Linköpings Universitet 2018-10-24
IDA
Kristian Sandahl

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 2018. 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) Which of the following statements are true? Answer with the statement letter only, no motivation is needed. (2)

A. We say that two requirements are *testable* if, and only if, they can always be satisfied simultaneously.
B. *Prototyping* can be used both for *requirements elicitation* and *requirements validation*.
C. An *Entity-Relationship diagram* is useful when we want to describe the dynamic behaviour of an embedded control system.
D. The IEEE Standard 830 for Software Requirements Specification encourages that you to adapt the disposition of headlines to your particular application.

1 b) *Scenario:* Your hairdresser salon has started a web-site for frequent customers. If you have signed up for the free membership you can login and book treatments. You can search for different employees to perform the treatment, date, or time. The customer can erase bookings if more than 24 hours remain before the start of the treatment. The web-site has also an interface for the hairdressers to use for management of the salon, for instance editing their working hours, customer bookings, and times for serving drop-in customers. The hairdressers can also edit advertisements from their vendors.

Task 1: Now, create a UML *use-case diagram* of the web-site consisting of two different *actors* and two different *use-cases*. Don’t forget the use case descriptions (texts). Only logging in and logging out are basic functions, not to be considered as individual use-cases. (4)

Task 2: Write two *functional* and two *non-functional* requirements of the web-site. (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. The process in SCRUM is the main focus, so the *sprint retrospective* continues as long as is necessary even if it delays the *sprint planning*.
B. A *task board* in SCRUM is an organization that sets priorities on changed and new *product backlog items*.
C. SCRUM can be used for many different types of projects, not only software development.
D. The *team* is a role in SCRUM.

2b) Shortly describe the following concepts in planning with a *GANTT-chart*: *phase, slack time* (also called *float time*), *critical path*, and *mile-stone*.

Hint: a short description is typically 1-2 sentences. (4)

2c) The following practices are popular in *agile* methods:

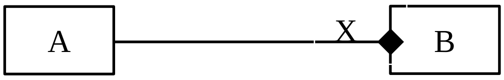
- *Test-driven development*
- *Pair programming*
- *User stories*
- *Refactoring*

Can any of these be used in a project run according to the *classical waterfall* model? For each of the practices give a short motivation for why or why not. (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. In a *UML state diagram* the *transitions* are denoted *event/action,* where *action* is the functions to execute during transition, and *event* is the underlying cause for the transition.
B. An advantage of a *pipe-and-filter architecture style* is that the resulting code has no redundancy.
C. The *Observer design pattern* is a good choice when a change to one object requires changing others.
D. In a *UML class diagram* using a *composition relation,* the multiplicity X can only be 1.



3 b) Define and describe the concepts of *coupling* and *cohesion* in software architecture. For each of the concepts, describe what you should strive for in software architecture design. Don’t forget to motivate your answer. (4)

3 c) Draw a *UML state diagram* of a telephone answering service with the following properties:

1. If the receiving subscriber does not answer after 4 signals, the answering service starts.
2. If the receiving phone is busy, the answering service starts immediately.
3. The answering service plays a message and accepts voice input from the calling subscriber.
4. If the receiving subscriber answers the phone when the answering service is working, the service is cancelled, started recordings deleted, and the conversation can start as a normal phone call.

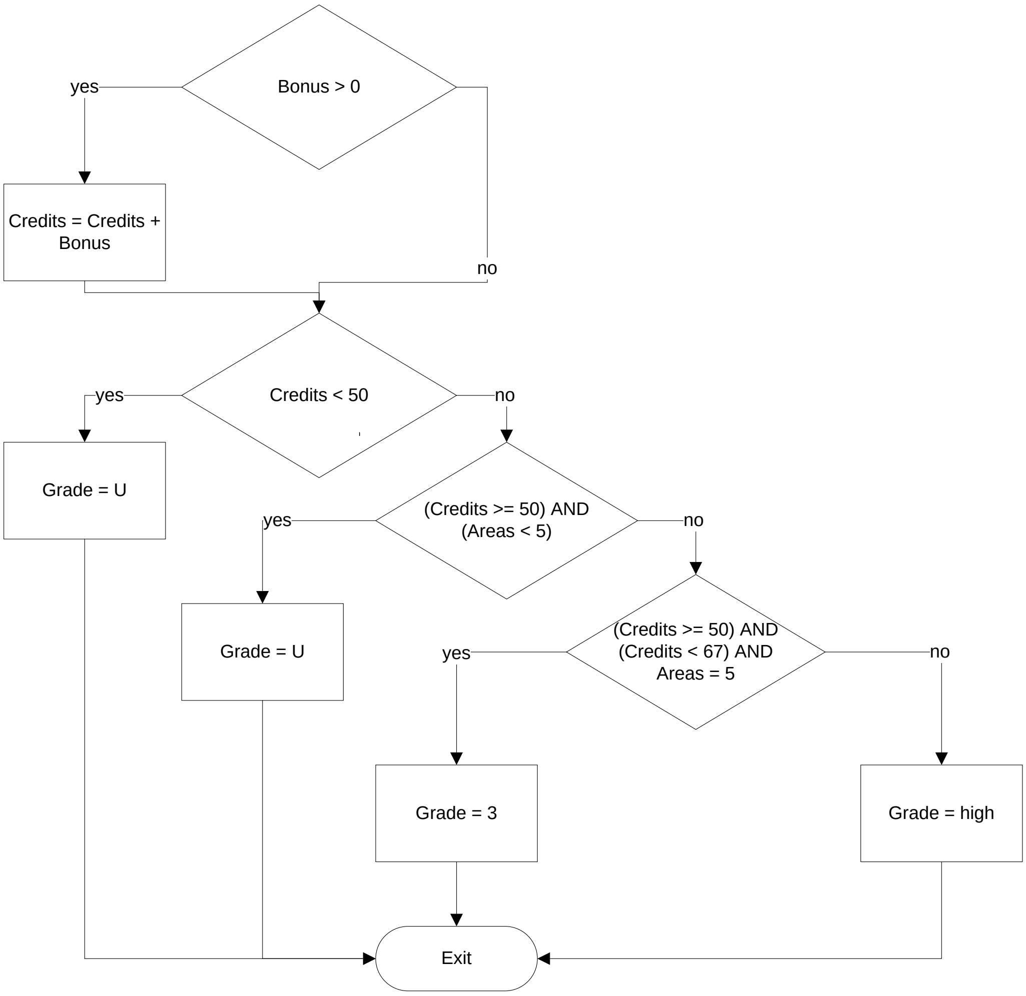
(4)

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. In the software engineering terminology an *error* is a human mistake leading to a *fault* in the program, which if executed can cause a *failure*.
B. *Usability testing* very often involves using a panel of test users performing representative tasks of the system.
C. If you want to achieve *branch coverage testing* of a program you can be satisfied with a smaller number of test cases as you would need if you require *statement coverage testing*.
D. A drawback of *Equivalence class testing* is that only valid input is tested.

4 b) For the *flowgraph* below write a set of test cases that ensures *branch coverage* with a minimum number of test cases.


Bonus > 0
yes
no
Credits = Credits + Bonus
Credits < 50
yes
no
Grade = U
(Credits >= 50) AND (Areas < 5)
yes
no
Grade = U
(Credits >= 50) AND (Credits < 67) AND Areas = 5
yes
no
Grade = 3
Grade = high
Exit

(4)

4 c) Provide the following about the concept *continuous delivery*:

- A definition
- Functions needed in an automated environment
- One advantage, and
- One potential problem.

(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. In object-oriented software, a high number of the maximum depth in the *inheritance tree* indicates that the software is harder to understand.
B. A high number of *reused* code lines can be an indicator of high *reliability* with the argument that the reused parts have been tested and run for a longer time than newly produced code.
C. If we draw a *flow-graph* of a program, that only contains *binary decisions*, then the *cyclomatic complexity*, V(G), of the program grows quadratically with the number of decisions.
D. The *reliability* can be approximated by MTBF/(1-MTBF), where MTBF denotes Mean Time Between Failures.

5 b) Compare the review methods *Inspection* and *Walk-through*[1] in terms of:

- Goal
- Participants
- Process
- Data collected

Hint: Make a table with the methods as columns and comparison criteria as rows. Write 1-2 sentences in each cell. (4)

5 c) Describe the following concepts of the CMMI[2]: *Maturity level, process area, specific goal,* and *specific practices.* (4)

[1] In older literature called: Structured walk-through

[2] The one we used in the course: CMMI for Development v 1.3, staged representation

Part 2: Advanced

6. Important concepts in SCRUM are:

- *Product owner*
- *Team*
- *Sprint*
- *Daily meeting*
- *Retrospective meeting*
- *Sprint review meeting*
- *Burn-down chart*

Select five of the concepts above that you can apply to *mitigate risks* in SCRUM projects. Write a motivation of why and how the concept can contribute to the risk mitigation (1-2 sentences for each of the concepts selected). (10)

7. *Scenario:* You are developing a system consisting of an app, an administrator interface, and a database. The app is used by students, employees and visitors to find their way on LiU’s four campuses. The app is run on a mobile device. The interface for the administrator is to be run on a personal computer. The administrator handles all input and changes to the database. You decide to develop the system in four iterations:

I. A skeleton system with login, security, networking and a fixed map over one campus as the only view in mobile units. A text-based interface for the administrator module.

II. A functional system for the mobile units with interactive maps of all campuses. The administrator gets functions for batch input and editing to populate the database.

III. A user-experience rich system where interfaces get color schemes and visual interactions with gestures. The search engine and a database for one campus are ready for field trial.

IV. A fully integrated system with all maps. It’s possible to connect to positioning of mobile devices and tag places. The administrator is connected to an event calendar and the technical system of Akademiska hus[3].

Task: Construct and describe four different *usability metrics* that can be used in the four iterations above. The metrics are described in terms of

a) A name and a short description.
b) How to obtain the metric.
c) How to calculate the numerical value.
d) Argumentation for why the metric contributes to measuring the usability of the system.
e) A decision of change you can take given the information by the metric. (“Continue as planned” is not an option here.)

[3] The organization responsible for construction, operation, and maintenance of buildings on the campuses.

8. *Scenario:* The trains in Sweden must be on time. Therefore you have started developing a passenger delay minimization system. The idea is to keep track of the passengers with tickets in their mobile phones. When a train is approaching the station, passengers are reminded to go to the platform. Passengers get reminders at different times depending on i) their distance to the platform and ii) a prognosis of queues to stairs and elevators; we want a constant flow of people moving.

At the platform the passengers get information about where to wait for boarding the train. The system knows which door of the train that is best for each passenger. Passengers can either get a map on the screen or be guided with a clicking sound, where the frequency becomes higher the closer they are to the correct boarding position.

If passengers send a "need assistance" message to the server, the train crew and the station hosts gets a list on their mobile units of people that need boarding assistance. The people needing assistance are prioritized depending on how far they are from their boarding position. A station host or a crew member can select a person to help, then the other personnel do not need to bother about this person. When done, the helper checks the passenger on the mobile device as ready, and a confirmation is sent to the chief crew member. The chief crew member has a special unit where he/she can observe the status of all passengers in need of assistance.

For safety reasons, you don't want people to run around when the train is in motion at the platform. So, you implement a safety controlling software according to the following specification: 5 seconds before the train reaches the first end of the platform, all information about boarding positions are erased from passengers' phones, and the text "Wait until train is stopped" is displayed. The boarding information appears again when the train has been standing still for 2 seconds.

Task: Draw a *UML sequence diagram* for the *messages* between the hardware nodes. Only passengers with a pre-booked assistance need are allowed to send "need assistance" messages. Use at least one *fragment*. (10)

9. *Scenario*: Same as in problem 8.

Task: Write *test-cases* for the software controlling the safety software. Identify *input* and *output variables* and equivalence classes. The test cases shall fulfill the *boundary value testing* as is defined in this course. (10)