GHENT UNIVERSITY

Course Specifications

Valid as from the academic year 2023-2024

Software Engineering (E761035)

Course size *(nominal values; actual values may depend on programme)*

Credits 6.0 **Study time 180 h**

Course offerings and teaching methods in academic year 2023-2024

A (semester 2)	Dutch	Gent	lecture
			seminar

Lecturers in academic year 2023-2024

Ongenae, Veerle	TWO5	lecturer-in-charge

Offered in the following programmes in 2023-2024	**crdts**	**offering**
Bachelor of Science in Engineering Technology(main subject Information Engineering Technology)	6	A
Linking Course Master of Science in Information Engineering Technology	6	A
Preparatory Course Master of Science in Information Engineering Technology	6	A

Teaching languages

Dutch

Keywords

UML, Systeemanalyse, Modellering, C#, Design Patterns, Dependency Injection, Computerwetenschappen (P170), Informatica (P175), Computertechnologie (T120)

Position of the course

The purpose of this course is to teach students advanced object oriented programming and design.
In the first part of this course, the software development is treated. The aim is to enable the student to bring small projects to a successful conclusion. Methods are thaught to produce high-quality software.
In addition, this course aims to provide students with insight into the available "design patterns" for software design and for typical software problems.

Contents

Part 1: Systems Analysis and Design

• Basics of good programming practice: characteristisation of good software and a good development.
• Reuse: how to reuse existing software and write code that can be reused.
• The different phases of the development process.
• Basics of UML.
• Requirements analysis and modelling: how to define the system to be developed.
• Design and realisation: converting a formal model to code.

Part 2: Design patterns

An overview of the most used "design patterns" and object oriented design principles: Strategy, Observer, Decorator,
Factory Method, Abstract Factory, Singleton, Command, Adapter, Facade, Template Method, Iterator, Composite, State, Proxy, MVC, Bridge, Builder, Chain of Responsibility, Flyweight, Interpreter, Mediator, Memento, Prototype, Visitor, inversion of control, dependency injection, ...
In addition, a number of advanced programming concepts are introduced: GUI programming, delegates, extension methods, asynchronous methods, ...

Initial competences

Being able to program and design in an object oriented way on an advanced level

Final competences

1 Apply principles of software design to the practice of production, maintenance and quality
2 Analyse, structure and translate a relatively complex problem into an object oriented design
3 Convert an object oriented design to a working computer program in Java and test this program critically
4 Explain the available "design patterns " for software design and for typical software problems and illustrate with examples
5 Estimate in which situation which pattern is suitable
6 Develop programs using patterns in a suitable way
7 Refactor programs according to some patterns

Conditions for credit contract

Access to this course unit via a credit contract is determined after successful competences assessment

Conditions for exam contract

This course unit cannot be taken via an exam contract

Teaching methods

Seminar, Lecture

Extra information on the teaching methods

• Lectures (24 hrs)
• Labs (36 hrs): individual work on PC

Learning materials and price

"C# 3.0 Design Patterns", Judith Bishop, O'Reilly, 2008, completed with teacher's course (Dutch), slides and examples of programming
Bundled slides are distributed via Hermes at approximately 6 €. The purchase of the book is not mandatory and can also be done through Hermes. Estimated cost 45 €.
Software: Visual Studio 2022 Community Edition, Visual Studio Code with extension PlantUML

References

“Head First Design Patterns”, Eric Freeman, Elisabeth Robson, Bert Bates & Kathy Sierra, O'Reilly Media
"Design Patterns: Elements of Reusable Object-Oriented Software", Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Addison-Wesley
“Praktisch UML”, 5e editie, Jos Warmer en Anneke Kleppe, ISBN 9789043020558

Course content-related study coaching

The student can always make an appointment with the teachers

Assessment moments

end-of-term and continuous assessment

Examination methods in case of periodic assessment during the first examination period

Skills test, Written assessment

Examination methods in case of periodic assessment during the second examination period

Skills test, Written assessment

Examination methods in case of permanent assessment

Skills test

Possibilities of retake in case of permanent assessment

examination during the second examination period is not possible

Extra information on the examination methods

Several computer tests on PC and same tasks during the labs.

Calculation of the examination mark

Exam: 60% (written examination (60%) and computer exercises(40%))
Exercises/Labs: 40% (tests en tasks)
In the second examination period: score = maximum (E, 40% L + 60% E), where L is the score of the lab and E the score of the exam in the second examination period