UNIVERSITY OF PADERBORN

FACULTY FOR COMPUTER SCIENCE, ELECTRICAL ENGINEERING AND MATHEMATICS
DEPARTMENT OF COMPUTER SCIENCE

MODULE HANDBOOK
MASTER PROGRAM COMPUTER SCIENCE V4 (IMA V4), ENGLISH

DATE: 3. JUNI 2024

Inhaltsverzeichnis

1 Präambel und Hinweise

Aus technischen Gründen wurde die Präambel des Modulhandbuches ausgelagert. Sie ist unter Modulhandbuch Informatik auf den Seiten zum Studium des Instituts für Informatik zu finden. Wir bitten um Beachtung dieser Präambel. Bei Fragen zu dieser Präambel wenden Sie sich bitte an die Fachberatung Informatik.

Bitte beachten Sie auch, dass

1. in diesem Modulhandbuch alle laut Prüfungsordnung vorgesehenen Module aufgelistet werden, auch wenn sie in dem entsprechenden Semester nicht angeboten werden.
2. dieses Modulhandbuch den Datenbestand zum Zeitpunkt der Erstellung beinhaltet.

2 Pflichtmodule

<table>
<tr><th colspan="5">Master Thesis</th></tr>
<tr><td colspan="5">Master Thesis</td></tr>
<tr><td colspan="2">Module number:
A.079.4010</td><td>Workload (h):
900</td><td>Credits:
30</td><td>Regular Cycle:
summer- / winter term</td></tr>
<tr><td colspan="2"></td><td>Semester number:
4</td><td>Duration (in sem.):
1</td><td>Teaching Language:
en</td></tr>
<tr><td>1</td><td colspan="4">Module structure:
<table>
<tr><th></th><th>Course</th><th>form of teachin</th><th>contact-time (h)</th><th>self-study (h)</th><th>status (C/CE)</th><th>group size (TN)</th></tr>
<tr><td>a)</td><td>2024.7090a
Master Thesis – Work Plan</td><td></td><td>30</td><td>120</td><td>C</td><td>1</td></tr>
<tr><td>b)</td><td>2024.7090b
Master Thesis</td><td></td><td>30</td><td>720</td><td>C</td><td>1</td></tr>
</table></td></tr>
<tr><td>2</td><td colspan="4">Options within the module:
none</td></tr>
<tr><td>3</td><td colspan="4">Admission requirements:
Module examinations of at least 48 credits must have been successfully completed. If the student wishes to specialize, at least three modules in the specialization area must be successfully completed and the topic of the Master Thesis must be chosen within the specialization area.</td></tr>
</table>

4	**Contents:** *Contents of the course Master Thesis – Work Plan:* After agreeing on the topic with the supervisor, an initial rough draft is made. On this basis and after a first literature research, the student has to submit a work plan documenting the results to be achieved including milestones for the work. The description of the task and objective to be worked on should not exceed five (in exceptions up to ten) pages and should be submitted to the supervisor within one month after the issue of the topic. The introductory presentation should at least present the planned work, the planned procedure and the hoped-for results and should have a duration of approx. 30 minutes. The introductory presentation is followed by a discussion. *Contents of the course Master Thesis:* The master's thesis comprises the work on a topic with written elaboration and an oral presentation of the results. In the master's thesis, the student demonstrates their ability to work independently on an appropriately challenging topic that also gives the opportunity to develop his or her own ideas. On the basis of the "state-of-the-art", the methods of computer science are to be applied systematically. The task of a Master's thesis may include, for example, the development of software, hardware, a proof or a literature research. The master thesis should not exceed 120 pages. The results of the master's thesis are to be presented in a concluding presentation open to the faculty, followed by a scientific debate. This will take place after the written thesis has been submitted (usually after four weeks at the latest) and should have a duration of approx. 30 minutes. The concluding presentation may include further elements (e.g. demonstration of software) in addition to lecture and debate. The concluding presentation will be included in the final grade. Concrete tasks for master theses are continuously published on the websites of the departments of the Institute of Computer Science.
5	**Learning outcomes and competences:** Within the scope of their Master's thesis, students work on a problem according to scientific methods within a certain period of time. The technical-methodical as well as interdisciplinary competences acquired in the course of the studies are to be applied accordingly. This includes in particular the structuring and planning of the individual work steps as well as the presentation of the results after completion of the thesis. The Master's thesis does not only deepen the specialized knowledge in a specific subject area of computer science, but above all, important "soft skills" are practiced, which are essential for later professional practice. Students will be able to • work independently on a problem from the field of computer science within a given period of time under professional and methodological supervision using scientific methods, • apply the interdisciplinary competences acquired in the course of their studies to solve complex problems, • plan and carry out the independent processing of an extensive task, • clarify requirements and general conditions of an extensive task with supervisors, • use their own creativity to solve problems, • independently and purposefully develop specialized knowledge and methods to solve subproblems, • present the results orally and in writing in the form of a scientific paper.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a) - b)	Master Thesis with Presentation	30-120 page-sand approx. 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Work plan and introductory presentation	up to 5 pages and 30 min	QP
b)	none		

The respective lecturer will specify the manner in which the course achievement will be conducted at the latest when the topic is handed out.

8 **Prerequisites for participation in examinations:**

Proof of the qualified participation is required for the final module examination.

9 **Prerequisites for assigning credits:**

Credits are awarded when the master thesis, including the final presentation with discussion, has been passed.

10 **Weighing for overall grade:**

The module is weighted with 50 credits (factor 5/3).

11 **Reuse in degree courses or degree course versions :**

keine

12 **Module coordinator:**

Prof. Dr.-Ing. Juraj Somorovsky

13 **Other Notes:**

Remarks of course Master Thesis – Work Plan:
Implementation method
In agreement with supervisor.
Learning Material, Literature
Depending on the topic.

Remarks of course Master Thesis:
Implementation method
Independent studies supported by individual advice and supervision
Learning Material, Literature
Depending on the thesis topic.

Project Group

Project Group			
Module number: M.079.4201	**Workload (h):** 600	**Credits:** 20	**Regular Cycle:** summer- / winter term
	Semester number: 2-3	**Duration (in sem.):** 2	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7091 Project Group	PG	240	360	C	16

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Project Group:
Recommended Proficiencies
Depending on the topic.

4 **Contents:**

Contents of the course Project Group:
In a project group, a group of usually 8-16 students works together over a period of one year (two semesters) on a research topic determined by the group organizer. Project groups introduce students to current research topics that are usually related to the group organizer's special area of interest, and the team working of the project group should be a preparation for industrial practice. Topics of project groups cover the whole range of research interests of the research groups in the Department of Computer Science.
Typically, the project group members are divided into subteams. The team selects a project group leader who controls the work of the whole team. The leader is responsible for internal communication and reporting potential issues to the group organizer. The team members meet regularly to discuss their progress with the group organizer and report on their current progress status. The team members successfully pass this module after submitting the source code and the group report, and providing the final presentation. Further constraints can be defined by the group organizer.

5 **Learning outcomes and competences:**

In project groups, participating students gain first-hand practical experience in working in a team and organizing a project; in doing so, they become prepared for daily work in their later professions. The students personally experience how to carry out extensive development processes in a team. Since the tasks are divided among the individual team members, the participating students become skilled in reporting their progress and research findings to the other group members.
Non-cognitive Skills

- Commitment
- Team work
- Learning competence
- Learning motivation
- Motivation
- Literacy (scientific)
- Self-monitoring

6 **Assessments:**

□Final module exam (MAP) □Module exam (MP) ⊠Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Software with documentation, presentation		75%, 25%

In the project group module, regular reports on the progress of work have to be prepared. In addition, the successful completion of projects must be demonstrated by submitting software and documentation. The results of the project work are to be demonstrated in a presentation. A grade is awarded for the entirety of the partial achievements. The software projects with documentation form 75% of the module grade, the presentation forms 25% of the module grade.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Progress reports or presentations		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted as 10 credits (factor 0.5).

11 **Reuse in degree courses or degree course versions :**

keine

12 **Module coordinator:**

Prof. Dr.-Ing. Juraj Somorovsky

13	**Other Notes:** *Remarks of course Project Group:* **Implementation method** • Developing knowledge on the selected systematic approaches, methods and tools relevant to the research topic, usually done in an introductory seminar phase. • Logical assigning "jobs" (assigning responsibilities to the individual group members). • Discovering and promoting the participants' special individual talents, which are either already apparent or which can be developed throughout the project - such as through seminar presentations or appropriate job assignments. • Setting up a process-oriented personnel structure, similar to the structure of an industrial design team; delegating subtasks to smaller subgroups who report their findings. • Regular progress reports made by individuals and subgroups. • Writing a highly distributed interim report and final report. **Learning Material, Literature** Depending on the topic.

Key Skills			
Key Skills			
Module number: M.079.4202	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer- / winter term
	Semester number: 2	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7092 Scientific work	L1	15	15	C	150
b)	2024.7092b Seminar	S2	30	120	C	15

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Seminar:
Recommended Proficiencies
Depending on the seminar topic.

4 **Contents:**

Contents of the course Wissenschaftliches Arbeiten:
The contents of the module are divided into four parts. First, the students learn how to work with scientific literature; for example, they are introduced to methods and tools for searching related works and managing references. Second, the students learn how to write scientific papers. This includes typical mistakes and best practices for structuring papers. Third, the students learn how to review scientific papers and give feedback to their peers to improve their work. Finally, the students learn how to create and deliver presentations. After every part, the students write a short test.

Contents of the course Seminar:
In seminars, students work independently on an individual research topic by using background literature from various sources. They describe their research topic in a presentation followed by discussion and a written report. The presentation material and the written report serve two different purposes: Whereas the presentation material supports the lectures (held within a specific time period), the written report provides students the opportunity to acquire detailed information on the reported topic at a later date.
Seminars usually consist of 8 to 15 related subtopics, each of which is researched by one participating students. Seminar topics cover the whole spectrum of research topics of the research groups in the Department of Computer Science.

5 **Learning outcomes and competences:**

Students will be able to

- independently develop a research-related topic area in computer science at a scientific level, in particular also through thorough literature research, which includes the study of scientific publications,
- explain concepts and facts relevant to the chosen topic area,
- select appropriate sources and handle literature appropriately
- sift and understand content from a variety of sources and synthesize it into an overall picture,
- present acquired knowledge in the form of elaborations in scientific style and in the form of a scientific presentation, prioritizing content
- structure a presentation along a content line and use various means to illustrate complex issues,
- discuss scientific topics with others
- reflect on and communicate their own working methods,
- absorb knowledge as an audience member from a lecture and exchange opinions and information in discussions.

6 **Assessments:**

□Final module exam (MAP) □Module exam (MP) ⊠Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)			
b)	Presentation with discussion, seminar paper	30-45 minutes and 15-30 pages	40%, 60%

7	**Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Short written exam	max. 30 min	QP
b)	none		

8	**Prerequisites for participation in examinations:** none
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** keine
12	**Module coordinator:** Prof. Dr.-Ing. Juraj Somorovsky
13	**Other Notes:** *Remarks of course Seminar:* **Implementation method** Seminar paper and presentation **Learning Material, Literature** Depending on the seminar topic.

General Studies			
General Studies			
Module number: 2024.7093	**Workload (h):** 300	**Credits:** 10	**Regular Cycle:** summer- / winter term
	Semester number: 2-3	**Duration (in sem.):** 2	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7093 General Studies – Master	diverse	150	150	C	100

2	**Options within the module:** Any courses outside of computer science can be chosen.
3	**Admission requirements:** none
4	**Contents:** Any combination of courses outside of computer science with the amount of 10 ECTS must be chosen. *Contents of the course Studium Generale – Master:* Depending on the selected courses
5	**Learning outcomes and competences:** Students expand their scientific horizon beyond the boundaries of computer science. Depending on the chosen course, they will have acquired skills in communication skills, teamwork and presentation techniques. Students will be able to • acquire knowledge and skills that are not specific to a particular field, but which may be important for the desired career, such as special knowledge of foreign languages, engineering, natural sciences, cultural studies or economics, • analyze a wide variety of issues in the relevant fields, • integrate subject-specific knowledge into an interdisciplinary context, • establish connections to the study of computer science, • expand their key competencies and, if applicable, foreign language competencies, which supports the formation of personality, also in intercultural terms.
6	**Assessments:**
7	**Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Qualified participation within general studies		QP

8	**Prerequisites for participation in examinations:** none
9	**Prerequisites for assigning credits:** The credit points are awarded after the qualified participation was completed.
10	**Weighing for overall grade:** The module is ungraded.
11	**Reuse in degree courses or degree course versions :** keine

12	**Module coordinator:** Prof. Dr.-Ing. Juraj Somorovsky
13	**Other Notes:** none

3 Wahlpflichtmodule

<table>
<tr><th colspan="5">Advanced Algorithms</th></tr>
<tr><td colspan="5">Advanced Algorithms</td></tr>
<tr><td colspan="2">Module number:
M.079.4002</td><td>Workload (h):
180</td><td>Credits:
6</td><td>Regular Cycle:
summer term</td></tr>
<tr><td colspan="2"></td><td>Semester number:
1-3</td><td>Duration (in sem.):
1</td><td>Teaching Language:
en</td></tr>
<tr><td>1</td><td colspan="4">Module structure:
<table>
<tr><th></th><th>Course</th><th>form of teachin</th><th>contact-time (h)</th><th>self-study (h)</th><th>status (C/CE)</th><th>group size (TN)</th></tr>
<tr><td>a)</td><td>2024.7011
Advanced Algorithms</td><td>L3
Ex2</td><td>75</td><td>105</td><td>C</td><td>70/35</td></tr>
</table></td></tr>
<tr><td>2</td><td colspan="4">Options within the module:
none</td></tr>
<tr><td>3</td><td colspan="4">Admission requirements:
none

Prerequisites of course Advanced Algorithms:
Recommended Proficiencies
Willingness and ability to learn the creative process of algorithm design and efficiency analysis using mathematical methods. Basic Knowledge of some basic algorithms and data structures and their analyses is assumed.</td></tr>
<tr><td>4</td><td colspan="4">Contents:

Contents of the course Advanced Algorithms:
This course presents advanced algorithms and algorithmic paradigms for fundamental problems. More precisely, methods like randomization and derandomization as well as concepts for approximation and online algorithms will be presented by illustrating their usefulness for important algorithmic problems. In all cases, the correctness and runtime will be rigorously analyzed.

• Randomized algorithmens and derandomization, for example, randomized rounding
• Online algorithms, for example, scheduling algorithms
• Approximation algorithms, for example, NP-hard problems</td></tr>
</table>

5 **Learning outcomes and competences:**

Students will be able to

- understand and apply basic analytical techniques,
- explain and apply basic algorithmic approaches,
- judge which effects these approaches have, and
- know the limits of using these approaches.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Prof. Dr. Christian Scheideler

13 **Other Notes:**

Remarks of course Advanced Algorithms:
Implementation Method
The lecture uses a blackboard and slides as well as small exercises for the students during the lecture. It will be supported by tutorial groups. Students have the opportunity in tutorial groups to work on problems in a group and to discuss solutions of the exercise sheets with the tutors.
Learning Material, Literature

- Slides of the lecture; exercise sheets
- Additional literature will be announced in the course

Advanced Complexity Theory

Advanced Complexity Theory

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4004	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	2024.7018 Advanced Complexity Theory	L3 Ex2	75	105	C	25

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Advanced Complexity Theory:
Recommended Proficiencies
Basic knowledge about complexity theory (e.g., Turing machines, NP-completeness)

4 **Contents:**

Contents of the course Advanced Complexity Theory:
Complexity Theory deals with determining the amount of resources (e.g. runtime, memory consumption) necessary and sufficient for solving a given algorithmic problem (e.g. Travelling Salesperson Problem (TSP)) on a given machine model (e.g. Turing machine). One approach is to define complexity classes like P, NP, PSPACE, in order to classify problem complexity by means of completeness in such classes, like the famous class of NP-complete problems. This gives conditional results like “If NP is not equal P, then TSP is not solvable in polynomial time.” This branch of Complexity Theory is often referred to as Structural Complexity Theory. In contrast, proving explicit lower bounds for given problems is the topic of the so-called Concrete Complexity Theory. As nobody is currently able to prove superlinear time bounds for explicitly defined problems on general computation models like Turing machines, one considers somewhat restricted models like 1-tape Turing machines, monotone Boolean circuits, Boolean circuits with bounded depth, algebraic computation models, and several kinds of parallel computation models. This lecture surveys approaches to prove such lower bound on various such models.

- Deterministic, non-deterministic and probabilistic time and space complexity classes, hierarchies, completeness
- Lower bounds for size and depth of different variants of Boolean circuits
- Lower bounds for algebraic computations

5 **Learning outcomes and competences:**

Students will be able to

- Learning outcome 1
- . . .
- 4 to 6 learning outcomes

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

keine

12 **Module coordinator:**

Prof. Dr. Johannes Blömer

13 **Other Notes:**

Remarks of course Advanced Complexity Theory:
Implementation method
Text of about 200 to 500 characters
Learning Material, Literature

- C.H. Papadimitiriou, Computational Complexity, Addison-Wesley
- S. Arora, B. Barak, Computational Complexity - A Modern Approach, Cambridge University Press
- Slides of the lecture, exercise sheets

Advanced Computer Architecture

Advanced Computer Architecture

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4005	180	6	winter term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	2024.7031 Advanced Computer Architecture	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Advanced Computer Architecture:
Recommended Proficiencies
Knowledge from the Bachelor course Computer Architecture is helpful.

4 **Contents:**

Contents of the course Advanced Computer Architecture:
The course teaches the essential concepts and methods used in the design of modern processors. In particular, advanced aspects of optimizing access times and throughput in the memory hierarchy, as well as approaches to exploiting parallelism at the instruction, data, and thread levels are discussed. The course covers the following topics:

- Fundamentals of computer architectures (refresher)
- Memory hierarchy design
- Instruction-level parallelism
- Data-level parallelism: Vector, SIMD and GPU architectures
- Thread-level parallelism
- Warehouse-scale computer
- Domain-specific computer architectures

5 **Learning outcomes and competences:**

Upon completion of this module, students will be able to

- explain the architecture of modern multi-level storage systems, mathematically model the average access time, and qualitatively describe and evaluate the influence of the main design parameters,
- explain the concepts of parallel processing at the data, instruction, thread and task levels and contrast algorithms for out-of-order execution,
- examine the limits of computing power for specific applications and architectures using the Roofline model,
- explain the common approaches and protocols for cache coherence in multiprocessor systems and demonstrate how they work with examples, and
- quantitatively evaluate different characteristics of modern computer systems through computer simulation and interpret the results.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Prof. Dr. Christian Plessl, Prof. Dr. Marco Platzner

13	**Other Notes:** *Remarks of course Advanced Computer Architecture:* **Implementation method** The course consists of a lecture and paper&pencil as well as practical exercises. The lecture is held with a beamer and blackboard. In the paper&pencil exercises, assignments are handed out and their solutions are presented and discussed in an exercise session. In the practical exercises, the effects of design decisions and optimisation options at the hardware and software level are examined and deepened on the computer with simulators of processor and memory systems using case studies. **Learning Material, Literature** • Lecture slides and exercise sheets • Exercise sheets and technical documentation for the for the computer-based exercises • Hennessey, Patterson: Computer Architecture: A Quantitative Approach (6th edition), Morgan Kaufmann, 2017. • Information about alternative and additional literature as well as teaching material on the course's website and in the lecture slides

Advanced Distributed Algorithms and Data Structures

Advanced Distributed Algorithms and Data Structures

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4006	180	6	winter term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7012 Advanced Distributed Algorithms and Data Structures	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Advanced Distributed Algorithms and Data Structures:
Recommended Proficiencies
Algorithms and data structures, distributed algorithms and data structures

4 **Contents:**

Contents of the course Advanced Distributed Algorithms and Data Structures:
After a short introduction of the foundations of graph and network theory as well as distributed programs, the lecture presents advanced methods in the area of distributed algorithms and data structures. Topics covered in the course are access control, synchronization, consensus, information dissemination, hybrid networks, scheduling, and optimization. In addition to presenting solutions to these topics, also concrete applications will be presented.
The lecture gives an introduction to state-of-the-art advanced distributed algorithms and data structures. In addition to the presentation of the corresponding protocols, their correctness and efficienty will be shown in a rigorous way. The lecture is structured as follows:

- Introduction
- Fooundations of graph and network theory
- Access control
- Synchronization
- Consensus
- Information dissemination
- Hybrid networks
- Scheduling
- Optimization

In addition to presenting solution to these topics, also concrete applications will be presented.

5 **Learning outcomes and competences:**

Students will be able to

- understand and apply basic analytical techniques,
- explain and use basic algorithmic approaches,
- judge which effects these approaches have, and
- know the limits of using these approaches.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Christian Scheideler
13	**Other Notes:** *Remarks of course Advanced Distributed Algorithms and Data Structures:* **Implementation Method** The lecture uses a blackboard and slides as well as small exercises for the students during the lecture. It will be supported by tutorial groups. Students have the opportunity in tutorial groups to work on problems in a group and to discuss solutions of the exercise sheets with the tutors. **Learning Material, Literature** • Slides of the lecture; exercise sheets • Additional literature will be announced in the course

Advanced Networked Systems

Advanced Networked Systems

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4096	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7035 Advanced Networked Systems	L2 Ex3	75	105	C	50/25

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Advanced Networked Systems:
Recommended Proficiencies
Knowledge of computer networks, operating systems, programming languages, C/C++ and Python programming in the Linux environment, and a keen interest in understanding how things work under the hood. Ability to read scientific papers professionally. Ability to code in a complex setting.

4 **Contents:**

Contents of the course Advanced Networked Systems:
The course will cover concepts and designs for modern networked systems adopted by the Internet and cloud data centers to meet the ever-increasing demands of data transfer and computation driven by big data and machine learning applications.

- Networking fundamentals (refresher)
- Data center networks (architectures, congestion control)
- Software-defined networks (SDN, OpenFlow)
- Programmable networks (P4, eBPF/XDP)
- Programmable network device architectures (RMT, SmartNICs)
- In-network computing (caching, aggregation)

5 **Learning outcomes and competences:**

Upon completion of this course, students will be able to

- gain knowledge of current research topics in networked systems.
- understand the design of these new networked systems technologies and reason about the design choices therein.
- build complex networked systems by applying some of these designs, analyze and evaluate the merits and limitations of these designs, and explain the design choices for the built systems.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** keine
12	**Module coordinator:** Prof. Dr. Lin Wang
13	**Other Notes:** *Remarks of course Advanced Networked Systems:* **Implementation Method** The course content will be taught with slides-based lectures, interactive exercises, and programming-based project assignments. Much of the course will be based on discussions of cutting-edge research topics, complemented with hands-on programming assignments. **Learning Material, Literature** • Lecture slides and exercise sheets • Project description documents for the programming assignments • Additional literature (e.g., research papers) on the course website and in the lecture slides

Algorithms for Complex Virtual Scenes

Algorithms for Complex Virtual Scenes

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4009	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7013 Algorithms for Complex Virtual Scenes	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3	**Admission requirements:** none *Prerequisites of course Algorithms for Complex Virtual Scenes:* **Recommended Proficiencies** Willingness and ability to learn the creative process of algorithm design and efficiency analysis using mathematical methods. Basic Knowledge of some basic algorithms and data structures and their analyses is assumed.
4	**Contents:** *Contents of the course Algorithms for Complex Virtual Scenes:* Walkthrough systems allow viewing and walking through a virtual 3D scene and finds application in architecture programs, simulations or games. The efficiency of real-time rendering algorithms is crucial for a smooth and fast rendering of the virtual 3D scene in a walkthrough system. There are different algorithmic approaches to reduce highly complex 3D geometric data and to achieve a rendering of the scene in real time. The lecture introduces different algorithmic approaches, e.g., visibility culling, simplification, level of detail, image-based rendering. The course includes the following contents: • Walkthrough problem • Spatial Data structures: kd-tree, BSP-tree, octree, loose octree • Level of detail: adaptive LOD management, mesh simplification, progressive meshes • Visibility culling: view frustum culling, potentially visible sets (PVS), dynamic analysis of PVS, hierarchical z-buffer, hierarchical occlusion maps, coherent hierarchical culling, aspect graph, visibility space partition • Replacement: color cubes, randomized z-buffer, hierarchical image caching • Parallel rendering: classification and modeling, parallel rendering as a sorting problem, hybrid sort-first/sort-last rendering
5	**Learning outcomes and competences:** Students will be able to • name, explain and apply basic algorithms and data structures for problems in the rendering of complex virtual scenes. • identify basic algorithmic problems in application problems of the rendering of complex virtual scenes and select suitable algorithms and data structures for them • analyze, compare and investigate runtime and memory estimation of spatial data structures and algorithms • evaluate what impact the choice of spatial data structures has on the efficiency of algorithms for rendering complex virtual scenes • develop own efficient visibility algorithms based on spatial data structures for other virtual scenes with special characteristics • develop own efficient approximation algorithms based on spatial data structures for other virtual scenes with special characteristics • discuss problems of the rendering of complex virtual scenes and corresponding solution proposals with experts in the field

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Dr. Matthias Fischer

13	**Other Notes:** *Remarks of course Algorithms for Complex Virtual Scenes:* **Implementation Method** The contents are taught by means of a presentation within the framework of a lecture. The lecture is usually held with beamer and blackboard. In exercises and assignments, design and analysis of algorithms are presented on selected examples and are deepened and developed by the students themselves in tutorials in small groups as well as in self study and supplemented by practical exercises. Sample solutions of exercise sheets are presented in central exercises. The expected activities of the students are participation in presence exercises and independent work on assignments. **Learning Material** • Lecture slides, exercise sheets, sample solutions if applicable, lecture recordings from previous years, blackboard transcription **Literature** • Real-Time Rendering; Tomas Akenine-Möller, Eric Haines; AK Peters, 2002. • Level of Detail for 3D Graphics; David Luebke, Martin Reddy, Jonathan D. Cohen; Morgan Kaufmann Publishers, 2002. • Algorithmen in der Computergraphik; Thomas Rauber; Teubner, 1993. • Wavelets for Computer Graphics: Theory and Applications; Eric Stollnitz, David H. Salesin, Anthony D. DeRose; Morgan Kaufmann Publishers, 1996. • Graphic Gems; Andrew S. Glassner; Academic Press; 1990. • Game Programming Gems; Mark DeLoura; Charles River Media; 2000. Computational Geometry • Computational Geometry - Algorithms and Applications; Mark de Berg, Marc de Kreveld, Mark Overmars; Springer Verlag, 2000. • Computational Geometry in C; Joseph O'Rourke; Cambridge University Press, 1998. • Algorithmic Geometry; Jean-Daniel Boissonnat, Herve Bronniman;Cambridge University Press, 1998. • Algorithmische Geometrie Grundlagen, Methoden, Anwendungen; Rolf Klein; Springer Verlag, 2005. General principles of computer graphics • 3D Computer Graphics; Alan Watt; Addison Wesley, 1999. • Computer Graphics, Principles and Practice; James Foley, Andries van Dam, Steven Feiner, John Hughes; Addison Wesley, 1995. • Computer Graphics; Donald Hearn, M. P. Baker; Prentice Hall, 2003. Additional literature will be announced in the course.

Concepts of Computer Science			
Concepts of Computer Science			
Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4203	180	6	winter term

<table>
<tr><td colspan="2"></td><td>Semester number:
1-3</td><td>Duration (in sem.):
1</td><td>Teaching Language:
en</td></tr>
<tr><td>1</td><td colspan="4">Module structure:
<table>
<tr><th></th><th>Course</th><th>form of teachin</th><th>contact-time (h)</th><th>self-study (h)</th><th>status (C/CE)</th><th>group size (TN)</th></tr>
<tr><td>a)</td><td>2024.7056 Concepts of Computer Science</td><td>L2 Ex3</td><td>75</td><td>105</td><td>C</td><td>70/35</td></tr>
</table></td></tr>
<tr><td>2</td><td colspan="4">Options within the module:
none</td></tr>
<tr><td>3</td><td colspan="4">Admission requirements:
none
Prerequisites of course Concepts of Computer Science:
Recommended Proficiencies
• Good general education
• Ability to read and analyze longer and complex texts from computer science as well as the humanities</td></tr>
</table>

4 **Contents:**

Contents of the course Concepts of Computer Science:
The course examines the interplay of digital artifacts and cognitive performance. For this purpose, the epistemological and methodological foundations will be developed that allow the diverse relationships between computer science systems and their application to be addressed in such a way that the computer science-specific consequences become apparent. In the class, these concepts will be systematically analyzed and evaluated historically, technically, and with respect to their potentials.
The lecture discusses the relevant theoretical and conceptual foundations of computer science. Special attention will be paid to differentiate between technical concepts and the sphere of usage. Against this background, theories of interactive systems will be explored in order to examine which role technical artifacts play with respect to processes of the mind. When developing computer systems, relevant data and processes need to be anticipated to a certain degree and modeled as formal systems. This raises issues like the question under which conditions such a formal description can be made in an adequate way and with which consequences regarding the reliability and responsible use of computer systems.
The course includes the following contents:

- Basic concepts of computer science
- Historical background of developments in computer science
- Digital media and mental processes
- Theories of digital media and interactive systems
- Paradigms of support and replacement of mental processes
- Modeling and formalization of data and processes
- Trustworthiness of systems

5 **Learning outcomes and competences:**

Students will be able to

- examine the characteristics of computer systems in a theory-based manner,
- explain cognitive-psychological, sociological and system-theoretical basics of computer science
- differentiate technical and non-technical issues and relate them adequately to each other,
- evaluate and compare current technological developments
- assess innovation potentials in the field of digital technologies,
- weigh risks and potentials for successful use of information technology systems.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7	**Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Assignments, course paper or progress reports		CA

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Dr. Harald Selke
13	**Other Notes:** *Remarks of course Concepts of Computer Science:* **Implementation method** The lecture follows a flipped classroom concept in which students learn about topics based on their reading of scientific literature as well as individual research. They will then present these topics in short presentations in the tutorials in the style of a mini-seminar. Building on this, the lecture then conveys connections between the literature covered in the tutorials and adds further facets. **Learning Material, Literature** • Lecture slides • Wardrip-Fruin, N.; Montfort, N. (eds.): The New Media Reader. Cambridge, Ma.: MIT Press, 2003. • Additional scientific literature will be announced in the lectures.

Data-driven Engineering			
Data-driven Engineering			
Module number: M.079.4204	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** winter term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.705b Data-driven Engineering	L2 Ex3	75	105	C	60/30

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Data-driven Engineering:
Recommended Proficiencies
Knowledge of contents from the module *name of module or modules* is beneficial.

4 **Contents:**

Contents of the course Data-driven Engineering:
Description of the content in English as a text of about 500 to 1000 characters
The course includes the following contents:

- First
- . . .
- 6 to 12 bullet points

5 **Learning outcomes and competences:**

Students will be able to

- Learning outcome 1
- . . .
- 4 to 6 learning outcomes

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	Type of examination	Duration or scope	Weighting for the module grade
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

none

8 **Prerequisites for participation in examinations:**

none

9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** keine
12	**Module coordinator:** Prof. Dr.-Ing. Roman Dumitrescu
13	**Other Notes:** *Remarks of course Data-driven Engineering:* **Implementation Method** Text of about 200 to 500 characters **Learning Material, Literature** • Literature 1 • Literature 2 • . . . • Additional literature will be announced in the course.

Data-Driven Innovation			
Data-Driven Innovation			
Module number: M.079.4076	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7052 Data-Driven Innovation	L2 Ex3	75	105	C	70/35

2 **Options within the module:**
none

3 **Admission requirements:**
none

4	**Contents:** *Contents of the course Data-Driven Innovation:* Digitalization is changing the market services of tomorrow and the way in which they are developed. Imparted methods of strategic planning and system engineering leave potentials untapped, while data-driven solutions capture these potentials. The module includes the following content: • Innovation • Smart Products • Data and Data Analysis • Smart Services • Platform Economy • Data-Driven Innovation • IT / OT • Competencies • Organizational Anchoring
5	**Learning outcomes and competences:** The students • acquire a solid understanding of Data-Driven Innovation, • know different methods and use cases • are able to apply the knowledge they have gained • are able tp to work out solutions independently and communicate them to the lecturers.
6	**Assessments:** ☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7	**Study Achievement:** none
8	**Prerequisites for participation in examinations:** none
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)

12	**Module coordinator:** Dr. Christian Koldewey, Prof. Dr.-Ing. Roman Dumitrescu
13	**Other Notes:** *Remarks of course Data-Driven Innovation:* **Implementation method** The module consists of two parts 1. lecture with slides: basics and concepts are explained in the lecture and illustrated with examples. 2. exercises (tutorial): In the exercises, knowledge is transferred and the concepts are applied. The exercises have to be prepared by the students themselves. **Learning Material, Literature** • Dumitrescu, R.; Albers, A.; Riedel, O.; Stark, R.; Gausemeier, J. (Hrsg.): Engineering in Deutschland – Status quo in Wirtschaft und Wissenschaft, Ein Beitrag zum Advanced Systems Engineering, Paderborn, 2021 – English Version: www.advanced-systems-engineering.de

Data Science for Software Engineering			
Data Science for Software Engineering			
Module number:	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number:	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	Data Science for Software Engineering	L2 Ex3	75	105	CE	30

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Data Science for Software Engineering:
Recommended Proficiencies
Good programming skills using Java and/or Python is helpful to make the assignments. Basic background on machine learning is helpful to understand some of the Data Science concepts.

4 **Contents:**

Contents of the course Data Science for Software Engineering:
Software engineers deal with software repositories in their daily work, such as when they develop source code in version control systems, or post issues in issue trackers, or communicate through emails in mailing lists, or discuss in forums and blogs. The big amount of data in software repositories, their continuous evolution, complexity and heterogeneity present a challenge for software engineers. In the past years, researchers proposed approaches that use techniques from the data science to support software engineers. This course will explain the application of data science techniques on software repositories to achieve common software engineering tasks. The course includes the following topics:

- Types and structure of software repositories.
- Clustering of source code.
- Natural language processing pipeline.
- Topic modeling.
- Word embedding.
- Information retrieval.
- Supervised machine learning.
- Statistical analysis.

Concepts are discussed in the lectures and applied using a set of group assignments to analyze opensource systems, ans achieve certain software architecture and maintenance tasks.

5 **Learning outcomes and competences:**

Students will be able to

- Clarify and discuss types and structure of software repositories.
- Clarify and discuss main concepts of data science techniques, and their application on sotware repositories.
- Apply data science techniques on large-scale software repositories.
- Derive useful implications from the analysis results.
- Summarize and report analysis results in a scientific format.
- Work in teams.
- Write scientific reports
- Present research results

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 30-45 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments and short presentations		CA

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** keine
12	**Module coordinator:** Dr. Mohamed Aboubakr Mohamed Soliman
13	**Other Notes:** *Remarks of course Data Science for Software Engineering:* **Implementation Method** The course focus on the application of data science methods in software engineering more than the mathematical background of data science methods. The main concepts of methods are conveyed through a presentation as part of a lecture and the application of methods is further investigated through group assignments and presentations. **Learning Material, Literature** Beside the slides, further learning materials from prominent publications in the software engineering literature will be provided for each topic.

Data Science in Industrial Applications

Data Science in Industrial Applications

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4075	180	6	winter term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7053 Data Science in Industrial Applications	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

4 **Contents:**

Contents of the course Data Science in Industrial Applications:
The course “Data Science in Industrial Applications” deals with the methods and techniques of data analysis in an industrial context. Students learn basic concepts of data analysis and how to apply them in practice.
The increasing networking of machines, sensors and IT systems in the context of Industry 4.0 has led to a rapid increase in the amount of available data. The analysis of data offers enormous potential for the automation of cognitive tasks, the optimization of processes and the further creation of value from data. The lecture will provide an overview of the challenges and solution approaches for the industrial application of Data Science. This includes the integration of industrial data sources from the field, the IT landscape in manufacturing companies and the setup of (Big Data) infrastructure, typical algorithms in the area of time series processing, optimization or image processing as well as the embedding in business processes. Theoretical and methodological basics, concepts and tools are introduced during the lecture and applied in small groups based on a case study as well as deepened in home exercises. The theoretical concepts for the planning, introduction and implementation of Industrial Data Science in theory are supplemented by practical real-life examples.

5 **Learning outcomes and competences:**

Students

- understand the challenges of applying Data Science in industrial applications,
- have an overview of typical application examples,
- are able to apply methods of signal processing, machine learning, and statistics to industrial problems,
- are able to plan the implementation of data acquisition, data architecture, and integration into business processes,
- are able to develop solutions on their own as well as in cooperation,
- are proficient in basic project management skills.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr.-Ing. Roman Dumitrescu

13	**Other Notes:** *Remarks of course Data Science in Industrial Applications:* **Implementation Method** The course includes lectures (slide-based), exercises (interactive), and project work. In the lectures, the theoretical basics of data analysis in an industrial context are taught. In the exercises, students have the opportunity to apply what they have learned by means of practical tasks. The project work offers the students the opportunity to apply the learned knowledge in a larger context. In the exercise, knowledge transfer and application of the concepts take place in a case study in the form of workshops and implementation of an industrial analytics application in independent group work. **Learning Material, Literature** A more detailed list of the lecture materials and references will be given in the first course. A good first insight into the subject area is given by: • Wiendahl, Hans-Peter; Wiendahl, Hans-Hermann (2019): Betriebsorganisation für Ingenieure. 9., vollständig überarbeitete Auflage. München: Hanser (Hanser eLibrary). • Zahn, Erich; Schmid, Uwe (1996): Grundlagen und operatives Produktionsmanagement. Mit 42 Tabellen. Stuttgart: Lucius & Lucius (Grundwissen der Ökonomik Betriebswirtschaftslehre, 1). • Günther Schuh; Achim Kampker: Strategie und Management produzierender Unternehmen: Handbuch Produktion und Management 1 (VDI-Buch) (German Edition). • Schuh, Günther; Riesener, Michael (2018): Produktkomplexität managen. Strategien - Methoden - Tools. 3., vollständig überarbeitete Auflage. München: Hanser (Hanser eLibrary). Online verfügbar unter http://www.hanser-elibrary.com/doi/book/10.3139/9783446453340. • Schuh, Günther; Schmidt, Carsten (2014): Produktionsmanagement. DOI: 10.1007/978-3-642-54288-6. • Bishop, Christopher M. (2006): Pattern recognition and machine learning. New York: Springer (Information science and statistics). • Cao, Longbing (2018): Data Science. In: ACM Comput. Surv. 50 (3), S. 1–42. DOI: 10.1145/3076253. • Geron, Aurelien (2019): Hands-On Machine Learning with Scikit-Learn and TensorFlow: O'Reilly Media. • Goodfellow, Ian; Bengio, Yoshua; Courville, Aaron (2016): Deep Learning. MIT Press. • James, Gareth; Witten, Daniela; Hastie, Trevor; Tibshirani, Robert (2013): An Introduction to Statistical Learning. New York, NY: Springer New York (103). • Mitchell, Tom M. (1997): Machine Learning. New York: McGraw-Hill (McGraw-Hill series in computer science). • Runkler, Thomas A. (2016): Data Analytics. Wiesbaden: Springer Fachmedien Wiesbaden. • Russell, Stuart (2009): Artificial Intelligence: A Modern Approach. 3rd Edition. Pearson. • Schutt, Rachel; O'Neil, Cathy (2013): Doing data science. Straight talk from the frontline. 1. ed. Beijing: O'Reilly.

Designing code analyses for large-scale software systems 1			
Designing code analyses for large-scale software systems 1			
Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4070	180	6	winter term

<table>
<tr><td colspan="2"></td><td>Semester number:
1-3</td><td>Duration (in sem.):
1</td><td>Teaching Language:
en</td></tr>
<tr><td>1</td><td colspan="4">Module structure:
<table>
<tr><th></th><th>Course</th><th>form of teachin</th><th>contact-time (h)</th><th>self-study (h)</th><th>status (C/CE)</th><th>group size (TN)</th></tr>
<tr><td>a)</td><td>2024.7041
Designing code analyses for large-scale software systems 1</td><td>L3
Ex2</td><td>75</td><td>105</td><td>C</td><td>70/35</td></tr>
</table></td></tr>
<tr><td>2</td><td colspan="4">Options within the module:
none</td></tr>
<tr><td>3</td><td colspan="4">Admission requirements:
none
Prerequisites of course Designing code analyses for large-scale software systems 1:
Recommended Proficiencies
A mature understanding of the Java programming languages and object-oriented programming will be helpful.</td></tr>
<tr><td>4</td><td colspan="4">Contents:
Contents of the course Designing code analyses for large-scale software systems 1:
Static code analysis is frequently used to find programming mistakes automatically, by searching for suspicious anti-patterns in a program’s code. This course will explain how to design static code analysis that are inter-procedural, i.e., consider the whole program, across procedure boundaries. Designing such analyses is challenging, as they need to handle millions of program statements efficiently and precisely. Example applications are drawn from the area of IT security.
This course is part of a combination DECA 1/2. In DECA 2 we will be covering current approaches directly our of research. We strongly recommend attending DECA 1 before DECA 2.
Topics covered include:
• Type systems and flow-insensitive, contraint-based analysis
• Lattices and fixed points
• Intra-procedural flow-sensitive static code analysis
• Interval analysis, widening and narrowing
• Call-graph construction
• Pointer Analysis
• Inter-procedural program analysis
• Call-strings approach to context-sensitive analysis
• Functional approach to context-sensitive analysis
• Value-based termination, VASCO
• Distributive analyses using IFDS
• Sensible arrangements of Flow Functions
• Distributive analyses using IDE
Throughout, we will discuss applications to software security.</td></tr>
</table>

5 **Learning outcomes and competences:**

Upon completion of the module, students will be able to

- name and distinguish the most important concepts and algorithms in the field of static program analysis
- explain the effects of various alternative design decisions when designing a static program analysis
- implement and apply simple static program analyses themselves and illustrate their function
- contrast and compare data structures and algorithms for static program analysis
- evaluate and justify the applicability of certain analysis procedures to specific application contexts, and
- develop tools for static program analysis by composing several analysis methods.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Prof. Dr. Eric Bodden

13	**Other Notes:** *Remarks of course Designing code analyses for large-scale software systems 1:* **Implementation method** Lectures and group exercises as well as practical programming labs using worldwide leading frameworks for static code analysis **Learning Material, Literature** • Thomas Reps, Susan Horwitz, and Mooly Sagiv. 1995. Precise interprocedural dataflow analysis via graph reachability. POPL '95 • Shmuel Sagiv, Thomas W. Reps, and Susan Horwitz. 1995. Precise Interprocedural Dataflow Analysis with Applications to Constant Propagation. TAPSOFT '95 • Akash Lal, Thomas Reps, and Gogul Balakrishnan. 2005. Extended weighted pushdown systems. CAV 2005 • Nomair A. Naeem, Ondrej Lhoták, and Jonathan Rodriguez. 2010. Practical extensions to the IFDS algorithm. CC 2010 • Yannis Smaragdakis, Martin Bravenboer, and Ondrej Lhoták. 2011. Pick your contexts well: understanding object-sensitivity. POPL 2011 • Eric Bodden. 2012. Inter-procedural data-flow analysis with IFDS/IDE and Soot. SOAP 2012 • Rohan Padhye, Uday P. Khedker. Interprocedural Data Flow Analysis in Soot using Value Contexts. SOAP 2013

Designing code analyses for large-scale software systems 2

Designing code analyses for large-scale software systems 2

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4071	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	2-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7042 Designing code analyses for large-scale software systems 2	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3	**Admission requirements:** none *Prerequisites of course Designing code analyses for large-scale software systems 2:* **Recommended Proficiencies** We strongly recommend that attendees have completed DECA 1 beforehand. A mature understanding of the Java and/or C++ programming languages and object-oriented programming will be helpful.
4	**Contents:** *Contents of the course Designing code analyses for large-scale software systems 2:* Static code analysis has the goal of finding programming mistakes automatically, by searching for suspicious anti-patterns in a program's code. This course will explain how to design static code analysis that are inter-procedural, i.e., consider the whole program, across procedure boundaries. Designing such analyses is challenging, as they need to handle millions of program statements efficiently and precisely. Example applications are drawn from the area of IT security. This course builds on the DECA 1 course. In DECA 2, we discuss novel concepts directly from research, for example so-called demand-driven analyses, which are characterized by a more precise and at the same time more efficient analysis, but also pushdown systems, which provide a allow elegant modeling and at the same time fast execution of program analyses. Last but not least, we explain current solutions to practical problems in static analysis, such as the use of reflection and native code. **Topics covered include:** • Program analysis of software product lines • Modeling call stacks and field accesses with Pushdown Systems • Modeling auxiliary analysis information with Weighted Pushdown Systems • Efficiency and precision gains through Demand-driven Program Analysis • Synchronized Pushdown Systems in the Boomerang framework • Applied Android code analysis with FlowDroid • Dealing with Reflection through TamiFlex • Hybrid static and dynamic analysis with Harvester • Learning source, sink and sanitizer definitions with SWAN and SWAN Assist • Explainable static analysis
5	**Learning outcomes and competences:** Upon completion of the module, students will be able to • name and explain the most important challenges in inter-procedural static program analysis • name and distinguish current methods in the field of inter-procedural static program analysis • explain the implications of various alternative design decisions in the design of an inter-procedural static program analysis. • contrast and compare data structures and algorithms for inter-procedural static program analysis and • evaluate and justify the applicability of current analysis techniques in a broad range of application contexts.

<table>
<tr><td>6</td><td>Assessments:
☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)
<table><tr><th>zu</th><th>Type of examination</th><th>Duration or scope</th><th>Weighting for the module grade</th></tr><tr><td>a)</td><td>Written or oral examination or report</td><td>120-180 min or 30-45 min or 30 min</td><td>100%</td></tr></table></td></tr>
<tr><td>7</td><td>Study Achievement:
<table><tr><th>zu</th><th>Type of achievement</th><th>Duration or Scope</th><th>SL / QT</th></tr><tr><td>a)</td><td>Assignments, course paper or progress reports</td><td></td><td>CA</td></tr></table></td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
Passing of course achievement</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Computer Engineering v4 (CEMA v4)</td></tr>
<tr><td>12</td><td>Module coordinator:
Prof. Dr. Eric Bodden</td></tr>
</table>

13	**Other Notes:** *Remarks of course Designing code analyses for large-scale software systems 2:* **Implementation method** Lectures and group exercises as well as programming exercises using widely used real-world static analysis frameworks (e.g. Soot, Phasar, FlowDroid) **Learning Material, Literature** • Context-, Flow-, and Field-sensitive Data-flow Analysis Using Synchronized Pushdown Systems (Johannes Späth, Karim Ali, Eric Bodden), In Proceedings of the ACM SIGPLAN Symposium on Principles of Programming Languages, pages 48:1–48:29, 3(POPL), 2019. • FlowDroid: Precise Context, Flow, Field, Object-sensitive and Lifecycle-aware Taint Analysis for Android Apps (Steven Arzt, Siegfried Rasthofer, Christian Fritz, Eric Bodden, Alexandre Bartel, Jacques Klein, Yves Le Traon, Damien Octeau, Patrick McDaniel), In Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation, pages 259–269, PLDI '14, ACM, 2014. • Codebase-Adaptive Detection of Security-Relevant Methods (Goran Piskachev, Lisa Nguyen Quang Do, Eric Bodden), In ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA), 2019. • Taming Reflection: Aiding Static Analysis in the Presence of Reflection and Custom Class Loaders (Eric Bodden, Andreas Sewe, Jan Sinschek, Hela Oueslati, Mira Mezini), In ICSE '11: International Conference on Software Engineering, pages 241–250, ACM, 2011.

Explainable Artificial Intelligence

Explainable Artificial Intelligence

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4091	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7025 Explainable Artificial Intelligence	L2 Ex1 P2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Explainable Artificial Intelligence:
Recommended Proficiencies
Basic knowledge in machine learning and programming

4 **Contents:**

Contents of the course Explainable Artificial Intelligence:
Explaining the predictions of machine learning models is important in an increasing number of applications. For example, bank customers would like to know why their loan was denied; machine learning engineers would like to debug and improve their models; managers would like to ensure regulatory compliance. This course aims to explain the predictions of machine learning models and introduces different explanation methods to do so. Explanation methods can be distinguished whether they are specific to a certain model or model-agnostic and whether they explain an individual prediction or the entire model.

- Introduction (e.g., importance of interpretability, evaluation of interpretability, datasets used in case studies)
- Interpretable models (e.g., linear regression, logistic regression, decision trees, decision rules)
- Global model-agnostic methods (e.g., partial dependence plots, permutation feature importance, global surrogate models)
- Local model-agnostic methods (e.g., LIME, Anchors, SHAP, counterfactual explanations)
- Model-specific methods (e.g., for neural networks)

5 **Learning outcomes and competences:**

After completing the module, students will be able to

- recognize and discuss the importance of interpretability
- explain and apply important explanation methods (e.g., interpretable models, model-agnostic methods, and model-specific methods)
- recognize characteristics of datasets, machine learning tasks, and machine learning models in application problems and argue which explanation method is appropriate for a given problem
- implement simple explanation methods from scratch
- extend and modify existing explanation methods
- discuss problems and proposed solutions with experts in the field
- read and discuss research literature in the area of XAI

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7	**Study Achievement:** zu | **Type of achievement** | **Duration or Scope** | **SL / QT** a) | Assignments, course paper or progress reports | | CA
8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Dr. Stefan Heindorf
13	**Other Notes:** *Remarks of course Explainable Artificial Intelligence:* **Implementation method** Slides and blackboard writing. Important concepts and techniques will be practiced through exercises in the lecture room and tutorials, and applied in a mini-project. **Learning Material, Literature** • Slides • Exercises • Book: Christoph Molnar. Interpretable machine learning. 2020. • Additional material and literature will be announced in the course.

Foundations of Cryptography			
Foundations of Cryptography			
Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4020	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7043 Foundations of Cryptography	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Foundations of Cryptography:
Recommended Proficiencies
Basic Knowledge in IT-Security and cryptography useful but not necessary, basic concepts of complexity theory and probability theory

4 **Contents:**

Contents of the course Foundations of Cryptography:
Cryptography is an important basic technique in IT security. Internet protocols such as TLS are based on cryptographic primitives such as key exchange, encryption and signatures. In this lecture, important basic concepts of modern cryptography will be introduced. These include encryption schemes, digital signatures, identification protocols, and multiparty computations. In all cases, formal security definitions are presented and, starting from mathematically precise assumptions, provably secure constructions are developed. An essential aspect of the lecture is the construction of efficient and secure cryptographic methods from assumptions that are as general as possible. Contents include:

- Symmetric and asymmetric encryption.
- Pseudorandom functions, one-way functions, permutations with trapdoors
- Hash functions and authentication codes
- Digital signatures, one-time signatures and random oracles.
- Identification protocols, Σ protocols.
- Security concepts such as unforgeable signatures and CPA- and CCA-secure encryption schemes.

5 **Learning outcomes and competences:**

Upon completion of the module, students will be able to:

- understand, explain and apply concepts and methods of modern cryptography.
- select appropriate cryptographic methods according to the security requirements of an application, e.g. distinguish where encryption methods and where authentication methods are appropriate.
- combine primitives of cryptography according to application requirements and prove the security of the combination.
- define new security concepts and design cryptographic methods that satisfy those concepts.
- understand and independently develop security proofs.
- acquire latest research results in the field of cryptography by reading scientific papers.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Prof. Dr. Johannes Blömer

13	**Other Notes:** *Remarks of course Foundations of Cryptography:* **Implementation method** Basic concepts are presented in a lecture. In addition, theoretical concepts are deepened in tutorials in small groups. Written exercises and reading groups will be used to practice the practical application of these concepts. **Learning Material, Literature** • Oded Gorldreich, Foundations of Cryptography I,II, • Jonathan Katz, Yehuda Lindell, Introduction to Modern Cryptogtraphy • Slides from the lectures • Lecture notes

Foundations of Knowledge Graphs			
Foundations of Knowledge Graphs			
Module number: M.079.4054	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** winter term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7026 Foundations of Knowledge Graphs	L2 Ex3	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Foundations of Knowledge Graphs:
Recommended Proficiencies
Knowledge of Graph theory and logics is beneficial.

4 **Contents:**

Contents of the course Foundations of Knowledge Graphs:
Knowledge graphs are used in an increasing number of applications. Large organisations such as Google Yahoo! and the BBC rely on these technologies to organise and manage the access to the the large amounts of data they manage. This lecture aims to present approaches for building, storing, integrating and using knowledge graphs. We will being by studying knowledge extraction techniques for unstructured data. These include named entity recognition, disambiguation and relation extraction. Technologies for storing and knowledge (e.g., triple stores) will be presented subsequently. Time-efficient and accurate approaches for knowledge integration and link prediction will be followed by a series of applications for knowledge graphs.

- Semantic networks
- Property graphs
- RDF graphs
- Query languages (e.g., Cypher, SPARQL)
- Knowledge extraction from text
- Knowledge extraction from semi-structured data
- Link discovery
- Machine learning approaches for link discovery
- Link prediction and tensor factorization

5 **Learning outcomes and competences:**

The students can carry out the following after the completion of the module:

- Model knowledge graphs;
- Describe the formal semantics of modeling languages;
- Create formal ontologies and check them for consistency;
- Model efficient imperative and descriptive languages;
- Train and execute knowledge extraction models.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Axel-Cyrille Ngonga Ngomo
13	**Other Notes:** *Remarks of course Foundations of Knowledge Graphs:* **Implementation method** 2 SWS of lectures within which the students will be presented with novel content weekly. The lecture will be self-contained with the students being presented with the premises for understanding particular aspects of knowledge graphs as well as with the corresponding conclusions and approaches derived from these premises. 1 SWS of exercises allow the students to deal with the concepts presented in the lecture through formal analysis and programming. The 2 SWS of mini-projects ensure that the students obtain a holistic understanding of the concepts learned by applying them to a more complex task than the one addressed in the exercises. **Learning Material, Literature** Slides, homework assignments

Human Factors in Security and Privacy			
Human Factors in Security and Privacy			
Module number: M.079.4092	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** winter term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7059 Human Factors in Security and Privacy	L3 Ex2	75	105	C	70/35

2 **Options within the module:**
none

3 **Admission requirements:**
none

4 **Contents:**

Contents of the course Human Factors in Security and Privacy:
Humans are important actors in security. A provable secure system is only useful if it can be actually used by users, and system designers need to account for human behavior if they wish to have both security and usability. In this class, we will examine factors of usability of security and privacy through a research-based, project-driven examination. We will cover core areas of security and privacy, as well as cover methods in human interaction (HCI) that can be used to measure the usability of security and privacy. Students are expected to complete problem sets on the topic and complete a research-based project. We will also practice academic conference reviewing, and model the academic publishing process while learning how to write and present academic research.
The course includes the following contents:

- How to write a scientific research paper?
- How to review a scientific research paper?
- How to conduct an independent scientific study in the field of Human Factors in Security and Privacy?
- Methodology: qualitative, quantitative and “mixed” methods.
- Introduction to research and scientific ethics
- Introduction to literature research
- Presenting scientific results at a conference
- How does scientific peer review work?

5 **Learning outcomes and competences:**

Students will be able to

- read and write peer reviews of scientific papers in the area of security, privacy, and usability.
- understand and apply research methods in human factors in usable security and privacy.
- develop relevant hypotheses and research questions in the space of usable security and privacy
- design and deploy a research study and analyze the results.
- describe, support, and effectively argue a result using the best practices of scientific writing.
- understand ethical issues related to human factors research in security and privacy.
- understand the major topics and themes of usable security and privacy.
- present research results in class.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7	**Study Achievement:** zu | **Type of achievement** | **Duration or Scope** | **SL / QT** a) | Assignments, course paper or progress reports | | CA
8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Yasemin Acar
13	**Other Notes:** *Remarks of course Human Factors in Security and Privacy:* **Implementation Method** • The contents are presented and elaborated in the lecture. In the accompanying tutorial, the lecture topics are deepened and discussed both in plenary and in small groups. In addition, a scientific conference with peer review will be simulated, in which students will review and discuss research papers during the semester and present them in short talks. **Learning Material, Literature** • Current freely available research papers will be provided in the course. • Redmiles, Elissa M., Yasemin Acar, Sascha Fahl, and Michelle L. Mazurek. A summary of survey methodology best practices for security and privacy researchers. 2017. https://drum.lib.umd.edu/bitstream/handle/1903/19227/CS-TR-5055.pdf • Additional literature will be announced in the course.

Introduction to Description Logics			
Introduction to Description Logics			
Module number: M.079.4098	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.079.0825 Introduction to Description Logics	L3 Ex2	75	105	CE	50/25

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Introduction to Description Logics:
Recommended Proficiencies
Knowledge of contents from the modules *Modellierung*, *Berechenbarkeit und Komplexität* and *Komplexitätstheorie* are beneficial – in particular knowedge on predicate logic and the fundamental complexity classes.

4 **Contents:**

Contents of the course Introduction to Description Logics:
This lecture and the tutorials introduce Description Logics. In detail we cover the following content:

- introduction of the Description Logic *ALC* and concept and role operators to extend *ALC*.
- introduction of DL knowledge bases and fundamental reasoning problems for DLs.
- relation of *ALC* to predicate logic and to modal logic
- model theory of *ALC*
- tableau algorithm for satisfiability in *ALC*
- Complexity analysis of reasoning in *ALC*
- reasoning metod for the DL *EL* (a fragment of *ALC*) and *ELI*
- Query answering in DLs and methods for it

5 **Learning outcomes and competences:**

Students will be able to

- use technical notions from the lecture competently and explain these notions
- master the syntax and semantics of the main elements of description Logic knowledge bases and their reasoning problems
- apply reasoning algorithms and can assess them in regard of computational complexity
- reconstruct proof methods for description logics and can apply them to instances

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	Type of examination	Duration or scope	Weighting for the module grade
a)	Written or oral examination	90-120 min or 30-45 min	100%

7 **Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Assignments		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr.-Ing. Anni-Yasmin Turhan

13 **Other Notes:**

Remarks of course Introduction to Description Logics:
Implementation Method
The course proceeds after the text book “An introduction to Description Logic”. The lecture is mainly presenting the content by slides. Proofs will be written at the black board or covered as “flipped class room”. The tutorials deepen and complement the content of the lecture.
Learning Material and literature

- “An Introduction to Description Logic” by Franz Baader, Ian Horrocks, Carsten Lutz, Uli Sattler (This book is freely available as E-Book, if accessed from the network of Paderborn university.)
- Additional literature will be announced in the course.

Introduction to Quantum Computation
Introduction to Quantum Computation

Module number:	Workload (h):	Credits:	Regular Cycle:
M.079.4059	180	6	winter term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7044 Introduction to Quantum Computation	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Introduction to Quantum Computation:
Recommended Proficiencies
Linear Algebra, algorithms

4 **Contents:**

Contents of the course Introduction to Quantum Computation:
This lecture introduces the fundamental concepts of quantum computation and information from a computer science perspective. This includes an introduction to quantum mechanics, quantum entanglement, quantum algorithms, quantum error correction, and quantum information theory.

- Quantum mechanics
- Quantum entanglement
- Quantum algorithms
- Quantum error correction
- Quantum information

5 **Learning outcomes and competences:**

Students are able to:

- Describe and apply the postulates of quantum mechanics
- Understand the use of entanglement as a resource
- Design and analyze fundamental quantum algorithms
- Apply the theory of error-correcting codes
- Understand and apply basic quantum information theory concepts such as entropy

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	Type of examination	Duration or scope	Weighting for the module grade
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Prof. Dr. Sevag Gharibian

13 **Other Notes:**

Remarks of course Introduction to Quantum Computation:
Implementation method
Slides and blackboard writing. All important concepts and techniques are further deepened with examples in exercises.
Learning Material, Literature

- Michael A. Nielsen, Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press
- Lecture slides, exercises

Machine Learning for Biometrics			
Machine Learning for Biometrics			
Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4088	180	6	winter term

	Semester number:	Duration (in sem.):	Teaching Language:
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7024 Machine Learning for Biometrics	L2 Ex3	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

4 **Contents:**

Contents of the course Machine Learning for Biometrics:
Biometric verification is defined as the automated recognition of individuals based on their behavioral or biological characteristics. The course will give an overview of modern biometric systems and specifically address their functionality and challenges. For this purpose, various approaches of machine learning will be introduced, which aim at enabling reliable biometric recognition (e.g. by means of face recognition). At the same time, biometric applications place very specific requirements on the underlying algorithms. The course will specifically address these requirements and how they can be met algorithmically and in the algorithmic learning process. This includes the topics of privacy, fairness, explainability, uncertainties, efficiency, attacks and their automated detection.
The course includes the following content:

- Biometric systems, operation modes, and evaluation
- Recap on traditional and deep learning
- Face, Iris, and fingerprint recognition
- Soft-biometrics and privacy
- Fairness and bias in biometric systems
- Explainability and confidence in biometric systems
- Biometric sample quality
- Efficient biometric systems
- Presentation attacks and detection
- Multi-biometric fusion
- Biometric indexing

<table>
<tr><td>5</td><td>Learning outcomes and competences:
Students will be able to
• independently evaluate biometric systems,
• train biometric recognition models for different modalities,
• automatically detect biometric attacks and make systems robust against such attacks,
• explain various challenges of biometric systems and name solution strategies to counter them,
• name and explain open research questions in biometrics.</td></tr>
<tr><td>6</td><td>Assessments:
☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)
<table>
<tr><th>zu</th><th>Type of examination</th><th>Duration or scope</th><th>Weighting for the module grade</th></tr>
<tr><td>a)</td><td>Written or oral examination or report</td><td>120-180 min or 30-45 min or 30 min</td><td>100%</td></tr>
</table></td></tr>
<tr><td>7</td><td>Study Achievement:
none</td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
none</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Computer Engineering v4 (CEMA v4)</td></tr>
<tr><td>12</td><td>Module coordinator:
Dr.-Ing. Philipp Terhörst</td></tr>
</table>

13	**Other Notes:** *Remarks of course Machine Learning for Biometrics:* **Implementation Method** First, students are given an overview of biometrics and its applications and basic functionalities. Then, required concepts of machine learning are introduced in a compact way. These will be applied and developed in context when dealing with specific biometric requirements. Parallel to the lecture, the theoretical concepts are practiced in the exercises using facial data. This is done in the form of short hand-written and implementation tasks. **Learning Material, Literature** • Anil K. Jain, Patrick Flynn, and Arun A. Ross. 2010. Handbook of Biometrics (1st. ed.). Springer Publishing Company, Incorporated. • Further literature will be announced in the lecture.

Model-Based Systems Engineering

Model-Based Systems Engineering

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4062	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7058 Model-Based Systems Engineering	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Model-Based Systems Engineering:
Recommended Proficiencies
Basics of Systems Engineerings

4 **Contents:**

Contents of the course Model-Based Systems Engineering:
Due to the technical change from mechatronic to intelligent technical systems (ITS), companies and development teams are facing many challenges. A key factor is the increase in complexity and networking of systems (products). Existing approaches in product development cannot cover this efficiently and effectively. Model-based Systems Engineering (MBSE) presents itself as a promising approach to solve these challenges. MBSE sees itself as a further development of systems engineering and builds on its foundations. Systems engineering, which is primarily based on documents, is extended by the introduction of models.
The course includes the following content:

- Intelligent Engineering Systems
- Model-based Systems Engineering 101
- Systems Modeling Fundamentals
- Languages and Methods - CONSENS, SysML
- Systems Architecting
- IT Tools for MBSE

5 **Learning outcomes and competences:**

The students

- acquire a solid understanding of Model-Based System Engineering
- know different methods, languages, and tools
- are able to apply the knowledge they have gained
- are able to work out solutions independently and communicate them to the lecturers.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

none

8 **Prerequisites for participation in examinations:**

none

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr.-Ing. Roman Dumitrescu
13	**Other Notes:** *Remarks of course Model-Based Systems Engineering:* **Implementation Method** The module consists of two parts 1. lecture with slides: basics and concepts are explained in the lecture and illustrated with examples. 2. exercises (tutorial): In the exercises, knowledge is transferred and the concepts are applied. The exercises have to be prepared by the students themselves. **Learning Material, Literature** • Gausemeier, J.; Dumitrescu, R.; Steffen, D.; Czaja, A.; Wiederkehr, O.; Tschirner, C.: Systems Engineering in industrial practice. Heinz Nixdorf Institute, University Paderborn, 2013, Under: https://www.hni.uni-paderborn.de/en/spe/systemsengineering/ • Dumitrescu, R.; Albers, A.; Riedel, O.; Stark, R.;Gausemeier, J. (Eds): Engineering in Germany – Status quo in Business and Science. Federal Ministry of Education and Research, 2021 Under: https://www.advanced-systems-engineering.de/#studie • Additional literature will be announced in the course.

Multi-Objective Optimisation			
Multi-Objective Optimisation			
Module number:	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	Multi-Objective Optimisation	L3 Ex2	75	105	CE	30/15

2 **Options within the module:**
none

3 **Admission requirements:**

Prerequisites of course Multi-Objective Optimisation:
Recommended Proficiencies
Solid basic knowledge of algorithms and data structures, mathematics, as well as basic knowledge of optimization are beneficial.

4 **Contents:**

Contents of the course Multi-Objective Optimisation:
Optimization problems are ubiquitous, and we all (approximately) solve them in everyday life, such as when finding routes with Google Maps to quickly get from point A to point B or deciding on a checkout lane with the shortest waiting queue (shortest expected waiting time) at the supermarket. However, optimization problems are rarely single-criteria. Instead, they are typically multi-criteria in nature, with the individual objectives usually conflicting with each other. For example, in route planning, the distance traveled may be relevant (shorter is better), and fuel consumption may also be a consideration (lower is better). The shortest route may lead through the city center with many stop-and-go maneuvers at red lights, especially during peak hours. On the other hand, a longer route around the city may consume less fuel. Accordingly, the goal in multi-objective optimization is to find a set of optimal compromise solutions.
This course provides a comprehensive introduction to multi-objective optimization and the associated challenges. In addition to classical general approaches, exact methods for selected combinatorial optimization problems are presented, along with heuristic (nature-inspired) methods. The course also covers heuristic solution approaches for problems with more than three criteria (many-objective optimization).

5 **Learning outcomes and competences:**

Students will be able to

- Explain, implement, and apply important exact algorithms for multi-criteria minimum spanning tree problems and multi-criteria shortest path problems
- Understand the limitations of exact algorithms for multi-criteria problems
- Explain and apply biologically inspired heuristics for multi-objective problems
- Assess, evaluate, and visualize the quality of computed results from multi-criteria algorithms
- Understand the challenges of problems with more than three criteria and explain solution approaches

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments		CA

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** keine
12	**Module coordinator:** Prof. Dr. Heike Trautmann
13	**Other Notes:** *Remarks of course Multi-Objective Optimisation:* **Implementation Method** Slide-based lecture with interspersed assignments. In the tutorial, the knowledge transfer and application of what has been learned takes place in both theoretical and practical assignments. **Learning Material, Literature** • Deb, Kalyanmoy. „Multi-Objective Optimization Using Evolutionary Algorithms“. • Ehrgott, Matthias. Multicriteria Optimization. Bd. 491. Lecture Notes in Economics and Mathematical Systems. Berlin, Heidelberg: Springer, 2000. • Additional literature will be announced in the course.

Post-Quantum Cryptography			
Post-Quantum Cryptography			
Module number: M.079.4089	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7015 Post-Quantum Cryptography	L3 Ex2	75	105	C	70/35

2 **Options within the module:**
none

3 **Admission requirements:**

none

Prerequisites of course Post-Quantum Cryptography:
Recommended Proficiencies
Basics of cryptography and complexity theory

4 **Contents:**

Contents of the course Post-Quantum Cryptography:
IT security is largely based on modern cryptographic methods. These include many methods of so-called public-key cryptography such as the RSA and Elgamal encryption methods, the RSA signature method, and the various variants of the Digital Signature Algorithm (DSA). In 1994, Peter Shor presented an efficient algorithm for computing prime factorization of integers and for computing discrete logarithms in finite groups. Thus, all the aforementioned methods of public-key cryptography are insecure if quantum computers of sufficient size and complexity can be realized. It is therefore important to develop alternatives to classical public-key methods that, at least according to current research, cannot be broken by quantum computers. Important candidates (and some close to standardization) for such post-quantum secure methods rely on techniques of error-correcting codes and the geometry of numbers. In this lecture, we will present and discuss important candidates for post-quantum secure methods. The course includes the following contents:

- introduction to codes, lattices and discretised Gaussian distributions
- lattice and code based enryption
- lattice based signatures
- lattices and zero-knowledge proofs
- lattice based group signatures

5 **Learning outcomes and competences:**

Students will be able to

- understand and explain the difference between classical and post-quantum security.
- explain the importance of post-quantum cryptography for selected applications.
- explain and apply concepts from the field of geometry of numbers and error-correcting codes.
- explain important constructions from post-quantum cryptography and prove their security.
- explain security assumptions from post-quantum cryptography and apply them to new post-quantum primitives.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7	**Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Assignments, course paper or progress reports		CA

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Johannes Blömer
13	**Other Notes:** *Remarks of course Post-Quantum Cryptography:* **Implementation Method** Basic concepts are presented in a lecture. In addition, theoretical concepts are deepened in tutorials in small groups as well as in written exercises. **Learning Material, Literature** References to current learning materials will be given in the lectures.

Privacy and Technology

Privacy and Technology

Module number:	Workload (h):	Credits:	Regular Cycle:
M.079.4087	180	6	winter term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7045 Privacy and Technology	L2 Ex3	75	105	CE	70/35

2	**Options within the module:** none

3	**Admission requirements:** none
4	**Contents:** *Contents of the course Privacy and Technology:* This course provides students with a basic understanding of privacy risks and principles, the most common technologies for addressing them and the human factors that shape their design. The course will analyze the adversary models and evaluation metrics underlying the design of privacy-enhancing technologies. Moreover a quick overview of usable security as well as identity management and dedicated case studies will be given. For that, a superficial knowledge of HCI basics is desirable. By reviewing relevant papers and giving presentations, the students will get familiar with the latest research in the field and gain knowledge about how to work scientifically. The course includes the following contents: • Privacy metrics and adversary models • Anonymous communications • Data-perturbative privacy-enhancing technologies • Anonymization algorithms for databases • Homomorphic encryption and zero knowledge proofs • Selective disclosure for identity management • Usable privacy • Applying privacy principles and case studies
5	**Learning outcomes and competences:** The students • are able to reason critically about privacy, • gain knowledge in the evaluation of privacy risks, • understand the design aspects of privacy-enhancing technologies, • get familiar with the latest research in the field and • analyze and discuss the space of solutions to a given privacy problem
6	**Assessments:** ☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7	**Study Achievement:**	

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Patricia Arias Cabarcos
13	**Other Notes:** *Remarks of course Privacy and Technology:* **Implementation Method** The contents are taught through a presentation in the form of a lecture. In addition, they are deepened in presence exercises in small groups, as well as through individual presentations. Through practical exercise, methods are implemented and applied. **Learning Material, Literature** • Lecture slides, scientific literature and specific readings will be provided during the course.

Quantum Algorithms			
Quantum Algorithms			
Module number: M.079.4072	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7014 Quantum Algorithms	L3 Ex2	75	105	C	70/35

2 **Options within the module:**
none

3 **Admission requirements:**

none

Prerequisites of course Quantum Algorithms:
Recommended Proficiencies
Linear Algebra, Quantum Computing

4 **Contents:**

Contents of the course Quantum Algorithms:
This lecture covers quantum algorithms from a computer science perspective. Topics include quantum circuits (e.g. Solovay-Kitaev theorem), quantum algorithms for algebraic problems (e.g. Hidden Subgroup problem), quantum walks, quantum query complexity, and adiabatic quantum computing.

- Quantum circuits
- Algebraic problems
- Quantum walks
- Query complexity
- Adiabatic computation

5 **Learning outcomes and competences:**

Students are able to:

- Describe universal gate sets
- Develop Quantum Fourier-Transform based algorithms
- Develop quantum walk-based algorithms
- Apply the quantum adiabatic theorem
- Give quantum query lower bounds

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Sevag Gharibian
13	**Other Notes:** *Remarks of course Quantum Algorithms:* **Implementation method** Slides and blackboard writing. All important concepts and techniques are further deepened with examples in exercises. **Learning Material, Literature** • Michael A. Nielsen, Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press • Andrew M. Childs, Wim van Dam, Quantum algorithms for algebraic problems, Reviews of Modern Physics, volume 82, 2010 • Lecture slides, exercises

Quantum Complexity Theory

Quantum Complexity Theory

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4063	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7046 Quantum Complexity Theory	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Quantum Complexity Theory:
Recommended Proficiencies
Linear Algebra, Quantum Computing

4 **Contents:**

Contents of the course Quantum Complexity Theory:
This lecture provides a brief review of introductory quantum computation, and subsequently moves into quantum complexity theory. Beginning to advanced topics will be covered, including quantum analogues of P and NP (denoted BQP, QCMA, and QMA), quantum satisfiability problems, quantum interactive proofs, and tensor networks. Along the way, semidefinite programming will be introduced as an important tool.

- Complexity classes BQP, QCMA, QMA
- Quantum algorithms for linear system solving
- Quantum Satisfiability Problems
- Quantum Interactive Proofs
- Semidefinite Programming

5 **Learning outcomes and competences:**

Students will be able to

- Distinguish language classes from promise classes
- Define fundamental quantum complexity classes, such as BQP and QMA
- Prove BQP-hardness results via polynomial-time reductions
- Prove QMA-hardness results via polynomial-time reductions
- Apply semidefinite programming to analyze quantum interactive proofs

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Sevag Gharibian
13	**Other Notes:** *Remarks of course Quantum Complexity Theory:* **Implementation method** Slides and blackboard writing. All important concepts and techniques are further deepened with examples in exercises. **Learning Material, Literature** • Michael A. Nielsen, Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press • S. Gharibian, Y. Huang, Z. Landau, S. W. Shin, Quantum Hamiltonian Complexity, Foundations and Trends in Theoretical Computer Science • Lecture slides, assignments

Quantum Information			
Quantum Information			
Module number: M.079.4090	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** de

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7040 Quantum Information	L3 Ex2	75	105	C	70/35

2 **Options within the module:**
none

3	**Admission requirements:** none *Prerequisites of course Quantum Information:* **Recommended Proficiencies** Linear Algebra
4	**Contents:** *Contents of the course Quantum Information:* Over the last century, Quantum mechanics has had profound impacts on both fundamental science and technology. The emerging field of Quantum Information Theory studies a paradigm for information processing empowered by quantum mechanics. This field has demonstrated that quantum information processing can outperform its classical counterpart and is a revolutionary direction to investigate future information technologies. Quantum Information Science incorporates techniques from computer science, mathematics, and physics. Of particular interest is quantum entanglement, which is the phenomenon that occurs when a group of particles is generated or interacts in a way such that the state of each particle cannot be described independently of the others, even when the particles are separated by arbitrarily large distances. Entanglement is a primary feature of quantum mechanics not present in classical physics and it is a resource behind most modern quantum technologies, such as quantum computers. This lecture introduces the advance concepts of quantum communication and information. The contents include: • Entanglement of two- and many-body systems • Quantum information processing and applications • Measures of Entanglement, Distance and Fidelity • Higher local dimensions (qubits vs qudits) • Quantum channels • Classical and quantum error correcting codes and their differences
5	**Learning outcomes and competences:** Students learn cutting-edge concepts at the intersection of computer science and quantum mechanics. This lecture equips students with advanced, interdisciplinary technical proficiency, enabling them to pursue careers in analysis intensive industries, technology start-ups, or research and development roles in leading technology companies or academia. To achieve this, the students get familiar with the basics of quantum mechanics and the related algebra. Furthermore, they will be able to: • understand the underlying concepts of entangled systems (two-body and many-body), • understand the fundamental idea of maximally entangled systems, classify and characterise them for practical applications, • describe the basic notion of higher local dimension particles (qubits vs qudits), • apply the theory of classical and quantum error correcting codes, and study their differences, • to work on interdisciplinary topics and, in particular, to acquire the basics of different disciplines.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	Type of examination	Duration or scope	Weighting for the module grade
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Dr. Zahra Raissi

13 **Other Notes:**

Remarks of course Quantum Information:
Implementation Method
Theoretical foundations and concepts will be taught in the form of lectures and deepened in practical exercise courses, group work as well as individual assignments.
Learning Material, Literature

- Michael A. Nielsen, Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2000.
- F. J. MacWilliams and N. J. A. Sloane. The Theory of Error-Correcting Codes, North-Holland Mathematical Library. North-Holland, Amesterdam, 1977. ISBN 9780444851932.
- Ingemar Bengtsson and Karol Zyczkowski, Geometry of quantum states: an introduction to quantum entanglement, Cambridge university press, 2006, ISBN 9780511535048.
- Lecture slides
- Exercises

Real World Crypto Engineering

Real World Crypto Engineering			
Module number: M.079.4067	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** winter term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7047 Real World Crypto Engineering	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Real World Crypto Engineering:
Recommended Proficiencies
Knowledge in programming, IT security and basic knowledge in cryptography

4 **Contents:**

Contents of the course Real World Crypto Engineering:
Strong cryptography is not always sufficient to protect primary security goals. Even if strong cryptographic algorithms are used, a lot can go wrong when they are implemented. This lecture will dive into the most important protocols and cryptographic protection mechanisms (e.g., TLS, SSH, WPA) and show their basic concepts. Then, we will present prominent attacks that ultimately break the desired security goals. Based on many cases, we will learn what is essential when designing and implementing cryptographic applications.
The course includes the following contents:

- Brief introduction to cryptography
- TLS (Transport Layer Security)
- Attacks on TLS (e.g., ROBOT, DROWN, or Invalid Curve)
- Evaluation of implementations with systematic methods (e.g., with fuzzing or state learning)
- SSH (Secure Shell)
- Signal
- Cryptocurrencies

5 **Learning outcomes and competences:**

Students will be able to

- Understand concepts behind major cryptographic protocols
- Understand and prevent common attacks on cryptographic protocols
- Conduct analyses of cryptographic implementations using systematic methods and standard tools
- Identify and assess implementation errors and security issues in cryptographic protocols

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Prof. Dr.-Ing. Juraj Somorovsky

13	**Other Notes:** *Remarks of course Real World Crypto Engineering:* **Implementation method:** The topics are conveyed through lecture presentations. They are further deepened through individual practical tasks. **Learning Material, Literature:** • Lecture slides and exercise sheets • Scientific literature • Additional literature will be announced in the course.

Reconfigurable Computing

Reconfigurable Computing

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4043	180	6	winter term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7034 Reconfigurable Computing	L2 Ex3	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Reconfigurable Computing:
Recommended Proficiencies
Knowledge of the Bachelor-level courses Digital Design, Programming, and Data Structures and Algorithms are beneficial.

4 **Contents:**

Contents of the course Reconfigurable Computing:
The course Reconfigurable Computing introduces into the field of computing with reprogrammable hardware structures. Computing systems built from reprogrammable hardware structures do not rely on a fixed hardware, but adapt their hardware architecture to the application under execution. The field was formed in the early 1990s when Field-programmable Gate Arrays (FPGAs) became commercially available that were powerful enough to be used for computing. Today, FPGA-based high-performance systems have outperformed state-of-the-art computers for many problems including database search, genomic sequence scanning, and cryptography. In embedded systems, FPGAs accelerate system functions, reduce system cost and energy consumption, and enable hardware-on-demand functionality. The course covers the following topics:

- Introduction to reconfigurable computing
- Evolution of programmable hardware devices
- FPGA architectures
- Computer-aided design for FPGAs
- High-level languages for programming FPGAs
- Application domains for FPGAs
- Comparison of devices, technologies, and reconfigurable systems

5 **Learning outcomes and competences:**

Students will be able to

- compare different reprogrammable hardware devices and describe their historical development,
- name the design steps and problems when designing with FPGAs,
- analyse algorithms for the design steps and apply them to examples,
- compare and evaluate current approaches to programming FPGAs,
- justify the suitability of different reprogrammable hardware components for different areas of application, and
- implement functions of medium complexity with modern FPGA design tools.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Marco Platzner
13	**Other Notes:** *Remarks of course Reconfigurable Computing:* **Implementation Method** The course consists of a lecture, and pencil&paper as well as practical exercises. The lecture is held with a beamer and blackboard. In the pencil&paper exercises, problems are handed out and their solutions are presented and discussed in a practice session. In addition, quizzes are offered for self-assessments. In the practical exercises, a tutorial on the design with FPGAs is carried out and then tasks are handed out, which are implemented as design or programming examples in groups of one to three participants. **Learning Material, Literature** • Lecture slides, assignment sheets for paper&pencil exercises, quizzes • Tutorial, assignment sheets for design and programming examples, technical documentation • Selected scientific articles • Additional literature will be announced in the course.

Software Architecture Design and Recovery			
Software Architecture Design and Recovery			
Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4094	180	6	winter term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.705a Software Architecture Design and Recovery	L2 Ex3	75	105	C	30

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Software Architecture Design and Recovery:
Recommended Proficiencies
A good understanding of Java and the principle of object-oriented programming is helpful.

4 **Contents:**

Contents of the course Software Architecture Design and Recovery:
Software architecture is concerned with the principal design decisions of a software system. These decisions have significant impact on the system's quality, such as maintainability, performance and security. This course will explain fundamental concepts of the software architecture field, as well as how to apply techniques to recover design decisions from existing software repositories. The course includes the following topics from software architecture field:

- Types of design decisions.
- Architectural components and recovery.
- Architectural solutions such as patterns, tactics and technologies.
- Architectural documentation.
- Software repositories.
- Architectural knowledge.
- Design processes.

Furthermore, the course discusses and applies common research methods:

- Grounded theory
- Case studies

5 **Learning outcomes and competences:**

Students will be able to

- clarify and discuss main concepts in the software architecture field, such as architectural solutions, components, and design processes
- analyze large-scale software systems for architectural design decisions
- execute design processes to make design decisions
- apply common research methods on software architecture problems
- summarize and report research results in a scientific format
- work in teams
- present their results to the audience

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

keine

12 **Module coordinator:**

Dr. Mohamed Aboubakr Mohamed Soliman

13	**Other Notes:** *Remarks of course Software Architecture Design and Recovery:* **Implementation Method** Lectures and group assignments on large open-source software systems, as well as presentations. Concepts are discussed in the lectures and applied using a set of group assignments on real open-source software systems. **Learning Material, Literature** • Bass, L., Clements, P., Kazman, R. (2012). Software Architecture in Practice. 3rd Edition, Addison-Wesley Professional. • Kruchten P, Lago P, van Vliet H (2006) Building Up and Reasoning About Architectural Knowledge. In: Quality of Software Architectures, Springer Berlin Heidelberg. • Additional literature will be announced in the course.

Software Product Lines

Software Product Lines

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4102	180	6	summer term
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	Software Product Lines	L2 Ex3	75	105	C	30

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Software Product Lines:
Recommended Proficiencies
Basic knowledge of logic (in particular propositional logic) and software engineering (in particular process models, UML class diagrams, design patterns) as well as programming experience (e.g., in Java) are required.

4	**Contents:** *Contents of the course Software Product Lines:* Modern software often has to be available on many platforms and adapted to many different user and customer needs. This applies to system software (e.g. operating systems), application software (e.g. word processing and games) and complex cyber-physical systems (e.g. automobiles). The resulting variety of configurations poses challenges for the development, testing, and maintenance of such systems. The lecture Software Product Lines teaches, among other things, how the configurability of systems can be modeled, which implementation techniques allow extensible and configurable software to be developed, and which strategies can still be used for meaningful testing despite an exponential number of variants. The course includes the following contents: • Introduction to software variability and inherent challenges • Modeling and analysis of the desired variability • Implementing variability at runtime (e.g. configuration options) and compile-time (e.g. clone-and-own) • Implementation of software product lines: – Implementing features using conditional compilation (e.g. preprocessors and build systems). – Modular implementation of features (e.g. components, services and plug-ins) – Limitations of object orientation and extensions of object orientation (e.g. feature modules, aspects) • Design patterns for software variability • Process models for the use and development of software product lines • Problems and dealing with feature interactions • Methods for static and dynamic quality assurance of software product lines • Evolution and maintenance of software product lines • Current topics from research and practice
5	**Learning outcomes and competences:** Students are able to • identify the limitations of traditional programming techniques with regard to the development of variable software. • describe modeling, analysis and configuration of variability in software product lines. • apply different implementation techniques for the development of software product lines. • evaluate the suitability of presented programming techniques for different development scenarios. • explain quality assurance techniques for software product lines and the associated challenges.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	Type of examination	Duration or scope	Weighting for the module grade
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

keine

12 **Module coordinator:**

Prof. Dr. Thomas Thüm

13 **Other Notes:**

Remarks of course Software Product Lines:
Implementation Method
The lecture provides a theoretical overview of techniques for modeling, implementing and analyzing configurable systems. In the exercise, this knowledge is further deepened through application tasks and practical understanding is promoted through the development of own software product lines with the aid of various programming techniques. Some of the software product lines to be developed can be chosen individually, so that students have the opportunity to gain programming experience in a domain of interest to them.
Learning Material, Literature

- Feature-Oriented Software Product Lines - Concepts and Implementation; Sven Apel, Don Batory, Christian Kästner, Gunter Saake; Springer, 2013
- Mastering Software Variability with FeatureIDE; Jens Meinicke, Thomas Thüm, Reimar Schröter, Fabian Benduhn, Thomas Leich, Gunter Saake; Springer, 2017

Unsupervised Learning and Evolutionary Optimisation Using R

Unsupervised Learning and Evolutionary Optimisation Using R			
Module number: M.079.4093	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** winter term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7027 Unsupervised Learning and Evolutionary Optimisation Using R	L3 Ex2	75	105	C	70

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Unsupervised Learning and Evolutionary Optimisation Using R:
Recommended Proficiencies

- Basic knowledge and interest in mathematics, statistics and probability theory
- Basic knowledge of programming

4 **Contents:**

Contents of the course Unsupervised Learning and Evolutionary Optimisation Using R:
The course includes the formal and applied concepts of unsupervised machine learning and its implementation in the statistical programming language R.
In particular, the following topics are covered in a theoretical and applied manner:

- Introduction to the statistical programming language R
- Data pre-processing and quality aspects of data
- (Stream) clustering techniques
- Dimensionality reduction techniques
- Basic principles of evolutionary optimisation, both single- and multi-objective
- Practical application of the methods using R in individual and group work

5 **Learning outcomes and competences:**

After completing the module, students will be able to ...

- properly assess data quality and select suitable techniques for data pre-processing
- explain and apply core methods of unsupervised learning
- understand the basic principles of evolutionary optimisation methods
- competently apply techniques to assess the quality of optimisation procedures
- use the statistical software R for statistical data analysis, unsupervised learning and evolutionary optimisation in a competent manner
- analyse problems in a team and present practice-relevant solutions

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

keine

12 **Module coordinator:**

Prof. Dr. Heike Trautmann

13	**Other Notes:** *Remarks of course Unsupervised Learning and Evolutionary Optimisation Using R:* **Implementation Method** An introduction to the statistical programming language R is given compactly in the first weeks of the course. Methods of unsupervised machine learning are covered within lecture presentations interleaved with interactive exercises. Methods understanding will be further deepened in tutorials focusing both on theory as well as application-oriented tasks using R. **Learning Material, Literature** Recommended for the statistical programming language R: • Hadley Wickham & Garrett Grolemund (2023). R for Data Science: Import, Tidy, Transform, Visualize, and Model Data. 2nd ed. O'Reilly • Torsten Hothorn and Brian S. Everitt (2014). A Handbook of Statistical Analyses Using R. Chapman & Hall/CRC Press, 3rd edition, 2014. • C. Heumann, M. Schomaker, and Shalabh. Introduction to Statistics and Data Analysis With Exercises, Solutions and Applications in R. Springer, 2017. The methods sections are based on a variety of references which will be announced in the lecture.

Usable Security and Privacy			
Usable Security and Privacy			
Module number: M.079.4086	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7048 Usable Security and Privacy	L2 Ex3	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

4 **Contents:**

Contents of the course Usable Security and Privacy:
Human factors and usability issues have traditionally played a limited role in security research and secure systems development. Usability issues have been largely disregarded by security experts due to their failure to acknowledge their significance and their insufficient knowledge to tackle them. Today there is consensus on the importance of understanding users behavior and improving usability to achieve true security. This course provides practical and research-oriented knowledge about usable security and privacy. Students will gain practical experience through focused presence exercises and work in small teams to conduct a semester-wide research project with the goal of designing and pretesting a user study on human-centered security and privacy. For that, the course will present research methods and give an introduction into HCI and usability concepts. The course will also address foundational and state-of-the-art research topics in the area, such as privacy and transparency enhancing tools, usable authentication, and developer-centered security. By reviewing relevant papers and giving presentations, the students will get familiar with the latest research in the field and gain knowledge about how to work scientifically.
The course includes the following contents:

- Security and privacy concepts
- Foundations of cryptography
- Privacy and transparency enhacing tools
- HCI and usability research methods
- Ethics in technology
- Quantitative and qualitative data analysis
- Usable authentication
- Usable privacy
- Developer-centered security

5 **Learning outcomes and competences:**

Students will

- gain an appreciation for the importance of usable security and privacy
- learn about the history of the field and main research areas and challenges
- are able to apply methodologies to conduct user research in security and privacy
- get familiar with the latest research in the field

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7	**Study Achievement:** zu / **Type of achievement** / **Duration or Scope** / **SL / QT** a) / Assignments, course paper or progress reports / / CA
8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Computer Engineering v4 (CEMA v4)
12	**Module coordinator:** Prof. Dr. Patricia Arias Cabarcos
13	**Other Notes:** *Remarks of course Usable Security and Privacy:* **Implementation method** Basic concepts are presented in a lecture style format. By engaging in presence exercises and conducting a research project in small groups focused on a user-study for usable security and privacy research throughout the semester, students can acquire more profound theoretical and practical knowledge. **Learning Material, Literature** • Lazar, J., Feng, J.H. and Hochheiser, H., 2017. Research methods in human-computer interaction. Morgan Kaufmann. • Redmiles, E.M., Acar, Y., Fahl, S. and Mazurek, M.L., 2017. A summary of survey methodology best practices for security and privacy researchers. • Slides and scientific literature references will be given during the course.

VLSI-Testing			
VLSI-Testing			
Module number: M.048.92027	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** winter term
	Semester number: 1.-3. Semester	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.048.92027 VLSI Testing	2L 2Ex, WS	60	120	C	30/30

2 **Options within the module:**

None

3 **Admission requirements:**

None

Prerequisites of course VLSI Testing:
Recommended: Digital Design

4 **Contents:**

Contents of the course VLSI Testing:
Short Description
The course "VLSI Testing" focuses on techniques for detecting hardware defects in micro-electronic circuits. Algorithms for test data generation and test response evaluation as well as hardware structures for design for test (DFT) and on-chip test implementation (BIST) are presented.
Contents
In detail the following topics are covered:

- Fault models
- Testability measures and design for test (DFT)
- Logic and fault simulation
- Automatic test pattern generation (ATPG)
- Built-in self-test (BIST), in particular test data compression and test response compaction
- Memory test

5 **Learning outcomes and competences:**

Domain competence:
After attending the course, the students will be able

- to describe fault models, DFT techniques, and test tools,
- to explain and apply the underlying models and algorithms for fault simulation and test generation,
- to analyze systems with respect to their testability and to derive appropriate test strategies.

Key qualifications:
The students

- are able to apply the practiced strategies for problem solving across varying disciplines,
- have experience in presenting their solutions to their fellow students, and
- know how to improve their competences by private study.

<table>
<tr><td>6</td><td>Assessments:
☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)<table><tr><th>zu</th><th>Type of examination</th><th>Duration or scope</th><th>Weighting for the module grade</th></tr><tr><td>a)</td><td>Written or Oral Examination or Presentation</td><td>120-180 min or 30-45 min or 30 min</td><td>100%</td></tr></table></td></tr>
<tr><td>7</td><td>Study Achievement:
none</td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
None</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination (MAP) was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Computer Engineering v3 (CEMA v3), Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Electrical Systems Engineering v3 (ESEMA v3) - Amtssprache, Master’s Program Electrical Systems Engineering (ESEMA v2)</td></tr>
<tr><td>12</td><td>Module coordinator:
Prof. Dr. Sybille Hellebrand</td></tr>
</table>

13 **Other Notes:**

Remarks of course VLSI Testing:

ATTENTION - IMPORTANT NOTICE
The course doesn't take place in summer term 2024. Please see the notice boards of the group.

Course Homepage
`https://ei.uni-paderborn.de/en/electrical-engineering/date/teaching/electrical-engineering/overview`
Implementation

- Lecture based on slide presentation, extensions on blackboard
- Exercises in small groups based on exercise sheets with students presenting their own solutions
- Hands-on exercises using various software tools

Teaching Material, Literature
Additional material can be found in panda

- Michael L. Bushnell, Vishwani D. Agrawal, „Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits,"Boston, Dordrecht, London: Kluwer Academic Publishers, 2000
- Laung-Terng Wang, Cheng-Wen Wu, Xiaoqing Wen, „VLSI Test Principles and Architectures: Design for Testability," Morgan Kaufmann Series in Systems on Silicon, ISBN: 0123705975

Web Security			
Web Security			
Module number: M.079.4073	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** summer term
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	2024.7049 Web Security	L3 Ex2	75	105	C	70/35

2	**Options within the module:** none
3	**Admission requirements:** none *Prerequisites of course Web Security:* **Recommended Proficiencies** Knowledge in programming, IT security and basic knowledge in cryptography
4	**Contents:** *Contents of the course Web Security:* Modern web applications and web services usually consist of multiple layers. They are based on different (often complex) technologies that are constantly being developed. Their complexity is often the reason for new types of attacks that can be observed on the web every day. In this lecture, we will focus on the most important technologies and learn what you have to consider while securing your web applications. We will introduce prominent and widespread attacks and show how to prevent them. These range from typical attacks from the OWASP Top 10 list, such as XSS or SQL Injection, to attacks on web services and Single Sign-On standards (e.g., on SAML and OpenID Connect). Based on many cases, we will learn what is important in the design and implementation of secure web applications. The course includes the following contents: • Introduction to web technologies • Web Attacks – Cross-Site Scripting (XSS) – Cross-Site Request Forgery (CSRF) – Clickjacking – SQL injection • XML and SAML – Attacks on XML parsers – Attacks on XML Signature • JSON and OpenID Connect (OIDC) – Attacks on OIDC
5	**Learning outcomes and competences:** Students will be able to • Understand security concepts behind web applications • Understand and prevent common attacks on web applications • Carry out practical analyses of web applications with common tools • Identify and assess implementation errors and security problems in web applications

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	Type of examination	Duration or scope	Weighting for the module grade
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4)

12 **Module coordinator:**

Prof. Dr.-Ing. Juraj Somorovsky

13 **Other Notes:**

Remarks of course Web Security:
Implementation method:
The topics are conveyed through lecture presentations. They are further deepened through individual practical tasks.
Learning Material, Literature:

- Lecture slides and exercise sheets
- Scientific literature
- Additional literature will be announced in the course.

4 Focus Areas

4.1 Classical and Quantum Algorithm Design

Coordination

Prof. Dr. Christian Scheideler

Modules in this Focus Area

- Advanced Algorithms
- Advanced Complexity Theory
- Advanced Distributed Algorithms and Data Structures
- Algorithms for Complex Virtual Scenes
- Foundations of Cryptography
- Introduction to Quantum Computation
- Post-Quantum Cryptography
- Quantum Algorithms
- Quantum Complexity Theory
- Quantum Information

Description

In this specialization area, students learn essential techniques for designing efficient algorithms, both for classical sequential and distributed models and for quantum computing. Application areas include efficient algorithms for computer graphics, distributed systems, optimization problems, game theory, and Big Data. In addition, the bounds for the design of efficient algorithms in classical complexity theory and quantum computing are explored. Furthermore, constructive applications of the limits of designing efficient algorithms in the area of classical cryptography, post-quantum cryptography, and IT security are presented.

4.2 Computer and Communication Systems

Coordination

Prof. Dr. Marco Platzner

Modules in this Focus Area

- Advanced Computer Architecture
- Advanced Distributed Algorithms and Data Structures
- Advanced Networked Systems
- Reconfigurable Computing
- VLSI Testing

Description

Computing and communication technologies are not only two classical subfields of computer science but also provide the technical infrastructure for all applications of computer science. Both subfields have developed rapidly in the last decades and especially the increasing fusion of computing and communication is the basis of many modern computer science systems.

The specialization area “Computer and Communication Systems” deals in depth and in technical detail with various aspects of modern computer and communication systems. In the case of computer systems, the focus is on the analysis and evaluation of computer architectures, systematic methods for the design and optimization of computer systems, in particular the interaction of hardware and software, as well as programming models and methods for the parallel and specialized computer architectures, which are becoming increasingly important. In communication systems, architectures, methods, and systems of modern communication technology are studied at different levels of abstraction, starting with physical transmission and ending with application design in distributed environments. Different classes of systems are covered, from classical mobile communication, ad hoc networks and vehicular communication to networking in data centers and architectures of the future Internet.

4.3 Data Science and Intelligent Systems

Coordination

Prof. Dr. Axel-Cyrille Ngonga Ngomo

Modules in this Focus Area

- Advanced Algorithms
- Advanced Distributed Algorithms and Data Structures
- Data Science for Software Engineering
- Data Science in Industrial Applications
- Explainable Artificial Intelligence
- Foundations of Knowledge Graphs
- Introduction to Description Logics
- Machine Learning for Biometrics
- Multi-Objective Optimisation
- Unsupervised Learning and Evolutionary Optimisation Using R

Description

Data-intensive systems are being used in an increasing number of application areas. In this Focus Area, algorithms and architectures underlying these systems are investigated. The Focus Area is based on two non-disjoint research areas: Data Science and Intelligent Systems.

Data Science is a young scientific discipline at the intersection of computer science, statistics, mathematics, and the engineering sciences that has become one of the most influential areas in the research landscape in recent years. It plays a significant role in the digitization and "datafication" of our society, not only in industry and research but also in private life. In academia, it is often considered a "fourth paradigm" alongside the empirical, theoretical, and computational approaches. Roughly speaking, the goal of Data Science is the development of methodological and algorithmic foundations of an automated generation of useful knowledge from data, as well as the implementation of corresponding foundations in the form of computer systems. The specialization "Data Science" equips the students with solid theoretical basic knowledge as well as practical skills, which give them the profile of a modern "Data Scientist". For this purpose, courses are offered in three directions: Mathematical and Algorithmic Foundations, Data Analytics, Software and Systems.

Of particular importance for the Focus Area are intelligent systems, ergo systems whose behavior is controlled by artificial intelligence (AI) methods and algorithms. Such systems are continuously gaining in importance, not only at the scientific level within computer science, but also in the social and societal context: autonomous or semi-autonomous systems such as service robots, self-driving cars or medical diagnostic systems are increasingly changing our private and professional lives. In addition to methodological advances and an increase in computing power through faster hardware, the rapid development of AI systems in the last decade is primarily due to a data explosion: the availability of large amounts of data or sensor-based observations from their environment enables intelligent systems to independently optimize their behavior through adaptation and learning. This Focus Area addresses important aspects of the design of intelligent systems and provides the corresponding theoretical and methodological foundations. The content ranges from topics such as machine learning to data management of structured data for explainable AI.

4.4 Security

Coordination

Prof. Dr. Eric Bodden

Modules in this Focus Area

- Advanced Distributed Algorithms and Data Structures
- Designing code analyses for large-scale software systems 1
- Designing code analyses for large-scale software systems 2
- Foundations of Cryptography
- Human Factors in Security and Privacy

- Introduction to Quantum Computation
- Machine Learning for Biometrics
- Post-Quantum Cryptography
- Privacy and Technology
- Quantum Complexity Theory
- Real World Crypto Engineering
- Usable Security and Privacy
- Web Security

Description

In all areas of life, digital technologies, such as the (Industrial) Internet of Things, Cyber-Physical Systems, Digital Automotives, Digital Health or Industry 4.0, offer immense innovation potential. However, increasing digitization requires new approaches to safely exploit this potential. In order to be able to tackle this challenge, there is a great need in industry, research and teaching for well-trained computer science experts with in-depth knowledge of IT security. In the specialization area “IT Security”, solid theoretical basic knowledge is taught in combination with practical skills. The course covers technical skills from the field of IT security (e.g. software security, formal verification, basics of modern cryptography and communication security), in which typical security vulnerabilities and attack techniques are presented and countermeasures and their effectiveness are examined.

Since security cannot be seen independently of concrete applications and different applications have different security requirements, specialized competencies in modern application fields with special security requirements (e.g., communication protocols in the mobile and automotive fields) as well as supplementary qualifications in the areas of algorithms and quantum computing are also covered.

4.5 Software Engineering

Coordination

Prof. Dr. Yasemin Acar

Modules in this Focus Area

- Concepts of Computer Science
- Data-Driven Engineering
- Data-Driven Innovation
- Data Science for Software Engineering
- Data Science in Industrial Applications
- Designing code analyses for large-scale software systems 1
- Designing code analyses for large-scale software systems 2
- Human Factors in Security and Privacy
- Model-Based Systems Engineering
- Software Architecture Design and Recovery

- Software Product Lines

Description

Software has continuously pervaded all areas of private, professional and public life. Without effective, efficient and reliable software, the world as we know it would be unthinkable. But how do you build up-to-date software in such a way that it can reliably and efficiently process even large amounts of data? And how does one build high quality software using current methods and test its properties in the most automated way possible? And how does one deal with the important human factor in software development?

In the Software Engineering module, you will learn an up-to-date, research-oriented view of these questions. You will learn about current challenges and which methods and tools software engineers can use to meet them. To this end, you will learn constructive methods for the realization of functional and non-functional aspects of software-intensive systems, as well as analytical methods with which you can sustainably test the effective implementation and goal achievement with respect to these aspects. Furthermore, you will gain in-depth knowledge about the different roles in software development and which factors in the cooperation of these roles can have which influences. Further, you will learn how to systematically record and judge these influences by means of empirical studies.

5 Modules in the Winter Semester

6 Modules in the Summer Semester

Erzeugt am 3. Juni 2024 um 16:22.