Syllabus

Course description

Course title	Agile Software Engineering
Course code	76106
Course title additional	/
Scientific sector	INF/01
Teaching language(s)	English
Degree course	Master in Software Engineering
Other degree courses (loaned)	/
Lecturer(s)	Prof. Xiaofeng Wang, xwang@unibz.it, room BZ B1 4.33 Prof. Andrea Janes, ajanes@unibz.it, room BZ B1 4.24
Teaching assistant(s)	/
Semester	1
Course year	1
CP	12
Teaching hours	80
Lab hours	40
Individual study	60
Planned office hours	36
Contents summary	The course belongs to the type "caratterizzanti – discipline informatiche".
Course content	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
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 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. D1.6 understand the elements of corporate and professional culture. Applying knowledge and understanding 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. D2.4 ability to define an innovative technical solution to an application problem that respects technical, functional, and organisational constraints and requirements. Making judgements D3.2 ability to plan and re-plan a technical project activity and to carry it out within the defined deadlines and objectives. D3.3 ability to define work objectives compatible with the available time and resources. D3.4 ability to reconcile conflicting project objectives, find acceptable compromises within the limits of cost, resources, time, knowledge, or risk. D3.5 ability to work with broad autonomy, taking responsibility for projects and structures. Communication skills 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. D4.4 ability to prepare and deliver presentations with technical content in English for diverse audiences. D4.5 ability to interact and collaborate in the realisation of a project or research with peers and experts. Learning skills D5.2 ability to independently keep up to date with developments in the

	most important fields of information technology. 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 objective and learning outcomes (additional information)	/
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. 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; industry, innovation and infrastructure; responsible consumption and production

Course module

Course constituent title	Agile Software Engineering M1 - Agile Processes and Practices
Course code	76106A
Scientific sector	INF/01

Teaching language(s)	English
Lecturer(s)	Prof. Xiaofeng Wang, xwang@unibz.it, room BZ B1 4.33
Teaching assistant(s)	/
Semester	1
CP	6
Responsible lecturer	Prof. Xiaofeng Wang, xwang@unibz.it, room BZ B1 4.33
Teaching hours	40
Lab hours	20
Individual study	90
Planned office hours	18
Contents summary	Origin and evolution of agile software development Major agile frameworks and key agile practices Scaling agile: distributed and/or large agile software development People-centric and teamwork in agile software development Continuous experimentation using agile approaches AI-enabled agile processes
Course content	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/ Agile Essentials on Agile Alliance website: https://www.agilealliance.org/agile-essentials/ Modern Agile: https://modernagile.org/ Subject Librarian: David Gebhardi, David.Gebhardi@unibz.it
Supplementary readings	Highsmith, Jim. Agile Software Development Ecosystems. Boston, 2002. Research papers on agile software development, which will be distributed during the lectures

Freie Universität Bozen
Libera Università di Bolzano
Università Liedia de Bulsan

Course module

Course constituent title	Agile Software Engineering M2 - Continuous Integration and Delivery
Course code	76106B
Scientific sector	INF/01
Teaching language(s)	English
Lecturer(s)	Prof. Andrea Janes, ajanes@unibz.it, room BZ B1 4.24
Teaching assistant(s)	/
Semester	1
CP	6
Responsible lecturer	Prof. Andrea Janes, ajanes@unibz.it, room BZ B1 4.24
Teaching hours	40
Lab hours	20
Individual study	90
Planned office hours	18
Contents summary	Configuration Management Containerization with Docker & Kubernetes Applied Microservice-oriented Software Engineering Monolith to Microservices Migration Continuous Integration & Delivery Techniques DevOps as a Software Development Paradigm
Course content	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. 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)