Otto-Friedrich Universität Bamberg

Module Handbook

Module Handbook International Software System Science

Faculty of Information Systems and Applied Computer Sciences

According to the current version of the study and examination regulations of 2025 for the Master's degree programme International Software Systems Science at the Otto Friedrich University of Bamberg. In effect starting summer semester 2026.

Valid: 12.03.2026

Hinweis zur Weitergeltung älterer Fassungen eines Modulhandbuchs:

1. Geltungsbeginn

Die im vorliegenden Modulhandbuch enthaltenen Modulbeschreibungen gelten erstmals für das Semester, das auf dem Deckblatt angegeben ist.

2. Übergangsbestimmung

a. Studierende, die gemäß bisher geltendem Modulhandbuch ein Modul bereits in Teilen absolviert haben (vgl. Nr. 2b), schließen das Modul nach der bisher geltenden Fassung des Modulhandbuchs ab.

   Diese Übergangsbestimmung gilt ausschließlich für den dem versäumten/nicht bestandenen/nicht absolvierten regulären Prüfungstermin unmittelbar folgenden Prüfungstermin. Auf Antrag der oder des Studierenden kann der Prüfungsausschuss in begründeten Fällen eine Verlängerung der Übergangsfrist festlegen.

b. Ein Modul ist in Teilen absolviert, wenn die Modulprüfung nicht bestanden oder versäumt wurde. Gleiches gilt für den Fall, dass zumindest eine Modulteilprüfung bestanden, nicht bestanden oder versäumt wurde.

   Ferner gilt ein Modul als in Teilen absolviert, sofern sich die oder der Studierende gemäß bisher geltendem Modulhandbuch zu einer dem jeweiligen Modul zugeordneten Lehrveranstaltung angemeldet hat.

3. Geltungsdauer

Das Modulhandbuch gilt bis zur Bekanntgabe eines geänderten Modulhandbuchs auch für nachfolgende Semester.

Modules

AISE-Auto: Automation of First- and Higher-Order Logic....7
AISE-UL: Universal Logic & Universal Reasoning....9
AlgoK-Algo-M: Algorithms....12
COMNET-CN-M: Computer Networks....14
COMNET-NES-M: Networked Embedded Systems....16
DS-ConvAI-M: Advanced Dialogue Systems and Conversational AI....18
DSG-DistrSys-M: Distributed Systems....21
DT-CPP-M: Advanced Systems Programming in C++ (Master)....23
DT-DBCPU-M: Database Systems for modern CPU....25
EESYS-ADAML-M: Applied Data Analytics and Machine Learning in R....27
EESYS-ES-M: Energy Efficient Systems....30
ESE-ESEng-M: Evidence-based Software Engineering....33
GdI-FPRS-M: Functional Programming of Reactive Systems....35
GdI-IFP-M: Introduction to Functional Programming....38
HCI-US-B: Ubiquitous Systems....40
ISPL-MDP-M: Managing Digital Platforms....43
ISoSySc-Proj-M: Master's Project in Software System Science....45
ISoSySc-Sem-M: Master's Seminar in Software System Science....47
MOBI-ADM-M: Advanced Data Management....49
MOBI-DSC-M: Data Streams and Complex Event Processing....51
NLPROC-DL4NLP-M: Deep Learning for Natural Language Processing....53
NLProc-EA-M: Emotion Analysis....55
NLProc-ILT-M: Impact of Language Technology....58
NLProc-PGM4NLP-M: Probabilistic Graphical Models for Natural Language Processing....60
PSI-AdvaSP-M: Advanced Security and Privacy....62
PSI-DiffPriv-M: Introduction to Differential Privacy....65
SNA-OSN-M: Project Online Social Networks....67
SSS-PraktIntKon-M: Internship in an International Context....69
SSS-Thesis-M: Master's Thesis in Software Systems Science....70

SWT-ASV-M: Applied Software Verification..... 72

SYSNAP-OSE-M: Operating Systems Engineering.....75

SYSNAP-PMAP-M: Processor Microarchitecture and Performance.....78

SYSNAP-Virt-M: Virtualization.....80

VIS-IVVA-M: Advanced Information Visualization and Visual Analytics..... 83

xAI-DL-M: Deep Learning..... 85

Index by areas of study

1) A1 Software System Science (Module Group) ECTS: 30 - 69

a) S1: Theory (Specialisation Area) ECTS: 6 - 30

AISE-Auto: Automation of First- and Higher-Order Logic (6 ECTS, every summer semester)..... 7

AISE-UL: Universal Logic & Universal Reasoning (6 ECTS, every winter semester)..... 9

AlgoK-Algo-M: Algorithms (6 ECTS, every summer semester)..... 12

GdI-FPRS-M: Functional Programming of Reactive Systems (6 ECTS, every summer semester)..... 35

GdI-IFP-M: Introduction to Functional Programming (6 ECTS, every winter semester)..... 38

b) S2: Systems (Specialisation Area) ECTS: 6 - 30

Deviating from the study and subject examination regulations, the examination format for the module **SYSNAP-OSE-M** will be “written or oral examination” starting in the summer semester 2026.

COMNET-CN-M: Computer Networks (6 ECTS, every summer semester)..... 14

COMNET-NES-M: Networked Embedded Systems (6 ECTS, every winter semester)..... 16

DSG-DistrSys-M: Distributed Systems (6 ECTS, every summer semester)..... 21

DT-CPP-M: Advanced Systems Programming in C++ (Master) (6 ECTS, every winter semester)..... 23

MOBI-DSC-M: Data Streams and Complex Event Processing (6 ECTS, every winter semester)..... 51

SYSNAP-OSE-M: Operating Systems Engineering (6 ECTS, every summer semester)..... 75

SYSNAP-PMAP-M: Processor Microarchitecture and Performance (6 ECTS, every summer semester)..... 78

SYSNAP-Virt-M: Virtualization (6 ECTS, every winter semester)..... 80

c) S3: Software (Specialisation Area) ECTS: 6 - 30

DT-DBCPU-M: Database Systems for modern CPU (6 ECTS, every summer semester)..... 25

ESE-ESEng-M: Evidence-based Software Engineering (6 ECTS, every winter semester)..... 33

MOBI-ADM-M: Advanced Data Management (6 ECTS, every summer semester)..... 49

PSI-AdvaSP-M: Advanced Security and Privacy (6 ECTS, every summer semester)..... 62

PSI-DiffPriv-M: Introduction to Differential Privacy (6 ECTS, every winter semester)..... 65

SWT-ASV-M: Applied Software Verification (6 ECTS, every summer semester)..... 72

2) A2 Domain-specific Software System Science (Module Group) ECTS: 0 - 39

DS-ConvAI-M: Advanced Dialogue Systems and Conversational AI (6 ECTS, every summer semester)....18

EESYS-ADAML-M: Applied Data Analytics and Machine Learning in R (6 ECTS, every winter semester)....27

EESYS-ES-M: Energy Efficient Systems (6 ECTS, every summer semester)....30

HCI-US-B: Ubiquitous Systems (6 ECTS, every winter semester)....40

ISPL-MDP-M: Managing Digital Platforms (6 ECTS, every summer semester)....43

NLPROC-DL4NLP-M: Deep Learning for Natural Language Processing (6 ECTS, every summer semester)....53

NLProc-EA-M: Emotion Analysis (6 ECTS, every semester)....55

NLProc-ILT-M: Impact of Language Technology (6 ECTS, every winter semester)....58

NLProc-PGM4NLP-M: Probabilistic Graphical Models for Natural Language Processing (6 ECTS, every winter semester)....60

SNA-OSN-M: Project Online Social Networks (6 ECTS, every winter semester)....67

VIS-IVVA-M: Advanced Information Visualization and Visual Analytics (6 ECTS, every winter semester)....83

xAI-DL-M: Deep Learning (6 ECTS, every winter semester)....85

3) A3 Seminar and Project (Module Group) ECTS: 9

ISoSySc-Proj-M: Master's Project in Software System Science (6 ECTS, every semester)....45

ISoSySc-Sem-M: Master's Seminar in Software System Science (3 ECTS, every semester)....47

4) A4 Master's Thesis (Module Group) ECTS: 30

SSS-Thesis-M: Master's Thesis in Software Systems Science (30 ECTS, every semester)....70

5) A5 International Experience (Module Group) ECTS: 12 - 27

a) Guided Study Abroad (Elective Area) ECTS: 0 - 27

Modules amounting to 27 ECTS credits can be included in this area, which are completed as part of a guided study abroad program at a foreign university, provided that they differ significantly from the modules to be completed in accordance with these regulations and can be assigned to module groups A1, A2 or A3.

b) Internship in an International Context (Elective Area) ECTS: 0 - 12

SSS-PraktIntKon-M: Internship in an International Context (12 ECTS, every semester)....69

c) Foreign Languages (Elective Area) ECTS: 0 - 15

Module AISE-Auto Automation of First- and Higher-Order Logic *Automation of First- and Higher-Order Logic*	6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Dr. Christoph Benzmüller	
Contents: This course provides an introduction to the theory and practice of automatic theorem proving. Interest is in the automation of classical propositional logic, first level classical logic, and higher level classical logic. The exact emphasis may vary from year to year. This also applies to the proof calculi considered in each case (tableaux, resolution, etc.), as well as the concrete implementation methodology chosen for the practical exercises.	
Learning outcomes: The students will acquire competencies regarding the development of sound and complete proof calculi for classical logic, and the application of a uniform abstract proof technique (abstract consistency) for achieving completeness results. They also acquire competencies for implementing such proof calculi with modern functional and agent-oriented programming languages. In addition, the course will explore ideas regarding the integration of machine learning techniques in automated theorem systems.	
prerequisites for the module: none	
Recommended prior knowledge: First basic knowledge in logic and first programming skills are recommended, but not mandatory (and can be worked up in an additional tutorial/exercise group parallel to the course).	**Admission requirements:** none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Automation of First- and Higher-Order Logic **Mode of Delivery:** Lectures and Practicals **Language:** German **Frequency:** every summer semester	**6,00 Weekly Contact Hours**
Learning outcome: The students will acquire competencies regarding the development of sound and complete proof calculi for classical logic, and the application of a uniform abstract proof technique (abstract consistency) for achieving completeness results. They also acquire competencies for implementing such proof calculi with modern functional and agent-oriented programming languages. In addition, the course will explore ideas regarding the integration of machine learning techniques in automated theorem systems.	
Contents: This course provides an introduction to the theory and practice of automatic theorem proving. Interest is in the automation of classical propositional logic, first level classical logic, and higher level classical logic.	

The exact emphasis may vary from year to year. This also applies to the proof calculi considered in each case (tableaux, resolution, etc.), as well as the concrete implementation methodology chosen for the practical exercises.	

Examination Oral examination / Duration of Examination: 30 minutes	

Module AISE-UL Universal Logic & Universal Reasoning *Universelle Logik & Universelles Schließen*	6 ECTS / 180 h
(since WS25/26) Person responsible for module: Prof. Dr. Christoph Benzmüller	
Contents: Knowledge representation and reasoning applications in computer science, AI, philosophy and math typically employ very different logic formalisms. Instead of a "single logic that serves it all" (as envisioned already by Leibniz) an entire "logic zoo" has been developed, in particular, during the last century. Logics in this zoo, e.g., include modal logics, conditional logics, deontic logics, multi-valued logics, temporal logics, dynamic logics, hybrid logics, etc. In this lecture course we will introduce, discuss and apply a meta logical approach to universal logical reasoning that addresses this logical pluralism. The core message is this: While it might not be possible to come up with a universal object logic as envisioned by Leibniz, it might in fact be possible to have a universal meta logic in which we can semantically model, analyse and apply various species from the logic zoo. Classical higher order logic (HOL) appears particularly suited to serve as such a universal meta logic, and existing reasoning tools for HOL can fruitfully be reused and applied in this context.	
Learning outcomes: The participants of this course will, in combination with a hands-on introduction to Isabelle/HOL, learn about HOL, about semantical embeddings (SSE technique) of non-classical logics in HOL, and about proof automation of these logics in Isabelle/HOL. They will conduct practical exercises regarding the application of the SSE technique in philosophy, mathematics or artificial intelligence, including, normative reasoning and machine ethics.	
Remark: The main language of instruction in this course is English. The overall workload of 180h for this module consists of: • weekly classes: 22h • tutorials: 8h • Work on assignment: 90h • Literature study 40h • preparation for and time of the final exam: 20h	
prerequisites for the module: none	
Recommended prior knowledge: Basic knowledge about classical and non-classical logics, theoretical computer science.	**Admission requirements:** none

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester Semester

Module Units	
AISE-UL: Universal Logic & Universal Reasoning (Universelle Logik & Universelles Schließen) **Mode of Delivery:** Lectures and Practicals **Lecturers:** Prof. Dr. Christoph Benzmüller	**2,00 Weekly Contact Hours**

Language: English **Frequency:** every winter semester **Learning outcome:** The participants of this course will, in combination with a hands-on introduction to Isabelle/HOL, learn about HOL, about semantical embeddings (SSE technique) of non-classical logics in HOL, and about proof automation of these logics in Isabelle/HOL. They will conduct practical exercises regarding the application of the SSE technique in philosophy, mathematics or artificial intelligence, including, normative reasoning and machine ethics. **Contents:** Introduction to and discussion of tools and practical issues closely related to the topics discussed in the lecture as well as solutions of problems that come up during working on the practical assignment. **Literature:** will be announced in lecture course	
Examination Written examination, AISE-UL: Universal Logic & Universal Reasoning (Universelle Logik & Universelles Schließen) **Description:** Oral examination concerning the topics discussed in the lecture, exercises and assignment. Students may choose English or German as the language for the written assignment and oral examination. Examinations will take at the end of the summer term or at the beginning of the winter term (students may choose one of them). Students are assumed to work on an advanced modelling assignment ('schriftliche Hausarbeit') during the semester that is introduced at the beginning of the semester and uses the most important technologies (such as the See technique) discussed during the semester. **Note:** Without working on the modelling assignment over the term students may run into problems during their oral examination (Kolloquium) as we discuss questions concerning topics from the lectures as well as from the assignment; questions about the assignment are based on the assignment solution modelled by the students.	

Module Units	
AISE-UL: Universal Logic & Universal Reasoning (Universelle Logik & Universelles Schließen) **Mode of Delivery:** Practicals **Lecturers:** Prof. Dr. Christoph Benzmüller **Language:** English **Frequency:** every winter semester **Learning outcome:** The participants of this course will, in combination with a hands-on introduction to Isabelle/HOL, learn about HOL, about semantical embeddings (SSE technique) of non-classical logics in HOL, and about proof automation of these logics in Isabelle/HOL. They will conduct practical exercises regarding the application of	**2,00 Weekly Contact Hours**

the SSE technique in philosophy, mathematics or artificial intelligence, including, normative reasoning and machine ethics. **Contents:** Knowledge representation and reasoning applications in computer science, AI, philosophy and math typically employ very different logic formalisms. Instead of a "single logic that serves it all" (as envisioned already by Leibniz) an entire "logic zoo" has been developed, in particular, during the last century. Logics in this zoo, e.g., include modal logics, conditional logics, deontic logics, multi-valued logics, temporal logics, dynamic logics, hybrid logics, etc. In this lecture course we will introduce, discuss and apply a meta logical approach to universal logical reasoning that addresses this logical pluralism. The core message is this: While it might not be possible to come up with a universal object logic as envisioned by Leibniz, it might in fact be possible to have a universal meta logic in which we can semantically model, analyse and apply various species from the logic zoo. Classical higher order logic (HOL) appears particularly suited to serve as such a universal meta logic, and existing reasoning tools for HOL can fruitfully be reused and applied in this context. **Literature:** will be announced in lecture course	

Module AlgoK-Algo-M Algorithms *Algorithmen*	6 ECTS / 180 h
(since WS25/26) Person responsible for module: Prof. Dr. Isolde Adler	
Contents: Algorithms and algorithmic problem solving are at the heart of computer science. This module introduces students to the design and analysis of efficient algorithms. Students learn how to quantify the efficiency of an algorithm and what algorithmic solutions are efficient. Techniques for designing efficient algorithms are taught, including efficient data structures. We begin with standard methods such as Divide-and-Conquer and Dynamic Programming. We then move on to more advanced techniques and we discuss ways of dealing with computationally intractable problems and large data sets. This is done using illustrative and fundamental problems relevant to Computer Science and AI.	
Learning outcomes: On completion of the module student should be able to: - Demonstrate an understanding of what constitutes an efficient and an inefficient solution to a computational problem, - Analyse the efficiency of algorithms, - Evaluate and justify appropriate ways to provide efficient solutions for computational problems, - Identify and apply different design principles in the design of algorithms, - Describe efficient algorithms for a range of computational problems, along with their computational complexity, - Articulate the key concepts and critically evaluate approaches in a clear and rigorous manner, - Appreciate and understand in-depth the role of proofs in the area of algorithm design, - Recognise how the methods learned can be extended and used to solve other problems.	
Remark: The workload for this module is approxmately structured as follows: • Participation in lectures and tutorials: 45 hrs • Preparing and revising the lectures and tutorials: 60 hours • Solving the worksheets: 45 hrs • Exam preparation: 30 hrs	
prerequisites for the module: none	
Recommended prior knowledge: Prerequisites: Basic knowledge of algorithms and data structures, proof techniques, mathematical skills. Good English language skills.	**Admission requirements:** none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Algorithms **Mode of Delivery:** Lectures and Practicals	**4,00 Weekly Contact Hours**

Lecturers: Prof. Dr. Isolde Adler
Language: English/German

Contents:

The lectures introduce the topics, providing an in-depth explanation including motivation, intuition, examples and proofs, as well as tools, techniques and applications.

The tutorials consist of hands-on problem solving, including exam-style problems.

Literature:

- Jon Kleinberg and Éva. Tardos: Algorithm Design, Pearson/Addison-Wesley 2006.
- Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani: Algorithms, McGraw-Hill, 2006
- Anany Levitin, Design and analysis of algorithms, Pearson/Addison-Wesley 2007.
- Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullmanm, Data structures and algorithms, Addison-Wesley 1987
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Introduction to algorithms, 1st ed. MIT press and McGraw-Hill 1990 or 2nd ed. MIT press and McGraw-Hill 2001 or 3rd ed. MIT press and McGraw-Hill 2009.
- Kenneth H. Rosen: Discrete Mathematics and its Applications. McGraw-Hill, 2012.
- K. Houston: How to Think Like a Mathematician: A Companion to Undergraduate
  Mathematics. Cambridge University Press, 2009

Examination

No type selected

Description:

Oral exam (30 minutes) or written exam (90 minutes).

Depending on the number of participants, the exam will either be an oral exam or a written exam. The mode of examination will be communicated in the first lecture.

It is possible to contribute to your overall module grade by solving worksheets regularly and successfully, and by participating actively in the tutorials. However, it is also possible to achieve a "first" (1,0) by excelling in the exam.

Module COMNET-CN-M Computer Networks *Computer Networks*	6 ECTS / 180 h
(since SS26) Person responsible for module: Prof. Dr. Florian Klingler	
Contents: The Computer Networks lecture covers conceptual and technological foundations of computer networks and the Internet; particularly, layers 1–4 of the ISO/OSI model are addressed. In addition, approaches and tools for the quantitative analysis of communication protocols are discussed. The lecture is accompanied by a blackboard-based exercise sessions. • Physical layer: signal propagation, modulation, Shannon limits • Data link layer: ARQ, FEC, framing. Medium access control methods (Aloha, CSMA, CSMA/CD) • Network layer: routing as a graph problem and as a network problem; standard algorithms (Dijkstra, Bellman–Ford); routing vs. forwarding; case study IP (longest prefix matching, BGP, …) • Transport layer: congestion control, flow control, fairness, case study TCP • Description of services and protocols; quantitative analysis of communication protocols (e.g., Aloha, CSMA, TCP throughput) • Short Introduction to Wireless Networks	
Learning outcomes: Students of the course -- are able to identify the essential tasks involved in the design and construction of computer networks and describe fundamental architectural approaches; -- are able to list different solutions to a given problem, identify their advantages and disadvantages, and select an appropriate solution based on the requirements; -- are able to identify weaknesses in existing solutions and develop new communication protocols, as well as evaluate their performance.	
Remark: Students can select this couse only if they have not participated in COMNET-RN-B (Bachelor's evel course).	
prerequisites for the module: none	
Recommended prior knowledge: a good understanding of algorithms, Programming and Software, as well as basic knowledge of efficient algorithms; Good programming skills in C/C++ or Python, etc.; Knowledge of programming, programming languages, software engineering, database systems, modeling, data structures and algorithms, digital systems, and calculus for computer scientists.	**Admission requirements:** none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Computer Networks **Mode of Delivery:** Lectures and Practicals	**4,00 Weekly Contact Hours**

Language: English
Frequency: every summer semester

Learning outcome:
Students of the course
-- are able to identify the essential tasks involved in the design and construction of computer networks and describe fundamental architectural approaches;
-- are able to list different solutions to a given problem, identify their advantages and disadvantages, and select an appropriate solution based on the requirements;
-- are able to identify weaknesses in existing solutions and develop new communication protocols, as well as evaluate their performance.

Contents:
The Computer Networks lecture covers conceptual and technological foundations of computer networks and the Internet; particularly, layers 1–4 of the ISO/OSI model are addressed. In addition, approaches and tools for the quantitative analysis of communication protocols are discussed. The lecture is accompanied by a blackboard-based exercise sessions.

• Physical layer: signal propagation, modulation, Shannon limits
• Data link layer: ARQ, FEC, framing. Medium access control methods (Aloha, CSMA, CSMA/CD)
• Network layer: routing as a graph problem and as a network problem; standard algorithms (Dijkstra, Bellman–Ford); routing vs. forwarding; case study IP (longest prefix matching, BGP, …)
• Transport layer: congestion control, flow control, fairness, case study TCP
• Description of services and protocols; quantitative analysis of communication protocols (e.g., Aloha, CSMA, TCP throughput)
• Short Introduction to Wireless Networks

Examination
No type selected
Description:
Written exam (120 min) or oral exam (30 min) depending on number of participants. The exact format will be announced at the beginning of the event.

Module COMNET-NES-M Networked Embedded Systems *Networked Embedded Systems*	6 ECTS / 180 h
(since SS26) Person responsible for module: Prof. Dr. Florian Klingler	

Contents:

Contents of the course Networked Embedded Systems:
The objective of this course is gain insights into the operation and programming of embedded systems. A strong focus is on wireless sensor networks. We study the fundamentals of such sensor networks. In the scope of the exercises, we discuss selected topics in more detail.

- Design and architecture of embedded systems - Architecture of embedded systems, programming paradigms
- Sensor networks - Principles and applications
- Wireless communications - Concepts of modulation and encoding on the physical layer
- Wireless access - Typical medium access protocols for low-power sensor nodes
- Routing - Ad hoc routing and data centric communication
- Cooperation and clustering - Clustering algorithms, guaranteed connectivity

Learning outcomes:

The learning objective is to understand the fundamental concepts of networked embedded systems. Students understand these concepts and are able to apply this knowledge.

Non-cognitive Skills

- Commitment
- Learning competence

Remark:

Contact time: 60h
Self study: 120h

prerequisites for the module:

none

Recommended prior knowledge: • System software and system-level programming • C/C++ programming • Basic computer networks		**Admission requirements:** none
Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Networked Embedded Systems **Mode of Delivery:** Lectures and Practicals **Language:** English **Frequency:** every winter semester	**4,00 Weekly Contact Hours**
Learning outcome: siehe Modulbeschreibung	
Contents:	

siehe Modulbeschreibung	

Examination No type selected **Description:** Written exam (120 min) or Oral exam (30 min). The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.	

Module DS-ConvAI-M Advanced Dialogue Systems and Conversational AI *Advanced Dialogue Systems and Conversational AI*	6 ECTS / 180 h
(since SS25) Person responsible for module: Prof. Dr. Stefan Ultes	

Contents:

This module deals with state-of-the-art approaches to Conversational AI - text-based or speech-based dialogue interaction through language - and its modelling and realisation through machine learning and deep learning. Building upon content of the module DS-IDS-B, it dives into the technical realization of chatbots and spoken dialogue systems ranging from a modular pipeline architecture to end-to-end neural models including Large Language Models (LLMs). The module can be successfully completed without prior knowledge on dialogue systems.

Learning outcomes:

In this course, students will learn about various technological aspects of Conversational AI with a focus on state-of-the-art neural and deep learning approaches. After successfully completing this course, students will be able to

- know the inner workings of Large Language Models and how they can be applied to build dialogue systems
- compare state-of-the-art methods of Conversationa AI with each other based on the models' apabilties and limitations
- understand basic and advanced concepts of neural language modelling like Recurrent Neural Networks and Transformer models
- able to apply extensions of language models to build dialogue systems
- able to explain how neural language models can be used for building dialogue system
- able to explain lingusitic encoding strategies and their impact on down-stream computation
- understand theoretical foundations of Conversational AI and dialogue systems technology and modelling
- understand various technological aspects of Conversational AI with a focus on state-of-the-art neural and deep learning approaches to sequential and non-sequential supervised learning
- understand dialogue modelling through reinforcement learning and deep reinforcement learning and how to derive a suitable objective function
- understand how to make use of advanced deep learning architectures like recurrent neural networks and transformers for their application on various problems of dialogue systems and the dialogue system itself

The lecture is accompanied by practicals and assignments that will help participants to develop practical, hands-on experience. In those practicals, students will implement and evaluate different approaches for dialogue systems and its modules using machine learning algorithms using Python and its respective commonly used libraries. Thus, students gain the ability to:

- apply llms to conv ai-related tasks and to build dialogue systems
- apply different prompting strategies including RAG and how to evaulate them
- examin implementations of dialogue system modules and how to evaluate them

Remark:

The lecture is conducted in English. The workload of this module is expected to be roughly as follows:

- Lecture: 21h
- Preparation of lectures and analysis of further sources: 30h
- Practicals accompanying lecture: 21h
- Work on the actual assignments: 75h
- Preparation for exam: 30h

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
Good working knowledge of programming (e.g., in Python); Recommended (not mandatory) completion of modules: Einführung in Maschinelles Lernen/Introduction to Machine Learning (KogSys-ML-B), Einführung in die Dialogsysteme/Introduction to Dialogue Systems [DS-IDS-B], Deep Learning [xAI-DL-M]	none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** Semester

Module Units

1. Advanced Dialogue Systems and Conversational AI	**2,00 Weekly Contact Hours**

Mode of Delivery: Lectures
Lecturers: Prof. Dr. Stefan Ultes
Language: English/German
Frequency: every summer semester

Learning outcome:
see module description

Contents:
The lecture will be held in English. The following is a selection of topics that will be addressed in the course:

- Large Language Models and their application in Conversational AI
- End-to-end Neural Dialogue Generators
- Machine-larning based methods to various spoken dialogue system modules
- Statistical Spoken Dialogue Systems
- Evaluation techniques

Literature:
AI and NLP generally:

- Artificial Intelligence: A Modern Approach (Stuart Russell and Peter Norvig)
- Deep Learning (Ian Goodfellow and Yoshua Bengio)
- Speech and Language Processing (Dan Jurafsky and James Martin)

Covnersational AI and Dialogsysteme:

- Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots (Michael McTear)
- Transforming Conversational AI: Exploring the Power of Large Language Models in Interactive Conversational Agents (Michael McTear)
- Natural Language Generation (Ehud Reiter)

• Natural Language Processing with Transformers: Building Language Applications With Hugging Face (Lewis Tunstall) **Reinforcement Learning:** • Reinforcement Learning: An Introduction (Richard Sutton and Andrew Barto) • Grokking Deep Reinforcement Learning (Miguel Morales)	
2. Advanced Dialogue Systems and Conversational AI (Practicals) **Mode of Delivery:** Practicals **Lecturers:** N.N. **Language:** English/German **Frequency:** every summer semester **Learning outcome:** see module description **Contents:** Further exploration of concepts discussed in the lecture, often accompanied by assigments and programming exercises implemented in Python and the corresponding machine/deep learning libraries.	**2,00 Weekly Contact Hours**

Examination Oral examination / Duration of Examination: 30 minutes **Description:** Depending on the number of participants, the exam can also be a written exam (Klausur/E-Prüfung). The final decision will be announced within the first three weeks of the lecture period. The content that is relevant for the exam consists of the content presented both in the lectures and in the practicals (including the assignments).	

Module DSG-DistrSys-M Distributed Systems *Distributed Systems*	6 ECTS / 180 h 40 h Präsenzzeit 140 h Selbststudium

(since SS26)
Person responsible for module: Prof. Dr. Tobias Distler

Contents:
Nowadays infrastructure and business relies more or less on distributed systems of various flavors. Most of our civilization would not work any more if all distributed systems would fail. So, that should be a good reason for anyone planning to work in the context of IT to learn at least about the characteristics and basic issues of such systems. The course introduces to the different flavors of and issues with distributed systems, discusses the most basic problems arising with this kind of systems and presents solutions and techniques that are essential to make distributed systems work. Additionally, the course also teaches how to build simple distributed systems using Java-based technologies like process interaction, synchronization, remote message invocation and web service infrastructure. Students are required to work (in groups) on assignments in order to combine the theoretical concepts with practical experience and ... Yes, we program!

Learning outcomes:
Students know about the characteristics and different flavors of distributed systems and understand the essential differences compared to monolithic, centralized systems as well as their consequences when designing and building distributed systems. Students are able to apply the basic algorithmic techniques and programming paradigms in order to build simple distributed systems themselves. Students have gained basic experience with practically building and running distributed systems.

Remark:
The language of instruction in this course is English.

The overall workload of 180h for this module consists of:

- weekly classes: 22.5h
- tutorials: 22.5h
- Work on assignment: 75h
- Literature study 30h
- preparation for and time of the final exam: 30h

This course is intended for 2nd/3rd year bachelor students as well as master students which have not enrolled in a similar course during their bachelor studies. In case of questions don't hesitate to contact the person responsible for this module.

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
Knowledge of the basics of computer science in general, esp. operating systems, as well as practical experience in Java programming, as the subjects taught in DSG-EiAPS-B and DSG-EiRBS-B. Preferable also knowledge about multithreading and synchronization like, e.g., the subject-matters of DSG-PKS-B.	none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
1. Distributed Systems – Lecture **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Tobias Distler **Language:** English **Frequency:** every summer semester **Literature:** • Andrew S. Tanenbaum and Maarten van Steen. *Distributed Systems (3rd Edition)*, 2017. • Additional literature will be provided during the semester.	**2,00 Weekly Contact Hours**
2. Distributed Systems – Tutorial **Mode of Delivery:** Practicals **Lecturers:** Prof. Dr. Tobias Distler **Language:** English/German **Frequency:** every summer semester **Contents:** The tutorials provide further context on selected topics from the lecture, and offer advice as well as feedback on the associated programming assignment.	**2,00 Weekly Contact Hours**

Examination Coursework Assignment and Colloquium / Duration of Examination: 15 minutes Duration of Coursework: 3 months **Description:** Oral examination concerning the topics discussed in the lecture, exercises and assignment. **Students may choose English or German as the language for the oral examination.** Examinations will take place at the end of the summer term or at the begin of the winter term (students may choose one of them). Students are assumed to work on a programming assignment ('schriftliche Hausarbeit') during the semester that is introduced at the beginning of the semester and uses the most important technologies discussed during the semester. **Note:** Without working on the programming assignment over the term students may run into problems during their oral examination (Kolloquium) as we discuss questions concerning topics from the lectures as well as from the assignment; questions about the assignment are based on the assignment solution programmed by the students.	

Module DT-CPP-M Advanced Systems Programming in C++ (Master) *Fortgeschrittene Systemprogrammierung in C++ (Master)*	6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Dr. Maximilian Schüle	
Contents: In diesem Modul wird die fortgeschrittene Systemprogrammierung in C++ gelehrt. Dabei lernen die Teilnehmer nicht nur ihr Wissen in kleinen Programmierhausaufgaben anzuwenden sondern auch das gelernte Wissen in einer übergreifenden Projektarbeit zu kombinieren.	
Learning outcomes: Anwendung komplexer C++-Systemprogrammierung in eigenständiger Projektarbeit	
prerequisites for the module: none	

Recommended prior knowledge: none		**Admission requirements:** none
Frequency: every winter semester	**Recommended semester:** from 3.	**Minimal Duration of the Module:** 1 Semester

Module Units	
Fortgeschrittene Systemprogrammierung in C++ (Master) **Mode of Delivery:** Lectures and Practicals **Lecturers:** Prof. Dr. Maximilian Schüle **Language:** English **Frequency:** every winter semester	**4,00 Weekly Contact Hours**
Learning outcome: Anwendung komplexer C++-Systemprogrammierung in eigenständiger Projektarbeit	
Contents: In diesem Modul wird die fortgeschrittene Systemprogrammierung in C ++ gelehrt. Dabei lernen die Teilnehmer nicht nur ihr Wissen in kleinen Programmierhausaufgaben anzuwenden sondern auch das gelernte Wissen in einer übergreifenden Projektarbeit zu kombinieren.	
Literature: Primary • C++ Reference Documentation • Lippman, 2013. C++ Primer (5th edition). • Stroustrup, 2013. The C++ Programming Language (4th edition). • Meyers, 2015. Effective Modern C++. 42 Specific Ways to Improve Your Use of C++11 and C++14. Supplementary • Aho, Lam, Sethi & Ullman, 2007. Compilers. Principles, Techniques & Tools (2nd edition).	

• Tanenbaum, 2006. Structured Computer Organization (5th edition).	
Examination Portfolio / Duration of Coursework: 4 months	

Module DT-DBCPU-M Database Systems for modern CPU *Datenbanksysteme für moderne CPU*	6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Dr. Maximilian Schüle	
Contents: This lecture covers the implementation of database systems, including how to leverage modern hardware architectures, for example vector intrinsics (AVX-512) and CUDA programming for GPU. Diese Vorlesung behandelt die Implementierung von Datenbanksystemen, einschließlich der Nutzung moderner Hardware-Architekturen, z.B. Vektorinstruktionen (AVX-512) und CUDA-Programmierung für die GPU.	
Learning outcomes: Understand the concepts of database systems and be able to implement database systems, also for modern hardware Konzepte von Datenbanksystemen verstehen und Datenbanksysteme implementieren können inkl. für moderne Hardware	
prerequisites for the module: none	

Recommended prior knowledge: MOBI-DBS-B		**Admission requirements:** none
Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Datenbanksysteme für moderne CPU **Mode of Delivery:** Lectures and Practicals **Lecturers:** Prof. Dr. Maximilian Schüle **Language:** English **Frequency:** every summer semester	**4,00 Weekly Contact Hours**
Learning outcome: Understand the concepts of database systems and be able to implement database systems, also for modern hardware Konzepte von Datenbanksystemen verstehen und Datenbanksysteme implementieren können inkl. für moderne Hardware	
Contents: This lecture covers the implementation of database systems, including how to leverage modern hardware architectures, for example vector intrinsics (AVX-512) and CUDA programming for GPU. Diese Vorlesung behandelt die Implementierung von Datenbanksystemen, einschließlich der Nutzung moderner Hardware-Architekturen, z.B. Vektorinstruktionen (AVX-512) und CUDA-Programmierung für die GPU.	
Literature:	

• Theo Härder, Erhard Rahm.Datenbanksysteme: Konzepte und Techniken der Implementierung.Springer, Berlin; 2nd ed. • Hector Garcia-Molina, Jeff Ullman, Jennifer Widom. *Database Systems: The Complete Book* • D. E. Knuth.The Art of Computer Programming Volume III • Joseph M. Hellerstein, Michael Stonebraker, James Hamilton.Architecture of a Database System • Franz Faerber, Alfons Kemper, Per-Åke Larson, Justin J. Levandoski, Thomas Neumann, Andrew Pavlo.Main Memory Database Systems	

Examination Written examination / Duration of Examination: 90 minutes	

Module EESYS-ADAML-M Applied Data Analytics and Machine Learning in R *Applied Data Analytics and Machine Learning in R*	6 ECTS / 180 h

(since SS21)
Person responsible for module: Prof. Dr. Thorsten Staake

Contents:

This course provides the theoretical foundation and conveys hands-on skills in the fields of data analytics and machine learning using the statistics software GNU R. It uses real-word datasets from the realm of energy efficiency and consumer behavior and conveys the subject matter through real-world examples and practical challenges.

Following a refresher in descriptive statistic, the course covers

- an introduction to the statistics software GNU R,
- the design of field experiments and the use of Information Systems to collect behavioral data,
- techniques to formulate, solve, and interpret linear and logistic regression analyses,
- techniques to formulate, solve, and interpret clustering analyses,
- setting up, training, and evaluating machine learning algorithms, including KNN, regression, and support vector machines, and
- ethical issues and data privacy regulations.

Learning outcomes:

After a successful participation in this course, participants can

- translate new business and research questions that can be answered using empirical methods into suitable experimental designs,
- plan and conduct corresponding experiments,
- choose suitable methods from the set of methods presented in class to analyze the data,
- explain their design choices, the choice of methods, and the steps of the analyses,
- apply the methods correctly and efficiently using the statics software R,
- adjust the methods if needed to solve new and specific problems based on an understanding of the necessary theories,
- interpret the outcome of such analyses and identify the strengths and limitations of the approaches, and
- reflect upon data protection, privacy and ethical issues related to powerful techniques for data acquisition and analytics.

Remark:

The lecture will be held as a self-paced, video-based online lecture.

The tutorials take place once per week as in-classroom events.

The online lecture includes instructional videos (scripted, i.e., with subtitles), reading material, exemplary data sets, and a multitude of online and offline tasks. It also includes an online discussion forum.

The online lecture is supported by three classroom lectures (in addition to the classroom tutorials):

1. Classroom lecture: The introductory event includes a course overview and motivation. Moreover, credentials to access the online resources will be announced. Date: First week of the semester.

2. Classroom lecture: This intermediate session includes a review of the concepts covered so far. It should help participants to self-assess their learning progress. Date: Announced in the first week of the semester.
3. Classroom lecture: Exam preparation and Q&A. Date: Last week of the semester.

An introduction to the statistics software GNU R will be given as in-classroom event during the tutorials at the beginning of the semester.

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
This course requires a basic understanding of statistics (e.g., from a bachelor-level course). A statistics repetition and is part of the online material of the course and the of the first tutorials and should be complemented in self-study if necessary. Basic familiarity with a programming language.	none

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units

1. Lectures Data Analytics in Energy Informatics **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Thorsten Staake **Language:** German/English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Contents: The video-based online lecture is divided into two parts. Part 1 conveys the statistical basics required for the module, including, for example, properties of random distributions and descriptive and injunctive statistics. This part serves as refresher of bachelor-level statistics and thereby enables students with no statistics-knowledge beyond a basic introductory course to participate. Part 2 covers the methods outlined in “Module EESYS-DAE-M” subsection “Contents”. It includes both, the theory behind the concepts and their application using R. Both, Part 1 and Part 2 use datasets and examples from industry and research and provides many hands-on examples. In order to deepen the understanding and to ease the transfer of the methods to new problems and settings, mini-tasks and small exercises are part of the online lecture.	
Literature: Reading material will be announced in class.	

2. Practicals Data Analytics in Energy Informatics **Mode of Delivery:** Practicals **Language:** German/English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Contents: In the classroom tutorial, participants apply the methods, tools, and theories conveyed in the lecture to exemplary problems and to new challenges. This includes solving smaller tasks (e.g., acing case studies, working on concrete	

data problems) on paper and using the statistics software GNU R. Tasks are addressed individually or in small teams. The tutorials can also cover new content, especially when its immediate application supports the learning process. Selected tutorials contain a self-assessment of the learning progress. An introduction to GNU R is given in the first sessions.	

Examination Written examination / Duration of Examination: 90 minutes **Description:** The examination covers subject matter taught in the lectures and tutorials. The examination can also cover transfers of the subject matter to new problems and settings. Students can achieve up to 90 points. Through the voluntary completion of coursework (“bonus exercises”) during the semester, participants can collect up to 12 additional points that are counted towards the exam, given that the exam is passed also without points from bonus exercises. Bonus exercises can take the form of written assignments, presentations, or smaller software projects. Points from bonus exercises are only valid in the semester they have been earned in and in the immediately following semester. In the first week of the course, the publishing dates of bonus exercise tasks, the submission deadlines, and the points per bonus exercise will be announced. It is possible to pass the exam with a grade of 1.0 also without points from bonus exercises. Exam questions are stated in English, answers can be given in German or English.	

Module EESYS-ES-M Energy Efficient Systems *Energieeffiziente Systeme*		6 ECTS / 180 h
(since WS19/20) Person responsible for module: Prof. Dr. Thorsten Staake		
Contents: The course covers the design and application of Information Systems that help increase energy efficiency and reduce greenhouse gas emissions. It is directed to computer science and Information Systems students that want to apply their skills to challenges in the fields of energy, mobility, production, and sustainable consumption/consumer behavior. The course introduces methods and theories from behavioral economics, operations management, and simulation analysis that help to understand, analyze, and shape both, industry processes and consumer behavior in the field of sustainability. Also covered are cost/benefit considerations on a micro- and macro-level (including, for example, rebound effects) and a discussion on the economic and societal implications of the subject matter. The course includes an introduction to physics and energy engineering to allow students with very limited knowledge in these fields to participate successfully.		
Learning outcomes: Successful participants of this course shall acquire the skills to • explain the physical and technical principals covered in this course and apply them to new problems, • explain the components, influencing factors, requirements and challenges related to electric mobility and describe the contribution that Information Systems can make to solve the challenges; moreover, successful participants shall be able to set up data-based simulations to derive important characteristic variables related to electric vehicles, such as electric reachability, peak loads to electric grids, etc., • outline, assess, and conceptually model the potential of Information Systems and the effects to heating and room climate applications, • explain in detail the characteristics of and implications from environmental business Information Systems, • explain the discussed behavioral theories (e.g., the prospect theory), make use of them when building Information Systems that support decision making and behavioral change, and be able to evaluate the effectiveness of such systems, and • evaluate the effects of the tools and methods introduced, including their micro- and macro-economic effects, and critically assess the techniques used to perform such evaluations. Moreover, successful participants shall be able to apply the acquired skills to new challenges and adjust and extend them as needed. Finally, the participants shall realize the scope for design and the potential that results from their IT studies to favorably shape a sustainable and socially desirable development of our society.		
prerequisites for the module: none		
Recommended prior knowledge: none		**Admission requirements:** none
Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
1. Lectures Energy Efficient Systems **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Thorsten Staake **Language:** German/English **Frequency:** every summer semester **Contents:** The lecture covers the topics mentioned in “Module EESYS-ES-M”, subsection “Contents”. It uses traditional lecture elements, discussions, exercises, and group work to support participants in reaching the learning objectives. Special emphasis is placed on working on cases and on discussions of studies and scientific publications. Methods, tools, and theories are introduced with references to practical challenges and are applied to exemplary problems. For selected topics, the lecture relies on flipped classroom elements for which participants need to acquire knowledge in advance (e.g., through reading tasks), which is then critically reflected and extended in the classroom sessions. **Literature:** Weiterführende Unterlagen werden in der Veranstaltung bekanntgegeben.	**2,00 Weekly Contact Hours**
2. Practicals Energy Efficient Systems **Mode of Delivery:** Practicals **Language:** German/English **Frequency:** every summer semester **Contents:** The first tutorials convey basics in physics and electrical engineering in order to also allow students who did not take related modules to participate in this course. Subsequently, participants apply the methods, tools, and theories conveyed in the lecture to exemplary problems and to new challenges. Tutorials include small tasks, case studies, and reviews of scientific publications that are addressed individually or in small teams. The tutorials can also cover new content, especially when its immediate application supports the learning process. Selected tutorials contain a self-assessment of the learning progress. **Literature:** Reading material will be announced in class.	**2,00 Weekly Contact Hours**

Examination	
Written examination / Duration of Examination: 90 minutes **Description:** The examination covers subject matter taught in the lectures and tutorials. The examination can also cover transfers of the subject matter to new problems and settings. Students can achieve up to 90 points. Through the voluntary completion of coursework (“bonus exercises”) during the semester, participants can collect up to 12 additional points that are counted	

towards the exam, given that the exam is passed also without points from bonus exercises. Bonus exercises can take the form of written assignments, presentations, or smaller software projects. Points from bonus exercises are only valid in the semester they have been earned in and in the immediately following semester. In the first week of the course, the publishing dates of bonus exercise tasks, the submission deadlines, and the points per bonus exercise will be announced. It is possible to pass the exam with a grade of 1.0 also without points from bonus exercises. Exam questions are stated in English, answers can be given in German or English.	

Module ESE-ESEng-M Evidence-based Software Engineering *Evidence-based Software Engineering*	6 ECTS / 180 h
(since WS25/26) Person responsible for module: Prof. Dr. Matthias Galster	

Contents:
Software development is full of fashions, hype, and folklore, yet we often lack clear knowledge about what truly works—and under which conditions. Furthermore, developers frequently face difficult choices between alternatives, such as selecting the most suitable design, which practices to apply (e.g., whether to adopt pair programming in their team), or which technologies could improve their productivity (e.g., whether generative AI could support code reviews). Evidence-based software engineering, inspired by evidence-based medicine, seeks to strengthen professional decision-making by grounding it in the best available evidence. This course examines how to find or produce, interpret, critically appraise, report, and apply different forms of evidence in software engineering practice. It also presents practical techniques to address real-world software development challenges in a systematic and informed way.

Learning outcomes:
After completing this course, students will be able to:

- Understand the importance of evidence in software development (when and why does evidence matters) and the meaning of evidence for practitioners and different stakeholders in software development
- Select and apply evidence-based methods and techniques to create, synthesize and use evidence in a software engineering context (e.g., when selecting development practices, tools).
- Understand how to report evidence to different stakeholders
- Assess ethical considerations when dealing with evidence in software engineering

Remark:
The main language of instruction is English.

prerequisites for the module:
None

Recommended prior knowledge: The course assumes that students have knowledge of computer science and software engineering (e.g., analysis, design, implementation, programming, testing).	**Admission requirements:** None

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units

Evidence-based Software Engineering **Mode of Delivery:** Lectures and Practicals **Language:** English/German **Frequency:** every winter semester	**4,00 Weekly Contact Hours**
Learning outcome: See module description.	
Contents:	

See module description. Labs will complement lectures. Also, students may deliver short presentations. **Literature:** Literature will be announced in the lectures.	

Examination Internship report **Description:** The exam might be based on a mini-project including a report and presentations.	

Module GdI-FPRS-M Functional Programming of Reactive Systems *Functional Programming of Reactive Systems*	6 ECTS / 180 h
(since WS23/24) Person responsible for module: Prof. Ph.D. Michael Mendler	

Contents:

Based on an existing basic knowledge of functional programming (FP), the aim of this module is to develop advanced skills in the use of FP languages to structure and solve algorithmic problems in designing interactive and concurrent systems. We will study advanced programming abstractions specifically developed for the functional modelling of synchronous reactive systems. Following the methodological structure of the introductory course GDI-IFP, this advanced course, too, combines both practical programming with a focused discussion of pertinent underlying mathematical concepts. Though we use Haskell as our main language we may also look at other FP languages such as F#, ML or OCAML where appropriate.

Learning outcomes:

At the end of this course students should

- be familiar with advanced FP programming concepts and their application (e.g., class mechanism, type families, higher-rank polymorphism, monad and arrow abstractions, lenses, continuation-style programming, stream programming, concurrency abstractions)
- be able to use these advanced language concepts to solve complex algorithmic problems efficiently, in particular involving the use of memory, concurrency and interaction
- be able use the Haskell stack build tool and understand the mechanisms of package management
- appreciate the importance of functional abstraction for conciseness and efficiency of programming complex applications
- be familiar with the second-order polymorphic lambda calculus (Hindley-Milner predicative let-polymorphism, impredicative System F) as an operational semantics behind (eager, lazy) functional programming
- be able to explain the encoding of recursive data structures in type theory
- have an elementary understanding of the execution model of functional languages and transformation to operational code through defunctionalisation and abstract machines.
- by able to use FP (specifically Haskell) as a development tool for the design of new programming languages

Remark:

The workload for this module splits up roughly like this:

- participation in lectures and tutorials: 45 hrs
- preparation of classes and tutorials as well literature research: 60 hrs
- solving (ungraded) programming exercises and participation in lab sessions: 45 hrs
- exam preparation: 30 hrs

prerequisites for the module:

none

Recommended prior knowledge:	**Admission requirements:**
Elementary programming skills in a functional programming language, such as from module GdI-IFP-B; Basic knowledge in the use of	none

temporal and modal logic specification formalisms such as from GdI-MTL-B. English language skills at Level B2 (UniCert II) or above. Module Introduction to Functional Programming (GdI-IFP-M) - recommended		
Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units

1. Advanced Functional Programming — **2,00 Weekly Contact Hours**

Mode of Delivery: Lectures
Lecturers: Prof. Ph.D. Michael Mendler
Language: English/German
Frequency: every summer semester

Contents:
Through class presentations and direct interactions with the students the lecturer introduces the topics of the course in detail, poses exercises and suggests literature for self-study.

Literature:

- S. Marlow: The Haskell 2010 Language Report. https://www.haskell.org/onlinereport/haskell2010/
- V. Zsók, Z. Horváth, R. Plasmeijer: Central European Functional Programming School. Springer 2012.
- S. Marlow: Parallel and Concurrent Programming in Haskell: Techniques for Multicore and Multithreaded Programming, O'Reilly 2013.
- B. O'Sullivan, J. Goerzen, D. Stewart: Real World Haskell. O'Reilly 2009.
- Ch. Okasaki: Purely Functional Data Structures, CUP 1998
- F. Rabhi, G. Lapalme: Algorithms - A Functional Approach.
- D. Syme, A. Granicz, A. Cisternino: Expert F#4.0, Apress 2015.
- B. Pierce: Types and Programming Languages. MIT Press 2002. (esp. Chapters 23+25)
- H. Barendregt, W. Dekkers, R. Statman: Lambda Calculus with Types. CUP 2013.

2. Functional Programming of Reactive Systems — **2,00 Weekly Contact Hours**

Mode of Delivery: Practicals
Lecturers: Prof. Ph.D. Michael Mendler
Language: English/German
Frequency: every summer semester

Contents:
The tutorials deepen the students' understanding of the theoretical concepts and constructions covered in the lectures through practical exercises. Participants are given the opportunity to discuss their solutions to homework question sheets and sample solutions are presented by the tutors or lecturer for selected exercises. The tutorials also provide exam preparation.

Literature:

The literature will be announced in class. Here are some general pointers on FP languages and synchronous programming. • S. Marlow: The Haskell 2010 Language Report. https://www.haskell.org/onlinereport/haskell2010/ • V. Zsók, Z. Horváth, R. Plasmeijer: Central European Functional Programming School. Springer 2012. • S. Marlow: Parallel and Concurrent Programming in Haskell: Techniques for Multicore and Multithreaded Programming, O'Reilly 2013. • D. Syme, A. Granicz, A. Cisternino: Expert F#4.0, Apress 2015. • H. Barendregt, W. Dekkers, R. Statman: Lambda Calculus with Types. CUP 2013. • Benveniste, A. et al: The Synchronous Languages 12 years later. Proc. IEEE, Vol 91(1), January 2003. • Berry, G.: SCADE: Synchronous design and validation of embedded control software. In: Next Generation Design and Verification Methodologies for Distributed Embedded Control Systems. Proc. GM R&D Workshop, Bangalore, January 2007. pp. 19-33. • Potop-Butucaru et. al: The Synchronous Hypothesis and Synchronous Languages. In Richard Zurawski. *Embedded Systems Design and Verification*, CRC Press, pp.6-1-6-27, 2009.	
Examination Written examination / Duration of Examination: 90 minutes **Description:** The examination language is English. The form of examination is either oral (30 minutes) or written (90 minutes) depending on the number of participants. The form of examination will be determined at the beginning of the semester and announced in class.	

Examination Oral examination / Duration of Examination: 30 minutes **Description:** The examination language is English. The form of examination is either oral (30 minutes) or written (90 minutes) depending on the number of participants. The form of examination will be determined at the beginning of the semester and announced in class.	

Module GdI-IFP-M Introduction to Functional Programming *Introduction to Functional Programming*	6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Ph.D. Michael Mendler	
Contents: The aim of this module is to provide an introduction to functional programming using Haskell. This course develops both elementary practical programming skills and discusses the typed lambda calculus and its role as an operational semantics for functional programming, stressing the importance of types and type checking for static program analysis.	
Learning outcomes: At the end of this course students should be familiar with important language constructs of Haskell and their semantics (e.g., expressions, local declarations, higher-order function abstraction, recursion, lazy and eager evaluation, referential transparency, algebraic data types, monads); be able to use these language concepts to solve algorithmic problems; be familiar with the lambda calculus as an operational semantics behind functional programming; understand the difference between imperative and declarative programming styles; have an appreciation of the close relationship between programming language types and specification and the role of type checking as a static program analysis method; be familiar with polymorphic Hindley-Milner style type systems.	
Remark: The main language of instruction in this course is English. However, the lectures and/or tutorials may be delivered in German if all participating students are fluent in German.	
prerequisites for the module: none	
Recommended prior knowledge: Elementary concepts in logic and discrete mathematics for computer scientists; Basic programming skills; English language skills at Level B2 (UniCert II) or above.	**Admission requirements:** none

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
1. Introduction to Functional Programming **Mode of Delivery:** Lectures **Lecturers:** Prof. Ph.D. Michael Mendler **Language:** English/German **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Contents: Through prepared class presentations and direct interactions with the students the lecturer introduces the topics of the course in detail, poses exercises and suggests literature for self-study.	
Literature: • Pierce, B. C.: Types and Programming Languages, MIT Press, 2002	

• Thompson, S.: Haskell – The Craft of Functional Programming, Addison-Wesley 1999.	
2. Introduction to Functional Programming **Mode of Delivery:** Practicals **Lecturers:** Prof. Ph.D. Michael Mendler **Language:** English/German **Frequency:** every winter semester **Contents:** The tutorials deepen the students' understanding of the theoretical concepts and constructions covered in the lectures through practical exercises. Participants are given the opportunity to discuss their solutions to homework question sheets and sample solutions are presented by the tutors or lecturer for selected exercises. The tutorials also provide exam preparation.	**2,00 Weekly Contact Hours**
Examination Written examination / Duration of Examination: 90 minutes **Description:** 90 min written examination. The exam takes place during the regular exam period after the end of the semester.	

Module HCI-US-B Ubiquitous Systems *Ubiquitäre Systeme*	6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Dr. Tom Gross	
Contents: Theoretical, methodological, and practical foundation of Ubiquitous Computing	
Learning outcomes: The aim of this module is to teach advanced knowledge and skills in the aerea of ubiquitous systems as well as abroad theoretical and practical methodological expertise concerned with the design, conception and evaluation of ubiquitous systems. Students of this course learn the relevant literature and systems in breadth and depth and should be able to critical review new litarature and systems	
Remark: htp://www.uni-bamberg.de/hci/leistungen/studium The workload for this module is roughly structured as following: • Attendance of the lectures and assignments: 45 hours • Credits of the lecture (incl.research and study of additional sources): ca. 30 Hours • Credits of the assignments ((incl.research and study of additional sources, excluding optional homework assignment): ca. 30 hours • Solving the optional homework assignments: overall ca. 45 hours • Exam preparation: ca. 30 hours (based on the above mentioned preparation and revision of the subject material) The default language of instruction in this course is German, but can be changed to English on demand. All course materials (incl. exams) are available in English.	
prerequisites for the module: none	
Recommended prior knowledge: Module Algorithms and data structures (MI-AuD-B) Module Introduction to Algorithms, Programming and Software (DSG-EiAPS-B)	**Admission requirements:** Passing the written exam

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Ubiquitous Systems **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Tom Gross **Language:** German/English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Contents: This lecture gives an introduction to the subject of Ubiquitous Computing—that is, the paradigm of invisible computing, with computers embedded into everyday objects that act as client and server and communicate with each other—and includes the following conceptual, technical and methodological topics:	

- Basic concepts
- Base technology and infrastructures
- Ubiquitous systems and prototypes
- Context awareness
- User interaction
- Ubiquitous systems in a broad context and related topics

Literature:
The course is based on a compilation of different sources; as additional sources and as a reference are recommended:

- Krumm, J. (Ed.). Ubiquitous Computing Fundamentals. Taylor & Francis Group, Boca Raton, FL, 2010.

Examination
Oral examination
Description:
The oral exam takes 30 minutes and is worth a total of 90 points. Depending on the number of attendees the form of the exam can be changed to a written exam with 90 minutes and a total of 90 points. The final form of the exam is announced in the first lecture at the beginning of the term.

During the semester students can do assignments, which are optional. They are 12 points in total. The type of optional homework assignments as well as the deadlines are announced in detail at the beginning of the term. If the oral exam is passed (as a rule 50% of the points have to be reached) the points from the assignments are a bonus and added to the points from the oral exam. In any case, a top grade of 1,0 is also reachable without solving the assignments.

Module Units

Ubiquitous Systems — **2,00 Weekly Contact Hours**
Mode of Delivery: Practicals
Lecturers: Scientific Staff Mensch-Computer-Interaktion
Language: German/English
Frequency: every winter semester

Contents:
Practical assignments based on the subjects of the lecture including the programming of small prototypes

Literature:
Cf. lecture

Examination
Written examination / Duration of Examination: 90 minutes
Description:
In Abhängigkeit der Teilnehmerzahl wird die Modulprüfung entweder in Form einer Klausur oder in Form einer mündlichen Prüfung durchgeführt.

Die Festlegung erfolgt zu Semesterbeginn und wird im ersten Lehrveranstaltungstermin bekannt gegeben. In der Klausur über 90 min. können 90 Punkte erzielt werden. Es besteht die Möglichkeit, optionale Studienleistungen zu erbringen. Diese umfassen insgesamt 12 Punkte. Die Art der optionalen Studienleistungen sowie deren Bearbeitungsfrist werden zu Beginn der Lehrveranstaltung verbindlich bekannt gegeben. Ist die Prüfung bestanden (in der Regel sind hierzu 50 % der Punkte erforderlich), so werden die durch optionale Studienleistungen erreichten Punkte als Bonuspunkte angerechnet. Eine 1,0 ist in der Prüfung auf jeden Fall auch ohne Punkte aus der Bearbeitung optionaler Studienleistungen erreichbar.	

Module ISPL-MDP-M Managing Digital Platforms *Managing Digital Platforms*	6 ECTS / 180 h
(since SS25) Person responsible for module: Prof. Dr. Thomas Kude	

Contents:

PLEASE NOTE: This module will not be offered in the summer semester 2025 due to Prof. Kude´s research semester. The exam will take place.

Digital platforms are ubiquitous in industries and in society and both researchers and practitioners have recognized their disruptive potential. Large technology companies, such as Apple, Alibaba, Amazon, or SAP, rely on a platform business model and the emergence of the thriving platform economy has contributed to the meteoric rise of some platform owners to top the lists of the most valuable companies in the world. The central actors in the context of digital platforms include the platform owner that provides the platform itself along with interfaces and other resources, outside third-party actors that provide complementary products and services, as well as the users of the platform. For example, in the context of mobile app ecosystems, complementors can leverage platform functionality of iOS or Android to create apps and use Apple’s App Store or the Google Play Store to offer them to iPhone or Android users.

In this course, we develop a comprehensive understanding of the management of digital platforms through an in-depth exploration of the roles and mechanisms of digital platforms and the surrounding ecosystems. After laying the foundations of digital platform management, we will dive into advanced questions of platform design and management, e.g., related to platform launch, to governing third-party contributions, or to key success factors for the various actors in digital platform ecosystems. The course relies on both theoretical insights and practical cases across industries and companies.

We will work on central topics of managing platform ecosystems, including, but not limited to:

- Foundations of digital platforms
- Launching and monetizing digital platforms
- Digital platform governance
- The role of complementors in digital platforms

Learning outcomes:

After the course, participants will be able to...

- Recognize the growing importance of digital platforms
- Analyze the underlying mechanisms and the roles of different actors in digital platform ecosystems
- Make decisions regarding the governance of different types of platforms
- Develop strategies and business models for complementor organizations that benefit from and depend on digital platforms

Remark:

The required workload of 180h is approximately subdivided into:

- 56h for participation in lecture and exercise
- 124h for preparation and post-processing of sessions as well as exam preparation

prerequisites for the module:

none

Recommended prior knowledge:	**Admission requirements:**
Good command of the English language	none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Managing Digital Platforms **Mode of Delivery:** **Lecturers:** Prof. Dr. Thomas Kude **Language:** English **Frequency:** every summer semester **Literature:** The specific literature that we will use in the course will be communicated or distributed in class or through the learning platform (VC).	**4,00 Weekly Contact Hours**

Examination Written examination / Duration of Examination: 90 minutes **Description:** The exam questions will include the content from lecture, exercises, and assignments. Students can reach 90 points in the exam. Students may obtain additional points to improve their grade though the voluntary participation in group or individual assignments These points can be included in the exam points if a student would pass the exam without the additional points. The respective assignments, the available time, and the points that can be reached in each assignment will be communicated if and once such voluntary assignments are offered. The best grade (1,0) can be reached without participating in the voluntary assignments.	

Module ISoSySc-Proj-M Master's Project in Software System Science *Master Project in Software System Science*	6 ECTS / 180 h

(since WS25/26)
Person responsible for module: Prof. Ph.D. Michael Mendler
further responsible : (qua office the degree programme representative for the International Software Systems Science master's degree)

Contents:
This module covers the application of foundational methods in the area of Software Systems Science in the context of a practical project. The available topics are determined by the teaching and research unit that is offering the project.

Learning outcomes:
Building on the knowledge and skills obtained from the lectures and seminars of the graduate degree studies, the project tackles a scientific research question or an advanced software development problem with relevance for the area of software systems science that will typically be related to the current scientific research of the offering teaching unit. Depending on the project's objectives, the work may be pursued as an individual project or a group project. The project provides competencies for conducting independent academic research; problem solving skills exploiting the current state of the art in science and technology, competencies for goal-oriented self-organisation, time management and team work; competencies written and oral presentations following academic standards.

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
The recommended specific study prerequisites are determined and communicated by the teaching unit offering the module. Typically, there may be thematically pertinent modules from the respective teaching unit that should have been successfully attended before.	none

Frequency: every semester	**Recommended semester:** from 2.	**Minimal Duration of the Module:** 1 Semester

Module Units

Master's Project in Software System Science **Mode of Delivery:** **Language:** English **Frequency:** every semester	**4,00 Weekly Contact Hours**

Learning outcome:
As described for the module.

Contents:
The contents of the master's projects will be determined and communicated by the teaching unit offering the module.

Literature:
The reading list will be announced at the beginning of the project and communicated by the teaching unit offering the module.

Examination Coursework Assignment and Colloquium **prerequisites for module examination:** Regular participation in teaching classes **Description:** The assessment is based on a written homework and a colloquium. The work duration and deadline for the written deliverable as well as the duration of the colloquium will be determined by the supervisor of the project, at the beginning of the semester.	

Module ISoSySc-Sem-M Master's Seminar in Software System Science *Master Seminar in Software System Science*	3 ECTS / 90 h

(since SS25)
Person responsible for module: Prof. Ph.D. Michael Mendler
further responsible : (qua office the degree programme representative for the International Software Systems Science master’s degree)

Contents:
Independent academic research of a specific topic in the area of Software Systems Science and the structuring, analysis and presentation of the findings using scientific methods.

Learning outcomes:
Competencies for the systematic identification, critical and systematic analysis of the scientific literature pertinent to a given specific topic related to Software Systems Science; competencies for structuring of complex research issues in a delimited area with a critical assessment of the available competing approaches in the state of the art; advanced skills to communicate scientific results efficiently while applying academic standards; basic skills to generate own scientific texts; competencies to review as well as assess existing published academic work; further development of the student’s scientific curiosity supported by a self-confident, research-oriented attitude towards software systems science.

Remark:
The master’s seminar is chosen from among the offerings in one of the areas of Computer Science, Applied Computer Science or Information Systems. The available seminars can be identified in UnivIS with the key phrase "ISoSySc-Sem" and must be explicitly specified as suitable for master’s students.

prerequisites for the module:
none

Recommended prior knowledge: The recommended specific study prerequisites are determined and communicated by the teaching unit offering the module. Typically, there may be thematically pertinent modules from the respective teaching unit that should have been successfully attended before.		**Admission requirements:** Master's Seminar in Software System Science
Frequency: every semester	**Recommended semester:** from 3.	**Minimal Duration of the Module:** 1 Semester

Module Units

Master's Seminar in Software System Science **Mode of Delivery:** Seminar **Language:** English **Frequency:** every semester	**2,00 Weekly Contact Hours**

Contents:
The contents of the master’s projects will be determined and communicated by the teaching unit offering the module

Literature:
The reading list will be announced at the beginning of the project and communicated by the teaching unit offering the module.

Examination Coursework Assignment with presentation **prerequisites for module examination:** Regular participation in teaching classes **Description:** The assessment is based on a written homework and presentation. Alternatively, the assessment can be based on a written homework and a colloquium. The work duration and deadline for the written deliverable as well as the duration of the presentation or colloquium will be determined by the lecturer of the seminar, at the beginning of the semester.	

Module MOBI-ADM-M Advanced Data Management *Advanced Data Management*	6 ECTS / 180 h 45 h Präsenzzeit 135 h Selbststudium
(since SS21) Person responsible for module: Prof. Dr. Daniela Nicklas	
Contents: With the rapid growth of the internet and more and more observable processes, many data sets became so large that they cannot be processed with traditional database methods any more. This modul covers advanced data management and integration techniques (also known under the term “big data”) that are useful when dealing with very large data sets.	
Learning outcomes: The students will understand the challenges of big data, and will be able to apply some of the new techniques to deal with it.	
Remark: The main language of instruction in this course is English. However, the lectures and/or tutorials may be delivered in German if all participating students are fluent in German. The written reports/seminar essay and the presentation may be delivered in English or in German.	
prerequisites for the module: none	
Recommended prior knowledge: Foundations of relational databases, relational algebra and SQL; e.g. from Modul SEDA-DMS-B: Data management systems	**Admission requirements:** none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
1. Lectures Advanced Data Management **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Daniela Nicklas **Language:** English **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
Contents: The lecture will cover various algorithms for clustering, association rule mining, or page ranking and their scalable processing using map and reduce methods, data integration, data cleansing and entity recognition. The exercises will be built upon the Hadoop framework. The language of the course will be announced in the first lecture.	
Literature: L. Wiese, Advanced Data Management, For SQL, NoSQL, Cloud and Distributed Databases. Berlin, Boston: De Gruyter, 2015	
2. Practicals Advanced Data Management **Mode of Delivery:** Practicals **Lecturers:** Prof. Dr. Daniela Nicklas	**2,00 Weekly Contact Hours**

Language: English
Frequency: every summer semester

Contents:
see Lectures

The language of the course will be announced in the first lecture.

Examination
Written examination / Duration of Examination: 75 minutes
Description:
Central written exam. The examination language is English.

The exam questions will be in English. The questions can be answered in English or German. The content that is relevant for the exam consists of the content presented in the lecture and in the practical assignments.

The exam consists of 7 tasks of which only 6 will be graded. The exam time includes a reading time of 15 minutes to select the tasks to be completed within the scope of the choices.

Participants who submit solutions for practical assignments can achieve bonus points. Details regarding the number of assignments, the number of bonus points per assignment, the conversion factor from bonus points to exam points (e.g., 10:1) and the type of assignments will be announced in the first practical assignment session.

If the points achieved in the exam are sufficient to pass the exam on its own (generally, this is the case when at least 50% of the points have been obtained), the converted bonus points will be added to the points achieved in the exam.

The grade 1.0 can be achieved without the bonus points.

Module MOBI-DSC-M Data Streams and Complex Event Processing *Data Streams and Complex Event Processing*	6 ECTS / 180 h 45 h Präsenzzeit 135 h Selbststudium
(since WS25/26) Person responsible for module: Prof. Dr. Daniela Nicklas	
Contents: The management of data streams and foundations of event processing: Applications, systems, query languages, continuous query processing, and security in distributed data stream management systems. The modul covers the following topics: Architectures of data stream management systems; Query languages; Data stream processing; Complex event processing; Security in data stream management systems; Application of data stream management systems	
Learning outcomes: Understand the challenges of data stream management and complex event processing Recognize and link basic building blocks of data stream management tasks in different frameworks and systems Develop and program queries on data streams and event streams in different query languages to process data streams and detect event patterns Understand basic implementation techniques for data stream operators Understand the main security challenges and solutions in data stream management systems	
prerequisites for the module: none	
Recommended prior knowledge: Foundations of relational databases, relational algebra and SQL; e.g. from Modul MOBI-DBS-B: Database Systems	**Admission requirements:** none

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Data Streams and Complex Event Processing **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Daniela Nicklas **Language:** English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Learning outcome: Understand the challenges of data stream management and complex event processing Recognize and link basic building blocks of data stream management tasks in different frameworks and systems Develop and program queries on data streams and event streams in different query languages to process data streams and detect event patterns Understand basic implementation techniques for data stream operators	

Understand the main security challenges and solutions in data stream management systems **Contents:** The management of data streams and foundations of event processing: Applications, systems, query languages, continuous query processing, and security in distributed data stream management systems. The modul covers the following topics: Architectures of data stream management systems; Query languages; Data stream processing; Complex event processing; Security in data stream management systems; Application of data stream management systems	
Examination Oral examination / Duration of Examination: 15 minutes **Description:** oral or written exam (will be announced in class at the beginning of the semester). The examination language is English.	

Module Units	
Data Streams and Complex Event Processing **Mode of Delivery:** Practicals **Language:** English **Frequency:** every winter semester **Contents:** see lecture	**2,00 Weekly Contact Hours**

Examination Written examination / Duration of Examination: 60 minutes **Description:** oral or written exam (will be announced in class at the beginning of the semester). The examination language is English.	

Module NLPROC-DL4NLP-M Deep Learning for Natural Language Processing *Deep Learning for Natural Language Processing*	6 ECTS / 180 h
(since WS25/26) Person responsible for module: Prof. Dr. Roman Klinger further responsible : Dr. Sean Papay	
Contents: In this course, students will recep the fundamentals of machine learning, neural networks, and deep learning, before exploring in depth the applications of deep learning to natural language processing (NLP). We will discuss approaches and techniques such as: • Recurrent Neural networks / LSTMs • Transformers • Autoregressive Models • Sequence classification • Sequence labeling and their application to tasks such as: • Distributional semantics and embeddings • Named Entity Recognition • Relation extraction • Machine translation • Language modeling • Automatic speech recognition	
Learning outcomes: Students should strengthen their knowledge of deep learning models and techniques, and gain experience in applying these techniques to natural language processing. Students should gain practical experience in using python and libraries like pytorch in order to design, train, and apply deep neural networks to NLP tasks.	
prerequisites for the module: Master student	
Recommended prior knowledge: Prior knowledge of and experience with Machine Learning. Strong programming ability, preferably with python.	**Admission requirements:** none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Deep Learning for Natural Language Processing **Mode of Delivery:** Seminar **Lecturers:** Dr. Sean Papay **Language:** English **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
Learning outcome:	

Students should strengthen their knowledge of deep learning models and techniques, and gain experience in applying these techniques to natural language processing. Students should gain practical experience in using python and libraries like pytorch in order to design, train, and apply deep neural networks to NLP tasks.

Contents:

In this course, students will recep the fundamentals of machine learning, neural networks, and deep learning, before exploring in depth the applications of deep learning to natural language processing (NLP). We will discuss approaches and techniques such as:

- Recurrent Neural networks / LSTMs
- Transformers
- Autoregressive Models
- Sequence classification
- Sequence labeling

and their application to tasks such as:

- Distributional semantics and embeddings
- Named Entity Recognition
- Relation extraction
- Machine translation
- Language modeling
- Automatic speech recognition

Literature:

Deep Learning by Ian Goodfellow and Yoshua Bengio and Aaron Courville

Examination

Written examination / Duration of Examination: 60 minutes

Module NLProc-EA-M Emotion Analysis *Emotion Analysis*	6 ECTS / 180 h

(since WS25/26)
Person responsible for module: Prof. Dr. Roman Klinger

Contents:
This module discusses the fundaments of emotion theories in psychology and how they can be used for computational modeling, such that computers can interpret emotions in text.

We discuss psychological emotion theories, including appraisal theories, which explain how events are evaluated regarding their emotional connotation. We discuss how lexical resources can be created that contain emotion words and their common emotion interpretation. We look into machine learning methods to estimate models for emotion analysis and structured analysis, including role labeling.

The course is taught as a 2 SWS lecture with assignments that the students present, accompanied by a practical project that students work on independently under supervision.

Learning outcomes:
The students obtained a good understanding of the psychological background on emotions as it is required to develop emotion analysis systems. Based on the use case of emotion analysis, they obtained a deeper insight in how natural language processing systems are created, including resource creation, model training, evaluation, and interpretation.

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
Having attended at least one of the following courses (ideally all of them, as the emotion analysis course describes a concrete application of the methods taught in these lectures): Information Extraction and Text Mining Natural Language Processing Deep Learning for Natural Language Processing	none

Frequency: every semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Emotion Analysis **Mode of Delivery:** Lectures and Practicals **Language:** English/German **Frequency:** every summer semester	**4,00 Weekly Contact Hours**
Learning outcome:	

The students obtained a good understanding of the psychological background on emotions as it is required to develop emotion analysis systems. Based on the use case of emotion analysis, they obtained a deeper insight in how natural language processing systems are created, including resource creation, model training, evaluation, and interpretation.

Contents:

The emotion analysis course serves two purposes: (1) it communicates knowledge about emotion theories and emotion modeling in natural language processing. (2) students learn on one special topic how to develop NLP systems. The course consists of the following lectures:

1. Emotion theories

2. Text corpus creation and annotation

3. Dictionary-based methods for emotion analysis

4. Machine and deep learning based methods

5. Rules to infer emotion labels via cognitive evaluation of events

6. Emotion intensity regression

7. Text segmentation for emotion stimulus detection

In four assignments, student learn to (a) develop annotated corpora, (b) train classification models on annotated data, (c) conduct literature review, and (d) familiarize themselves with emotion role labeling, develop a corpus or a model.

Literature:

There is no dedicated book on the topic, but the following material covers parts of what we discuss in the lecture:

Bing Liu: Sentiment Analysis. 2020.

https://www.cambridge.org/core/books/sentiment-analysis/sentiment-analysis/0AB94D9F40A0402AB6EA566CB5776D30

Jurafsky/Martin: Speech and Language Processing. 2025.

https://web.stanford.edu/~jurafsky/slp3/

Feldman Barrett/Lewis/Haviland-Jones: Handbook of Emotions. 2018. https://www.guilford.com/books/Handbook-of-Emotions/Barrett-Lewis-Haviland-Jones/9781462536368/contents	

Examination Coursework Assignment and Colloquium / Duration of Examination: 1 hours **Description:** In the emotion analysis course, students are required to submit the following: * Slides for 4 assignments, and present a subset of them * A colloquium discussion of 15 minutes about the content of the course (after the term ended) * A report on an own small project (of approximately 3 ECTS), together with a short report.	

Module NLProc-ILT-M Impact of Language Technology *Impact of Language Technology*	6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Dr. Roman Klinger	
Contents: Topics include • Value Sensitive Design, • Bias and Discrimination, • Intersectionality, • Emergent Bias in Translation Technologies, • Content Moderation and Toxicity Detection, • Documentation and Transparency, • Science Communictation, • Privacy	
Learning outcomes: This course aims to deepen our understanding of the ethical issues associated with deploying NLP technology. We will explore how to identify those likely to be affected by the technology (both direct and indirect stakeholders), assess the risks involved, and design systems that better align with stakeholder values. Through discussions of readings from the expanding body of research on fairness, accountability, transparency, and ethics in NLP and related fields, as well as value-sensitive design, we will address the following questions: - What specific harms can arise from the use of NLP systems? - How can we fix, prevent, or mitigate these harms? - What responsibilities do we have as NLP researchers and developers in this context?	
prerequisites for the module: none	

Recommended prior knowledge: Required is an understanding of machine learning techniques and mechanisms, knowledge of NLP is a plus but not required		**Admission requirements:** none
Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Societal Impact of Language Technology **Mode of Delivery:** Lectures and Practicals **Language:** English **Frequency:** every winter semester	**4,00 Weekly Contact Hours**
Learning outcome: This course aims to deepen our understanding of the ethical issues associated with deploying NLP technology. We will explore how to identify those likely to be affected by the technology (both direct and indirect stakeholders), assess the risks involved, and design systems that better align with stakeholder values.	

Through discussions of readings from the expanding body of research on fairness, accountability, transparency, and ethics in NLP and related fields, as well as value-sensitive design, we will address the following questions: - What specific harms can arise from the use of NLP systems? - How can we fix, prevent, or mitigate these harms? - What responsibilities do we have as NLP researchers and developers in this context?	
Contents: Topics include • Value Sensitive Design, • Bias and Discrimination, • Intersectionality, • Emergent Bias in Translation Technologies, • Content Moderation and Toxicity Detection, • Documentation and Transparency, • Science Communictation, • Privacy	
Literature: A variety of different sources from current research are used. Among them: Birhane, A. 2021. The Impossibility of Automating Ambiguity. Artificial Life, 27(1):44-61. Lewis, J.E. et al, 2020. Indigenous Protocol and Artificial Intelligence Friedman, B. (1996). Value-sensitive design. ACM Interactions, 3 (6), 17-23. Leidner, J. L., & Plachouras, V. (2017). Ethical by design: Ethics best practices for natural language processing. In Proceedings of the first ACL workshop on ethics in natural language processing (pp. 30-40). Valencia, Spain: Association for Computational Linguistics	
Examination Written examination / Duration of Examination: 60 minutes	

Module NLProc-PGM4NLP-M Probabilistic Graphical Models for Natural Language Processing *Probabilistic Graphical Models for Natural Language Processing*	6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Dr. Roman Klinger	

Contents:

The course will provide an introduction to probabilistic graphical models, through the lens of natural language processing. Some topics covered will include

- Directed graphical models / Bayesian networks
- Undirected graphical models / Markov random fields
- Conditional random fields
- Causal modeling
- Structured prediction with graphical models
- Inference and sampling from graphical models
- Training methods for graphical models
- Neural graphical models.

Learning outcomes:

The goal of this course is to provide an introduction to probabilistic graphical models, and their use in natural language processing. We will start with formalisms for directed and undirected graphical models, before branching out into more specific applications for specific task domains in natural language processing. Students should leave with a basic understanding of how probabilistic graphical models work, and how they can be applied to tasks within natural language processing.

prerequisites for the module:

none

Recommended prior knowledge:	**Admission requirements:**
Students should have prior experience with probability theory, statistics, or machine learning. Prior experience of natural language processing might be helpful, but is not required.	none

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units

Probabilistic Graphical Models for Natural Language Processing **Mode of Delivery:** Lectures and Practicals **Language:** English **Frequency:** every winter semester	**4,00 Weekly Contact Hours**

Learning outcome:

The goal of this course is to provide an introduction to probabilistic graphical models, and their use in natural language processing. We will start with formalisms for directed and undirected graphical models, before branching out into more specific applications for specific task domains in natural language processing. Students should leave with a basic understanding of how probabilistic graphical models work, and how they can be applied to tasks within natural language processing.

Contents:

The course will provide an introduction to probabilistic graphical models, through the lens of natural language processing. Some topics covered will include

- Directed graphical models / Bayesian networks
- Undirected graphical models / Markov random fields
- Conditional random fields
- Causal modeling
- Structured prediction with graphical models
- Inference and sampling from graphical models
- Training methods for graphical models
- Neural graphical models.

Literature:

Klinger, R., & Tomanek, K. (2007). *Classical probabilistic models and conditional random fields*. TU, Algorithm Engineering.

Lafferty, J., McCallum, A., & Pereira, F. (2001, June). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Icml (Vol. 1, No. 2, p. 3).

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Examination

Written examination / Duration of Examination: 60 minutes

Module PSI-AdvaSP-M Advanced Security and Privacy *Advanced Security and Privacy*	6 ECTS / 180 h 45 h Präsenzzeit 135 h Selbststudium

(since SS25)
Person responsible for module: Prof. Dr. Dominik Herrmann

Contents:
Information security and privacy are relevant in almost all information systems today. Many real-world use cases have complex security and privacy requirements involving multiple parties. Often there are multiple stakeholders with different, sometimes even contradictory interests. For instance, some use cases call for a solution that allows a service provider to process sensitive data without learning its content. In other cases it is not the content but some meta information such as location and usage intensity that has to be protected. And then there are scenarios where seemingly harmless pieces of data can be used to disclose or infer very personal pieces of information about an individual.

This module covers advanced techniques for information security and privacy that can be used to satisfy the complex requirements of practical systems. It builds upon the basic concepts in information security that are introduced in the module "Introduction to Security and Privacy" (PSI-IntroSP-B).

Learning outcomes:
This module is designed to bring students towards the research boundaries in the field of security and privacy technologies by covering a selection of contemporary topics in depth. The focus of the module is on technical safeguards that can be used by system designers and users to enforce properties such as confidentiality and integrity. Moreover, sophisticated attacks on security and privacy are explained.

Successful students will be able to explain attack strategies and defenses discussed in recent research papers. They will also be able to analyze whether a particular attack or defense is relevant in a specific scenario. Finally, they will be able to implement selected attacks and defenses with a programming language of their choice.

Remark:
This module is taught in English. It consists of a lecture and tutorials. During the course of the tutorials there will be theoretical and practical assignments (task sheets). Assignments and exam questions can be answered in English or German.

Lecture and tutorials are partially taught in form of a paper reading class. Participants are expected to read the provided literature in advance and participate in the discussions.

Workload breakdown:

- Lecture: 22.5 hours (2 hours per week)
- Tutorials: 22.5 hours (2 hours per week)
- Preparation and studying during the semester: 30 hours
- Assignments: 67.5 hours
- Preparation for the exam (including the exam itself): 37.5 hours

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
Participants should be familiar with basic concepts in information security and privacy, which can be acquired, for instance, by taking the module "Introduction to Security and Privacy" (PSI-IntroSP-B).	none

This includes basic knowledge about the commonly used security terminology, common types of malware and attacks, buffer overflows and related attacks, cryptography, network security, web security, and concepts of privacy. Moreover, participants should have practical experience with at least one scripting or programming language such as Python or Java. Module Introduction to Security and Privacy (PSI-IntroSP-B) - recommended		
Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
1. Advanced Security and Privacy **Mode of Delivery:** Lectures **Language:** English/German **Frequency:** every summer semester **Learning outcome:** cf. module description **Contents:** Selected topics: • Authentication techniques • Privacy on the web (e.g., online tracking) • Privacy enhancing technologies (e.g., Tor) • Security and privacy aspects of e-mail • Usability aspects in security and privacy • Ethical aspects information security • Advanced techniques in software security (e.g., symbolic execution) • Advanced cryptographic building blocks • Other current topics in privacy and security Some parts of the lecture are aligned with current events and recently published research. The selected topics are therefore subject to change. **Literature:** Selected books: • R. Anderson: Security Engineering • A. Shostack: Threat Modelling • J.-P. Aumasson: Serious Cryptography • W. Stallings: Computer Security: Principles and Practice • B. Schneier et al.: Cryptography Engineering • J. Erickson: Hacking: The Art of Exploitation • J. Katz & Y. Lindell: Introduction to Modern Cryptography • L. Cranor & S. Garfinkel: Security and Usability	**2,00 Weekly Contact Hours**
2. Tutorials for Advanced Security and Privacy **Mode of Delivery:** Practicals **Language:** English/German	**2,00 Weekly Contact Hours**

Frequency: every summer semester	

Examination Written examination / Duration of Examination: 110 minutes **Description:** The exam time includes a reading time of 20 minutes. The content that is relevant for the exam consists of the content presented in the lecture and tutorials (including the assignments) as well as the content of the discussed papers. The maximum number of points that can be achieved in the exam is 100. Participants that solve assignments can collect bonus points. Details regarding the total number of bonus points, the number of assignments, the number of points per assignment, and the type of assignments will be announced in the first lecture. If the points achieved in the exam are sufficient to pass the exam on its own (generally, this is the case when at least 50 points have been obtained), the bonus points will be added to the points achieved in the exam. The grade 1.0 can be achieved without the bonus points.	

Module PSI-DiffPriv-M Introduction to Differential Privacy *Introduction to Differential Privacy*	6 ECTS / 180 h
(since WS23/24) Person responsible for module: Prof. Dr. Dominik Herrmann further responsible : Graf, Christian Alexander	

Contents:

The protection of personal data is an organizational as well as a technical challenge. Privacy-by-design is a reasonable requirement that is anything but easy to implement. This is especially true if a system deals with data that is meant to be published.What is more, a mathematically meaningful definition of privacy has only been available for less than a decade.

The lecture addresses different concepts and approaches for de-identification and attacks on privacy of published datasets. Its focus is on bringing you an in-depth understanding of differential privacy. Theoretical foundations, concepts and examples of state-of-the-art algorithms are introduced and explored in greater depth by means of practical exercises.

Contents:

1. Fundamental concepts of Data Privacy (8h)

- Outline of topic and its impact on society and economy
- A short history of data privacy
- Privacy by design and privacy frameworks
- Attacker models and attack patterns
- Different approaches to define privacy and their downsides
- Motivation and conceptual idea of Differential Privacy

2. Mathematical Foundations (20h)

- a review of important concepts from analysis, stochastic and statistics
- properties of important distributions, e.g. Gauss-, Exponential- and Laplace-distribution
- some useful theorems

3. An overview over common methods used in statistical disclosure control (10h)

- common methods used for de-identification and approaches to define privacy in depth
- common methods used for disclosure risk estimation and determination of data utility

4. Algorithmic foundations of Differential Privacy (16h)

- generalized data base models
- randomized algorithms
- mathematical definition and properties of differential privacy
- measuring privacy-loss and utility
- post processing immunity of dp-methods
- alternative dp definitions

5. Different approaches to achieve Differential Privacy (10h)

For instance:

- DIP (distribution invariant differential privacy)

- GAN-approaches
- Existing Software frameworks for de-identification

Learning outcomes:

- understand and apply de-identification approaches and attacks on privacy
- understand and apply fundamental stochastic and statistical methods used in statistical disclosure control
- understand the mathematical concepts of differential privacy following Dwork et. al.
- apply examples for dp-algorithms in example scenarios
- know different approaches towards differential privacy

prerequisites for the module:
none

Recommended prior knowledge:		**Admission requirements:**
none		none
Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** Semester

Module Units

Lecture and Tutorial — **4,00 Weekly Contact Hours**

Mode of Delivery: Lectures and Practicals
Language: English
Frequency: every winter semester

Contents:
see module description

Literature:
Provisional recommended literature:

- Claire McKay Bowen: Protecting Your Privacy in a Data-Driven World
- Dwork, Roth: The Algorithmic Foundations of Differential Privacy, Foundations and Trends in
- Theoretical Computer Science Vol. 9, Nos. 3–4 (2013)
- SPECIAL ISSUE: A New Generation of Statisticians Tackles Data Privacy, CHANCE Magazine, 33:4, 2020.

Literature on probability theory and statistics:

- Ludwig Fahrmeier, Rita Künstler, Iris Pigeot, Gerhard Tutz, Statistik - Der Weg zur Datenanalyse, 8. Auflage, Springer, 2016.
- William Feller, An Introduction to Probability Theory and Its Applications Vol.I, 3.Auflage, John Wiley & Sons, 1968.
- David J.C. MacKay: Information Theory, Inference, and Learning Algorithms., Cambridge University Press, 2003.

Examination
/ Duration of Examination: 90 minutes

Module SNA-OSN-M Project Online Social Networks *Projekt zu Online Social Networks*	6 ECTS / 180 h

(since SS23)
Person responsible for module: Prof. Dr. Oliver Posegga

Contents:
This module is an introduction to the analysis of online social networks. The aim is twofold: to provide students with the tools necessary to undertake research into online networks, and to give an overview of the type of questions these data can answer.

Learning outcomes:
At the conclusion of the course, students should know not only how to calculate basic network metrics on pre-existing data sets, but also how to capture an online social network efficiently with the intent of answering a specific research question.

Further goals:

- Learn how the radical innovation process in small teams works
- Learn how to collaborate in multidisciplinary intercultural virtual teams
- Learn how to find trendsetter and trends on the Internet and social media
- Learn how to predict trends using SNA und statistical forecasting techniques

Remark:
The main language of instruction in this course is English. The written reports/seminar essay and the presentation have to be delivered in English.

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
We recommend attending at least one of the following courses: • Social Network Analysis (SNA-ASN-M) • Theories of Social Networks (SNA-NET-M)	keine

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units

Online Social Networks **Mode of Delivery:** Practicals **Lecturers:** Prof. Dr. Oliver Posegga **Language:** English/German **Frequency:** every winter semester	**4,00 Weekly Contact Hours**

Contents:
The course will define online networks, examine how they differ from offline social networks, and consider theoretical and methodological issues associated with their analysis. The sessions will explore different strategies to retrieve and analyze online network data, and present different empirical scenarios to which those tools have been applied.

Literature:

• Gloor, P. A. Swarm Creativity, Competitive Advantage Through Collaborative Innovation Networks. Oxford University Press, 2006 Further literature will be announced in the lecture.	

Examination Coursework Assignment and Colloquium / Duration of Examination: 30 minutes Duration of Coursework: 4 months **prerequisites for module examination:** Regelmäßige Teilnahme an der Lehrveranstaltung **Description:** Die Gewichtung der Prüfungsleistungen Hausarbeit und Kolloquium wird zu Beginn der Lehrveranstaltung von der Dozentin bzw. dem Dozenten bekannt gegeben.	

Module SSS-PraktIntKon-M Internship in an International Context *Praktikum im internationalen Kontext*	12 ECTS / 360 h
(since WS24/25) Person responsible for module: Prof. Ph.D. Michael Mendler	
Contents: As an internship in an international context, a subject-specific internship geared to the professional field of Software Systems Science must be proven, which must be completed in an international context, preferably abroad. The internship can be completed in a foreign or internationally operating domestic company (or research institution) in private or public hands. An internship placement must be chosen in such a way that it meets the training objectives of § 39 Para. 1.	
Learning outcomes: • Gain work experience in an international context, for international students specifically in the German labour market • Transfer and application of the (theoretical) knowledge learned at the university in the industrial practice • Reflection on one's own strengths and weaknesses by taking responsibility for small projects, to boost confidence in one’s abilities, to improve social skills • To learn to communicate constructively in a team, to create technical solutions in a partially specified context, under time and resource constraints • Networking with potential employers	
Remark: Proof of the internship must be provided in the form of an internship certificate from the organizational unit where the internship was completed and a written internship report. The internship certificate and the internship report must be submitted together to the module manager.	
prerequisites for the module: none	
Recommended prior knowledge: It is strongly recommended that you only start an internship once you have earned at least 30 ECTS credits in module groups A1 and A2.	**Admission requirements:** none

Frequency: every semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Examination Praktikumsbericht, unbenotet **Description:** The report must contain at least 4 pages of continuous text, not including the cover page.	

Module SSS-Thesis-M Master's Thesis in Software Systems Science *Master Thesis in Software Systems Science*	30 ECTS / 900 h
(since SS23) Person responsible for module: Prof. Ph.D. Michael Mendler further responsible : Professors of Computer Science	
Contents: The module for the master's thesis comprises 30 ECTS credit points and is assessed through a written exam in the form of a master's thesis document and an oral exam conducted as a colloquium. The topic of the master's thesis must be taken from one of the research areas specified in Appendix 2a of the study an examination regulations. Topics outside of these areas may also be admitted on request but must be individually approved by the examination board. For such an exception it must be plausibly justified that the chosen topic is related to the curriculum of the master's degree programme in International Software Systems Science.	
Learning outcomes: Through the successful completion of the master's thesis the examinee • demonstrates that they are able to conduct independent research; • produce technical solutions to a research problem of substantial size, • arising and identified from the current state of the art and • critically evaluate the contributions made. on the basis of the specific knowledge acquired during their degree studies.	
prerequisites for the module: The master's thesis cannot be registered and thus confirmed by the examination board until at least 60 ECTS credit points have been successfully completed towards the degree.	
Recommended prior knowledge: It is assumed that candidates are familiar with academic research and have the necessary skills for independent literature research and technical writing such as acquired through a bachelor thesis.	**Admission requirements:** none

Frequency: every semester	**Recommended semester:** 4.	**Minimal Duration of the Module:** 1 Semester

Examination Coursework Assignment / Duration of Coursework: 6 months **Description:** The marks obtained from the written work is weighted 67% of the total grade for the master's thesis module.	

Examination Colloquium **Description:** The examination includes a presentation (Kolloquium) of a duration between 20 and 60 minutes. The purpose of the presentation is for the student to defend their	

main results of the thesis. The thesis will be weighted with 67%, the presentation with 33%. The presentation will take place before or after the grading of the thesis, according to the student's preference.	

Module SWT-ASV-M Applied Software Verification *Applied Software Verification*	6 ECTS / 180 h

(since SS26)
Person responsible for module: Prof. Dr. Gerald Lüttgen

Contents:
The sharp increase in the complexity of software systems has made the limitations of traditional code review and software testing for software quality assurance evident. Today's software industry is thus introducing more formal techniques to software testing and verification such as concolic testing, deductive formal verification and software model checking, all of which make heavy use of modern SMT solvers.

This module teaches theoretical and practical concepts of automated software verification. It introduces students to the logic and algorithmic foundations of SMT solvers, the automated reasoning about software and, in particular, software model checking. In addition, modern software verification tools are presented and employed to tackle practical verification problems.

Learning outcomes:
On completion of this module, students will be able to apply modern software verification tools and understand their underlying principles, techniques and algorithms.

Remark:
The main language of instruction is English. The lectures and practicals may be delivered in German if all participating students are fluent in German.

The total workload of 180 hrs. is split approximately as follows:

- 30 hrs. attending the lectures
- 30 hrs. attending the practicals
- 60 hrs. preparing and reviewing the lectures and practicals, including researching literature, studying material from additional sources and applying software tools
- 30 hrs. working on the assignment
- 30 hrs. preparing for the colloquium

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
Basic knowledge in algorithms and data structures, mathematical logic and theoretical computer science.	none

Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:**
		1 Semester

Module Units

1. Applied Software Verification **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Gerald Lüttgen **Language:** English/German **Frequency:** every summer semester	**2,00 Weekly Contact Hours**

Contents:
The lectures introduce students to (i) the ideas and concepts behind formal techniques in software testing and verification, including concolic testing,

deductive verification and contract-based reasoning; (ii) the workings of SMT solvers on the basis of SAT solving and decision procedures; (iii) the modelling of software as finite automata by applying predicate abstraction and abstraction-refinement techniques; (iv) the modelling and automatic checking of linear-time properties using Büchi automata and state space exploration; (v) the widely deployed technique of bounded model checking.

Literature:

- Baier, C., Katoen, J.-P. Principles of Model Checking. MIT Press, 2008.
- Biere, A., Heule, M., Van Maaren, H., Walsh, T. Handbook of Satifiability, 2nd ed. IOS Press, 2021.
- Clarke, E., Grumberg, O., Kroening, D., Peled, D. and Veith, H. Model Checking. 2nd. ed. MIT Press, 2018.
- Huth, M. and Ryan, M. Logic in Computer Science: Modelling and Reasoning about Systems. 2nd ed. Cambridge University Press, 2004.
- Kroening, D. and Strichman, O. Decision Procedures: An Algorithmic Point of View, 2nd. ed. Springer, 2018.
- Sen, K., Marinov, D. and Agha, G. CUTE: A Concolic Unit Testing Engine for C. In European Software Engineering Conference (ESEC) / International Symposium of Foundations of Software Engineering (FSE), pp. 263–272. ACM, 2005.

2. Applied Software Verification

2,00 Weekly Contact Hours

Mode of Delivery: Practicals

Lecturers: Scientific Staff Praktische Informatik, insbesondere Softwaretechnik und Programmiersprachen

Language: English/German

Frequency: every summer semester

Contents:

The practicals take place as inverted classroom and exercise the theoretical and practical concepts taught in the lectures by applying them to solve verification problems using automated verification tools, and also by engaging in pen-and-paper exercises. Typically every semester week, an exercise sheet is distributed to students who are expected to solve its exercises at home. In the practicals moderated by a lecturer, students are given the opportunity to present their solutions and to discuss them with their peers.

Literature:

- see the corresponding lectures -

Examination

Coursework Assignment and Colloquium / Duration of Examination: 20 minutes
Duration of Coursework: 3 weeks

Description:

The assignment consists of theoretical and practical exercises that require students to apply the knowledge conveyed in the lectures and practicals. The

colloquium covers questions testing this knowledge, on the basis of the submitted solutions to the assignment.	

Module SYSNAP-OSE-M Operating Systems Engineering *Operating Systems Engineering*	6 ECTS / 180 h
(since SS26) Person responsible for module: Prof. Dr. Michael Engel	

Contents:

Operating systems and related system software such as hypervisors form the basis of today's computer systems. The design and implementation of the core parts of system software can have significant impact not only on the performance of a computer system, but also on other aspects such a safety, security, and energy efficiency. Thus, the design and implementation of operating systems is a highly relevant topic for students working in all areas of computer science, from small embedded systems to large virtualized Cloud infrastructures.

This module concentrates on the central part ("kernel") of an operating system, i.e. the part of the system running in a privileged processor mode that interacts directly with hardware. Based on seminal publications, students will investigate different architectures of kernels, such as monolithic, micro- and exokernels, hypervisors and also unikernels. Mechanisms and policies of operating systems will be analyzed with respect to their functional as well as non-functional properties. The analysis of mechanisms dependent on a specific processor architecture will be explained using the modern and open RISC-V processor architecture.

A central part of this module will consist of code reading and the development of pieces of code for a small operating system. Different aspects of operating system functionality will be demonstrated through existing code. Constraints of, extension possibilities for, as well as alternative approaches to implement a given functionality will be discussed; this discussion will then form the basis for the implementation of a given feature in the practical exercises. An example for this is the discussion of file systems; here, features of a given traditional inode-based file system will be discussed and analyzed and alternative implementations, such as log-structured file systems, will be investigated and implemented in a basic form.

Learning outcomes:

The module is designed to enable students to not only understand the internals of operating systems, but also learn about different aspects of their implementation and the interaction between hardware and software. Starting from a thorough analysis of the internals of modern operating systems, this module will continue to present and discuss novel and non-traditional approaches to operating systems in the second half of the semester.

Successful students will be able to understand design and implementation aspects of system software as well as to comprehend and critically analyze proposed new approaches from the literature. They will also be able to understand the structure of and extend a given operating system code base with new functionality and test as well as evaluate functional and non-functional properties of the implementation. By writing system-level code running directly on hardware (or a hardware emulator), students will also be able to gain a better understanding of the operation of hardware and its interaction with software.

prerequisites for the module:

none

Recommended prior knowledge:	**Admission requirements:**
Participants should be familiar with basic concepts of operating systems and computer architecture, e.g. as acquired by	-

taking the module "Grundlagen der Rechnerarchitektur und Betriebssysteme" (Inf-GRABS-B). In addition, knowledge of C programming, debugging using gdb, using the Unix command line, and software construction tools (e.g. make) are useful.		
Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units

1. Operating Systems Engineering — **2,00 Weekly Contact Hours**

Mode of Delivery: Lectures
Lecturers: Prof. Dr. Michael Engel
Language: German/English
Frequency: every summer semester

Learning outcome:
cf. module description

Contents:
cf. module description

Literature:

- Russ Cox, Frans Kaashoek and Robert Morris, "xv6: a simple, Unix-like teaching operating system", MIT PDOS group 2020, https://pdos.csail.mit.edu/6.S081/2020/xv6/book-riscv-rev1.pdf
- Zhao Jiong, "A Heavily Commented Linux Source code", http://www.oldlinux.org/download/ECLK-5.0-WithCover.pdf
- Marshall Kirk McKusick et al., "The Design and Implementation of the 4.4 BSD Operating System", Addison-Wesley 1996, ISBN-13: 978-0132317924
- Uresh Vahalia, "Unix: the New Frontiers", Pearson 1996, ISBN-13: 978-0131019089
- John Lions, "Commentary on the 6th Edition Unix System", 1977, https://warsus.github.io/lions-/
- David Patterson and Andrew Waterman, "The RISC-V Reader: An Open Architecture Atlas", Strawberry Canyon 2017, ISBN-13: 978-0999249116$
- Andrew Waterman, Krste Asanovic and John Hauser (eds.), "The RISC-V Instruction Set Manual Volume II: Privileged Architecture", Document Version 20211203, https://github.com/riscv/riscv-isa-manual/releases/download/Priv-v1.12/riscv-privileged-20211203.pdf

In addition, selected papers will be provided.

2. Operating Systems Engineering — **2,00 Weekly Contact Hours**

Mode of Delivery:
Lecturers: Prof. Dr. Michael Engel
Language: German/English
Frequency: every summer semester

Learning outcome:
cf. module description

Contents:

cf. module description	

Examination Written examination / Duration of Examination: 90 minutes **Description:** Oral exam (30 minutes) or electronic (written) exam (90 minutes). Depending on the number of participants, the exam will either be an oral exam or a written exam. The mode of examination will be communicated in the first lecture. Students are assumed to work on a programming assignment ('schriftliche Hausarbeit') during the semester that is introduced at the beginning of the semester and uses the most important technologies discussed during the semester. **Note:** Without working on the programming assignment over the term students may run into problems during their oral examination (Kolloquium) as we discuss questions concerning topics from the lectures as well as from the assignment; questions about the assignment are based on the assignment solution programmed by the students.	

Module SYSNAP-PMAP-M Processor Microarchitecture and Performance *Processor Microarchitecture and Performance*	6 ECTS / 180 h
Person responsible for module: Prof. Dr. Michael Engel further responsible : Werner Haas	
Contents: Modern computer systems include high-performance processors which enable computationally demanding applications such as video and audio processing, handling of big data amounts or deep neural networks. Exploiting this performance potential for modern applications, however, is difficult, since increased performance levels could only be achieved by introducing additional complexity into the architecture of computer systems – for example, multiprocessor and multicore systems, multi-level memory hierarchies, and memory models with relaxed consistency. This course gives an insight into architectural details of modern processor architecture and their impact on non-functional properties. Whereas performance is the central topic of the course, additional non-functional properties such as energy consumption and security will be discussed. In addition to gaining theoretical insight into modern features of processor and system architecture, the course also discusses the interaction of software and hardware and how to optimize software for given architectural features.	
Learning outcomes: The module is designed to enable students to not only understand the internals of modern microprocessors and computer systems, but also learn about the non-functional properties involved and how the interaction between hardware and software relates to these. Starting with an overview of contemporary processors, this module will present and discuss different performance-improving aspects of processor architectures and their impact on software. Successful students will develop an understanding of modern processor architectures and the related systems as well as the resulting non-functional properties. They can comprehend and critically analyze existing and proposed new approaches from the literature. By writing code and analyzing the impact of different architectural features on the software, students will be able to gain a better understanding of the operation of hardware and its interaction with software and be able to optimize software for a given architecture and memory hierarchy.	
prerequisites for the module: verpflichtende Nachweise de	
Recommended prior knowledge: Fundamentals of computer architecture and operating systems, e.g. module PSI-EiRBS-B Operating Systems Engineering (SYSNAP-OSE-M) and/or Virtualization (SYSNAP-Virt-M)	**Admission requirements:** none
Frequency: every summer semester **Recommended semester:** **Minimal Duration of the Module:** 1 Semester	

Module Units	
1. Lecture Processor Microarchitecture and Performance **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Michael Engel	**2,00 Weekly Contact Hours**

Language: English/German **Frequency:** every summer semester **Learning outcome:** cf. module description **Contents:** 1 Intro/Recap: stored program arch, ISA, abstraction, iron law of performance 2 Simple pipelining: pipeline hazards, superscalar processing, exception handling 3 Caches: direct mapped, set/fully associative, memory hierarchy 4 Virtual memory: segmentation, paging, TLB, aliases/synonyms, VP/PP caches 5/6 Out of order execution – register renaming, Tomasulo algorithm – memory disambiguation, load/store queues 7/8 Branch prediction – branch history – branch targets 9 Symmetric multiprocessing: sequential consistency, cache coherence protocols 10 Virtualisation: processor modes, sensitive instructions, multi-level translation 11 Side channels: cache state, timing sources, resource contention 12 Transient execution attacks: Meltdown, Spectre, Retpoline **Literature:** John L. Hennessy, David A. Patterson Computer Architecture: A Quantitative Approach Morgan Kaufmann, 6th Edition 2017 ISBN-13: 978-0128119051 John Paul Shen, Mikko H. Lipasti Modern Processor Design: Fundamentals of Superscalar Processors Waveland Pr Inc, Reprint Edition 2013 ISBN-13: 978-1478607830	
2. Exercises Processor Microarchitecture and Performance **Mode of Delivery:** **Lecturers:** Prof. Dr. Michael Engel **Language:** English/German **Frequency:** every summer semester **Learning outcome:** cf. module description **Contents:** cf. module description **Literature:** cf. module description	**2,00 Weekly Contact Hours**
Examination Portfolio / Duration of Coursework: 3 months	

Module SYSNAP-Virt-M Virtualization *Virtualisierung*	6 ECTS / 180 h
(since SS26) Person responsible for module: Prof. Dr. Michael Engel	
Contents: Virtualization is the basis of a significant part of the Internet infrastructure today. It is used in different contexts such as system-level virtualization for co-hosting virtual machines in Cloud infrastructures or just-in-time translation of JavaScript code in web applications. This module discusses virtualization technologies on all layers of the hardware/software stack, from system-level virtualization to virtual machines for high-level languages. Based on publications and real-world code examples, students will investigate different architectures of virtual machines. The design and implementation of virtualization technologies will be analyzed through the investigation of real-world open-source code examples for common hardware, such as x86, ARM and RISC-V.	
Learning outcomes: The module is designed to enable students to understand the different approaches to virtualization and learn details about their design and implementation. Students will learn to analyze the advantages and disadvantages of virtualization on different layers of a computer system and will gain experience in isolation and security properties of virtualized systems. Successful students will be able to understand design and implementation aspects of different virtualization approaches as well as to comprehend and critically analyze proposed new approaches from the literature. They will also be able to understand the structure of and extend a given virtualization system code base with new functionality and test as well as evaluate functional and non-functional properties of the implementation.	
prerequisites for the module: none	
Recommended prior knowledge: Participants should be familiar with basic concepts of operating systems and computer architecture, e.g. as acquired by taking the module "Grundlagen der Rechnerarchitektur und Betriebssysteme" (Inf-GRABS-B). In addition, knowledge of C programming, debugging using gdb, using the Unix command line, and software construction tools (e.g. make) are useful.	**Admission requirements:** -
Frequency: every winter semester **Recommended semester:** **Minimal Duration of the Module:** 1 Semester	

Module Units	
1. Virtualisierung **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Michael Engel **Language:** German/English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Learning outcome: c.f. module description	
Contents:	

c.f. module description

Literature:

- Jim Smith and Ravi Nair,
  Virtual Machines: Versatile Platforms for Systems and Processes
  Morgan Kaufmann, 1st edition 2005, ISBN-13: 978-1558609105
- Steven Hand, Andrew Warfield, Keir Fraser, Evangelos Kotsovinos, Dan Magenheimer
  Are Virtual Machine Monitors Microkernels Done Right?
  Proceedings of HotOS'05, 2005
- Gernot Heiser, Volkmar Uhlig and Joshua LeVasseur,
  Are virtual-machine monitors microkernels done right?,
  ACM SIGOPS Oper. Syst. Rev., vol. 40, number 1, 2006
- Barham, Paul, et al.,
  Xen and the art of virtualization,
  ACM SIGOPS operating systems review 37.5 (2003): 164-177
- Heiser, Gernot, and Kevin Elphinstone.
  L4 microkernels: The lessons from 20 years of research and deployment,
  ACM Transactions on Computer Systems (TOCS) 34.1 (2016): 1-29
- Engler, Dawson R., M. Frans Kaashoek, and James O'Toole Jr.,
  Exokernel: An operating system architecture for application-level resource management,
  ACM SIGOPS Operating Systems Review 29.5 (1995): 251-266
- Aycock, John,
  A brief history of just-in-time,
  ACM Computing Surveys (CSUR) 35.2 (2003): 97-113

Additional selected papers will be provided as required.

2. Virtualisierung — **2,00 Weekly Contact Hours**

Mode of Delivery:
Lecturers: Prof. Dr. Michael Engel
Language: German/English
Frequency: every winter semester

Learning outcome:
c.f. module description

Contents:
c.f. module description

Examination
Written examination / Duration of Examination: 90 minutes
Description:
Oral exam (30 minutes) or electronic (written) exam (90 minutes). Depending on the number of participants, the exam will either be an oral exam or a written exam. The mode of examination will be communicated in the first lecture.

Students are assumed to work on a programming assignment ('schriftliche Hausarbeit') during the semester that is introduced at the beginning of the

semester and uses the most important technologies discussed during the semester.	

Module VIS-IVVA-M Advanced Information Visualization and Visual Analytics *Advanced Information Visualization and Visual Analytics*	6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Dr. Fabian Beck	
Contents: The course discusses methods for interactive information visualization and systems for explorative visual analysis. Visualizations blend with algorithmic solutions and get adopted to domain-specific needs. Giving a research-oriented perspective, the design and evaluation of such methods is the focus of the course, as well as their practical and interdisciplinary application in various fields.	
Learning outcomes: The students recognize the possibilities and limitations of data visualization and are able to apply visualization methods to concrete application examples. They understand the foundations of visual perception and cognition as well as their implications for the visual representation of data. They have a sound overview of possibilities for the visual representation of abstract data and are able to adapt visualization techniques to new problems and justify design decisions. On a conceptual level, they are able to integrate visualization techniques with interaction techniques and algorithmic solutions and design visual analytics solutions. They can evaluate visualization techniques in quantitative and qualitative user studies.	
Remark: The workload for this module typically is as follows: • Lecture and exercise sessions: 45h • Preparation and review of the lecture: 30h • Work on exercises and assignments: 75h • Preparation for the exam: 30h	
prerequisites for the module: none	
Recommended prior knowledge: Basic knowledge in information visualization (e.g., as provided through VIS-GIV-B) is recommended; knowledge in programming, algorithms and data structures, human-computer-interaction, and machine learning and data science can be beneficial.	**Admission requirements:** none

Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
1. Advanced Information Visualization and Visual Analytics **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Fabian Beck **Language:** English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Contents: See module description	

Literature: Further material and reading will be announced in the course.	
2. Advanced Information Visualization and Visual Analytics **Mode of Delivery:** Practicals **Lecturers:** N.N. **Language:** English **Frequency:** every winter semester **Contents:** In the exercise sessions, lecture contents are expanded upon and their application is practiced.	**2,00 Weekly Contact Hours**

Examination Written examination / Duration of Examination: 90 minutes **Description:** By voluntarily handing in graded assignments (semesterbegleitende Studienleistungen) during the semester, points can be collected to improve the grade, which can be credited to the exam, provided that the exam is also passed without points from assignments. At the beginning of the course, it will be announced whether graded assignments are offered. If offered, the number, type, scope and processing time of the assignments as well as the number of achievable points per assignment and in the module examination will also be announced at this time. A grade of 1.0 can also be achieved without points from the assignments.	

Module xAI-DL-M Deep Learning *Deep Learning*		6 ECTS / 180 h
(since WS24/25) Person responsible for module: Prof. Dr. Christian Ledig		
Contents: Deep Learning is a form of machine learning that learns hierarchical concepts and representations directly from data. Enabled by continuously growing dataset sizes, compute power and rapidly evolving open-source frameworks Deep Learning based AI systems continue to set the state of the art in many applications and industries. The course will provide an introduction to the most relevant techniques in the field of Deep Learning and a broad range of its applications.		
Learning outcomes: In this course students will learn/recap some fundamentals from mathematics and machine learning that are critical for the introduction of the concept of Deep Learning. Participants will learn about various foundational technical aspects including optimization and regularization strategies, cost functions and important network architectures such as Convolutional Networks. Students will further get an insight into more advanced concepts such as sequence modelling and generative modelling. Participants will further learn about representative architectures of important algorithm categories, e.g., classification, detection, segmentation, some of their concrete use cases and how to evaluate them. The lecture is accompanied by exercises and assignments that will help participants develop practical, hands-on experience. In those exercises students will learn how to implement and evaluate Deep Learning algorithms using Python and its respective commonly used libraries.		
Remark: The lecture is conducted in English. The workload of this module is expected to be roughly as follows: • Lecture: 22.5h (equals the 2 SWS) • Preparation of lectures and analysis of further sources: 30h (over the 15 weeks term) • Exercise classes accompanying lecture: 22.5h (equals the 2 SWS) • Work on the actual assignments: 75h (over the 15 weeks term) • Preparation for exam: 30h		
prerequisites for the module: none		
Recommended prior knowledge: Strongly recommended: Good working knowledge of programming (in particular Python), Mathematics for Machine Learning [xAI-MML] Further recommended (or similar): Bachelorproject Erklärbares Maschinelles Lernen [xAI-Proj-B], Lernende Systeme / Machine Learning [KogSys-ML-B], Einführung in die Künstliche Intelligenz / Introduction to AI [KogSys-KI-B], Algorithmen und Datenstrukturen [AI-AuD-B]		**Admission requirements:** none
Frequency: every winter semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
1. Deep Learning **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Christian Ledig **Language:** English/German **Frequency:** every winter semester **Learning outcome:** c.f. module description **Contents:** The lecture will be held in English. The following is a selection of topics that will be addressed in the course • Relevant concepts in linear algebra, probability and information theory • Deep feedforward networks • Convolutional Neural Networks • Regularization, Batch Normalization • Optimization (Backpropagation, Stochastic Gradient Decent) and Cost Functions • Classification (binary, multiclass, multilabel) • Object Detection & Segmentation • Generative Modelling • Attention mechanisms & Transformer Networks • Evaluation of ML approaches **Literature:** • Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep Learning, MIT Press, 2016 • Zhang, Lipton, et al.: Dive into Deep Learning (https://d2l.ai/) Further literature will be announced at the beginning of the course.	**2,00 Weekly Contact Hours**
2. Deep Learning **Mode of Delivery:** Practicals **Lecturers:** N.N. **Language:** English/German **Frequency:** every winter semester **Learning outcome:** see module description **Contents:** Further exploration of concepts discussed in the lecture, often accompanied by assigments and programming exercises implemented in Python and the corresponding machine/deep learning libraries. **Literature:** see lecture description	**2,00 Weekly Contact Hours**

Examination	
Written examination / Duration of Examination: 90 minutes **Description:**	

The content that is relevant for the exam consists of the content presented in the lecture and exercises/tutorials (including the assignments) as well as additional content of the discussed literature, which will be highlighted. Participants can collect bonus points by working on and solving the assignments discussed during the exercises/tutorials. Details regarding the number of assignments, the number of points per assignment, and the type of assignments will be announced in the lecture. If the points achieved in the exam are sufficient to pass the exam on its own, the bonus points (at most 20% of the maximum achievable points in the exam) will be added to the points achieved in the exam. The grade 1.0 can be achieved without the bonus points.	

Module Handbook Summary

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
	A1 Software System Science **Specialisation Area: S1: Theory**		**30 - 69** **6 - 30**		
AISE-Auto	Automation of First- and Higher-Order Logic	every summer semester(1)	6	6 Lectures and Practicals	Oral examination 30 minutes
AISE-UL	Universal Logic & Universal Reasoning	every winter semester(1)	6	2 Lectures and Practicals 2 Practicals	Written examination (AISE-UL: Universal Logic & Universal Reasoning (Universelle Logik & Universelles Schließen))
AlgoK-Algo-M	Algorithms	every summer semester(startend SS24)	6	4 Lectures and Practicals	Sonstiges
GdI-FPRS-M	Functional Programming of Reactive Systems	every summer semester	6	2 Lectures 2 Practicals	Written examination 90 minutes Oral examination 30 minutes
GdI-IFP-M	Introduction to Functional Programming	every winter semester	6	2 Lectures 2 Practicals	Written examination 90 minutes
	Specialisation Area: S2: Systems		**6 - 30**		
	Deviating from the study and subject examination regulations, the examination format for the module **SYSNAP-OSE-M** will be “written or oral examination” starting in the summer semester 2026.				
COMNET-CN-M	Computer Networks	every summer semester	6	4 Lectures and Practicals	Sonstiges
COMNET-NES-M	Networked Embedded Systems	every winter semester	6	4 Lectures and Practicals	Sonstiges
DSG-DistrSys-M	Distributed Systems		6	2 Lectures 2 Practicals	Coursework Assignment and Colloquium

		every summer semester(2026)			3 months 15 minutes
DT-CPP-M	Advanced Systems Programming in C++ (Master)	every winter semester(1)	6	4 Lectures and Practicals	Portfolio 4 months
MOBI-DSC-M	Data Streams and Complex Event Processing	every winter semester(1)	6	2 Lectures 2 Practicals	Oral examination 15 minutes Written examination 60 minutes
SYSNAP-OSE-M	Operating Systems Engineering	every summer semester(1)	6	2 Lectures 2	Written examination 90 minutes
SYSNAP-PMAP-M	Processor Microarchitecture and Performance	every summer semester(1)	6	2 Lectures 2	Portfolio 3 months
SYSNAP-Virt-M	Virtualization	every winter semester(1)	6	2 Lectures 2	Written examination 90 minutes
	Specialisation Area: S3: Software		**6 - 30**		
DT-DBCPU-M	Database Systems for modern CPU	every summer semester(1)	6	4 Lectures and Practicals	Written examination 90 minutes
ESE-ESEng-M	Evidence-based Software Engineering	every winter semester	6	4 Lectures and Practicals	Internship report
MOBI-ADM-M	Advanced Data Management	every summer semester(1)	6	2 Lectures 2 Practicals	Written examination 75 minutes
PSI-AdvaSP-M	Advanced Security and Privacy	every summer semester(1)	6	2 Lectures 2 Practicals	Written examination 110 minutes
PSI-DiffPriv-M	Introduction to Differential Privacy		6	4 Lectures and Practicals	90 minutes

		every winter semester(1)			
SWT-ASV-M	Applied Software Verification	every summer semester	6	2 Lectures 2 Practicals	Coursework Assignment and Colloquium 3 weeks 20 minutes

Module Handbook Summary

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
	A2 Domain-specific Software System Science		**0 - 39**		
DS-ConvAI-M	Advanced Dialogue Systems and Conversational AI	every summer semester(1)	6	2 Lectures 2 Practicals	Oral examination 30 minutes
EESYS-ADAML-M	Applied Data Analytics and Machine Learning in R	every winter semester	6	2 Lectures 2 Practicals	Written examination 90 minutes
EESYS-ES-M	Energy Efficient Systems	every summer semester	6	2 Lectures 2 Practicals	Written examination 90 minutes
HCI-US-B	Ubiquitous Systems	every winter semester	6	2 Lectures 2 Practicals	Oral examination Written examination 90 minutes
ISPL-MDP-M	Managing Digital Platforms	every summer semester(1)	6	4	Written examination 90 minutes
NLPROC-DL4NLP-M	Deep Learning for Natural Language Processing	every summer semester	6	2 Seminar	Written examination 60 minutes
NLProc-EA-M	Emotion Analysis	every semester(1)	6	4 Lectures and Practicals	Coursework Assignment and Colloquium 1 hours
NLProc-ILT-M	Impact of Language Technology	every winter semester(1)	6	4 Lectures and Practicals	Written examination 60 minutes
NLProc-PGM4NLP-M	Probabilistic Graphical Models for Natural Language Processing	every winter semester(1)	6	4 Lectures and Practicals	Written examination 60 minutes
SNA-OSN-M	Project Online Social Networks	every winter semester	6	4 Practicals	Coursework Assignment and Colloquium 4 months 30 minutes

VIS-IVVA-M	Advanced Information Visualization and Visual Analytics	every winter semester(1)	6	2 Lectures 2 Practicals	Written examination 90 minutes
xAI-DL-M	Deep Learning	every winter semester(1)	6	2 Lectures 2 Practicals	Written examination 90 minutes

Module Handbook Summary

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
	A3 Seminar and Project		**9**		
ISoSySc-Proj-M	Master's Project in Software System Science	every semester	6	4	Coursework Assignment and Colloquium
ISoSySc-Sem-M	Master's Seminar in Software System Science	every semester	3	2 Seminar	Coursework Assignment with presentation

Module Handbook Summary

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
	A4 Master's Thesis		**30**		
SSS-Thesis-M	Master's Thesis in Software Systems Science	every semester	30		Coursework Assignment 6 months Colloquium

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
	A5 International Experience		**12 - 27**		
	Elective Area: Guided Study Abroad		**0 - 27**		
	Modules amounting to 27 ECTS credits can be included in this area, which are completed as part of a guided study abroad program at a foreign university, provided that they differ significantly from the modules to be completed in accordance with these regulations and can be assigned to module groups A1, A2 or A3.				
	Elective Area: Internship in an International Context		**0 - 12**		
SSS-PraktIntKon-M	Internship in an International Context	every semester(1)	12		Writen Report on Practical Training
	Elective Area: Foreign Languages		**0 - 15**		