New book
“Formal Methods for Software Engineering Languages, Methods, Application Domains”

Bernd-Holger Schlingloff, Markus Roggenbach

IFIP WG Meeting, March 2021

Motivation

Software engineering and Formal Methods:

- Current software engineering practices fail to deliver dependable software.
- Formal methods are capable of improving this situation, and are beneficial and cost-effective for mainstream software development.
- Education in Formal Methods is key to progress things.
- Education in Formal Methods needs to be transformed.

From "Rooting Formal Methods Within Higher Education Curricula for Computer Science and Software Engineering, a White Paper, Springer 2021.
`https://link.springer.com/chapter/10.1007/978-3-030-71374-4_1`

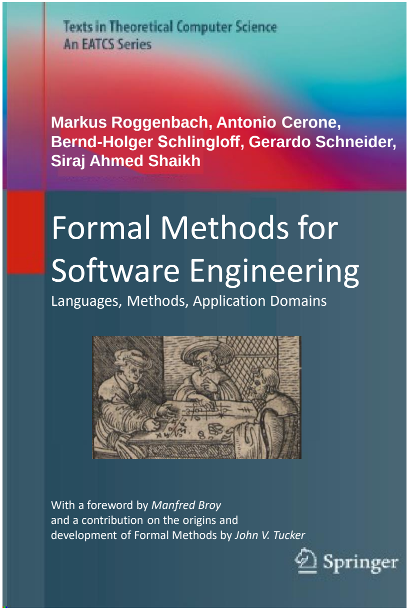
Last Slide from our Presentation in Prague, 2019

The challenge of Software Engineering has been acknowledged for decades now; the notion of software crisis has been related to the debate on software industry since early days.

Safe to presume the state of the industry will continue this course unless methods and practices are addressed.

Our work is motivated by the very challenge. The book is due out. Watch this space!

Available in Autumn 2021



Texts in Theoretical Computer Science
An EATCS Series

Markus Roggenbach, Antonio Cerone,
Bernd-Holger Schlingloff, Gerardo Schneider,
Siraj Ahmed Shaikh

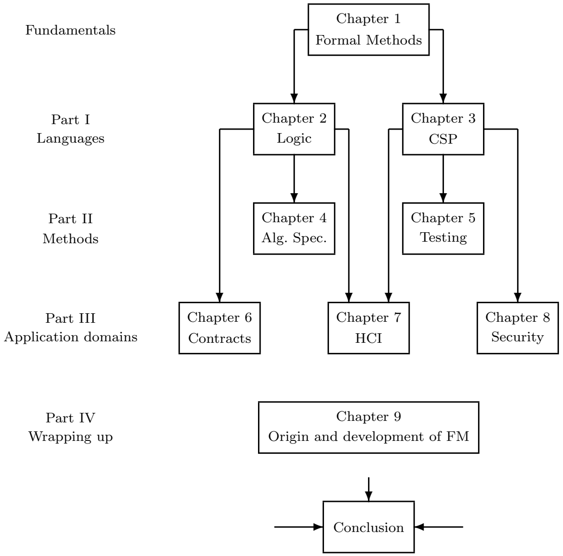
Formal Methods for
Software Engineering

Languages, Methods, Application Domains

With a foreword by *Manfred Broy*
and a contribution on the origins and
development of Formal Methods by *John V. Tucker*

Springer

Table of Contents



1. Formal Methods

- Motivating example: International Space Station, Byzantine Generals, Verification with CSP
- More thorough example: Regular Expressions
  - denotational, operational, and axiomatic semantics
  - regular replacement

Definition

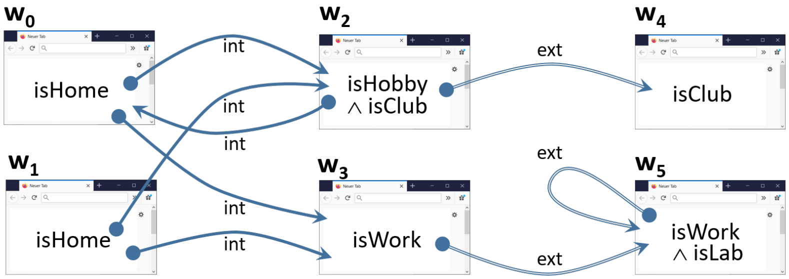
A Formal Method consists of syntax, semantics, and method.

- Formal Methods in Software Development
  - Software lifecycle, V-model
  - Use of formal methods
  - Taxonomy of formal methods
- Comparative surveys and case studies; success stories
- How to get started

2. Logic

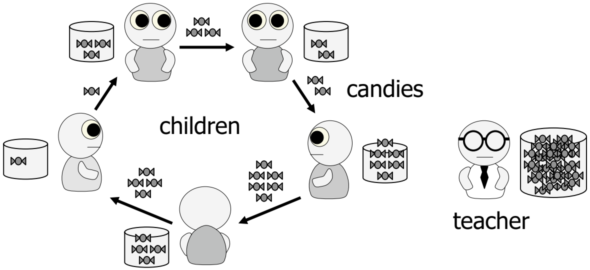
- Motivating example: car configuration
- Propositional logic as an institution
- PL, FOL, MSOL: syntax, semantics, calculi
- The logic of CASL
- Non-classical logics: modal, deontic, temporal

All demonstrated with examples from computer science.



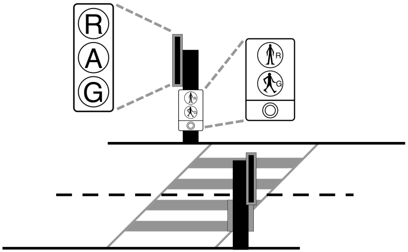
3. CSP

- ATM – for introduction of all CSP operators
- Starting Jet Engine controller – for introducing operational and denotational semantics of the traces model
- Modelling buffers – for studying refinement and motivating refusals
- Children's Puzzle – for demonstrating analyses with tools: Simulation, model-checking, theorem proving

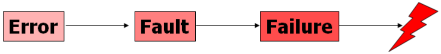


4. CASL

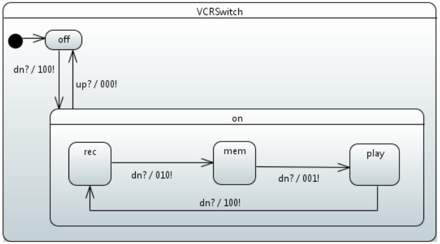
- Modelling, validating, consistency checking, and testing: Telephone Database
  - Automated theorem proving with Hets and Spass
  - Random testing of Java programs with ConGu
- Verification of Ladder Logic programs (as applied by Siemens for railway interlockings): Controller of a Pelican Crossing
  - PLCs; induction verification without and with invariants
- Structuring Specifications: Sorted Lists
  - Methodologically motivated application of structuring



5. Specification Based Testing

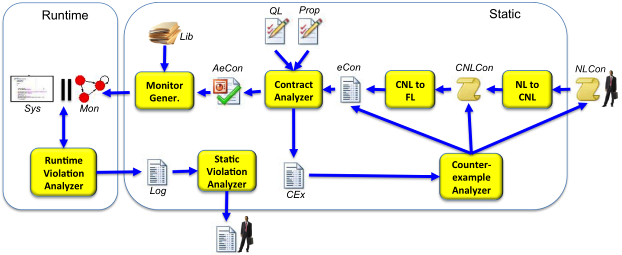


- State based testing: video recorder
  - State machine modelling, test generation and coverage
  - Conformance testing, IOCO
- Data based testing: calendar app
  - Test data generation from CASL specifications
  - Test evaluation
  - Completeness of test suites
- Tool support for testing



6. E-Contracts

- Running example: airline ground crew
- Contract language (CL) based on deontic logic
- Translation of controlled natural language into CL
- Conflict analysis of e-contracts



7. Human Computer Interaction

- Human errors and cognition
- Human memory and memory processes
  - CSP and TL modelling of brain capacity
- Human behaviour and interaction
  - CSP and TL modelling of cognitive, automatic, and deliberate control
- Analyses of interactions with model checking for
  - Cognitive overload
  - Task failure

Demonstrated with examples from ATM cash withdrawal, air traffic control, brain memory, car driving, and screen scanning.

8. Security Protocols

- Background
  - Public key cryptography, principles of security
  - Security protocols, attacks, Dolev/Yao intruder model
- CSP encodings
  - How to model security protocols in CSP
  - How to model different forms of authentication in CSP
- Analysis
  - Protocol modelling in CSPm and verification with modelchecking in FDR
    (attacker had bounded reasoning depth)
  - Protocol verification with rank functions
    (attacker has unbounded reasoning depth)

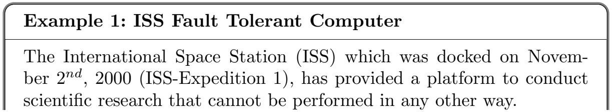
Running example: Needham/Schroeder and its correction.

9. History of Formal Methods (by John V. Tucker)

- Where do Formal Methods for software engineering come from?
- Logic
- Specifying programming languages and programs
- Specifications of data
- Reasoning and proof
- Concurrency
- Formal methods enter specialist areas

Writing Style

- Each chapter starts with a small anecdote or puzzle
- Examples play a huge role:



Example 1: ISS Fault Tolerant Computer

The International Space Station (ISS) which was docked on November 2^{nd}, 2000 (ISS-Expedition 1), has provided a platform to conduct scientific research that cannot be performed in any other way.

  - 77 different examples
  - often developed in several steps
    e.g., the corrected Needham/Schroeder protocol in eight steps.
- Concise formalisms are being used as far as possible.
- Frequent applications and links to computer science are given.
- Chapters end with annotated bibliography and research directions.
- The book includes a foreword by Manfred Broy.

Audience and Use

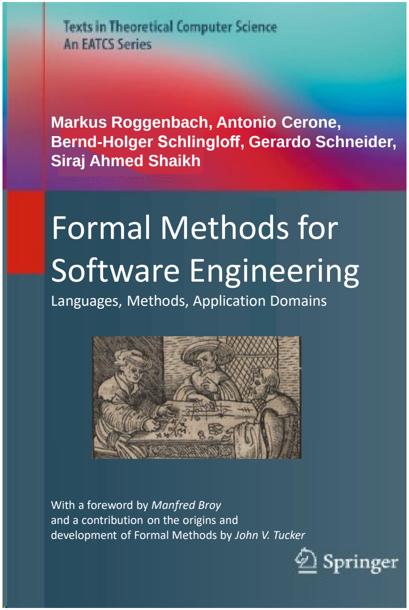
The book is appropriate for the following levels:

- Final year BSc students
- MSc students
- PhD students in the early phases of their research

The book can be used as:

- Underlying textbook for a university course (lab sheets and online tools available).
- Primary source for a seminar (2-3 student talks per chapter).
- Individual chapters can provide basic material for advanced module in the respective subjects.
- Self study for PhD students (crash course on a specific subject)

Available in Autumn 2021



Texts in Theoretical Computer Science
An EATCS Series

Markus Roggenbach, Antonio Cerone,
Bernd-Holger Schlingloff, Gerardo Schneider,
Siraj Ahmed Shaikh

Formal Methods for
Software Engineering

Languages, Methods, Application Domains

With a foreword by *Manfred Broy*
and a contribution on the origins and
development of Formal Methods by *John V. Tucker*

Springer