Hochschule Niederrhein
University of Applied Sciences

Ingenieurwissenschaften und Informatik
Faculty of Engineering and Computer Science

Master Informatik: Modulhandbuch

Master Informatik (PO2027)

Herausgeber Fachbereich Ingenieurwissenschaften und Informatik
Stand 2026-08-10

Inhaltsverzeichnis

Einleitung

Dieses Modulhandbuch beschreibt den Masterstudiengang Informatik. Es macht transparent, welche fortgeschrittenen fachlichen, methodischen und überfachlichen Kompetenzen die Studierenden im Studienverlauf erwerben und wie diese Kompetenzen in Kernmodulen, Anwendungs- und Wahlbereichen, Softwareentwicklungsprojekt, Forschungs- und Entwicklungsprojekt sowie Masterarbeit aufgebaut, angewendet und integriert werden. Das Modulhandbuch dient Studierenden, Lehrenden und externen Interessierten als Orientierung über Aufbau, Ziele und Inhalte des Studiums.

Studiengangsprofil

Der Masterstudiengang Informatik vertieft und erweitert die im Bachelorstudium erworbenen Informatikkompetenzen. Im Mittelpunkt steht die Fähigkeit, komplexe informatische Problemstellungen eigenständig zu analysieren, geeignete Methoden und Technologien auszuwählen, anspruchsvolle Systeme zu entwerfen und Ergebnisse fachlich fundiert zu bewerten.

Der Studiengang reagiert auf die zunehmende Komplexität softwareintensiver, vernetzter und datengetriebener Systeme. Moderne Informatik umfasst nicht nur Implementierung, sondern auch Architekturentscheidungen, Qualitätssicherung, Betrieb, Sicherheit, Daten- und KI-Methoden, wissenschaftlich fundierte Evaluation sowie verantwortliche Gestaltung technischer Lösungen. Die Studierenden entwickeln deshalb Kompetenzen, die fachliche Tiefe mit methodischer Reflexion, projektorientiertem Arbeiten und gesellschaftlicher Verantwortung verbinden.

Ein besonderes Gewicht liegt auf der Verbindung von anwendungsorientierter Entwicklung und wissenschaftsnaher Arbeitsweise. Die Studierenden bearbeiten fortgeschrittene Informatikthemen, übernehmen Verantwortung in komplexeren Softwareprojekten, formulieren Forschungs- und Entwicklungsfragen und schließen das Studium mit einer eigenständigen Masterarbeit ab.

Kurzporträt

Worum geht es? Der Master Informatik vertieft fortgeschrittene Konzepte, Methoden und Technologien der Informatik und verbindet sie mit Softwareentwicklung, Daten, Künstlicher Intelligenz, Sicherheit, Forschung und verantwortlicher Systemgestaltung.
Wofür braucht man es? Die Kompetenzen werden benötigt, um komplexe informatische Systeme zu analysieren, zu entwerfen, zu implementieren, zu betreiben, wissenschaftlich zu untersuchen und in anspruchsvollen beruflichen oder forschungsnahen Kontexten verantwortlich weiterzuentwickeln.
Wieso ist es interessant? Der Studiengang ermöglicht es, aktuelle Informatikthemen nicht nur anzuwenden, sondern ihre Grundlagen, Grenzen, Wirkungen und Entwicklungsmöglichkeiten kritisch zu verstehen und aktiv mitzugestalten.

Studiengangskonzept

Der Masterstudiengang Informatik ist modular aufgebaut und führt fachliche Vertiefung, projektorientierte Anwendung, außerfachliche Perspektiven und wissenschaftlich fundierte Eigenarbeit zusammen. Die Kernmodule bilden den fachlichen Kern des Masterstudiums und sichern die fachliche Breite und Tiefe des Studiengangs. Sie werden als seminaristische Lehrveranstaltungen im Format 4SL angeboten und behandeln unter anderem effiziente Algorithmen, statistisches Lernen, sichere Anwendungssysteme, moderne KI-Systeme, sprachbasierte Systeme, Computer Vision, High Performance Computing, eingebettete Systeme, Assistenzsysteme, fortgeschrittenes Software Engineering, Deep Learning und Data Engineering. Durch ein leichtes Überangebot, beispielsweise die Wahl von zehn aus zwölf Kernmodulen, entsteht eine begrenzte Wahlmöglichkeit, ohne den gemeinsamen fachlichen Kern des Studiums aufzugeben.

Der Anwendungs- und Kontextbereich ergänzt die Informatikmodule um anerkannte oder besonders ausgewiesene Lehrangebote aus Anwendungsgebieten der Informatik sowie aus gesellschaftlichen, rechtlichen, ethischen, wirtschaftlichen oder organisatorischen Kontexten. Dadurch können Studierende externe, internationale oder wechselnde interne Angebote in ihr Studium einbringen und ihr fachliches Profil um anwendungs- und kontextbezogene Perspektiven erweitern.

Das Softwareentwicklungsprojekt ist als mehrstufiger projektorientierter Studienbereich angelegt. Die Studierenden arbeiten an langlebigen Softwareprodukten, erschließen bestehende Systeme, setzen anspruchsvolle Änderungen um und übernehmen zunehmend technische, organisatorische und wirtschaftliche Verantwortung. Dadurch werden fortgeschrittene Kompetenzen in professioneller Softwareentwicklung, Qualitätssicherung, Teamarbeit, Projektorganisation und Reflexion aufgebaut.

Im Forschungs- und Entwicklungsprojekt bearbeiten die Studierenden eine aktuelle anwendungsorientierte Forschungs- oder Entwicklungsfrage der Informatik. Sie recherchieren den Stand der Technik beziehungsweise Forschung, wählen geeignete Methoden aus, entwickeln oder evaluieren Lösungen und stellen Ergebnisse wissenschaftlich nachvollziehbar dar.

Den Abschluss bildet die Masterarbeit mit Kolloquium. In ihr bearbeiten die Studierenden eigenständig eine anspruchsvolle informatische Fragestellung und zeigen, dass sie fachliche, methodische und überfachliche Kompetenzen des Studiums in einer zusammenhängenden wissenschaftlichen oder wissenschaftlich fundierten anwendungsorientierten Arbeit integrieren können.

Qualifikationsziele

Die Qualifikationsziele beschreiben, welche fachlichen, methodischen, sozialen und kommunikativen Kompetenzen die Studierenden im Studienverlauf aufbauen und in den Modulen anwenden, vertiefen und integrieren.

Ziel	Qualifikationsziel
Z1	Fortgeschrittene Konzepte, Methoden und Denkweisen der Informatik auf komplexe praxisnahe Problemstellungen anwenden, kritisch reflektieren und situationsgerecht kombinieren.
Z2	Mathematische, theoretische, technische und methodische Grundlagen der Informatik vertieft nutzen, um informatische Systeme zu modellieren, zu analysieren, zu entwerfen und fundiert zu bewerten.
Z3	Algorithmen, Datenstrukturen, Architekturen und Programmierparadigmen für anspruchsvolle Problemstellungen auswählen, anpassen, anwenden und bewerten.
Z4	Komplexe Softwaresysteme, vernetzte Anwendungen und datenintensive Systeme unter Berücksichtigung funktionaler und nichtfunktionaler Anforderungen spezifizieren, entwerfen, implementieren, testen, dokumentieren, betreiben und weiterentwickeln.
Z5	Fortgeschrittene Methoden des Software Engineering zielgerichtet einsetzen, für konkrete Projektkontexte angemessen auswählen und anpassen.
Z6	Technologien, Werkzeuge, Architekturen, Softwarekomponenten und IT-Systeme nach fachlichen, technischen, wirtschaftlichen, sicherheitsbezogenen und qualitativen Kriterien auswählen, integrieren und eigene Lösungen begründet beurteilen.
Z7	Daten, Methoden der Künstlichen Intelligenz und Data Science für geeignete Anwendungsfälle auswählen, einsetzen, Ergebnisse kritisch interpretieren und Grenzen, Risiken sowie Anforderungen an eine verantwortungsvolle Nutzung berücksichtigen.
Z8	Anwendungsorientierte Forschungs- und Entwicklungsfragen der Informatik formulieren, geeignete Methoden auswählen, Untersuchungen oder Evaluationen durchführen, Ergebnisse nachvollziehbar darstellen, kritisch einordnen und fachliche Entscheidungen fundiert begründen.
Z9	Komplexere fachliche Aufgaben selbständig und im Team bearbeiten, Projektergebnisse adressatengerecht kommunizieren sowie dabei relevante gesellschaftliche, ethische, rechtliche, ökologische, wirtschaftliche und sicherheitsbezogene Aspekte informatischer Systeme berücksichtigen.

Ziele-Module-Matrix

Die Matrix zeigt, in welchen Modulen die übergreifenden Qualifikationsziele aufgebaut und vertieft werden. A bedeutet Anwendung und Vertiefung, B kennzeichnet Transfer und Beherrschung auf einem höheren Integrationsniveau. x bedeutet abhängig vom konkret gewählten Fach oder Projektthema.

Legende: Z1 = fortgeschrittene Konzepte und Denkweisen anwenden; Z2 = Grundlagen vertieft nutzen; Z3 = Algorithmen, Datenstrukturen, Architekturen und Programmierparadigmen auswählen und bewerten; Z4 = komplexe Systeme spezifizieren, entwickeln, betreiben und weiterentwickeln; Z5 = fortgeschrittenes Software Engineering einsetzen; Z6 = Technologien, Werkzeuge und IT-Systeme auswählen, integrieren und beurteilen; Z7 = Daten, KI und Data Science verantwortungsvoll nutzen; Z8 = Forschungs- und Entwicklungsfragen bearbeiten; Z9 = selbständig und im Team arbeiten, kommunizieren und Verantwortung reflektieren. A = vertiefen/anwenden; B = beherrschen/transferieren; x = abhängig vom konkreten Wahlfach oder Projektthema.

Modul	Z1	Z2	Z3	Z4	Z5	Z6	Z7	Z8	Z9
MIN111: Effiziente Algorithmen	A	B	B			A		A	A
MIN112: Statistical Learning	A	B	A			A	B	A	A
MIN113: Entwurf sicherer Anwendungssysteme	A	A	A	B	B	B		A	B
MIN114: Architektur und Engineering moderner KI-Systeme	B	A	A	B	A	B	B	B	B
MIN115: Sprachbasierte Systeme	A	A	A	B	A	A	B	A	A
MIN116: Computer Vision	A	A	A			A	B	A	A
MIN117: High Performance Computing	A	B	B	A		B		A	A
MIN118: Architektur eingebetteter Systeme	B	A	A	B	A	B		A	B
MIN119: Fortgeschrittene Assistenzsysteme	A	A	A	A	A	A	A	A	B
MIN120: Fortgeschrittenes Software Engineering	B	A	B	B	B	B		A	B
MIN121: Fortgeschrittenes Deep Learning	A	B	A			A	B	B	A
MIN122: Fortgeschrittenes Data Engineering	A	A	A	B	A	B	B	A	A
MIN201: Anwendungs- oder Kontextmodul 1	x							x	B
MIN202: Anwendungs- oder Kontextmodul 2	x							x	B
MIN203: Anwendungs- oder Kontextmodul 3	x							x	B
MIN301: Softwareentwicklungsprojekt 1	B	A	A	B	B	B		A	B
MIN302: Softwareentwicklungsprojekt 2	B	A	B	B	B	B		A	B
MIN303: Softwareentwicklungsprojekt 3	B	A	B	B	B	B		A	B
MIN401: Forschungs- und Entwicklungsprojekt	B	B	B	B	A	B	x	B	B
MIN501: Masterarbeit mit Kolloquium	B	B	B	B	B	B	x	B	B

Studienplan

Die folgenden Studienverlaufspläne zeigen eine empfohlene Verteilung der Module im Vollzeit- und Teilzeitstudium. Sie dienen der Orientierung; verbindlich sind die Regelungen der Prüfungsordnung und die jeweils angebotenen Lehrveranstaltungen.

Abbildung 1: Studienverlaufsplan Master Informatik, Vollzeit

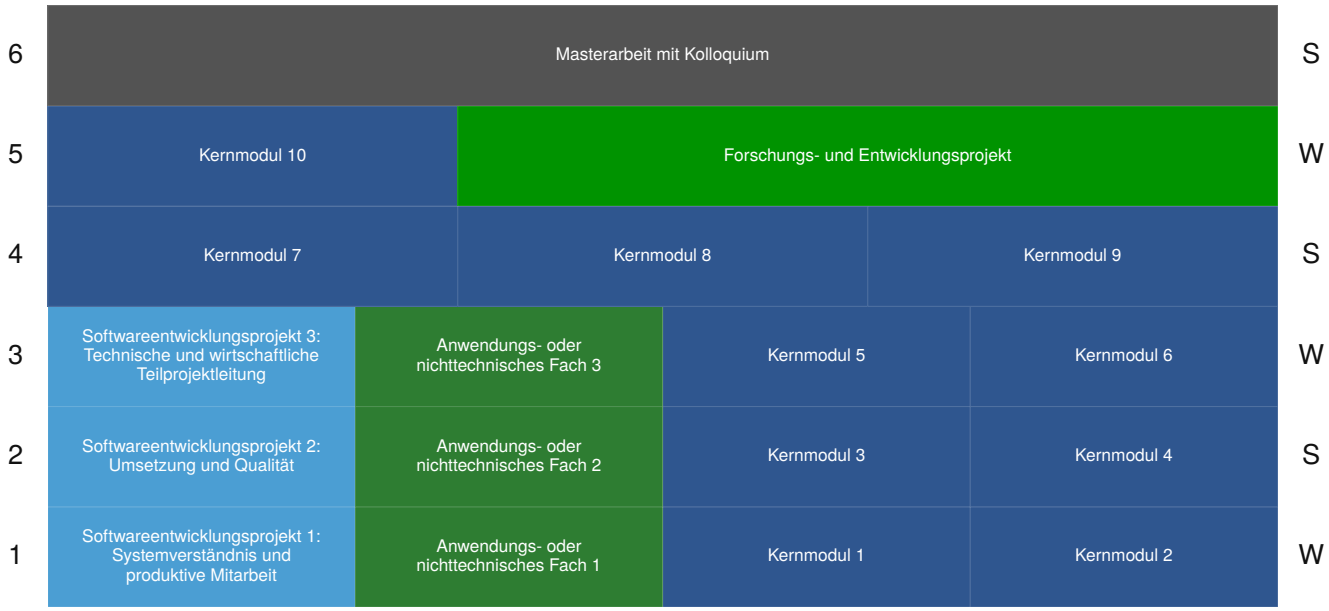

Abbildung 2: Studienverlaufsplan Master Informatik, Teilzeit

Modulbeschreibungen

Dieses Modulhandbuch beschreibt die Module des Masterstudiengangs Informatik mit Angaben zu Verwendbarkeit, Modultyp, Angebotshäufigkeit, Dauer, Credits, Benotung, Semesterwochenstunden, Workload, Voraussetzungen, Lernergebnissen, Inhalten, Prüfungsformen, Literatur und fachlicher Zuordnung. Die Modulbeschreibungen sind nach Modulbereichen geordnet und ermöglichen damit eine Orientierung über Studienverlauf, fachliche Profilbildung, Projektanteile, Forschungs- und Entwicklungsarbeit sowie Abschlussphase.

So liest man eine Modulbeschreibung

Jede Modulbeschreibung folgt dem gleichen Aufbau: Zuerst stehen organisatorische Angaben wie Verantwortliche, Credits, Semesterwochenstunden (SWS) und Workload. Die **Lernergebnisse** sind im Format **WAS** (Kompetenz nach Abschluss), **WOMIT** (Tätigkeiten und Arbeitsweisen, mit denen die Kompetenz erworben wird) und **WOZU** (Nutzen für Studium, Praxis und Beruf) beschrieben. Ein farbiges Abzeichen zeigt, ob ein Modul **Pflicht** oder **Wahlpflicht** ist. Am Ende jedes Moduls gibt ein **Kurzporträt** einen schnellen, allgemeinverständlichen Überblick. Unklare Abkürzungen wie ECTS, SWS oder V/Ü/P/S sind im Glossar am Ende des Handbuchs erklärt.

Kernmodule

Die Kernmodule bilden das fachliche Fundament des Masterstudiengangs Informatik. Sie vertiefen zentrale Konzepte, Methoden und Technologien der Informatik und eröffnen zugleich Spezialisierungsmöglichkeiten in Algorithmen, Software Engineering, sicheren Anwendungssystemen, daten- und KI-basierten Systemen, High Performance Computing, eingebetteten Systemen und Assistenzsystemen.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss der Kernmodule fortgeschrittene informatische Konzepte, Methoden und Werkzeuge auf komplexe Problemstellungen anwenden, geeignete Lösungsansätze auswählen, technische Entscheidungen begründen und Ergebnisse kritisch bewerten. Sie entwickeln dabei Kompetenzen in algorithmischem Denken, daten- und KI-orientierter Systementwicklung, sicherer Softwarearchitektur, Systemintegration, wissenschaftlicher Einordnung und fachlicher Kommunikation.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Modulbereich
  - effiziente Algorithmen, Datenstrukturen und statistische Verfahren analysieren, implementieren und bewerten,
  - sichere, datenintensive, KI-basierte, sprachbasierte und visuelle Anwendungssysteme entwerfen und prototypisch umsetzen,
  - verteilte, parallele, eingebettete und assistive Systeme hinsichtlich Architektur, Qualität, Sicherheit und Einsatzbedingungen untersuchen,
  - fortgeschrittene Software-Engineering-Methoden, Architekturkonzepte, Testverfahren und KI-gestützte Entwicklungswerkzeuge reflektiert einsetzen,
  - aktuelle wissenschaftliche und technische Entwicklungen erschließen, auf konkrete Aufgabenstellungen übertragen und nachvollziehbar dokumentieren.
- **WOZU**: Damit sie anspruchsvolle Entwicklungs-, Analyse-, Forschungs- und Bewertungsaufgaben in der Informatik fachlich fundiert bearbeiten können. Der Modulbereich verbindet theoretische Vertiefung mit anwendungsnaher Systemgestaltung und bereitet auf Softwareentwicklungsprojekt, Forschungs- und Entwicklungsprojekt, Masterarbeit sowie anspruchsvolle berufliche Tätigkeiten vor.

Kurzporträt Kernmodule

Worum geht es? Die Kernmodule zeigen die Breite moderner Informatik auf Masterniveau: von effizienten Algorithmen und Statistical Learning über sichere Anwendungen, KI-Systeme, Sprache und Bildverarbeitung bis zu parallelen, eingebetteten und assistiven Systemen. Die Studierenden vertiefen nicht nur einzelne Technologien, sondern lernen, Architekturen, Methoden und Ergebnisse kritisch einzuordnen.
Wofür braucht man es? Die Kompetenzen werden benötigt, um komplexe informatische Systeme zu entwerfen, zu analysieren, qualitätsgesichert umzusetzen und verantwortungsvoll zu bewerten. Sie sind die fachliche Grundlage für größere Softwareprojekte, Forschungs- und Entwicklungsaufgaben, Masterarbeiten und berufliche Rollen mit technischer Verantwortung.
Wieso ist es interessant? Der Modulbereich macht sichtbar, wie vielfältig und dynamisch Informatik im Master ist. Studierende können anspruchsvolle technische Themen vertiefen, aktuelle Entwicklungen wie KI, sichere Architekturen oder datenintensive Systeme fachlich greifen und eigene Interessen zu einem tragfähigen Kompetenzprofil verbinden.

MIN111: Effiziente Algorithmen

Modulverantwortung	Prof. Dr. Jochen Rethmann	**Lehrperson(en)**	Prof. Dr. Jochen Rethmann
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Efficient Algorithms
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Gute Programmierkenntnisse in mindestens einer Programmiersprache, grundlegende Kenntnisse in den Bereichen Algorithmen und Datenstrukturen, Kombinatorik, Stochastik, Berechenbarkeit, Komplexitätsklassen und Turingmaschinen.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, ausgehend von gelernten Konzepten effiziente Algorithmen und geeignete Datenstrukturen für viele Fragestellungen aus der Praxis zu entwickeln und zu bewerten, verschiedene Klassen von Algorithmen zu differenzieren und sich in neue technisch-wissenschaftliche Themen einzuarbeiten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- verschiedene Algorithmen und Datenstrukturen in einer Programmiersprache implementieren sowie ihre Effizienz vergleichen.
- Algorithmen hinsichtlich Laufzeit und Speicherverbrauch analysieren und geeignete Lösungsansätze für unterschiedliche Problemstellungen vergleichen.
- die Lehrveranstaltungen nachbereiten, Literatur studieren sowie zusätzliche Übungsaufgaben eigenständig lösen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um Programme zu entwickeln, die auch bei großen Eingaben schnell, ressourcenschonend und skalierbar arbeiten. Damit können sie algorithmische Lösungsansätze in Folgemodulen, Projekten und der beruflichen Praxis fundiert auswählen und bewerten.

Inhalte

- Entwurfsmethoden wie bspw. Divide and Conquer, dynamische Programmierung, Greedy, Backtracking
- Graph-Algorithmen wie bspw. Zusammenhangsprobleme, Netzwerkfluss, Matching
- spezielle Graphklassen wie bspw. Co-Graphen, Vergleichbarkeits-, chordale und planare Graphen
- Algorithmische Geometrie wie bspw. Voronoi-Diagramme, konvexe Hülle, Scan-Line-Verfahren
- Exponentialzeit-Algorithmen bspw. für SAT, Independent Set und Vertex-Cover
- Datenstrukturen wie bspw. Union-Find-Strukturen, Fibonacci-Heaps, balancierte Binärbäume, B-Bäume, Tries, Intervallbäume, Segmentbäume, Prioritätssuchbäume
- spezielle Themen wie bspw. randomisierte Algorithmen, Approximations- und Online-Algorithmen

Prüfungsvorleistung

Keine angegeben. #### Prüfungsleistung Klausurarbeit (120 Minuten).

Literatur

- Ottmann, Widmayer: Algorithmen und Datenstrukturen. Spektrum Akademischer Verlag.
- Cormen, Leiserson, Rivest: Introduction to Algorithms. MIT Press.
- Knuth: The Art of Computer Programming. Addison-Wesley.
- Gurski, Rothe, Rothe, Wanke: Exakte Algorithmen für schwere Graphenprobleme.
- Klein: Algorithmische Geometrie. Springer Verlag.
- Hromkovič: Randomisierte Algorithmen. Vieweg + Teubner Verlag.

Kurzporträt MIN111: Effiziente Algorithmen

Worum geht es? Effiziente Algorithmen vertieft den Entwurf und die Analyse leistungsfähiger Verfahren für anspruchsvolle Problemstellungen. Die Studierenden beschäftigen sich mit Entwurfsmethoden, Graphalgorithmen, Datenstrukturen, algorithmischer Geometrie sowie exakten, approximativen und randomisierten Verfahren.
Wofür braucht man es? Die Kompetenzen werden gebraucht, wenn Software nicht nur korrekt, sondern auch schnell, skalierbar und ressourcenschonend sein muss. Sie helfen, Lösungsansätze fundiert auszuwählen, Laufzeit und Speicherbedarf einzuschätzen und technische Grenzen realistisch zu bewerten.
Wieso ist es interessant? Interessant ist das Modul, weil kleine algorithmische Entscheidungen große Auswirkungen auf Machbarkeit und Performance haben können. Viele praktische Probleme wirken zunächst ähnlich, verlangen aber sehr unterschiedliche algorithmische Ideen und Abwägungen.

MIN112: Statistical Learning

Modulverantwortung	Prof. Dr. Christoph Dalitz	**Lehrperson(en)**	Prof. Dr. Christoph Dalitz
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Statistical Learning
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Grundbegriffe der Statistik werden als bekannt vorausgesetzt, wie z.B. Wahrscheinlichkeitsdichte, Erwartungswert oder Konfidenzintervall. Ferner werden grundlegende Programmierkenntnisse in einer objektorientierten Sprache angenommen.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, Regressions- und Klassifikationsmodelle zu modellieren, zu schätzen und statistisch fundiert zu bewerten sowie hochdimensionale Daten explorativ zu analysieren. Sie können Modellannahmen erkennen, die Vorhersagegüte systematisch beurteilen, Modelle vergleichen und Verfahren zur Merkmalsauswahl, Dimensionsreduktion und Clusteranalyse anwenden.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- die Bayessche Fehlerrate in einem Beispiel analytisch berechnen und die optimale Reject-Threshold herleiten
- eine Likelihood-Funktion berechnen und numerisch maximieren
- die Eigenschaften modellfreier Dichteschätzer untersuchen
- verschiedene Methoden zur Variablen- bzw. Modell-Auswahl quantitativ vergleichen
- Vorhersagemodelle anhand mehrerer Kennzahlen vergleichen, Unterschiede auf statistische Signifikanz testen und Zusammenhänge zwischen Metriken und Normen herleiten
- verschiedene Visualisierungen hochdimensionaler Daten berechnen und qualitativ vergleichen
- Clustering-Algorithmen anwenden, die Anzahl verborgener Cluster schätzen und die Übereinstimmung eines Clusterings mit einem Ground-Truth-Clustering bewerten

WOZU. Die Studierenden nutzen diese Kompetenzen, um die in anderen Modulen entwickelten Machine Learning Systeme einzuordnen, systematisch zu untersuchen und statistisch zu bewerten. Ferner nutzen sie sie, um in der betrieblichen oder wissenschaftlichen Praxis anfallende Daten explorativ zu untersuchen, Vorhersagen zu machen und deren Genauigkeit abzuschätzen.

Inhalte

- The two cultures: Explain versus predict
- Bayessche Entscheidungstheorie
- Klassifikation: Modellbasierte und modellfreie Schätzung von Wahrscheinlichkeiten
- Regression: Modellbasierte und modellfreie Methoden und deren Eigenschaften
- Metrische Räume
- Methoden zur Performance-Schätzung
- Feature Selection und Importance
- Multi-Dimensional-Scaling und Manifold Learning
- Clustering und dessen Evaluation

Prüfungsvorleistung

Keine angegeben. #### Prüfungsleistung Mündliche Prüfung. Sie wird als Gruppenprüfung durchgeführt.

Literatur

- G. Shmuell: “To explain or to predict.” Statistical Science 25(3), pp. 289-310 (2010)
- T. Hastie, R. Tibshirani, J. Friedman: *The Elements of Statistical Learning.* 2nd ed., Springer (2009)
- G. James, D. Witten, T. Hastie, R. Tibshirani: *An Introduction to Statistical Learning with Applications in R.* 2nd ed., Springer (2021)
- S. Theodoridis, K. Koutroumbas: *Pattern Recognition.* 4th ed., Academic Press (2008)
- wissenschaftliche Fachartikel zu den speziellen Themen

Kurzporträt MIN112: Statistical Learning

Worum geht es? Statistical Learning behandelt die statistischen Grundlagen moderner Datenanalyse und Machine-Learning-Verfahren. Im Mittelpunkt stehen Modellierung, Schätzung, Modellvergleich, Performanzbewertung, Dimensionsreduktion, Feature Selection und Clustering.
Wofür braucht man es? Die Kompetenzen werden benötigt, um datengetriebene Modelle nicht nur anzuwenden, sondern ihre Aussagekraft, Unsicherheit und Grenzen kritisch zu beurteilen. Sie sind zentral für Data Science, Machine Learning, Evaluation und datenbasierte Entscheidungsunterstützung.
Wieso ist es interessant? Spannend ist das Modul, weil es hinter die Oberfläche trainierter Modelle blickt. Studierende lernen, warum gute Vorhersagen, erklärende Modelle und belastbare statistische Aussagen nicht automatisch dasselbe sind.

MIN113: Entwurf sicherer Anwendungssysteme

Modulverantwortung	Prof. Dr. Martin Grothe	**Lehrperson(en)**	Prof. Dr. Martin Grothe
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Design of Secure Application Systems
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Empfohlen werden Kenntnisse in Programmierung, Software Engineering, Softwarearchitektur, Softwaretests und Versionsverwaltung. Erwartet werden außerdem Grundlagen zu Web- und API-basierten Anwendungen, Datenbanken, Betriebssystemen, Rechnernetzen sowie zu grundlegenden Schutzzielen, Bedrohungen und Schutzmaßnahmen der IT-Sicherheit.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, neue und bestehende komplexe Anwendungssysteme hinsichtlich ihrer Sicherheitsanforderungen, Bedrohungen, Schwachstellen und Architekturentscheidungen zu analysieren. Sie können geeignete Sicherheitsarchitekturen entwickeln, bestehende Systeme risikobasiert absichern und strukturell verbessern, ausgewählte Maßnahmen implementieren und deren Wirksamkeit überprüfen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- fachliche Prozesse, Systemgrenzen, schützenswerte Werte, Datenflüsse, Schnittstellen und Vertrauensgrenzen analysieren,
- bestehende Architekturen, Quelltexte, Konfigurationen und Abhängigkeiten erschließen und sicherheitsrelevante Schwächen identifizieren,
- Bedrohungsmodelle erstellen, Angriffsflächen, Missbrauchsszenarien und Risiken priorisieren und daraus überprüfbare Sicherheitsanforderungen und Akzeptanzkriterien ableiten,
- Architekturvarianten und Verbesserungsmaßnahmen hinsichtlich Sicherheit, Wartbarkeit, Performance, Kompatibilität und Betriebsaufwand vergleichen,
- Prinzipien und Muster sicherer Softwarearchitekturen sowie Identitäts-, Authentifizierungs-, Autorisierungs-, Daten- und Secret-Management-Konzepte entwerfen oder verbessern,
- Härtungs-, Refactoring- und Migrationsmaßnahmen für bestehende Anwendungssysteme planen und eine Sicherheitsarchitektur als lauffähigen Prototyp oder abgesicherten Systemteil umsetzen,
- automatisierte Sicherheits-, Integrations-, Regressions- und Missbrauchstests entwickeln und die Wirksamkeit der Maßnahmen bewerten,
- Architekturentscheidungen, Änderungen, Prüfergebnisse, technische Grenzen und verbleibende Risiken dokumentieren, präsentieren und verteidigen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um neue Anwendungssysteme sicher zu entwerfen sowie bestehende und gewachsene Systeme systematisch zu untersuchen, abzusichern und nachhaltig weiterzuentwickeln. Sie können Sicherheitsmaßnahmen risikobasiert priorisieren, technisch umsetzen und deren Wirksamkeit nachvollziehbar nachweisen.

Inhalte

- Security by Design, Secure by Default, Privacy by Design und Defense in Depth
- Greenfield- und Brownfield-Entwicklung: sicherer Neuentwurf sowie Sicherheitsanalyse und Verbesserung bestehender Systeme
- Sicherheitsanforderungen, Schutzbedarf, Security User Stories, Misuse Cases und überprüfbare Akzeptanzkriterien
- Systemkontexte, Datenflussmodelle, Angriffsflächen, Vertrauensgrenzen und Bedrohungsmodellierung mit Angreiferprofilen, STRIDE, Attack Trees und Geschäftslogikbedrohungen
- Schwachstellen-, Risiko- und Priorisierungsverfahren für technische und architektonische Sicherheitsprobleme
- Sichere Architekturprinzipien und -muster, Sicherheits-Trade-offs und Architecture Decision Records
- Identitäts- und Zugriffsarchitekturen: Authentifizierung, Autorisierung, Sitzungen, Tokens, Dienstidentitäten und Mandantenisolation
- Sichere Web-, API- und Servicearchitekturen einschließlich Eingabevalidierung, serverseitiger Autorisierung, sicherer Kommunikation und Fehlerbehandlung
- Daten-, Schlüssel-, Zertifikats- und Secret-Management
- Sicherheitsorientiertes Refactoring, Systemhärtung, Resilienz und Missbrauchsschutz (Rate Limiting, Timeouts, Idempotenz, sichere Degradation)
- Security Logging, Monitoring, Auditierbarkeit und Sicherheitsverifikation durch Reviews, negative Tests sowie statische und dynamische Analyse
- Sicherheitsstandards und Rahmenwerke (OWASP ASVS, OWASP SAMM, NIST SSDF, BSI) sowie praktische Analyse und Absicherung eines Anwendungssystems

Prüfungsvorleistung

Erfolgreiche Bearbeitung semesterbegleitender Aufgaben, beispielsweise zur Sicherheitsanalyse bestehender Systeme, Bedrohungsmodellierung, Architekturentwicklung, Implementierung und Sicherheitsverifikation.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Projektarbeit mit Präsentation und Fachgespräch ausgestaltet und umfasst den sicheren Entwurf eines neuen Anwendungssystems oder die Analyse, Absicherung und strukturelle Verbesserung eines bestehenden Systems. Die Studierenden erstellen ein Bedrohungsmodell, leiten überprüfbare Sicherheitsanforderungen ab, entwickeln und bewerten geeignete Architekturmaßnahmen und implementieren Sicherheitsmechanismen in einem lauffähigen Prototyp oder Bestandssystem. Die Wirksamkeit der Maßnahmen wird durch automatisierte Sicherheits-, Integrations-, Regressions- und Missbrauchstests überprüft. Architekturentscheidungen, Änderungen, Prüfergebnisse, Auswirkungen und verbleibende Risiken werden dokumentiert und fachlich verteidigt.

Literatur

- R. Anderson: *Security Engineering: A Guide to Building Dependable Distributed Systems*.
- A. Shostack: *Threat Modeling: Designing for Security*.
- D. Johnsson, D. Deogun, D. Sawano: *Secure by Design*.
- M. Fowler: *Refactoring: Improving the Design of Existing Code*.
- OWASP: *Application Security Verification Standard* und *Software Assurance Maturity Model*, jeweils in aktueller Fassung.
- Aktuelle wissenschaftliche Publikationen, Standards und technische Dokumentationen zu sicherer Softwareentwicklung, Bedrohungsmodellierung und Sicherheitsarchitekturen.

Kurzporträt MIN113: Entwurf sicherer Anwendungssysteme

Worum geht es? Entwurf sicherer Anwendungssysteme behandelt Sicherheit als Architektur- und Entwicklungsaufgabe. Die Studierenden analysieren Bedrohungen, Schutzbedarfe, Vertrauensgrenzen, Identitäts- und Zugriffskonzepte sowie Sicherheitsmechanismen in neuen und bestehenden Anwendungssystemen.
Wofür braucht man es? Die Kompetenzen werden gebraucht, um Software nicht nachträglich zu flicken, sondern Sicherheit systematisch zu entwerfen, umzusetzen und zu überprüfen. Sie sind relevant für Websysteme, APIs, Unternehmensanwendungen, Plattformen und sicherheitskritische Dienste.
Wieso ist es interessant? Interessant ist das Modul, weil Sicherheit selten aus einer einzelnen Maßnahme entsteht. Erst das Zusammenspiel von Architektur, Code, Betrieb, Tests, Datenflüssen und Angreifermodellen entscheidet, ob ein System belastbar ist.

MIN114: Architektur und Engineering moderner KI-Systeme

Modulverantwortung	Prof. Dr. Christoph Quix	**Lehrperson(en)**	Prof. Dr. Christoph Quix Prof. Dr. Duc Duy Pham
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Architecture and Engineering of Modern AI Systems
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Die Studierenden sollten über grundlegende Kompetenzen in Machine Learning und Deep Learning verfügen. Empfohlen werden insbesondere Kenntnisse zu generativen KI-Modellen, Transformer-Architekturen und Verfahren zur Bewertung von KI-Modellen sowie praktische Programmierkenntnisse, vorzugsweise in Python. Grundlagen der Softwaretechnik, verteilter Systeme und des Datenmanagements sind für die Bearbeitung der praktischen Aufgaben hilfreich. * Empfohlen ab dem 2. Fachsemester

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine komplexe, gegebenenfalls multimodale KI-Systemlösung für eine offene Problemstellung eigenständig zu entwickeln und deren Architektur, Verhalten und Qualität anhand technischer und wissenschaftlicher Kriterien begründet zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- funktionale und nichtfunktionale Anforderungen an KI-basierte Softwaresysteme analysieren und daraus geeignete Systemarchitekturen und Qualitätsziele ableiten,
- heterogene KI-Modelle, Datenquellen, Wissenskomponenten und klassische Softwaredienste zu einem Gesamtsystem integrieren,
- mehrstufige KI-Workflows und agentische Systeme mit expliziten Kontrollflüssen, Zuständen und Rollen entwerfen und implementieren,
- unterschiedliche Verfahren zur Wissens- und Kontextintegration, insbesondere Retrieval- und graph-basierte Ansätze, auswählen und vergleichend untersuchen,
- Verfahren zur Planung, Aufgabenzerlegung, Ergebnisprüfung und iterativen Verbesserung in kontrollierte Systemabläufe einbetten,
- Modelle und Systemkomponenten unter Berücksichtigung von Ergebnisqualität, Latenz, Kosten, Ressourcenbedarf, Datenschutz und Betriebsanforderungen auswählen und kombinieren,
- Testfälle, Evaluationsdatensätze, Metriken und Baselines zur Bewertung einzelner Komponenten und vollständiger Systemabläufe entwickeln sowie Fehler, Unsicherheiten und unerwünschte Verhaltensweisen durch Logging, Tracing, Monitoring und systematische Fehleranalysen untersuchen,
- aktuelle wissenschaftliche Arbeiten zu modernen KI-Systemen einordnen und darauf aufbauend in einer Gruppe einen ausgewählten Teilaspekt eines modernen KI-Systems implementieren, evaluieren und die Ergebnisse in Form eines wissenschaftlich strukturierten Kurzbeitrags dokumentieren und präsentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um komplexe KI-basierte Softwaresysteme für anspruchsvolle Anwendungs- und Forschungsaufgaben fachlich fundiert, zuverlässig und nachvollziehbar zu entwickeln und über ihren Lebenszyklus hinweg zu betreiben.

Inhalte

- Architekturprinzipien und Entwurfsmuster komplexer KI-Systeme sowie Integration heterogener KI-Modelle, Datenquellen, Wissenskomponenten, Werkzeuge und Softwaredienste
- Orchestrierung mehrstufiger, zustandsbehafteter und ereignisgesteuerter KI-Workflows sowie agentische und kooperative Systeme mit Rollen, Zielen, Planung und menschlichen Kontrollmechanismen
- Verfahren für Reasoning, Aufgabenzerlegung, Suche, Verifikation und iterative Ergebnisverbesserung
- Wissens- und Kontextintegration durch Retrieval, Graphen und Memory-Mechanismen, multimodale KI-Systeme sowie standardisierte Schnittstellen und Protokolle für Kontexte, Werkzeuge und Agenten
- Modellrouting, Modellkaskaden, Caching, Fallback-Strategien und ressourcenbewusste Systemausführung
- Test und Evaluation probabilistischer KI-Systeme auf Komponenten-, Integrations- und Systemebene einschließlich Robustheits-, Sicherheits- und Regressionstests
- Versionierung, Experimentverwaltung, automatisierte Bereitstellung und kontinuierliche Evaluation
- Deployment, Skalierung, Monitoring, Beobachtbarkeit und Optimierung von Qualität, Latenz, Durchsatz und Ressourcenbedarf
- Sicherheit, Datenschutz, Zugriffskontrolle, Nachvollziehbarkeit, Auditierbarkeit und verantwortliche Systemgestaltung
- Analyse aktueller wissenschaftlicher Arbeiten und Entwicklung eigener weiterführender Forschungsfragen

Die konkreten Modelle, Architekturen, Protokolle und Werkzeuge werden entsprechend dem jeweiligen Stand von Wissenschaft und Technik ausgewählt.

Prüfungsvorleistung

Keine angegeben. #### Prüfungsleistung Studien- oder Projektarbeit. Sie wird als Portfolio mit Projektarbeit, schriftlicher Ausarbeitung, Repository, Ergebnispräsentation und Fachgespräch ausgestaltet. Gruppen von drei bis vier Studierenden bearbeiten einen ausgewählten

Teilaspekt eines modernen KI-Systems. Im Fachgespräch werden die individuellen Beiträge, das Verständnis der eingesetzten Verfahren sowie die Fähigkeit zur Begründung, Bewertung und Übertragung der getroffenen Architekturentscheidungen geprüft. Die individuellen Leistungen werden zusätzlich anhand der im Portfolio dokumentierten Arbeiten bewertet.

Literatur

- Huyen, C.: *AI Engineering: Building Applications with Foundation Models*. O'Reilly Media, 2024.
- Albada, M.: *Building Applications with AI Agents: Designing and Implementing Multiagent Systems*. O'Reilly Media, 2025.
- Lakshmanan, V., Hapke, H.: *Generative AI Design Patterns: Solutions to Common Challenges When Building GenAI Agents and Applications*. O'Reilly Media, 2025.

Kurzporträt MIN114: Architektur und Engineering moderner KI-Systeme

Worum geht es? Architektur und Engineering moderner KI-Systeme behandelt KI-Anwendungen als zusammengesetzte Softwaresysteme. Die Studierenden entwerfen Workflows, integrieren Modelle, Datenquellen, Wissenskomponenten und Dienste und untersuchen Qualität, Kosten, Latenz, Datenschutz und Betrieb.
Wofür braucht man es? Die Kompetenzen werden für AI Engineering, KI-Produktentwicklung und datenintensive Softwaresysteme benötigt. Sie helfen, KI-Modelle nicht isoliert zu betrachten, sondern zuverlässig in überprüfbare, wartbare und verantwortbare Systemarchitekturen einzubetten.
Wieso ist es interessant? Spannend ist das Modul, weil moderne KI-Systeme oft aus vielen beweglichen Teilen bestehen. Die Herausforderung liegt darin, probabilistische Modelle, Softwaretechnik, Evaluation und menschliche Kontrolle zu einem tragfähigen Gesamtsystem zu verbinden.

MIN115: Sprachbasierte Systeme

Modulverantwortung	Prof. Dr. Klaus Weidenhaupt	**Lehrperson(en)**	Prof. Dr. Michael Gref Prof. Dr. Klaus Weidenhaupt
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Language-Based Systems
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Programmierkenntnisse in Python, mathematische Grundlagen (lineare Algebra, Stochastik), Grundlagen des maschinellen Lernens

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, ein vollständiges Sprachdialogsystem auf Basis eines fundierten algorithmischen Verständnisses der zugrunde liegenden Sprach- und Textmodelle softwaretechnisch zu konzipieren, zu implementieren und systematisch zu evaluieren.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie * in einer projektbasierten Arbeitsweise die konzeptionellen und algorithmischen Prinzipien der Sprach- und Informationsverarbeitung analysieren und praktisch anwenden. * akustische Modelle für Speech-to-Text/Text-to-Speech, moderne Deep-Learning-Architekturen (wie Transformer) für das Textverständnis und die Textgenerierung sowie formale Information-Retrieval-Verfahren für die Anbindung externer Wissensquellen trainieren, parametrisieren, adaptieren und kombinieren. * sie die innere Funktionsweise dieser Modelle reflektieren, theoriegeleitete Architekturentscheidungen treffen und die Teilsysteme sowie vortrainierte Modelle anhand etablierter quantitativer Metriken evaluieren. * aktuelle Forschungsansätze auf dem Gebiet der sprachbasierten Systeme analysieren, anwenden und ihre Stärken und Schwächen bewerten. * Ergebnisse zielgruppengerecht dokumentieren und präsentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um robuste Mensch-Maschine-Schnittstellen für komplexe Anwendungsdomänen (wie intelligente Assistenzsysteme oder smarte Unternehmenssuche) nicht nur als “Blackbox” zu nutzen, sondern Sprach-Pipelines zielgerichtet zu adaptieren, zu optimieren und fundiert in umfassende Softwarearchitekturen zu integrieren.

Inhalte

Das Modul verbindet das theoriegeleitete Verständnis von Sprachtechnologien mit der praktischen Realisierung eines umfassenden Sprachdialogassistenten (Project-Based Learning). Die Inhalte teilen sich in auditive Sprachsignalverarbeitung (Speech) und textbasierte Sprachverarbeitung (Language & IR) auf: * Systemarchitektur moderner Sprachassistenten: Pipeline-Architekturen im Vergleich zu End-to-End-Systemen * Herausforderungen der multimodalen Mensch-Maschine-Interaktion * Auditive Signalverarbeitung: digitale Audiosignalverarbeitung und Merkmalsextraktion (z. B. Mel-Spektrogramme, MFCCs) * Speech-to-Text (ASR): akustische und sprachliche Modelle, Voice Activity Detection und Wake-Word Detection * Text-to-Speech (TTS): generative Ansätze der Sprachsynthese * Textrepräsentation: Vektorraummodelle und hochdimensionale Text Embeddings * Deep Learning für NLP: Transformer-Architekturen und Large Language Models für Textverständnis und -generierung * Dialogmodellierung: Intent-Erkennung, Entitätsextraktion und Modellierung von Konversationsflüssen * Information Retrieval: klassische probabilistische und algebraische Modelle (z. B. BM25), semantische Suche und neuronale Reranking-Verfahren * Wissensintegration: Anbindung externer Datenquellen an Sprachmodelle, insbesondere Retrieval-Augmented Generation, Knowledge Graphs und Graph-RAG * Systemintegration: Zusammenführung von Audio- und Text-Pipelines zu einem reaktiven Gesamtsystem * Systematische Evaluation: fachspezifische Metriken und deren statistische Aussagekraft (z. B. WER, Precision/Recall, BLEU/ROUGE)

Prüfungsvorleistung

Erfolgreiche Präsentation von Projektmeilensteinen im Semesterverlauf (z. B. Architekturkonzept und erste Teilsystem-Demonstration).

Prüfungsleistung

Studien- oder Projektarbeit. Sie wird als Projektarbeit ausgestaltet und umfasst die Implementierung eines lauffähigen sprachbasierten Softwaresystems, eine Begleitdokumentation mit methodisch-algorithmischer Begründung der Designentscheidungen sowie eine abschließende Präsentation und ein Fachgespräch.

Literatur

- Jurafsky, D., & Martin, J. H. (2026): Speech and Language Processing. (Aktuellste Auflage / Online Draft). Prentice Hall.
- Alammar, J., & Grootendorst, M. (2024): Hands-On Large Language Models: Language Understanding and Generation. O’Reilly Media.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008): Introduction to Information Retrieval. Cambridge University Press.
- Rabiner, L. R., & Schafer, R. W. (2011): Theory and Applications of Digital Speech Processing. Pearson.
- Yu, D., & Deng, L. (2015): Automatic Speech Recognition: A Deep Learning Approach. Springer.
- Semesteraktuelle wissenschaftliche Publikationen und Fachartikel zur Erarbeitung neuester Ansätze (z.B. aus Interspeech, ACL, SIGIR).

Kurzporträt MIN115: Sprachbasierte Systeme

Worum geht es? Sprachbasierte Systeme behandelt die technische Grundlage moderner Sprach- und Dialogsysteme. Dazu gehören Audioverarbeitung, Speech-to-Text, Text-to-Speech, Sprachmodelle, Information Retrieval, Wissensanbindung und die Integration zu einem lauffähigen Assistenzsystem.
Wofür braucht man es? Die Kompetenzen werden gebraucht, um sprachbasierte Schnittstellen, Unternehmenssuche, Dialogsysteme oder intelligente Assistenten fachlich fundiert zu entwickeln und zu bewerten. Sie verbinden maschinelles Lernen, Softwarearchitektur und Mensch-Maschine-Interaktion.
Wieso ist es interessant? Interessant ist das Modul, weil Sprache für Menschen selbstverständlich wirkt, technisch aber aus vielen schwierigen Teilschritten besteht. Studierende sehen, wie akustische Signale, Textmodelle, Retrieval und Dialoglogik zusammenwirken.

MIN116: Computer Vision

Modulverantwortung	Prof. Dr. Regina Pohle-Fröhlich	**Lehrperson(en)**	Prof. Dr. Regina Pohle-Fröhlich
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Computer Vision
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Programmierkenntnisse in Python Grundlagen des Maschinellen Lernens

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, komplexe Aufgaben im Bereich der Bildanalyse eigenständig zu bearbeiten, geeignete Algorithmen auf Basis eines fundierten algorithmischen Verständnisses auszuwählen, diese zu implementieren und die Ergebnisse fachlich fundiert zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- grundlegende klassische Bildverarbeitungs-Techniken analysieren, implementieren und praktisch anwenden,
- moderne Deep-Learning Ansätze kennen lernen, die innere Funktionsweise reflektieren und vortrainierte Modelle anhand etablierter quantitativer Metriken evaluieren,
- aktuelle Forschungsansätze auf dem Gebiet der Bildanalyse analysieren, anwenden und ihre Stärken und Schwächen bewerten,
- Ergebnisse dokumentieren und in einem strukturierten Bericht mit Reflexion über Grenzen und mögliche Verbesserungen präsentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um Bildanalyse Lösungen in interdisziplinären Projekten (z. B. autonome Fahrzeuge, medizinische Bilddiagnostik, Qualitätskontrolle in der Industrie oder Augmented Reality Anwendungen) konzeptionell zu planen, technisch umzusetzen und die Resultate wissenschaftlich zu bewerten.

Inhalte

- grundlegende Bildverarbeitungs-Techniken (Grauwert-Transformation, Filterverfahren, Farbraum-Konversion)
- klassische Segmentierungsalgorithmen (histogrammbasiert, regionen-basiert, kanten-basiert, graphen-basiert)
- Merkmal-Extraktionsverfahren (z.B. Farb- und Formmerkmale, Deskriptoren, Histogram of Oriented Gradients (HOG))
- etablierte Verfahren zur Objekt- und Strukturdetektion (z.B. Canny Operator, Hough-Transformation, Template Matching)
- Netzwerkarchitekturen für Bildklassifikationsaufgaben
- Deep-Learning-Ansätze zur Objektdetektion und Segmentierung
- Algorithmen zur geometrischen Computer Vision (Stereo-Geometrie, Structure from Motion)
- Algorithmen zur Verarbeitung von Point Clouds

Prüfungsvorleistung

Erfolgreiche Präsentation der Projektergebnisse

Prüfungsleistung

Mündliche Prüfung. Sie wird als Gruppenprüfung durchgeführt.

Literatur

- W. Burger, M.J. Burge: Digital Image Processing, Springer, 2022
- W. Shen, C. Si, C. Yang, Y. Yu: Hands-On Computer Vision, Springer, 2026
- S. Liu, M. Zhang, P. Kadam, C.C. J. Kuo: 3D Point Cloud Analysis, Springer, 2021
- Y.-J. Zhang: 3D Computer Vision, Springer, 2024
- S. Esakkirajan, T. Veerakumar, B. N. Subudhi: Digital Image Processing, Springer, 2025

Kurzporträt MIN116: Computer Vision

Worum geht es? Computer Vision behandelt Verfahren zur Analyse, Interpretation und Bewertung visueller Daten. Die Studierenden arbeiten mit klassischer Bildverarbeitung, Segmentierung, Merkmalen, Objekterkennung, Deep Learning, geometrischer Computer Vision und Point Clouds.

Wofür braucht man es? Die Kompetenzen werden in Anwendungen wie medizinischer Bildanalyse, Qualitätskontrolle, autonomen Systemen, Robotik, Assistenzsystemen oder Augmented Reality benötigt. Sie helfen, visuelle Verfahren passend auszuwählen, umzusetzen und ihre Ergebnisse kritisch zu bewerten.

Wieso ist es interessant? Spannend ist das Modul, weil Bilder für Menschen unmittelbar verständlich wirken, für technische Systeme aber komplexe Messdaten sind. Schon Beleuchtung, Perspektive, Rauschen oder Trainingsdaten können Ergebnisse stark verändern.

MIN117: High Performance Computing

Modulverantwortung	Prof. Dr. Nils Kopal	**Lehrperson(en)**	Prof. Dr. Nils Kopal Prof. Dr. Jens Brandt Prof. Dr. Steffen Goebbels
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	High Performance Computing
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Programmierkenntnisse in C, mathematische Grundlagen (Lineare Algebra), Grundlegendes Verständnis von Parallelität

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, Algorithmen der Numerik zu parallelisieren und auf Prozessen, Threads und GPUs zu implementieren. Sie können den Einfluss der Parallelisierung auf Laufzeit und Speicherbedarf analysieren und die Konzepte parallelen Rechnens auf neue Problemstellungen übertragen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- in einer anwendungsorientierten Arbeitsweise die Grundlagen des verteilten Rechnens kennenlernen und bewerten.
- Algorithmen der Numerik parallelisieren und unter Verwendung von GPUs und/oder Open MPI implementieren.
- den Einfluss der Parallelisierung auf Laufzeit und Speicherbedarf analysieren.
- Ergebnisse zielgruppengerecht dokumentieren und präsentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um rechenintensive Algorithmen auf paralleler Hardware zu implementieren. Sie erlangen grundlegende Kenntnisse über Verfahren des Scientific Computing, die in entsprechenden Projekten genutzt werden können. Darüber hinaus sind die Studierenden in der Lage, die Konzepte der parallelen Programmierung auch auf Aufgabenstellungen außerhalb der Numerik anzuwenden.

Inhalte

- Architektur paralleler Systeme
- Design paralleler Algorithmen
- Parallelisierung mittels Prozessen und Threads, Kommuniketion über Shared Memory (Open MP) und Messaging (Open MPI)
- Verwendung von GPUs zum parallelen Rechnen (CUDA/OpenCL)
- Anwendung auf Verfahren der linearen Algebra (Matrixmultiplikation, Lösen von Gleichungssystemen, schnelle Fourier-Transformation)

Prüfungsvorleistung

Erfolgreiche Implementierung und Erklärung von parallelen Beispiel-Algorithmen.

Prüfungsleistung

Mündliche Prüfung (45 Minuten).

Literatur

- George Karniadakis, Robert M. Kirby: Parallel Scientific Computing in C++ and MPI. Cambridge University Press, 2003
- Gene H. Golub, James M. Ortega: Scientific Computing - An Introduction with Parallel Computing. Academic Press, 2014
- Efstratios Gallopoulos, Bernard Philippe, Ahmed H. Sameh: Parallelism in Matrix Computations, Springer, 2015
- Michael Quinn: Parallel Programming in C with MPI and OpenMP. McGraw-Hill, 2003
- Peter Pacheco: Parallel Programming with MPI. Morgan Kaufmann, 1996

Kurzporträt MIN117: High Performance Computing

Worum geht es? High Performance Computing behandelt paralleles und verteiltes Rechnen für rechenintensive Aufgaben. Die Studierenden beschäftigen sich mit parallelen Architekturen, Prozessen, Threads, Message Passing, GPUs und Anwendungen aus dem Scientific Computing.
Wofür braucht man es? Die Kompetenzen werden gebraucht, wenn einzelne Rechner, sequentielle Programme oder naive Algorithmen nicht mehr ausreichen. Sie sind relevant für Simulation, Datenanalyse, KI, numerische Verfahren und alle Anwendungen mit hohem Rechenbedarf.
Wieso ist es interessant? Interessant ist das Modul, weil mehr Hardware nicht automatisch mehr Geschwindigkeit bedeutet. Erst geeignete Zerlegung, Kommunikation, Speicherzugriffe und Messung zeigen, ob Parallelisierung tatsächlich hilft.

MIN118: Architektur eingebetteter Systeme

Modulverantwortung	Prof. Dr. Benedikt Janßen	**Lehrperson(en)**	Prof. Dr. Matteo Zella Prof. Dr. Benedikt Janßen
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Architecture of Embedded Systems
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Empfohlen werden auf Bachelor-Niveau erworbene Kompetenzen in C-Programmierung, Rechner- und Systemarchitekturen, Betriebssystemen, Software Engineering sowie in der Modellierung softwareintensiver Systeme. Erwartet werden außerdem Grundlagen zu Echtzeitverhalten, Kommunikation sowie Safety und Security. Unterschiedliche Vorkenntnisse werden durch geeignete Materialien zum angeleiteten Selbststudium ausgeglichen; im Modul werden diese Kompetenzen im Kontext heterogener Embedded-Systemarchitekturen vertieft.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, für heterogene eingebettete Subsysteme innerhalb komplexer technischer Gesamtsysteme selbstständig ein wissenschaftlich und technisch begründetes Systemkonzept zu entwickeln, prototypisch umzusetzen und zu validieren. Sie können unter unvollständigen oder konkurrierenden Anforderungen begründete Architekturentscheidungen zu Hardware-Software-Aufteilung, Laufzeitumgebung, Kommunikation, Echtzeitverhalten, Zuverlässigkeit, Safety, Security sowie Ressourcen- und Energiebedarf treffen,

WOMIT. indem sie

- neue oder unvollständig spezifizierte Anwendungsszenarien analysieren, widersprüchliche Anforderungen und Wissenslücken identifizieren und daraus überprüfbare funktionale und nichtfunktionale Anforderungen ableiten,
- aktuelle wissenschaftliche und technische Ansätze für Embedded-Systemarchitekturen selbstständig erschließen, kritisch vergleichen und für den jeweiligen Systemkontext auswählen,
- Architekturen heterogener Embedded-Subsysteme modellieren und Hardware-Software-Grenzen, Laufzeitumgebungen sowie Kommunikationsschnittstellen begründet festlegen,
- Echtzeit-, Kommunikations-, Isolations- und Fehlerbehandlungsmechanismen zu einer konsistenten Systemarchitektur integrieren,
- ein ausgewähltes Teilsystem prototypisch realisieren und sein Verhalten anhand von Modellen, Messdaten, Fehlerfällen und quantifizierten Qualitätskriterien validieren,
- Architekturvarianten hinsichtlich Echtzeitverhalten, Zuverlässigkeit, Safety, Security, Wartbarkeit, Ressourcen- und Energiebedarf vergleichen, ihre Entscheidungen dokumentieren und fachlich verteidigen,

WOZU. um in interdisziplinären Entwicklungsteams wissenschaftlich und empirisch fundierte Architekturentscheidungen für heterogene eingebettete Subsysteme in neuen oder unvollständig bestimmten Anwendungskontexten zu treffen, prototypisch umzusetzen, kritisch zu validieren und gegenüber Fachkolleg:innen zu vertreten.

Inhalte

- Eingebettete Subsysteme im Kontext komplexer technischer Gesamtsysteme
- Analyse offener Anforderungen und technischer Randbedingungen
- Aktuelle Architekturansätze für heterogene Embedded Systems
- Modellierung und Bewertung von Hardware-Software-Architekturen
- Hardware-Software-Partitionierung und Auswahl von Laufzeitumgebungen
- Integration von Echtzeit-, Kommunikations-, Isolations- und Fehlerbehandlungsmechanismen
- Prototypische Realisierung eines ausgewählten Teilsystems
- Mess- und modellbasierte Validierung unter Normal-, Grenz- und Fehlerbedingungen
- Quantitativer Vergleich von Architekturvarianten
- Dokumentation und fachliche Verteidigung von Architekturentscheidungen

Prüfungsvorleistung

Keine angegeben. #### Prüfungsleistung Studien- oder Projektarbeit. Sie wird als semesterbegleitende Projektarbeit mit Fachgespräch ausgestaltet. Die Studierenden bearbeiten eine offene Entwurfs- und Prototyping-Aufgabe zu einem heterogenen eingebetteten Subsystem innerhalb eines größeren technischen Gesamtsystems. Als Grundlage können geeignete aktuelle Projekte dienen, wenn eine abgegrenzte Aufgabenstellung abgeleitet wird. Die Projektarbeit umfasst die Analyse unvollständiger oder konkurrierender Anforderungen, die selbstständige Erschließung geeigneter wissenschaftlicher und technischer Ansätze, die Modellierung und den begründeten Vergleich von Architekturvarianten sowie die prototypische Realisierung eines ausgewählten Teilsystems. Das Systemverhalten wird anhand von Modellen, Messdaten und quantifizierten Qualitätskriterien unter Normal-, Grenz- und Fehlerbedingungen validiert. Dabei werden insbesondere Echtzeitverhalten, Zuverlässigkeit, Safety, Security, Wartbarkeit sowie Ressourcen- und Energiebedarf berücksichtigt. Das Fachgespräch dient dem individuellen Nachweis des Verständnisses der getroffenen Annahmen, Architekturentscheidungen, technischen Umsetzung, Validierungsmethoden und Grenzen der Lösung.

Literatur

- Peter Marwedel: Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things. 4th Edition. Springer, 2021.
- Edward A. Lee, Sanjit A. Seshia: Introduction to Embedded Systems: A Cyber-Physical Systems Approach. 2nd Edition. MIT Press, 2017.
- Wayne Wolf: Computers as Components: Principles of Embedded Computing System Design. 4th Edition. Morgan Kaufmann, 2017.
- Hermann Kopetz: Real-Time Systems: Design Principles for Distributed Embedded Applications. 2nd Edition. Springer, 2011.
- Tammy Noergaard: Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers. 2nd Edition. Newnes, 2012.

Kurzporträt MIN118: Architektur eingebetteter Systeme

Worum geht es? Architektur eingebetteter Systeme behandelt heterogene eingebettete Subsysteme als Teil größerer technischer Gesamtsysteme. Die Studierenden entwerfen Systemkonzepte, wägen Hardware-Software-Aufteilung, Laufzeitumgebung und Kommunikationsschnittstellen ab und betrachten Echtzeitverhalten, Zuverlässigkeit, Safety, Security sowie Ressourcenbedarf als gemeinsame Architekturfragen.
Wofür braucht man es? Die Kompetenzen werden benötigt, wenn Software eng mit Sensorik, Aktorik, Spezialhardware oder physischen Prozessen gekoppelt ist. Sie helfen, Architekturentscheidungen für industrielle, mobile, medizinische oder cyber-physische Systeme nicht nur technisch umzusetzen, sondern unter konkurrierenden Randbedingungen nachvollziehbar zu begründen.
Wieso ist es interessant? Spannend ist das Modul, weil eingebettete Systeme unter echten Randbedingungen arbeiten: Zeit, Energie, Kosten, Sicherheit und Zuverlässigkeit zählen gleichzeitig. Dadurch werden Architekturentscheidungen sichtbar folgenreich, weil sie festlegen, was ein technisches System später leisten, aushalten und verantwortbar absichern kann.

MIN119: Fortgeschrittene Assistenzsysteme

Modulverantwortung	Dr. Andreas Kitzig	**Lehrperson(en)**	Dr. Andreas Kitzig Prof. Dr. Duc Duy Pham Prof. Dr. Thomas Nitsche Prof. Dr. Edwin Naroska
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Advanced Assistance Systems
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Empfohlen werden grundlegende Kenntnisse in Programmierung, Software Engineering sowie Machine Learning. Je nach Fokussierung können Kenntnisse in Sensorik, Signalverarbeitung, Robotik oder eingebetteten Systemen hilfreich sein.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, Assistenzsysteme für anwendungsnahe Problemstellungen zu konzipieren, prototypisch umzusetzen und hinsichtlich technischer Machbarkeit, Nutzbarkeit, Sicherheit und gesellschaftlicher Verantwortung begründet zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine anwendungsnahe Problemstellung für ein Assistenzsystem analysieren und eingrenzen,
- Anforderungen aus Nutzungskontext, technischer Machbarkeit und gesellschaftlichen Rahmenbedingungen ableiten,
- einen geeigneten Lösungsansatz entwerfen und fachlich begründen,
- ausgewählte Methoden und Komponenten anwendungsbezogen auswählen, kombinieren und anwenden,
- einen Prototyp oder eine technische Untersuchung iterativ entwickeln,
- Zwischenergebnisse regelmäßig präsentieren, diskutieren und auf Basis von Feedback weiterentwickeln,
- die entstandene Lösung hinsichtlich Nutzen, Grenzen, Risiken und Akzeptanz bewerten,
- die Ergebnisse nachvollziehbar dokumentieren und fachlich präsentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um in Forschung, Entwicklung und Anwendung verantwortbare Assistenzsysteme zu gestalten, die Menschen in konkreten Nutzungskontexten sinnvoll unterstützen.

Inhalte

- Begriff, Einordnung und Typen von Assistenzsystemen
- Anwendungsszenarien aus Industrie, Forschung und gesellschaftlichen Kontexten
- Mensch-System-Interaktion, UI/UX und multimodale Interaktion
- Sensorik und Aktorik
- Signalverarbeitung und Systemintegration
- KI, Mustererkennung und Sensorfusion als mögliche Bausteine
- Robotik und eingebettete Systeme als projektabhängige Vertiefungen
- Safety und IT-Sicherheit
- Projektbezogene Konzeption, prototypische Umsetzung und Evaluation eines Assistenzsystems

Prüfungsvorleistung

Keine angegeben. #### Prüfungsleistung Studien- oder Projektarbeit. Sie wird als semesterbegleitende Projektarbeit mit Zwischenpräsentationen und Fachgespräch ausgestaltet.

Literatur

- Shneiderman, B.: Human-Centered AI. Oxford University Press, 2022.
- Szeliski, R.: Computer Vision: Algorithms and Applications. Springer, 2022.
- Thrun, S.; Burgard, W.; Fox, D.: Probabilistic Robotics. MIT Press, 2005.
- Ergänzende aktuelle Fachpublikationen, technische Dokumentationen und projektspezifische Literatur.

Kurzporträt MIN119: Fortgeschrittene Assistenzsysteme

Worum geht es? Fortgeschrittene Assistenzsysteme behandelt Systeme, die Menschen in konkreten Arbeits-, Forschungs- oder Alltagssituationen technisch unterstützen. Die Studierenden analysieren Nutzungskontexte, wählen geeignete Sensorik, KI-, Interaktions- oder Systemkomponenten aus und entwickeln prototypische Lösungen.
Wofür braucht man es? Die Kompetenzen werden für Anwendungen benötigt, in denen technische Machbarkeit, Nutzbarkeit, Sicherheit und gesellschaftliche Verantwortung zusammen betrachtet werden müssen. Sie sind relevant für Industrie, Gesundheit, Mobilität, Forschung, Mensch-Technik-Interaktion und intelligente Umgebungen.
Wieso ist es interessant? Interessant ist das Modul, weil Assistenzsysteme nur dann überzeugen, wenn sie wirklich zum Menschen und zum Kontext passen. Technik, Akzeptanz, Nutzen und Risiken müssen gemeinsam gedacht werden.

MIN120: Fortgeschrittenes Software Engineering

Modulverantwortung	Prof. Dr. Thomas Nitsche	**Lehrperson(en)**	Prof. Dr. Thomas Nitsche Prof. Dr. Nils Kopal
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Advanced Software Engineering
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Empfohlen werden auf Bachelor-Niveau erworbene Programmierkompetenzen in einer Objekt-orientierten Sprache, sowie grundlegende Kenntnisse im Bereich Software Engineering, die im Software-Pfad des Bachelor-Studiums vermittelt wurden.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, fortgeschrittene Methoden, Vorgehensmodelle, Standards und Arbeitsformen des modernen Software Engineerings anzuwenden und deren Chancen und Risiken zu beurteilen. Sie können die Planung, den Entwurf und die Durchführung sowie den Test und die Qualitätssicherung bei der Entwicklung komplexer Softwaresysteme übernehmen. Darüber hinaus können sie Softwarearchitekturstile und -muster zielgerichtet einsetzen und Systeme zu modellieren.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- fortgeschrittene Methoden, Vorgehensmodelle und Standards des Software Engineerings auf konkrete Aufgabenstellungen anwenden und hinsichtlich ihrer Eignung bewerten,
- aktuelle Verfahren und Methoden auswählen und nutzen um wartungsfreundlichen Code zu entwickeln,
- moderne KI- und Agenten-basierte Entwicklungsumgebungen und Prozesse einsetzen,
- Prinzipien der Softwarearchitektur sowie Architekturstile, Architekturdokumentationen, Muster, Frameworks und Referenzarchitekturen mit standardisierten Verfahren untersuchen, bewerten und zielgerichtet einsetzen,
- Verfahren und Systeme zum Testen von Software anwenden und deren Beitrag zur Qualitätssicherung beurteilen,
- modellbasierte Entwicklungsansätze sowie formale Modelle und Verifikationsverfahren erproben,
- Übungsaufgaben im Rahmen einer seminaristischen Lehrveranstaltung selbstständig und unter Einbezug ergänzender Fachliteratur bearbeiten.

WOZU. Die Studierenden nutzen diese Kompetenzen, um komplexe Softwaresysteme methodisch fundiert zu planen, zu entwerfen, zu entwickeln, zu testen und qualitätszusichern. Sie sind dadurch in der Lage, geeignete Architektur-, Modellierungs- und Entwicklungsentscheidungen zu treffen, deren Chancen und Risiken abzuschätzen und die Qualität, Nachvollziehbarkeit und Verlässlichkeit von Softwaresystemen sicherzustellen.

Inhalte

- Forteschrittene Software-Architektur (Prinzipen, Architektur-Stile, Architekturdokumentation, Muster, Frameworks, Referenzarchitektur)
- KI- und Agenten-basierte bzw. unterstützte Entwicklungsumgebungen wie z.B. Open Code
- Clean Code und SOLID-Design-Framework
- Architektur-Stile, z.B. Microservices
- Anti Patterns, Dark Patterns
- Technische Schulden
- Bewertungsverfahren wie z.B. ATAM (Architecture Tradeoff Analysis Method) oder SAAM (Software Architecture Analysis Method)
- Fortgeschrittenes Testen und Testsysteme, z.B. Test Driven Development und Behaviour Driven Development

Prüfungsvorleistung

Präsentation eines wissenschaftlichen Papers im Bereich Software Engineering (ggfs. in einer Gruppe)

Prüfungsleistung

Mündliche Prüfung.

Literatur

- Gernot Starke: Effektive Softwarearchitekturen: Ein praktischer Leitfaden. 10. Auflage. Carl Hanser Verlag, 2024. ISBN 978-3-446-47672-1.
- Robert C. Martin: Clean Code: A Handbook of Agile Software Craftsmanship. 2. Auflage. Addison-Wesley Professional (Pearson), 2025. ISBN 978-0-13-539857-9.
- Mark Richards, Neal Ford: Handbuch moderner Softwarearchitektur: Architekturstile, Patterns und Best Practices. 2. Auflage. O'Reilly, 2026. ISBN 978-3-96009-277-3.
- Maurício Aniche: Effective Software Testing: A Developer's Guide. Manning Publications, Shelter Island, 2022. 328 Seiten. ISBN 978-1-63343-993-1.
- Eric Evans: Domain-Driven Design: Tackling Complexity in the Heart of Software. 1. Auflage. Addison-Wesley Professional, 2003. ISBN 978-0-321-12521-7

Kurzporträt MIN120: Fortgeschrittenes Software Engineering

Worum geht es? Fortgeschrittenes Software Engineering behandelt Methoden und Entscheidungen für die Entwicklung komplexer Softwaresysteme. Themen sind Softwarearchitektur, Architekturentscheidungen, Clean Code, technische Schulden, Testverfahren, modellbasierte Ansätze und KI-gestützte Entwicklungswerkzeuge.
Wofür braucht man es? Die Kompetenzen werden gebraucht, um größere Softwaresysteme methodisch zu planen, zu entwerfen, zu testen und langfristig weiterzuentwickeln. Sie sind wichtig für Architekturrollen, technische Leitung, Qualitätssicherung und anspruchsvolle Entwicklungsaufgaben.
Wieso ist es interessant? Spannend ist das Modul, weil gute Software nicht nur aus funktionierendem Code besteht. Architektur, Verständlichkeit, Testbarkeit, Wartbarkeit und Entwicklungsprozesse entscheiden darüber, ob Systeme langfristig tragfähig bleiben.

MIN121: Fortgeschrittenes Deep Learning

Modulverantwortung	Prof. Dr. Michael Gref	**Lehrperson(en)**	Prof. Dr. Michael Gref
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch oder Englisch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Advanced Deep Learning
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Empfohlen werden Kenntnisse des maschinellen Lernens und Deep Learning, insbesondere überwachte Lernverfahren, Grundlagen neuronaler Netze, gradientenbasierte Optimierung, Modelltraining, Evaluation und praktische Erfahrung mit datengetriebenen Modellen.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, fortgeschrittene Deep-Learning-Architekturen und Lernparadigmen für komplexe datenbasierte Problemstellungen auszuwählen, prototypisch umzusetzen oder zu adaptieren, experimentell zu evaluieren und hinsichtlich Leistungsfähigkeit, Grenzen, Aufwand und Anwendungskontext kritisch zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- fortgeschrittene Deep-Learning-Architekturen und Lernparadigmen, insbesondere für Sequenzmodellierung, Repräsentationslernen, generative Modellierung und multimodale Aufgabenstellungen, analysieren und voneinander abgrenzen,
- zentrale Modellierungsprinzipien fortgeschrittener Deep-Learning-Verfahren nachvollziehen und für geeignete Aufgabenstellungen begründet auswählen,
- eigene Modelle trainieren sowie vortrainierte Modelle auswählen, anwenden, anpassen und experimentell untersuchen,
- unterschiedliche Modellfamilien sowie Verfahren zur effizienten Modellanpassung hinsichtlich ihrer Prinzipien, Einsatzmöglichkeiten und Grenzen vergleichen,
- Modelle und Experimente qualitativ und quantitativ evaluieren und Ergebnisse im Hinblick auf Generalisierung, Robustheit, Verzerrungen, Interpretierbarkeit und praktische Einsetzbarkeit bewerten,
- aktuelle wissenschaftliche Publikationen und technische Entwicklungen erschließen, reproduzierbare Experimente ableiten und Ergebnisse strukturiert dokumentieren, präsentieren und verteidigen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um moderne Deep-Learning-Verfahren in forschungs- und entwicklungsnahen Informatik-Kontexten fachlich fundiert auszuwählen, anzupassen, kritisch zu bewerten und gegenüber fachkundigen Zielgruppen nachvollziehbar zu begründen.

Inhalte

- Vertiefung neuronaler Netze und Deep-Learning-Training: Modellarchitektur, Optimierung, Regularisierung, Generalisierung und experimentelle Analyse
- Sequenzmodellierung und Transformer: rekurrente Architekturen, Attention, Encoder-Decoder-Strukturen und Skalierung
- Repräsentationslernen und Foundation Models: selbstüberwachtes Lernen, Pretraining, Transfer Learning, Fine-Tuning sowie One-/Few-Shot- und metrisches Lernen
- Generative und multimodale Modelle: Autoencoder, Diffusion Models, gemeinsame Repräsentationen und Text-Bild-Modelle
- Effiziente Modellanpassung und Modellkompression: Adapterverfahren, Low-Rank-Ansätze, Quantisierung und Distillation
- Robustheit, Sicherheit und Interpretierbarkeit moderner Deep-Learning-Modelle
- Evaluation: Metriken, Benchmarking, Fehleranalyse, Bias, Fairness und Grenzen automatisierter Bewertung
- Wissenschaftliches Arbeiten: aktuelle Publikationen erschließen, Experimente reproduzieren und Ergebnisse kritisch einordnen

Prüfungsvorleistung

Erfolgreiche Bearbeitung semesterbegleitender Aufgaben, z. B. zur Analyse, Implementierung, Adaption oder Evaluation fortgeschrittener Deep-Learning-Verfahren.

Prüfungsleistung

Studien- oder Projektarbeit. Sie wird als Projektarbeit oder Hausarbeit mit Präsentation und Fachgespräch ausgestaltet. Die Prüfung umfasst die eigenständige Bearbeitung einer fortgeschrittenen Deep-Learning-Fragestellung einschließlich theoretischer Einordnung, praktischer Umsetzung oder experimenteller Analyse, Dokumentation der Ergebnisse sowie fachlicher Verteidigung.

Literatur

- I. Goodfellow, Y. Bengio, A. Courville: *Deep Learning*.
- C. M. Bishop: *Pattern Recognition and Machine Learning*.
- Aktuelle wissenschaftliche Publikationen und technische Dokumentationen zu fortgeschrittenen Deep-Learning-Architekturen, Foundation Models, generativen Modellen und effizienter Modellanpassung.

Kurzporträt MIN121: Fortgeschrittenes Deep Learning

Worum geht es? Fortgeschrittenes Deep Learning behandelt moderne neuronale Architekturen und Lernparadigmen für komplexe datenbasierte Aufgaben. Dazu gehören Transformer, Repräsentationslernen, Foundation Models, generative Modelle, multimodales Lernen, effiziente Anpassung und Evaluation.
Wofür braucht man es? Die Kompetenzen werden benötigt, um Deep-Learning-Verfahren für anspruchsvolle Anwendungen auszuwählen, anzupassen und kritisch zu bewerten. Sie sind relevant für KI-Entwicklung, Forschung, datengetriebene Produkte und die Bewertung moderner Modellfamilien.
Wieso ist es interessant? Interessant ist das Modul, weil viele aktuelle KI-Durchbrüche auf wenigen grundlegenden Ideen beruhen, die sehr unterschiedlich kombiniert werden. Studierende lernen, Hype, technische Leistungsfähigkeit und reale Grenzen auseinanderzuhalten.

MIN122: Fortgeschrittenes Data Engineering

Modulverantwortung	Prof. Dr. Christoph Quix	**Lehrperson(en)**	Prof. Dr. Christoph Quix
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	Advanced Data Engineering
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Empfohlen werden Kompetenzen aus den Bereichen Data Engineering, Datenbanksysteme und Datenmodellierung. Die Studierenden sollten komplexe SQL-Anfragen sowie relationale und multidimensionale Datenmodelle entwerfen können. Kenntnisse nichtrelationaler Datenbanksysteme, von Datenaufbereitungs-Pipelines sowie grundlegende Programmierkenntnisse in Python oder einer vergleichbaren Sprache sind wünschenswert.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, skalierbare Datenplattformen für datenintensive und KI-basierte Anwendungsszenarien methodisch auszuwählen, anzuwenden und hinsichtlich ihrer funktionalen und nichtfunktionalen Eignung systematisch zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Anforderungen an skalierbare Datenplattformen analysieren und daraus begründete Architekturentscheidungen für Speicherung, Verarbeitung und Bereitstellung von Daten ableiten,
- physische Speicher- und Indexstrukturen algorithmisch untersuchen und deren Auswirkungen auf Anfrageverarbeitung und Skalierbarkeit beurteilen,
- verteilte Anfragepläne, Kostenmodelle und Ausführungsstrategien analysieren und Verfahren zur Partitionierung, Parallelisierung und Optimierung anwenden,
- geeignete relationale und nicht-relationale Datenmodelle und Speichersysteme vergleichen,
- Datenqualitätsprüfungen, Metadatenmanagement, Data Lineage, Monitoring und weitere DataOps-Verfahren in Datenpipelines integrieren,
- Architekturen moderner Datenplattformen unter Berücksichtigung technischer, betrieblicher und organisatorischer Anforderungen entwerfen und dokumentieren,
- die Skalierbarkeit, Latenz, den Durchsatz und den Ressourcenbedarf datenintensiver Systeme anhand geeigneter Experimente messen und kritisch bewerten,
- aktuelle wissenschaftliche Arbeiten des Data Engineering einordnen und darauf aufbauend in einer Gruppe einen ausgewählten Teilaspekt einer skalierbaren Datenplattform implementieren, evaluieren und die Ergebnisse in Form eines wissenschaftlich strukturierten Kurzbeitrags dokumentieren und präsentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um technisch und organisatorisch tragfähige Datenplattformen für komplexe datenintensive Anwendungen fundiert auszuwählen, weiterzuentwickeln und zu beurteilen.

Inhalte

- Grundlagen skalierbarer und verteilter Datenverarbeitung: horizontale Skalierung, Partitionierung, Replikation, Datenlokalität, Konsistenz und Fehlertoleranz
- Physische Datenorganisation: zeilen- und spaltenorientierte Speicherung, Dateiformate, Kompression, Partitionierung, Sortierung und Datenverteilung
- Index- und Zugriffsstrukturen wie z. B. B+-Bäume, Hash-Indizes, Bitmap-Indizes sowie Indexverfahren für vektorielle Ähnlichkeitssuche
- Verteilte Anfrageverarbeitung und -optimierung: Anfragepläne, Parallelisierung, Shuffle-Operationen, verteilte Join-Verfahren, Kostenmodelle und adaptive Ausführung
- Spezialisierte Datenbanksysteme und Datenmodelle wie z. B. Key-Value-, Wide-Column-, Dokumenten-, Graph-, Zeitreihen-, Such- und Vektordatenbanken
- Architekturen moderner Datenplattformen (Data Warehouse, Data Lake, Lakehouse, Data Fabric, Data Mesh) sowie offene Speicher-, Tabellen- und Metadatenkonzepte
- Data Stream Processing und Zeitreihendaten: Ereignis- und Zeitmodelle, Fenster, Watermarks, verspätete Ereignisse und Verarbeitungssemantiken
- Datenplattformen für KI-Systeme: Feature-, Embedding- und Retrieval-Pipelines, vektorielle Daten sowie Schnittstellen zum Model-Lifecycle-Management
- Datenqualität, automatisierte Qualitätsprüfungen, Data Lineage, Monitoring, Observability, Metadatenmanagement und DataOps
- Organisatorische Einordnung anhand aktueller Referenzmodelle des Datenmanagements und der Data Governance
- Experimentelle Evaluation datenintensiver Systeme anhand von Skalierbarkeit, Latenz, Durchsatz und Ressourcenbedarf
- Analyse und Diskussion aktueller Methoden und wissenschaftlicher Veröffentlichungen aus dem Data Engineering

Prüfungsvorleistung

Keine gesonderte Prüfungsvorleistung. Die Bestandteile des Portfolios werden semesterbegleitend erarbeitet.

Prüfungsleistung

Studien- oder Projektarbeit. Sie wird als Portfolio mit Projektarbeit, schriftlicher Ausarbeitung, Repository, Ergebnispräsentation und Fachgespräch ausgestaltet. Gruppen von drei bis vier Studierenden bearbeiten einen ausgewählten Teilaspekt einer skalierbaren Datenplattform. Im Fachgespräch werden die individuellen Beiträge, das Verständnis der eingesetzten Verfahren sowie die Fähigkeit zur Begründung, Bewertung und Übertragung der getroffenen Architekturentscheidungen geprüft. Die individuellen Leistungen werden zusätzlich anhand der im Portfolio dokumentierten Arbeiten bewertet.

Literatur

- Kleppmann, Martin: *Designing Data-Intensive Applications*. O'Reilly, 2017.
- Reis, Joe; Housley, Matt: *Fundamentals of Data Engineering*. O'Reilly, 2019.

Kurzporträt MIN122: Fortgeschrittenes Data Engineering

Worum geht es? Fortgeschrittenes Data Engineering behandelt den Aufbau skalierbarer Datenplattformen und Datenpipelines. Die Studierenden beschäftigen sich mit verteilten Datenarchitekturen, Datenintegration, Verarbeitung großer Datenmengen, Qualität, Governance, Betrieb und technischer Bewertung.
Wofür braucht man es? Die Kompetenzen werden gebraucht, um Daten zuverlässig für Analyse, KI, Reporting und operative Systeme bereitzustellen. Sie sind zentral für Data Platforms, Data Warehousing, Lakehouse-Architekturen, Echtzeitdatenverarbeitung und datengetriebene Anwendungen.
Wieso ist es interessant? Spannend ist das Modul, weil Datenarbeit oft unsichtbare Infrastruktur ist, aber über den Erfolg datengetriebener Systeme entscheidet. Schlechte Datenflüsse, unklare Semantik oder fehlende Qualitätssicherung wirken sich direkt auf Analyse und KI aus.

Anwendungs- und Kontextbereich

Der Anwendungs- und Kontextbereich ergänzt die fachlichen Informatikmodule um anwendungsbezogene und außerfachliche Perspektiven. Das Modul dient als flexibler Platzhalter für anerkannte oder besonders ausgewiesene Lehrangebote aus Anwendungsgebieten der Informatik sowie aus gesellschaftlichen, wirtschaftlichen, organisatorischen, rechtlichen, ethischen, ökologischen oder kommunikativen Kontexten.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Bereichs ausgewählte anwendungsbezogene oder kontextbezogene Fragestellungen fachlich fundiert bearbeiten und die dabei erworbenen Kompetenzen in Bezug auf informatische Tätigkeitsfelder, technische Entwicklungen und ihr individuelles Studienprofil einordnen.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Modulbereich
  - fachliche Inhalte aus einem anerkannten oder besonders ausgewiesenen Anwendungs- oder Kontextangebot erschließen,
  - gesellschaftliche, wirtschaftliche, organisatorische, rechtliche, ethische, ökologische oder kommunikative Perspektiven auf technische Entwicklungen nachvollziehen,
  - Aufgabenstellungen, Positionen, Lösungswege und Ergebnisse nachvollziehbar darstellen,
  - die erworbenen Kompetenzen in Bezug auf das eigene Studium, informatische Berufsfelder und verantwortliche Technikgestaltung reflektieren.
- **WOZU**: Damit sie ihr Informatikstudium durch passende Anwendungs- und Kontextangebote sinnvoll ergänzen und technische Entscheidungen in größeren anwendungsbezogenen, gesellschaftlichen, rechtlichen oder ethischen Zusammenhängen beurteilen können. Der Bereich schafft Flexibilität für externe, internationale oder wechselnde Angebote.

Kurzporträt Anwendungs- und Kontextbereich

Worum geht es? Der Bereich schafft Raum für ausgewählte Anwendungsgebiete der Informatik sowie für Kontextthemen wie Ethik, Recht, Organisation, Wirtschaft oder Gesellschaft.
Wofür braucht man es? Studierende können interne, externe oder internationale Lehrangebote in ihr Studium einbringen und ihr Profil um anwendungs- und kontextbezogene Perspektiven erweitern.
Wieso ist es interessant? Informatik wirkt in konkreten Anwendungsfeldern und gesellschaftlichen Zusammenhängen. Der Bereich ermöglicht, solche Zusammenhänge flexibel aufzugreifen und passende Zusatzkompetenzen ins Masterstudium einzubinden.

MIN211: Arbeitswelt im gesellschaftlichen Wandel

Modulverantwortung	Prof. Dr. Christoph Dalitz	**Lehrperson(en)**	Dr. Dr. Klaus Henning
Verwendbarkeit	Master Informatik	**Fächergruppe**	Anwendungs- und Kontextbereich
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	4SL	**Engl. Titel**	The Changing World of Work in Society
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, zentrale Entwicklungslinien der Arbeitswelt zu benennen, quantitative und qualitative Trends des Arbeitsmarktes zu beschreiben sowie Ideen und Forderungen zur politischen Gestaltung von technischem Wandel, Automatisierung und Digitalisierung wirtschafts-, gesellschafts- und sozialpolitisch kritisch zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Entstehung, Grundlagen und Funktionsweise des Arbeitsmarktes sowie die Entwicklung von Arbeitsvolumen und Arbeitslosigkeit nachvollziehen,
- Formen und Ebenen der Arbeitsbeziehungen historisch einordnen und ihre Folgen für Arbeitsbedingungen untersuchen,
- Ziele, Formen und Entstehung wohlfahrtsstaatlicher Politik vergleichen und den deutschen Wohlfahrtsstaat international einordnen,
- Trends des Arbeitsmarktes anhand von Statistiken, Fachtexten und aktuellen Fallbeispielen analysieren,
- wirtschafts-, gesellschafts- und sozialpolitische Gestaltungsvorschläge in Diskussionen und Gruppenarbeiten begründet abwägen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um die gesellschaftlichen Folgen von Automatisierung und Digitalisierung einzuschätzen, an deren technischer Grundlage sie beruflich mitwirken. Sie können technische Entwicklungen in arbeitsmarkt-, sozial- und wirtschaftspolitische Zusammenhänge einordnen und ihre eigene Rolle in betrieblichen wie gesellschaftlichen Aushandlungsprozessen begründet bestimmen.

Inhalte

- Technischer Wandel, Automatisierung und Digitalisierung als Treiber veränderter Arbeitsprozesse
- Neue Qualifikationsanforderungen sowie Arbeits- und Unternehmensorganisationen
- Bedeutungsgewinn des Dienstleistungssektors und Wandel der klassischen Industriearbeit
- Wandel des Arbeitsmarktes: Entstehung, Grundlagen und prägende Trends
- Entwicklung des Arbeitsvolumens und Ursachen von Arbeitslosigkeit
- Wandel der Arbeitsbeziehungen: historische Entstehung, Formen und Ebenen
- Folgen veränderter Arbeitsbeziehungen für Arbeitsbedingungen und die Vertretung von Arbeitnehmerinteressen
- Wandel des Wohlfahrtsstaates: Ziele, Formen und Entstehung wohlfahrtsstaatlicher Politik
- Der deutsche Wohlfahrtsstaat im internationalen Vergleich
- Aktuelle Herausforderungen des Wohlfahrtsstaates und Ansätze zu seiner Reform

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Mündliche Prüfung. Im Prüfungsgespräch ordnen die Studierenden zentrale Entwicklungslinien der Arbeitswelt ein, belegen quantitative und qualitative Trends des Arbeitsmarktes an Beispielen und bewerten wirtschafts-, gesellschafts- und sozialpolitische Gestaltungsansätze begründet.

Literatur

- themenabhängig

Kurzporträt MIN211: Arbeitswelt im gesellschaftlichen Wandel

Worum geht es? Das Modul untersucht, wie technischer Wandel, Automatisierung und Digitalisierung die Arbeitswelt verändern. Behandelt werden der Arbeitsmarkt und seine Trends, die Arbeitsbeziehungen zwischen Beschäftigten, Unternehmen und Verbänden sowie der Wohlfahrtsstaat mit seinen Leistungen, Grenzen und Reformdebatten. Vorlesung, Diskussion und Gruppenarbeit wechseln sich dabei ab.
Wofür braucht man es? Informatische Systeme automatisieren Tätigkeiten, verschieben Qualifikationsanforderungen und verändern ganze Berufsbilder. Wer solche Systeme entwirft, trifft damit Entscheidungen mit arbeitsmarkt- und sozialpolitischer Wirkung. Das Modul liefert das begriffliche und empirische Rüstzeug, um diese Wirkungen einzuschätzen und in betrieblichen wie politischen Debatten sachlich zu vertreten.
Wieso ist es interessant? Ob Digitalisierung Arbeit vernichtet oder nur verlagert, wird seit zwei Jahrhunderten gestritten – mit erstaunlich wiederkehrenden Argumenten. Das Modul führt an die Daten heran, mit denen sich solche Behauptungen prüfen lassen, und macht sichtbar, dass die Folgen technischen Wandels nicht der Technik allein entspringen, sondern politisch gestaltet werden.

MIN299: Ausgewählte Fragen im Anwendungs- und Kontextbereich

Modulverantwortung	Studiendekan:in des Studiengangs Informatik	**Lehrperson(en)**	Alle Lehrenden des Studiengangs
Verwendbarkeit	Master Informatik	**Fächergruppe**	Anwendungs- und Kontextbereich
Modultyp	Wahl	**Sprache**	Deutsch oder Englisch
Angebot	nach Anerkennung oder gemäß Wahlpflichtangebot	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	nach anerkannter Lehrveranstaltung	**Engl. Titel**	Selected Topics in Application and Context Studies
Workload	Präsenzstudium: nach anerkannter Lehrveranstaltung Selbststudium: nach anerkannter Lehrveranstaltung		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, ausgewählte anwendungsbezogene oder kontextbezogene Fragestellungen ergänzend zum regulären Modulangebot fachlich fundiert zu bearbeiten und die dabei erworbenen Kompetenzen in ihr individuelles Studienprofil sowie in informatische Berufsfelder einzuordnen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- fachliche Inhalte aus einem anerkannten oder besonders ausgewiesenen Anwendungs- oder Kontextangebot erschließen,
- gesellschaftliche, wirtschaftliche, organisatorische, rechtliche, ethische, ökologische oder kommunikative Fragestellungen bearbeiten,
- Aufgabenstellungen, Positionen, Lösungswege und Ergebnisse nachvollziehbar dokumentieren,
- die erworbenen Kompetenzen in Bezug auf das eigene Studium, informatische Berufsfelder und verantwortliche Technikgestaltung reflektieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um ihr Informatikstudium durch anerkannte externe, internationale oder wechselnde interne Anwendungs- und Kontextangebote sinnvoll zu ergänzen und technische Entwicklungen in größeren Kontexten beurteilen zu können.

Inhalte

- Anerkannte oder besonders ausgewiesene Anwendungs- und Kontextangebote
- Gesellschaftliche, rechtliche, ethische, wirtschaftliche oder organisatorische Kontexte der Informatik
- Themen, Methoden und Werkzeuge abhängig von der konkreten Lehrveranstaltung
- Fachliche Einordnung der erworbenen Kompetenzen in das Informatikstudium und informatische Berufsfelder

Prüfungsvorleistung

Nach anerkannter Lehrveranstaltung.

Prüfungsleistung

Nach anerkannter Lehrveranstaltung oder gemäß Anerkennungsentscheidung.

Literatur

- Literatur, Materialien, Dokumentation oder weitere Quellen werden abhängig von den konkreten Inhalten durch die Lehrperson oder im Rahmen der Anerkennung bekanntgegeben.

Kurzporträt MIN299: Ausgewählte Fragen im Anwendungs- und Kontextbereich

Worum geht es? Das Modul dient als flexibler Platzhalter für anerkannte oder besonders ausgewiesene Anwendungs- und Kontextangebote. Die konkreten Inhalte hängen von der jeweiligen Lehrveranstaltung oder Anerkennungsentscheidung ab.
Wofür braucht man es? Es ermöglicht Studierenden, externe, internationale oder wechselnde interne Angebote in ihr Studium einzubringen und ihr fachliches Profil um anwendungs- und kontextbezogene Perspektiven zu ergänzen.
Wieso ist es interessant? Technische Entscheidungen wirken immer in konkreten Anwendungsfeldern sowie in gesellschaftlichen, rechtlichen, ethischen, wirtschaftlichen und organisatorischen Zusammenhängen. Das Modul schafft Raum, passende Zusatzkompetenzen anzuerkennen und sinnvoll in das Masterstudium einzubinden.

Softwareentwicklungsprojekt

Das Softwareentwicklungsprojekt ist ein zentraler projektorientierter Studienbereich im Master Informatik. Es vermittelt fortgeschrittene Kompetenzen in der professionellen, arbeitsteiligen und wirtschaftlich verantworteten Entwicklung langlebiger Softwareprodukte.

Die Studierenden durchlaufen drei aufeinanderfolgende Module mit jeweils 5 ECTS. Sie arbeiten semesterübergreifend im gemeinsamen organisatorischen Rahmen eines fiktiven Softwareentwicklungsunternehmens, übernehmen jedoch abhängig von ihrem Studienfortschritt unterschiedliche Rollen, Aufgaben und Verantwortung. Dabei können sie an demselben Produkt oder an unterschiedlichen Produkten eines gemeinsamen Portfolios arbeiten.

Die methodische Rahmung ist die eines Planspiels, in dem technische, organisatorische und wirtschaftliche Prozesse professioneller Softwareentwicklung nachgebildet werden. Die Softwareentwicklung selbst erfolgt an lauffähigen Produkten, die von den Studierenden analysiert, verändert, getestet, dokumentiert, integriert und weiterentwickelt werden. Auf diese Weise werden typische Situationen realer Entwicklungsorganisationen erfahrbar, darunter Einarbeitung und Onboarding, arbeitsteilige Umsetzung, Qualitätssicherung, technische Koordination, Release und Betrieb, Integration fremder Beiträge sowie Übergabe und Wissenssicherung.

Die Studierenden übernehmen mit zunehmendem Studienfortschritt Rollen wie Junior oder Senior Developer, Feature-Verantwortliche, Technical Lead, Product Owner oder Teilprojektleitung. Sie bearbeiten Anforderungen und Aufträge, planen Aufgaben und Releases, treffen technische Entscheidungen und stimmen sich mit weiteren Projektbeteiligten ab. Dabei berücksichtigen sie begrenzte Zeit-, Budget- und Ressourcenvorgaben sowie die Auswirkungen ihrer Entscheidungen auf Qualität, Wartbarkeit, Betrieb und Lebenszykluskosten.

Die technische, organisatorische und wirtschaftliche Verantwortung steigt über die drei Module systematisch an:

1. **Softwareentwicklungsprojekt 1: Systemverständnis und produktive Mitarbeit**
   Die Studierenden erschließen sich ein bestehendes, langlebiges und gewachsenes Softwareprodukt. Sie analysieren Produktvision, fachliche Domäne, Architektur, Codebasis und Entwicklungsprozesse, arbeiten sich in die Projektorganisation ein und leisten erste abgegrenzte, qualitätsgesicherte Beiträge. Dabei ordnen sie ihre Aufgaben in den betrieblichen Zweck des Produkts ein und schätzen den Aufwand ihrer eigenen Tätigkeiten.
2. **Softwareentwicklungsprojekt 2: Umsetzung und Qualität**
   Die Studierenden übernehmen größere, nicht vollständig vorstrukturierte Umsetzungspakete in einem bestehenden Produktkontext. Sie planen Arbeitspakete und Ressourcen, entwickeln tragfähige Lösungsansätze und setzen Features, Schnittstellen oder technische Verbesserungen qualitätsgesichert um. Technische Alternativen bewerten sie auch hinsichtlich Aufwand, Risiken, Betriebsfolgen, Wartbarkeit und Lebenszykluskosten. Ergänzend übernehmen sie zeitweise die fachliche Koordination kleinerer Arbeitsgruppen.
3. **Softwareentwicklungsprojekt 3: Technische und wirtschaftliche Teilprojektleitung**
   Die Studierenden koordinieren abgegrenzte Produkt- oder Teilprojektbereiche. Sie priorisieren Anforderungen, planen Roadmaps, Releases, Kapazitäten und Ressourceneinsatz, bereiten technische und wirtschaftliche Entscheidungen vor und koordinieren Beiträge weiterer Projektbeteiligter. Dazu gehören Kunden- und Anforderungsabstimmung, Reviews und Abnahmen, Onboarding, Übergabe und Wissenssicherung sowie die Steuerung von Risiken, Abhängigkeiten und Planabweichungen. Wirtschaftliche Aufgaben umfassen insbesondere Aufwandsschätzung, Angebotskalkulation und -bewertung, Kosten- und Terminprognosen, Make-or-Buy- und Beschaffungsentscheidungen sowie die Bewertung von Wertbeitrag und Lebenszykluskosten.

Nach Abschluss der Modulfolge sind die Studierenden in der Lage, über einzelne Entwicklungsaufgaben hinaus Verantwortung für größere Softwareprodukte und arbeitsteilige Entwicklungsprozesse zu übernehmen. Sie können bestehende Produkte erschließen, größere Teilaufgaben eigenständig und qualitätsgesichert umsetzen sowie abgegrenzte Produktbereiche technisch, organisatorisch und wirtschaftlich koordinieren.

Das Softwareentwicklungsprojekt ergänzt die fachlichen Kernmodule und ist vom Forschungs- und Entwicklungsprojekt abgegrenzt. Während dort wissenschaftliche Fragestellungen, Forschungsmethoden und Evaluation im Mittelpunkt stehen, fokussiert das Softwareentwicklungsprojekt die professionelle und nachhaltige Weiterentwicklung langlebiger Softwareprodukte.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Softwareentwicklungsprojekts bestehende Softwareprodukte systematisch erschließen, größere Entwicklungsaufgaben qualitätsgesichert umsetzen und abgegrenzte Produkt- oder Teilprojektbereiche technisch, organisatorisch und wirtschaftlich koordinieren. Sie entwickeln dabei Kompetenzen in Systemverständnis, Softwarearchitektur, Qualitätssicherung, Teamarbeit, Projektsteuerung, Dokumentation und wirtschaftlicher Bewertung technischer Entscheidungen.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Modulbereich
  - Produktvision, fachliche Domäne, Architektur, Codebasis und Entwicklungsprozesse bestehender Systeme analysieren,
  - Anforderungen, Tickets, Epics und Arbeitspakete konkretisieren, schätzen, planen und umsetzen,
  - Features, Schnittstellen oder technische Verbesserungen implementieren, testen, reviewen und integrieren,
  - Entwicklungsprozesse, Reviews, Iterationen, Dokumentation, Deployment- und Betriebsaspekte praktisch anwenden,
  - technische Alternativen hinsichtlich Qualität, Aufwand, Risiken, Wartbarkeit, Betriebsfolgen und Lebenszykluskosten bewerten,
  - schrittweise mehr Verantwortung für Koordination, Priorisierung, Übergabe und wirtschaftliche Entscheidungen übernehmen.
- **WOZU**: Damit sie professionelle Softwareentwicklung nicht nur als Implementierungsaufgabe verstehen, sondern als nachhaltige Weiterentwicklung komplexer Produkte in arbeitsteiligen Organisationen. Der Modulbereich bereitet auf verantwortliche Rollen in Softwareentwicklung, Architektur, technischer Koordination, Produktentwicklung und Projektleitung vor.

Kurzporträt Softwareentwicklungsprojekt

Worum geht es? Das Softwareentwicklungsprojekt bildet eine realitätsnahe Entwicklungsorganisation nach. Studierende arbeiten an langlebigen Softwareprodukten, steigen in bestehende Systeme ein, setzen anspruchsvolle Änderungen um und übernehmen zunehmend Verantwortung für Qualität, Koordination und wirtschaftliche Folgen.
Wofür braucht man es? In der Praxis entsteht Software selten auf der grünen Wiese. Die Kompetenzen helfen, gewachsene Systeme zu verstehen, Änderungen sauber zu planen, technische Entscheidungen nachvollziehbar zu begründen und im Team verlässlich zu liefern.
Wieso ist es interessant? Der Bereich macht professionelle Softwareentwicklung erfahrbar: mit echten Abhängigkeiten, unvollständigen Informationen, Reviews, Betrieb, Aufwandsschätzung und Rollenwechseln. Dadurch wird deutlich, wie technische Qualität, Organisation und wirtschaftliche Verantwortung zusammenhängen.

MIN301: Softwareentwicklungsprojekt 1: Systemverständnis und produktive Mitarbeit

Modulverantwortung	Studiendekan:in des Fachbereichs	**Lehrperson(en)**	Professor:innen des Studiengangs Informatik
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2SL+2P	**Engl. Titel**	Software Development Project 1: System Understanding and Productive Contribution
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Kompetenzen in Programmierung, Software Engineering, Softwareentwurf, Qualitätssicherung und Versionsverwaltung auf dem Niveau eines abgeschlossenen Bachelorstudiums der Informatik. Die Studierenden sollten in der Lage sein, bestehende Codebasen, technische Dokumentation und Entwicklungsprozesse eigenständig zu erschließen und in arbeitsteiligen Softwareentwicklungskontexten mitzuwirken.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, sich innerhalb eines realitätsnahen Entwicklungsbetriebs systematisch in ein bestehendes, langlebiges und gewachsenes Softwareprodukt einzuarbeiten. Sie können dessen fachlichen und betrieblichen Zweck, Architektur, Codebasis und Entwicklungsprozesse erschließen und abgegrenzte Beiträge unter Berücksichtigung von Qualitätsanforderungen, Aufwand und Auswirkungen auf das Gesamtprodukt nachvollziehbar umsetzen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Produktvision, fachliche Domäne, relevante Anspruchsgruppen und zentrale Nutzungsszenarien analysieren und den betrieblichen Zweck des Produkts erläutern,
- Architektur, Codebasis, Schnittstellen, technische Abhängigkeiten und bestehende Entwurfsentscheidungen für einen abgegrenzten Produktbereich nachvollziehen,
- sich anhand vorhandener Dokumentation, Entwicklungsartefakte und strukturierter Onboarding-Prozesse in Produkt, Team, Entwicklungsumgebung und Arbeitsabläufe einarbeiten,
- Anforderungen, Tickets und Akzeptanzkriterien für abgegrenzte Entwicklungsaufgaben verstehen, offene Fragen klären und den eigenen Aufwand schätzen,
- Änderungen an bestehenden Produktteilen entwerfen, lauffähig implementieren, versionieren und in den vorhandenen Build- und Integrationsprozess einbringen,
- ihre Beiträge durch geeignete Tests, Code Reviews und technische Dokumentation qualitätssichern und erhaltenes Feedback in die Weiterentwicklung einbeziehen,
- KI-gestützte Entwicklungswerkzeuge transparent und qualitätsbewusst einsetzen und deren Ergebnisse kritisch überprüfen,
- die Auswirkungen ihrer Änderungen auf angrenzende Systemteile, Wartbarkeit, Betrieb, Weiterentwicklungsfähigkeit und Wertbeitrag des Produkts einschätzen und ihren eigenen Beitrag reflektieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um in langlebigen Softwareprodukten auch bei unvollständigem Vorwissen fundiert handlungsfähig zu werden und produktiv an professioneller Softwareentwicklung mitwirken zu können.

Inhalte

- Produktvision, fachliche Domäne und Nutzungsszenarien
- Aufbau, Zweck und fachliche Grenzen eines Softwareprodukts im Produktportfolio eines fiktiven Unternehmens
- Architekturüberblick, zentrale Komponenten und Schnittstellen gewachsener Softwaresysteme
- Codebasis, technische Abhängigkeiten, Modulstruktur und bestehende Entwurfsentscheidungen
- Entwicklungsumgebung, Build-Prozess und lokale Ausführung
- Bewerbungs- und Onboarding-Simulation mit Kurzprofil, Gespräch, Teamzuordnung sowie Einrichtung von Arbeitsumgebung und Berechtigungen
- Projektorganisation, Kommunikationswege, organisatorische Abläufe und Entscheidungsprozesse im Planspielrahmen eines Softwareentwicklungsunternehmens
- Anforderungen, Tickets, Issues und vorhandene Projektdokumentation
- Lauffähige Umsetzung abgegrenzter Änderungen an bestehenden Produktteilen
- Code Reviews sowie -Tests, z.B. Unit- und Modultests
- Transparenter, datenschutz- und qualitätsbewusster Einsatz von LLMs und Softwareagenten
- Einschätzung von Auswirkungen eigener Änderungen auf angrenzende Systemteile, Wartbarkeit und Weiterentwicklungsfähigkeit sowie Dokumentation und Begründung eigener Beiträge im Produktkontext

Prüfungsvorleistung

Keine separate Prüfungsvorleistung. Die semesterbegleitenden Meilensteine, Präsentationen und Reflexionsanteile sind Bestandteil der Prüfungsleistung.

Prüfungsleistung

Studien- oder Projektarbeit. Sie wird als Projektportfolio mit Fachgespräch ausgestaltet. Das Portfolio dokumentiert die systematische Einarbeitung in Produkt, Architektur, Codebasis und Entwicklungsprozesse, die bearbeiteten Anforderungen und die Schätzung des eigenen Aufwands, die lauffähig umgesetzten und integrierten Softwarebeiträge, die angewandten Test-, Review- und Qualitätssicherungsmaßnahmen, die Einordnung der Beiträge in den fachlichen, technischen und betrieblichen Produktkontext sowie die Reflexion des eigenen Vorgehens und der Zusammenarbeit im Entwicklungsteam. Das Portfolio entsteht semesterbegleitend entlang vereinbarter Meilensteine. Die Studierenden stellen ihren Arbeitsstand und ihre Beiträge regelmäßig in Weeklys, Reviews und Retrospektiven vor und berücksichtigen erhaltenes Feedback bei der weiteren Bearbeitung. Das Fachgespräch dient der individuellen Prüfung des Systemverständnisses, der umgesetzten Beiträge, der Qualitätsmaßnahmen und der Einordnung der eigenen Arbeit in das Gesamtprodukt.

Literatur

- Sommerville, Ian: Software Engineering. 10., aktualisierte Auflage. Pearson Studium, 2018. ISBN 978-3-86894-344-3.
- Sommerville, Ian: Modernes Software-Engineering: Entwurf und Entwicklung von Softwareprodukten. 1. Auflage, Pearson Studium, 2020. ISBN 978-3-86894-396-2.
- Fowler, Martin: Refactoring: Improving the Design of Existing Code. 2nd Edition. Addison-Wesley Professional, 2018. ISBN 978-0-13-475759-9.
- Weitere produkt-, technologie- und kontextspezifische Fachliteratur sowie Dokumentation werden semesterbezogen in der jeweils aktuellen Fassung bereitgestellt.

Kurzporträt MIN301: Softwareentwicklungsprojekt 1: Systemverständnis und produktive Mitarbeit

Worum geht es? Softwareentwicklungsprojekt 1 führt in ein bestehendes, langlebiges Softwareprodukt und dessen Entwicklungsorganisation ein. Die Studierenden analysieren Produktvision, Domäne, Architektur, Codebasis und Prozesse und leisten erste qualitätsgesicherte Beiträge.
Wofür braucht man es? Die Kompetenzen werden gebraucht, um in gewachsenen Softwareprojekten schnell handlungsfähig zu werden. Sie helfen beim Onboarding, beim Verstehen fremder Systeme und bei der Umsetzung abgegrenzter Änderungen ohne das Gesamtprodukt aus dem Blick zu verlieren.
Wieso ist es interessant? Spannend ist das Modul, weil reale Software selten vollständig erklärt oder perfekt strukturiert ist. Studierende lernen, mit vorhandenen Artefakten, unvollständigem Wissen und Teamprozessen produktiv umzugehen.

MIN302: Softwareentwicklungsprojekt 2: Umsetzung und Qualität

Modulverantwortung	Studiendekan:in des Fachbereichs	**Lehrperson(en)**	Professor:innen des Studiengangs Informatik
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2SL+2P	**Engl. Titel**	Software Development Project 2 (Implementation and Quality)
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Empfohlen werden Kompetenzen entsprechend Softwareentwicklungsprojekt 1 oder vergleichbare Kompetenzen in der systematischen Einarbeitung in bestehende Softwareprodukte, der Umsetzung und Integration abgegrenzter Änderungen, der Versionsverwaltung sowie der Anwendung grundlegender Qualitätssicherungsmaßnahmen. Die Studierenden sollten bestehende Architektur- und Code-Strukturen nachvollziehen, Entwicklungsaufgaben eigenständig bearbeiten und ihre Beiträge im Produktkontext begründen können.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, ein größeres, nicht vollständig vorstrukturiertes Umsetzungspaket in einem bestehenden Softwareprodukt eigenständig zu planen, zu entwerfen, qualitätsgesichert umzusetzen und in das Gesamtprodukt zu integrieren. Sie können technische Lösungsalternativen unter Berücksichtigung von Qualität, Entwicklungsaufwand, Risiken, Betriebsfolgen und Lebenszykluskosten bewerten sowie die Umsetzung in einer kleinen Arbeitsgruppe fachlich koordinieren.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Anforderungen für einen abgegrenzten Produktbereich analysieren, als Epic mit Akzeptanzkriterien und Definition of Done konkretisieren und in tragfähige Arbeitspakete überführen,
- das Umsetzungspaket in Arbeitspakete zerlegen, Abhängigkeiten und Risiken identifizieren, Aufwände und Ressourcen schätzen und den Bearbeitungsfortschritt verfolgen,
- alternative Lösungsansätze mit Blick auf Architektur, Schnittstellen und bestehende Code-Strukturen entwerfen und hinsichtlich technischer Eignung, Entwicklungsaufwand, Betriebsfolgen und Folgekosten vergleichen„
- Features, Schnittstellen oder technische Verbesserungen implementieren, versionieren und in das bestehende Softwareprodukt integrieren„
- eine geeignete Qualitätssicherungsstrategie entwickeln und automatisierte Tests, Integrationstests, Reviews und Continuous-Integration-Verfahren anwenden,
- Deployment-, Betriebs-, Abhängigkeits-, Lizenz- und Sicherheitsaspekte berücksichtigen und auf kleinere Betriebs- oder Sicherheitsstörungen angemessen reagieren,
- die Umsetzung in einer kleinen Arbeitsgruppe fachlich koordinieren, Aufgaben abstimmen, Beiträge anderer prüfen und weniger erfahrene Teammitglieder unterstützen,
- technische Entscheidungen, Qualitätsmaßnahmen, Aufwände und Abweichungen nachvollziehbar dokumentieren, die Auswirkungen auf Wartbarkeit, Betrieb und Lebenszykluskosten bewerten und die Ergebnisse fachlich vertreten.

WOZU. Die Studierenden nutzen diese Kompetenzen, um in größeren Softwareprodukten Verantwortung für die vollständige Planung, Umsetzung, Qualitätssicherung und Integration eines abgegrenzten Features oder technischen Teilbereichs zu übernehmen. Sie können technische Entscheidungen nicht nur hinsichtlich ihrer unmittelbaren Funktionalität, sondern auch hinsichtlich Aufwand, Risiken, Betriebsfähigkeit, Wartbarkeit und wirtschaftlicher Folgewirkungen beurteilen. Das Modul bereitet sie darauf vor, in Softwareentwicklungsprojekt 3 über einzelne Umsetzungspakete hinaus Produkt- oder Teilprojektbereiche technisch, organisatorisch und wirtschaftlich zu koordinieren.

Inhalte

- Anforderungsanalyse und Epic-Planung mit Akzeptanzkriterien, Definition of Done, Abgrenzung und Abhängigkeiten
- Zerlegung größerer Aufgaben in Arbeitspakete, Aufwandsschätzung, Ressourcenplanung und Fortschrittskontrolle
- Entwurf und Bewertung von Lösungsalternativen im bestehenden Architektur-, Schnittstellen- und Code-Kontext
- Implementierung, Versionierung und Integration größerer Features, Schnittstellen oder technischer Verbesserungen
- Qualitätssicherungsstrategien, Testkonzepte, Testautomatisierung, Integrationstests, Code Reviews und Continuous Integration
- Deployment, Betrieb, Monitoring sowie Abhängigkeits-, Schwachstellen-, Lizenz- und Incident-Management
- Technische Schulden, Qualitätskosten, Entwicklungs-, Betriebs- und Wartungsaufwände sowie Lebenszykluswirkungen technischer Entscheidungen
- Fachliche Koordination einer kleinen Arbeitsgruppe, Integration fremder Beiträge, KI-gestützte Entwicklung, Dokumentation und Reflexion

Prüfungsvorleistung

Keine separate Prüfungsvorleistung. Die semesterbegleitenden Meilensteine, Präsentationen und Reflexionsanteile sind Bestandteil der Prüfungsleistung.

Prüfungsleistung

Studien- oder Projektarbeit. Sie wird als Projektportfolio mit Fachgespräch ausgestaltet. Das Portfolio dokumentiert die Analyse und Konkretisierung des übernommenen Umsetzungspakets, die Arbeits-, Aufwands- und Ressourcenplanung, den Entwurf und die Bewertung wesentlicher

Lösungsalternativen, die lauffähige Implementierung und Integration des Softwarebeitrags, die angewandten Test-, Review- und Qualitätssicherungsmaßnahmen, die Berücksichtigung von Deployment, Betrieb, Sicherheit und Abhängigkeiten, die Bewertung von Entwicklungsaufwand, Betriebsfolgen und Lebenszykluskosten sowie die fachliche Koordination, die individuellen Beiträge und die Reflexion des Vorgehens. Das Portfolio entsteht semesterbegleitend entlang vereinbarter Meilensteine. Die Studierenden stellen Arbeitsstand, technische Entscheidungen, Qualitätsmaßnahmen und Abweichungen regelmäßig in Weeklys, Reviews und Retrospektiven vor und beziehen erhaltenes Feedback in die weitere Bearbeitung ein. Das Fachgespräch dient der individuellen Prüfung des Verständnisses der technischen Entscheidungen, der Qualitätssicherung, der wirtschaftlichen Folgewirkungen und der eigenen Rolle im Produktkontext.

Literatur

- Sommerville, Ian: Modernes Software-Engineering: Entwurf und Entwicklung von Softwareprodukten. 1. Auflage, Pearson Studium, 2020. ISBN 978-3-86894-396-2.
- Meszaros, Gerard: xUnit Test Patterns: Refactoring Test Code. Addison-Wesley Professional, 1st Edition, 2007. ISBN 978-0-13-149505-0.
- Fowler, Martin: Refactoring: Improving the Design of Existing Code. 2nd Edition. Addison-Wesley Professional, 2018. ISBN 978-0-13-475759-9.
- Starke, Gernot: Effektive Softwarearchitekturen: Ein praktischer Leitfaden. 10., überarbeitete Auflage. Carl Hanser Verlag, 2024. ISBN 978-3-446-47672-1.
- Weitere produkt-, technologie- und kontextspezifische Fachliteratur sowie Dokumentation werden semesterbezogen in der jeweils aktuellen Fassung bereitgestellt.

Kurzporträt MIN302: Softwareentwicklungsprojekt 2: Umsetzung und Qualität

Worum geht es? Softwareentwicklungsprojekt 2 konzentriert sich auf größere Umsetzungspakete in einem bestehenden Produktkontext. Die Studierenden planen Arbeitspakete, entwerfen Lösungsalternativen, setzen Features oder technische Verbesserungen um und sichern Qualität durch Tests, Reviews und Integration.
Wofür braucht man es? Die Kompetenzen werden gebraucht, um nicht nur einzelne Tickets, sondern zusammenhängende Entwicklungsaufgaben verantwortlich zu bearbeiten. Sie helfen, Aufwand, Risiken, Betriebsfolgen, Wartbarkeit und technische Qualität in Entscheidungen einzubeziehen.
Wieso ist es interessant? Interessant ist das Modul, weil technische Umsetzung hier mit Planung und Verantwortung zusammenkommt. Eine Lösung muss nicht nur funktionieren, sondern in Architektur, Teamprozess, Betrieb und Lebenszyklus des Produkts passen.

MIN303: Softwareentwicklungsprojekt 3: Technische und wirtschaftliche Teilprojektleitung

Modulverantwortung	Studiendekan:in des Fachbereichs	**Lehrperson(en)**	Professor:innen des Studiengangs Informatik
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2SL+2P	**Engl. Titel**	Software Development Project 3: Technical and Economic Subproject Leadership
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Empfohlen werden Kompetenzen entsprechend Softwareentwicklungsprojekt 1 und 2 oder vergleichbare Kompetenzen in der systematischen Einarbeitung in bestehende Softwareprodukte, der eigenständigen Planung und qualitätsgesicherten Umsetzung größerer Entwicklungspakete sowie der Abstimmung und Integration arbeitsteilig erbrachter Softwarebeiträge. Die Studierenden sollten technische Entscheidungen im Produktkontext begründen, Aufwände und Risiken einschätzen und kleinere Arbeitsgruppen fachlich koordinieren können.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, die Weiterentwicklung eines abgegrenzten Bereichs eines langlebigen Softwareprodukts technisch, organisatorisch und wirtschaftlich zu koordinieren. Sie können Kundenbedarfe und Produktziele in eine priorisierte Planung überführen, Ressourcen, Aufwände, Kosten, Risiken und Abhängigkeiten berücksichtigen, Beiträge weiterer Projektbeteiligter fachlich einordnen und integrieren sowie Entscheidungen gegenüber technischen und nichttechnischen Anspruchsgruppen begründet vertreten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Kundenbedarfe, Produktziele, Anforderungen, technische Abhängigkeiten und Rahmenbedingungen analysieren und in eine abgestimmte Spezifikation mit überprüfbaren Abnahme- und Erfolgskriterien überführen,
- Aufgaben nach Wertbeitrag, Aufwand, Risiken und Abhängigkeiten priorisieren und daraus eine Roadmap, Releases, Meilensteine und bearbeitbare Arbeitspakete ableiten,
- technische Lösungsalternativen bewerten und dabei Qualität, Wartbarkeit, Sicherheit, Betrieb, Ressourceneinsatz und Lebenszykluskosten berücksichtigen,
- Aufwände, Kapazitäten, Kosten und Termine planen, den Projektfortschritt anhand geeigneter Kennzahlen beurteilen und bei Abweichungen steuernd eingreifen,
- Angebote, Beschaffungs- oder Make-or-Buy-Alternativen projektbezogen analysieren und technische sowie wirtschaftliche Entscheidungsgrundlagen aufbereiten,
- Aufgaben für weitere Projektbeteiligte fachlich und technisch vorbereiten, deren Umsetzung begleiten und Beiträge hinsichtlich Anforderungen, Qualität und Integrationsfähigkeit überprüfen,
- Kunden-, Anforderungs-, Review- und Abnahmegespräche durchführen sowie Onboarding, Wissenssicherung, Release, Betriebsübergabe und Offboarding koordinieren,
- Entscheidungen, Risiken, Planänderungen, Integrationsprozesse und wirtschaftliche Auswirkungen nachvollziehbar dokumentieren und ihre Koordinations- und Führungsrolle reflektieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um Verantwortung für abgegrenzte Produkt- oder Teilprojektbereiche in arbeitsteiligen Softwareentwicklungsorganisationen zu übernehmen. Sie können technische Weiterentwicklung, Projektsteuerung und wirtschaftliche Rahmenbedingungen miteinander verbinden, Beiträge unterschiedlicher Beteiligter koordinieren und die nachhaltige Weiterentwicklung sowie den Betrieb eines Softwareprodukts sicherstellen.

Inhalte

- Kundenbedarfe, Produktziele, Stakeholder und abgestimmte Spezifikationen einschließlich Abnahme- und Erfolgskriterien
- Priorisierung nach Wertbeitrag, Aufwand und Risiko sowie Roadmap-, Release-, Meilenstein- und Aufgabenplanung
- Technische Entscheidungsfindung unter Berücksichtigung von Architektur, Qualität, Sicherheit, Betrieb und Weiterentwicklungsfähigkeit
- Aufwandsschätzung, Kapazitäts- und Ressourcenplanung, Kosten- und Terminprognosen sowie Projektcontrolling
- Wirtschaftliche Bewertung von Produktentscheidungen, Lebenszykluskosten, Angebotskalkulation und -bewertung sowie Make-or-Buy- und Beschaffungsentscheidungen
- Fachliche Koordination, Aufbereitung von Aufgaben, Review, Integration und Rückmeldung zu Beiträgen weiterer Projektbeteiligter
- Kundenkommunikation, Abnahme, Release, Betriebsübergabe, Onboarding, Offboarding und Wissenssicherung
- Risiko-, Abhängigkeits-, Änderungs- und Entscheidungsmanagement sowie Dokumentation und Reflexion der Teilprojektleitung

Prüfungsvorleistung

Keine separate Prüfungsvorleistung. Die semesterbegleitenden Meilensteine, Präsentationen und Reflexionsanteile sind Bestandteil der Prüfungsleistung.

Prüfungsleistung

Studien- oder Projektarbeit. Sie wird als Projektportfolio mit Fachgespräch ausgestaltet. Das Portfolio dokumentiert die Analyse des Kundenbedarfs und die Spezifikation des verantworteten Produkt- oder Teilprojektbereichs, die Priorisierung, Roadmap, Release-, Ressourcen-,

Kosten- und Terminplanung, die Vorbereitung und Begründung wesentlicher technischer und wirtschaftlicher Entscheidungen, die Koordination, Überprüfung und Integration der Beiträge weiterer Projektbeteiligter, das Risiko-, Änderungs- und Projektcontrolling, die durchgeführten Kunden-, Review-, Abnahme-, Übergabe- und Onboarding-Prozesse sowie die Reflexion der eigenen Koordinations- und Führungsrolle. Das Portfolio entsteht semesterbegleitend entlang vereinbarter Meilensteine. Die Studierenden stellen den Projektstatus, bestehende Abweichungen, offene Entscheidungen und zentrale Ergebnisse regelmäßig in Planungs-, Review- und Abstimmungsformaten vor und berücksichtigen erhaltenes Feedback in der weiteren Steuerung. Das Fachgespräch dient der individuellen Prüfung der technischen, organisatorischen und wirtschaftlichen Entscheidungsfähigkeit, des System- und Rollenverständnisses sowie der Koordinationsleistung.

Literatur

- Starke, Gernot: Effektive Softwarearchitekturen: Ein praktischer Leitfaden. 10., überarbeitete Auflage. Carl Hanser Verlag, 2024.
- Bass, Len; Clements, Paul; Kazman, Rick: Software Architecture in Practice. 4th Edition. Addison-Wesley Professional, 2021.
- Rupp, Chris; die SOPHISTen: Requirements-Engineering und -Management. Carl Hanser Verlag, 2014.
- Fournier, Camille: The Manager's Path: A Guide for Tech Leaders Navigating Growth and Change. O'Reilly Media, 2017.
- Weitere produkt- und kontextspezifische Dokumentation wird semesterbezogen bereitgestellt.

Kurzporträt MIN303: Softwareentwicklungsprojekt 3: Technische und wirtschaftliche Teilprojektleitung

Worum geht es? Softwareentwicklungsprojekt 3 behandelt die technische und wirtschaftliche Koordination abgegrenzter Produkt- oder Teilprojektbereiche. Die Studierenden priorisieren Anforderungen, planen Releases und Kapazitäten, koordinieren Beiträge und bereiten technische sowie wirtschaftliche Entscheidungen vor.
Wofür braucht man es? Die Kompetenzen werden für Rollen benötigt, in denen technische Verantwortung über die eigene Implementierung hinausgeht. Sie sind relevant für Technical Leads, Product Owner, Teilprojektleitungen und Architektur- oder Koordinationsaufgaben.
Wieso ist es interessant? Spannend ist das Modul, weil technische Entscheidungen hier ausdrücklich organisatorische und wirtschaftliche Folgen haben. Studierende erleben, wie Qualität, Aufwand, Risiko, Kundennutzen und Lebenszykluskosten gegeneinander abgewogen werden müssen.

Forschungs- und Entwicklungsprojekt

Das Forschungs- und Entwicklungsprojekt führt die Studierenden an offene, anwendungsorientierte und wissenschaftsnahe Fragestellungen der Informatik heran. Im Mittelpunkt stehen die eigenständige Eingrenzung einer Fragestellung, die Auswahl geeigneter Methoden, die Entwicklung oder Evaluation technischer Artefakte und die nachvollziehbare Darstellung der Ergebnisse.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Forschungs- und Entwicklungsprojekts eine aktuelle informatische Forschungs- oder Entwicklungsfrage formulieren, methodisch bearbeiten, Ergebnisse evaluieren und fachlich begründet einordnen. Sie entwickeln dabei Kompetenzen in wissenschaftlicher Recherche, methodischer Planung, Prototyping, Evaluation, kritischer Reflexion, Dokumentation und Präsentation.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Modulbereich
  - sich in den Stand der Technik beziehungsweise Forschung eines Themengebiets einarbeiten,
  - aus offenen Problemstellungen bearbeitbare Forschungs- oder Entwicklungsfragen ableiten,
  - geeignete wissenschaftliche, technische oder empirische Methoden auswählen und begründen,
  - prototypische Lösungen, Modelle, Verfahren, Experimente oder Studien entwickeln beziehungsweise anwenden,
  - Ergebnisse anhand geeigneter Kriterien auswerten und kritisch reflektieren,
  - ihre Arbeit in wissenschaftsnahen Formaten dokumentieren, präsentieren und diskutieren.
- **WOZU**: Damit sie komplexe und offene Aufgabenstellungen in Forschung, Entwicklung und anspruchsvoller Berufspraxis strukturiert bearbeiten können. Der Modulbereich bereitet unmittelbar auf die Masterarbeit vor und stärkt die Fähigkeit, technische Arbeit methodisch nachvollziehbar und fachlich belastbar zu begründen.

Kurzporträt Forschungs- und Entwicklungsprojekt

Worum geht es? Der Bereich dreht sich um offene Fragen, für die es nicht schon eine fertige Musterlösung gibt. Studierende recherchieren, planen ein methodisches Vorgehen, entwickeln oder evaluieren Lösungen und stellen ihre Ergebnisse wissenschaftsnah dar.
Wofür braucht man es? Forschung und Entwicklung verlangen die Fähigkeit, unklare Ausgangslagen zu strukturieren und Entscheidungen nachvollziehbar zu begründen. Diese Kompetenzen sind wichtig für Masterarbeiten, innovationsnahe Entwicklung, technische Evaluation und wissenschaftliche Anschlussfähigkeit.
Wieso ist es interessant? Das Projekt bietet Raum, aktuelle Themen der Informatik eigenständig zu vertiefen und eigene Fragestellungen zu verfolgen. Besonders reizvoll ist die Verbindung aus technischem Bauen, methodischer Untersuchung und kritischer Bewertung der Ergebnisse.

MIN401: Forschungs- und Entwicklungsprojekt

Modulverantwortung	Studiendekan:in des Fachbereichs	**Lehrperson(en)**	verschiedene Lehrende des Fachbereichs, abhängig von den angebotenen Forschungs- und Entwicklungsthemen
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	10 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2SL+4P	**Engl. Titel**	Research and Development Project
Workload	• Präsenz: 70 Std. • Selbst: 200 Std.		

Empfohlene Voraussetzungen

Empfohlen werden Kompetenzen in Informatik auf dem Niveau eines abgeschlossenen Bachelorstudiums sowie die Fähigkeit, sich eigenständig in fachliche und wissenschaftliche Fragestellungen einzuarbeiten. Je nach Projektthema können spezifische fachliche Vorkenntnisse erforderlich sein, die bei der Themenvergabe benannt werden.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine aktuelle Forschungs- oder Entwicklungsfragestellung der Informatik eigenständig beziehungsweise in einer Kleingruppe methodisch fundiert zu bearbeiten, wissenschaftlich zu dokumentieren, zu evaluieren und fachlich zu vertreten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- sich eigenständig in den Stand der Technik beziehungsweise den Stand der Forschung eines gewählten Themengebiets einarbeiten und aus einer offenen fachlichen Problemstellung eine bearbeitbare Forschungs- oder Entwicklungsfrage ableiten,
- geeignete wissenschaftliche, technische oder empirische Methoden zur Bearbeitung der Fragestellung auswählen und begründen,
- ein Forschungs- oder Entwicklungsprojekt in einer Kleingruppe planen, strukturieren und semesterbegleitend umsetzen,
- prototypische Lösungen, Modelle, Verfahren, Experimente, Studien oder technische Artefakte entwickeln beziehungsweise anwenden,
- geeignete Kriterien zur Bewertung der Ergebnisse festlegen, die Resultate nachvollziehbar auswerten und Zwischenergebnisse mit den betreuenden Lehrenden reflektieren,
- ihre Ergebnisse in Form eines wissenschaftlichen Reports beziehungsweise Papers darstellen und Rückmeldungen aus einem Review-Prozess zur Verbesserung nutzen,
- Projektergebnisse in einem Vortrag, Poster oder vergleichbaren Präsentationsformat aufbereiten und diskutieren,
- Grundsätze guter wissenschaftlicher Praxis berücksichtigen, einschließlich eines transparenten und reflektierten Umgangs mit KI-Werkzeugen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um offene, forschungsnahe und methodisch anspruchsvolle Fragestellungen im späteren beruflichen oder wissenschaftlichen Umfeld strukturiert, nachvollziehbar und wissenschaftlich begründet bearbeiten zu können.

Inhalte

- Bearbeitung einer aktuellen Forschungs- oder Entwicklungsfragestellung aus der Informatik oder einem angrenzenden ingenieurwissenschaftlichen, biomedizinischen beziehungsweise anwendungsbezogenen Kontext
- Themenfindung und Eingrenzung einer bearbeitbaren Forschungs- oder Entwicklungsfrage
- Recherche und Aufarbeitung des Stands der Technik beziehungsweise des Stands der Forschung
- Planung, Strukturierung und methodische Durchführung eines Forschungs- oder Entwicklungsprojekts
- Entwicklung, Umsetzung oder Anwendung prototypischer Lösungen, Experimente, Studien, Modelle oder technischer Artefakte
- Anwendungsbezogene Vertiefungen, z. B. Biosignalverarbeitung, medizinische Bildanalyse, datengetriebene Assistenzsysteme, Mensch-Technik-Interaktion, intelligente Sensorsysteme oder KI-gestützte Entscheidungsunterstützung
- Auswertung, Evaluation und kritische Reflexion der Projektergebnisse
- Wissenschaftliches Arbeiten und Schreiben, wissenschaftliches Englisch sowie Erstellung eines wissenschaftlichen Reports beziehungsweise Papers
- Diskussion, Review und Überarbeitung wissenschaftlicher Ergebnisse
- Gute wissenschaftliche Praxis und reflektierter Einsatz von KI-Werkzeugen
- Präsentation und Diskussion der Projektergebnisse, im Wintersemester vorzugsweise im Rahmen der Engineering Days; für das Sommersemester ist ein geeignetes Präsentationsformat noch festzulegen

Prüfungsvorleistung

Keine separate Prüfungsvorleistung. Semesterbegleitende Abstimmungen, Zwischenstände, Präsentationsanteile und Reflexionsanteile sind Bestandteil der Prüfungsleistung.

Prüfungsleistung

Studien- oder Projektarbeit. Sie wird als wissenschaftlicher Projektbericht beziehungsweise Paper mit Präsentation und Fachgespräch ausgestaltet. Der Projektbericht dokumentiert die Forschungs- oder Entwicklungsfrage, den Stand der Technik beziehungsweise Forschung, das methodische Vorgehen, die Umsetzung, die Evaluation sowie die kritische Reflexion der Ergebnisse. Er kann sich an einem wissenschaftlichen Paperformat orientieren, beispielsweise mit einem Umfang von ca. 6 bis 8 Seiten. Die Präsentation dient der fachlichen Vorstellung und Diskussion der Projektergebnisse. Das Fachgespräch dient insbesondere der individuellen Einordnung und Reflexion der erbrachten Leistung,

des methodischen Vorgehens, der Evaluation und der wissenschaftlichen Aussagekraft der Ergebnisse.

Literatur

- Die Literatur ist abhängig vom jeweiligen Projektthema. Relevante Fachliteratur, wissenschaftliche Veröffentlichungen, technische Dokumentationen, Standards und gegebenenfalls Normen werden im Rahmen der Projektbearbeitung recherchiert und durch die betreuenden Lehrenden ergänzt.
- Grundlagenliteratur zum wissenschaftlichen Arbeiten und Schreiben in der Informatik
- Grundlagenliteratur zu Forschungsmethoden der Informatik und angrenzender Disziplinen
- Projektspezifische Literatur aus Bereichen wie Software Engineering, Datenanalyse, Künstliche Intelligenz, Mensch-Computer-Interaktion, IT-Sicherheit, biomedizinische Informatik, Biosignalverarbeitung oder medizinische Bildanalyse
- Literatur und Leitlinien zu guter wissenschaftlicher Praxis sowie zum verantwortungsvollen Einsatz von KI-Werkzeugen

Kurzporträt MIN401: Forschungs- und Entwicklungsprojekt

Worum geht es? Das Forschungs- und Entwicklungsprojekt behandelt eine offene, aktuelle Fragestellung der Informatik oder eines angrenzenden Anwendungsfelds. Die Studierenden recherchieren den Stand der Technik oder Forschung, planen ein methodisches Vorgehen, entwickeln oder evaluieren Artefakte und dokumentieren Ergebnisse wissenschaftsnah.
Wofür braucht man es? Die Kompetenzen werden gebraucht, um neuartige oder unklare Problemstellungen methodisch belastbar zu bearbeiten. Sie sind wichtig für Masterarbeiten, innovationsnahe Entwicklung, technische Evaluation und wissenschaftlich fundierte Entscheidungsprozesse.
Wieso ist es interessant? Interessant ist das Modul, weil es Raum für eigene Fragestellungen und echte Erkenntnisarbeit bietet. Nicht die Anwendung eines bekannten Rezepts steht im Vordergrund, sondern das begründete Vorgehen bei offenen Aufgaben.

Abschluss

Der Abschlussbereich bündelt die im Masterstudium erworbenen fachlichen, methodischen und überfachlichen Kompetenzen in einer eigenständigen wissenschaftlichen oder wissenschaftlich fundierten anwendungsorientierten Arbeit. Die Masterarbeit mit Kolloquium zeigt, dass die Studierenden eine anspruchsvolle informatische Fragestellung selbstständig bearbeiten, dokumentieren und fachlich vertreten können.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Bereichs eine komplexe informatische Forschungs-, Entwicklungs- oder Evaluationsfrage eigenständig bearbeiten, den Stand von Wissenschaft und Technik einordnen, ein geeignetes Vorgehen entwickeln, Ergebnisse kritisch reflektieren und im Kolloquium fachlich verteidigen.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Abschlussbereich
  - eine anspruchsvolle Problemstellung analysieren, eingrenzen und in eine bearbeitbare Fragestellung überführen,
  - relevante wissenschaftliche und technische Quellen recherchieren, strukturieren und bewerten,
  - geeignete Methoden, Modelle, Verfahren oder Werkzeuge auswählen, anwenden und begründen,
  - Ergebnisse systematisch auswerten und ihren eigenständigen Beitrag herausarbeiten,
  - Grenzen, Validität, Unsicherheiten und relevante gesellschaftliche, ethische, rechtliche oder sicherheitsbezogene Aspekte reflektieren,
  - Vorgehen und Ergebnisse schriftlich dokumentieren, präsentieren und in einer fachlichen Diskussion verteidigen.
- **WOZU**: Damit sie zeigen, dass sie komplexe informatische Aufgaben auf Masterniveau selbstständig, methodisch fundiert und verantwortlich bearbeiten können. Der Abschluss bereitet auf anspruchsvolle berufliche Tätigkeiten, forschungsnahe Entwicklung und gegebenenfalls wissenschaftliche Weiterqualifikation vor.

Kurzporträt Abschluss

Worum geht es? Im Abschlussbereich bearbeiten die Studierenden eine größere informatische Fragestellung eigenständig und über einen längeren Zeitraum. Die Masterarbeit verbindet Recherche, methodisches Vorgehen, technische oder analytische Arbeit, Auswertung, Reflexion und wissenschaftliche Darstellung.
Wofür braucht man es? Die Masterarbeit zeigt, dass Studierende komplexe Aufgaben selbstständig strukturieren und fachlich tragfähige Ergebnisse erarbeiten können. Das ist wichtig für anspruchsvolle Entwicklungs-, Analyse-, Architektur-, Forschungs- und Leitungsaufgaben.
Wieso ist es interessant? Der Abschluss bietet die Möglichkeit, ein eigenes Thema vertieft zu verfolgen und die im Studium aufgebauten Kompetenzen sichtbar zusammenzuführen. Im Kolloquium wird die Arbeit nicht nur präsentiert, sondern auch fachlich eingeordnet und verteidigt.

MIN501: Masterarbeit mit Kolloquium

Modulverantwortung	Studiendekan:in des Fachbereichs	**Lehrperson(en)**	Alle Professor:innen des Studiengangs
Verwendbarkeit	Master Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	30 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	Keine angegeben.	**Engl. Titel**	Master's Thesis with Colloquium
Workload	Gesamtworkload: 900 Stunden		

Empfohlene Voraussetzungen

Die formalen Voraussetzungen für die Zulassung zur Masterarbeit mit Kolloquium ergeben sich aus der Prüfungsordnung. Empfohlen wird, das Modul nach Abschluss der übrigen Module aufzunehmen, sodass die im Studium erworbenen fachlichen, methodischen und überfachlichen Kompetenzen in einer eigenständigen wissenschaftlichen Arbeit zusammengeführt werden können.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine anspruchsvolle wissenschaftliche oder wissenschaftlich fundierte anwendungsorientierte Fragestellung der Informatik eigenständig und methodisch nachvollziehbar zu bearbeiten und dabei einen kritisch reflektierten, eigenständigen Erkenntnis-, Entwicklungs- oder Transferbeitrag zu leisten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine komplexe informatische Problemstellung analysieren, fachlich eingrenzen und daraus eine präzise Forschungs-, Entwicklungs- oder Evaluationsfrage ableiten,
- den relevanten Forschungs- und Entwicklungsstand beziehungsweise den Stand von Wissenschaft und Technik systematisch recherchieren, strukturieren und kritisch bewerten,
- Anforderungen, Rahmenbedingungen, Annahmen und Bewertungskriterien bestimmen und daraus ein geeignetes Untersuchungs-, Entwicklungs- oder Evaluationsdesign ableiten,
- wissenschaftliche und informatische Methoden, Modelle, Verfahren und Werkzeuge begründet auswählen, kombinieren, anpassen und gegebenenfalls weiterentwickeln,
- einen anspruchsvollen Lösungsansatz konzipieren, umsetzen, untersuchen oder evaluieren,
- Daten und Ergebnisse systematisch auswerten, im Verhältnis zur Fragestellung und zum bestehenden Wissensstand interpretieren sowie den eigenständigen Beitrag der Arbeit herausarbeiten,
- methodische Entscheidungen, Validität, Unsicherheiten, Übertragbarkeit und Grenzen sowie relevante gesellschaftliche, ethische, rechtliche, ökologische, wirtschaftliche und sicherheitsbezogene Aspekte kritisch reflektieren,
- Vorgehensweise, Ergebnisse und Schlussfolgerungen wissenschaftlich dokumentieren sowie im Kolloquium fachlich einordnen, präsentieren und begründet verteidigen.

WOZU. Die Studierenden nutzen diese Kompetenz, um neuartige und komplexe informatische Forschungs- und Entwicklungsaufgaben in wissenschaftsnahen und anspruchsvollen beruflichen Tätigkeitsfeldern selbstständig, methodisch fundiert und verantwortungsvoll zu lösen.

Inhalte

- Präzisierung und fachlich-wissenschaftliche Einordnung einer komplexen informatischen Problemstellung sowie Formulierung einer Forschungs-, Entwicklungs- oder Evaluationsfrage
- Systematische Recherche, Strukturierung und kritische Bewertung des Forschungs- und Entwicklungsstands beziehungsweise des Stands von Wissenschaft und Technik
- Analyse von Anforderungen, Rahmenbedingungen, Annahmen und Bewertungskriterien
- Konzeption eines geeigneten Untersuchungs-, Entwicklungs- oder Evaluationsdesigns
- Begründete Auswahl, Anpassung und gegebenenfalls Weiterentwicklung wissenschaftlicher und informatischer Methoden
- Konzeption, Umsetzung, Untersuchung oder Evaluation eines anspruchsvollen Lösungsansatzes
- Systematische Auswertung und Interpretation der Ergebnisse, Herausarbeitung des eigenständigen Beitrags sowie kritische Reflexion von Validität, Übertragbarkeit und Grenzen
- Wissenschaftliche Ausarbeitung sowie Präsentation, fachliche Einordnung und Verteidigung der Arbeit im Kolloquium

Prüfungsvorleistung

Keine angegeben. #### Prüfungsleistung Masterarbeit mit Kolloquium als zusammenhängende und gemeinsam bewertete Prüfungsleistung, davon 27 ECTS für die Masterarbeit und 3 ECTS für das Kolloquium. Die Masterarbeit ist eine wissenschaftliche schriftliche Ausarbeitung zu einer eigenständig bearbeiteten anspruchsvollen Fragestellung der Informatik. Sie umfasst die systematische und kritische Auseinandersetzung mit dem relevanten Forschungs- und Entwicklungsstand beziehungsweise dem Stand von Wissenschaft und Technik, ein methodisch fundiertes Vorgehen, einen eigenständigen Erkenntnis-, Entwicklungs- oder Transferbeitrag und die kritische Reflexion der Ergebnisse. Das Kolloquium umfasst die mündliche Präsentation, fachliche Einordnung und wissenschaftliche Verteidigung der Fragestellung, des Vorgehens, der Ergebnisse und des eigenständigen Beitrags der Masterarbeit.

Literatur

- Aktuelle fachlich einschlägige wissenschaftliche Veröffentlichungen und Monografien
- Aufgabenbezogene Forschungsberichte, Standards, Spezifikationen und technische Dokumentationen
- Literatur zu den für die jeweilige Fragestellung relevanten Forschungs-, Entwicklungs- und Evaluationsmethoden

- Die konkrete Literatur richtet sich nach der Aufgabenstellung und wird in Abstimmung mit der betreuenden Lehrperson festgelegt; Auswahl, wissenschaftliche Qualität und Relevanz der verwendeten Quellen sind von den Studierenden eigenständig zu beurteilen.

Kurzporträt MIN501: Masterarbeit mit Kolloquium

Worum geht es? In der Masterarbeit bearbeiten die Studierenden selbstständig eine anspruchsvolle Fragestellung der Informatik. Sie ordnen den Stand von Wissenschaft und Technik ein, wählen ein geeignetes Vorgehen, entwickeln oder untersuchen eine Lösung und verteidigen ihre Ergebnisse im Kolloquium.
Wofür braucht man es? Die Masterarbeit zeigt, dass komplexe informatische Aufgaben eigenständig, methodisch fundiert und verantwortungsvoll bearbeitet werden können. Sie ist ein wichtiger Nachweis für anspruchsvolle Entwicklungs-, Analyse-, Architektur-, Forschungs- oder Leitungstätigkeiten.
Wieso ist es interessant? Interessant ist die Masterarbeit, weil sie fachliche Vertiefung und persönliche Schwerpunktsetzung verbindet. Studierende können ein Thema über längere Zeit eigenständig verfolgen und ihren Beitrag sichtbar in einen größeren fachlichen Kontext stellen.

Glossar

Dieses Glossar erklärt Abkürzungen und wiederkehrende Begriffe, die in den Modulbeschreibungen verwendet werden.

ECTS European Credit Transfer System. Ein ECTS-Punkt (auch „Credit") entspricht etwa 30 Stunden studentischem Arbeitsaufwand (Workload) und ist die europaweit vergleichbare Maßeinheit für den Umfang eines Moduls.

SWS Semesterwochenstunden. Gibt an, wie viele Zeitstunden Lehrveranstaltung pro Woche während der Vorlesungszeit eines Semesters stattfinden, aufgeschlüsselt nach Veranstaltungsart (siehe V, Ü, P, S/SL).

V Vorlesung. Lehrveranstaltungsform, in der fachliche Inhalte überwiegend im Vortrag vermittelt werden.

Ü Übung. Lehrveranstaltungsform zur Vertiefung und Einübung der in der Vorlesung vermittelten Inhalte, häufig anhand von Aufgaben.

P Praktikum. Praktische Lehrveranstaltung, in der Inhalte im Labor oder an Versuchsaufbauten selbst durchgeführt werden.

S Seminar. Lehrveranstaltungsform mit stärker eigenständiger, oft projekt- oder präsentationsorientierter Arbeitsweise.

SL Seminaristischer Unterricht. Lehrveranstaltungsform, die das freie Vortragen von Inhalten (wie in einer Vorlesung) mit dem interaktiven Dialog kombiniert.

Workload Der geschätzte studentische Gesamtarbeitsaufwand für ein Modul, aufgeteilt in Präsenzstudium (Teilnahme an Lehrveranstaltungen) und Selbststudium (eigenständige Vor- und Nachbereitung, Prüfungsvorbereitung).

Modultyp Kennzeichnet, ob ein Modul verpflichtend zu belegen ist (**Pflicht**) oder aus einem Angebot ausgewählt werden kann (**Wahlpflicht**).

WAS / WOMIT / WOZU Format zur Beschreibung von Lernergebnissen: **WAS** die Studierenden nach dem Modul können, **WOMIT** (mit welchen Inhalten und Tätigkeiten) sie dieses Lernergebnis erreichen und **WOZU** sie diese Kompetenzen im weiteren Studium oder Beruf benötigen.

Ziele-Module-Matrix Übersicht, die zeigt, in welchen Modulen die übergreifenden Qualifikationsziele eines Studiengangs aufgebaut und vertieft werden.

Capstone-Modul Abschließendes Modul eines Lernpfads, in dem die zuvor erworbenen fachlichen und methodischen Kompetenzen in einer größeren, integrierenden Projektaufgabe zusammengeführt und angewendet werden.

Lernpfad Thematisch gebündelte Abfolge von Pflichtmodulen, die inhaltlich aufeinander aufbauen und in einem Capstone-Modul münden. Lernpfade orientieren sich an einem typischen Berufs- bzw. Tätigkeitsprofil des Studiengangs.

Kurzporträt Kurzbeschreibung eines Moduls oder Lernpfads entlang der Fragen „Worum geht es¿', „Wofür braucht man es?" und „Wieso ist es interessant?", gedacht als schneller Einstieg für Studierende.