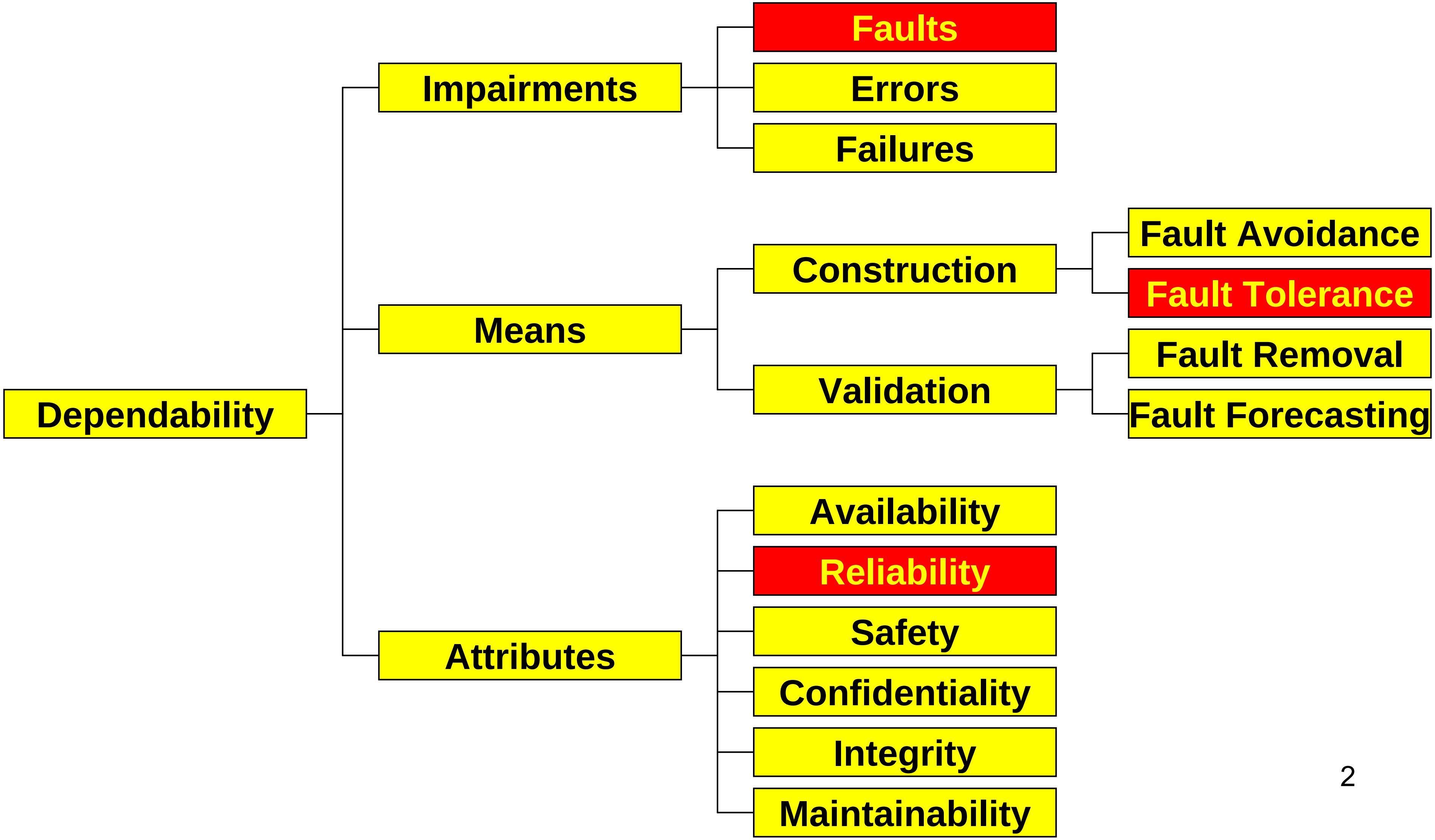
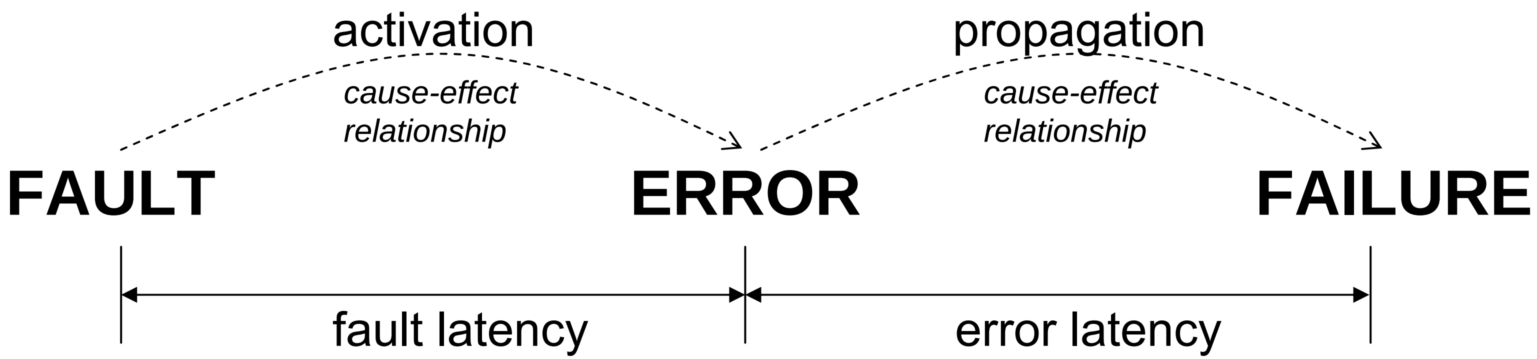
Fault, Failure & Reliability

Lee, Kyoungwoo

Dependability Concept Classification



Relationship b/w Fault/Error/Failure



FAULT

- **a physical defect that occurs within hw or sw components**
- **defect for hw tech**
- **bug for sw tech**

Physical Universe

✓ **where a fault occurs**
✓ **physical entities making up a system**

ERROR

- **a deviation from accuracy or correctness**
- **the manifestation of a fault**

Informational Universe

✓ **where an error occurs**
✓ **units of information (eg: data words)**

FAILURE

- **the nonperformance of some action that is due or expected**
- **malfunction**

External Universe

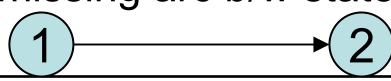
✓ **where a failure occurs**
✓ **the user of a system ultimately see the effects**

Fault

- Definition
  - A defect at the HW or SW component
    - Defect for HW technology
    - Bug for SW technology
  - A hypothesized cause of failure
- Classification

1. NATURE
   1. Type
      1. HW fault: defect within a HW component (eg: ***NOR*** gate instead of ***NAND*** gate)
      2. SW fault: bug within a SW module (eg: ***if*** $A < B$ instead of ***if*** $A \leq B$)
   2. Duration
      1. Permanent fault (static/hard faults) : remains active for a significant period of time (eg: damaged or incorrectly implemented component)
      2. Temporary fault (dynamic/soft faults)
         1. Transient fault: appears for a very short period of time and disappears (eg: soft error)
         2. Intermittent (periodic) fault: appears, disappears, and reappears (eg: a parasitic signal emitted by a part of an electronic system disturbs another part during the operation)
2. ORIGIN
   1. Where
      1. Internal fault: origin of the fault is product itself (eg: incorrectly designed component)
      2. External fault: the fault results from user or environments (eg: operator mistakes or soft error)
   2. When
      1. Creation: origin of the cause to faults is during specification, design, and production
      2. Operation: faults occur at operation

Faults (example)

<table>
<tr><td></td><td colspan="2">HW fault</td><td colspan="2">SW fault</td></tr>
<tr><td rowspan="3">Creation
➔Potentially Permanent, internal</td><td>Incorrect specification fault (eg: incorrect architecture)</td><td rowspan="3">HW Redundancy</td><td rowspan="3">SW Redundancy (N-Version)</td><td>Specification fault (eg: incorrect algorithm)</td></tr>
<tr><td>Poor design fault (eg: missing arc b/w states)
</td><td>Design fault (eg: program bug)
If A < B then S = S+1;</td></tr>
<tr><td>Production fault (eg: stuck at ‘1’ or ‘0’ / short circuit)</td><td>Programming fault (eg: program bug / coding mistake)</td></tr>
<tr><td rowspan="2">Operation</td><td>Soft error
➔external, transient</td><td>HW/Data/Temporal/SW Redundancy</td><td rowspan="2">HW/Data/Temporal/SW Redundancy</td><td rowspan="2">Operator mistakes
➔external, transient</td></tr>
<tr><td>Wear-out corrosion
➔permanent, internal</td><td>HW Redundancy</td></tr>
</table>

Faults at HW layer

- HW faults at each component (device or system)
- Example
  - Incorrect specification
  - Poor design
  - Implementation mistakes
  - Random device defects
  - Component wear-out
  - EM (Electro Migration), TDDB (Time Dependant Dielectric Breakdown), TC (Thermal Cycling)
  - Soft errors

Permanent faults ➔fault-tolerant by HW Redundancy (RM)

Transient faults ➔fault-tolerant by HW (RM), by Data (ECC), by Temporal (CP), or by SW (N-programming) Redundancy

Faults at OS layer

- Software faults
- Example
  - Incorrect design
  - OS bugs
  - Design faults/implementation mistakes
  - Unexpected operation

Permanent faults unless debugged OS is substituted

Temporary faults ➔tolerated by debugging, updating, or rebooting

Faults at Application layer

- Software faults
- Example
  - Incomplete specification
  - Incorrect algorithm
  - Design mistakes
  - Programming bugs
  - Coding mistakes
  - Wrong install
  - User mistakes

Permanent faults unless debugged Or tolerated by SW Redundancy (Incomplete specification through Coding mistakes)

Temporary faults ➔ tolerated by updating, or rebooting (Wrong install, User mistakes)

Dependability Evaluation Techniques

- Several approaches to quantitative evaluation
  - Failure Rate
  - Reliability and Unreliability
  - Availability/Maintainability/Performability/Safety/Analysis
  - MTBF (Mean Time Between Failures)
  - MTTF (Mean Time To Failure)
  - MTTR (Mean Time To Repair)

Failure Rate

- Definition
  - The expected number of failures of a type of device or system per a given time period
  - The speed at which components are likely to fail
- Notation
  - λ

Reliability

- Definition
  - R(t) : The reliability of a component or system
  - The conditional probability that the component operates correctly throughout the interval (t_0,t), given that it was operating correctly at the time, t_0
    - The time interval varies according to applications
    - (eg) Many space applications (repair is impossible)
      - The time intervals being considered can be extremely long, perhaps as many as ten years
    - (eg) Aircraft flight control
      - No more than several hours
      - The reliability throughout the interval may be 0.9_7 or higher
  - $R(t) = N_o(t)/N = N_o(t)/\{N_o(t)+N_f(t)\}$
    - N identical components into operation at t_0
    - $N_f(t)$: the number of failed components at t
    - $N_o(t)$: the number of working components at t
    - Assumption: once a component fails, it remains failed indefinitely

Unreliability

- Definition
  - The probability that a component has not survived the time interval $[t_0,t]$
  - $Q(t) = N_f(t)/N = N_f(t)/\{N_o(t)+N_f(t)\}$
- $R(t) = 1 – Q(t)$ at any time t

Failure Rate Function

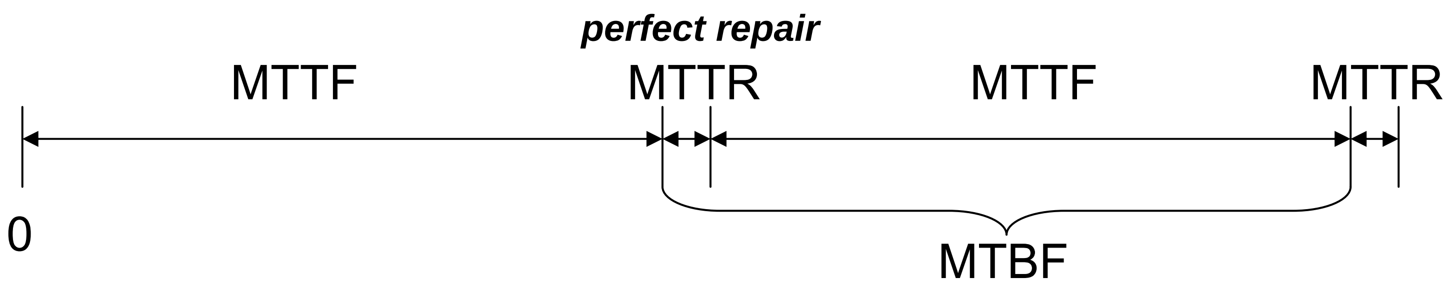
- z(t) : hazard function, hazard rate, or failure rate function
  - $z(t) = 1/N_o(t)*dN_f(t)/dt$
    - $dN_f(t)/dt$ is the instantaneous rate at which components are failing
    - The units are failures per unit of time
- Reliability
  - $dR(t)/dt = -z(t)*R(t)$ from $z(t) = 1/N_o(t)*dN_f(t)/dt$
    - $z(t) = 1/N_o(t)*dN_f(t)/dt = 1/N_o(t)*(-N)*dR(t)/dt = \{dR(t)/dt\}/(-N_o(t)/N\} = \{dR(t)/dt\}/(-R(t))$
      - $R(t) = 1.0 - N_f(t)/N$
      - $dR(t)/dt = -(1/N)*dN_f(t)/dt \Leftrightarrow dN_f(t)/dt = (-N)*dR(t)/dt$
    - $z(t) = \{-1/R(t)\}*\ dR(t)/dt$
  - $R(t) = e^{-\int z(t)dt}$
  - $R(t) = e^{-\lambda t}$
    - (Assumption) the failure rate function has a constant value of λ
    - Exponential Failure Law
      - The exponential relationship b/w the reliability and time
      - The reliability varies exponentially as a function of time for a constant failure rate function

MTTF

- MTTF (Mean Time To Failure)
  - The expected time that a system will operate before the first failure occurs
  - MTTF = $\Sigma^{N}_{i=1} t_i / N$
    - N identical systems
    - t_i : each system, i, operates for a time, t_i, before encountering the first failure
  - MTTF = $\int_{-\infty}^{\infty} t f(t) dt$
    - The expected value of the time of failure
    - f(t) is the failure density function
      - $f(t) = dQ(t)/dt$
    - The integral runs from 0 to ∞
    - MTTF = $\int_0^{\infty} R(t)\, dt$
      - $\text{MTTF} = \int_0^{\infty} t f(t) dt = \int_0^{\infty} t dQ(t)/dt\, dt = -\int_0^{\infty} t dR(t)/dt\, dt$
        $= [-tR(t) + \int R(t)\, dt]_0^{\infty} = [-\infty * R(\infty) + 0 * R(0)] + \int_0^{\infty} R(t) dt$
        $= \int_0^{\infty} R(t)\, dt$

MTBF/MTTR

- MTBF (Mean Time Between Failure)
  - The average time between failures of a system
  - MTBF = T/n_{avg}
    - $n_{avg} = \Sigma^{N}_{i=1} n_i / N$
    - Each of the N systems is operated for some time T
    - n_i is the number of failures for T
    - n_{avg} is the average number of failures
    - The total operation time, T, divided by the average number of failures experienced during the time T
- MTTR (Mean Time To Repair)
  - The average time to repair the system and place it back into operation
  - MTTR = $\Sigma^{N}_{i=1} t_i / N$
    - The i^{th} of N faults requires a time, t_i, to repair
- MTBF = MTTF + MTTR
  - (Assumption) All repairs to a system make the system perfect once again, just as it was when it was new



References

- Dhiraj K. Pradhan, “Fault-Tolerant Computer System Design”, Prentice Hall, 1996, ISBN 0-13-057887-8
- Jean-Claude Geffroy and Gilles Motet, “Design of Dependable Computing Systems”, Kluwer Academic Publishers, 2002, ISBN 1-4020-0437-0
- FOLDOC, “http://foldoc.doc.ic.ac.uk/foldoc/”
- WIKIPEDIA, “http://en.wikipedia.org/wiki/Main_Page”