

LEHRVERANSTALTUNGSBESCHREIBUNG

Software Architektur und Programmierumgebungen

Akademisches Jahr 2025-2026

1. Angaben zum Programm

1.1. Hochschuleinrichtung	Universitatea Babes-Bolyai
1.2. Fakultät	Mathematik und Informatik
1.3. Department	Informatik
1.4. Fachgebiet	Informatik
1.5. Studienform	Bachelor
1.6. Studiengang / Qualifikation	Informatik in deutscher Sprache
1.7. Form des Studiums	Präsenzstudium

2. Angaben zum Studienfach

2.1. LV-Bezeichnung		Software Architektur und Programmierumgebungen					Code der LV	MLG5013
2.2. Lehrverantwortlicher – Vorlesung			Prof. Dr. Holger Klus					
2.3. Lehrverantwortlicher – Seminar			Prof. Dr. Holger Klus					
2.4. Studienjahr	2	2.5. Semester	2	2.6. Prüfungsform	E	2.7. Art der LV		Pflichtfach

3. Geschätzter Workload in Stunden

3.1. SWS	4	von denen: 3.2 Vorlesung	2	3.3. Seminar/Übung/Projekt	2
3.4. Gesamte Stundenanzahl im Lehrplan	56	von denen: 3.5 Vorlesung	28	3.6 Seminar/Übung/Projekt	28
Verteilung der Studienzeit:					Std.
Studium nach Handbücher, Kursbuch, Bibliographie und Mitschriften					24
Zusätzliche Vorbereitung in der Bibliothek, auf elektronischen Fachplattformen und durch Feldforschung					25
Vorbereitung von Seminaren/Übungen, Präsentationen, Referate, Portfolios und Essays					36
Tutoriat					5
Prüfungen					4
Andere Tätigkeiten:					
3.7. Gesamtstundenanzahl Selbststudium			94		
3.8. Gesamtstundenanzahl / Semester			150		
3.9. Anrechnungspunkte			6		

4. Voraussetzungen (falls zutreffend)

4.1. zur Lehrveranstaltung	• Fähigkeit, in einer höheren Programmiersprache zu programmieren • Fortgeschrittene Programmierungsmethoden • Datenbanken • Betriebssysteme.
4.2. kompetenzbezogene	• Konzepte über Datenbanken • Konzepte über Computernetzwerke

5. Bedingungen (falls zutreffend)

5.1. zur Durchführung der Vorlesung	Vorlesungsraum, Beamer, Laptop
5.2. zur Durchführung des Seminars / der Übung	• Labor mit Computers ausgerichtet • Programmierungsumgebungen für Java und .NET

6.1. Spezifische erworbene Kompetenzen¹

Berufliche/Wesentliche Kompetenzen	K2.1 Identifizierung geeigneter Methoden für die Entwicklung von Softwaresystemen K2.2 Identifizierung und Erklärung geeigneter Mechanismen für die Spezifizierung von Softwaresystemen K2.3 Benutzung der Methoden, Spezifizierungsmechanismen und Entwurfsmedien für die Entwicklung von Software-Anwendungen K2.4 Benutzung von geeigneten Kriterien und Methoden für die Auswertung von Software-Anwendungen K2.5 Entwurf von spezifischen Software-Anwendungen
Transversale Kompetenzen	TK1 Anwendung der Regeln für gut organisierte und effiziente Arbeit, für verantwortungsvolle Einstellungen gegenüber der Didaktik und der Wissenschaft, für kreative Förderung des eigenen Potentials, mit Rücksicht auf die Prinzipien und Normen der professionellen Ethik TK2 Effizienter Ablauf der Tätigkeiten in einer interdisziplinären Gruppe, das Entwickeln der Kapazitäten für empathische zwischenmenschliche Kommunikation, Verknüpfung und Zusammenarbeit mit unterschiedlichen Gruppen TK3 Anwendung von effizienten Methoden und Techniken für Lernen, Informieren und Recherchieren, für das Entwickeln der Kapazitäten der praktischen Umsetzung der Kenntnisse, der Anpassung an die Bedürfnisse einer dynamischen Gesellschaft, der Kommunikation in rumänischer Sprache und in einer internationalen Verkehrssprache

6.2. Lernergebnisse

Kenntnisse	Der/Die Studierende kennt: branchenübliche Techniken für Design, Implementierung, Test und Wartung komplexer Softwaresysteme. Die erworbenen Kenntnisse umfassen den Projektlebenszyklus, eine kleine Anzahl von UML-Diagrammen, Anforderungserhebung, die Arbeit mit Tasks (Aufgaben) sowie die Verwendung von Git im Team.
Fähigkeiten	Der/Die Studierende ist in der Lage, in kleinen bis mittelgroßen Teams (6-30 Teammitgliedern) zu arbeiten. Der/Die Studierende hat die folgenden Fähigkeiten erworben oder vertieft: Requirements Engineering (Anforderungstechnik), UML, Teamarbeit, Git, GUI/Design, Softwarearchitekturen, C# + .NET, Codequalität, Kommunikationsfähigkeit, Unit-Testing, Datenbankzugriff, Webentwicklung, Entwurfsmuster (Design Patterns), Deployment, Clean Code.
Verantwortung und Autonomie	Der/die Studierende verfügt über die Fähigkeit, sowohl selbstständig als auch in Teams um Software zu gestlten.

7. Ziele (entsprechend der erworbenen Kompetenzen)

7.1 Allgemeine Ziele der Lehrveranstaltung	• die Befähigung Softwaresysteme zu entwerfen
7.2 Spezifische Ziele der Lehrveranstaltung	• .NET Plattform • Programmierfähigkeiten in C#. • Schablone in client-server und web Anwendungen.

8. Inhalt

8.1 Vorlesung	Lehr-und Lernmethode	Anmerkungen
1. Automatische Konstruktionswerkzeuge. Gradle.	Vortrag, Unterrichtsgespräch, Problematisierung	

¹ Man kann Kompetenzen oder Lernergebnisse, oder beides wählen. Wenn nur eine Option ausgewählt wird, wird die Tabelle für die andere Option gelöscht, und die beibehaltene Option erhält die Nummer 6.

2. Objektorientierte Modelle für Zugang zur Datenbanken • JDBC • ADO.NET	Vortrag, Unterrichtsgespräch, Problematisierung	
3. Inversion of Control Spring	Vortrag, Unterrichtsgespräch, Problematisierung	
4. Klient-Server Anwendungen: der Proxy Design Pattern	Vortrag, Unterrichtsgespräch, Problematisierung	
5. Klient-Server Anwendungen (Fortsetzung): Der Proxy Design Pattern	Vortrag, Unterrichtsgespräch, Problematisierung	
6. Remote Procedure Call • Remoting • RMI • Spring Remoting	Vortrag, Unterrichtsgespräch, Problematisierung	
7. Enterprise Application Integration: Protocol Buffers, gRPC, Thrift	Vortrag, Unterrichtsgespräch, Problematisierung	
8. Object Relational Mapping: Strategien. Hibernate, Entity Framework	Vortrag, Unterrichtsgespräch, Problematisierung	
9. Enterprise Application Integration - Asynchronous messaging systems: Activemq, Rabbitmq, JMS	Vortrag, Unterrichtsgespräch, Problematisierung	
10. REST	Vortrag, Unterrichtsgespräch, Problematisierung	
11. WebSockets	Vortrag, Unterrichtsgespräch, Problematisierung	
12. Entwurf von Web-Anwendungen mit Hilfe von Frameworks	Vortrag, Unterrichtsgespräch, Problematisierung	
13. Sicherheit der Web-Anwendungen	Vortrag, Unterrichtsgespräch, Problematisierung	
14. NoSQL	Vortrag, Unterrichtsgespräch, Problematisierung	
Literatur In deutscher Sprache: 1. J. Staud, Unternehmensmodellierung: Objektorientierte Theorie und Praxis mit UML 2.0, Springer 2010. 2. D. Louis, S. Strasser, C#, M+T Verlag, 2002 3. O. Vogel et al, Software Architektur: Grundlagen – Konzepte – Praxis, Elsevier 2005 4. M. Simons, Moderne Software Architektur mit Spring 5, 2018 5. G. Starke, Effektive Softwarearchitekturen, Hanser 2017. Joseph Albahari and Ben Albahari, C# 4.0 in a Nutshell, Fourth Edition, O'Reilley, 2010. Allgemeine Literatur: 1. Larman, C.: Applying UML and Design Patterns: An Introduction to OO Analysis and Design and Unified Process, Berlin, Prentice Hall, 2002. 2. Fowler, M., Patterns of Enterprise Application Architecture, Addison-Wesley, 2002. 3. Hohpe, G., Woolf, B., Enterprise integration patterns, Addison-Wesley, 2003. 4. ***, Microsoft Developer Network, Microsoft Inc., http://msdn.microsoft.com/ 5. ***, The Java Tutorial, SUN Microsystems, Inc, 2004. http://download.oracle.com/javase/tutorial/		
8.2 Labor	Lehr-und Lernmethode	Anmerkungen

1. Verwendung eines Build Automation Tools. Auswahl des Projektthemas.	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
2. Zugang zu einer relationalen Datenbank erstellen.	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
3. Konfigurieren einer Anwendung mit IoC.	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
4-5. Entwurf und Implementierung der Dienste (Services) mit dem Proxy-Pattern	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
6-7. RMI/Remoting	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
8. Enterprise Application Integration (Protobuf, gRPC, Thrift)	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
9. ORM Werkzeuge	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
10. Asynchronous Messaging Systems	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
11. REST Dienste	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
12. WebSockets	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
13. Web Sicherheit	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
NoSQL	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
1. Verwendung eines Build Automation Tools. Auswahl des Projektthemas.	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
Literatur in deutscher Sprache 1. J. Staud, Unternehmensmodellierung: Objektorientierte Theorie und Praxis mit UML 2.0, Springer 2010. 2. D. Louis, S. Strasser, C#, M+T Verlag, 2002 3. O. Vogel et al, Software Architektur: Grundlagen – Konzepte – Praxis, Elsevier 2005 4. M. Simons, Moderne Software Architektur mit Spring 5, 2018 5. G. Starke, Effektive Softwarearchitekturen, Hanser 2017		

9. Verbindung der Inhalte mit den Erwartungen der Wissensgemeinschaft, der Berufsverbände und der für den Fachbereich repräsentativen Arbeitgeber

Die Vorlesung entspricht den IEEE und ACM Richtlinien für Informatik Curricula. Die Vorlesung ist wichtig für die Software Firmen

10. Prüfungsform

Veranstaltungsart	10.1 Evaluationskriterien		10.2 Evaluationsmethoden	10.3 Anteil an der Gesamtnote
10.4 Vorlesung	der Kenntnisstand in Bezug auf den Lernstoff zur Entwurf einer Verteilten Anwendung		Quiz während der Vorlesung	10%
	Das Anwenden der Konzepte zur Entwurf eines kleinen Klient-Server Systems		Praktische Prüfung	50%
10.5 Übung+Labor	die Fertigkeit, die in den Vorlesungen und Übungen erworbenen Kenntnisse für das Lösen konkreter Aufgaben einzusetzen		Hausaufgaben, entworfene Systeme, Dokumentationen	30%
			Hausaufgaben während der Laborstunden	10%
10.6 Minimale Leistungsstandards				
Für das Bestehen der Prüfung muss die Mindestnote 5 erzielt werden				

11. SDD-Nachhaltigkeits-Logos (Sustainable Development Goals)²

Nicht anwendbar.

Ausgefüllt am:
17.04.2025

Vorlesungsverantwortlicher
Prof. Dr. Holger Klus

Seminarverantwortlicher
Prof. Dr. Holger Klus

Genehmigt im Department am:

...

Departmentleiter
Conf. dr. Adrian STERCA

² Bitte belassen Sie nur die Logos, die entsprechend den [Regularien zu Anwendung der Nachhaltigkeits-Logos im akademischen Betrieb](#) dem jeweiligen Studienfach entsprechen und löschen Sie diejenigen Logos, inklusive das allgemeine *Nachhaltigkeits-Logo* falls dieses nicht zutrifft. Falls keines der Logos für das Studienfach anwendbar ist, löschen Sie alle mit der Angabe „nicht anwendbar“.