Model Transformation in Practice

BSSE System and Software Engineering

DASIA 2007
May 29th – June 1st 2007
Naples, Italy

Ralf Gerlich[1], Daniel Sigg[2], Rainer Gerlich[3]

[1]<ralf.gerlich@bsse.biz> [2]<daniel.sigg@bsse.biz> [3]<rainer.gerlich@bsse.biz>

Dr. Rainer Gerlich System and Software Engineering

Auf dem Ruhbühl 181
88090 Immenstaad

http://www.bsse.biz/

Outline

- The Model
- UML2 experiment+results
- 3ADL experiment+results
- General Considerations
- Conclusions

Project Reference

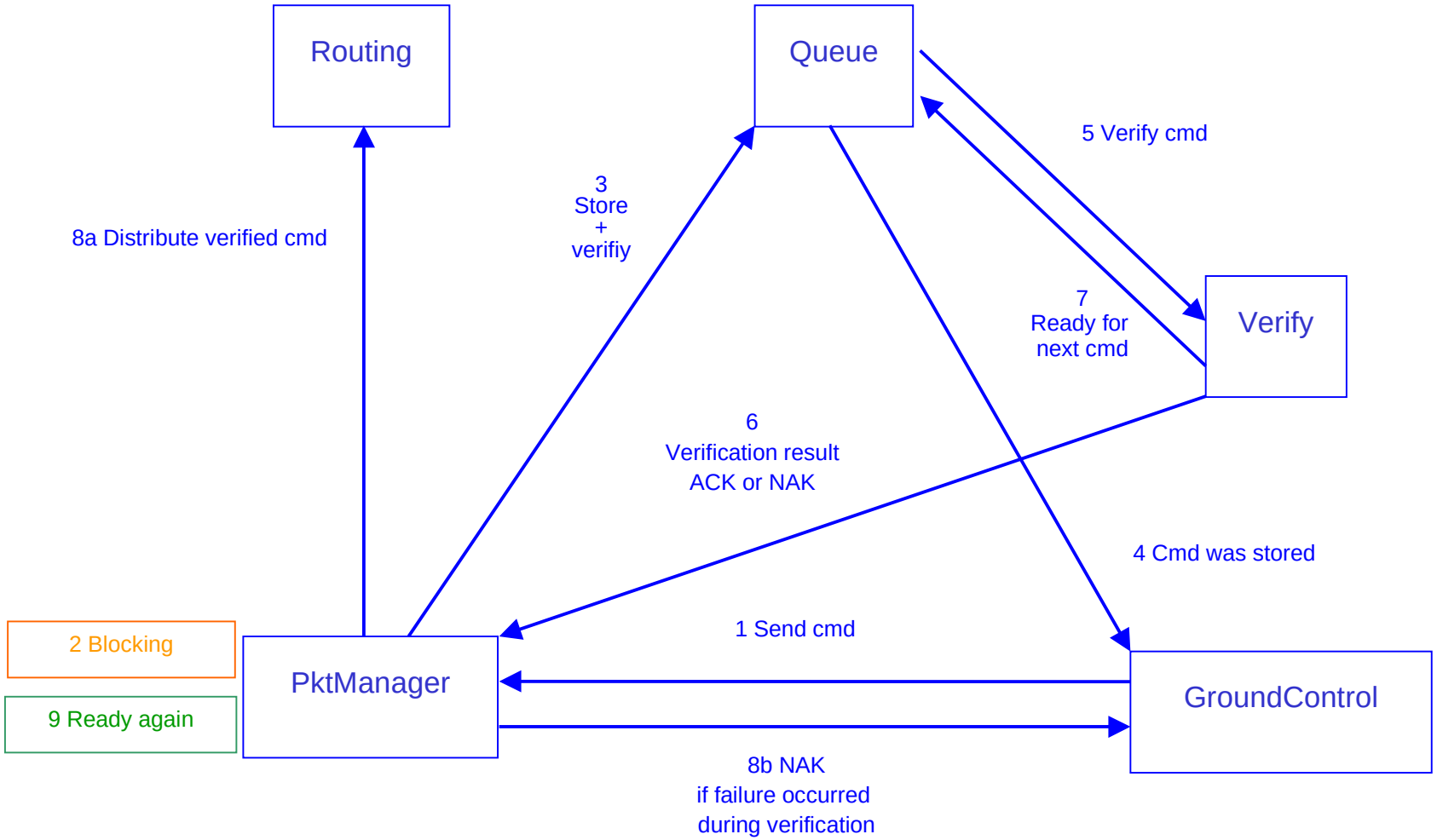
- ACG
  - Automatic Code Generation
  - ESTEC Contract 18056/04/NL/JA
  - Astrium-ST (prime), SynSpace, SciSys

- ASSERT
  - Automated proof-based System and Software Engineering of Real-Time Systems
  - EU project CL 004033, FP6
  - headed by ESA

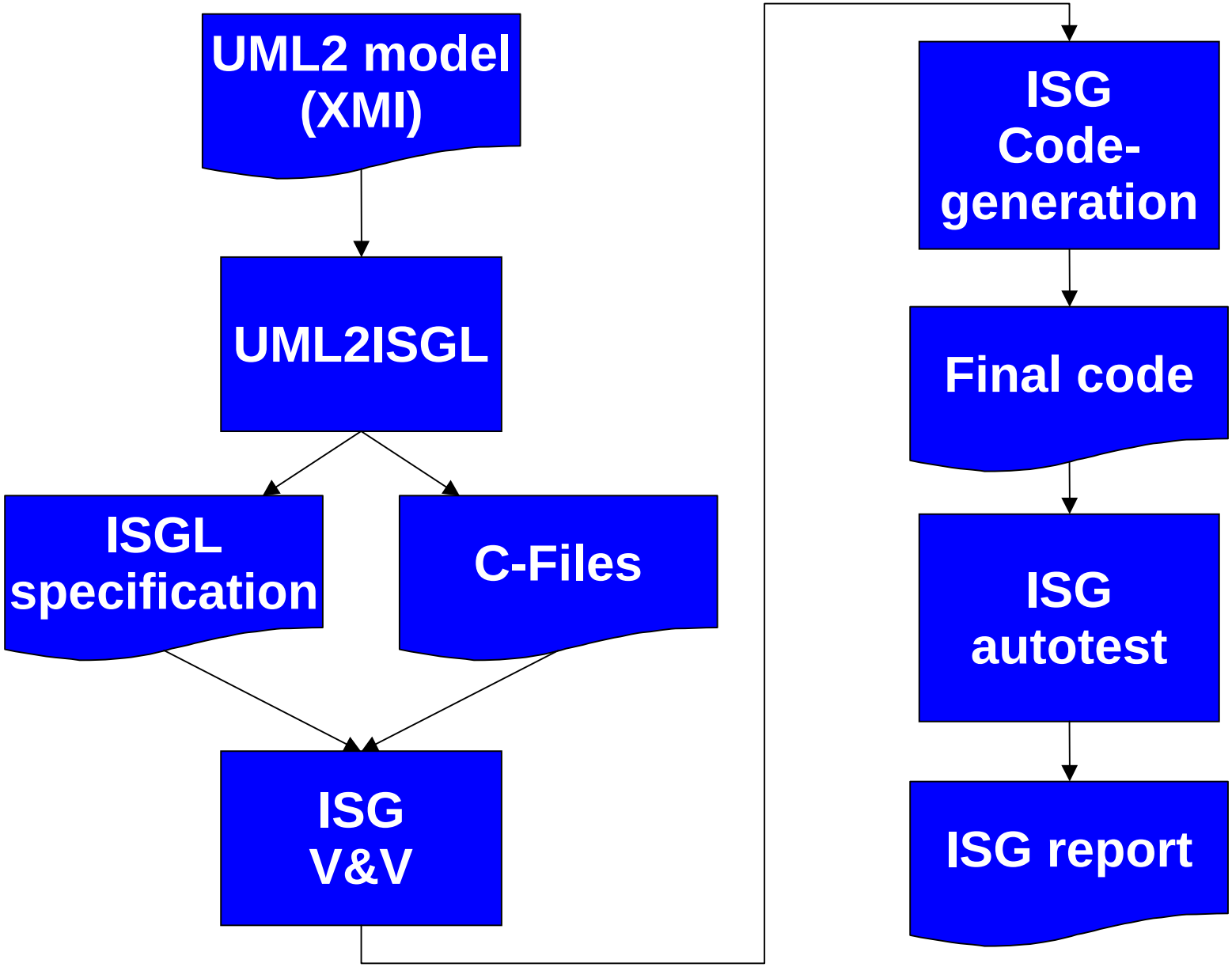
Considered languages

- ACG: UML2
  - pure S/W architecture (via class diagrams)
  - behaviour specified via StateCharts
  - only the subset used in the model
- ASSERT: 3ADL (AADL-based UML-profile)
  - architectural specification (components, subcomponents)
  - behaviour via StateCharts (not part of 3ADL)
- ISGL
  - behaviour via Finite Statemachines
  - architecture+communication via own concepts

TMTC Manager



UML2ISGL Process



ISG Experiments

- ISG automatically stimulates system
  - on target or in simulation environment
  - optional fault injection (loss of signal, invalid signal)
- records observed behaviour
  - state and transition coverage
  - timing diagrams
  - message-sequence diagrams

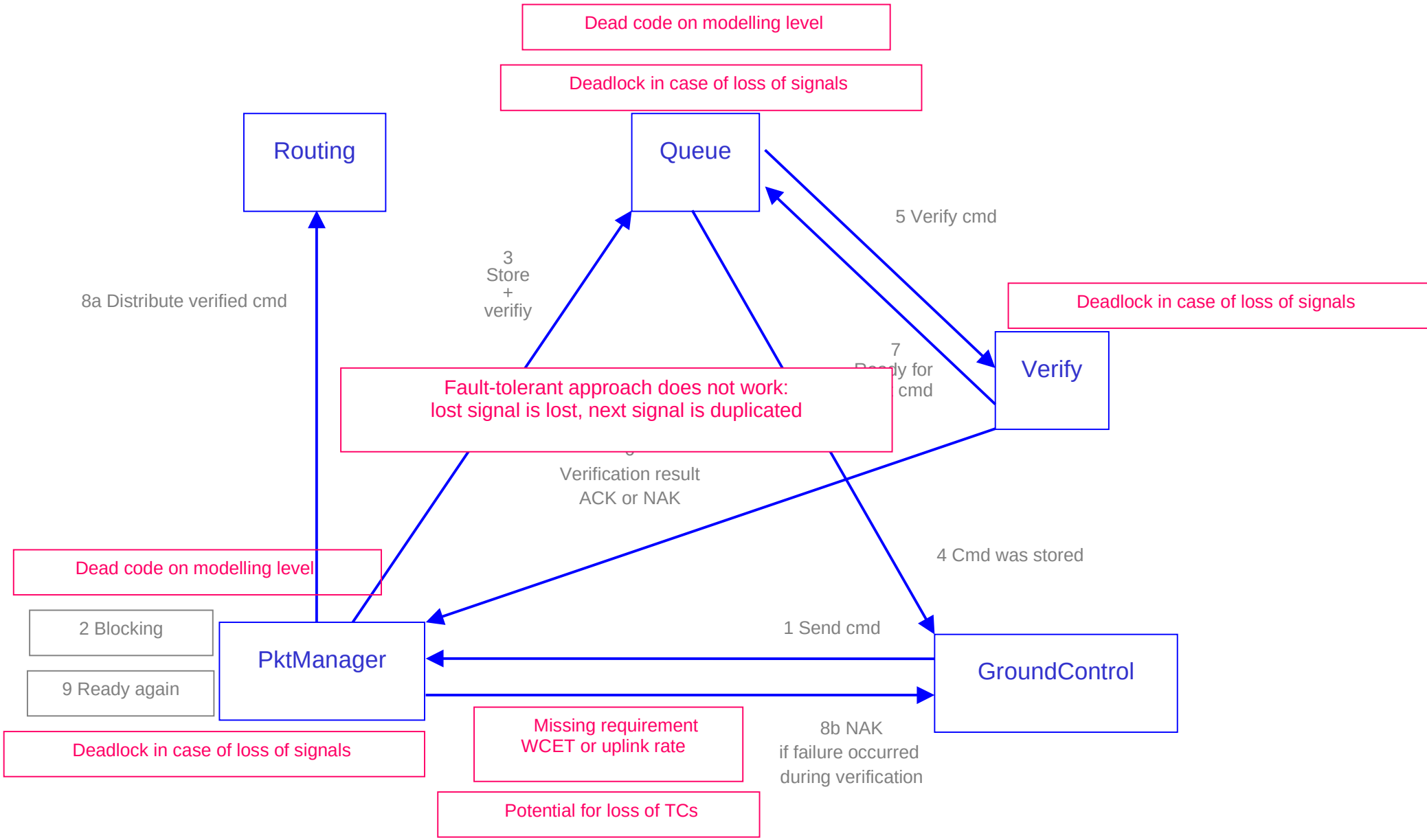
UML2 Transformation

- Original model based on TAU
  - no conforming XMI-export (files invalid, unreadable)
- re-modelling using Eclipse UML2
- transformation to ISGL
  - active classes with StateMachines become ISGL processes
  - Use Cases and MSCs not considered (too informal)
  - direct mapping of communication
  - conditions, opaque actions become C-functions
  - H/W architecture (CPU-mapping, channels) hardcoded (not available from model)

UML2 results

- Model was formally incomplete
  - interaction with ground, but ground not modelled
- Performance bottleneck
  - on reception of TC receiving process blocks
  - until TC was completely processed
  - no requirement on maximum burst rate known
- Missing coverage
  - System was stimulated without failure
  - Missing coverage indicated fault tolerance handling
  - enforced random loss of signals produced deadlock
- Not detected with other tools!

Faults found



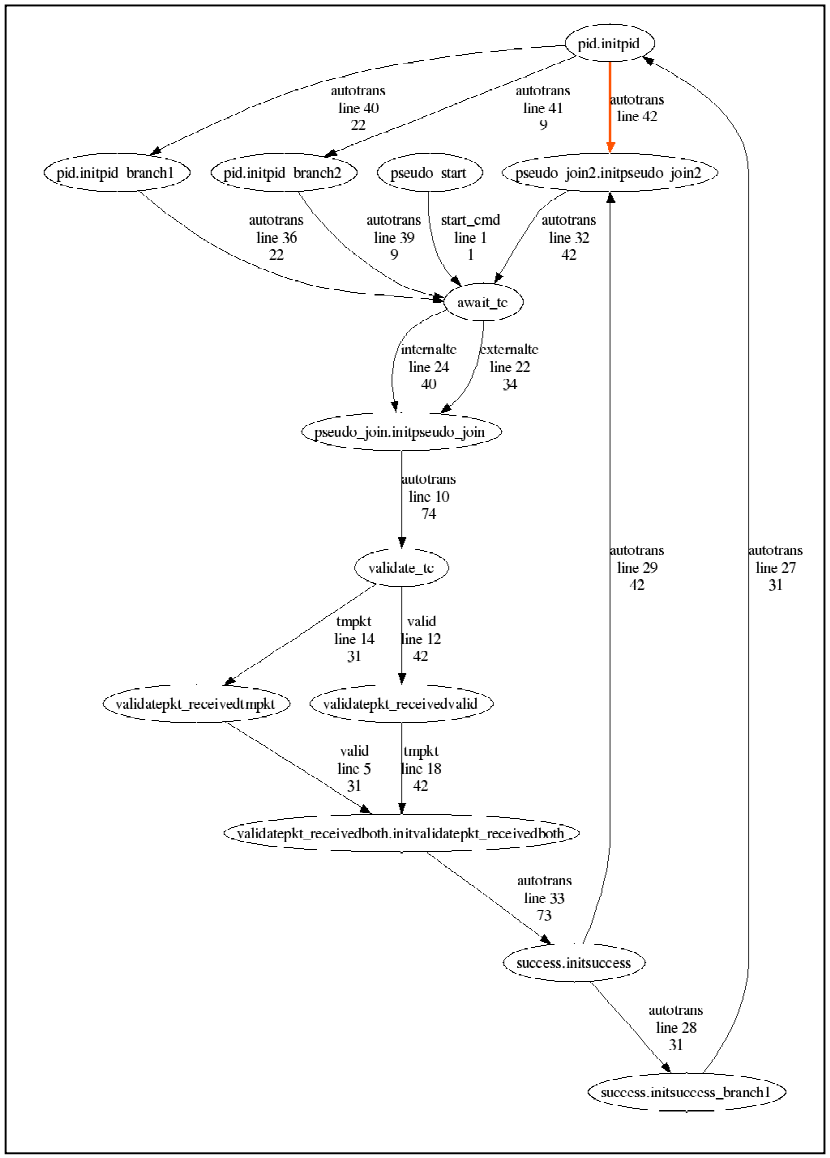
3ADL Experiment

- same TMTC-Manager
  - behavioural specification kept
- new architectural specification
  - AADL hierarchical decomposition, ports
  - based on 3ADL architectural stereotypes
- transformation to ISGL
  - channels, CPUs now specified in model
  - flattening of hierarchical decomposition
  - thread types contain behaviour ? processes
- results as for the UML2 experiment
  - as expected!

Benefits of Model Exchange

- Different tools provide different focus
  - e.g. schedulability vs. behaviour
- Comparison of results for same-focus tools
  - e.g. results from different code generators in test
- Better view on overall system properties
  - combine information on different aspects
  - identify faults not identified by a single tool
- Languages are compared
  - 3ADL: better architectural modelling
  - UML2: more additional capabilities (e.g. behaviour)
- benefits of n-version code generation
  - “truly independent development teams”

Complementary Capabilities: Coverage

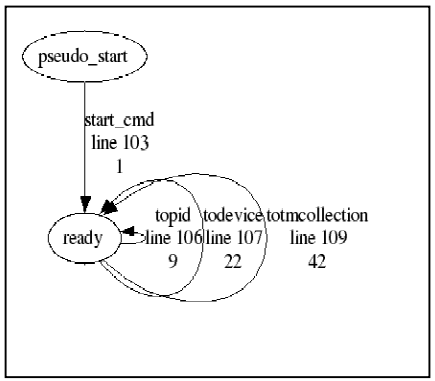


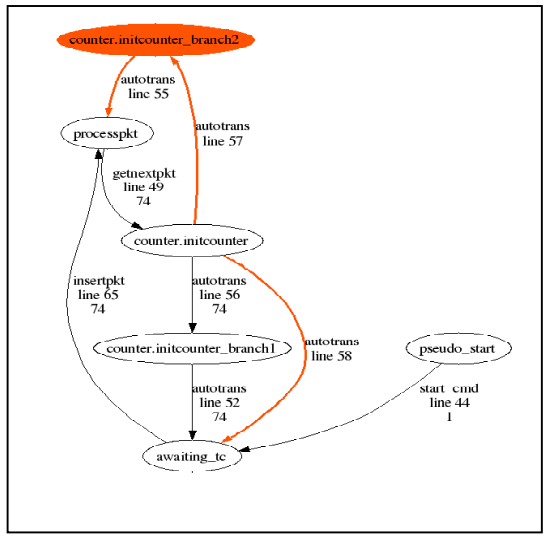
Missing FSM coverage:

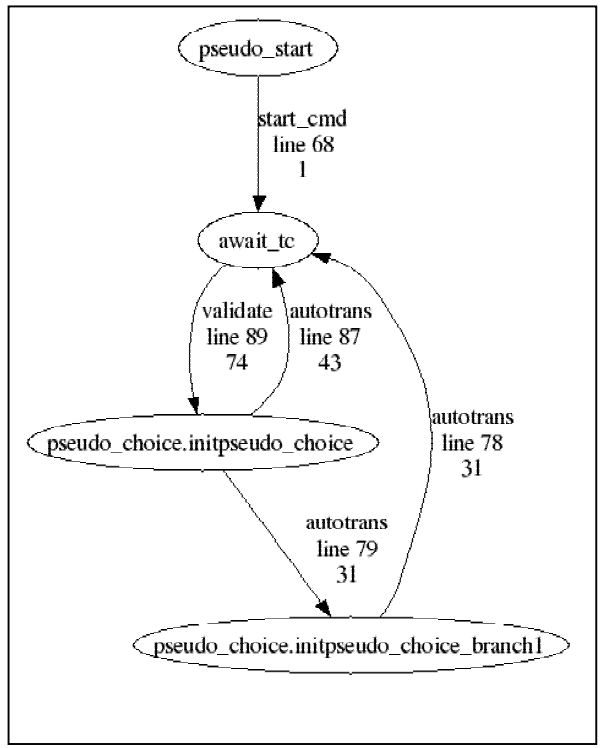
fault-tolerant branch on modelling level

⇓

enforce loss of signals

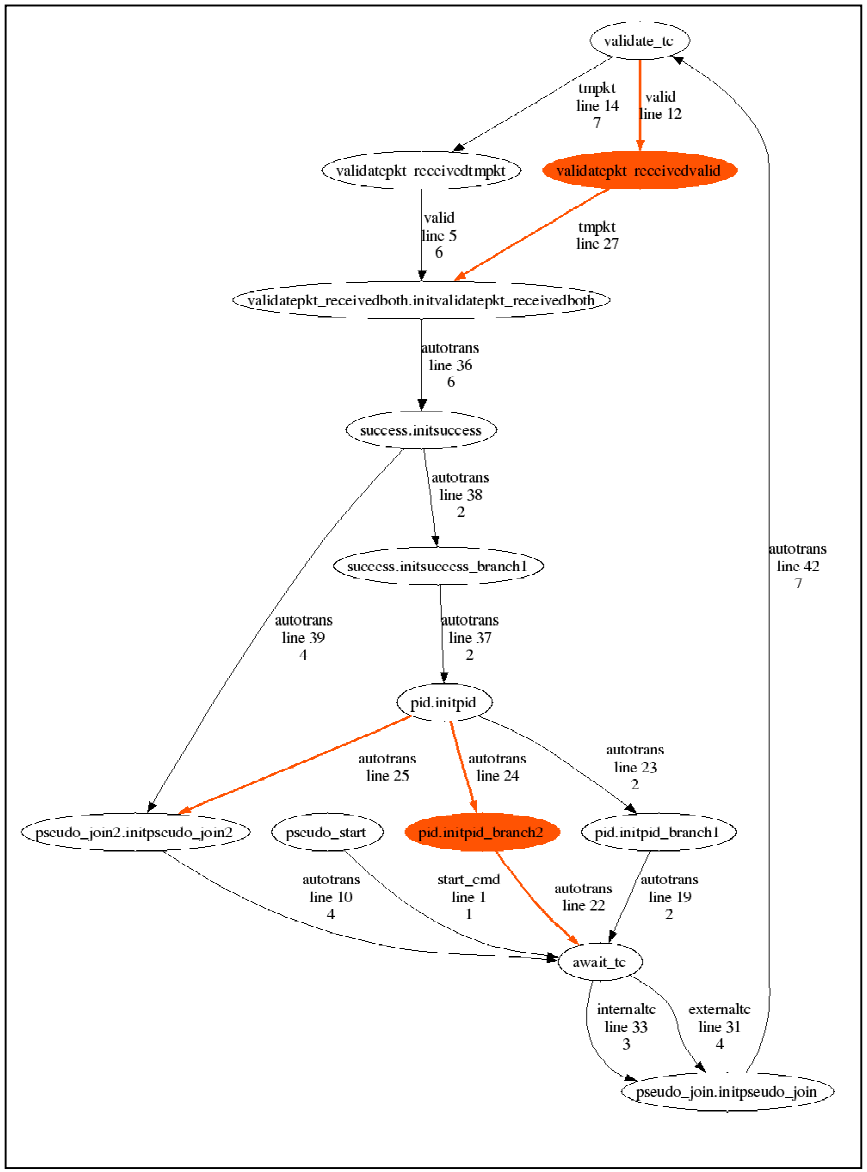






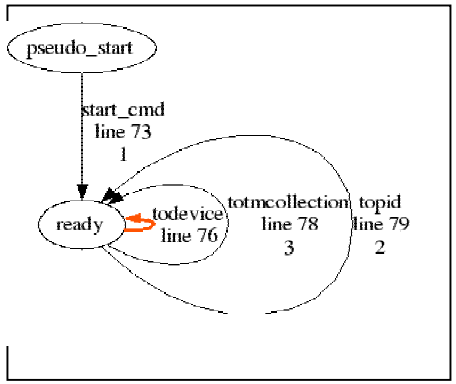
Complementary Capabilities: Coverage

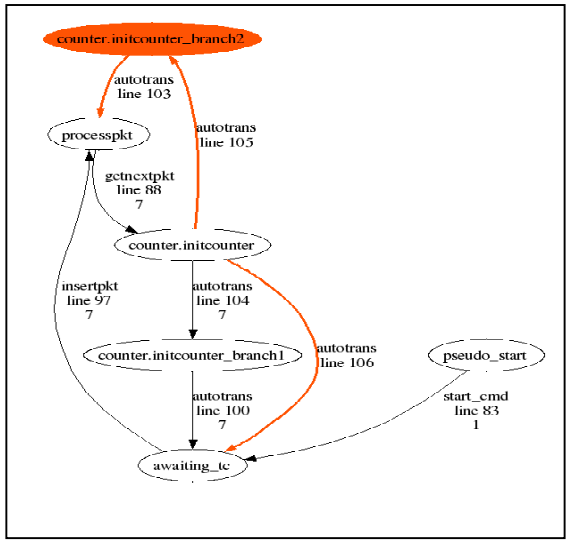
BSSE System and Software Engineering

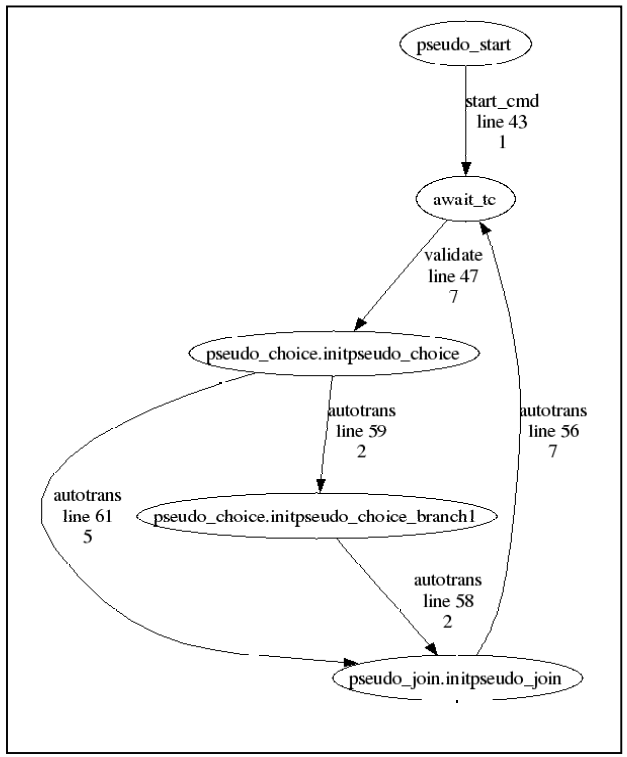


Loss of signals
(10% probability)

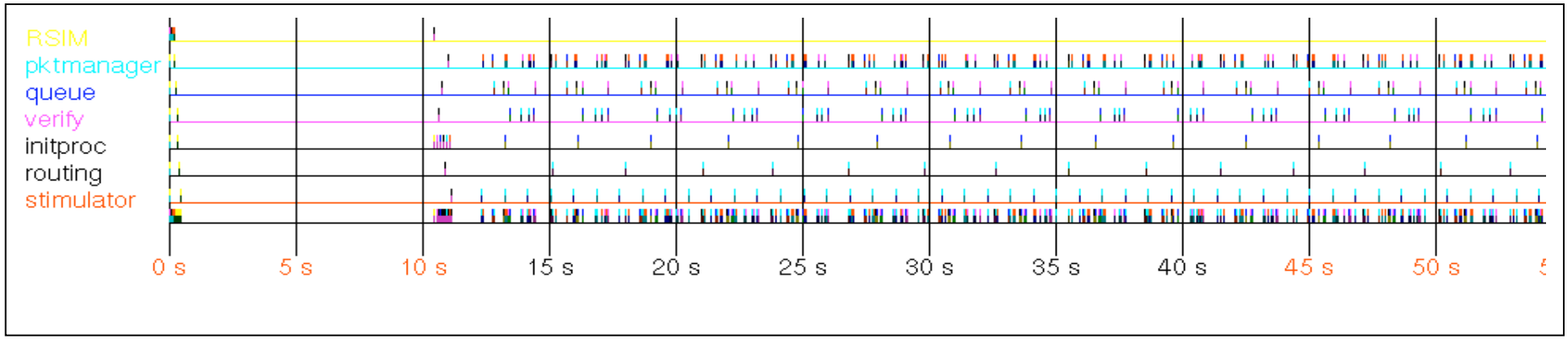
makes it worse

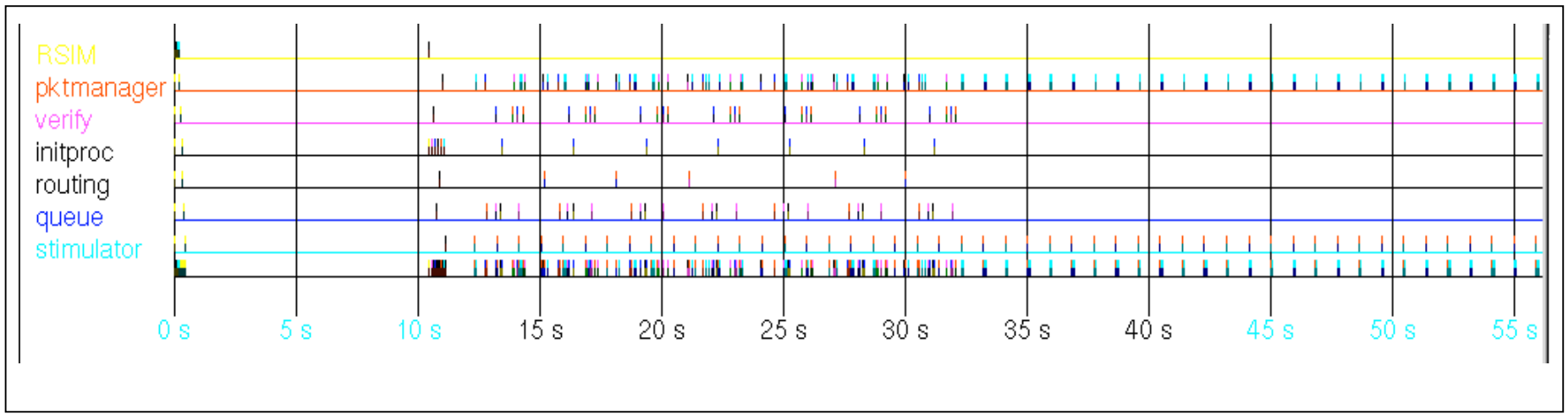




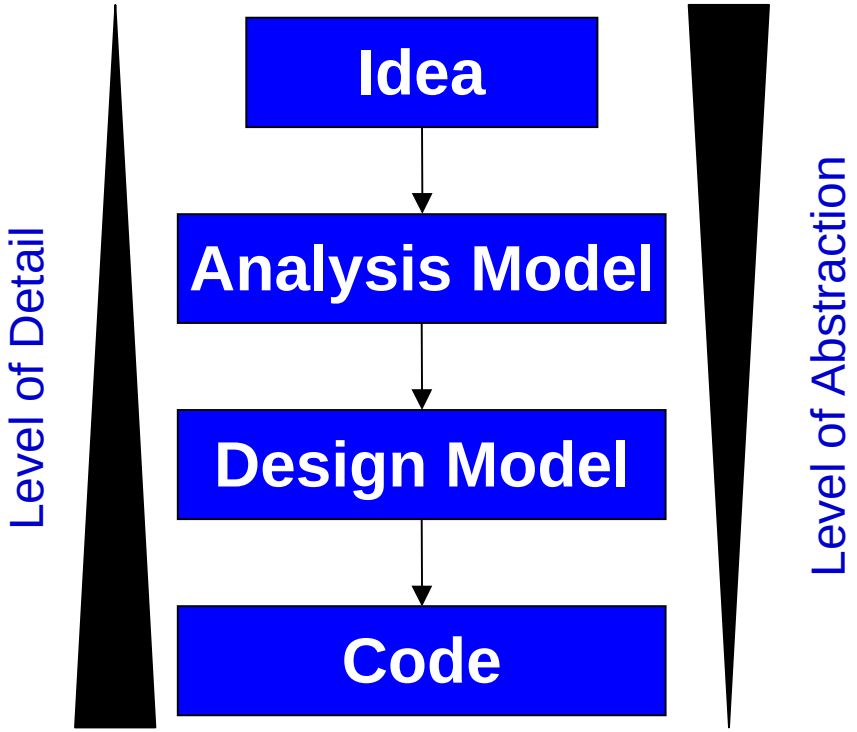


Complementary Capabilities: Timing





Expressiveness of Models



- Model is abstract, details are ignored
- Implementation is concrete
  - acquire details
    - from language semantics
    - from platform (e.g. OS API, H/W architecture)

Language Semantics

- Information on the System is either
  - in the Model, or
  - in the Language Semantics, or
  -  in the Implementer (programmer, code generator)
- Semantics:
  - What is allowed?
  - What does it mean?
- Semantics means Information
  - for verification
  - for disambiguation
  - for optimisation

Semantics in UML2

- Multiple Triggers on one Transition
  - runtime behaviour not defined (deliberately, SVP)
  - order relevant?
  - both required to trigger or one sufficient?
- Accessing Signal Contents
  - not formally defined in abstract syntax
  - only by opaque expression
  - modeller's (language) choice ? tool's problem
- Missing Targets for Signals
  - UML2 allows specification via ports
  - but does not insist on specification

Semantics in 3ADL (Profile)

- Architecture mainly empty shells
  - only modelling tool for decomposition
- Port connections and buses
  - Two ports connected via many levels of hierarchy
  - Partial connections can be bound to different buses
- No concept of behaviour
  - Not original focus of 3ADL Profile
  - Reuse of concept from UML2 lead to similar problems as for UML2 experiment

Conclusions

- 3ADL more suitable
  - constructive approach: Define allowed elements
  - therefore more suitable for tool-based analysis
- Assumptions needed to allow transformation
  - might not be applicable for other applications
- More faults detected in tested model
  - observed on level of generated code
  - analysis found them in model as well
  - not visible by pure review or stepping
- More rules, more value!
  - more strict modelling concepts required