Systeme hoher Sicherheit und Qualität
Universität Bremen, WS 2017/2018

Lecture 13:

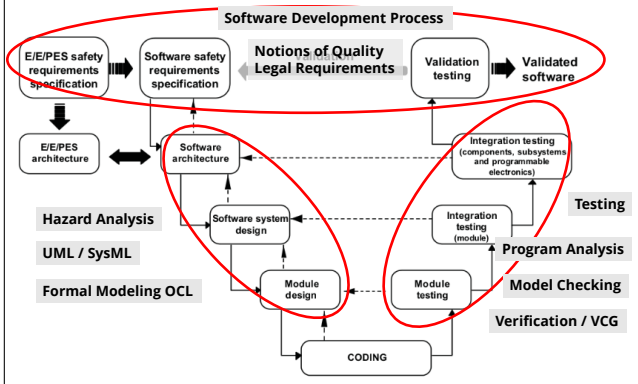
Concluding Remarks

Christoph Lüth, Dieter Hutter, Jan Peleska

Universität Bremen

Where are we?

The Global Picture



Examples of Formal Methods in Practice

- Hardware verification:
  - Intel: formal verification of microprocessors (Pentium/i-Core)
  - Infineon: equivalence checks (Aurix Tricore)
- Software verification:
  - Microsoft: Windows device drivers
  - Microsoft: Hyper-V hypervisor (VCC, VeriSoft project)
  - NICTA (Aus): L4.verified (Isabelle)
- Tools used in Industry (excerpt):
  - AbsInt tools: aiT, Astree, CompCert (C)
  - SPARK tools (ADA)
  - SCADE (MatLab/Simulink)
  - UPAAL, Spin, FDR2, other model checkers

Safe and Secure Systems – Uni Bremen

- AG Betriebssysteme - Verteilte Systeme / Verified Systems (Peleska)
  - Testing, abstract interpretation
- AG Datenbanksysteme (Gogolla)
  - UML, OCL
- AG Modelling of Technical Systems (Ehlers)
  - Modeling, decision procedures, synthesis
- AG Rechnerarchitektur / DFKI (Drechsler, Hutter, Lüth)
  - System verification, model checking, security
- AG Softwaretechnik (Koschke)
  - Software engineering, reuse

Organisatorisches

- Bitte nehmt an der **Evaluation auf stud.ip** teil!
- Was war euer Eindruck vom Übungsbetrieb im Vergleich zum herkömmlichen Übungsbetrieb?
  - Man lernt mehr – weniger?
  - Es ist mehr – weniger Arbeit?
  - Kommentare in Freitextfeldern bei der stud.ip Evaluation.
- Wir bieten an folgenden Terminen mündliche Prüfungen an:
  - Mi, 07.02.2018
  - Do, 15.02.2018
  - Mi, 28.02.2018

  Anmeldung per Mail etc.

Questions*

* Which might be asked in an exam, hypothetically speaking.

Universität Bremen

General Remarks

- The exam lasts 20-30 minutes, and is taken solitary.
- We are not so much interested in well-rehearsed details, but rather in principles.
- We have covered a lot of material – an exam may well not cover all of it.
  - We will rather go into detail then spend the exam with well-rehearsed phrases from the slides.
  - Emphasis will be on the later parts of the course (SysML/OCL, testing, static analysis, Floyd-Hoare logic, model-checking) rather than the first.
  - If you do not know an answer, just say so – we can move on to a different question.

Lecture 01: Concepts of Quality

- What is quality? What are quality criteria?
- What could be useful quality criteria?
- What is the conceptual difference between ISO 9001 and the CMM (or Spice)?

Lecture 02: Legal Requirements

- What is safety?
- Norms and Standards:
  - Legal situation
  - What is the machinery directive?
  - Norm landscape: first, second, third-tier norms
  - Important norms: IEC 61508, ISO 26262, DIN EN 50128, Do-178B/C, ISO 15408,...
- Risk Analysis:
  - What is SIL, and what is for? What is a target SIL?
  - How do we obtain a SIL?
  - What does it mean for the development?

Lecture 03: SW Development Process

- Which software development models did we encounter?
- How do the following work, and what are their respective advantages/disadvantages:
  - Waterfall model, spiral model, agile development, MDD, V-model
- Which models are appropriate for safety-critical systems?
- Formal software development:
  - What is it, and how does it work?
  - What kind of properties are there, how are they defined?
  - Development structure: horizontal vs. vertical, layers and views

Lecture 04: Hazard Analysis

- What is hazard analysis for, and what are its main results?
- Where in development process is it used?
- Basic approaches:
  - bottom-up vs. top-down (what does that mean?)
- Which methods did we encounter?
  - How do they work, advantages/disadvantages?

Lecture 05: High-level design with SysML

- What is a model (in general, in UML/SysML)?
- What is UML, what is SysML, what are the differences?
- Basic elements of SysML for high-level design:
  - Structural diagrams
    - Package diagram, block definition diagram, internal block diagram
  - Behavioural Diagrams:
    - Activity diagram, state machine diagram, sequence diagram
  - How do we use this diagrams to model a particular system, e.g. a coffee machine?

Lecture 06: Formal Modeling with OCL

- What is OCL? What is used for, and why?
- Characteristics of OCL (pure, not executable, typed)
- What can it be used for?
- OCL types:
  - Basic types
  - Collection types
  - Model types
- OCL logic: four-valued Kleene logic

Lecture 07: Testing

- What is testing, what are the aims? What can testing achieve, what not?
- What are test levels (and which do we know)?
- What are test methods?
- What is a black-box test? How are the test cases chosen?
- What is a white-box test?
- What is the control-flow graph of a program?
- What kind of coverages are there, and how are they defined?

Lecture 08: Static Program Analysis

- What is that? What is the difference to testing?
- What is the basic problem, and how is it handled?
- What does we mean when an analysis is sound/complete?

  What is over/under approximation?
- What analysis did we consider? How did they work?
  - What are the gen/kill sets?
  - What is forward/backward analysis?

Lecture 09: Floyd-Hoare-Logic

- What is the basic idea, and what are the basic ingredients?
- Why do we need assertions, and logical variables?
- What do the following notations mean:
  - $\vDash \{P\}\ c\ \{Q\}$
  - $\vDash [P] c\ [Q]$
  - $\vdash \{P\}\ c\ \{Q\}$
- How does Floyd-Hoare logic work?
- What rules does it have?
- How is Tony Hoare's last name pronounced?

Lecture 10: Verification Cond. Generation

- What do completeness and soundness of the Floyd-Hoare logic mean? Which of these properties does it have?
- What is the weakest precondition, and how do we calculate it?
- What are program annotations, why do we need them, and how are they used?
- What are verification conditions, and how are they calculated?

Lecture 11/12: Model Checking

- What is model-checking, and how is it used? What is the difference to Floyd-Hoare logic?
- What is a FSM/Kripke structure?
- Which models of time did we consider?
- For LTL, CTL:
  - What are the basic operators, when does a formula hold, and what kind of properties can we formulate?
  - Which one is more powerful?
  - Are they decidable (with which complexity)?
- Which tools did we see? What are their differences/communalities?

Thank you, and good bye.