# ID2207 Modern Methods in Software Engineering 7.5 credits

## Moderna metoder inom Software Engineering

This is a translation of the Swedish, legally binding, course syllabus.

## Establishment

Course syllabus for ID2207 valid from Autumn 2008

## Grading scale

A, B, C, D, E, FX, F

## Education cycle

Second cycle

## Main field of study

## Specific prerequisites

Knowledge and skills corresponding to Introduction to Computer Science and Operating systems courses. Knowledge of Java is desirable.

## Language of instruction

The language of instruction is specified in the course offering information in the course catalogue.

# Intended learning outcomes

The course aims both in giving students knowledge about modern software development methods and developing skills in usage the methods.

Our goal is to present a variety of approaches to software development and discuss their applicability boundaries, benefits, restriction and complementariness.

During the course students should learn about Software Engineering methods. In particular, they:
1. Learn methods for dealing with complexity and changes in software construction. This means that students should get understanding of main approaches to abstraction, models, decomposition and software life-cycle
2. Understand basic components of software development process. This means that students should learn main methods and approaches to requirement elicitation and analysis, system and object design
3. Learn some modern approaches to software development. This means that students should learn about agile methods of software development and have experience in applying them to desing a software system
4. To get experience in evaluating different methods for producing of a high quality software system within time. This means that students should get practice in comparison different approaches to software development.

# Course contents

Introduction and basic concepts of Software Engineering (SE). Abstraction/Models and Decomposition. Software Life-Cycle. Unified process. Software Modeling language. Unified Modeling Language (UML). Requirements elicitation and analysis. System design. Object design. Applying patterns. Refactoring. Mapping models to code. Testing. Agile software development and agile modeling. Basics of Extreme Programming. Software project management.

Practical part of the course includes exercises and a small software development project applying SE methods

# Course literature

Text-book and selected papers (to be provided in the course)

# Examination

- ANN1 - Assignment, 3.0 credits, grading scale: P, F
- TEN1 - Examination, 4.5 credits, grading scale: A, B, C, D, E, FX, F

Based on recommendation from KTH's coordinator for disabilities, the examiner will decide how to adapt an examination for students with documented disability.

The examiner may apply another examination format when re-examining individual students.

If the course is discontinued, students may request to be examined during the following two academic years.

## Other requirements for final grade

Written examination (TEN1 4,5hp.)
Homework and project assignment (ANN1 3hp.)

## Ethical approach

- All members of a group are responsible for the group's work.
- In any assessment, every student shall honestly disclose any help received and sources used.
- In an oral assessment, every student shall be able to present and answer questions about the entire assignment and solution.