Terminology Issues in Dependable Computing

Algirdas Avizienis
Distinguished Professor Emeritus
University of California, Los Angeles
and
Vytautas Magnus University, Kaunas, Lithuania
Prologue: my fortuitous encounter with the dependability problem

• 1960: I complete the Ph.D. on computer arithmetic at the University of Illinois and move to the Jet Propulsion Laboratory that Caltech operates for NASA
• JPL is assigned the mission to explore the planets of our solar system by the means of unmanned interplanetary spacecraft
• I am asked to investigate the design of an on-board computer that can survive during a journey of several years and then deliver a specified performance at planetary encounters
• No such unique requirement had existed anywhere in the world until the JPL mission was established by NASA
Prologue: continued

- **1967**: The paper “Design of fault-tolerant computers” by A. Avizienis at the Fall Joint Computer Conference introduces the concept of fault tolerance and describes the JPL-STAR (Self-Testing-And-Repairing) computer design.
- **1970**: Lab model of JPL-STAR is demonstrated, a U.S. patent is granted, and the STAR design is chosen for the 15-year “Grand Tour” mission to four planets.
- IEEE Computer Society Technical Committee on Fault-Tolerant Computing (TC-FTC) is founded, I serve as first Chair.
Prologue: concluded

- **1971**: First IEEE International Symposium on Fault-Tolerant Computing (FTCS-1) takes place in Pasadena, CA, USA, with JPL support. (The 42\textsuperscript{nd}, now “DSN” is in Boston this June)
- *Bad news*: NASA budget is affected by the war in Vietnam and the Grand Tour mission is cancelled, JPL-STAR is an orphan.
- **1972**: *Good news*: the NSF awards a five-year grant to transfer fault tolerance research to UCLA, where the “Dependable Computing and FT Systems Laboratory” continues work until 1994. About 10 faculty, 20 foreign scholars, and 50 graduate students take part in its research.
Our Field’s Goal: deliver expected service under adverse conditions

Our Field’s Top Concepts:

- dependability
- robustness
- high confidence
- survivability
- trustworthiness
- high assurance
- self-healing
- resilience
- fault management

How are they related?
The Concepts of Dependability: a Quest for Structure and Clarity

- **1981**: First meeting of IFIP Working Group 10.4 in Portland, Maine, USA, includes a workshop on the concepts and terminology. A. Avizienis is the founding Chair of the WG.
- **1982**: FTCS-12 in Santa Monica, CA, USA, has a session on the concepts of dependability.
- **1992**: Joint work by members of WG 10.4 appears in the book “Dependability: Basic Concepts and Terminology”, J.-C. Laprie, A. Avizienis and H. Kopetz, editors (850 citations in Google Scholar)
- **2001**: Report “Fundamental Concepts of Dependability” by A. Avizienis, J.-C. Laprie and B. Randell (600 citations in Google Scholar)
2004: The “Taxonomy” Milestone

“Basic concepts and taxonomy of dependable and secure computing” by Algirdas Avižienis, Jean-Claude Laprie, Brian Randell and Carl Landwehr appears in:


This paper summarizes and extends the long-term efforts of the authors and of their colleagues in IFIP WG 10.4 and IEEE CS TC-FTC, as presented next.

Currently Google Scholar lists nearly 2000 citations.

The Basic Concepts

- **Service** delivered by a system (the **provider**): its behavior as it is perceived by its user(s)
- **User**: another system that receives service from the provider
- **Function** of a system: what the system is intended to do
- **Specification** (functional): description of the system function
- **Correct service**: when the delivered service implements the system function
The Basic Concepts (cont.)

- **Service failure**: event that occurs when the delivered service deviates from correct service, either because the system does not comply with the specification, or because the specification did not adequately describe its function.

- **Failure modes**: the ways in which a system can fail, ranked according to failure severities.

- **Error**: part of system state that may cause a subsequent service failure; errors are *latent* or *detected*.

- **Fault**: known or hypothesized cause of an error; faults are *dormant (vulnerabilities)* or *active*. 
Two Definitions of Dependability

**Dependability**: ability to deliver service that can justifiably be trusted (*qualitative*)

**Dependability**: ability to avoid service failures that are more frequent or more severe than is acceptable (*quantitative*)

When service failures are more frequent or more severe than acceptable, we have a **dependability failure**
A Taxonomy of D&S

Attributes
- Availability
- Reliability
- Safety
- Confidentiality
- Integrity
- Maintainability

Means
- Fault Prevention
- Fault Tolerance
- Fault Removal
- Fault Forecasting

Threats
- Faults
- Errors
- Failures

Dependability and Security
Dependability

- Readiness for usage
- Continuity of service
- Absence of catastrophic consequences on the user(s) and the environment
- Absence of unauthorized disclosure of information
- Absence of improper system alterations
- Ability to undergo repairs and evolutions

Security

- Availability
- Reliability
- Safety
- Confidentiality
- Integrity
- Maintainability

Authorized actions

Absence of unauthorized access to, or handling of, system state
The Elementary Fault Classes

Faults

- Phase of creation or occurrence
  - Operational faults
    - occur during service delivery of the use phase
  - Internal faults
    - originate inside the system boundary
  - External faults
    - originate outside the system boundary and propagate errors into the system by interaction or interference
- System boundaries
  - Natural faults
    - caused by natural phenomena without human participation
  - Human-Made faults
    - result from human actions
    - Hardware faults
      - originate in, or affect, hardware
    - Software faults
      - affect software, i.e., programs or data
      - Malicious faults
        - introduced by a human with the malicious objective of causing harm to the system
      - Non-Malicious faults
        - introduced without a malicious objective
        - Deliberate faults
          - result of a harmful decision
        - Non-Deliberate faults
          - introduced without awareness
      - Accidental faults
        - introduced inadvertently
      - Incompetence faults
        - result from lack of professional competence by the authorized human(s), or from inadequacy of the development organization
      - Permanent faults
        - presence is assumed to be continuous in time
      - Transient faults
        - presence is bounded in time
- Phenomenological cause
  - Dimension
  - Objective
  - Intent
  - Capability
  - Persistence
Fault Classification

Development Faults
- Development Faults
  - Operational Faults
- Internal Faults
  - External Faults
- Natural Faults
  - Human-Made Faults
- Hardware Faults
  - Software Faults
- Non-Malicious Faults
  - Malicious Faults
- Non-Deliberate Faults
  - Deliberate Faults
- Accidental Faults
  - Incompetence Faults
- Permanent Faults
  - Transient Faults

Examples:
- Software Flaws
- Logic Bombs
- Hardware Errata
- Production Defects
- Physical Deterioration
- Physical Interference
- Intrusion Attempts
- Viruses & Worms
- Input Mistakes

Physical Faults

Interaction Faults
Human-made Faults

Objective

Non-malicious

Non-deliberate (Mistake)

Non-deliberate

Deliberate (Bad decision)

Deliberate

Malicious

Intent

Accidental

Incompetence

Accidental

Incompetence

Intrusion attempts

Capability

Individuals & organizations

Interaction (operators, maintainers) & Development (designers)

Malicious logic faults: logic bombs, Trojan horses, trapdoors, viruses, worms, zombies

Decision by independent professional judgement by board of enquiry or legal proceedings in court of law
Service Failure Modes

Failures

- Domain
  - Content failures
    - Early timing failures
  - Late timing failures
  - Halt failures
  - Erratic failures

- Detectability
  - Signaled failures
  - Unsigned failures

- Consistency
  - Consistent failures
  - Inconsistent failures

- Consequences
  - Minor failures
    - ●
    - ●
  - Catastrophic failures
Faults, Errors and Service Failures

The fault causes an error – it is the part of the total system state that may lead to a service failure.

The error can be propagated inside the system – that is, it causes more errors during computation.

When an error reaches the service interface, it causes a service failure – it is a transition from correct to incorrect service. The service failure is an event that initiates a service outage.

The return to correct service is a service restoration.
Recommendations for the Handbook

Introduce the concept **error**: part of the system state that was caused by a fault and may lead to a (service) failure.

Define (service) **failure** as an event: “the transition of delivered service (at the service interface) from correct to incorrect service.

Eliminate the concept “failure tolerance”.

Introduce the concepts of **service outage** and **service restoration**.

Define **fault tolerance** as “the means to avoid service failures in or after the presence of faults”.
Development failures

Development process terminates before the system is accepted for use and placed into service

Incomplete or faulty specifications
Excessive number of specification changes
Inadequate design wrt functionality or performance
Too many development faults
Insufficient predicted dependability
Faulty estimates of development costs

Partial development failures

- Budget or schedule overruns
- Downgrading to less functionality, performance, dependability
The Varieties of Maintenance

**Repairs** of the system:
- **Corrective M**: removal of reported faults
- **Preventive M**: discovery & removal of dormant faults

**Modifications** of the system:
- **Adaptive M**: adjustment to environment changes
- **Augmentive M**: augmentation of system function
The Means of Achieving Dependability and Security

- **fault tolerance**: means to avoid service failures in the presence of faults;
- **fault prevention**: means to prevent the occurrence or introduction of faults;
- **fault removal**: means to reduce the number and severity of faults;
- **fault forecasting**: means to estimate the present number, the future incidence, and the likely consequences of faults.
Groupings of the Means for Dependability and Security

Means for Dependability and Security
- Fault Prevention
- Fault Tolerance
- Fault Removal
- Fault Forecasting

<table>
<thead>
<tr>
<th>Dependability and Security Provision</th>
<th>Dependability and Security Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault Avoidance</td>
<td>Fault Acceptance</td>
</tr>
</tbody>
</table>

A. Avizienis
Fault Tolerance Techniques

Fault Tolerance
  - Error Detection
    - Concurrent Detection
      - Preemptive Detection
  - Recovery
    - Error Handling
      - Rollback
      - Rollforward
      - Compensation
    - Fault Handling
      - Diagnosis
      - Isolation
      - Reconfiguration
      - Reinitialization
Fault Removal

•During Development:
  Verification
  Deterministic testing
  Statistical (random) testing
  Fault injection

•During use:
  Preventive maintenance
  Corrective maintenance
Fault Forecasting

• **Qualitative** (ordinal) evaluation:
  Identify, classify and rank failure modes

• **Quantitative** (probabilistic) evaluation:
  Modeling
  Operational testing
  Benchmarking
Fault Prevention

• **Qualitative** (ordinal) evaluation:
  Identify, classify and rank failure modes

• **Quantitative** (probabilistic) evaluation:
  Modeling
  Operational testing
  Benchmarking
Our Field’s Goal: deliver expected service under adverse conditions

Our Field’s Top Concepts:

- dependability
- high confidence
- robustness
- resilience
- survivability
- high assurance
- trustworthiness
- self-healing
- fault management

How are they related?
# Four Essentially Equivalent Concepts

<table>
<thead>
<tr>
<th>Concept</th>
<th>Dependability</th>
<th>High Confidence</th>
<th>Survivability</th>
<th>Trustworthiness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>1) ability to deliver service that can justifiably be trusted</td>
<td>consequences of the system behavior are well understood and predictable</td>
<td>capability of a system to fulfill its mission in a timely manner</td>
<td>assurance that a system will perform as expected</td>
</tr>
<tr>
<td></td>
<td>2) ability of a system to avoid service failures that are more frequent or</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>more severe than is acceptable</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Threats</td>
<td>present</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1) development faults (e.g., software flaws, hardware errata, malicious</td>
<td>• internal and external threats</td>
<td>1) attacks (e.g., intrusions, probes, denials of service)</td>
<td>1) hostile attacks (from hackers or insiders)</td>
</tr>
<tr>
<td></td>
<td>logic)</td>
<td>• naturally occurring hazards and malicious attacks from a sophisticated and</td>
<td>2) failures (internally generated events due to, e.g., software design errors, hardware degradation, human errors, corrupted data)</td>
<td>2) environmental disruptions (accidental disruptions, either man-made or natural)</td>
</tr>
<tr>
<td></td>
<td>2) physical faults (e.g., production defects, physical deterioration)</td>
<td>well-funded adversary</td>
<td>3) accidents (externally generated events such as natural disasters)</td>
<td>3) human and operator errors (e.g., software flaws, mistakes by human operators)</td>
</tr>
<tr>
<td></td>
<td>3) interaction faults (e.g., physical interference, input mistakes, attacks,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>including viruses, worms, intrusions)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reference</td>
<td>This paper</td>
<td>“Information Technology Frontiers for a New Millennium (Blue Book 2000)” [NSTC</td>
<td>“Survivable network systems” [Ellison et al. 1999]</td>
<td>“Trust in cyberspace” [Schneider 1999]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2000)]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
On the Dependability of Scientific and Technical Texts

The Goal: to treat the content of the texts of documents as a part of the development process of information processing systems
Building a System

Professional Team
methods, experience, experiments, imagination

Customer Team
needs expectations

Requirement specification

System development

System delivery
Prerequisites Of Design
manuscripts, research papers, patents, reports, product manuals, specifications, design and program documentations, handbooks, monographs, textbooks, etc., etc.
TEXT Faults

FORM Faults
- Orthographical
- Morphological
- Syntactical

CONTENT Faults
- Semantic
  - Correctness
  - Consistency
  - Definition
- Originality
  - Unintentional
  - Intentional or Malicious
  - Hoax
- Structural
  - Completeness
  - Redundancy
  - Incrementalism
  - Reference

Detected and removed by editors also, tools exist: CLAT, etc.
Detected and removed by expert evaluators tools are needed – this is our research
Ontology Faults

“Ontology”: A structured representation of the relationships between the concepts of a field (taxonomy with more than one relationship)

An “ontology fault” exists when the relationship of one top concept of a field to the others is not identified in a field that has two or more top concepts
A Spacecraft Relay Chain for Interstellar Missions

1. Launch a low cost DiSTAR spacecraft every N months; the design can evolve continuously
2. Use the chain of spacecraft to relay communications to Earth and back to the leading spacecraft
3. Introduce redundancy at spacecraft level: every spacecraft can dependably communicate to M = 2, 3, or more, closest neighbors; then the loss of M-1 adjacent spacecraft is tolerable
4. Slow down all spacecraft ahead of the gap to repair the chain
5. Never stop launching better and better DiSTAR spacecraft!