Adaptability and Fault Tolerance

Rogério de Lemos
University of Kent, UK

- **Context**: self-* and dependability;
- **Focus**: adaptability and fault tolerance;
- **State of the art**;
- **Conclusions**;
Self-* and Dependability

- **Dependability:**
  - the ability to deliver service that can justifiably be trusted;

- **Self-* properties of systems:**
  - the support for autonomy;
  - self-adaptable, self-managing, self-optimising, self-healing, self-repairing, self-configuring, etc.

- **Adaptability:**
  - the ability of a system of accommodating changes while providing its specified services;
    - run-time changes;
Dependability - the ability to avoid service failures that are more frequent and more severe than is acceptable;

- **threats** - undesired, but in principle expected circumstances:
  - faults, errors and failures;

- **attributes** - properties of the system:
  - reliability, availability, integrity, confidentiality, and safety;

- **technologies** - methods and techniques for providing and reach confidence on ability to attain dependability:
  - rigorous design, validation & verification, fault tolerance, and system evaluation;
**Dependability - Threats**

(Yves Deswarte & David Powell)

- **Fault**: That part of system state which may lead to a failure.
- **Error**: Occurs when delivered service deviates from implementing the system function.
- **Failure**: Adjudged or hypothesized cause of an error.

(process)

1. **Fault** → **Error**
2. **Error** → **Failure**
3. **Failure** → **Fault**
4. **Fault** → **Error**

Rogério de Lemos
Adaptability - Initiators

- Changes:
  - the act, process, or result of altering or modifying;
  - internal changes:
    - component failures, overload of resources, etc.
  - external changes:
    - environmental, requirements, etc.

- There is no fundamental chain of adaptability initiators;
Threats and Initiators

- Changes correspond to events (faults):
  - changes can be dormant if not activated;

- What is the consequence of change (errors)?
  - what would be the equivalent to error free and erroneous states?
  - these states are created when changes are activated and can remain latent until detected;

- What is the equivalent of failure?
  - unsuccessful adaptation?
  - the system might continue to provide its services, but ignoring the change;
Fault avoidance: build a system with no faults:

- rigorous design - fault prevention;
  - formal and rigorous notations, processes, adapters, etc.
- verification & validation - fault removal;
  - model checking, fault injection, testing, simulation, etc.

Fault acceptance: impossible to rid the system of faults:

- fault tolerance;
- system evaluation - fault forecasting;
  - empirical approaches, Markov models, etc.
Fault tolerance aims at avoiding the failure of the system:

- **error detection:**
  - detects the presence of errors;

- **recovery:**
  - transforms a system state that contains errors or faults into a error free state, or faults that can be re-activated;
    - error handling:
      - eliminates errors from the system state;
    - fault handling:
      - prevents faults from being activated again;
      - diagnosis, isolation and reconfiguration;
Fault Tolerance
(Yves Deswarte & David Powell)

Fault

Error

Error Detection

that part of system state which may lead to a failure

occurs when delivered service deviates from implementing the system function

Failure

Error Handling

Fault Handling

Diagnosis, Isolation, Reconfiguration, Reinitialization

Rollback, Rollforward, Compensation

adjudged or hypothesized cause of an error
Fault tolerance is about system structuring;

- structure is what enables the system to generate the behaviour;
- determines how effectively this structuring can be used to provide means of **error confinement**;
  - avoid the propagation of errors;
  - what interactions can exist and at what rate;
- it is not restricted to system architecture;

**Structural flexibility the basis for adaptation;**
Fault Assumptions

Faults are undesirable, though expected circumstances:

- systems can fail in many different ways;

In the design of fault-tolerant systems, it is essential to define assumptions:

- **nature** of faults - dictates the type of redundancy that must be implemented:
  - space or time;
  - replication or diversification;

- **rate** of faults - influences the amount of redundancy needed to attain a given dependability;
Fault Assumptions

How a component behaves when it fails:

- crash fault being the simplest and most restrictive (or well-defined) type;
- Byzantine being the least restrictive;

The different types of changes need to be classified:

- behavioural assumptions;
Adaptive fault tolerance

- property that enables a system to maintain and improve fault tolerance by adapting to changes in environment and policy;
  - monitor the system;
  - reconfigure the application when its configuration of it is not appropriate for the dependability requirements;
- distributed systems:
  - different layers:
    - middleware / fault tolerance /adaptation;
  - consensus problem;
State of the Art

- **AQuA** - CORBA based operating system;
  - dynamic replication of objects;
- **Proteus**:
  - dynamic fault tolerance through adaptive reconfiguration;
  - allows to specify the degree of dependability at the application level;
State of the Art

- **Chameleon** - adaptive infrastructure;
  - allows different levels of availability requirements;
  - explicit representation of adaptive policies;
  - provides dependability through the use of ARMORs (Adaptive, Reconfigurable, and Mobile Objects for Reliability):
    - managers for monitoring and recovering resources;
    - daemons for providing communication;
    - common ARMORs for providing application-required dependability;
  - enables multiple fault tolerance strategies to co-exist;
Architectural fault tolerance

- Error detection and recovery:
  - techniques based on exception-handling;
  - application dependent;
  - iC2C and iFTE;

- Fault handling
  - system reconfiguration;
  - replacement of components, connectors and configurations;
  - dynamic reconfiguration;
Bio-inspired computing and statistical methods:

- data-oriented approaches
  - data mining large quantities of observations for identifying patterns;
- anomaly (fault and intrusion) detection;
  - neural networks, genetic algorithms, etc.;
  - adaptive error detection using artificial immune systems:
    - problem: how to learn from rare events!
- statistical learning techniques (SLT) applied to system recovery;
Conclusions

- Changes are like faults, though:
  - they might be desired/undesired and expected/unexpected;
- Classification of the types of changes:
  - otherwise becomes application dependent;
    - e.g., exception handling for the support of fault tolerance;
- How system structuring affects adaptability?
  - is software that flexible for supporting run-time change?
    - impact of design-time change;
  - to scope the impact of change;
    - confinement of the consequence of change;
Conclusions

Dependability and adaptability:

- the ability to deliver service:
  - D: rigorous design and fault tolerance;
  - A: rigour in the specification/reasoning about adaptability;
  - A: most work has focused on system reconfiguration;

- confidence on that ability:
  - D: V&V and system evaluation;
  - A: very little has been done here;
    - adaptability vs. predictability;