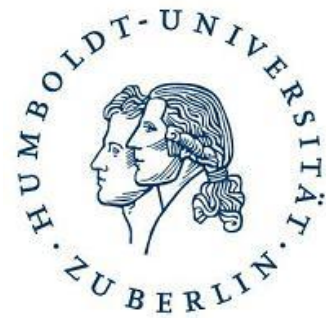


SUMMERSOC 2014

Wed July 3rd 10:30 - 12

Wed July 3rd 15 – 16.30



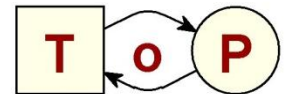
Tutorial

Formal Methods for SOC

1. A Survey



Wolfgang Reisig



Theory of
Programming

Prof. Dr. W. Reisig

Modeling ...

“Models as first class citizens” (G. Booch)

“Model houses” instead of Software houses

Open models (in analogy to open source software)

Models in this Summer school:

models of software?

models of systems?

data models? behavioral models?

Monday: Uwe Breitenbücher:

“How *model* all these things?”

“Winary” is a tool to graphically *model* TOSCA topologies and plans

Topology Template (also referred to as the *topology model* of a service) defines the *structure* of a service.

Tuesday: Pablo Fernandez:

“... a *modeling* tool to write SLAs”

... formalize real agreements in documents
by means of propositional logic,
with a grain of “first this, then that”

Tuesday: Holger Schwarz:

SIMPL Workflow *modeler* ...

Tuesday: Sebastian Lehrig

model-driven development

“logic component”

model: a structured (graphical) representation

Modeling: effort > risk

more precise:

effort to construct a model > gains obtained by a model

to reverse this: Increase the right side!

models are not only to clarify,

but also to verify and to automatically generate code.

summing up

Modeling mostly refers to *static structures*,
e.g. data structures
dominated by the relational data model

Modeling the *behavior*
of systems is rare ...

This talk's topic: behavior modeling

Formal Methods: A Survey

Modelling

- elementary: *discrete steps*
- *a behavior (run): a set of step occurrences. finite? infinite?*
a sequence in
- *system: a set*
an *event structure*
 - represented aut
system, Petri net, BPMN, CHAM, Tim
 - represented gra
algebra with π -calc, μ graphs
 - including data:
Alg. Spec., ALLOY, ASM, B, CASL, DSLs, FOCUS, TLA, pattern lang., PN schema, Z

Analysis

- the problem of decidability
- *loosely coupled systems*
- *interacting while computing*
- *running forever*
- bisimulation
- refinement checking
- structural analysis

Design

- correct by construction
- controlled composition
- stepwise refinement
- implementation is implication
- composition is conjunction

what is so special about SOC?

- *loosely coupled systems*
- *interacting while computing*
- *running forever*

classical theory does not apply

Formal Methods: A Survey

Modelling

- elementary: *discrete steps*
- *a behavior (run): a set of step occurrences.* finite? infinite?
a sequence in time? a partial order ?
- *system:* a set of runs? a tree of runs?
an *event structure*?
- represented automata like: transition system, Petri net, statechart, MSC/LSC, BPMN, CHAM, Timed Aut., Hybrid Aut. ,
- represented grammar like: process algebra with π -calculus, bigraphs
- including data:
Alg. Spec., ALLOY, ASM, B, CASL, DSLs, FOCUS, TLA, pattern lang., PN schema, Z

Analysis

- the problem of decidability
- 1st order logic
- **temporal logic**
- **model checking**
- theorem proving
- certification
- (inductive) invariants
- structural analysis
- **simulation**
- **bisimulation**
- refinement checking

Design

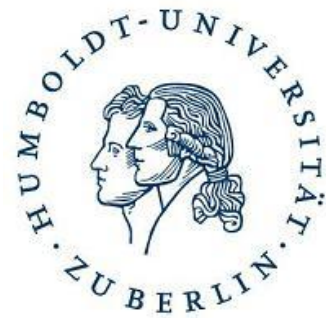
- correct by construction
- controlled composition
- stepwise refinement
- implementation is implication
- composition is conjunction

***today and tomorrow:
some highlights
of all this***

SUMMERSOC 2014

Wed July 3rd 10:30 - 12

Wed July 3rd 15 – 16.30



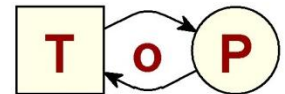
Tutorial

Formal Methods for SOC

1. A Survey



Wolfgang Reisig



Theory of
Programming

Prof. Dr. W. Reisig

the end