



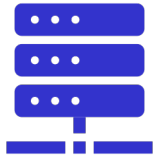
POLITECNICO
MILANO 1863

Computing Infrastructures - System Dependability Overview



POLITECNICO MILANO 1863

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?



What dependability is?

- A measure of how much we trust a system...
...from a microwave oven up to an airplane and a large datacenter!



What dependability is?

- 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

Continuity of
correct service

- Availability

Readiness for
correct service

- Maintainability

Ability for easy
maintainance

- Safety

Absence of
catastrophic
consequences

- Security

Confidentiality and
integrity of data



Why dependability?



Why dependability?

Functional Verification

A lot of effort is devoted to make sure the implementation

- matches specifications
- fulfills requirements
- meets constraints
- optimizes selected parameters (performance, energy, ...)

Nevertheless, even if all above aspects are satisfied ... things may go wrong

► systems fail

systems fail ... because something broke

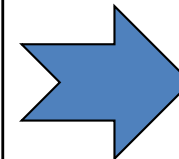
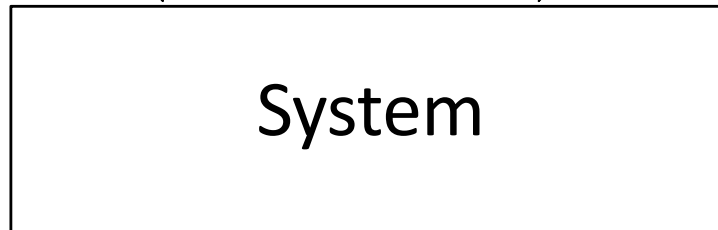
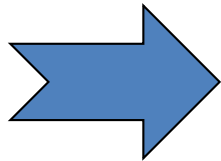


Why dependability?

Defects, Process variation,
Degraded transistors

Radiation, Noise

Inputs



“Acceptable” Outputs
+ Performance
+ Integrity
+ Availability
+ Security

Design errors,
Software bugs
OS bugs

Malicious attacks,
Human errors



Failure effects

Why dependability?

- A failure may have high costs if it **impacts economic losses** or physical damage
- A single **system failure may affect a large number of people**
- ***Systems that are not dependable are likely not be used or adopted***
- Undependable systems may cause information loss with a high consequent recovery cost
- ...



When to think about dependability?



When to think about dependability?

Both at design-time and at runtime

Always!!!



When to think about dependability?

Both at **design-time** and at runtime

- Analyse the system under design
- Measure dependability properties
- Modify the design if required



When to think about dependability?

Both at design-time and at runtime

- Detect malfunctions
- Understand causes
- React



When to think about dependability?

- Failures occur in development & operation
 - Failures in development *should* be avoided
 - Failures in operation *cannot* be avoided (things break), they must be dealt with
- Design should take failures into account and guarantee that control and safety are achieved when failures occur
- Effects of such failures should be predictable and deterministic ... not catastrophic



Where to apply dependability?



Where to apply dependability?

Why dependability?

Once upon a time ...

...dependability has been a relevant aspect only for safety-critical and mission-critical application environments

- Space
- Nuclear
- Avionics

Huge costs, acceptable only when mandatory ...



Mission-critical

Mission-critical systems: a failure during operation can have serious or irreversible effects on the mission the system is carrying out

- Satellites
- Surveillance drones
- Unmanned vehicles
- Automatic Weather Stations (and more in general sensors) in harsh environments



Safety-critical systems

Safety-critical systems: a failure during operation can present a direct threat to human life

- aircraft control systems
- medical instrumentation
- railway signaling
- nuclear reactor control systems



Computing Infrastructures:



Datacenters

Downtime is the enemy of every data center.

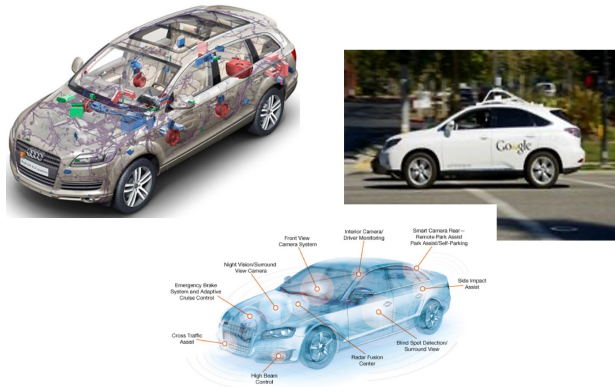
Aberdeen Research reports the following downtimes and incidents:

- “Average” performing facilities, 60 minutes with 2.3 incidents per year.
- Best-in-class organizations, 6 minutes with 0.3 incidents per year.



POLITECNICO MILANO 1863

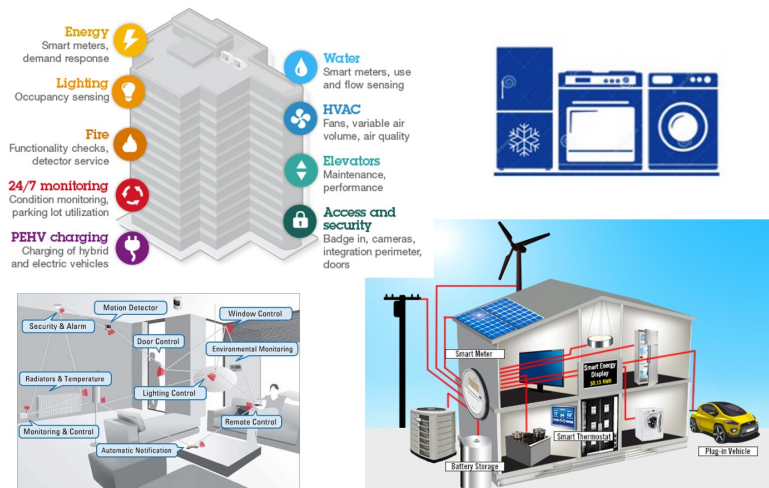
Computing Infrastructures



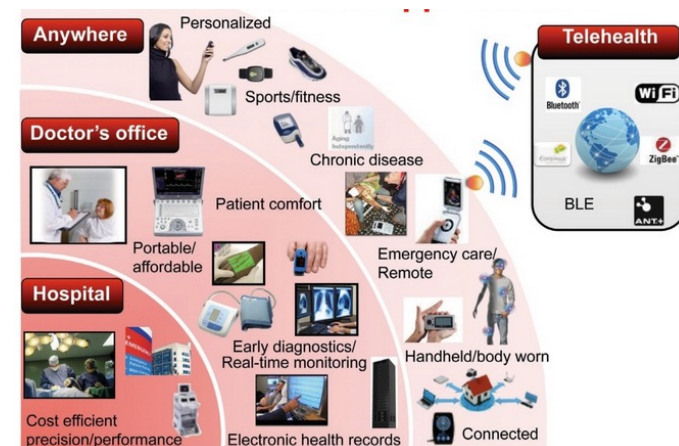
Automotive



Telecommunication



Smart Spaces



eHealth

Other scenarios...



POLITECNICO MILANO 1863

Anatomy of the scenarios: an example...

the nodes

- computing systems
- sensors and actuators

the communication

- network

**Everything has to work properly
for the overall system to be
working**

the cloud

- data storage
- data manipulation



How to provide dependability?



Failure avoidance vs tolerance paradigm

robust (computing) systems

AVOIDANCE

- Conservative design
- Design validation
- Detailed test
 - Hardware
 - Software
- Infant mortality screen
- Error avoidance

TOLLERANCE

- Error detection / error masking during system operation
- On-line monitoring
- Diagnostics
- Self-recovery & self-repair



Where to work

Technological level

- design and manufacture by employing reliable/robust components
 - *Highest dependability*
 - *High cost*
 - *Bad performance (generally devices from old generation)*



Where to work

Technological level

- design and manufacture by employing reliable/robust components

Architectural level

- integrate normal components using solutions that allow to manage the occurrence of failures
 - *High dependability*
 - *High cost*
 - *Reduced performance*



Where to work

Technological level

- design and manufacture by employing reliable/robust components

Architectural level

- integrate normal components using solutions that allow to manage the occurrence of failures

Software/application level

- develop solutions in the algorithms or in the operating systems that mask and recover from the occurrence of failures
 - *High dependability*
 - *High cost*
 - *Reduced performance*



Where to work

What do all solutions have in common?

- Cost
- Reduced performance

You have to pay for dependability

