

⋮

1. Introduction to Fault Tolerance

- What is *fault tolerance*?
- What is a *fault*?
- What distinguishes fault tolerant systems from others?
- Examples for fault tolerant and non fault tolerant systems.



⋮

1.1 Examples

- Examples of non fault tolerant systems
- Examples of fault tolerant systems



⋮

Examples of non-FTS

1. PC/Mac stand-alone system on my desk
 - frequent crashes (who hasn't lost a document?)
 - due to:
 - software bugs (commercial software / public domain)
 - viruses
 - incompatible software (more and more complex system)
 - hardware problems:
 - SCSI
 - hard disk / floppy failure
 - memory
 - processor (Pentium bug)



⋮

Examples of non-FTS

2. Workstation cluster in our department/company
 - often worse than stand-alone systems:
“Now it is sufficient for 1 machine to go down, to bring the cluster to a stand still”



⋮

Examples of non-FTS

- Software problems
- Hardware problems:
 - disk drives fail after 5 years (sooner than PC/Mac, since workstations are usually in 24 hour use)
 - mostly reliance on tape backup only (which from personal experience fails about once a week)
 - *solution*: RAID (random array of independent disks) makes it fault tolerant
 - network problems

⋮

⋮

Examples of non-FTS

3. Parallel systems we use for research
 - e.g. MasPar MP-1, MP-2
 - 16,384 PEs (SIMD-ACUs)
 - 1,024 PEs per board
 - if single PE fails \Rightarrow whole system inoperable (much higher failure probability than for single processor)
 - **Solution**: spare CPU board for replacement or reduction to half system size (8,192 PEs)

⋮

⋮

Examples of FTS

Application Areas

- (nuclear) power plants
- production environments
 - chemical industry (producing nitric acid is dangerous!)
 - automotive manufacturing (standing conveyor belt costs \$\$)

⋮

⋮

Examples of FTS

- Medical systems
 - life support systems
 - diagnostic system, e.g. x-ray (malfunctioning x-ray system lead to severe injuries/deaths of several patients!)

⋮



Examples of FTS

- Transportation systems
 - train/subway
 - ships
 - automobiles
 - ABS anti-locking-brakes
 - ESP electronic stability program
 - airbag activation
 - electronic ignition/fuel pump
 - aviation
 - *fly-by-wire*: “joy-sticks” instead of hydraulic steering systems, first introduced in Airbus A300



Examples of FTS

- Economics systems (networks) using Transaction Protocols
 - banking systems
 - stock market systems
 - ATM automated teller machines
 - BOCS (sport/music) event ticket sales
 - START world-wide flight reservation system (accessed by every travel agent)



Examples of FTS

- Telecommunication systems
 - e.g. Telstra/Optus telephone switching system
 - Operation: imagine the whole Australian phone system would break down for several hours
 - Accounting: German Telekom introduced an error when changing call charging times
 - millions (!) of non-retraceable (!!) accounting errors
 - every user got a flat call reduction



Examples of FTS

- Space Systems
 - satellites
 - imagine losing a satellite (several \$1,000,000) due to hardware/software failure, e.g. wrong steering command
 - unmanned probes for space exploration
 - Voyager, Galileo
 - manned space missions
 - US Space Shuttle, Russian Mir
 - Challenger/Columbia catastrophe
 - could FT design help??



⋮

Examples of FTS

- Robotics systems
 - manufacturing
 - autonomous vehicles
- Internet
 - redundancy through distributed information
- Military systems
- *What else?*

⋮

⋮

Examples of FTS

- Typical:
 - Specialized embedded system
 - Work in dangerous (hazardous) environments where hardware errors can occur
 - Applications where failure can lead to loss of money or injury/death

⋮

⋮

1.2 Definitions

[Levi, Agrawala]

- Def. 1 A **system** is an identifiable mechanism that maintains an observable behaviour at its interface with its environment, as a result of the set of all its possible system executions.
- Def. 2 A **process** is a communicating execution-instance of a program, whose implementation is achievable by a single-thread processor with a bounded address space and a set of resources.
- Def. 3. An **ASR** (authoritative system reference) is a fictitious entity which produces a correct behaviour of the system.

⋮

⋮

Definitions

[Levi, Agrawala]

- Def. 4 An **error** is a difference between the actual system behaviour and that produced by an ASR.
- Def. 5 A **failure** is an event which corresponds to the first occurrence of the generated error.
- Def. 6 A **fault** is a source which has the potential of generating errors [error is a manifestation of a fault]

⋮



Definitions

[Levi, Agrawala]

- Def. 7 A **permanent fault** is a potential source of generated errors whose life span coincides with the system's life span.
- Def. 8 A **transient fault** is a potential source of generated errors which has a life span significantly shorter than the system's recovery requirements.
- Def. 9 A **fail-stop process** is a process which reacts to any failure by an execution halt.



Definitions

[Pradhan]

- Def. 10 **MTTF** mean time to failure the expected time the system will operate before the first failure occurs (average time from N measurements).
- Def. 11 **MTTR** mean time to repair average time required to repair a system
- Def. 12 **MTBF** mean time between failure average time between failures of a system
 $MTBF = MTTF + MTTR$



1.3 Objectives of Fault Tolerance

[Johnson]

- **Dependability**
"quality of service" provided by a particular system, comprising: reliability, availability, safety, maintainability, performability, testability.
- **Reliability** (time interval)
 $R(t)$ conditional probability that system performs correctly throughout time interval $[t_0, t]$ given that it was performing correctly at time t_0 .



Objectives of Fault Tolerance

[Johnson]

- **Availability** (time point)
 $A(t)$ probability that a system is operating correctly and is available to perform its functions at the instant of time t .
- Availability can be high, even if the system has frequent periods of inoperability.
- The availability of a system depends not only on how frequently it becomes inoperable, but also how quickly it can be repaired.





Objectives of Fault Tolerance [Johnson]

- **Safety**

S(t) probability that a system will either perform its functions correctly or will discontinue its function in a manner that does not disrupt the operation of other systems or compromise the safety.

A system should be designed in a way that if it is not working correctly, it will fail in a safe manner.



Objectives of Fault Tolerance [Johnson]

- **Performability**

P(L,t) probability that the system performance will be \geq level L at time t.

Graceful Degradation: ability of a system to automatically decrease the level of performance to compensate for hardware and software faults.

e.g. multiprocessor: run with 1 PE less

e.g. fault in floating-point unit: switch to software emulation



Objectives of Fault Tolerance [Johnson]

- **Maintainability**

M(t) probability that a failed system will be restored to an operational state within period of time t.

A measure of the ease with which a system can be repaired, once it has failed.

- **Testability** Measure for the ease with which certain attributes

within a system can be tested.

e.g. automated testing routines



1.4 Categories of Fault Tolerance [Johnson]

Long-life applications

- e.g. space, satellites
- typical requirement: Availability (10 years) \geq 0.95
- outages in between are allowed.

Critical-computation applications

- e.g. critical to human safety: aircraft control system
- typical requirement: Reliability (3 years) \geq 0.9₇ (short for 0.9999999).



•
•

Categories of Fault Tolerance

[Johnson]

Maintenance-postponement applications

- when maintenance operations are extremely costly
- e.g. space systems and remote processing systems, like telephone switching systems (e.g. maintenance only once a month)

High availability applications

- e.g. banking, flight reservation

• • • • • • • •