Course Specifications
Valid as from the academic year 2019-2020

Software Engineering (E761035)

Lecturers in academic year 2019-2020
Ongenae, Veerle TW05

Course offerings and teaching methods in academic year 2019-2020
A (semester 2) Dutch seminar: practical PC room classes 36.0 h
lecture 24.0 h

Offered in the following programmes in 2019-2020

<table>
<thead>
<tr>
<th>Programme</th>
<th>Credits</th>
<th>Offering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bachelor of Science in Engineering Technology</td>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>Preparatory Course Master of Science in Information Engineering Technology</td>
<td>6</td>
<td>A</td>
</tr>
</tbody>
</table>

Teaching languages
Dutch

Keywords
UML, Systeemanalyse, Modellering, C#, Design Patterns, Dependency Injection, AOP, 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 taught 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: characterisation 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

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

Final competences
1. Being able to apply principles of software design to the practice of production, maintenance and quality
2. Being able to analyse, structure and translate a relatively complex problem into an
object oriented design
3 Being able to convert an object oriented design to a working computer program in Java and to test this program critically
4 To have insight into the available "design patterns " for software design and for typical software problems
5 To be able to know when which pattern is useful
6 Being able to develop programs using patterns in a suitable way
7 To be able to 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
Lecture, seminar: practical PC room classes

Extra information on the teaching methods
• Lectures (24 hrs)
• Labs (36 hrs): individual work on PC; presence required

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
"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

References

Course content-related study coaching
The student can always make an appointment with the teachers

Evaluation methods
end-of-term evaluation and continuous assessment

Examination methods in case of periodic evaluation during the first examination period
Written examination, skills test

Examination methods in case of periodic evaluation during the second examination period
Written examination, skills test

Examination methods in case of permanent evaluation
Skills test

Possibilities of retake in case of permanent evaluation
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

(Approved)