KTH Microelectronics and Information Technology

Exam in ID2207 Modern Methods in Software Engineering, 2010-10-18, 09:00-13:00

Rules

This exam is "closed book" and you are not allowed to bring any material or equipment (such as laptops, PDAs, or mobile phones) with you. The only exceptions are English to "your favorite language" dictionary and pencils.

Instructions

- Please read the entire exam first!
- Write clearly
- Each sheet of paper must contain your name, "personnummer", Problem number and a unique sheet number
- Write only on one page of a sheet. Do not use the back side
- Sort your sheets according to the problem's numbering!
- Only one Problem must be reported on each sheet
- If more than one sheet is needed the continuation should be clearly noted on the beginning of each sheet and the sheet numbers used should be consecutive
- Always motivate your answers. Lack of clearly stated motivation can lead to a reduction in the number of points given
- The tasks are not necessarily sorted in order of difficulty. If you get stuck it might be a good idea to go on to the next task.

Grading

If n is amount of your exam points and m is your bonus points earned in autumn 2010 then:
$n+m < 50$ fail (F)
$50 \leq n+m < 60$ grade E
$60 \leq n+m < 70$ grade D
$70 \leq n+m < 80$ grade C
$80 \leq n+m < 90$ grade B
$90 \leq n+m$ grade A

GOOD LUCK!

Mihhail Matskin, mobile 0704614269

Problem I. General questions

a) What are characteristics of software that make it different from other products? Briefly explain.

(4p)

b) A programming language is a notation for representing algorithms and data structures. List two advantages and two disadvantages of using a programming language as sole notation throughout the development process.

(4p)

Problem II. Software Life Cycle

a) Unified Software Development Process (UP) considers phases, iterations and workflows in software life cycles. Briefly explain each of them and how they are related.

Cycles consist of the four Phases: Inception (establish business case), Elaboration (specification of pruduct cases, design of architecture), Construction (product is built), Transition (period when product moves to beta phase). Each phase consists of a number of iterations. Each iteration is a set of use cases or considers certain risks. For each iteration, several workflows are performed parallelly (supporting workflows: Management, Environment, ... engineering workflows: requirements, design, ...)

(5p)

b) Explain what is common and what is different between transformational and deductive synthesis

Both base on deduction but they use different deduction methods: Synthesis is based on inference (axioms and rules are expressed as implications) Transformation is based on replacement (axioms are expressed as equations or rewrite rules).

(5p)

c) Explain what is common and what is different between transformational synthesis and Model Driven Architecture (MDA)

(5p)

Problem III. UML and OOP

a) What is/are difference(s) between Sequence diagrams and Communication diagrams? When each of them is more appropriate to use?

(4p)

b) How do we represent `include` and `extend` relationships in the textual description of use cases?

(6p)

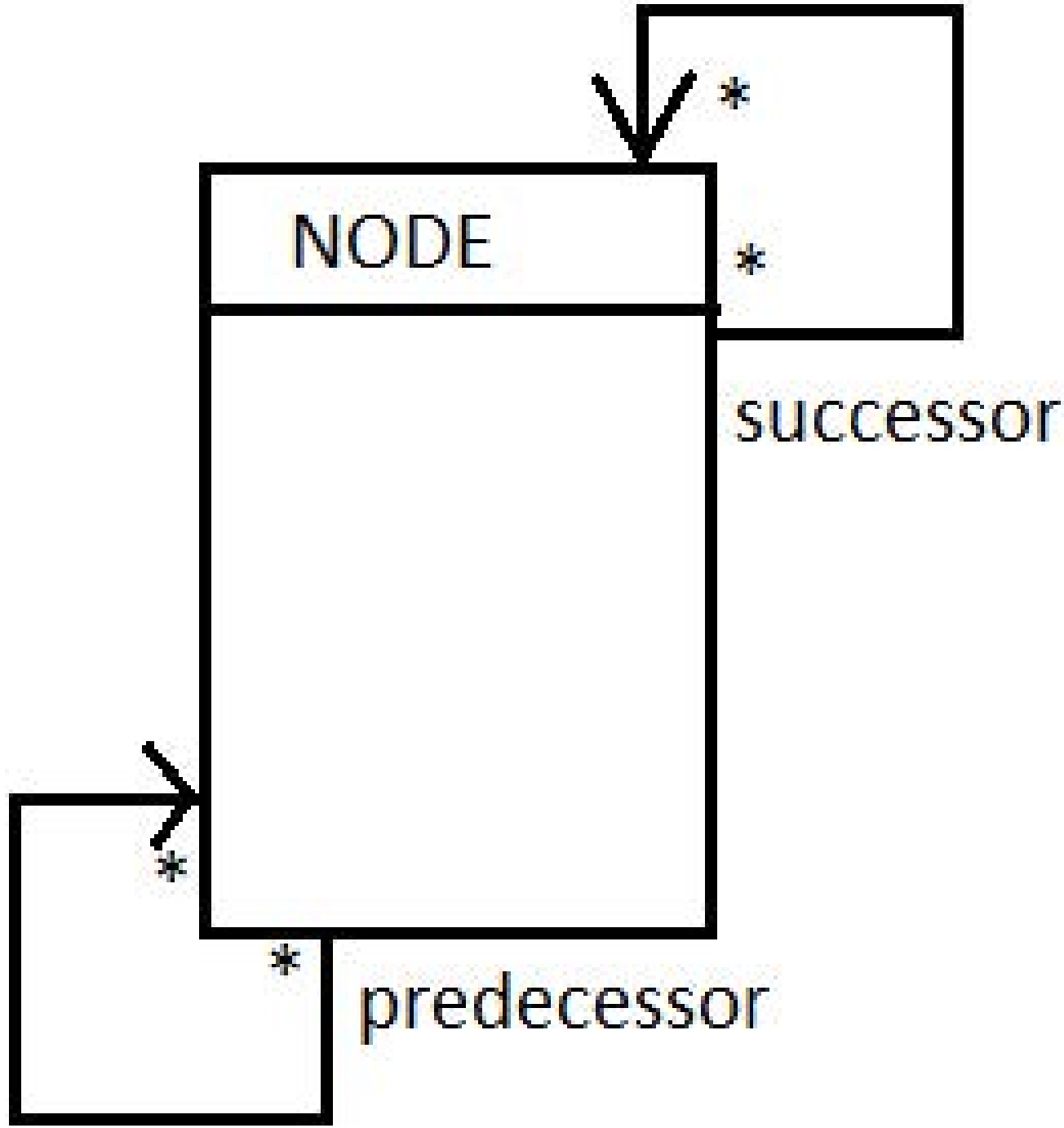
c) Using ONLY ONE CLASS draw a class diagram where each task has a successor and a predecessor tasks and one task may suspend execution of some other task(s).

(5p)

Problem IV. Requirements Elicitation

a) What are main categories of requirements in the FURPS+ model? Briefly explain them

(5p)


NODE
*
*
successor
*
*
predecessor

b) Which requirements validation criteria do you know? Briefly explain them. Give an example of a requirement that can't be validated.

(5p)

Problem V. Requirements Analysis

a) Consider a file system with a graphical user interface, such as Microsoft's Windows. The following objects were identified from a use case describing how to copy a file from a memory stick to a hard disk: File, Icon, TrashCan, Directory, Disk, Pointer. Specify which are entity objects, which are boundary objects, and which are control objects.

ENTITY (persistent information): File, Folder, Disk
BOUNDARY (interaction with user): Icon, TrashCan, Pointer
CONTROL (control tasks performed by the system): -none-

(4p)

b) Briefly explain the main idea of Aspect-Oriented Programming. Give an example

Some problems are not attached to any particular domain but they are spread across them
EXAMPLE: see next page

(5p)

Problem VI. System Design

a) Older compilers were designed according to a pipe and filter architecture, in which each stage would transform its input into an intermediate representation passed to the next stage. Modern development environments, including compilers integrated into interactive development environments with syntactical text editors and source-level debuggers, use repository architecture. Identify the design goals that may have triggered the shift from pipe and filter to repository architecture.

(5p)

b) How boundary conditions are modeled during system design activity? Why they can't be modeled during requirements activity?

(4p)

Problem VII. Object Design - Reuse

a) What is the Liskov Substitution principle? Which type of inheritance it defines? What are benefits of design following this principle?

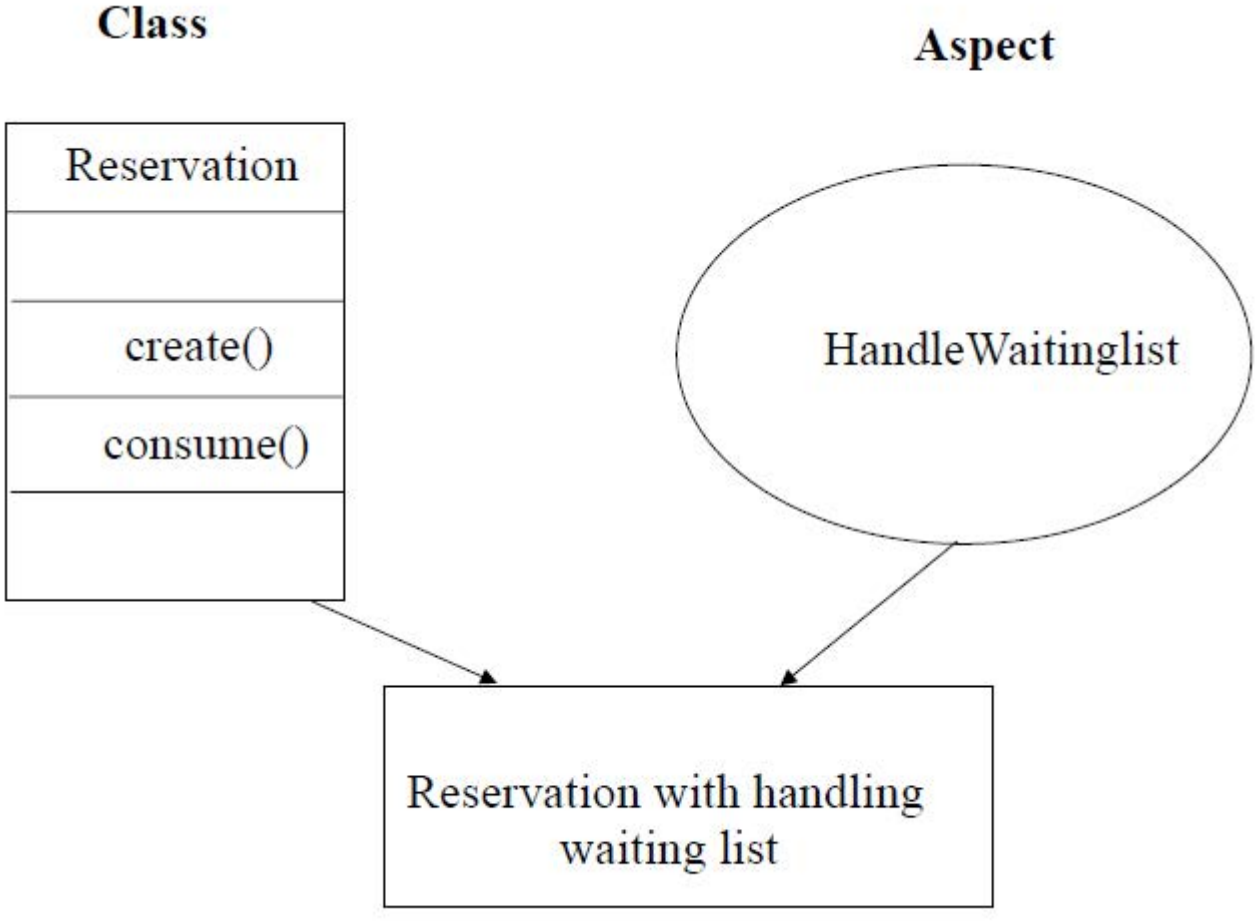
(5p)

b) What are similarities and differences between Adapter and Bridge patterns?

(4p)

Problem VIII. Object Design – Interface design

a) Consider a class diagram in the Figure 1.


Class
Aspect
Reservation
create()
consume()
HandleWaitinglist
Reservation with handling waiting list

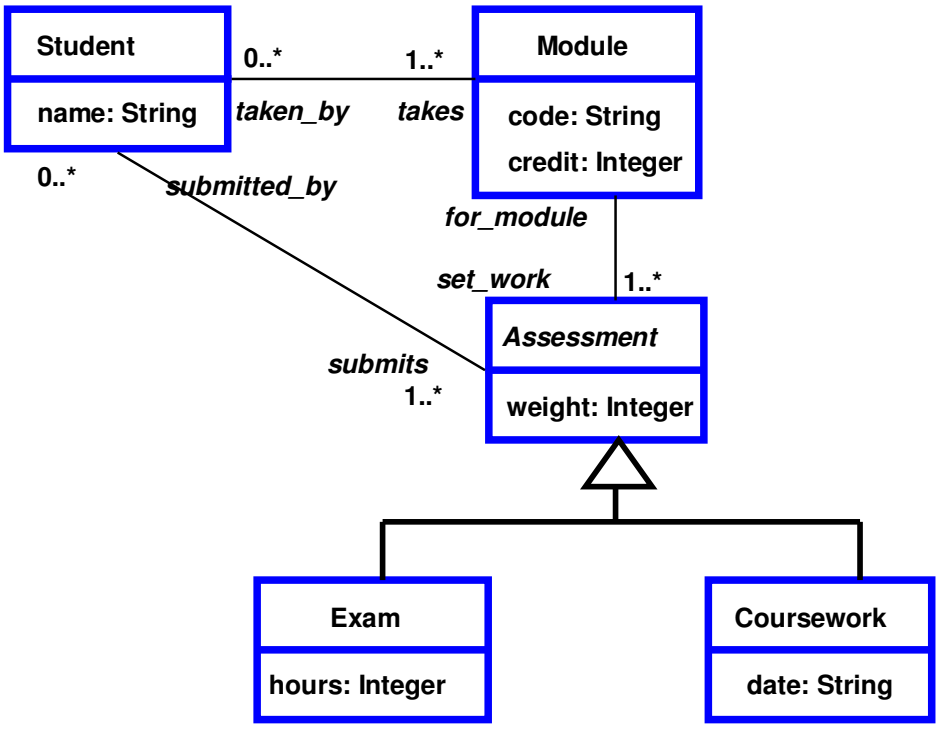


Figure 1.

Your task is to define in OCL the following constraints:

1) **Modules can be taken if they have more than seven students registered**
2) **The assessments for a module must total 100%**
3) **Students must register for 120 credits each year**

(6p)

Problem IX. Moving to Code

a) Explain realization of a unidirectional, one-to-one association in source code.

(4p)

Problem X. Testing.

a) Explain modified sandwich testing strategy. What are advantage(s) and disadvantage(s) of this strategy?

(5p)

b) Give a formula for calculation of the minimal number of tests required in the path testing. This calculation depends on the number of nodes and edges in the equivalent flow graph of the program.

(5p)

Problem XI. Agile Software Development.

a) What is Spike in the Extreme programming and when it is used? Give an example

(5p)

----------End of Exam----------