

# Inquiry- and Research-based Teaching in a Course on Model Checking

Sebastian Krings, Philipp Körner, Joshua Schmidt

Niederrhein University of Applied Sciences, University of Düsseldorf



# State of the Art

- Model checking research connects theoretical and practical aspects
- New algorithms are often implemented inside well-known model checkers
  - ▶ In development for many years
  - ▶ Complex
  - ▶ Large code volume
  - ▶ High entry barrier

## However

This is seldom taken into account by university courses, which often remain on the theoretical level.



## State of the Art ...

Our course wasn't any better.



# State of the Art ...

Learning objectives:

- Students can present and compare different techniques
- Students know common algorithms and can implement them
- Students can summarize selected literature and are able to criticize
- Students can write their own specifications and evaluate them

So obviously

A purely frontal lecture-based course is the way to go.



## State of the Art ...

Oh boy was our course missing the  
point.



# Major Shortcomings

- Learning results reduced due to missing hands-on experience
- Scope of thesis topics is limited, as students have not learned how to appropriately address practical problems
- Missing experience in project work, tool usage and working collaboratively
  - ▶ Students do not meet expectations from industry
  - ▶ Skills could be acquired en passant in a programming project



# Course Redesign

- High-level idea
  - ▶ Acquire the theoretical foundations by identifying and analyzing common software errors
  - ▶ Align these foundations with the body of knowledge
  - ▶ Design and implement a novel model checker as independently as possible
- Important aspects for success (Baron, et al.)
  - ▶ Selecting appropriate learning goals
  - ▶ Begin with problem-based learning before project work
  - ▶ Enable self-assessment and revision
  - ▶ Develop an atmosphere and social structures that support participation



# Major Challenges

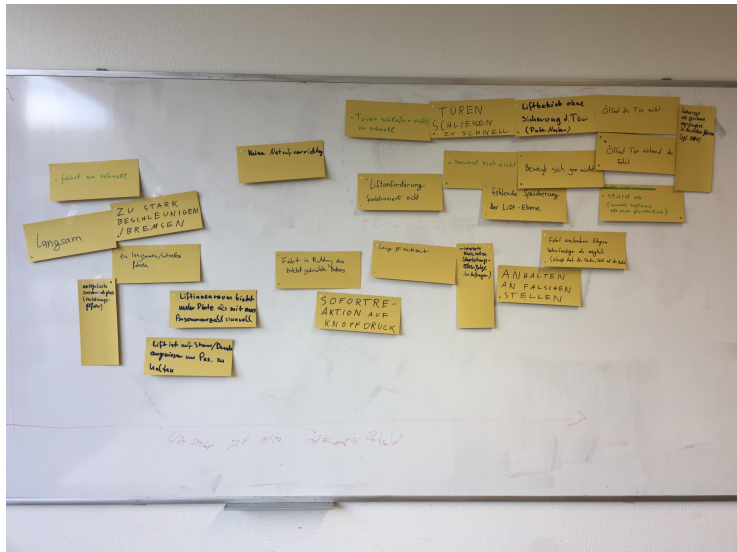
- Cognitive requirements are higher
  - ▶ Switch from knowledge reproduction to production
- Progression is less linear
- Individual workload is increased
  - ▶ Motivation and commitment has to be increased
- Research has to be controlled to avoid getting off track
  - ▶ As much freedom as possible, but guarantee intended learning outcomes
- Exams have to be prepared carefully to meet didactic requirements and exam regulations







# Problem-based: Sorted Hazard Collection



# Course Sessions

After problem-based introduction, iterate between

- Project management
- Implementation
- R&D sessions
- Reflection & evaluation sessions





# Project Management: Kanban



The image shows a Kanban board with five columns, each representing a different stage of the project workflow. The columns are: Backlog (3 items), To Do / Ready (3 items), In Development (2 items), Testing / In Review (1 item), and Done (100 items). Each column contains task cards with titles, descriptions, and assignees.

- Backlog (3):**
  - Issues which can not be resolved right now, either because they are blocked or because we do not have the resources to handle them right now. Added by wysib
  - Model checker performance drop for CounterErr2 #86 opened by leuschel
  - Implement Cardinality of finite Sets #87 opened by Marekias **enhancement**
- To Do / Ready (3):**
  - Issues which can be acted upon. However, nobody is working on them yet. Added by wysib
  - Exceptions logged but not shown in the UI #93 opened by heinzware
  - Spurious counter-example found in the presence of deferred sets #92 opened by leuschel
- In Development (2):**
  - Issues somebody is working on at the moment. Added by wysib
  - Generalized Union #48 opened by Marikes **enhancement**
- Testing / In Review (1):**
  - Resolved issues that have not been tested. Please test them and close + move them to done if they are indeed resolved. If further issues are discovered during testing, move them back to "In Development". Added by wysib
- Done (100):**
  - We are done! Added by wysib
  - Implement equalAst(...) Method to enable Node being checked for equality #101 opened by x-moa-x **enhancement**
  - Implement LTL model checking #96 opened by wysib **enhancement**
  - Büchi Automaton of G (1-1) Includes only accepting states #104 opened by wysib **bug invalid**






# Project Management: Kanban



**Backlog** 3  

 Issues which can not be resolved right now, either because they are blocked or because we do not have the resources to handle them right now. 


Added by wysiib



 **Model checker performance drop for CounterErr2** 



#86 opened by leuschel 

 **Implement Cardinality of finite Sets** 



#47 opened by Mareikes

**enhancement** 



**To Do / Ready** 3  

 Issues which can be acted upon. However, nobody is working on them yet. 

Added by wysiib

 **Exceptions logged but not shown in the UI** 

#93 opened by heinzware

 **Spurious counter-example found in the presence of deferred sets** 

#92 opened by leuschel



# Project Management: Kanban

The image shows a Kanban board with three columns: "In Development", "Testing / In Review", and "Done".

- In Development (2 items):**
  - Issues somebody is working on at the moment. Added by wyslib
  - Generalized Union #48 opened by Mareikes. enhancement
- Testing / In Review (1 item):**
  - Resolved issues that have not been tested. Please test them and close + move them to done if they are indeed resolved. If further issues are discovered during testing, move them back to "In Development". Added by wyslib
- Done (100 items):**
  - We are done! Added by wyslib
  - implement equalAst(...) Method to enable Node being checked for equality. 2 of 2. #101 opened by x-moe-x. enhancement
  - implement LTL model checking. 4 of 4. #96 opened by wyslib. enhancement
  - Büchi Automaton of G {1=1} includes only accepting states. #104 opened by wyslib. bug invalid



# R&D: Algorithm Development

MACHINE intbr  
VARIABLES  
C

INVARIANTS  
C: NATURAL

INITIALIZATION  
C := 0

OPERATIONS  
inc = C := C + 1  
err = PRE c < 0 THEN c := -1 END

END

Diagram: A sequence of circles containing C=0, C=1, C=2, C=3, with arrows indicating the flow. A box contains the invariant:  $0 \leq C \leq 100000$  and  $C \in \mathbb{N}$ . Annotations include "Verloren nicht invariant", "Clean invariant", and "Erreicht".

MACHINE lbr  
VARIABLES  
C, b

INVARIANTS  
C: INTEGER &  
b = TRUE

INITIALIZATION  
C := 0 & b := TRUE

OPERATIONS  
inc = ANY x  
WHERE x: INTEGER  
THEN C := C + x

Diagram: A state transition diagram with nodes (C=0, b=TRUE), (C=1, b=TRUE), (C=2, b=TRUE), (C=3, b=TRUE), and (C=100000, b=TRUE). A box contains the invariant:  $C \in \mathbb{Z}$  and  $b = \text{TRUE}$ . Annotations include "C=0 & b=TRUE" and "C=100000 & b=TRUE".



# Course Evaluation

Different approaches to course evaluation

- Peer review by other teachers
- Direct student feedback
- Grades
- Learning data analytics





# Grading

- Exam should measure both theoretical and practical aspects
- Ensure that grading complies with the examination regulations
- Improve constructive alignment (Biggs, 1996):
  - ▶ Formative part: constant participation documented using the Kanban board
  - ▶ Attitude, soft skills observed but hard to grade
  - ▶ Theoretical foundations by summative exam
- Combined exam verifies learning objectives!



# Grades

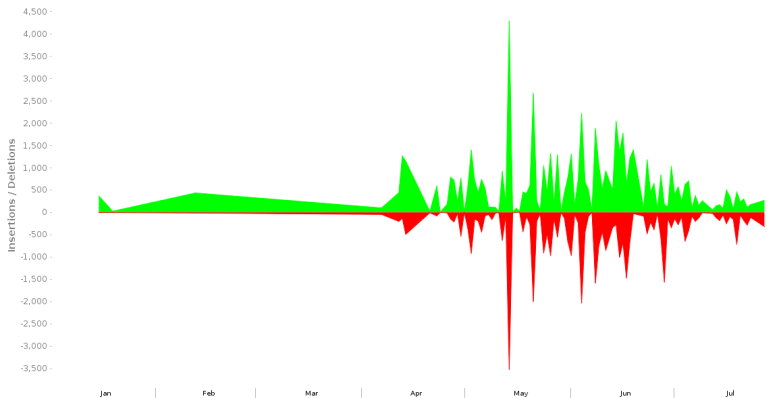
---

	2014	2015	2016	<b>2017</b>	2018
# Students	2	5	7	<b>6</b>	5
∅ Grade	1.85	2.58	1.71	<b>1.28</b>	1.88

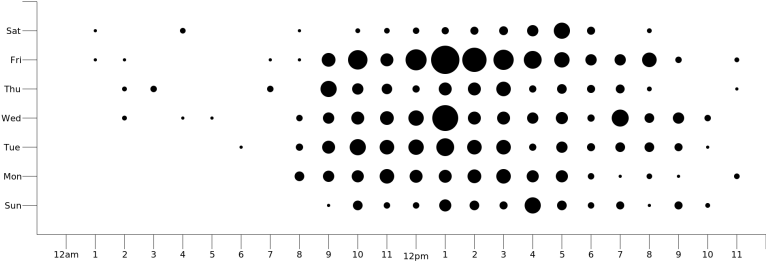
---



# Evaluation: Additions and Deletions to Sources



# Evaluation: Activity



# Experience Report on an Inquiry-Based Course on Model Checking

Sebastian Krings

Universität Düsseldorf, Hochschule Niederrhein

krings@cs.uni-duesseldorf.de

Philipp Körner, Joshua Schmidt

Universität Düsseldorf

{koerner,schmidt}@cs.uni-duesseldorf.de

### Abstract

coming to grips with internal workings. In consequence, typical courses on model checking stay on a



# Last Step of Research: Publication

- Not part of the course, as we could not reasonable expect anything
- Still 3 students were interested
- Outside of curriculum:
  - ▶ Discuss publication process, peer review, etc.
  - ▶ How to write interesting paper (mostly following Peyton Jones)
  - ▶ Brainstorm possible topics and ideas
  - ▶ Writing, meetups for synchronization



## One Question Remains

To what extent where our students  
doing relevant research?





## Writing a Model Checker in 80 Days: Reusable Libraries and Custom Implementation

Jessica Petrasch, Jan-Hendrik Oepen, Sebastian Krings, Moritz Gericke

Institut für Informatik, Universität Düsseldorf  
Universitätsstr. 1, D-40225 Düsseldorf

{jessica.petrasch, jan-hendrik.oepen, moritz.gericke}@hhu.de  
krings@cs.uni-duesseldorf.de

**Abstract:** During a course on model checking we developed BMoth, a full-stack model checker for classical B, featuring both explicit-state and symbolic model checking. Given that we only had a single university term to finish the project, a particular focus was on reusing existing libraries to reduce implementation workload.





# Conclusions

- Goals met
- Realization more hassle-free than anticipated
- Highly motivating for students, immediate sense of relevance
- Scaling and proper knowledge propagation difficult

## Summary

10 of 10, would teach that way again.



Thank you!

Any questions?

