Organization
The Lecture’s Goal
The goal is to enable you to systematically carry out small(er) commercial or open-source software projects.
Basic Goals

- To get a brief overview of “all” areas of software engineering
- To understand agile software development processes
- To get first hands-on experience and to learn to use basic software development tools
- To be able to perform object-oriented analysis and design
- To be able to read and create basic UML diagrams
- To be able to use basic design patterns
- To perform basic software quality assurance
Basic programming skills are required.

- Basic knowledge of object-oriented programming concepts is necessary i.e., you should readily understand the following terms:
  - (inner) class, interface
  - object
  - inheritance
  - polymorphism
  - virtual method
- Working knowledge of the Java programming language (Java 8)
- We may use further languages to discuss more advanced ideas
The Lecture’s Structure
What is Software Engineering?

Historical Background

Requirements Engineering

Software Project Management

Risk Management

Properties of Software Projects

Testing

Unit Testing

“Code Coverage”

Revision Control Systems

Software Engineering Tools

Software Engineering

Software Engineering Processes

Waterfall Model

Agile Methods

Object-Oriented Analysis and Design

UML Modeling

IDEs

Revision Control Systems

Unit Testing

“Code Coverage”

GIT

Object-Oriented Design

General Design Goals

Low Coupling

High Cohesion

Singleton

Template Method

...
Content / Structure of the Lecture

Software Engineering

What is Software Engineering?

Software Project Management

Testing

Software Engineering Tools

Object-Oriented Analysis and Design

UML Modeling

Historical Background

Properties of Software Projects

Requirements Engineering

Software Engineering Processes

Risk Management

Agile Methods

Waterfall Model

Dynamic Behavior

Static Structure

Class Diagrams

Sequence Diagrams

Communication Diagrams

Object Diagrams

Domain Modeling

Design Patterns

Idioms

General Design Goals

IDEs

Revision Control Systems

Unit Testing

“Code Coverage”

“Code Coverage”

GIT

Idioms

Design Patterns

General Design Goals

Low Coupling

High Cohesion

Singleton

Template Method

...
Organization
The Team

Dr. Michael Eichberg

Dominik Helm

Contact

Forum (D120 - Software Engineering)

Lecture

• Fridays 13:30-15:00 in S1 01 | A01 and S1 01 | A03

• The slides are in English
  (Key terms will be translated into German.)

• The slides will generally be available after the lecture
  (I will try hard to make a preliminary version available the day before the lecture.)

• The slides can be found at
Exercises

• Fridays 15:15-16:00 in S1 01 | A01 and S1 01 | A03

• Every week, we will have an exercise, starting next week.

• Exercises are expected to be solved in teams of 3 students.

• The content of the exercise is relevant for the exam.

• The exercises are the best way to prepare for the exam; do them on your own!

• Sign-up as a team until Oct. 28th; if you don’t have a team, we will assign you to a team.

• Go to our submission site to sign up for the exercises.
  http://submission.st.informatik.tu-darmstadt.de
  You have to be in the internal network of the TU Darmstadt.
Exercises - Bonus

• You can get a bonus by successfully completing the exercise.

• Exercise points will be converted to exam points as follows:
  \[ r = \frac{\text{gained exercise points}}{\text{all exercise points}} \]
  gained exam points = \( r \times \text{exam points required to get a full grade better} \)

• I.e., the maximum bonus is equivalent to getting the exam points necessary to get a full grade better (e.g., 2,0 => 1,0).

• The bonus cannot be used to pass the exam.
Written Exam

• The exam will be a closed-book exam.

• The date of the exam is: March, 27th 2019, 12:00
  (The rooms will be announced in the forum/moodle. The exam will take 90min.)

• You need to register for the exam in TUCaN.
  (There are no further prerequisites; “everyone” can attend the exam.)

• (Only) the very best students are expected to be able to solve the entire exam.
Essential Bibliography

- **Design Patterns - Elements of Reusable Object-Oriented Software**; Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides; Addison-Wesley, 1995

- **Applying UML and Patterns - An Introduction to Object-oriented Analysis and Design**; Craig Larman; Prentice Hall
Recent Episodes

Episode 211: Continuous Delivery on Windows with Rachel Laycock and Max Lincoln
Filed in Episodes by SE-Radio on September 30, 2014 • 0 Comments

Johannes talks with Rachel Laycock and Max Lincoln from ThoughtWorks about continuous delivery on Windows. The outline includes: introduction to continuous delivery; continuous integration; DevOps and ChatOps; decisions to be taken when implementing continuous delivery on windows; build tools on windows; packaging and deploy on windows; infrastructure automation and infrastructure as code with chef, puppet [...]

Continue Reading »

Episode 210: Stefan Tilkov on Architecture and