

Playing the Devil's Advocate: Testing Real-Time Systems

Jan Jürjens

Software & Systems Engineering
TU Munich, Germany



juerjens@in.tum.de

<http://www.jurjens.de/jan>



Testing Real-Time Systems

Very challenging.

For high level of assurance, would need **full coverage** (test every possible execution).

Usually **infeasible** (especially reactive systems).

Have **heuristics** for trade-off between development effort and reliability.

Need to ask yourself:

- How **complete** is the heuristic ?
- How can I **validate** it ?

Recent Trends in Academic Research

Model-based Testing (e.g. based on Real-time UML). Advantages:

- **Precise measures** for completeness.
- Can be **formally** validated.

Two complementary strategies:

- Conformance testing
- Testing for criticality requirements

Conformance Testing

Classical approach in model-based test-generation (much literature).

Can be superfluous when using **code-generation** [except to check your code-generator, but only once and for all].

Works independently of real-time requirements.

Conformance Testing: Caveats

- Complete test-coverage **still infeasible** (although can measure coverage).
- Can only test code against what is contained in model. Usually, model more abstract than code. May lead to „**blind spots**“.

For both reasons, may miss critical test-cases. Want: „**criticality testing**“.

Criticality Testing: Strategies

Internal: Ensure test-case selection from models does not miss critical cases: **Select** according to information on **criticality**.

External: Test code against possible **environment interaction** generated from parts of the model (e.g. deployment diagram with information on physical environment).

More info (papers, industrial courses):

<http://www.jurjens.de/jan>

This Track (B)

Three exciting talks on testing real-time systems:

- **Terry Price** (Entivity, UK): **Testing Real Time Systems using Flow Chart Programming with Machine Level Diagnostics**
- **Doron Cherkovsky** (Israel Aircraft Industries) **Design for Quality and Reliability in Real Time Systems**
- **Mike Rennie** (Deimos Space, Spain): **Embedded Real-Time Software Test Benches for Future Space Missions**



Criticality Testing

Shortcoming of classical model-based test-generation (conformance testing) motivates „criticality testing“.

Goal: model-based test-generation adequate for critical real-time systems.

Internal Criticality Testing

Need behavioral semantics of used specification language (precise enough to be understood by a tool).

Here: semantics for simplified fragment of UML in „pseudo-code“ (ASMs).

Select test-cases according to criticality annotations in the class diagrams.

Test-cases: critical selections of intended behavior of the system.

External Criticality Testing

Generate test-sequences representing the environment behaviour from the criticality information in the deployment diagrams.

More info (papers, industrial courses):

<http://www.jurjens.de/jan>