Linköpings Universitet
IDA
Kristian Sandahl

2017-01-07

Written exam for Software Engineering Theory

Course codes TDDC88, TDDC93, 725G64

Note: When I visit the exam, I will take a slow walk among all students, so you don't need to sit with your hand raised. Just call for my attention when I pass your desk.

Instructions to students, please read carefully

- **Explicitly forbidden aids:** Textbooks, machine-written pages, photocopied pages, pages of different format than A4, electronic equipment.
- Try to solve as many problems as possible.
- Motivate all solutions.
- Please, write and draw clearly.
- Write solutions for different areas (fundamental part) and different problems (advanced part) on separate sheets of paper.
- Label all papers with AID-number, date of examination, course code, examination code, and page number.
- You may write solutions in either Swedish or English.
- Please, note that the problems are not necessarily written in order of difficulty.
- TIP! Read through all exercises in the beginning of the exam. This will give you the possibility to ask questions about all parts of the exam, since the examiner will visit you in the beginning of the exam time.

Grading

The exam consists of two parts: Fundamental and Advanced.

The Fundamental part has problems worth 10 credits per area. Areas are: Requirements, Planning & Processes, Design & Architecture, Testing & SCM, and Software Quality. Thus the Fundamental part can give maximally 50 credits.

The Advanced part has problems worth 50 credits in total. Each problem typically requires a longer solution of several pages.

The maximum number of credits assigned to each problem is given within parentheses at the end of the last paragraph of the problem.

Pass condition: At least 4 credits per area in the Fundamental part **and** at least 50 credits in total. The total amount of credits also includes the bonus credits you might have got in lecture exercises autumn 2016. This gives you the mark 3. If you have at least 4 credits for 4 of the areas in the Fundamental part, then you can still pass if you have more than 60 credits in total.

Higher marks are given based on fulfilled *pass condition* **and** higher amounts of credits according to the following table:

Total credits	Mark
0-49	U (no pass)
50-66	3
67-83	4
84-	5

Multiple choice questions

In multiple choice questions we will ask you to write down the letters A, B, C, or D for the one or two statements that you think are true. Note that you should not write down the statements that you think are false. There are exactly two true statements per question, so answering with three or four alternatives with gives 0 credits.

For each statement that you select that is correct (i.e., that the statement is in fact true) you get one credit. For each statement that you select that is incorrect (i.e., that the statement is in fact false, but you believed it was true) you get minus one credit. Each multiple choice question can give maximum 2 credits and minimum 0 credits, i.e., you cannot get negative credits for one multiple choice question.

Example 1: Assume that you have written down statements A and C. If now statements A and B were true, and statements C and D were false, you would get +1 credit for writing down A, but -1 credit for writing down C. Hence, the total credits for the multiple choice question is 0.

Example 2: Assume that you have written down statement B. If now statement A and B were true, and statement and statement C and D were false, you would get +1 credit for the multiple choice question.

Example 3: Assume you correctly wrote both statement A and B. If now statement A and B were true, and statement and C and D were false, you would get +1 credit for writing down A, and +1 for writing down B. Hence, the total credits for the multiple choice question is 2.

Good Luck!

Kristian

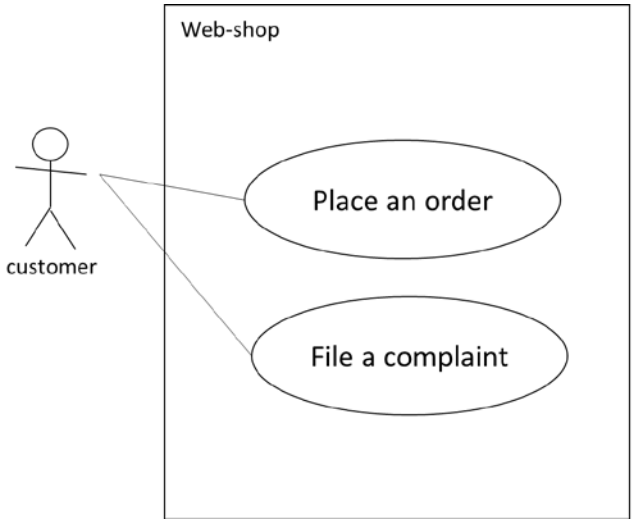
Problems

Part 1: Fundamental

Area 1: Requirements

1 a) Look at the UML use-case diagram below. Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. The rectangle named Web-shop is a class that will contain all methods to realize the use-cases.
B. From the diagram we know that the actor can initiate interaction with the system in order to fulfill the use case “Place an order”.
C. To determine whether the use case “Place an order” is fulfilled we need more information, not shown in the diagram.
D. The lines between actor and use-cases have a non-sharing property, which means that a new actor, purchaseResponsible, could not share the “Place an order” use-case with the customer actor



1 b) Scenario: You will design a management system for a car pool. The members of the car pool can sign up for two types of memberships, regular and frequent, with different tariffs for fixed cost and mileage cost. At the member portal, the members can book a car, change bookings, and change personal information.

Task: Write *two functional* and *two non-functional requirements* of the system. (4)

1 c) Explain the following concepts in the context of software requirements: *Traceability*, *stakeholder*, *process requirement*, and *user story*. You may use an example. (4)

Area 2: Planning and Processes

2 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. A *project* can have several goals, but a *process* can only have one goal.
B. A *project* occurs once, but a *process* can occur several times.
C. A *toll-gate* must always be preceded by a *mile-stone*.
D. It is fully possible to assign two or more people to a single activity in a project plan.

2b) Shortly describe each step of the *risk management process*. 1-2 sentences per step is probably enough. (4)

2c) Describe one advantage and one disadvantage of having very short (1 week) *iterations* in *iterative development*. Also describe one advantage and one disadvantage of having very long *iterations* (12 weeks) in *iterative developme*nt. (4)

Area 3: Design and Architecture

3 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. A *UML class diagram* shows how *objects* can be *instantiated*.
B. The *generalization* relation in a *UML class diagram* is used to describe *inheritance*.
C. An *abstract class* in UML can only have *abstract operations*.
D. By default, an *association* from class A to class B, means that *instances* of B visible from A are *ordered* and *non-unique*.

3 b). Describe two expected benefits from designing and documenting a *software architecture*. (4)

3 c) Draw a *UML state diagram* for the class car in a car pool management system. Use at least 4 *states* apart from start and stop *pseudo states*. The main idea behind a car pool can be found in the scenario of problem **1 b)**. (4)

Area 4: Testing and SCM

4 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. A good thing about the *top-down integration testing* strategy is that you most likely will test the *user interface* quite early.
B. A potential problem with the *top-down integration testing* strategy is that lower level functions are often trivial, off-the shelf software.
C. A good thing about the *big-bang integration testing* strategy is that you don't have to spend time writing *drivers* and *stubs*.
D. A potential problem with the *bottom-up integration testing* strategy is that *stubs* can have very many conditions.

4 b) The Human Resource (HR) department of the Royal Opera asked you to do a *boundary value analysis testing* of their system for retirement offerings. The software sends retirement information to dancers in the year when they will become 41 years old, to singers when they will be 52 years old, and to the rest of the personnel when they become 65 years old. It is assumed that you can read the birthdate from the personnel database and that you can access the current year from the computer clock. Answer with your *test table* for *boundary value analysis testing*. (4)

4 c) Describe the following concepts in the context of software configuration management: *Baseline*, *trunk*, *pull request*, and *regression test*. (4)

Area 5: Software Quality

5 a) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. One of the most important parts of the CMMI *process area Technical Solution (TS)* is to develop and evaluate different approaches to the design of the product.
B. One of the most important parts of the CMMI *process area Requirements Management (REQM)* is to *elicit* the true needs of all *stakeholders*.
C. *Software review* is a technique that fits well in accomplishing the specific goals of the CMMI *process area Verification (VER).*
D. You have reached a CMMI *maturity level* 3, if you fulfil more than half of the *process areas* on that *level*.

5 b) Describe the four types of *Software reviews* in terms of the goal and who is performing the review. (4)

5 c) Describe two different *software metrics* that can be used to predict the *maintainability* of a software system. Don't forget to motivate why the *metric* can be a good predictor. (4)

Part 2: Advanced

6. You are testing a function `nextDate2017(x: Date)` that takes a date of 2017 as input on the form 2017-mm-dd, and outputs the date of the day following immediately after x. Your task is to identify *equivalence classes* of the input and write 10 *test-cases* based on *equivalence class testing*. If you find more than 10 test-cases, just select 10 of those. (10)

7. Below you will find a list of concepts used in SCRUM. For each of the concepts, write down: a definition, expected benefits, potential problems, and a motivation for why or why not the concept can be beneficial to use also in a project using the *classical Water-fall model*. (20)

a) Sprint
b) Product owner
c) Product backlog
d) Burn-down chart
e) Retrospective meeting

8. Context: A super-market self-checkout system allows the customer to scan items, weigh loose-weight good (e.g. fruit), scan discount coupons, register customer card, and pay with card or cash. It is also possible to press a help button. The machine selects by random a few customers per day, that also have to make a manual checkout in order to prevent frauds.

Task: Draw a *UML Sequence diagram* of the interaction between the user and the system. Use at least 3 *nodes*, 12 *messages*, and 2 *fragments*. (10)

9. Write a short analysis of *UML Sequence diagrams*. Under which circumstances are they good in describing system dynamics? What kind of system dynamics is less fit for the UML Sequence diagrams? What are benefits and challenges from the perspective of:

- the designer who is drawing the diagram,
- the reviewer who is understanding and checking the diagram, and
- the programmer implementing the functions of the diagram.

Are there alternative ways you can use to describe behavior in problem 8. above? You may refer to the solution of problem 8. if you need an example. (10).