

Dependability

- Challenges, Research -

Prof. Jaco van de Pol
Universiteit Twente



The plan

- 5 questions on *What is Dependability?*
- Projects, examples, research

Q1: Are *our systems* dependable, why not?

- Systems get more *complex, distributed, open*
- Dependable systems should cope with:
 1. *Physical faults*.....hardware trends
 2. *Communication problems*.....wireless
 3. *Software errors*.....5-10/kloc
 4. *Human operator mistakes*.....bored,ignorant
 5. *Malicious attackers*.....inventive,criminal

Q2: What does *dependable* mean, exactly?

- Answer: a system is dependable, if:
 - One can rely on the services that it delivers
 - This comes with some justification (certificate)
- Depending on the application domain, it means
 1. Webshop.....*7x24 availability (99%)*
 2. Deep space mission.....*high reliability (MTTF)*
 3. E-commerce.....*integrity of transactions*
 4. Medical equipment.....*absolutely safe behaviour*
 5. E-voting.....*privacy for voters*
 6. Cloud compute server.....*QoS level agreement*

Q3: How to *build* dependable systems?

- Answer: by using dependability *means*:
 1. Fault detection.....*run-time monitoring, measure*
 2. Fault prevention.....*analysis, proof, design*
 3. Fault prediction.....*statistics, testing*
 4. Fault removal.....*diagnostics, reconfiguration*
 5. Fault tolerance.....*redundancy, architecture*

(‘fault’ can also be read as ‘intrusion’)

Q4: Where is our dependability department?

- Dependability crosses all system layers
- It is addressed by many sub-disciplines:
 1. *Software Engineering*
.....(software architecture, monitoring)
 2. *Formal Methods*
.....(correctness, performance, synthesis)
 3. *Communication Systems*
.....(error correction, radio communication)
 4. *Security*
.....(encryption, trust management)
 5. *Control Theory*
.....(robustness, self-adaptation)

Q5:

What are the research challenges?

- *Scalability of Modelling and Analysis*
 - State space explosion
 - Software for multi-core platforms
 - Performance, energy usage, average analysis
- *Automated synthesis*
 - Model Driven Engineering ... including the contents
 - Strategies from game theory, optimization
- *Factor out dependability as a separate concern*
 - #include <dependability.h>
 - safari -dependability=5
- *Can we learn from robustness in biology?*

3TU. Reliability in Consumer Electronics

Trade-off project (2000-2009)

Embedded Systems Institute, STC 3, NXP, Philips

The problem: Increase Perceived Reliability

Complexity increases, while life cycle decreases

Open systems: multiple manufacturers, networked

Research challenges:

User-centric design for reliability

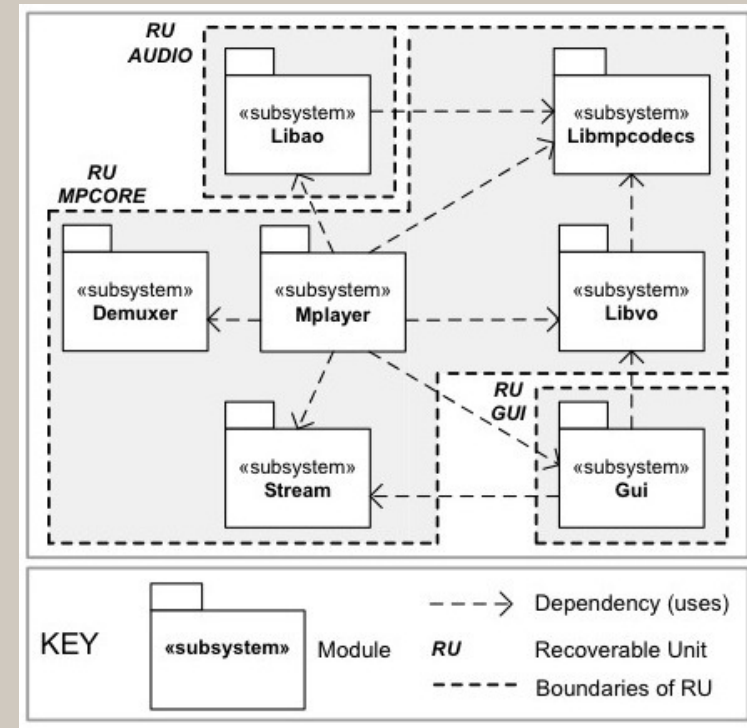
System architecture and construction

Analysis of reliability in consumer electronics

Trader: Technical solution

(thesis Hasan Sözer, U Twente)

- Decomposition into recoverable units
- Automatic decoupling of software
- Architecture for monitoring and restart
- Modelling and Analysis by means of *dynamic fault trees*
- Automatic optimization for most efficient system decomposition
- Compositional analysis methods based on *Interactive Markov Chains*



Media Player

Results:

- Much quicker restart due to *local recovery*
- Software errors become unnoticeable for the end-user

3TU. Railway Interlocking Requirements



Triggers for innovation:

- European standardization
- Better utilization of railways

Barriers for innovation:

- Certification of safety layer
- Decoupling operators-industry

Railway Interlocking Requirements

European FP7 Project: *INESS* (2008-2011)

Flexible solution:

- ERTMS: European Railway TMS
- Track-train communication

Challenges:

- Harmonize safety requirements
- Specify of interlockings (UML)
- *Automatic safety check on design*

Partners:

- Operators: ProRail, DB, ..., UIC
- Industry: Siemens, Thales, Alstom, ...
- CeDICT/LaQuSo: safety analysis



INESS: technical solution

- xUML class diagrams
 - tracks, signals, points, routes
- xUML state diagrams
 - express required behaviour
 - simulator: Cassandra
- Formalise xUML by translation
 - Promela, μ CRL,...
- Exhaustive *model checking*
 - verify basic signalling rules
 - toy: $81 \cdot 10^{12}$ states, 2 min.

