Linköpings Universitet
IDA
Kristian Sandahl

Written exam for Software Engineering Theory

Course codes TDDC88, TDDC93, 725G64

Note: This occasion has a schedule collision with a department activity. During the exam a student assistant will visit you with a mobile phone where you can talk to the examiner. As always it is possible to as the exam vigilator to call the examiner if you want to ask a question.

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 2013. 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. *Requirements elicitation* is the process of formulating known requirements in a mathematical notation.
B. Two requirements that do not contradict each other are said to be *traceable*.
C. *Entity-relationship modelling* is a kind of *conceptual modelling*.
D. *Unambiguous* requirements can only be interpreted in the intended way by the developers.

1 b) *Scenario:* A modern library system contains catalogues of books and journals. It can also give the users access to on-line information databases. In addition to this, the library provides citation handling systems for users writing reports. There is a member management system recording data about the members including volumes that are on loan or reservations made. Members can also be suspended if they don't follow the rules or damage library property. The librarians can also use external services, such as, other libraries or book vendors.

Task: Create a *use-case diagram* for a new library system consisting of two different actors and two different use-cases. Don't forget the use case texts. Only logging in and logging out are basic functions, not to be considered as use-cases. (4)

1 c) Write down two *functional requirements* of the library system described in the scenario of problem **1 b)** that can occur in a *software requirements specification*. Also write down a *quality requirement* and a *design constraint*. It is important that all requirements are *testable*. (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. Developing software according to the *waterfall model* makes it unnecessary to handle *risks*.
B. Developing software according to the *waterfall model* makes the project fit well to common project management practices.
C. Developing software according to an *iterative model* is well suited for fixed-price contracts.
D. Developing software according to an *iterative model* facilitates *continuous process improvement*.

2b) Describe the following concepts from SCRUM: *The sprint planning meeting, the daily SCRUM meeting, the burn-down chart,* and *the product owner.* (4)

2c) Explain two reasons for *Brook's law*: "Assigning more programmers to a project running behind schedule will make it even later..." (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 *three-tier client-server architecture* supports the use of intelligent load balancing
B. A *layered* architecture supports *incremental* development and testing
C. *Layer bridging* is a concept of *pipe-and-filter* architectures which means that some data can take a short-cut by passing a series of *filters* in a *pipe*.
D. In a *two-tier, fat-client architecture* the *presentation layer* is on the server-side, which sends ready-made displays to the *clients*.

3 b) Define the concept of *coupling* in the context of software architecture. Shall we strive for high or low coupling in software design? Don't forget to motivate you answer. (4)

3 c) Draw a *UML state chart* of a library book that can be in the shelf, on loan, reserved, or removed from the library by the librarian. Don't forget to write clear and logical information on the *transitions*.
(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 *baseline* is a configuration of the *base classes* in an object-oriented program.
B. A *development branch* is a sub-committee of the *change control board*.
C. When two persons do a *check-out* they both create their own copy of the file.
D. A c*onfiguration item* can be other important artifacts, not just source code.

4 b) *Scenario:* You are developing a system checking whether goods can be sent as a Swedish domestic letter. The rules are:

Minimal sizes

90 x 140 mm

Cylinder: Length: 100 mm, length + 2 x diameter: 170 mm

Maximal sizes

Length: 600 mm
Length + width + thickness: 900 mm

Cylinder: length 900 mm, length + 2 x diameter: 1040 mm

Task: Your task is to identify at least 4 *equivalence classes* and provide one *test case* per equivalence class for your software. (4)

4c) Describe the concepts from system level testing: *Function test, performance test, acceptance test,* and *installation test*. (4)

Area 5: Software Quality

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

A. The *author* in an *inspection* meeting must only listen even if he detects defects.
B. An *audit* is a *software review* with no written documents; the participants just talk about the ideas of the software.
C. You normally record data about the inspection itself in order to improve the expensive inspection process.
D. Both the *author* and the *inspection leader* take part in the *exit-and-follow-up phase* of the inspection process.

5 b) Suggest two different *software metrics* for measuring the *maintainability* of software. Don't forget to motivate why the metrics can give information of the maintainability. (4)

5 c) Describe the following concepts from the CMMI model: *Maturity level, process area, specific goal, and required component*. (4)

Part 2: Advanced

6. *Scenario:* Suppose that you've got the job to develop a new student portal for Linköping University. The functional requirements are the same as the old one, but it must be possible to be access the portal from a variety of mobile platforms. The requirements on usability and reliability are high. You and your four team members start 1 September 2014 and start *parallel testing* 10 January 2015.

Task: Make a list of five relevant *risks* that need to be monitored. Use your risks to demonstrate how you can calculate the *risk magnitude indicator*. Also make a *plan* for each of the risks. Select your risks so that you can give examples of four different types of risk planning: *Risk avoidance, risk transfer, risk mitigation,* and *definition of a contingency plan*.
Hint: You don't have to give examples of all types of risk planning for each of the risks. It is sufficient if all types of risk planning occur in your entire solution. So risk no 1 can have risk avoidance, risk no 2 risk has risk transfer, etc.
(10)

7. Describe four different actions you can take to ensure that your product meets high requirements on *usability*. These actions can be using a method, following a process, organizing the development, using a tool, selecting a design, or anything that you can make use of in a project. For each of the actions describe:

a) how the action contributes to high usability.
b) a way of how the outcome might be controlled with a *metric*.
c) a related *quality factor* that might either be strengthened or challenged by the action. (20)

8. *Scenario:*You have been asked to create a web-based election advice system (valkompass) for the general public. The system presents 25 different political statements where you are asked if you agree or not on a 5-grade scale. For each statement you also indicate how important it is to you on a 5-grade scale. Moreover you will be asked to rate you confidence in the different party leaders, and if there are any parties you are not interested in. Before the system is published the different candidates for the parliament have been asked to input their agreement and importance for the 25 statements. This information resides in a database, and an intelligent algorithm matches your input with the candidates'.

All candidates belong to a specific political party. In Sweden you vote for a party that has nominated its candidates. Each party prints a ballot where he names of the candidates are listed in order of preference from the party. If you have a favorite candidate who is not number 1 in the list, you can mark that person with a pen. If many people do the same, that person can become number 1.[1]

As output you can get: a list of how well you comply with each party in average, a list of the candidates with most similar result as yours, and statement-per-statement comparison between you and a particular candidate. Each candidate has also provided a little personal presentation and a link to his/her homepage. This information can be accessed from the system, by clicking the candidate's name.

The purpose is serious so there are high requirements on *information security*, and ease of use.

Tasks:

a) Draw a *UML class diagram* modelling the different concepts handled by the system.
   Hint: be sure to write *association* names if you use plain associations or associations with *navigability*. Special sematic arrow-head association, for instance, *generalization* do not need association names. (10)
b) Draw a *UML sequence diagram* of the interaction between the user and the application.
   Hint: There are only two *roles*, correctly modelling the interaction is the focus of the problem. (10)
c) Specify the intelligent algorithm on a high level. Focus on the order in which you intend to use which information. How would you like the system to handle trade-offs, for instance, if you agree well with candidates whose party leader you have low confidence in? The reader of the specification is an expert programmer, who just needs some directions on a high level.
   Hint: Use a flow chart to describe the data flow. It doesn't have to be UML or pseudocode. Your solution will be judged in your ability to identify important parts and the clarity of your instructions (5)
d) Identify a function of the system and create a table of 5 *test-cases* for that function. The test-cases are intended for use in test-driven design. (5)

[1] There are of course more peculiarities, but they are outside the scope of this scenario.