EXAM
FDA149 / TDDC54 Software Engineering
2006-02-16, 14:00–18:00

Name: ___________________________ Personnummer: ___________________________

Please mark solved problems with X:

<table>
<thead>
<tr>
<th>Question</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>answered (X)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>score (points)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Total score: ___________________________ Grade (U/G, 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 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 12 assignments. Read all assignments carefully and completely before you begin.

- Those who only do half the exam (corresponding to the second half of the course or TDDC18), please solve only assignments 6 to 12. Hand in a copy of your Ladok account documenting the result you scored in the PUM exam (i.e., TDDC01, TDDC06, TDBB62 or equivalent) together with your exam.

- 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 and 4 hours (or 20p and 2h for half the exam). You may thus plan about 5 minutes per point.

- Grading: For FDA149: U, 3, 4, 5. For TDDC54 (CUGS master students): U, G. The preliminary threshold for passing is 20 points (10 points for those who write half the exam).

- The exams will be corrected by 6 different teachers in sequence. Hence, expect at least 3 weeks until the result will be available.
Assignments

1. (5 p.) **Software processes**

   (a) Write down a decision table for selection of life cycle or process models. The columns represent properties of the problem, properties of the organisation or wanted quality attributes and constraints of the forthcoming systems. The rows are the life cycle or process models, and the squares contain short information of the relation between the column and row. Make sure you have at least five life cycle or process models and comments in at least 10 squares. Example:

<table>
<thead>
<tr>
<th>P1</th>
<th>Q2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waterfall model</td>
<td>The more change-prone requirements, the less appropriate is the model.</td>
</tr>
<tr>
<td>...</td>
<td>Easy to connect to a project management model.</td>
</tr>
</tbody>
</table>

   P1=Requirements volatility
   Q2=Manageability

2. (4 p.) **Use-case modeling in UML**

   (a) Read the following use-case diagram:

   ![Use-case Diagram](image)

   Think of some classes and draw a sequence diagram that captures most of the behaviour from the use case. Also write down use-case behaviour you have not included in the sequence diagram.

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

   (a) Write down detailed instructions of how to measure:

   (A) The maintainability of a system.
   Suitable keywords: complexity, size, documentation, comments, observed maintenance, replacement.
(B) The relevance of a prototype of a highly interactive system. Suitable keywords: perceived relevance, invoked commands, completeness, tasks completed, weighting measures

4. (1 p.) Design patterns
   
   (a) What is a design pattern? (1p)

5. (6 p.) Testing
   
   (a) Define the terms below: (2 p)
   
   (A) Fault
   
   (B) Failure
   
   (C) Test
   
   (D) Test case
   
   (b) The following figure illustrates the component hierarchy in a software system. Describe the sequence of tests for integration of the components using
   – a bottom-up approach and
   – a sandwich approach. (4 p)

   ![Component Hierarchy Diagram]

   The following assignments apply to the second part of the course.

6. (6 p.) OO technology
   
   (a) Explain the term syntactic substitutability. Explain the terms covariance and contravariance of parameter types, in the context of design by contract. Which of these two imposes an extensibility problem in OO-based component-based design, and why? (3p)

   (b) Explain the Syntactic Fragile Base Class Problem. What are its implications for the compatibility of binary components compiled from OO programs? (3p)
7. (1 p.) **Metaprogramming**

(a) Define the term *introspection*, and give an example of where and how it is used in some component system. (1p)

8. (1 p.) **Model-driven architecture (MDA)**

(a) Explain how MDA supports reuse. (1p)

9. (3 p.) **Enterprise JavaBeans (EJB)**

(a) Name two advantages of using Enterprise JavaBeans (EJB) when developing Java business applications. (2p)

(b) One could argue about whether Enterprise JavaBeans are object oriented or not. What are the arguments against? (1p)

10. (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. What is the purpose of each part? Which parts are to be written by the programmer, and which ones are generated and how?) (3p)

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

11. (3 p.) **Software Architecture Systems**

(a) 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? (1.5p)

(b) Define the term *layered architectural style* (also known as *onion architectural style*) and give an example of a software system with this style. (1.5p)

12. (2 p.) **Aspect-Oriented Programming**

(a) Aspect oriented programming simplifies software development by providing a mechanism for implementing concerns that crosscuts modules. Describe a major disadvantage with aspect oriented programming. (1p)

(b) What is an *advice* in aspect-oriented programming? (1p)