MTAT.03.094
Software Engineering

Course wrap-up, review and exam preparation

Dietmar Pfahl
email: dietmar.pfahl@ut.ee

Fall 2016
Schedule of Lectures

Week 01: Introduction to SE
Week 02: Requirements Engineering I
Week 03: Requirements Engineering II
Week 04: Analysis
Week 05: Development Infrastructure I
Week 06: Development Infrastructure II
Week 07: Architecture and Design
Week 08: Refactoring
Week 09: Verification and Validation I
Week 10: Guest Lecture I (Testing)
Week 11: Guest Lecture II (Software Quality Management)
Week 12: no lecture
Week 13: Verification and Validation II + Agile/Lean Methods
Week 14: Guest Lecture III (Software Craftsmanship)
Week 15: Course wrap-up, review and exam preparation
Week 16: no lecture
Course Information/Overview

• Level: Advanced course at bachelor's level (in English)
• Credits: 6 ECTS, 4 CP
• Prerequisite: MTAT.03.130 Object-oriented Programming (6 ECTS, 4 CP)
• Work load (per student):
  • Lectures: 26 hours
  • Lab work (incl. independent work): $14 \times (2 + 5) = 98$ hours
  • Exam preparation: 32 hours
• Assessment:
  • 7 Lab Assignment (team work) – 70% of grade
  • 1 Exam (written) – 30% of grade
• Grade scale: A (90%+), B(80%+), C(70%+), D(60%+), E(50%+), F
Assessment (1)

- Labs – 70% of total grade
- Exam – 30% of total grade

Proposed Exam Dates:
- Exam 1: Friday 06-Jan-2017
- Exam 2: Friday 13-Jan-2017
- (Retake: Friday 20-Jan-2017)

Room: 405 – exam 1 & 2
Time: 14.15 – 16.45 (150 min)

You can take either exam 1 or 2
You must register!

So far registered:
54 – Exam 1
14 – Exam 2
(of 112 students)
Assessment (2)

Labs – Practical Assessment

10 points per lab session. Total = 70 points.

If you get less than 30 out of 70 points in the labs, you will get a grade of 'F' in your first examination.

In this case, you will be given a second chance to improve your practical assessment (lab) score.

Condition: You have not received 0 marks in two or more labs due to non-submission

If your score after the second try is at least 30 out of 70, you will become eligible for a retake exam (korduseksam).

Exam – Conceptual Assessment

The Conceptual Assessment will consist of an exam worth 30 points.

Students who get less than 10 out of 30 in this exam, will get a grade of 'F', regardless of their Practical Assessment score.

This same rule will apply for the retake exam (korduseksam).
Lab Performance after 6 HW Assignments

Required minimum to qualify for exam
Schedule of Lectures – Relevant for Exam

Week 01: Introduction to SE
Week 02: Requirements Engineering I
Week 03: Requirements Engineering II
Week 04: Analysis
Week 05: Development Infrastructure I
Week 06: Development Infrastructure II
Week 07: Architecture and Design
Week 08: Refactoring
Week 09: Verification and Validation I
Week 10: Guest Lecture I (Testing)
Week 11: Guest Lecture II (Software Quality Management)
Week 12: no lecture
Week 13: Verification and Validation II + Agile/Lean Methods
Week 14: Guest Lecture III (Software Craftsmanship)
Week 15: Course wrap-up, review and exam preparation
Week 16: no lecture
Topic: Introduction to SE

- Know what a User Story is

Not covered due to my illness:
- Know how to define ‘Software Engineering’
- Know the elements of a software development process
- Know different kinds of development methods/processes
- Know the various characteristics (attributes) of software quality
Topic: Requirements Engineering I + II

- Know how to describe/define ‘Requirements Engineering’ (RE)
- Know about the difficulties eliciting and describing software requirements
- Know examples of conflicting requirements and how to deal with them
- Know about various types of RE contexts
  - types of project-settings, kinds of stakeholders, etc.
- Know the difference between ‘user requirements’ and ‘systems requirements’
- Know different ways of defining/specifying requirements
  - i.e., user stories and use cases
- Know the elements of a use case description
Topic: Requirements Engineering I + II (cont’d)

- Know the difference between functional and non-functional requirements (aka ’quality requirements’)
- Know the activities involved in the RE process (elicitation, analysis, specification, validation)
- Know the difficulties of requirements elicitation
- Know how to draw a use case diagram
- Know the elements of UML class diagrams
Topic: Analysis

• Know types of domain classes
  • business objects (e.g., order, receipt), real world objects (e.g., materials), actors/workers/persons (e.g., controller, customer), events (e.g., sale, payment)
• Know how to identify domain classes
  • E.g., by identifying nouns in domain descriptions
• Know how to identify attributes of domain classes
  • E.g., simple data types (like 'order number')
• Know how to create a domain model
  • Outside-in approach (boundary, control, entity concepts)
  • Setting-up an enterprise approach (‘workers’ vs. ‘things’, ‘doing’ vs. ‘knowing’ concepts)
Topic: Architecture and Design

- What is the main purpose of Architecture/Design? Why is it important? How is it related to quality?
- Know the purpose of architectural views
- Know examples of architectures (e.g., MVC)
- Know what Design Patterns are
Topic: Refactoring

- Know what 'refactoring' is
- Know examples of 'code smells' and examples of 'refactorings'
- Work through Martin Fowler’s introductory example on refactoring
Topic: Verification and Validation I + II

• Know the difference between ‘verification’ and ‘validation’
• Know what the ‘testing paradox’ is
• Know the difference between ‘error’, ‘fault’, ‘failure’
• Know the terms ‘test case’, ‘test suite’, ‘test oracle’, ‘test verdict’
• Know what testing levels, methods and types are
• Know the difference between ‘white-box’ and ‘black-box’ testing, and know why they are complementary
• Know how to apply ‘black-box’ and ‘white-box’ testing techniques
• Know test coverage criteria (and how to use them)
Topic: Agile/Lean Methods

- Know the spirit of the ‘agile manifesto’ and know how to explain/interpret it
- Know examples of agile software development methods
- Know elements of XP and Scrum (e.g., TDD, Planning Poker, Pair Programming)
- Know the difference between XP and Scrum
Exam – ‘Open Book’

MTAT.03.094 – Software Engineering

Written Exam – 10 January 2014

Start: 9:15 – End: 11:45

Important Notes:

- The exam is open-book and open-laptop. Web browsing is allowed, but you are not allowed to use e-mail clients nor Instant Messaging clients, nor to share any information “live” with anybody inside or outside the exam room.
- At the end of the exam you must submit both the question sheets and your answer sheets. To avoid that any of your solutions get lost, make sure to write your name (and student ID) on each sheet of paper that you submit. Also, please number the pages on your answer sheets.
- Write clearly. Answers that are illegible cannot be counted as correct answers. Only answers written in English will be marked.
- Total marks: 30 (equivalent to 30% total grade). You must get at least 10 marks in this exam to not fail the course.
Exam Structure

• Part 1 (10 marks): Multiple Choice – 30 min
• Part 2 (10 marks): Open Questions – 60 min
• Part 3 (10 marks): Problems where a solution needs to be developed (no coding) – 60 min
Exam – Part 1

Example (Exactly one correct answer):
• You execute a program that is supposed to add up 3 numbers n1, n2, and n3. You correctly enter the value ’1’ for each number and you see the following message on the screen: ’Sum = 4’. Which of the following terms describes your experience best:
  A. I made an error
  B. I debugged the program
  C. I located a fault
  D. I triggered a failure
Exam – Part 2

Example:

Assume that a list of requirements contains the following two user stories:
• (US1) As a customer, I want to narrow down hotel search results by country, so I can find the right hotel more quickly.
• (US2) As a hotel receptionist, I want to see the list of rooms not yet booked

Answer the following questions Q1 to Q3:
Q1: Which of the two user stories is of better quality? Justify your answer.
Q2: What are the three elements of a complete user story? Explain each part briefly
Q3: What is the purpose of the ‘why’-part in a user story? For whom and why is it useful?
Exam – Part 3

Example:

- The code snippet below shows a function that calculates fines for speeding when driving a car. Fines are calculated based on age of the driver (age), the excess speed (overspeed), and the number of penalty points (penaltypts) already accumulated in the driver’s police record.

```java
1 public static int speedingfine (int age, int overspeed; int penaltypts) {
2     int fine = 0;
3     if ((age >= 25) && (overspeed < 30) && (penaltypts < 3))
4         fine = fine + 100 * overspeed;
5     if ((age < 25) || (penaltypts >= 3))
6         fine = fine + (200 * overspeed);
7     if (overspeed >= 30)
8         fine = fine + 5000;
9     return fine; }
```

- To do: Provide a minimum set of test cases that achieves 100% statement coverage. Remember that complete test cases include both input values and expected output values. (Note: You will get a penalty, if the number of test cases is not the minimum)
Exam – Part 3 Remarks

In the old exams, you find this:
• Defect estimation using Capture-Recapture Models
• Effort estimation with Use Case Points

Not covered this year
→
Not in exam!
How to prepare for the exam?

- Work through old exams
- Prepare a cheat sheet

- Do not expect that you have enough time DURING the exam to search for answers in the internet!

- Bring your computer to exam
- Bring pen (and replacement pen) to exam
Training Quiz

(Exam Part 1)
Question 1

Which of the following statements does not explain why requirements engineering is difficult?

Answer choice:
1. Because no UML notation exists for describing requirements
2. Because requirements can change over time
3. Because it is difficult to identify the relevant stakeholders, and, once identified, the stakeholders have difficulties describing what they want/need
4. Because stakeholders have conflicting requirements
Question 2

• What are User Story Points not used for?

Answer choice:
1. To estimate project effort
2. To estimate project quality
3. To estimate project duration
4. To estimate team velocity
Question 3

- Which of the following statements about use case descriptions is not correct?

Answer choice:
1. A participating actor is an actor who helps achieve the goals of the initiating actor
2. An alternate flow describes exceptions from or extensions to the normal interaction scenario
3. Preconditions describe the state of the system before the start of the interaction scenario
4. A use case diagram is a graphical representation of a use case description
Question 4

• Which of the following statements about code refactoring is correct?

Answer choice:
1. Refactoring does always change the program design
2. Refactoring is done in order to add new functionality
3. Refactoring is a test activity
4. Before refactoring a component, a test suite for this component must be in place
Question 5

Which of the following statements is correct?

Answer choice:
1. Regression testing can efficiently be done manually
2. Equivalence class partitioning is a white-box testing technique
3. Black-box testing techniques exploit knowledge about the code that is tested
4. White-box testing techniques exploit knowledge about the code that is tested
Question 6

- You execute a calculation program and see the following message on the screen: ’Program aborted due to division by 0’. Which of the following describes your experience best?

Answer choice:
1. I made an error (while using the program)
2. I debugged an error
3. I triggered a failure
4. I localized a fault
Question 7

Which of the following types of meetings is not defined by SCRUM?

Answer choice:
1. SCRUM Master Meeting
2. Sprint Retrospective
3. Daily Scrum (or ‘Stand-Up Meeting’)
4. Sprint Planning Meeting
Question 8

• If three developers work two days each, how much effort have they spent?

Answer choice:
1. 6 person-days
2. 6 days
3. 2 person-days
4. None of the above
Prepare for Exam !!!!
Next Lecture

• Date/Time:
  • No more lectures

• For you to do:
  • Submit Lab Task 7 in time!
  • Prepare for exam
Thanks for coming to the lectures!

All the best for the exam!