WADS 2003 Panel: Fault Tolerance and Self-Healing

Moderator:
- Rogério de Lemos (University of Kent, UK)

Panellists:
- Valerie Issarny (INRIA, France)
- Philip Koopman (CMU, USA)
Motivation

- What makes *self-healing* different from *fault tolerance*:
  - avoid the tenet that it is a more appealing term;

- **Fault tolerance** - a means to dependability:
  - delivery of specified service, despite the presence of faults;

- **Self-healing** – being used by the software engineering community:
  - there is no consensus based definition;
Dependability Tree

Dependability

Attributes

Availability
Reliability
Safety
Confidentiality
Integrity
Maintainability

Means

Fault Prevention
Fault Tolerance
Fault Removal
Fault Forecasting

Threats

Faults
Errors
Failures

Avizienis, Laprie, and Randell 2001
Motivation

Software engineering:

- development of software systems based on rigorous (sometimes formal) languages and processes, supported by tools;
  - fault prevention:
    - e.g., goal structures, and UML;
  - fault removal (V&V):
    - e.g., model checking, and testing;

- build software systems without bugs;
Motivation

Software engineering and self-healing:

- build software systems that may have bugs;
  - e.g., components interface types do not match;
  - nature e.g., healing of a wound;

- build software systems that adapt to changes in the environment;
  - e.g., adjust performance depending on resources;
  - nature e.g., sweating regulates body temperature;
  - the notion of “healing” is not evident;
    - reacting to changes by adapting;
Opportunities (or not)

Dependability versus Software Engineering

- **Dependability perspective:**
  - *Self-healing is re-inventing the wheel* – all problems and solutions being investigated have already been mapped out;

- **Software engineering perspective:**
  - *Fault tolerance provides expensive solutions* – redundancies are expensive, leading to complex systems:
    - although solutions elegant, they are impractical;

- Is there a middle way between these two perspectives;
Challenges

- Is the software engineering community moving from design time into run time solutions?
  - the same type of problems that the hardware community faced decades ago?

- Are software engineering and dependability communities dealing with the same type of problems?

- Is the dependability “framework” too strict, or inappropriate, to self-healing?
  - why this?
  - what can and should be re-used?
Challenges

◆ At what stages of software development should “self-healing” be employed?
  ◆ fault tolerance has been effective at the later design stages;
  ◆ at the software architectural level?
    ◆ redundancies at the high level lead to waste of resources;

◆ If there are no system faults, what would be the framework that would allow the system to react to changes?
  ◆ what are the undesirable, though not unexpected circumstances?
  ◆ how these should be handled?
Challenges

Last but not least:

- If fault tolerance and self-healing deal with the same type of problems would it be wise to adopt the self-healing since it is a more appealing then the ‘old’ boring fault tolerance?
  - it is a more intuitive term, at least;
Conferences sponsored by IFIP WG10.4 and/or IEEE TC FTC:

- International Conference on Dependable Systems and Networks (DSN);
- IEEE Symposium on Reliable Distributed Systems (SRDS);
- International Symposium on Software Reliability Engineering (ISSRE);
- International Conference of Computer Safety, Reliability and Security (SAFECOMP);
- Regional conferences:
  - European Dependable Computing Conference (EDCC);
  - Pacific Rim Int. Symposium on Dependable Computing (PRDC);
  - Latin American Symposium on Dependable Computing (LADC);