<table>
<thead>
<tr>
<th>Course Title</th>
<th>Software Engineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course Code</td>
<td>76215</td>
</tr>
<tr>
<td>Scientific Sector</td>
<td>INF/01</td>
</tr>
<tr>
<td>Degree</td>
<td>Bachelor in Computer Science</td>
</tr>
<tr>
<td>Semester</td>
<td>2nd Semester</td>
</tr>
<tr>
<td>Year</td>
<td>2nd year</td>
</tr>
<tr>
<td>Credits</td>
<td>6</td>
</tr>
<tr>
<td>Total Lecturing Hours</td>
<td>40</td>
</tr>
<tr>
<td>Total Lab Hours</td>
<td>20</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>Students should have done the following courses: Computer Programming, Programming Project</td>
</tr>
<tr>
<td>Course Page</td>
<td>ole.unibz.it</td>
</tr>
</tbody>
</table>

**Specific Educational Objectives**

The course introduces the state-of-the-art in software engineering. It aims to demonstrate how this is transferred into practically applicable knowledge and skills for software development.

**Lecturer**

Claus Pahl

**Scientific Sector of the Lecturer**

INF/01

**Teaching Language**

English

**Office Hours**

After the lecture/lab times or by appointment (email). Faculty of CS, Piazza Domenicani 3, Office 1.11

**Teaching Assistant**

Same as lecturer

**Office Hours**


LIST OF TOPICS COVERED

- Software life-cycle: principles and methodologies
- Software processes and software project management
- Requirements engineering: elicitation and modeling
- System modeling and construction: UML, design patterns
- Software testing: principles and techniques
- Software management and evolution

TEACHING FORMAT

Frontal lectures, exercises, projects.

LEARNING OUTCOMES

Knowledge and understanding
- Know in detail principles, techniques and methods of planning, designing, developing and maintaining software;

Applying knowledge and understanding
- Be able to apply the own knowledge to the analysis, design, development and testing of information systems which satisfy given requirements;
- be able to solve typical problems in computer science, such as the definition of requirements, the analysis of possible methods for a solution, the selection of methods and tools as well as their application;
- be able to evaluate the quality of information systems and to identify critical aspects;
- be able to apply the own knowledge in different working contexts;

Making judgments
- be able to take the responsibility for software development projects

Communication skills
- be able to explain a project activity or a scientific study, also to non-experts
- be able to work in teams to implement software systems

Ability to learn
- have acquired learning capabilities that enable them to carry out project activities in companies, public institutions or in distributed development communities
- be able to learn cutting edge IT technologies and their strengths and limitations

ASSESSMENT

Written and project work: written exam with verification questions and written project report done in groups.

In case of a positive mark the project will count for all 3 regular exam sessions.

Projects have to be submitted BEFORE the final exam at the end of the semester; otherwise the exam cannot be registered.

ASSESSMENT LANGUAGE

English

EVALUATION CRITERIA AND WEIGHTING OF PARTS:

- 70% written exam
### CRITERIA FOR AWARDING MARKS

- 30% exercises/project.

**Criteria:**

Relevant for assessment of project and exam:
- clarity of answers,
- mastery of language,
- skills in critical thinking
- ability to summarize, evaluate, and establish relationships between topics,
- technical competence

Relevant for project assessment:
- ability to work in a team,
- creativity,
- development skills

### REQUIRED READINGS

The course will be based on lecture notes

### SUPPLEMENTARY READINGS


### SOFTWARE USED

Software Modelling (e.g. Argo UML, Papyrus, StarUML, UMLet)