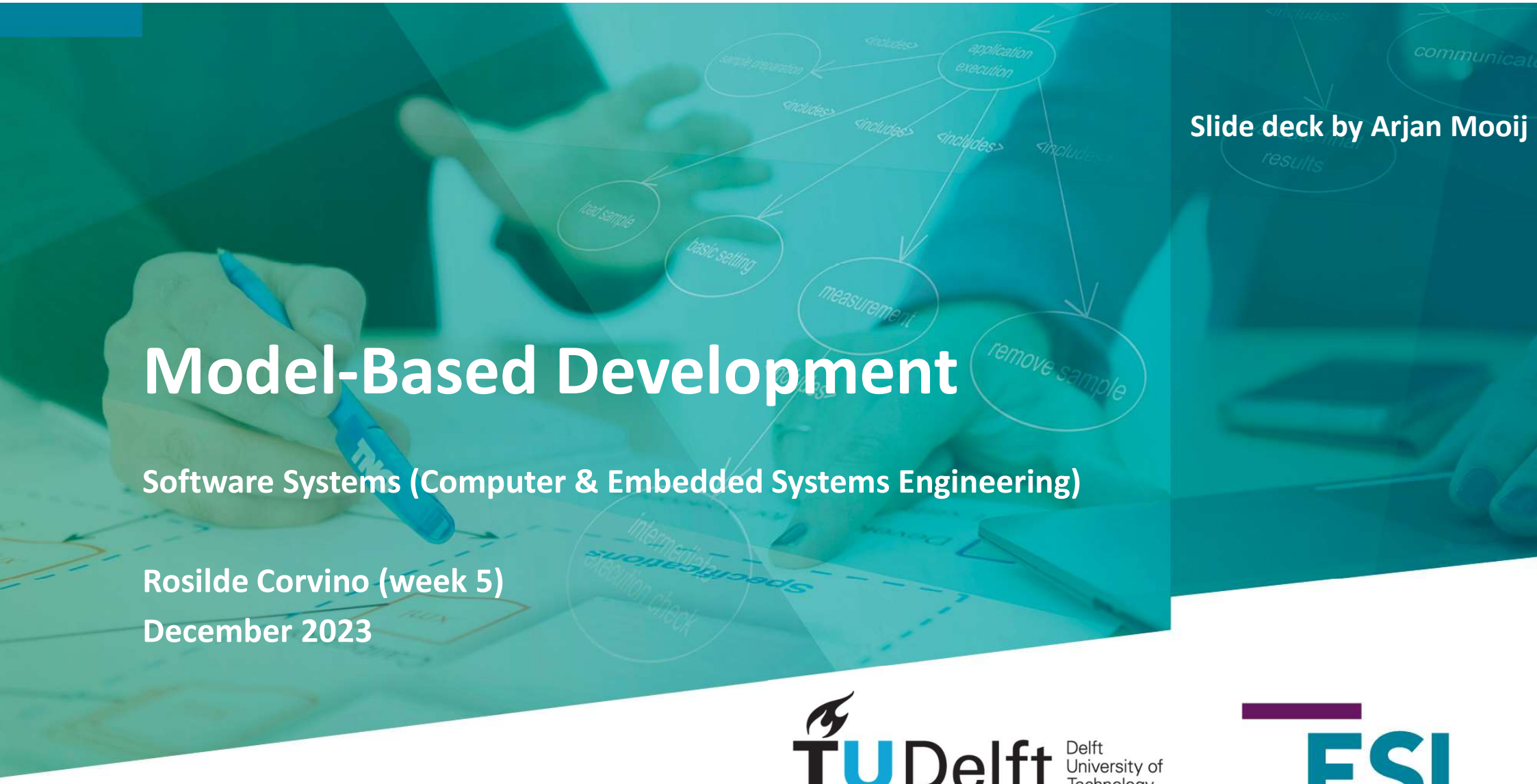


Model-Based Development

Software Systems (Computer & Embedded Systems Engineering)

Slide deck by Arjan Mooij

Rosilde Corvino (week 5)

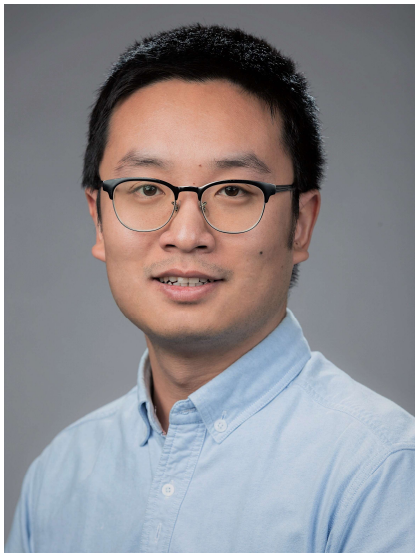
December 2023

An initiative of industry, academia and TNO

TU Delft Delft University of Technology

ESI

Learning facilitators



Jonathan Dönszelmann

Will Vadocz

Vivian Roest

Arthur de Groot

Julia Dijkstra

Rosilde Corvino (TNO-ESI)

Guohao Lan (TU Delft)

Teaching assistants (TU Delft)

TNO-ESI at a glance

Mission: ***Embedding leading edge methodologies into the Dutch high-tech systems industry to cope with the ever increasing complexity of their products.***

Synopsis

- ❑ **Foundation ESI started in 2002**
- ❑ **ESI acquired by TNO per January 2013**
- ❑ **~60 staff members, many with extensive industrial experience**
- ❑ **7 Part-time Professors**
- ❑ **Working at industry locations**
- ❑ **From embedded systems innovation to embedding innovation**

Focus

Managing complexity of high-tech systems

through

- **system architecting,**
- **system reasoning and**
- **model-driven engineering**

delivering

- **methodologies validated in cutting-edge industrial practice**

Partner Board

ASML, ThermoFisher Scientific, Philips, itec, Thales, Canon, Vanderlande, TNO, University of Amsterdam, TU Delft, Radboud University Nijmegen, University of Twente., TU/e Eindhoven University of Technology, Capgemini engineering

Objectives

At the end of the course, you should be able to:

- Explain some complexity challenges of software-intensive high-tech systems
- Explain 4 approaches for dealing with complexity
- Explain the purpose of Model-Based Development
- For 3 specific modeling techniques: explain their purpose and concepts + create basic models
- Compare Model-Based Development with other methodologies

Assessment:

- Modeling assignments for 3 modeling techniques (in groups of 2 students)
  - Maximum of 2 points per assignment
- Reflection document on Model-Based Development (individual)
  - Maximum of 4 points

Agenda for Model-Based Development
(Each week the Software Systems course has 2 lecture hours + 4 lab hours)

- **Week 5 Lecture**
  - 10 minutes Opening
  - 15 minutes Complexity challenges in large-scale software systems
  - 20 minutes Model-based development to manage the complexity

- **Week 5-8 Lectures/labs on 3 types of models (2 lecture hours for each type)**

- **Week 8 Lecture**
  - 35 minutes Reflection on 3 types of models
  - 10 minutes Wrap-up

Challenge

What are we talking about?

- **You are enrolled in a program on “Computer & Embedded Systems Engineering”**
- **You are following a course on “Software Systems”**
- **You may have an idea what you want to be working on after your studies**

What kind of systems do you know that are related to all of the following terms:

- Embedded systems
- Cyber-physical systems
- Software-intensive systems
- Dutch high-tech systems industry

Think/Write → Pair → Share

Dutch high-tech systems industry → Software-intensive systems

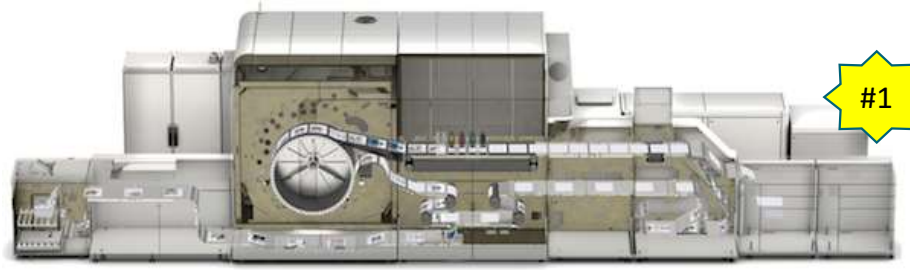
#1


#1


#1

#1

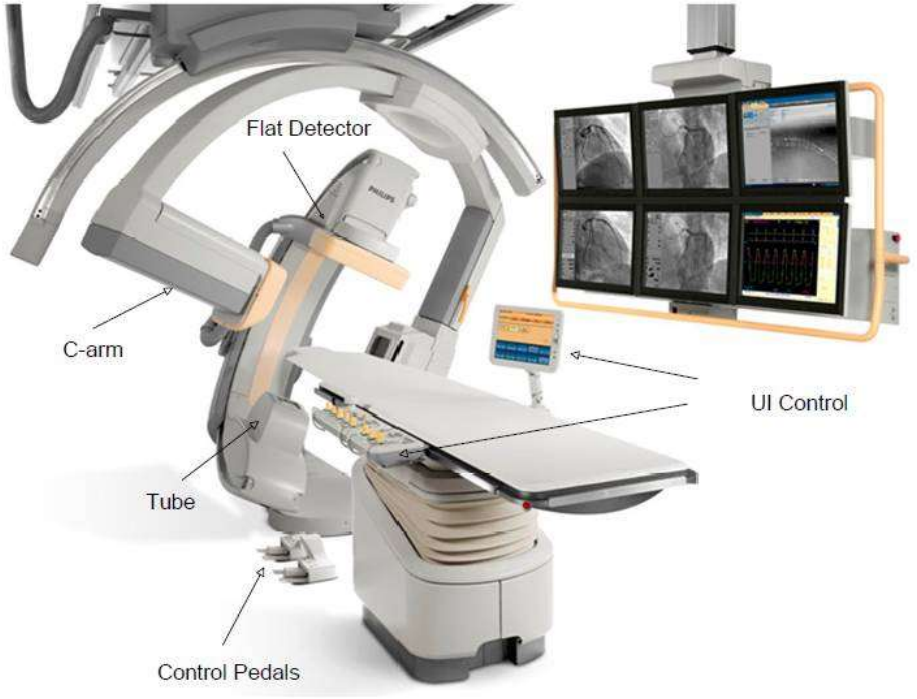

#1

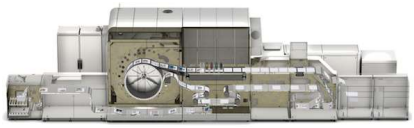

#1

Modeling assignment is inspired by interventional X-ray systems

- **Systems used during medical interventions:**
  - Minimally-invasive surgery
  - Cardiac, vascular and neurological
  - X-ray is used as “eyes of the surgeon”

- **1 or 2 X-ray planes (connected by a C-arm), consisting of:**
  - Tube: generates X-ray
  - Detector: receives X-ray
- **Table at which the patient lays**
  - Patient is positioned between the tube and detector
- **User interface for the surgeon:**
  - Tablet to select the medical procedure
  - Joysticks to move the table and C-arms
  - 3-6 pedals to control the system during a medical procedure
  - Screen to display visual images and video





Managing complexity in cyber-physical systems

Complexity = being difficult to understand, or lacking simplicity

Some related questions:

- **What makes such systems complex?**
- **What makes the software of such systems complex?**
- **What makes the design of such systems complex?**
- **What makes the design of the software inside such systems complex?**

Think/Write → Pair → Share

Complex software projects

- **Large code bases:** tens of thousands, or even millions, of lines of code (LOC)
- **Code reuse:** embedded software evolves over decades

- **Team effort:** multiple development teams that evolve over time
- **Multi-disciplinary:** closely related to physical systems and application domains

- **Integration:** interaction with other systems and humans (outside of your control)
- **Customization:** not many identical systems in the field

- **Performance:** quantitative performance criteria (hard real-time, competition)
- **Dependability:** non-functional requirements like availability, reliability, maintainability

Design Approaches

Approaches for dealing with complexity

- You have seen software-intensive systems from the Dutch high-tech systems industry
- You have seen that these systems and their development are complex
- You are following university education on challenging, complex topics

What kind of approaches do you know to handle complex challenges?

Think/Write → Pair → Share

Approaches for dealing with complexity

A. Abstraction: Identify high-level concepts that hide low-level details

- Architects design a building in terms of walls (instead of bricks)
- Aerospace engineers reason about processor instructions instead of individual transistors

B. Boundedness: Impose acceptable restrictions on the considered problem space

- Architects design a building for a specific set of usage scenarios (residential, industrial, retail)
- Aerospace engineers assume that airplanes do not need to be usable in outer space

C. Composition: Divide one problem into multiple independent smaller problems

- Architects design a new district in terms of buildings with separate foundations
- Aerospace engineers separate flight control from cabin control and in-flight entertainment

D. Duplication: Use multiple overlapping approaches for the same problem

- Architects use blueprints with multiple perspectives on the same building
- Aerospace engineers introduce fallback systems for safety-critical functionality

Model-Based Development

What do you think of when you hear the term “model-based development”?

What kind of engineering models do you know?

- **Level:** system, software, requirements, etc.
- **Representation:** graphical, textual, etc.
- **Etc.**

Think/Write → Pair → Share

Model-Based Development

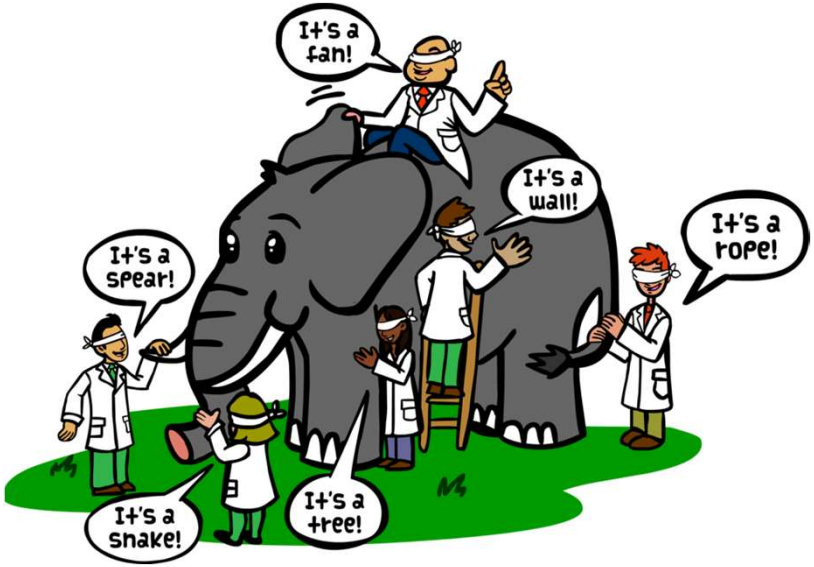
- **General modeling goals:**
  - Speeding up software development of large complex systems
    - Human understanding
    - Early validation
    - Code generation
    - Automated testing
  - Bridging the gap between application domain expertise and technical system realization
- **Model-based development uses all four approaches for dealing with complexity:**
  - Abstraction: Identify high-level concepts that hide low-level details
  - Boundedness: Impose acceptable restrictions on the considered problem space
  - Composition: Divide one problem into multiple independent smaller problems
  - Duplication: Use multiple overlapping approaches for the same problem
- **Notes:**
  - Modeling is for a specific purpose; there exist many different types of models
  - Modeling often helps you to detect important unclarities

Software development principles → also for modeling!

- **“Keep it simple, stupid” (KISS)**
  - https://en.wikipedia.org/wiki/KISS_principle
  - Most models work best if they are kept simple rather than made complicated
  - Simplicity should be a key goal in design, and unnecessary complexity should be avoided

- **“Don't repeat yourself" (DRY)**
  - https://en.wikipedia.org/wiki/Don%27t_repeat_yourself
  - Replace repetition of software patterns with abstractions and use data normalization (no variations)

Modeling for a specific purpose



- **In this course we will focus on the following 3 modeling techniques:**
  - Unified Modeling Language (UML)
  - Finite-State Machines (FSM)
  - Domain-Specific Languages (DSL)