Försättsblad till skriftlig tentamen vid Linköpings Universitet

Datum för tentamen	2011-10-22
Sal (2) Om tentan går i flera salar ska du bifoga ett försättsblad till varje sal och ringa in vilken sal som avses	TER1 TER2
Tid	14-18
Kurskod	TDDC88
Provkod	TEN1
Kursnamn/benämning **Provnamn/benämning**	Programutvecklingsmetodik, Tentamen
Institution	IDA
Antal uppgifter som ingår i tentamen	7
Jour/Kursansvarig Ange vem som besöker salen	David Broman, Kristian Sandahl
Telefon under skrivtiden	0707-90 90 75 (DB); 0706-68 19 57 (KS)
Besöker salen ca kl.	15
Kursadministratör/kontaktperson (namn + tfnr + mailaddress)	Åsa Kärrman 013-28 1454 asa.karrman@liu.se
Tillåtna hjälpmedel	Två A4-ark med handskrivna anteckningar. Engelskt lexikon eller flerspråkigt lexikon till eller från engelska. Two sheets of handwritten A4 pages. You may write on both pages, with any type of size and colour. One volume of dictionary to/from English or an English wordbook.
Övrigt	
Vilken typ av papper ska användas, rutigt eller linjerat	
Antal exemplar i påsen	

Linköpings Universitet
IDA
David Broman, Kristian Sandahl

Written exam for Software Engineering Theory

Course codes TDDC88, TDDC93, 725G64

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. Remember that the examiner has over 200 students, so plan for solving problems in a flexible order if you have to wait for clarification.

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 2011. This gives you the mark 3 in the Swedish system and a C in ECTS. 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 in Swedish system	Translation to ECTS
0-49	UK	Fx
50-66	3	C
67-83	4	B
84-	5	A

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 statement 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!

David and 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 where the requirements engineer writes down his/her understanding of the forthcoming system in a Software Requirements Specification.
B. A Feature is a requirement that is highly prioritized by a majority of stakeholders.
C. RAM – Requirements Abstraction Model helps us to identify related requirements on different abstraction levels.
D. One goal with Requirements Analysis is to detect and resolve conflicts between requirements.

1 b) Draw a UML Use-Case Diagram of a system for web auctions of consumer products, such as eBay or Tradera. There shall be two different use-cases and two actors in the diagram. Use-case texts for the use-cases shall be written in complete sentences. (4)

1 c) Write down two functional and two non-functional requirements of the LiU student portal. The requirements shall consist of complete sentences and be possible to use in system testing. (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. Time-boxing means that if not all tasks in a sprint are completed, then the sprint’s length is allowed to be extended maximal one more week.
B. If an activity is delayed, that is part of the critical path in a GANTT-chart, then the whole project will be delayed.
C. Agile software development means that tight customer collaboration makes documentation unnecessary.
D. The product backlog in Scrum is a prioritized list and anyone can add new items to this list.

2 b) Explain the purpose of COCOMO and Planning poker, the main ideas behind these approaches, and how they are different. (4)

2 c) Explain the Scrum concepts Product Backlog, Sprint Backlog, Task Board, and Impediment List. (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 software module named “FileIO” (abbreviation for File Input Output), containing only functions related to file handling, is an example of a module with high cohesion.
B. Integrity is a concept within information security and means that only authorized users may read protected information.
C. Apps in smart phones (e.g., IPhone and Android) that communicate with a data base server over a network are examples of two-tier fat-client architectures.
D. State-machines and sequence diagrams are example of UML diagrams describing the static structure of a software system.

3 b) Explain together with an example the differences between implementation, execution, and deployment architecture views. (4)

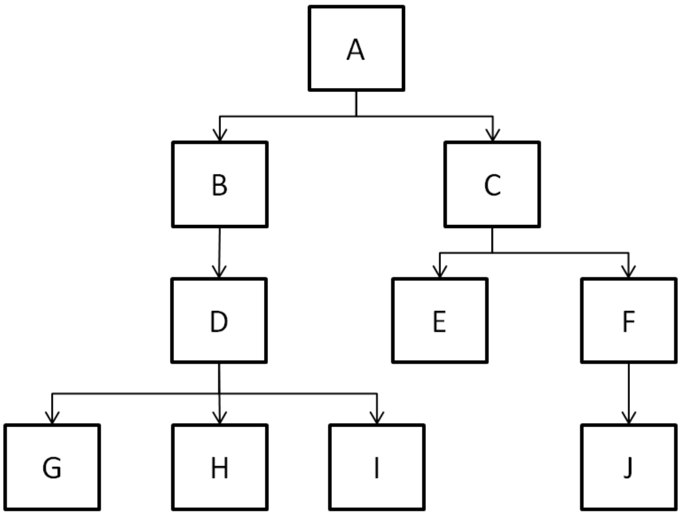
3 c) Explain what is similar and what is different between the two UML relations “composition” and “association” (4).

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 Fault of Commission can be due to a human error.
B. The JUnit framework contains operations for generating test-cases for a suitable white-box coverage criterion.
C. Smoke Tests are a kind of regression test.
D. A Stub is a small piece of software replacing an upper layer in integration testing, by calling the module under test with some predefined parameter settings.

4 b) Below you find a decomposition tree of an imaginary system of 10 modules. The arrows indicate dependency between modules. Describe the order of integration and testing if you are using a Top-Down integration strategy. Give one advantage and one disadvantage of Top-Down integration testing. (4)



4 c) Describe the following terms in the context of Configuration Management: Baseline, Development Branch, Commit[1], and Clone[2]. (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 usability metric, “number of commands invoked by the users” can be measured already with a paper prototype of the system.
B. Companies certify themselves according to ISO Standard 9001 in order to avoid documentation and concentrate on coding high-quality software.
C. The measurement, Non-Comment Lines of Code (NCLOC), is dependent on the implementation language used.
D. The process area Avoid Defects (AD) is mandated on CMMI level 3 in the staged representation.

[1] You may assume that you use Subversion without motivation

[2] You may assume that you use Git without motivation.

5 b) Some Process Areas in CMMI are related. Explain the similarities and differences between the two pairs:

- Project Planning (PP) – Project Monitoring and Control (PMC)
- Requirements Management (REQM) – Requirements Development (RD) (4)

5 c) You are a project manager in a CMMI level 4 company that has a database of measurements of anything you wish to know about your product, processes and resources. Now you meet your manager in the corridor and she tells you that the board wants four different metrics showing how much is left of your project in absolute or relative numbers.

Now, describe the *four* metrics you chose to use, with a thorough motivation of why they indicate how much that is left in your project. (4)

Part 2: Advanced

6. You have a shopping card in your favorite supermarket. It has a very flexible service. When you are shopping you insert the card before paying, and then you have the following alternatives:

1. You can decide to pay from the account you have in the supermarket’s own bank. That is done by yourself answering “yes” to the question “withdraw from card?”, which is show on the screen of the customer terminal.
2. If you answer “no” to the question you can either pay in cash or with another bank card. If you chose another card, you manage the payment yourself at the customer terminal.
3. If there is not enough money in the account you have in the supermarket’s own bank, you can ask the cashier to empty your account, and take the rest of the payment in cash or with another bank card.
4. The other alternative if there is not enough money in the account you have in the supermarket’s own bank, is to ask the cashier to cancel the purchase and add money to your account. This is done by re-inserting the card in the customer terminal, when the cashier tells you to do so. Then you type in the sum you would like to add, and give the cashier the sum in cash. An alternative is to draw the sum from another bank card. In the latter case, you inform the cashier, who will ask you to remove the card, and insert the other bank card. Regardless of how you pay, you will get a receipt, and then you redo the purchase with the shopping card of the supermarket (now there are enough money on the account).
5. Regardless of how you want to pay, the system checks if you are eligible for any discounts, in that case the purchase sum is updated. After a fulfilled payment you get bonus credits registered.

On your way home you think about how such a system can be designed, and after a good meal, you sit down and start sketching a possible design with UML diagrams.

a) Draw a class diagram describing the payment system with shopping cards. There shall be at least five classes, all with at least one attribute, and one operation. Associations shall have a name and multiplicity information, unless you use UML relationships. For each class you shall motivate why you chose to represent this model element as a class in the system. 1-2 sentences will do for each class. (10)

b) Draw one or more sequence diagrams describing the wordy alternatives above for the customer to pay. You need at least 3 roles participating in the exchange of messages and at least 10 messages. You need to use fragments to make the diagram fairly simple to read. A valid solution to this task has at least one fragment. Omit all low-level signaling setting up connections between different parts of the system. (10)

7. Assume that you are responsible of performing manual system testing of a web-based flight booking system. Your current task is to verify that the search function for available departures is working correctly. In the search page, there are three input text fields: “From city”: the name of the city for departure, “To city”: the name of the city for arrival, and “Departure date” that is a text string of format “YYYY-MM-DD”, where YYYY means the year (e.g., 2011), MM the month (e.g. 07), and DD the day (e.g. 05). When a user has filled in the information in all fields, he/she clicks on a “Search” button and the system returns a page with possible flights. The following list of requirements gives a partial view of the system:

1. The system shall have a maximal response time of 2 seconds, where the response time means the elapsed time between user request and user response.
2. The system shall validate that the date of travel is a valid date.
3. If a user inputs a city name that is misspelled, the system shall suggest a city name that is close to the input text. For example, if the user input is “Linkpoing”, the system shall ask “Did you mean city Linköping”.
4. The system shall list flights that are on the departure date as well as the flights that are available the day before and day after the departure date.

The system has a database that is storing information about all possible flights. For each possible flight the following information is stored in the database:

- Name of the city that the flight is departing from
- Name of the of arrival city
- Date of departure
- Time of departure
- Flight time

You may make further assumptions as long as you describe them clearly.

a) Define a test database with at least 10 flights. Define at least five important equivalence classes (both valid and invalid). Explain why you have chosen these database items and these equivalence classes. You may make further assumptions as long as you give clear explanations (10).

b) Write down 8 test cases of the system using boundary value testing. Discuss to what level your test plan covers testing of all stated requirements. Explain in detail both where your plan is strong and weak, and how it could be further improved to be more complete (10).

c) Explain how software inspections can be used to discover faults. Give a brief overview of the inspection process and its roles. Discuss and argue for or against using software inspections together with testing in the given scenario (10).