DD2455 Theoretical Foundations of Object-Orientation 7.5 credits

Teoretiska grunder för objektorientering

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

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

Establishment

Course syllabus for DD2455 valid from Spring 2009

Grading scale

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

Education cycle

Second cycle

Main field of study

Specific prerequisites

Language of instruction

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

Intended learning outcomes

The overall aim of the course is to provide an understanding of two advanced methods for ensuring object-oriented software quality, namely: (i) live sequence charts (LSCs) for requirements analysis, and (ii) invariant analysis of software code.

This understanding means that after the course you should be able to:

Critically assess an informal text-based software requirements document, identifying ambiguities, omissions and inconsistencies. Translate such a document into object-oriented requirements using a Noun/Verb/Relational-Phrase methodology. Construct a data dictionary, and relate this to the original requirements document using hypertext technology.

Identify, express in text, and formalise using LSCs the important use-cases in a requirements document. You should be able to distinguish between normal and exceptional scenarios.

Draw LSCs using the Play-Engine, given a pre-existing user interface model.

You should be able to organise your charts within a project, using the Play-Engine. You should also be able to modify the different components of a project with an understanding of the technical effects of any changes

Exercise a set of LSCs using the Play-Engine simulator to study their interaction, and how they co-operate to achieve the user requirements. You should understand what incomplete and inconsistent requirements are, and how to identify these by using simulation.

Translate a short piece (less than 20 lines) of object-oriented code into a flowchart. You should understand the meaning of a valid labelling of such a chart by logical assertions according to Floyd's invariant assertion method.

Construct a valid labelling of a flowchart by means of dragging a pre-condition forwards, or a postcondition backwards, and using basic logical transformations. You should be able to use your knowledge of programming to synthesize loop invariants and thus prove a program is mathematically correct with respect to a software requirement.

Understand the syntax of a simple sequential object-oriented programming language and be able to correctly add extra features to the syntax.

Understand the operational semantics of a simple sequential object-oriented programming language and be able to correctly add extra features to the semantics (for example garbage collection) that define the meaning of new syntactic features.

Understand the logic of a simple sequential object-oriented programming language and be able to explain the meaning of rules of inference.

Understand the concept of an abstract data type, how it relates to a class definition, and be able to define new simple data types.

Course contents

- A review of object-oriented themes, terminology, computational model, relationship to other programming paradigms, software lifecycle, new trends.
- Principles of sequential program correctness. Operational semantics of sequential programs. Hoare's logic and Floyd's correctness analysis using labeled flowcharts. Programming by contract and Eiffel.
- Abstract data types, signatures, equations, semantics. Objects as algebras. Correctness revisited.
- Polymorphism. Inheritance and the subtype relation, flavours of polymorphism, type abstraction, existential types and information hiding.
- Concurrency. Axioms for communicating processes, traces, interleaving semantics of concurrency.

Course literature

A. Eliëns, Principles of Object-oriented Software Development, 2nd Edition, 2000, Addison-Wesley, ISBN 0-201-39856-7.

Examination

- HEM1 - Home Work, 1.5 credits, grading scale: P, F
- TEN1 - Examination, 6.0 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.

Other requirements for final grade

Written examination (TEN1; 6 university credits), home work (HEM1; 1,5 university credits).

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.