

# Syllabus

## *Kursbeschreibung*

<b>Titel der Lehrveranstaltung</b>	Software Systems Engineering
<b>Code der Lehrveranstaltung</b>	76265
<b>Zusätzlicher Titel der Lehrveranstaltung</b>	
<b>Wissenschaftlich-disziplinärer Bereich</b>	
<b>Sprache</b>	Englisch; Italienisch
<b>Studiengang</b>	Bachelor in Informatik
<b>Andere Studiengänge (gem. Lehrveranstaltung)</b>	
<b>Dozenten/Dozentinnen</b>	Prof. Barbara Russo, Barbara.Russo@unibz.it <a href="https://www.unibz.it/en/faculties/engineering/academic-staff/person/2242">https://www.unibz.it/en/faculties/engineering/academic-staff/person/2242</a> Dr. Eduardo Martins Guerra, Eduardo.MartinsGuerra@unibz.it <a href="https://www.unibz.it/en/faculties/engineering/academic-staff/person/43879">https://www.unibz.it/en/faculties/engineering/academic-staff/person/43879</a>
<b>Wissensch. Mitarbeiter/Mitarbeiterin</b>	
<b>Semester</b>	Zweites Semester
<b>Studienjahr/e</b>	2
<b>KP</b>	12
<b>Vorlesungsstunden</b>	70
<b>Laboratoriumsstunden</b>	50
<b>Stunden für individuelles Studium</b>	180
<b>Vorgesehene Sprechzeiten</b>	
<b>Inhaltsangabe</b>	<ul style="list-style-type: none"> <li>- Software and systems architecture principles</li> <li>- Software architecture design process</li> <li>- Architectural components and frameworks</li> </ul>

	<ul style="list-style-type: none"> <li>- Approaches for architectural partitioning</li> <li>- Architectural patterns and styles</li> <li>- Integrating AI Components into Architectural Designs</li> <li>- Verification and validation</li> <li>- Techniques for black-box and white-box testing</li> <li>- Techniques and tools for test automation</li> <li>- Integration and regression testing</li> <li>- Web testing</li> <li>- AI for test case generation</li> </ul>
<b>Themen der Lehrveranstaltung</b>	<p>The objective of the Software Systems Architecture Module is to provide students with a solid understanding of the role of software architecture within the software development lifecycle. Students will learn to design architectures using established patterns and best practices, gain an overview of various architectural styles and emerging trends, evaluate and balance trade-offs related to quality attributes, and apply different architectural approaches to the development of high-quality software.</p> <p>Tools and Techniques for Software Testing is designed to equip students with the ability to select, use, customize, and deploy software testing tools and techniques. It also enables them to configure and integrate these tools to support testing activities throughout the collaborative software development process.</p>
<b>Stichwörter</b>	Software testing, prompt engineering for testing, agentic systems for testing
<b>Empfohlene Voraussetzungen</b>	Basic courses in Programming and Software Engineering. Familiarity with UML and the basics of object-oriented programming.
<b>Propädeutische Lehrveranstaltungen</b>	
<b>Unterrichtsform</b>	The course includes frontal lectures and lab exercises.
<b>Anwesenheitspflicht</b>	Attendance is not compulsory. Non-attending students have to contact the lecturer at the start of the course to agree on the modalities of the independent study.
<b>Spezifische Bildungsziele und erwartete Lernergebnisse</b>	<p>Knowledge and Understanding</p> <ul style="list-style-type: none"> <li>- D1.8 To have a thorough knowledge of the main fundamentals techniques and methods of software design, development and</li> </ul>

	<p>maintenance</p> <p>Applying knowledge and understanding</p> <ul style="list-style-type: none"> <li>- D2.5 Be able to apply the own knowledge to the , design, development and testing of information systems which satisfy given requirements</li> <li>- D2.7 Be able to conduct simple experiments about information systems by collecting measures about the behaviour of the system.</li> <li>- D2.10 Be able to solve typical problems in computer science based on software engineering methodologies, such as the definition of requirements, the of possible methods for a solution, the selection of the most appropriate methods and tools as well as their application</li> <li>- D2.11 Be able to evaluate the quality of information systems and to identify critical aspects.</li> </ul> <p>Ability to make judgments</p> <ul style="list-style-type: none"> <li>- D3.1 Be able to collect and interpret useful data and to judge information systems and their applicability.</li> <li>- D3.2 Be able to work autonomously according to the own level of knowledge and understanding.</li> <li>- D3.3 Be able to take the responsibility for development of projects or IT consulting.</li> </ul> <p>Communication skills</p> <ul style="list-style-type: none"> <li>- D4.1 Be able to use one of the three languages English, Italian and German, and be able to use technical terms and communication appropriately.</li> <li>- D4.3 Be able to negotiate with a customer for the definition of the pre-requisites and features of information systems.</li> <li>- D4.4 Be able to structure and write technical documentation.</li> </ul> <p>Learning skills</p> <ul style="list-style-type: none"> <li>- D5.1 Have developed learning capabilities to pursue further studies with a high degree of autonomy.</li> <li>- D5.2 Have acquired learning capabilities that enable to carry out project activities in companies, public institutions or in distributed development communities.</li> <li>- D5.3 Be able to follow the fast technological evolution and to</li> </ul>
--	---

	learn cutting edge IT technologies and innovative aspects of last generation information systems.
<b>Spezifisches Bildungsziel und erwartete Lernergebnisse (zusätzliche Informationen)</b>	
<b>Art der Prüfung</b>	<p>A final unified written exam will be performed with verification questions from both modules. Each module will have the following additional activities during the course:</p> <p>Software Systems Architecture: Lab assessment is composed by assignments that should be performed and delivered in each week. Optional activities might worth extra points in the final module grade.</p> <p>Tools and Techniques for Software Testing: Project work with group assignments</p>
<b>Bewertungskriterien</b>	<p>The written exam evaluates the student's ability to master the course terminology, assess tools and techniques in relation to their specific domain of use and technical details, solve exercises, and clearly summarize theoretical concepts. The assessment of project and lab work focuses on timely completion of assignments, the delivery of functional solutions, and the development or customization of high-quality software that meets the specified requirements.</p> <p>The final grade is calculated as the average of the grades obtained in each module. For Software Systems Architecture, the grade is based on two components: 50% from lab work and 50% from the module-specific questions in the final written exam. Optional activities within the module may provide extra credit. For Tools and Techniques for Software Testing, the grade is composed of 80% from project work and 20% from the relevant questions in the written exam. Completion and positive evaluation of the project is a prerequisite for accessing the written exam.</p> <p>For non-attending students, module instructors may offer two options: either complete lab assessments or projects under the same conditions as attending students, or undergo an additional</p>

	<p>evaluation based on questions related to lab or project content.</p> <p>Students must pass both modules to pass the course. A positive grade in one module remains valid for all three regular exam sessions within the academic year.</p>
<b>Pflichtliteratur</b>	<ul style="list-style-type: none"> <li>• Robert Martin. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Pearson, London, England, 1st edition, September 2017. ISBN 978-0-13-449416-6.</li> <li>• Mark Richards. Software Architecture Patterns. O'Reilly Media, Inc., 2015. ISBN 978-1-4919-2540-9.</li> <li>• Paul C. Jorgensen, Software Testing: A Craftsman's Approach, Fourth Edition, CRC Press, 2013 ISBN 1466560681</li> <li>• Mauricio Anniche Effective Software Testing: A developer's guide ; 2022 Manning Publications ISBN 9781633439931  <a href="https://unibz.primo.exlibrisgroup.com/discovery/fulldisplay?docid=cdi_proq">https://unibz.primo.exlibrisgroup.com/discovery/fulldisplay?docid=cdi_proq</a> </li> </ul>
<b>Weiterführende Literatur</b>	<p>Len Bass, Paul Clements, and Rick Kazman. Software Architecture in Practice. Addison-Wesley Professional, Harlow, 3rd edition, September 2012. ISBN 978-0-321-81573-6.</p>
<b>Weitere Informationen</b>	<ul style="list-style-type: none"> <li>- Java (<a href="https://openjdk.org">https://openjdk.org</a>)</li> <li>- A Java IDE (e.g., <a href="https://eclipseide.org">https://eclipseide.org</a>)</li> <li>- Git (<a href="https://git-scm.com">https://git-scm.com</a>)</li> <li>- Maven (<a href="https://maven.apache.org">https://maven.apache.org</a>)</li> <li>- Issue trackers</li> <li>- JaCoCo (<a href="https://www.eclemma.org/jacoco/">https://www.eclemma.org/jacoco/</a>)</li> <li>- FindBugs (<a href="https://findbugs.sourceforge.net">https://findbugs.sourceforge.net</a>)</li> <li>- JUnit 5 (<a href="https://junit.org">https://junit.org</a>)</li> <li>- FitNesse (<a href="https://fitnesse.org">https://fitnesse.org</a>)</li> </ul>
<b>Ziele für nachhaltige Entwicklung (SDGs)</b>	Hochwertige Bildung

## *Kursmodul*

<b>Titel des Bestandteils der Lehrveranstaltung</b>	Software Systems Architecture
<b>Code der Lehrveranstaltung</b>	76265A
<b>Wissenschaftlich-disziplinärer Bereich</b>	IINF-05/A
<b>Sprache</b>	Englisch

<b>Dozenten/Dozentinnen</b>	Dr. Eduardo Martins Guerra, Eduardo.MartinsGuerra@unibz.it <a href="https://www.unibz.it/en/faculties/engineering/academic-staff/person/43879">https://www.unibz.it/en/faculties/engineering/academic-staff/person/43879</a>
<b>Wissensch. Mitarbeiter/Mitarbeiterin</b>	
<b>Semester</b>	Zweites Semester
<b>KP</b>	6
<b>Verantwortliche/r Dozent/in</b>	
<b>Vorlesungsstunden</b>	40
<b>Laboratoriumsstunden</b>	20
<b>Stunden für individuelles Studium</b>	90
<b>Vorgesehene Sprechzeiten</b>	
<b>Inhaltsangabe</b>	<ul style="list-style-type: none"> <li>- Software and systems architecture principles</li> <li>- Software architecture design process</li> <li>- Architectural components and frameworks</li> <li>- Approaches for architectural partitioning</li> <li>- Architectural patterns and styles</li> <li>- Integrating AI Components into Architectural Designs</li> </ul>
<b>Themen der Lehrveranstaltung</b>	my SNS HomeSyllabiSyllabi OverviewCourse 76265Module 76265A Module (course constituents)
<b>Unterrichtsform</b>	Der Kurs umfasst Frontalunterricht und Laborübungen.
<b>Pflichtliteratur</b>	<p>Robert Martin. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Pearson, London, England, 1st edition, September 2017. ISBN 978-0-13-449416-6.</p> <p>Mark Richards. Software Architecture Patterns. O'Reilly Media, Inc., 2015. ISBN 978-1-4919-2540-9.</p>

<b>Weiterführende Literatur</b>	Kyle Brown, Bobby Woolf, and Joseph Yoder. Cloud Application Architecture Patterns: Designing, Building, and Modernizing for the Cloud, 2025, Oreilly & Associates Inc, ISBN 978-1098116903
---------------------------------	---

## *Kursmodul*

<b>Titel des Bestandteils der Lehrveranstaltung</b>	Tools and Techniques for Software Testing
<b>Code der Lehrveranstaltung</b>	76265B
<b>Wissenschaftlich-disziplinärer Bereich</b>	INFO-01/A
<b>Sprache</b>	Italienisch
<b>Dozenten/Dozentinnen</b>	Prof. Barbara Russo, Barbara.Russo@unibz.it <a href="https://www.unibz.it/en/faculties/engineering/academic-staff/person/2242">https://www.unibz.it/en/faculties/engineering/academic-staff/person/2242</a>
<b>Wissensch. Mitarbeiter/Mitarbeiterin</b>	
<b>Semester</b>	Zweites Semester
<b>KP</b>	6
<b>Verantwortliche/r Dozent/in</b>	
<b>Vorlesungsstunden</b>	30
<b>Laboratoriumsstunden</b>	30
<b>Stunden für individuelles Studium</b>	90
<b>Vorgesehene Sprechzeiten</b>	
<b>Inhaltsangabe</b>	<ul style="list-style-type: none"> <li>- Verification and validation</li> <li>- Techniques for black-box and white-box testing</li> <li>- Techniques and tools for test automation</li> <li>- Integration and regression testing</li> <li>- Web testing</li> <li>- AI for test case generation</li> </ul>
<b>Themen der Lehrveranstaltung</b>	Tools and Techniques for Software Testing is designed to equip students with the ability to select, use, customize, and deploy software testing tools and techniques. It also enables them to

	configure and integrate these tools to support testing activities throughout the collaborative software development process and agentic systems.
<b>Unterrichtsform</b>	The course includes frontal lectures and lab exercises.
<b>Pflichtliteratur</b>	<ul style="list-style-type: none"> <li>· Paul C. Jorgensen, Software Testing: A Craftsman's Approach, Fourth Edition, CRC Press, 2013 ISBN 1466560681</li> <li>· Mauricio Aniche Effective Software Testing: A developer's guide, 2022 Manning Publications ISBN 9781633439931  <a href="https://unibz.primo.exlibrisgroup.com/discovery/fulldisplay?docid=cdi_proquest_e">https://unibz.primo.exlibrisgroup.com/discovery/fulldisplay?docid=cdi_proquest_e</a> </li> </ul>
<b>Weiterführende Literatur</b>	