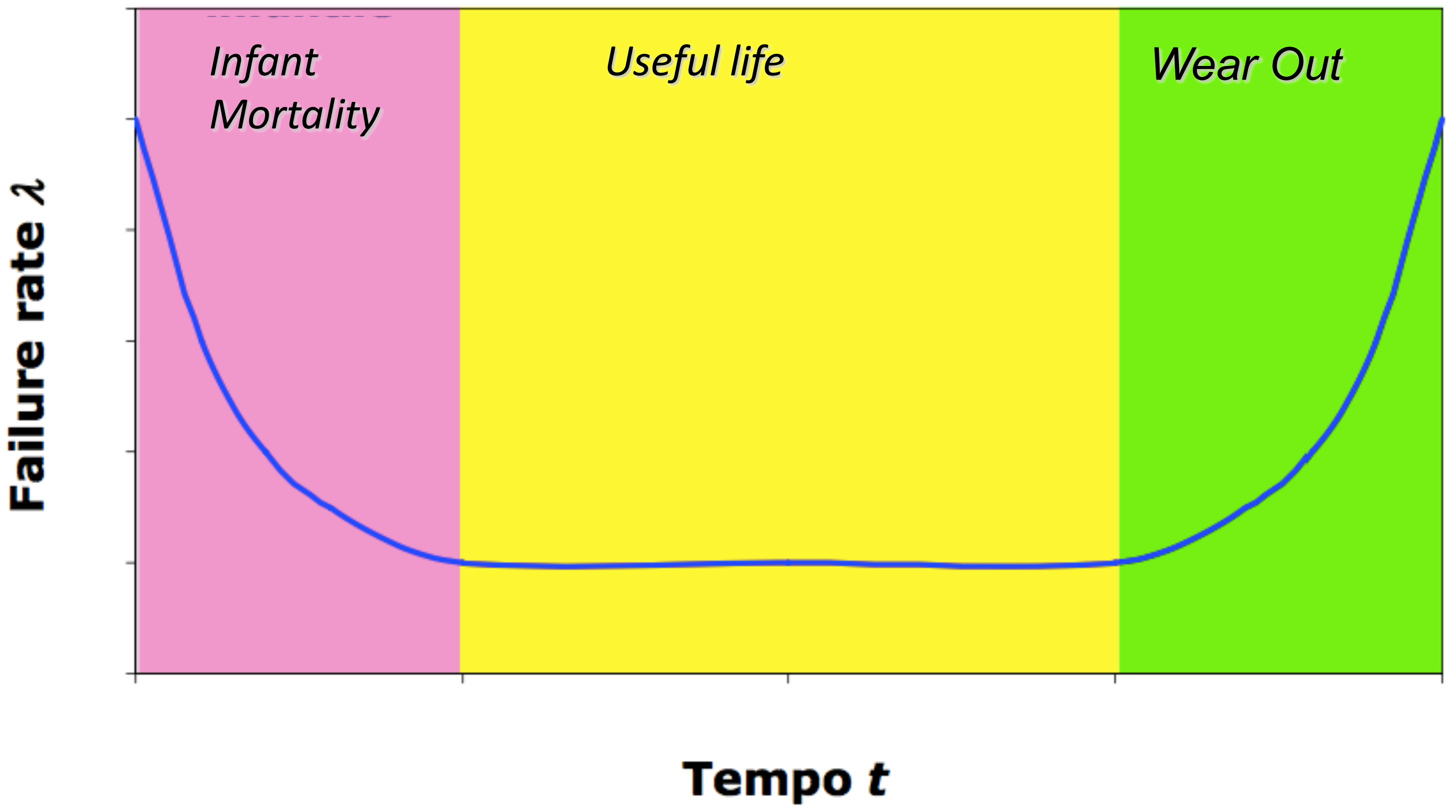
- Function $n(t) / n_0$ is the empirical function of reliability that as $n_0 \rightarrow \infty$ converges to the value: $\boldsymbol{n(t) / n_0 \rightarrow R(t)}$

Different types of faults

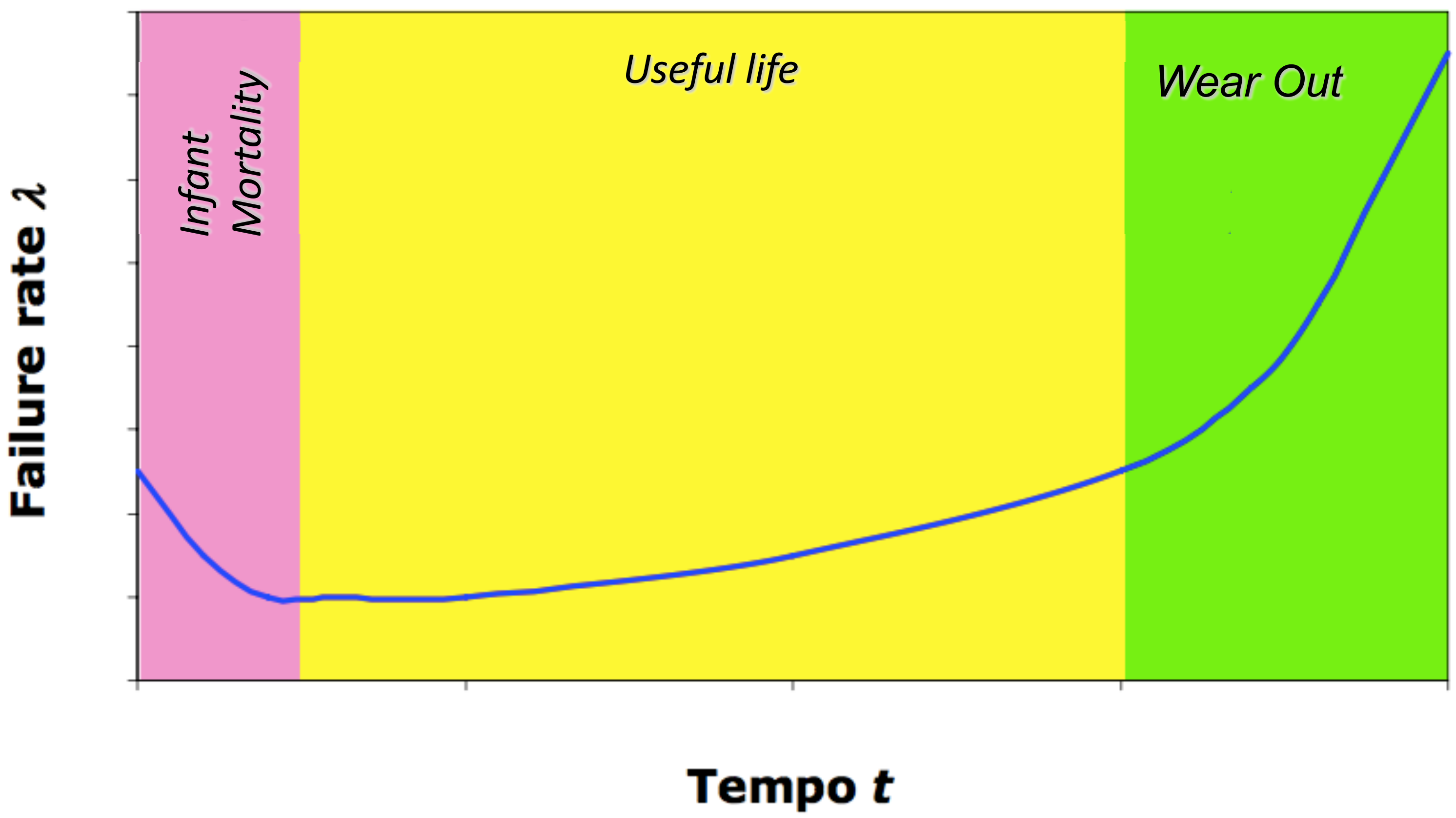
- **Hardware faults**:
  - Electronic
  - Mechanical

- **Software faults**:

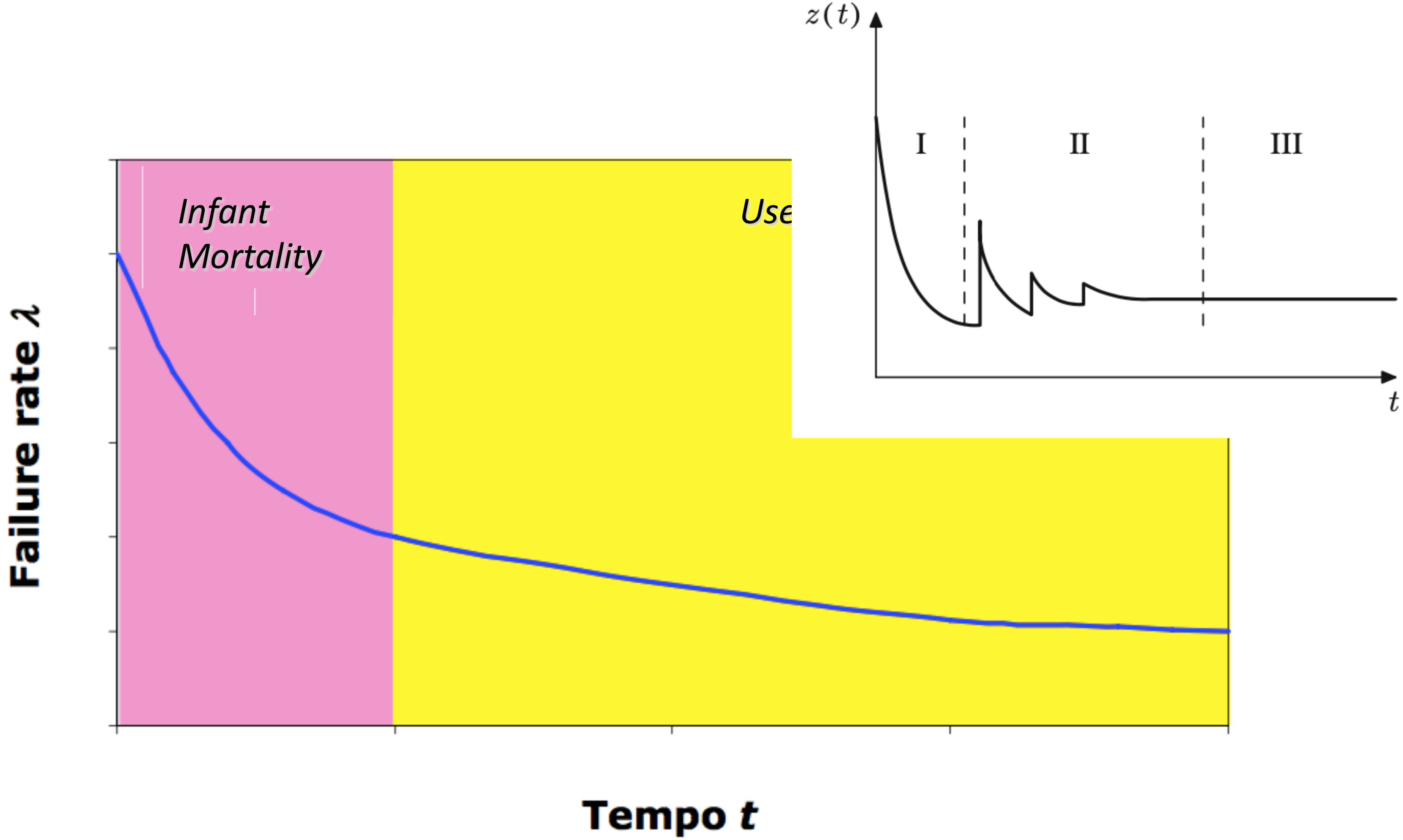
Hardware: electronic components



Hardware: Mechanical components



Software



R(t) … what to do?

Exploitation of R(t) information is used to compute, for a complex system, its reliability in time, that is the <u>expected lifetime</u>

– computation of the MTTF

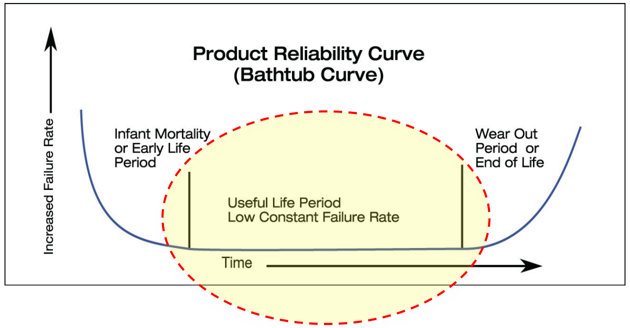
Computation of the overall reliability starting from the components' one

Within the course we consider only a reliability with an exponential distribution:

- Constant Failure rate

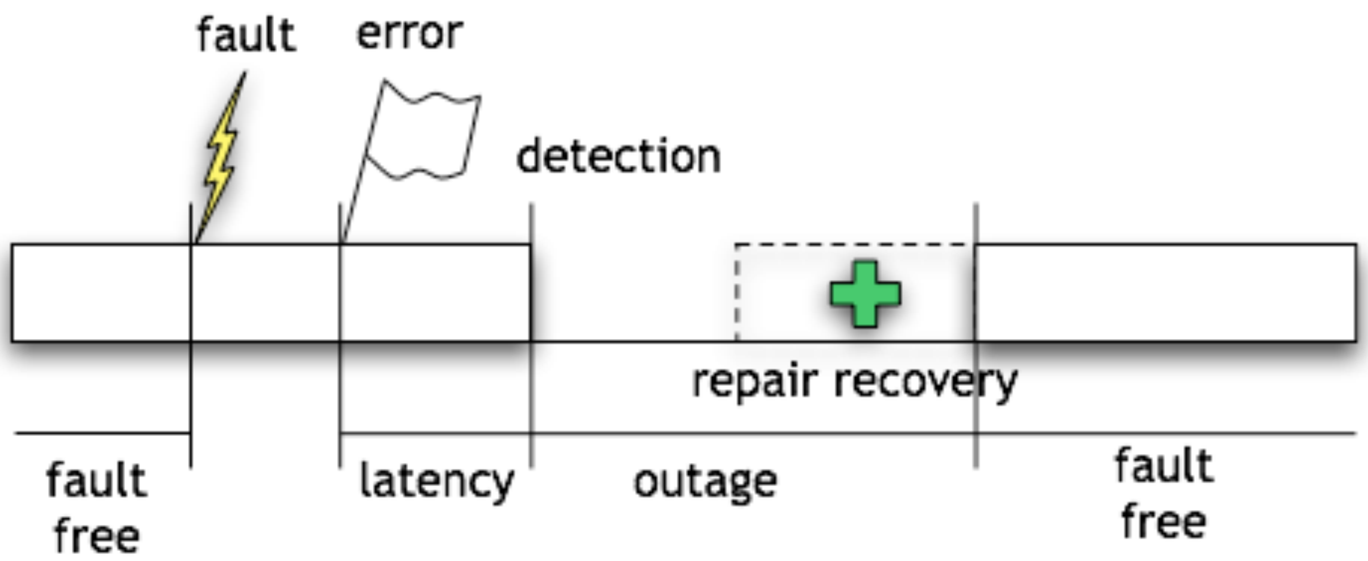
$$R(t) = e^{-\lambda t}$$

$$MTTF = \int_0^\infty R(t)\,dt = \frac{1}{\lambda}$$



Reliability terminology

Term	Description
Fault	A defect within the system
Error	A deviation from the required operation of the system or subsystem
Failure	The system fails to perform its required function



Reliability terminology

An example: a flying drone with an automatic radar-guided landing system

Fault: electromagnetic disturbances interfere with a radar measurement

Error: the radar-guided landing system calculates a wrong trajectory

Failure: the drone crashes to the ground

Reliability terminology

Another example: a tele-surgery system

Fault: radioactive ions make some memory cells change value (bitflip)

Error: some frames of the video stream are corrupted

Failure: the surgeon kills the patient

Reliability terminology

Not always the *fault – error – failure chain* closes

example: a tele-surgery system

Fault: radioactive ions make some memory cells change value (bitflip) but the corrupted memory does not involve the video stream

Error: no frames are corrupted

Failure: the surgeon carries out the procedure

Reliability terminology

Not always the *fault – error – failure chain* closes

example: a tele-surgery system

Fault: radioactive ions make some memory cells change value (bitflip) but the corrupted memory does not involve the video stream

Error: no frames are corrupted

Failure: the surgeon carries out the procedu

Non activated fault

Reliability terminology

Not always the *fault – error – failure chain* closes

example: a flying drone with automatic radar-guided landing

Fault: electromagnetic disturbances interfere with a radar measurement

Error: the radar-guided landing system calculates a wrong trajectory, but then, based on subsequent correct radar measurements it is able to recover the right trajectory

Failure: the drone safely lands

Reliability terminology

Not always the *fault – error – failure chain* closes

example: a flying drone with automatic radar-guided landing

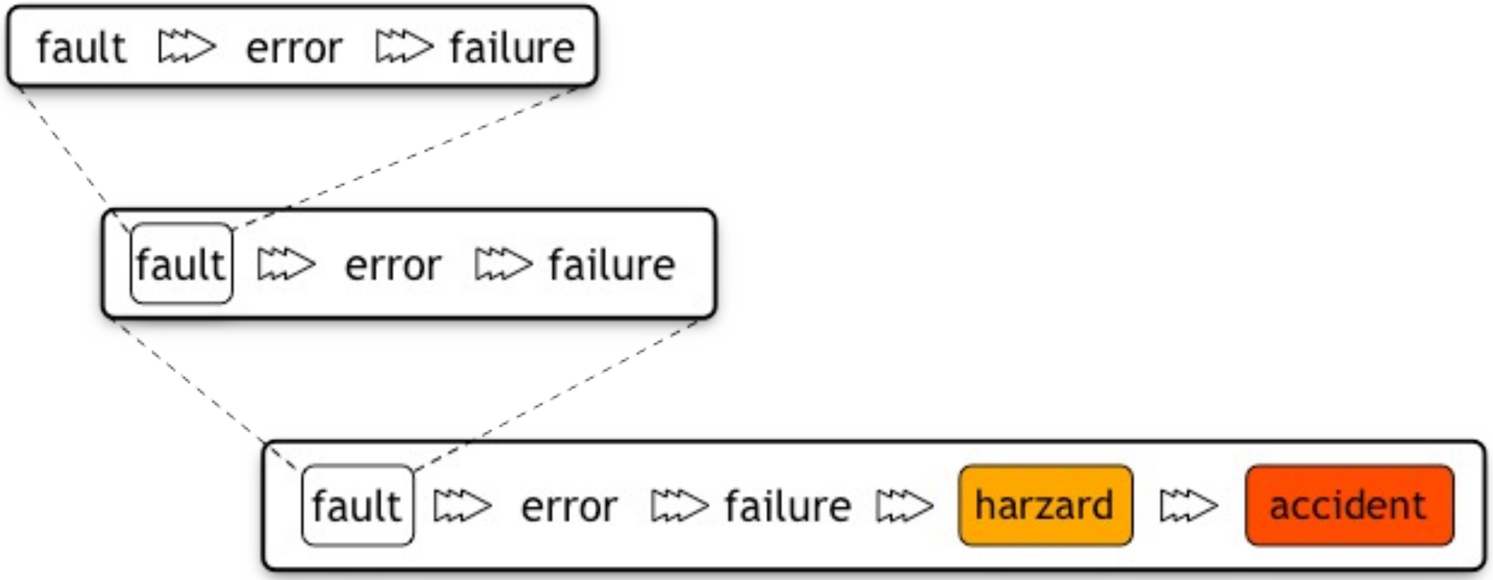
Fault: electromagnetic disturbances interfere with a radar measurement

Error: the radar-guided landing system calculates a wrong trajectory, but then, based on subsequent correct rada[…]ments it is ab[…]

Non propagated (or absorbed) error

Failure: the drone safely lands

Fault hierarchy



Fault-error-failure cascades can lead to life-threatening hazards