Linköpings Universitet
IDA
Kristian Sandahl

2021-10-27

Written exam for Software Engineering Theory

Course codes TDDC88, TDDC93, 725G64

Note: When we visit the exam, I and/ or Lena will pass by all students, so you don't need to sit with your hand raised. Just call for our attention when we pass your desk.

Instructions to students, please read carefully

- **Explicitly forbidden aids:** Textbooks, machine-written pages, photocopied pages, pages of different format than A4, other electronic equipment than the computer for running Wiseflow. A silent keyboard, mouse and a webcam might be brought.
- Try to solve as many problems as possible.
- Motivate all solutions.
- Please, draw clearly.
- You may write solutions in either Swedish or English, but don't mix in the solution of the same problem.
- Please, note that the problems are not necessarily written in order of difficulty.
- TIP! Read 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 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 2021. 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 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 a maximum of 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 statements C and D were false, you would get +1 credit for the multiple-choice question.

Example 3: Assume you correctly wrote both statements A and B. If now statements A and B were true, and statements 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) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. A requirement is said to be *ambiguous* if it cannot be interpreted in more than one way.
B. A requirement is said to be *traceable* if it is easy to find the software components that realize the requirements.
C. Requirements analysis involves *conceptual modelling*, for instance writing *use-cases*.
D. Requirements *validation* is achieved when all requirements are correctly spelled and made up by complete sentences.

1 b) *Scenario:* In a social media system users can upload pictures, short movies, and sound files. There are features of basic editing of the media files and sharing and commenting functions. There are personal user pages where you can show examples of your work and describe ambition level and dreams when it comes to media production. It's possible to submit files to panels of well-renowned members that can give you constructive criticism on how to improve.

Task: Write a *use-case* of the system described above and draw a *UML use-case diagram*. Write two *user stories* of the above system where at least one is related to the use-case. (4)

Take a photo of the diagram and add that as an appendix to the answer. Let the name of the appendix start with the characters "1b".

1 c) For each of the requirements R1 and R2 below, describe one good property that the requirement has and one good property that it is lacking.

R1: The processing time of an average customer registration shall be optimal.
R2: The customer shall be able to login with a Facebook-ID and then be able to listen to music from the personal playlist. (4)

Area 2: Design and Architecture

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

A. An *implementation (code) view* of a system can be illustrated with a *UML package diagram*.
B. An *execution view* of a system shows which source code *artifacts* that shall be compiled together.
C. A *deployment view* of a system shows on which hardware different components shall execute.
D. An *implementation (code) view* of a system is used to analyze the need of bandwidth and processing power for a *client-server architecture.*

2 b) Describe two benefits of using a *layered architectural style*. A benefit is documented with:

- one sentence description of the benefit
- a quality factor that is improved by the benefit
- a motivation of why the quality factor will be improved as a result of using the *layered architecture*. (4)

2 c) Draw a *UML sequence diagram* that shows an *ALT fragment*. It does not need to be a large, complicated diagram. The important thing is that you demonstrate that you understand what an ALT fragment is. The syntax must be correct and the semantics reasonable. (4)

Take a photo of the diagram and add that as an appendix to the answer. Let the name of the appendix start with the characters “2c”.

Area 3: Testing and SCM

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

3 b) The 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, to singers when they will be 52, and to the rest of the personnel when they become 65. It is assumed that you can read the position and 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 testing. (4)

3 c) Describe three activities, automated or manual, that are performed in *Continuous Delivery*. What is the difference between *Continuous Delivery* and *Continuous Deployment*? (4)

Area 4: Planning and Processes

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

A. The *buffer time* is the difference between *internal* and *external deadlines*.
B. A *tollgate* is described by activities that has to be finished before a certain date.
C. The *critical path* in a Gantt-chart contains activities performed by the most expensive resources.
D. A consequence of *time-boxing* can be that you need to postpone the implementation of low-priority functions.

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

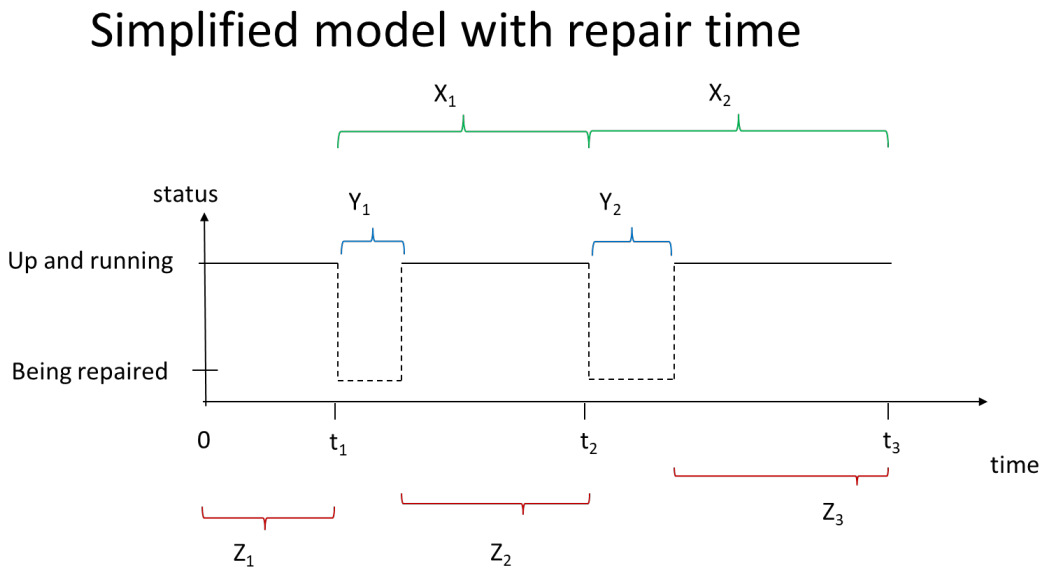
4 c) Describe two advantages and two disadvantages of moving from the *classical waterfall* model to an *iterative* model. (4)

Area 5: Software Quality

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

Consider the failure model below where failures occur at time t_1, t_2,.... Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. The mean of all X's can be used (together with other metrics) to estimate *availability*.
B. A high value of the mean of all X's indicates high *reliability*.
C. The mean of all Y's is called *Mean Time To Failure*.
D. The mean of all Z's is called *Mean Time Between Failures*.



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

5 c) Scenario: You have a fairly large development company and you have a good reputation when it comes understand the customers' true needs and you deliver fast. However, you need to spend too much time in repairing defects of the products, since your reliability of the products vary a lot.

Task: Describe a *CMMI process area* that you think would help you to deliver less faulty software. Summarize the process area's *purpose*, *specific goals*, and *introductory notes* in 5-6 sentences and motivate how this could help you. (4)

Part 2: Advanced

6. *Scenario:* Your company of 10 skilled software engineers has been assigned to develop a safety critical software in a medical equipment. The calendar time for the development is not too short, but there is no room for unnecessary delays. After the start in January 2022 you are supposed to deliver in June 2022. Hence, you have started making preparations by:

- Sending one of your developers to a course in formal methods for requirements engineering.
- Placing an order for a logic inference tool to help you proving the consistency and correctness of your requirements.
- Outsourcing the coding to a company in India that is evaluated on CMMI level 5
- Contracting a medical expert of 100 hours to be used in different types of review and testing.

Task: Identify two *project-specific* risks with high *risk magnitude*. Create four different *plans* per risk: *avoidance*, *transfer*, *mitigation*, and a *contingency plan*. The risks shall use information from the scenario. You can add your own assumptions about the project if you write them down.
(10)

7. *SCRUM* and *Kanban* are popular frameworks for organizing development teams in software engineering. SCRUM is more prescriptive than Kanban since there are more practices, roles, and artifacts mandated by SCRUM. Select five concepts that are prescribed by SCRUM, but not in Kanban. For each of the concepts, write down

a) A description of the concept
b) A motivation of why the concept is *agile*
c) A motivation of whether the concept could be used by a team using Kanban
d) A motivation of whether the concept could be used by a team using the *classical waterfall* model.

A description typically has one sentence, motivations 2-4 sentences.
(20)

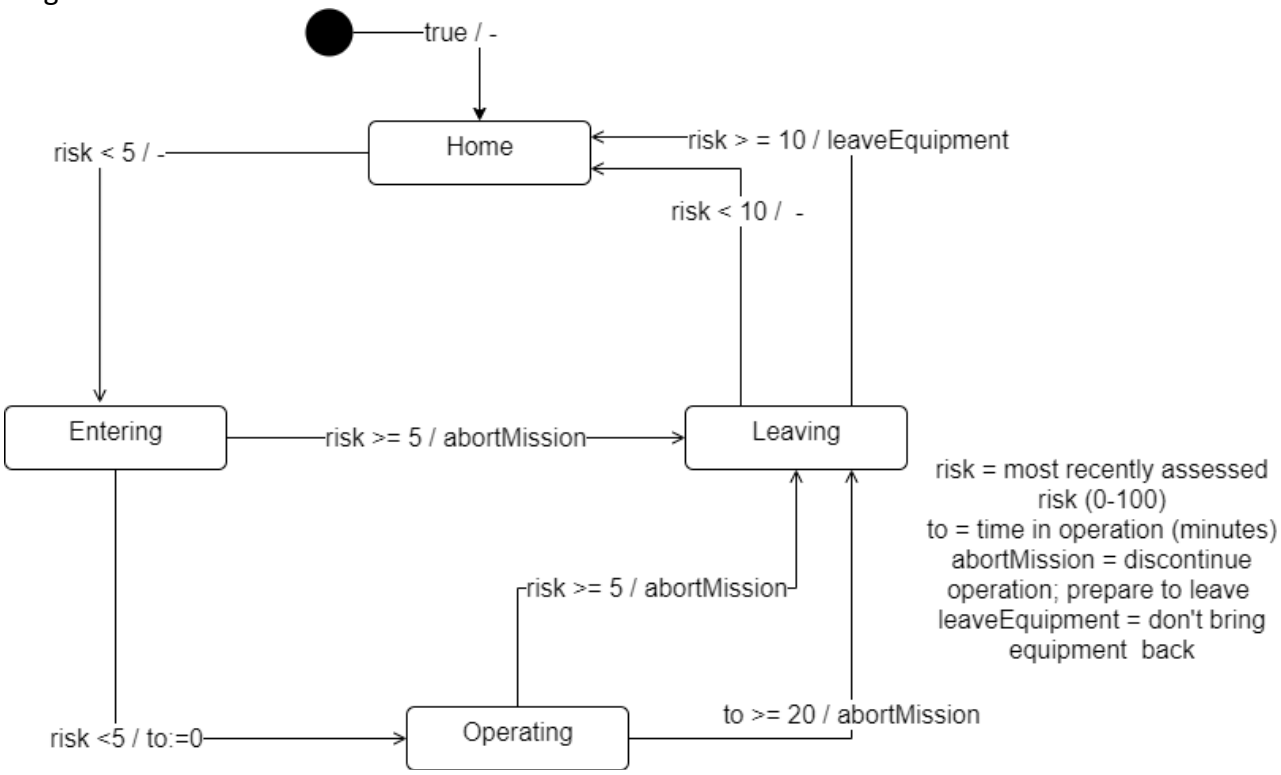
8. *Scenario*: You are developing a risk management system for missions in dangerous zones in catastrophic areas and your goal is to assess the aggregated risk level (0-100) in a zone with three different means:
1. You send up a drone to measure different properties from the air and calculate a risk level from the measurements.
2. You drop a swarm of several simple and expendable measurement and communication units and calculate the risk level from these measurements.
3. You drop an expensive robot with advanced sensors and capability to lower the risk. Likewise, a risk level is calculated.

It is important to be able to shift the source of the risk level. For instance, you might want to send in a swarm first before sending the expensive robot.

Task: Draw a *UML class diagram* of the concepts involved. You only need to have the most important attributes and methods of the classes. Use the *Strategy design* patterns for full credits. If you don't know the Strategy design pattern, make a sensible diagram, you will get credits for such model too. (10)

Take a photo of the diagram and add that as an appendix to the answer. Let the name of the appendix start with the characters "8".

9. *Scenario*: Same as in problem 8, but with the extension that you can send in a human operator into the zone. The states of the class human are shown in the state diagram below.



Task: Create a *test table* where the union of the *test cases* exercises all *transitions* shown in the diagram at least once. Besides the usual properties of a test case, the test case shall specify which state(s) that are covered. Also include current state (before the execution of the test case) and next state (the state that is reached after the execution of the test case) for each test case in the table. (10)