Systeme Hoher Qualität und Sicherheit
Vorlesung 13 vom 27.01.2014: Concluding Remarks

Christoph Lüth & Christian Liguda

Universität Bremen

Wintersemester 2013/14

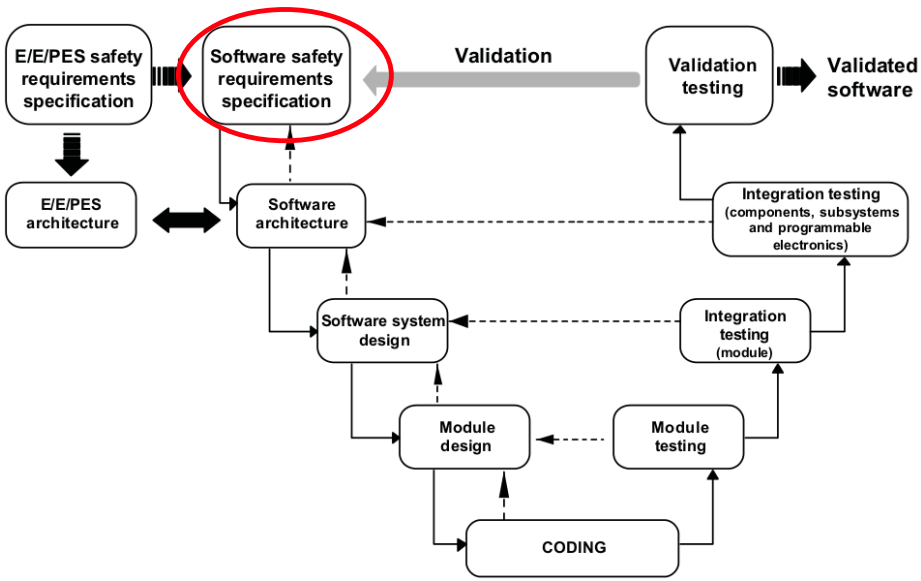
Where are we?

- Lecture 1: Concepts of Quality
- Lecture 2: Concepts of Safety and Security, Norms and Standards
- Lecture 3: Quality of the Software Development Process
- Lecture 4: Requirements Analysis
- Lecture 5: High-Level Design & Formal Modelling
- Lecture 6: Detailed Specification, Refinement & Implementation
- Lecture 7: Testing
- Lecture 8: Static Program Analysis
- Lecture 9: Verification with Floyd-Hoare Logic
- Lecture 10: Verification Condition Generation
- Lecture 11: Model-Checking with LTL and CTL
- Lecture 12: NuSMV and Spin
- **Lecture 13: Concluding Remarks**

Summary I

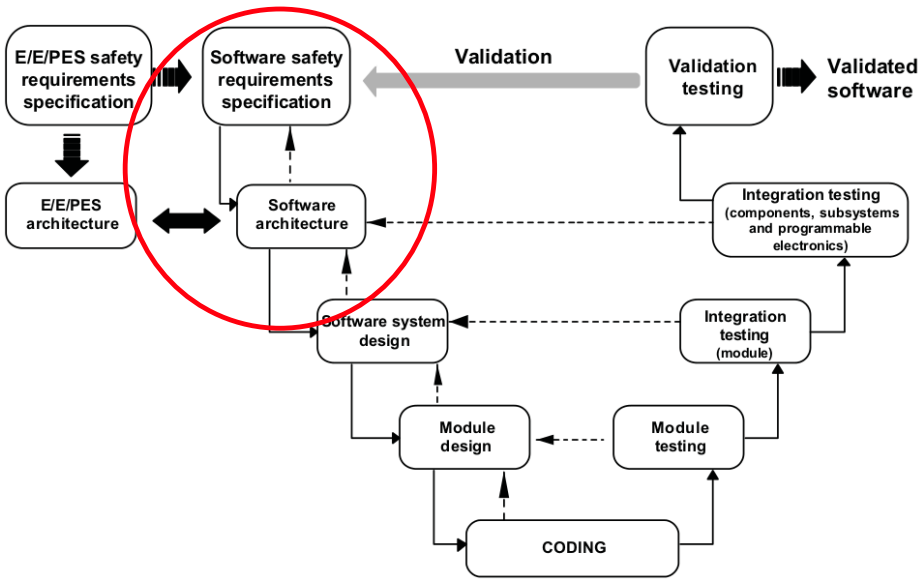
- This lecture series was about developing systems of **high quality** and **high safety**.
- Quality is measured by **quality criteria**, which guide improvement the development process.
- Safety is “freedom from unacceptable risks”.
- Both high quality and safety can be achieved by the means described in this lecture series.
- Moreover, there is the legal situation: the machinery directive and other laws require (indirectly) you use these techniques where appropriate.

Quality in the Software Development Process



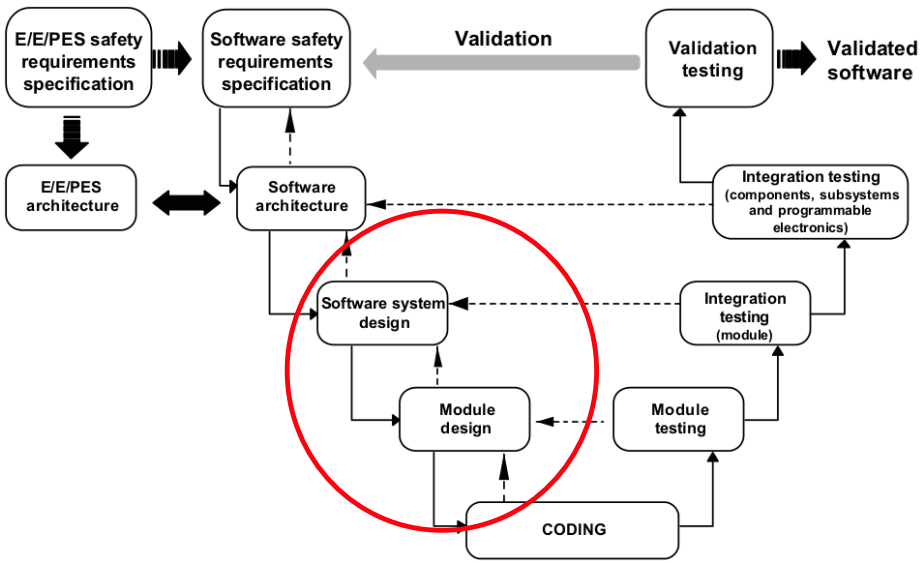
- Requirements analysis

Quality in the Software Development Process



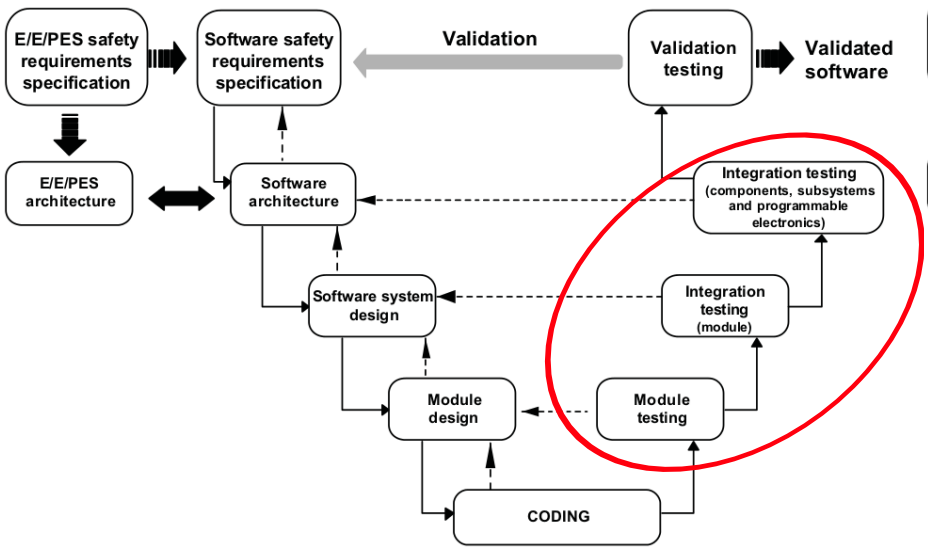
- Requirements analysis
- High-level specifications and formal modelling
  - The Z specification language

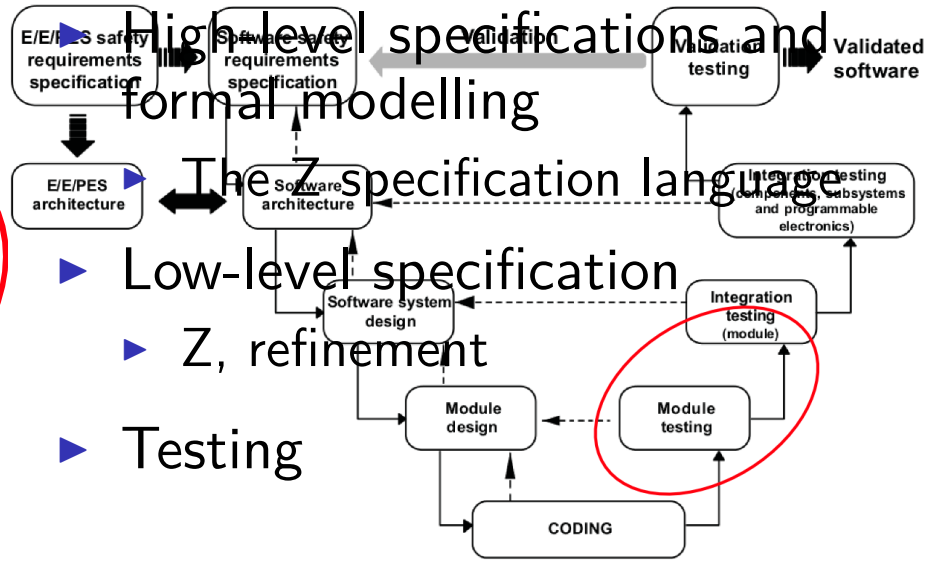
Quality in the Software Development Process



- Requirements analysis
- High-level specifications and formal modelling
  - The Z specification language
- Low-level specification
  - Z, refinement

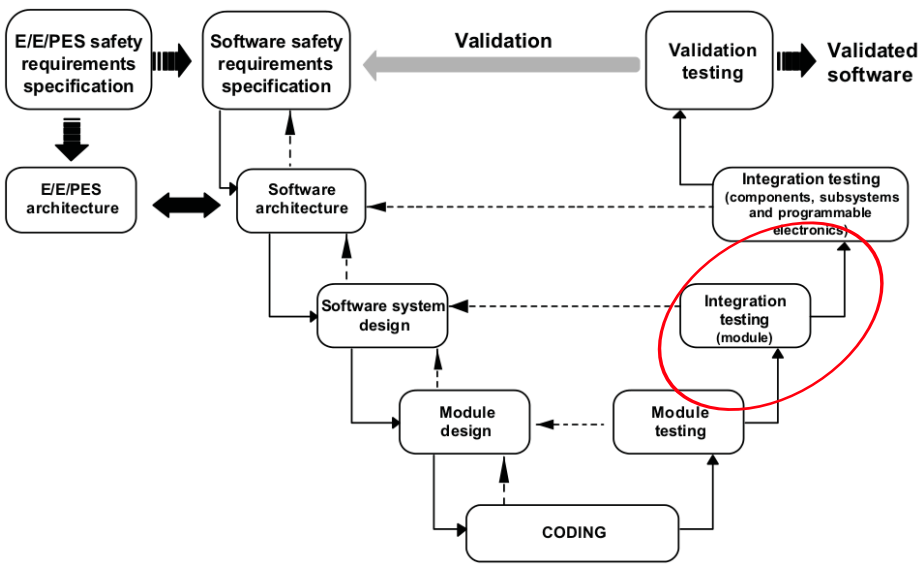
Quality in the Software Development Process





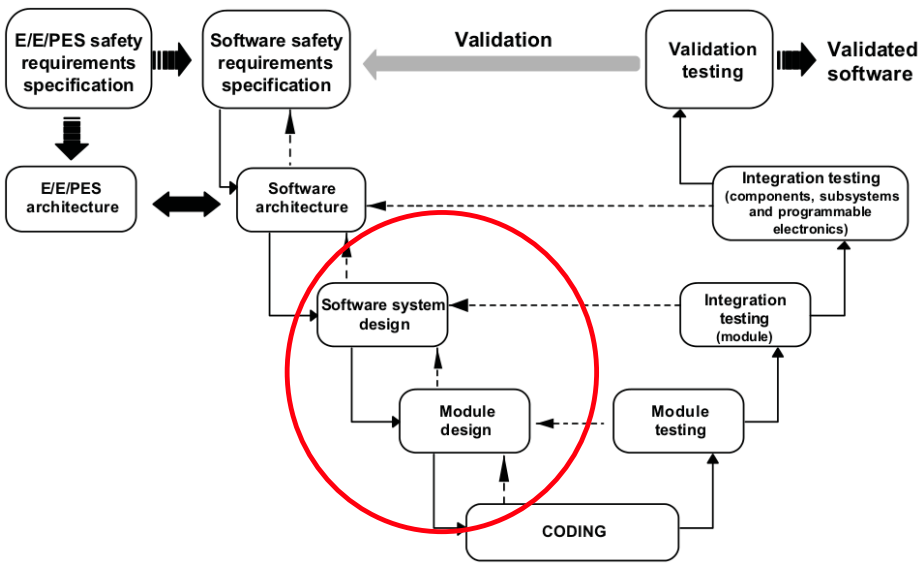
- Requirements analysis
- High-level specifications and formal modelling
  - The Z specification language
- Low-level specification
  - Z, refinement
- Testing

Quality in the Software Development Process



- Requirements analysis
- High-level specifications and formal modelling
  - The Z specification language
- Low-level specification
  - Z, refinement
- Testing
- Static Program Analysis

Quality in the Software Development Process



- Requirements analysis
- High-level specifications and formal modelling
  - The Z specification language
- Low-level specification
  - Z, refinement
- Testing
- Static Program Analysis
- Floyd-Hoare Logic

Quality in the Software Development Process

- Requirements analysis
- High-level specifications and formal modelling
  - The Z specification language
- Low-level specification
  - Z, refinement
- Testing
- Static Program Analysis
- Floyd-Hoare Logic
- Model-Checking

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 CMM?

Lecture 02: Concepts of Safety and Security

- 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, ISO 15408
- Risk analysis:
  - What is a SIL? Target SIL?
  - How do we obtain a SIL? What does it mean for the development?

Lecture 03: Quality of the Software Development Process

- Which software development models did we encounter?

Lecture 03: Quality of the Software Development Process

- Which software development models did we encounter?
- Waterfall, spiral, agile, MDD, V-model:
  - How does it work?
  - What are the advantages and disadvantages?
- Which models are appropriate for safety-critical developments?
- What are the typical artefacts (and where do they occur)?
- Formal software development:
  - What is it, and how does it work?
  - How can we define properties, what kind of properties are there, how are they defined?
  - Development structure: horizontal vs. vertical, layers and views

Lecture 04: Requirements Analysis

- What is hazard analysis?
- Where (in the development process) is it used?
- Basic approaches: bottom-up vs. top-down, and what do they mean?
- Which methods did we encounter?

Lecture 04: Requirements Analysis

- What is hazard analysis?
- Where (in the development process) is it used?
- Basic approaches: bottom-up vs. top-down, and what do they mean?
- Which methods did we encounter?
- FMEA, FTA, Event traces — how do they work, advantages/disadvantages?
- What are the prime verification techniques?

Lecture 05: High-level Design & Formal Modelling

- High-level specification and modelling:
  - What is it, where in the development process does it take place, what formalisms are useful?
- What is Z?
- Basic elements of Z:

Lecture 05: High-level Design & Formal Modelling

- High-level specification and modelling:
  - What is it, where in the development process does it take place, what formalisms are useful?
- What is Z?
- Basic elements of Z: Axioms, Schema, Mathematical Toolkit

Lecture 06: Detailed Specification, Refinement & Implementation

- What is refinement? How is it used in the development process?
- What kind of refinements did we encounter?
- What does refinement preserve?
- How do we do refinements in Z?
- How do we go from implementation to code — in general, and in Z?

Lecture 07: Testing

- What is testing, and what are the aims? What can it achieve, what not?
- What are test elevels?
- What is a black-box test? How are 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

- Is what? Where in the development process is it used? What is the difference to testing?
- What is the basic problem, and how is circumvented?
- What does it mean when we say an analysis is sound, or safe?
- What are false positives?
- Did we consider inter- or intraprocedural analysis?
- What examples for forward/backward analysis did we encounter?

Lecture 09: Verification with Floyd-Hoare Logic

- What is Floyd-Hoare logic, what does it do (and what not), and where is used in the development process?
- How does it work?
- What do the notations $\{P\}\, p\, \{Q\}$ and $[P]\, p\, [Q]$ mean
- What rules does the Floyd-Hoare logic have?
- How are they used?
- Which properties does it have?

Lecture 10: Verification Condition Generation

- What does VCG do?
- How is it related to Floyd-Hoare logic?
- What is a weakest precondition, and how do we calculate it?
- What are program annotations? Why are they used? How are they used?
- Which tools do VCG?

Lecture 11: Model-Checking with LTL and CTL

- What is model-checking, and how is it used? How does it compare with Floyd-Hoare logic?
- What is the basic question?

Lecture 11: Model-Checking with LTL and CTL

- What is model-checking, and how is it used? How does it compare with Floyd-Hoare logic?
- What is the basic question? $\mathcal{M} \models \phi$
  - What do we use for $\mathcal{M}$, ϕ, and do we prove it?
- What is a finite state machine, and what is temporal logic?
- 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?
  - Which one is decidable, and with which complexity?
- What is the basic problem (and limitation) of model-checking?
- Which tools did we see to model-check LTL/CTL?

Module Exams (Modulprüfungen)

- You may select **two** of the following areas:
  - Lectures 1– 4: Quality, Norms and Standards, Development Processes, Requirements Analysis
  - Lecture 5 – 6: Formal Modelling and Refinement, Z
  - Lecture 7 – 8: Testing and Static Program Analysis
  - Lecture 9 – 10: Floyd-Hoare Logic and Verification Condition Generation
  - Lecture 11 – 12: Model-Checking with LTL and CTL
- Questions will come from all lectures, but we will concentrate on your chosen areas.

Assessments (Fachgespräche)

- Questions will pertain to exercises.
- You may try to improve your grade; in this case, expect questions about the lecture material as well.