# Syllabus

## *Course Description*

| | |
|---|---|
| **Course Title** | Agile Software Engineering |
| **Course Code** | 76106 |
| **Course Title Additional** | |
| **Scientific-Disciplinary Sector** | INFO-01/A |
| **Language** | English |
| **Degree Course** | Master in Software Engineering |
| **Other Degree Courses (Loaned)** | |
| **Lecturers** | Prof. Xiaofeng Wang,<br>Xiaofeng.Wang@unibz.it<br>https://www.unibz.it/en/faculties/engineering/academic-staff/person/31445 |
| **Teaching Assistant** | |
| **Semester** | First semester |
| **Course Year/s** | 1 |
| **CP** | 12 |
| **Teaching Hours** | 80 |
| **Lab Hours** | 40 |
| **Individual Study Hours** | 180 |
| **Planned Office Hours** | 36 |
| **Contents Summary** | The course belongs to the type "caratterizzanti – discipline informatiche". |
| **Course Topics** | The course aims to equip students with both an agile mindset and practical professional skills essential for modern software engineering. It covers the foundations and core principles of agile software development, exploring various agile approaches and applying key engineering and project management practices in real-world contexts. Emphasis is placed on teamwork, agile collaboration techniques, and the challenges of scaling agile methods in distributed and large-scale projects. In parallel, the |

| | |
|---|---|
| | course introduces students to tools and techniques widely used in DevOps environments, including virtualization, containerization, microservice architectures, automation of the software lifecycle, continuous integration, deployment and delivery, as well as log, configuration, and system monitoring. Through this integrated approach, students gain a comprehensive understanding of contemporary software development practices. |
| **Keywords** | Agility, DevOps, Sustainability, Quality. |
| **Recommended Prerequisites** | |
| **Propaedeutic Courses** | |
| **Teaching Format** | The course combines interactive lectures with practical project work to provide both theoretical foundations and hands-on experience in agile software development. |
| **Mandatory Attendance** | Attendance is not compulsory, but non-attending students have to contact the lecturer at the start of the course to agree on the modalities of the independent study. |
| **Specific Educational Objectives and Learning Outcomes** | Knowledge and understanding<br>D1.5 know the fundamentals, techniques, and methods of design, customisation and implementation of software to support the automation of new-generation software systems for industrial production, company business, education, and society.<br>D1.6 understand the elements of corporate and professional culture.<br><br>Applying knowledge and understanding<br>D2.2 know how to design and carry out empirical studies of software systems in order to acquire measurements of their behaviour and evaluate experimental hypotheses in different application fields, such as business, industry, education, or research.<br>D2.4 ability to define an innovative technical solution to an application problem that respects technical, functional, and organisational constraints and requirements.<br><br>Making judgements<br>D3.2 ability to plan and re-plan a technical project activity and to carry it out within the defined deadlines and objectives.<br>D3.3 ability to define work objectives compatible with the available |

| | |
|---|---|
| | time and resources.<br>D3.4 ability to reconcile conflicting project objectives, find acceptable compromises within the limits of cost, resources, time, knowledge, or risk.<br>D3.5 ability to work with broad autonomy, taking responsibility for projects and structures.<br><br>Communication skills<br>D4.3 ability to work and co-ordinate the work of a multi-disciplinary project team, to identify activities aimed at achieving the project objectives.<br>D4.4 ability to prepare and deliver presentations with technical content in English for diverse audiences.<br>D4.5 ability to interact and collaborate in the realisation of a project or research with peers and experts.<br><br>Learning skills<br>D5.2 ability to independently keep up to date with developments in the most important fields of information technology.<br>D5.3 ability to extend incomplete knowledge with regard to the final objective of the project, in the context of a problem-solving activity. |
| **Specific Educational Objectives and Learning Outcomes (additional info.)** | |
| **Assessment** | The assessment for both courses in this module consists of two components: a project (50%) and an oral exam (50%). Attending students complete a team-based development project, while non-attending students analyze an existing one. The oral exam evaluates individual theoretical understanding and the ability to discuss project outcomes. A passing project evaluation is required to access the oral exam, and both components must be passed to complete the module. A positively assessed project remains valid for three sessions.<br>This assessment structure supports the learning outcomes of this course as follows. It contributes to the acquisition of knowledge and understanding (D1.5, D1.6) by engaging students in the application of software development techniques and fostering reflection on corporate and professional contexts. It enhances the |

| | |
|---|---|
| | ability to apply knowledge (D2.2, D2.4) by requiring the design and empirical evaluation of software solutions that respect technical and organizational constraints. The project work also develops judgment skills (D3.2–D3.5) as students plan, manage, and adapt project activities under real-world limitations while taking increasing responsibility for their work. Communication skills (D4.3–D4.5) are strengthened through teamwork, technical discussions, and oral presentations in English. Finally, the course promotes learning skills (D5.2, D5.3) by encouraging students to independently acquire new knowledge and address open problems throughout the project and oral examination. |
| **Evaluation Criteria** | For both attending and non-attending students, the project work is evaluated based on the quality of the solution or analysis. For attending students, the quality of teamwork is also considered. The oral exam evaluates the ability to summarize, assess, and relate different topics, along with the clarity and precision of the responses. |
| **Required Readings** | (See module descriptions) |
| **Supplementary Readings** | (See module descriptions) |
| **Further Information** | (See module descriptions) |
| **Sustainable Development Goals (SDGs)** | Decent work and economic growth, Responsible consumption and production, Industry, innovation and infrastructure |

# *Course Module*

| | |
|---|---|
| **Course Constituent Title** | Agile Processes and Practices |
| **Course Code** | 76106A |
| **Scientific-Disciplinary Sector** | INFO-01/A |
| **Language** | English |
| **Lecturers** | |
| **Teaching Assistant** | |
| **Semester** | First semester |
| **CP** | 6 |
| **Responsible Lecturer** | |
| **Teaching Hours** | 40 |

| | |
|---|---|
| **Lab Hours** | 20 |
| **Individual Study Hours** | 90 |
| **Planned Office Hours** | 18 |
| **Contents Summary** | • Origin and evolution of agile software development<br>• Major agile frameworks and key agile practices<br>• Scaling agile: distributed and/or large agile software development<br>People-centric and teamwork in agile software development<br>• Continuous experimentation using agile approaches<br>• AI-enabled agile processes |
| **Course Topics** | The Agile Software Development course aims to instill an agile mindset in future software engineers and enhance their ability to work effectively on software development projects using agile methods. The course focuses on understanding the foundations and core principles of agile software development, exploring various agile approaches, and applying key engineering and project management practices in real-world contexts. It also emphasizes improving teamwork through agile collaboration techniques and addresses how to scale agile development beyond its typical settings, including in distributed and large-scale projects. |
| **Teaching Format** | The course combines interactive lectures with practical project work to provide both theoretical foundations and hands-on experience in agile software development. |
| **Required Readings** | - Agile Manifesto: http://agilemanifesto.org/<br>- Agile Essentials on Agile Alliance website: https://www.agilealliance.org/agile-essentials/<br>- Modern Agile: https://modernagile.org/<br>- Subject Librarian: David Gebhardi, David.Gebhardi@unibz.it |
| **Supplementary Readings** | - Highsmith, Jim. Agile Software Development Ecosystems. Boston, 2002.<br>- Research papers on agile software development, which will be distributed during the lectures |

# *Course Module*

| | |
|---|---|
| **Course Constituent Title** | Continuous Integration and Delivery |

| | |
|---|---|
| **Course Code** | 76106B |
| **Scientific-Disciplinary Sector** | INFO-01/A |
| **Language** | English |
| **Lecturers** | |
| **Teaching Assistant** | |
| **Semester** | First semester |
| **CP** | 6 |
| **Responsible Lecturer** | |
| **Teaching Hours** | 40 |
| **Lab Hours** | 20 |
| **Individual Study Hours** | 90 |
| **Planned Office Hours** | 18 |
| **Contents Summary** | • Configuration Management<br>• Containerization with Docker & Kubernetes<br>• Applied Microservice-oriented Software Engineering<br>• Monolith to Microservices Migration<br>• Continuous Integration & Delivery Techniques<br>• DevOps as a Software Development Paradigm |
| **Course Topics** | The course is designed to equip students with practical professional skills relevant to modern software engineering. It focuses on the application of development techniques and tools commonly used in DevOps environments, including virtualization and containerization, microservice architectures, automation of the software lifecycle, continuous integration, deployment and delivery, as well as log management, configuration management, and system monitoring. |
| **Teaching Format** | The course combines interactive lectures with practical project work to provide both theoretical foundations and hands-on experience in agile software development. |
| **Required Readings** | Lecture notes will be handed out during the course.<br>Subject Librarian: David Gebhardi, David.Gebhardi@unibz.it |
| **Supplementary Readings** | - Robert C Martin: Clean Architecture: A Craftsman's Guide to Software Structure and Design. Pearson (2017) |

| | |
|---|---|
| | - Vaughn Vernon: Domain-Driven Design Distilled. Addison-Wesley Professional (2016) |