Otto-Friedrich Universität Bamberg

Module Handbook

Master's Degree Programme
International Software System Science

Faculty of Information Systems
and Applied Computer Sciences

Notice on the validity of older versions of a module handbook:

1. date of validity

The module descriptions contained in this module handbook are valid for the first time for the semester indicated on the cover sheet.

2. transition regulations

a.   Students who have already completed parts of a module according to a previously valid module handbook shall complete the module according to the previously valid version of the module handbook.

This transition regulation shall apply exclusively to the regular examination date immediately following the missed/not passed/not completed examination. At the request of the student, the examination board may, in justified cases, determine an extension of the transition period.

b.   A module shall be deemed to have been completed in parts if the module examination has not been passed or missed. The same shall apply if at least one module examination has been passed, failed or missed.

Furthermore, a module shall be deemed to have been *partly completed* if the student has registered for a course assigned to the respective module in accordance with the previously applicable module handbook.

3. period of validity

This module handbook is valid for subsequent semesters *until the announcement of a changed module handbook*.

Notice on the validity of newer versions of examination regulations:

The provisions in § 36 apply in accordance with the Examination Regulations of 18.01.2021 (valid for students who started their studies from the summer semester 2021 onwards): A specialization field is indicated in the final examination certificate at the request of the student. A corresponding form for the application is published by the examination board.

List of deviations in the module program MSc. International Software System Science compared to the Examination Regulations of 06.03.2015

- The examination **duration** specifications in the column "Prüfungen" have been deleted from the study and examination regulations. The examination duration of a module is specified in the module handbook.
- The module **SWT-PCC-M** (elective module for students starting before WS19/20, compulsory module for students starting from WS19/20 on) will no longer be offered as of SS21. It was first replaced by the module SWT-CPS-M until summer semester 2023 and from winter semester 2023/24 on by the module **SWT-SWQ-M**, which was an elective module in module group A1 before. As of winter semester 2024/25 **SWT-SWQ-M** can no longer be offered and is replaced by **GdI-IFP-M.**
- The elective module **GdI-FP-M** will no longer be offered as of SS21.
- The module KogSys-ML-M is now offered as KogSys-ML-B and therefore is no longer eligible in the Master's program International Software System Science.
- The elective module **DSG-SRDS-M** will no longer be offered as of WS22/23.
- The elective module **SME-STE-M** will no longer be offered as of WS23/24.
- The examination form for **DT-DBCPU-M** was changed from oral examination to written examination.

Equivalence List M.Sc. International Software Systems Science
Examination Regulations of 06.03.2015

In the following you will find a list of modules whose name or abbreviation has been changed without a significant change to the module. If a module listed in the column "previous module" was successfully completed, the module listed in the column "new module" cannot be taken.

previous module			new module		
module abbreviation	module name	valid until (semester)	module abbreviation	module name	valid from (semester)
MOBI-DSC	Data Streams and Complex Event Processing	SS 18	MOBI-DSC-M	Data Streams and Complex Event Processing	WS 1819
GdI-AFP-M	Advanced Functional Programming	WS 2021	GdI-FPRS-M	Functional Programming of Reactive Systems	SS 21
EESYS-DAE-M	Data Analytics in der Energieinformatik	SS 21	EESYS-ADAML-M	Applied Data Analytics and Machine Learning in R	WS 2122
KogSys-ML-M	Lernende Systeme (Machine Learning)	SS22	KogSys-ML-B	Einführung in Maschinelles Lernen	WS22/23
xAI-MML-M	Mathematics for Machine Learning	SS24	xAI-MML-B	Mathematics for Machine Learning	WS24/25
GdI-IFP-B	Introduction to Functional Programming	SS24	GdI-IFP-M	Introduction to Functional Programming	WS24/25

Date of the equivalence and deviations list: 10.01.2025

Modules

Table of Contents

Index by areas of study

1) A1 Software Systems Science (Module group) ECTS: 36 - 54

Students who started their studies **before summer semester 2021** must complete modules of **30 - 48 ECTS points in A1**.

a) compulsory part (Elective Area) ECTS: 24

Students, who started their studies before winter semester 2019/20, do not have a compulsory area. The four modules listed here can be taken as electives instead.

All other students: Pleae note that the module **SWT-PCC-B** is no longer offered as of the winter semester 2020/21. It was first replaced by the module **SWT-CPS-M** until summer semester 2023 and from winter semester 2023/24 on by the module **SWT-SWQ-M**. As of winter semester 2024/25 SWT-SWQ-M can no longer be offered and is replaced by **GdI-IFP-M.**

b) elective modules (Elective Area) ECTS: 12 - 30

Students, who started their studies **before winter semester 2019/20**, must complete **30-48 ECTS points** in this area.

2) A2 Domain-specific Software Systems Science (Module group) ECTS: 0 - 18

3) A3 Seminar and Project (Module group) ECTS: 9
In A3 students who started their studies **before summer semester 2021** must complete **12 ECTS points** in this area.

a) Seminar (Elective Area) ECTS: 3

b) Project (Elective Area) ECTS: 6
Students, who started their studies **before SS21**, must complete one project of **9 ECTS** points in this area. Deviating from the modules listed below one of the following modules must be completed:

- DSG-Project-M

- KTR-SSSProj-M
- MOBI-PRS-M
- PSI-ProjectCAD-M
- SWT-PR2-M

4) A4: Masters Thesis (Module group) ECTS: 30

5) A5 International Experience (Module group) ECTS: 27

In A5 Students who started their studies **before summer semester 2021** can complete modules of **up to 30 ECTS points**.

According to the examination regulations (StuFPO) Appendix 1, students have **four options** regarding the areas from which the ECTS points can be acquired (independent from their enrollment semester):

(1) to study modules of software systems science at a university abroad for at least one semester or

(2) to accomplish a traineeship in an international context, preferentially abroad, that covers topics of the occupational field of software systems science with a volume of at least 360 working hours (12 ECTS credits).

(3) to accomplish credits in modules of foreign languages (neither English nor native language)

(4) to accomplish *further* modules of module groups A1 and A2 (Examination Regulations, App. 1).

a) Guided graduate study abroad (Area) ECTS: 0 - 27

Students who started their studies **before summer semester 2021**, can accomplish modules of up to **30 ECTS** points in this area.

The courses that are selected at a foreign university have to be approved by learning agreements. For own planning security reasons, learning agreements have to be signed by those Professors at University of Bamberg responsible for the chosen subject, as well as the head of the Examination Board, before the graduate study abroad is initiated.

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

Regarding the elective area 5b, *Internship in an international context*, with an equivalent workload of 12 ECTS credits, a foreign or internationally acting domestic company (or research institute) may be selected.
It has to offer a specific internship related to relevant topics of software systems science. The documentation of the internship requires the delivery of the following items to the degree programme representative:

- written report of 4 pages at least, reporting on the tasks and achievements, and
- a certificate issued by the hosting institution or the organizational unit that has realized the internship.

c) Foreign languages (Area) ECTS: 0 - 15

In the elective area 5c, *Foreign languages*, students take modules offered by the University's Language Centre.

Excluded are modules of the English language and modules of the language in which the university entrance qualification was obtained.

Details, in particular the modules available for selection and the respective Module examinations are described (in German) in the *Modulhandbuch des Sprachenzentrums der Otto-Friedrich-Universität Bamberg.*

Students who started their studies **before summer semester 2021**, can accomplish modules of up to **18 ECTS** points in this area.

d) further modules from module groups A1 and/ or A2 (Area) ECTS: 0 - 27

In this area additional, not previously completed modules from A1 or A2 module groups' required elective options can be completed in accordance with the Examination Regulations, Appendix 1.

Students who started their studies **before summer semester 2021** can complete modules of **up to 30 ECTS points** in this area.

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 WS22/23)

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:

Recommended prior knowledge: Basic knowledge about classical and non-classical logics, theoretical computer science.	**Admission requirements:** non

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 Algorithms *Algorithmen*	6 ECTS / 180 h

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:

Recommended prior knowledge: Prerequisites: Basic knowledge of algorithms and data structures, proof techniques, mathematical skills. Good English language skills.	**Admission requirements:** none

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

Module Units

Algorithms **Mode of Delivery:** Lectures and Practicals **Lecturers:** Prof. Dr. Isolde Adler	**4,00 Weekly Contact Hours**

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 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 **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Stefan Ultes **Language:** English/German **Frequency:** every summer semester	**2,00 Weekly Contact Hours**

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	**2,00 Weekly Contact Hours**
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.	

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-DSAM-M Distributed Systems Architectures and Middleware *Distributed Systems Architecture and Middleware*	6 ECTS / 180 h 45 h Präsenzzeit 135 h Selbststudium

(since WS19/20)
Person responsible for module: Prof. Dr. Guido Wirtz

Contents:

This course introduces students to the ideas, benefits, technologies and issues related to server-centric distributed systems and middleware in general. The core topics are centered around component technologies such as Java EJBs, Business-to-Business technologies like EDI and ebXML, and Cloud Computing facilities like Google App Engine and Windows Azure. Thus the course introduces and discusses in-depth topics concerning distributed middleware and its practical use:

- Characteristics and Foundations of Distributed Systems
- Classical Middleware and Services
- Concurrency and Synchronization
- Component Technologies
- Cloud Computing, in particular platform as a service
- Business-to-Business Technologies

The selection of topics and teaching method of this course reflects the Distributed Systems Group's (DSG) dedication to integrate business and IT, theory and practice, research and teaching. You not only will be taught the classical way, but you will have hands-on experience on middleware development and middleware tools. Also, you will get the chance to discuss selected publications with your lecturers.

Learning outcomes:

Students are able to evaluate, plan, design and implement server-centric distributed systems. Students are familiar with recent approaches and standards for building and managing such systems, know about the central problems involved as well as ways to overcome these issues. Students have hands-on experience with up-to-date middleware and tools for building server-centric systems.

Remark:

The main language of instruction in this course is English.

prerequisites for the module:

Basic knowledge in software engineering and in distributed systems as introduced, e.g., in the module DSG-IDistrSys-B (or DSG-DistrSys-M).

Recommended prior knowledge:	Admission requirements:
Basic knowledge in software engineering and in distributed systems as introduced, e.g., in the module DSG-IDistrSys.	none

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

Module Units

1. Lectures Distributed Systems Architecture and Middleware **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Guido Wirtz **Language:** English **Frequency:** every winter semester	2,00 Weekly Contact Hours

Learning outcome: c.f. overall module description	
Contents: c.f. overall module description	
Literature: This is a fast emerging field with new insights every year. So, up-to-date literature will be provided at the beginning of each course.	
2. Practicals Distributed Systems Architecture and Middleware **Mode of Delivery:** Practicals **Lecturers:** Scientific Staff Praktische Informatik **Language:** English/German **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Learning outcome: c.f. overall module description	
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: c.f. overall module description	
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 winter term or at the begin of the summer 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 DSG-DistrSys-M Distributed Systems *Distributed Systems*	6 ECTS / 180 h 45 h Präsenzzeit 135 h Selbststudium

(since SS20)
Person responsible for module: Prof. Dr. Guido Wirtz

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. Module Introduction to Parallel and Distributed Programming (DSG-PKS-B) - recommended	none

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

Module Units	

1. Lecture Distributed Systems **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Guido Wirtz **Language:** English/German **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
Learning outcome: c.f. module description	
Contents: c.f. module description	
Literature: • George Coulouris, Jean Dollimore, Tim Kindberg, Gordon Blair: Distributed Systems - Concepts and Design. Pearson Education UK, 2011 (5. edition); • Andrew Tanenbaum, Marten van Steen: Distributed Systems - Principles and Paradigms, 2017 (3rd edition) • Additional research literature will be provided during the term for selected readings and discussions	
2. Tutorial Distributed Systems **Mode of Delivery:** Practicals **Lecturers:** Scientific Staff Praktische Informatik **Language:** German **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
Learning outcome: c.f. module description	
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.	

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.	23

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

Module DSG-SOA-M Service-Oriented Architecture and Web Services *Service-Oriented Architecture and Web Services*	6 ECTS / 180 h 45 h Präsenzzeit 135 h Selbststudium
(since SS20) Person responsible for module: Prof. Dr. Guido Wirtz	

Contents:

Building enterprise-scale IT systems requires sound concepts for integrating software. **Service-oriented architectures (SOAs)** have been the number one answer to this integration challenge for years. Indeed, service orientation is and will be a cornerstone in modularizing large IT landscapes and alignment with business needs is the driving factor for service engineering. A SOA composes an IT system from services in a loosely-coupled manner. Each service implements a business task and therefore have a clear value attribution. When business needs change, the loose coupling of services allows for quick adjustment of the SOA. In recent years, Microservices have been put forward as a new paradigm for organizing software-intensive systems as a set of small services that communicate using lightweight communication technologies and are *independently deployable by fully automated deployment machinery*. Conceptually, Microservices and SOA share a lot, but the Microservices paradigm puts a lot more emphasis on automation in development and therefore is a better fit for modern development practices.

When moving beyond company boundaries and opening up the solution space is necessary, **software ecosystems (SECOs)**come into play. Software ecosystems integrate software contributions from independent organizational entities and enable software products and solutions that a single company cannot realize alone. Prominent representatives of software ecosystems are Android and the Playstore or iOS and the AppStore. But the paradigm of software ecosystems goes far beyond mobile platforms and also covers application areas in the cloud domain or the embedded domain.

Skilled software architects therefore reconcile the business views and technical views for the benefit of the enterprise and therefore need both, advanced knowledge in business process and workflow management as well as a rock-solid understanding of service engineering and distributed computing.

This course will introduce you to the world of architectures for large-scale software by giving a brief overview on distributed systems and software architecture in general. Then SOAs as an architectural paradigm and Web Services (WSDL + REST) as SOA implementation technology will be treated in detail. SOA will be contrasted to Microservices and the development aspects that Microservices focuses on will be discussed. Software ecosystems then will be introduced as a paradigm for organizing software systems and container technology (Linux Containers (LXC) and Docker) as a frequent implementation means for software ecosystems will be introduced. In particular, we will investigate what building industry-grade ecosystems based on container technology means in practice.

- Conceptual Foundations of SOA
- SOA Characterisitics
- Microservices
- WSDL and Basic Web Services
- REST-ful Services
- Software Ecosystems
- Container technology

The selection of topics and teaching method of this course reflects the Distributed Systems Group's (DSG) dedication to integrate business and IT, theory and practice, research and teaching. You not only will be taught the classical way, but you will have hands-on experience on service development and SOA tools.

Also, you will get a grasp of current services research and you will get the chance to discuss selected publications with your lecturers.

Learning outcomes:

Students know about the different aspects of service-oriented architectures and their practical use. Students

- Understand the characteristics of SOAs, Microservices and SECOs and its implications on IT systems.
- Know relevant technologies and standards in the field and being able to combine some of these to develop basic Web Services and service compositions
- Being able to compare WSDL Web Services to REST Web Services
- Being able to use container technology for integrating software
- Being able to judge IT architectures from a SOA/Microservices/SECO perspective.
- Being able to understand and discuss scientific work in the area

Remark:

The main 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 final exam: 30h

prerequisites for the module:

Basic knowledge in software engineering and in distributed systems as introduced, e.g., in the modules DSG-IDistrSys-B or DSG-DistrSys-M.

Recommended prior knowledge:	Admission requirements:
Basic knowledge in software engineering and distributed systems.	none
Module Introduction to Distributed Systems (DSG-IDistrSys) - recommended	

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

Module Units

1. Lectures Service-Oriented Architecture and Web Services	**2,00 Weekly Contact Hours**
Mode of Delivery: Lectures	
Lecturers: Prof. Dr. Guido Wirtz, Scientific Staff Praktische Informatik	
Language: English	
Frequency: every summer semester	
Learning outcome:	
c.f. overall module description	
Contents:	
c.f. overall module description	
Literature:	

SOA is still a fast emerging field - most recent version of standards and up-to-date literature will be provided at the beginning of each course.	
2. Practicals Service-Oriented Architecture and Web Services **Mode of Delivery:** Practicals **Lecturers:** Scientific Staff Praktische Informatik **Language:** English/German **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
Learning outcome: c.f. overall module description	
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: c.f. overall module description	
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. S**tudents 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:

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	**2,00 Weekly Contact Hours**
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. Practicals Energy Efficient Systems **Mode of Delivery:** Practicals **Language:** German/English **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
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.	

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 GAMES-DGS-M Designing Gamified Systems *Designing Gamified Systems*	6 ECTS / 180 h

(since SS25)

Person responsible for module: Prof. Dr. Benedikt Morschheuser

Contents:

Driven by the rising popularity of digital games, technology, business, and society are increasingly influenced by trends in the gaming industry. One of the probably most important phenomena of this multi-faceted development is "gamification", which refers to the use of design principles and features from games when designing information systems, processes, or services to afford similar positive experiences, skills, and practices as found in games and facilitate changes in behaviors.

In recent years, gamification has become an umbrella term that encompasses and includes, to varying degrees, other related technological developments such as serious games, game-based learning, exergames, games with a purpose/human computation games and persuasive technology. Further, gamification is applied across industries, such as marketing, commerce, software development, education, logistics, and healthcare.

This course:

- teaches key concepts, design patterns, and approaches of gamified and hedonic (i.e., games and video games) information systems.
- offers deep insights into the theoretical foundations of game design, motivational psychology, and information system design.
- introduces methods and frameworks for designing gamified systems and managing gamification projects.
- discusses latest research findings and the potential impact of gamification on society, economy, and everyday life.

The course is complemented by a practical design project, where students in a team select and apply design methods & techniques in order to create a prototype of a gamified / hedonic information system. Within this project, the students can apply the knowledge and skills acquired in this lecture and their studies in a challenging context.

Learning outcomes:

Students gain knowledge in the conceptual, technological and theoretical foundations of gamified and hedonic information systems. They learn to describe and discuss their underlying design principles as well as their psychological and behavioral outcomes. They can analyze and compare existing gamified and hedonic information systems and learn to critically reflect relevant ethical aspects of using gamification in practice. Students learn state-of-the-art methods, techniques, and tools for successfully implementing gamification projects and are able to select and apply them. Students train their creativity and prototyping skills. Further, they can improve their collaboration and presentation skills in this course.

Remark:

The workload for this module is roughly broken down as follows:

- Participation in lectures (~40h)
- Participation at project sessions (~10h)
- Self-study and work on group project (~110h)
- Preparation for exam (~20h)

The number of participants for this course is limited. If, due to capacity restrictions, a selection of students in courses with limited admission capacity becomes necessary, a decision on admission will be made after the registration period has expired. Please also note that registration for the course does not automatically lead to admission nor registration for the module examination. Please see website for details on the application process.

prerequisites for the module:
none

Recommended prior knowledge:	**Admission requirements:**
Profound English skills are required. Further, creativity, experience in HCI, user experience, prototyping, and software engineering are helpful. Participating students should be motivated to work on a challenging topic in interdisciplinary groups.	Active participation in the group project

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

Module Units

Designing Gamified Systems **Mode of Delivery:** **Lecturers:** Prof. Dr. Benedikt Morschheuser **Language:** Deutsch / English on demand **Frequency:** every summer semester	**4,00 Weekly Contact Hours**

Learning outcome:
Students gain knowledge in the conceptual, technological and theoretical foundations of gamified and hedonic information systems. They learn to describe and discuss their underlying design principles as well as their psychological and behavioral outcomes. They can analyze and compare existing gamified and hedonic information systems and learn to critically reflect relevant ethical aspects of using gamification in practice. Students learn state-of-the-art methods, techniques, and tools for successfully implementing gamification projects and are able to select and apply them. Students train their creativity and prototyping skills. Further, they can improve their collaboration and presentation skills in this course.

Contents:
This course:

- teaches key concepts, design patterns, and approaches of gamified and hedonic (i.e., games and video games) information systems.
- offers deep insights into the theoretical foundations of game design, motivational psychology, and information system design.
- introduces methods and frameworks for designing gamified systems and managing gamification projects.
- discusses latest research findings and the potential impact of gamification on society, economy, and everyday life.

Literature:
Koivisto, J & Hamari, J. (2019). The rise of motivational information systems: A review of gamification research. International Journal of Information Management, 45. pp. 191-210.

Morschheuser, B., Hassan, L., Werder, K., Hamari, J. (2018). How to design gamification? A method for engineering gamified software. Information & Software Technology, 95. pp. 219-237.

Radoff, J. (2011). Game On: Energize Your Business with Social Media Games. Wiley, USA. Salen, K. (2004). Rules of play: game design fundamentals. MIT Press, Cambridge, USA.
Further literature will be made available in the lecture.

Examination
Coursework Assignment and Colloquium

Module GdI-CSNL-M Computational Semantics of Natural Language *Computational Semantics of Natural Language*	6 ECTS / 180 h

(since WS23/24)

Person responsible for module: Prof. Ph.D. Michael Mendler

further responsible : Luke Burke

Contents:

The formal study of natural language syntax and semantics has developed as a very lively sub-field of linguistics in the past 50 years, with the typed lambda calculus in particular providing a way of giving compositional analyses of meanings in natural language. Recently, monads and continuations have been employed as tools in natural language syntax and semantics. The aim of this module is to introduce the use of monads and continuations in natural language semantics and to discuss different approaches to the formal representation of quantifier scope ambiguities in natural language. The basics of natural language semantics (typed lambda calculus) will be briefly introduced, before discussing a continuation-based approach to quantification in natural language, which will be contrasted with other approaches. Monads representing focus, intensionality and non-determinism in natural language will be discussed. We will look at how analyses of the meaning of sentences can be represented in Haskell.

Importantly, the course may differ slightly from other courses in that assessment will not concentrate on technical exercises; rather, we require careful reading and dissection of relevant literature on the topic, since the primary mode of assessment will be via seminar presentations and essays, and you will be assessed on your understanding of, and your independent analysis of, relevant literature discussed in lectures. Independent reading of this literature will in fact be essential.

This course may also be of interest to students in philosophy and linguistics.

Learning outcomes:

At the end of this course students should be familiar with different approaches to the formal representation of quantifier scope ambiguities in natural language; be familiar with how monads and continuations have been used in natural language semantics; be familiar with the use of Haskell to formalise analyses in natural language semantics; be able to produce and manipulate terms of the typed lambda calculus to represent how meanings combine; have an understanding of how both logics and trees have been used to represent natural language syntax; be acquainted with logics such as Montague's "Intensional Logic" and Gallin's Ty2.

Remark:

The workload for this module consists of:

- participation in lectures and tutorial sessions: 45hrs
- individual preparation and reading: 105hrs
- exam preparation and oral exam: 30hrs

prerequisites for the module:

Recommended prior knowledge:	**Admission requirements:**
Willingness to read relevant literature, critically discuss and analyse it and write about it. Basic logic (GdI-Mfl-1: Mathematik fur Informatik or an equivalent level of understanding). Some knowledge of modal logic more basic than that required for (GdI-MTL: Modal and	English language skills at Level B2 (UniCert II) or above.

Temporal Logic). Knowledge of the typed lambda calculus (abstraction and application) and elementary Haskell (GdI-IFP: Introduction to Functional Programming) would be very useful, though not essential.

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

Module Units	
Computational Semantics of Natural Language **Mode of Delivery:** Lectures and Practicals **Language:** English **Frequency:** every summer semester	**4,00 Weekly Contact Hours**
Contents: Through prepared class presentations, essay writing, and direct interactions with the students the lecturer introduces the topics of the course in detail. The seminars deepen the students' understanding of the theoretical concepts and constructions covered in the lectures through presentations, which involve comparing alternative analyses of linguistic phenomena.	
Literature: • van Eijck, J. And Unger, Christina, "Computational Semantics with Functional Programming", Cambridge University Press 2010 • Barker, C. and Shan, C.-C., "Continuations and natural language", Volume 53. Oxford studies in Theoretical Linguistics, Oxford University Press, 2014 • Carpenter, Bob, "Type-Logical Semantics", MIT Press (1997) • Keenan, Edward, and Stabler, Edward, "Mathematical structures in Language", CSLI publications, Stanford, 2016 • Gallin, Daniel, "Intensional and Higher-Order Modal logic. North Holland, 1975.	

Examination Portfolio / Duration of Examination: 80 minutes **Description:** The components of the portfolio will be announced at the beginning of each semester.	

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:

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 **Mode of Delivery:** Lectures **Lecturers:** Prof. Ph.D. Michael Mendler **Language:** English/German **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
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 **Mode of Delivery:** Practicals **Lecturers:** Prof. Ph.D. Michael Mendler **Language:** English/German **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
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-MCI-M Human-Computer Interaction *Mensch-Computer-Interaktion*	6 ECTS / 180 h

(since WS21/22)
Person responsible for module: Prof. Dr. Tom Gross

Contents:
Advanced theoretical, methodological, and practical foundation of Human-Computer Interaction

Learning outcomes:
The aim of this module is to teach advanced knowledge and skills in the area of human-computer interaction as well as a broad 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 are later able to critical review new literature and systems.

Remark:
http://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, but without 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	
Human - Computer Interaction **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Tom Gross **Language:** German/English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Contents: After an introduction into the subject the following topics are covered in this lecture: - Mobile human-computer interaction	

- Adaptivity and adaptibility
- Information visualisation
- Tangible user interaction
- Usability engineering
- Usability and economics

Literature:

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

- Jacko, J.A. and Sears, A., (Eds.). Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications. Lawrence Erlbaum, Hillsdale, NJ, 2002.
- Hammond, J., Gross, T. and Wesson, J., (Eds.). Usability: Gaining a Competitive Edge. Kluwer Academic Publishers, Dordrecht, 2002.

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	
Human-Computer Interaction **Mode of Delivery:** Practicals **Lecturers:** Scientific Staff Mensch-Computer-Interaktion **Language:** German/English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
Contents: Practical assignments based on the subjects of the lecture.	
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 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:

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

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	**4,00 Weekly Contact Hours**
Literature: The specific literature that we will use in the course will be communicated or distributed in class or through the learning platform (VC).	

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 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:

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:

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:** 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:

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 KTR-GIK-M Foundations of Internet Communication *Grundbausteine der Internet-Kommunikation*	6 ECTS / 180 h 45 h Präsenzzeit 135 h Selbststudium

(since SS20)

Person responsible for module: Prof. Dr. Udo Krieger

Contents:

The course provides an introduction to the theoretical foundations of important technical issues related to the fundamentals of Internet communication, the data link layer, routing and transport protocols in IP networks, as well as advanced topics such as real-time communication and security in IP networks. The implementation of the learnt concepts in terms of predetermined configuration tasks in the communication laboratory by small teams of students constitutes the tutorial part of the course. For this purpose, guidelines, technical instructions, and tools will be provided.

The implementation tasks include the configuration and testing of computer networks in the laboratory setting. Operating system and required software components like Wireshark, Atheris and Vyatta software router will be provided. The basic handling of the hardware and software itself will be perfomed by the students as part of their individual intellectual efforts within te couse.

Learning outcomes:

The important skill to provide a qualified assessment of current communicaton technologies and corresponding practical knowledge can only be acquired by team-oriented processes subject to time constraints and the clear specification of technical and administrative objectives. In the course Foundations of Internet Communication and its tutorials in the router laboratory students will learn to work independently with a high level of responsibility as self-confident member of a successful team.

It is the objective of the course that the students acquire practical knowledge on modern data communication in Internet and learn how communication concepts can be developed, implemented and judged with th highest level of expertise.

The course is open to bachelor students in their transition phase to the master prgram. It attempts to prepare for the job in communication industry related fields. Master students in the first semester and exchange students from abroad are invited to join the course.

Remark:

The module can be selected by exchange students and master students speaking only English.

The workload is composed of the following items:

- participation in lectures, tutorials in the laboratory, laboratory meetings: 45 hours
- preparation, execution, post-processing of lectures and tutorials in the laboratory: 100 hours
- preparation of the examination: 35 hours

prerequisites for the module:

Recommended prior knowledge:	**Admission requirements:**
- data communication similar to module KTR-Datkomm-B - fundamental knowledge on programming in JAVA (or C++) - working knowledge on LINUX is recommended, but not assumed Module Algorithms and Data Structures (AI-AuD-B) - recommended	governed by examination regulations (StuFPO)

Module Introduction to Algorithms, Programming and Software (DSG-EiAPS-B) - recommended Module Data communication (KTR-Datkomm-B) - recommended		
Frequency: every summer semester	**Recommended semester:**	**Minimal Duration of the Module:** 1 Semester

Module Units	
Foundations of Internet Communication **Mode of Delivery:** Lectures and Practicals **Lecturers:** Prof. Dr. Udo Krieger **Language:** English/German **Frequency:** every summer semester	**4,00 Weekly Contact Hours**

Learning outcome:

The important skill to provide a qualified assessment of current communicaton technologies and corresponding practical knowledge can only be acquired by team-oriented processes subject to time constraints and the clear specification of technical and administrative objectives. In the course Foundations of Internet Communication and its tutorials in the router laboratory students will learn to work independently with a high level of responsibility as self-confident member of a successful team.

It is the objective of the course that the students acquire practical knowledge on modern data communication in Internet and learn how communication concepts can be developed, implemented and judged with th highest level of expertise.

The course is open to bachelor students in their transition phase to the master prgram. It attempts to prepare for the job in communication industry related fields. Master students in the first semester and exchange students from abroad are invited to join the course.

Contents:

The course provides an introduction to the theoretical foundations of important technical issues related to the fundamentals of Internet communication, the data link layer, routing and transport protocols in IP networks, as well as advanced topics such as real-time communication and security in IP networks. The implementation of the learnt concepts in terms of predetermined configuration tasks in the communication laboratory by small teams of students constitutes the tutorial part of the course. For this purpose, guidelines, technical instructions, and tools will be provided.

The implementation tasks include the configuration and testing of computer networks in the laboratory setting. Operating system and required software components like Wireshark, Atheris and Vyatta software router will be provided. The basic handling of the hardware and software itself will be perfomed by the students as part of their individual intellectual efforts within the couse.

The organization of the laboratories is following the framework of industry. It comprises definition, preparation, implementation and presentation phases. An incremental processing is performed like in industrial projects. It means
• a segmentation into specific work packages,

• its division into tasks and subtasks including milestones
• the presentation of intermediate results
• a final report with presentation

Further laboratories related to current research issues in "Future Generation Internet" will be integrated into the course on demand. Details are discussed in the first lecture.

An actual list of studied topics and related references are presented in the first lecture.

The language of the course wil be announced during the first lecture.

Literature:
Foundations:
• J. Liebeherr, M. Elzarki: Mastering Networks, An Internet Lab Manual, Pearson Education, Boston, 2004.

Further references related to specific workpackages:
• Kurose, J., Ross, K.W.: Computer Networking – a Top-Down Approach, Addison-Wesley, 2013 .
• Tanenbaum, A. S.: Computer Networks, Pearson Education, 2010.
• Leon-Garcia, A., Widjaja, I.: Communication Networks, McGraw-Hill, Boston, 2nd ed. 2004.
• Flaig, G., u.a.: Internet-Telefonie, Open source Press, München, 2006.
An up-to-date list is provided by the course.

Examination
Coursework Assignment and Colloquium / Duration of Examination: 30 minutes
Duration of Coursework: 4 months
Description:
The evaluation of the course will take place after completion of all lectures within the examination cycle. It is based on following items:
• assessment of the chapters composed by the candidate in the final course report about all workpackages written by a team of students
• presentation and explanation of specific tasks and outcomes of laboratories by an individual colloquium lasting 30 minutes

The evaluation rules of these components will be announced during the first lecture. The overall individual grading has to reach the level "satisfactory/ ausreichend (4.0)" to pass the examination of the module.

The language of the examination wil be announced during the first lecture.

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 WS20/21) 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 SS25)

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 **Learning outcome:**	**2,00 Weekly Contact Hours**

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

Module NLProc-ANLP-M Applied Natural Language Processing *Angewandte maschinelle Sprachverarbeitung*	6 ECTS / 180 h

(since WS24/25) Person responsible for module: Prof. Dr. Roman Klinger	

Contents:

The module of Applied Natural Language Processing can be filled with various specialized courses from the Chair of Fundamentals of Natural Language Processing. This includes specialized lectures and seminars. The module is worth 6 credit points. Therefore two classes of 3 credit points can be combined or one 6 credit point class can be taken.

Learning outcomes:

The student learns about recent research topics in natural language processing, learns to familiarize themselves with state of the art methods and challenges, and learns to build on knowledge from machine learning, deep learning, structured learning, computational linguistics, and semantics.

prerequisites for the module:

Recommended prior knowledge: • Information Retrieval and Text Mining (recommended) • Algorithmisches Sprachverstehen (recommended) • Deep Learning (very helpful)	Admission requirements: none

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

Module Units	
1. Explainable AI Methods and Applications in Natural Language Processing **Mode of Delivery:** Seminar **Language:** English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**

Learning outcome:

The course aims to examine current research directions on various topics of explainablility in NLP. In particular, we will focus on explaining the inner workings of neural-networks and other models traditionally considered to be "black boxes." Participants will read and discuss research papers, learning to critically question the results presented within, identify gaps in knowledge and develop ideas for further research.

Contents:

Topics include

· Bias and bias detection

· Model attribution analysis

· Interpretability of word embeddings

· Attention-based explainability

· Adversarial attacks and counterfactuals

· Model Debugging and Error Analysis

Literature:

Zhang, Edwin, et al. "Transcendence: Generative Models Can Outperform The Experts That Train Them." *arXiv preprint arXiv:2406.11741* (2024).

Li, Jiaoda, Yifan Hou, and Mrinmaya Sachan Ryan Cotterell. "What Do Language Models Learn in Context? The Structured Task Hypothesis." *arXiv e-prints* (2024): arXiv-2406.

Goldfarb-Tarrant, Seraphina, et al. "Bias beyond English: Counterfactual tests for bias in sentiment analysis in four languages." *arXiv preprint arXiv:2305.11673* (2023).

Marco Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, pages 97–101, San Diego, California. Association for Computational Linguistics.

2. Multi-Modal Language Technology **Mode of Delivery:** Seminar **Language:** English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
3. Argument Mining **Mode of Delivery:** Seminar **Language:** English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
4. Large Language Models for Natural Language Understanding **Mode of Delivery:** Seminar **Language:** English **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
5. Emotion Analysis **Mode of Delivery:** Lectures and Practicals **Lecturers:** Prof. Dr. Roman Klinger **Language:** English **Frequency:** every summer semester	**2,00 Weekly Contact Hours** **3.0 ECTS**

Learning outcome:

This class 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.

The content is:

- Emotion theories
- Appraisal theories
- Lexicons and applications of emotion analysis
- Machine and deep learning for emotion analysis
- Event interpretation
- Structured analyses of emotions

6. Emotion Analysis Project	**2,00 Weekly Contact**
Mode of Delivery:	**Hours**
Lecturers: Prof. Dr. Roman Klinger	**3.0 ECTS**
Language: English/German	
Frequency: every semester	
Learning outcome:	
The student learns to apply emotion analysis methods to new tasks and domains.	
Contents:	
This course extends the Emotion Analysis lecture of 3 credit points to a course of 6 credit points can only be taken together with the lecture. This is not a standard "project" course but an extension of the emotion analysis lecture which can not be taken without the lecture.	
Examination Internship report **Description:** This module can be filled with either a specialized lecture (3 ECTS) together with a short additional project (another 3 ECTS) or with a seminar; or two seminars (each 3 ECTS). Depending on the class type, the exam will take on a different modality. Typically, for seminars it is a presentation and a written paper ("Referat + Hausarbeit") while for the emotion analysis course it is Hausarbeit + Colloquium. Other future classes might decide on a different modality. Students will be informed in the first week of a lecture about that.	

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:

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:

Recommended prior knowledge: 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.	**Admission requirements:** 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:

Recommended prior knowledge: 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).	**Admission requirements:** 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	**2,00 Weekly Contact Hours**

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. 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	6 ECTS / 180 h
Introduction to Differential Privacy	

(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:

Recommended prior knowledge:	**Admission requirements:**
none	none

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

Module Units	
Lecture and Tutorial **Mode of Delivery:** Lectures and Practicals **Language:** English **Frequency:** every winter semester	**4,00 Weekly Contact Hours**
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:

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

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	30 ECTS / 900 h
Master Thesis in Software Systems Science	

(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:	**Admission requirements:**
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.	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 WS24/25)
Person responsible for module: Prof. Dr. Gerald Lüttgen

Contents:

This module focuses on the increasingly important field of automated software verification, which aims at increasing the quality of today's complex computer systems. Students will be introduced to modern automated software verification and, in particular, to software model checking, and will be familiarised with a variety of important formal verification concepts, techniques and algorithms, as well as with state-of-the-art verification tools.

Learning outcomes:

On completion of this module, students will be able to thoroughly analyse software using modern software verification tools and understand the state-of-the-art techniques and algorithms that drive cutting-edge development environments offered by major software companies.

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 lectures (Vorlesungen)
- 30 hrs. attending practicals (Übungen)
- 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 (Hausarbeit)
- 30 hrs. preparing for the colloquium (Kolloquium)

prerequisites for the module:

Recommended prior knowledge: Basic knowledge in algorithms and data structures, mathematical logic and theoretical computer science.	**Admission requirements:** 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 **Frequency:** every summer semester **Contents:** The lectures (Vorlesungen) will address the following topics in automated software verification: (i) state machines, linear-time properties and algorithms for state space exploration; (ii) LTL model checking; (iii) SAT solving and bounded model checking; (iv) decision procedures and SMT solving; (v) software	**2,00 Weekly Contact Hours**

model checking; (vi) predicate abstraction. In addition, state-of-the-art software verification tools will be introduced.

Literature:
- Baier, C., Katoen, J.-P. Principles of Model Checking. MIT Press, 2008.
- Clarke, E., Grumberg, O., Kroening, D., Peled, D. and Veith, H. Model Checking. 3rd. ed. MIT Press, 2018.
- Huth, M. and Ryan, M. Logic in Computer Science. 2nd ed. Cambridge University Press, 2004.
- Kroening, D. and Strichman, O. Decision Procedures: An Algorithmic Point of View. Springer, 2008.
- Loeckx, J. and Sieber, K. The Foundations of Program Verification. 2nd ed. Wiley, 1987.

2. Applied Software Verification **Mode of Delivery:** Practicals **Lecturers:** Scientific Staff Praktische Informatik, insbesondere Softwaretechnik und Programmiersprachen **Language:** English **Frequency:** every summer semester	**2,00 Weekly Contact Hours**
Contents: Students will practice the various theoretical and practical concepts taught in the lectures (Vorlesungen) by applying them to solve verification problems using several modern model-checking tools, and also by engaging in pen-and-paper exercises. Emphasis will be put on presenting and discussing the solutions to the exercises by and among the students, within the timetabled practicals (Übungen).	
Literature: - see the corresponding lectures -	

Examination Coursework Assignment and Colloquium / Duration of Examination: 20 minutes Duration of Coursework: 3 weeks **Description:** Assignment (Hausarbeit) consisting of questions that practice, review and deepen the knowledge transferred in the lectures and practicals (Vorlesungen und Übungen). The assignment is set in English language, while answers may be provided in either English or German. Colloquium (Kolloquium) consisting of questions testing the knowledge transferred in the lectures and practicals (Vorlesungen und Übungen), on the basis of the submitted solutions to the assignment (Hausarbeit). The colloquium can be held electively in English or German language.	

Module SYSNAP-OSE-M Operating Systems Engineering *Operating Systems Engineering*	6 ECTS / 180 h

(since WS24/25)

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:

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 **Mode of Delivery:** Lectures **Lecturers:** Prof. Dr. Michael Engel **Language:** German/English **Frequency:** every summer semester	**2,00 Weekly Contact Hours**

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 **Mode of Delivery:** **Lecturers:** Prof. Dr. Michael Engel **Language:** German/English **Frequency:** every summer semester	**2,00 Weekly Contact Hours**

Learning outcome:

cf. module description

Contents:

cf. module description

Examination

Coursework Assignment and Colloquium / Duration of Examination: 30 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 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	**2,00 Weekly Contact Hours**
Learning outcome: cf. module description	
Contents: cf. module description	
Literature: cf. module description	
Examination Portfolio / Duration of Coursework: 3 months	

Module SYSNAP-Virt-M Virtualization *Virtualisierung*	6 ECTS / 180 h

(since WS24/25)
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:	**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 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 **Mode of Delivery:** **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	

Examination Coursework Assignment and Colloquium / Duration of Examination: 30 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 winter term or at the begin of the summer 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. | |

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	**2,00 Weekly Contact Hours**
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. Deep Learning **Mode of Delivery:** Practicals **Lecturers:** N.N. **Language:** English/German **Frequency:** every winter semester	**2,00 Weekly Contact Hours**
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	

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.	99

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
	A1 Software Systems Science		**36 - 54**		
	Students who started their studies **before summer semester 2021** must complete modules of **30 - 48 ECTS points in A1**.				
	Elective Area: compulsory part		**24**		
	Students, who started their studies before winter semester 2019/20, do not have a compulsory area. The four modules listed here can be taken as electives instead.				
	All other students: Pleae note that the module **SWT-PCC-B** is no longer offered as of the winter semester 2020/21. It was first replaced by the module **SWT-CPS-M** until summer semester 2023 and from winter semester 2023/24 on by the module **SWT-SWQ-M**. As of winter semester 2024/25 SWT-SWQ-M can no longer be offered and is replaced by **GdI-IFP-M.**				
DSG-DSAM-M	Distributed Systems Architectures and Middleware	every winter semester	6	2 Lectures 2 Practicals	Coursework Assignment and Colloquium 3 months 15 minutes
GdI-IFP-M	Introduction to Functional Programming	every winter semester	6	2 Lectures 2 Practicals	Written examination 90 minutes
KTR-GIK-M	Foundations of Internet Communication	every summer semester(on demand also WS)	6	4 Lectures and Practicals	Coursework Assignment and Colloquium 4 months 30 minutes
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
	Elective Area: elective modules		**12 - 30**		
	Students, who started their studies **before winter semester 2019/20**, must complete **30-48 ECTS points** in this area.				
AISE-Auto	Automation of First- and Higher-Order Logic	every summer semester(1)	6	6 Lectures and Practicals	Oral examination 30 minutes

Module Handbook Summary

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	Algorithms	alle 4 Semester(1)	6	4 Lectures and Practicals	Sonstiges
DSG-DistrSys-M	Distributed Systems	every summer semester(2020)	6	2 Lectures 2 Practicals	Coursework Assignment and Colloquium 3 months 15 minutes
DSG-SOA-M	Service-Oriented Architecture and Web Services	every summer semester	6	2 Lectures 2 Practicals	Coursework Assignment and Colloquium 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
DT-DBCPU-M	Database Systems for modern CPU	every summer semester(1)	6	4 Lectures and Practicals	Written examination 90 minutes
GdI-FPRS-M	Functional Programming of Reactive Systems	every summer semester	6	2 Lectures 2 Practicals	Written examination 90 minutes Oral examination 30 minutes
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	every winter semester(1)	6	4 Lectures and Practicals	90 minutes

Module Handbook Summary

SWT-ASV-M	Applied Software Verification	every summer semester	6	2 Lectures 2 Practicals	Coursework Assignment and Colloquium 3 weeks 20 minutes
SYSNAP-OSE-M	Operating Systems Engineering	every summer semester(1)	6	2 Lectures 2	Coursework Assignment and Colloquium 3 months 30 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	Coursework Assignment and Colloquium 3 months 30 minutes

Module Handbook Summary

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
	A2 Domain-specific Software Systems Science		**0 - 18**		
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
GAMES-DGS-M	Designing Gamified Systems	every summer semester(2025)	6	4	Coursework Assignment and Colloquium
GdI-CSNL-M	Computational Semantics of Natural Language	every summer semester(1)	6	4 Lectures and Practicals	Portfolio 80 minutes
HCI-MCI-M	Human-Computer Interaction	every winter semester	6	2 Lectures 2 Practicals	Oral examination 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
NLProc-ANLP-M	Applied Natural Language Processing	every semester(1)	6	2 Seminar 2 Seminar 2 Seminar	Internship report

				2 Seminar	
				2 Lectures and Practicals	
				2	
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**		
	In A3 students who started their studies **before summer semester 2021** must complete **12 ECTS points** in this area.				
	Elective Area: Seminar		**3**		
ISoSySc-Sem-M	Master's Seminar in Software System Science	every semester	3	2 Seminar	Coursework Assignment with presentation
	Elective Area: Project		**6**		

Students, who started their studies **before SS21**, must complete one project of **9 ECTS** points in this area. Deviating from the modules listed below one of the following modules must be completed:

- DSG-Project-M
- KTR-SSSProj-M
- MOBI-PRS-M
- PSI-ProjectCAD-M
- SWT-PR2-M

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
ISoSySc-Proj-M	Master's Project in Software System Science	every semester	6	4	Coursework Assignment and Colloquium

Module Handbook Summary

ID	Module	Semester	ECTS	Weekly Contact Hours	Examination
	A4: Masters 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		**27**		

In A5 Students who started their studies **before summer semester 2021** can complete modules of **up to 30 ECTS points**.

According to the examination regulations (StuFPO) Appendix 1, students have **four options** regarding the areas from which the ECTS points can be acquired (independent from their enrollment semester):

(1) to study modules of software systems science at a university abroad for at least one semester or

(2) to accomplish a traineeship in an international context, preferentially abroad, that covers topics of the occupational field of software systems science with a volume of at least 360 working hours (12 ECTS credits).

(3) to accomplish credits in modules of foreign languages (neither English nor native language)

(4) to accomplish *further* modules of module groups A1 and A2 (Examination Regulations, App. 1).

	Area: Guided graduate study abroad		**0 - 27**		

Students who started their studies **before summer semester 2021**, can accomplish modules of up to **30 ECTS** points in this area.

The courses that are selected at a foreign university have to be approved by learning agreements. For own planning security reasons, learning agreements have to be signed by those Professors at University of Bamberg responsible for the chosen subject, as well as the head of the Examination Board, before the graduate study abroad is initiated.

	Area: Internship in an International context		**0 - 12**		

Regarding the elective area 5b, *Internship in an international context*, with an equivalent workload of 12 ECTS credits, a foreign or internationally acting domestic company (or research institute) may be selected.

It has to offer a specific internship related to relevant topics of software systems science. The documentation of the internship requires the delivery of the following items to the degree programme representative:

- written report of 4 pages at least, reporting on the tasks and achievements, and
- a certificate issued by the hosting institution or the organizational unit that has realized the internship.

SSS-PraktIntKon-M	Internship in an International Context	every semester(1)	12		Writen Report on Practical Training

	Area: Foreign languages		**0 - 15**		

In the elective area 5c, *Foreign languages*, students take modules offered by the University's Language Centre.

Excluded are modules of the English language and modules of the language in which the university entrance qualification was obtained.

Details, in particular the modules available for selection and the respective Module examinations are described (in German) in the *Modulhandbuch des Sprachenzentrums der Otto-Friedrich-Universität Bamberg.*

Students who started their studies **before summer semester 2021**, can accomplish modules of up to **18 ECTS** points in this area.

Area: further modules from module groups A1 and/ or A2   0 - 27

In this area additional, not previously completed modules from A1 or A2 module groups' required elective options can be completed in accordance with the Examination Regulations, Appendix 1.

Students who started their studies **before summer semester 2021** can complete modules of **up to 30 ECTS points** in this area.