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

EXAM

TDDD05 / TDDC18 Component-based Software

2009-01-15, 14.00–18:00, TER2

Examinator: Christoph Kessler

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

Hjälpmedel / Admitted material:

Engelsk ordbok / Dictionary from English to your native language

General instructions

- This exam has 9 assignments and 4 pages, including this one.
  Read all assignments carefully and completely before you begin.
- Please use a new sheet of paper for each assignment. Number all your sheets.
- 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 passing (grade 3) is 20 points.

  International master students and exchange students will receive ECTS grades. Due to anonymization of exams, the ECTS grades will be set automatically by the central university administration, applying the LiU guideline for translation from Swedish grades: 5=A, 4=B, 3=C, U=FX.

1. (6.5 p.) **OO technology and design by contract**

   (a) What is a contract? (0.5p)

   (b) How do interfaces relate to contracts? (0.5p)

   (c) Are objects components?? Considering Szyperski's widely accepted definition of a component, objects (as in ordinary object-oriented programming) have certain properties in common with components, but there are also important differences. Explain what properties are in common, and what the differences are. (2p)

   (d) Components such as language libraries, system functions etc. may be subject to further development e.g. by the component manufacturer even after releases have been shipped to clients who happily use them in their own programs.
   Describe the *Semantic (!) Fragile Base Class Problem*. Give also a simple example demonstrating the problem.
   Which fundamental property of OO languages causes the problem?
   Name a policy for component developers that allows to avoid this problem in practice. (3.5p)

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

   (a) What is *introspection*, and why is it so important for the flexibility of component-based systems? (1p)

   (b) Give examples for
   (i) static metaprogramming at base level,
   (ii) static metaprogramming at meta level,
   (iii) dynamic metaprogramming at base level. (1.5p)

   (c) Give a C++ example definition of an arbitrary generic *data type* using the C++ template mechanism, and show how to generate a concrete data type from it. Explain carefully what the C++ compiler generates from your code. (2.5p)

   (d) In short: What is the Meta-Object Facility (MOF), and what is its purpose? (1p)

3. (3 p.) **Java Reflection**

   Write a Java program using Java Reflection: Your program should read in a string `cname` that is supposed to be the name of a loadable Java class, and print a list of the signatures (return type, method name, list of parameter types) of all methods of that class (or an error message if that class cannot be loaded). Explain carefully what the used reflection operations do.

   (If you do not recall the exact Java syntax, use your own one.) (3p)

4. (6 p.) **CORBA**

   (a) Without middleware support, a program written in programming language A usually cannot directly call a method written in a different programming language B even if they would execute on the same computer. Give one of the technical reasons. (0.5p)

(b) 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. In particular, what does each of these parts do at a call? and which parts are generated, and from what source?) (3p)

(c) (i) What is the purpose of an interoperable object reference (IOR) in CORBA?
(ii) What data does an IOR contain?
(iii) Where are IORs created?
(iv) Name two possibilities of how a client can get hold of an IOR. (2.5p)

5. (3 p.) **JavaBeans**

(a) JavaBeans and Enterprise Java Beans (EJB) have, despite the similar name, very different characteristics and purpose. Explain shortly what JavaBeans is and what Enterprise Java Beans is, highlighting two key differences between them. (2p)

(b) If there was no reflection mechanism is Java, the JavaBeans component model would not work. Why? (1p)

6. (6 p.) **Enterprise JavaBeans (EJB)**

(a) Enterprise JavaBeans (EJB) allows for separation of business logic from some middleware services. Give two examples of such middleware services and describe how Enterprise JavaBeans handles them. (2p)

(b) For each Enterprise JavaBean (EJB), the developer must specify which kind of bean it is.
(A) What are the 3 different kinds of Enterprise JavaBeans?
(B) Why does the EJB system need to distinguish between different kinds of beans? Give 2 examples of what this information can be used for. (3p)

(c) Compared to EJB 2.0, EJB 3.0 involves several simplifications. What is the key difference with respect to object orientation? (1p)

7. (2.5 p.) **Aspect-Oriented Programming (AOP)**

Write an Aspect-J program that inserts immediately before each call to a function `foo` a `print` statement that prints a log message containing the function's name to standard output. Explain carefully what the different constructs that you use *are* (using Aspect-J terminology), and what they *do*.

If you do not remember the exact Aspect-J syntax, invent your own one, and explain what it means.

8. (6 p.) **Model-driven architecture (MDA)**

(a) Describe the main concepts of Model-Driven Architecture (MDA, as proposed by the OMG). (2.5p)

(b) Give two main arguments how using MDA can increase programmer productivity. (1p)

(c) Compared to traditional software development, will switching to model-driven software development pay off immediately? (Justify your answer carefully. A simple 'yes' or 'no' gives no points.) (1p)

(d) The model-driven development approach used by SAAB, as presented in the guest lecture, is also known as MDA-light, which is a simplification of the "official" MDA approach as advocated by the OMG.

What is the main simplification? (short answer)

What are the main advantage(s) and disadvantage(s) of that approach? (1p)

9. (1 p) **Web services**

(a) In short: What is the relationship between Web Services and Service-Oriented Architecture (SOA)? (0.5p)

(b) What is the purpose of the language BPEL for web services? (0.5p)

Good luck!