Linköpings universitet
IDA Department of Computer and Information Sciences
Doc. Christoph Kessler

EXAM

FDA149 Software Engineering

2006-02-10, 14:00–18:00

Name: _____ **Personnummer:** _____

Please mark solved problems with X:

Question	1	2	3	4	5	6	7	8	9	10	11
answered (X)											
score (points)											

Total score: _____ **Grade (U/3/4/5):** _____

Sign. Examinator: _____

Examinator: Christoph Kessler

Jour / tentavakt (Linköping): Christoph Kessler (070-3666687, 013-282406)

External supervisors for exams written outside Linköping,

please confirm with your signature below that you have checked the identity of the candidate and supervised this exam 10/2/2006 from 14:00–18:00 before you send it by mail to Christoph Kessler, IDA, Linköpings universitet, 58183 Linköping.

Supervisor (name, signature): _____ _____

Hjälpmedel / Admitted material:

None.

General instructions

- This exam has 11 assignments. Read all assignments carefully and completely before you begin.
- It is recommended that you use a new sheet for each assignment. Number all your sheets, and mark each sheet on top with your name, personnummer, and a page number.
- You may answer in either English or Swedish.
- Write clearly. Unreadable text will be ignored.
- Be precise in your statements. Unprecise formulations may lead to a reduction of points.
- Motivate clearly all statements and reasoning.
- Explain calculations and solution procedures.
- The assignments are *not* ordered according to difficulty.
- The exam is designed for 40 points. You may thus plan about 5 minutes per point.
- Grading: U, 3, 4, 5. The preliminary threshold for grade 3 is 20 points.

Assignments

1. (4 p.) **Software processes**

   (a) Write down two drawbacks of deploying a prototyping process model. (2p)

   (b) Write down two rules of *eXtreme Programming*. (2p)

2. (5 p.) **Modeling in UML**

   (a) Name two UML diagram types (of your choice, but *not* the class diagram), describe shortly (2–3 sentences each) what they specify, and give a small example. (2p)

   (b) A library system is to be constructed according to the following principles:

   (1) The system must allow employees of the library to search the book database. It should also provide extended information about the books, e.g. where it was bought, the price and who has borrowed it. The extended information must only be available to library employees.

   (2) The book database must be available on the Web where every one can perform simple search on title, author or publication year.

   (3) Library employees can register new users and issue library cards. It should also be possible to unregister users or change their contact information (address and phone number)

   (4) Employees can add new books and remove old ones.

   (5) A registered user, who has a library card, can view her/his loans on the Web. This function requires that the user has logged in. The user can also extend the loan period while logged in.

   (6) An employee can register new loans, returned books and extend loans.

   Now, perform a noun-analysis of the text and write down at least two classes with at least two methods and/or attributes each, using UML notation. There should be at least one relation between the classes.
   (Note: You do not have to account for your considerations during noun-analysis, just write down the resulting model. If you are uncertain about how to draw a certain UML notation element, invent your own notation but give a clear explanation of what it should denote.) (3 p)

3. (4 p.) **Software project organization**

   (a) There are many models and methods for software development effort estimation. Write down the name of one of them. Write down a motivation for your choice and list the data you would need to use the model. (2p)

   (b) Write down two arguments for using inspections as a means to validating software. (2p)

4. (4 p.) **Testing**

   (a) Suppose a program contains N decision points, each of which has two branches. How many test cases are needed to perform path testing on such a program? If there are M choices at each decision point, how many test cases are needed for path testing? Can the program's structure reduce this number? Give an example to support your answer. (1p)

   (b) Name and describe briefly the following:

   A - one kind of Black Box Testing,

   B - one kind of White Box Testing (Coverage Analysis),

   C - one kind of Integration Testing. (3p)

5. (4 p.) **OO technology**

   (a) Class A, which features a single, externally visible method `m` with the signature
   `Type0 m ( Type1 param1, Type2 param2 )`,
   has been used in a large legacy software system for years but became too slow for the increased throughput and should be replaced. There are several faster candidates on the component market that promise to be able to replace A. Apart from performance aspects, how can we make sure that a class B "fits" as a replacement of A? Formally speaking, what is the condition for a class B to be able to substitute class A in all its uses (syntactic substitutability)? (2p)

   (b) What is the *Syntactic Fragile Base Class Problem*? (2p)

6. (2 p.) **Metaprogramming**

   Define the terms *reflection* and *reification*. (2p)

7. (3 p.) **Model-driven architecture**

   Give a short description of Model-driven architecture (MDA) (main ideas). (2p)
   What is the main motivation for it? (1p)

8. (2 p.) **Enterprise JavaBeans (EJB)**

   Name two advantages of using Enterprise JavaBeans (EJB) when developing Java business applications.

9. (4 p.) **CORBA**

   (a) How does CORBA achieve programming language transparency for static calls? (Describe the principle and those main parts involved in CORBA static calls that are primarily relevant for enabling language transparency.) (3p)

   (b) What does the CORBA trader service do? (1p)

10. (6 p.) **Software Architecture Systems**

    (a) What is the component model of software architecture systems? (in other words, what are the (major) different building blocks of a software architecture configuration, and what is their specific purpose?) (2p)

    (b) Define the term *layered architectural style* (also known as *onion architectural style*) and give an example of a software system with this style. (2p)

    (c) Software architecture systems are said to be a first step towards separation of concerns. Which concerns are separated, and what are the advantages of this? (2p)

11. (2 p.) **Aspect-Oriented Programming**

    Aspect oriented programming simplifies software development by providing a mechanism for implementing concerns that crosscuts modules. Describe a major disadvantage with aspect oriented programming. (2p)