### Advanced Java programming

**A short description of the course, topics:**
The purpose of the course is to acquire knowledge on, and enhance competence in, Java Standard Edition, beyond the fundamental language concepts and standard libraries:
- Generic definitions
- Annotations
- Reflection
- Multithreading
- Memory management, garbage collection
- Input-output, serialization
- Database management and persistence: JDBC and the fundamentals of JPA
- Network programming: TCP and UDP; HTTP
- Program design principles and best practices
- Exceptions, assertions
- Logging and testing

**Literature:**

### Analysis of distributed systems

**A short description of the course, topics:**
The goal of the subject is to give an overview for the student about how can we explain the parallel behaviour by algebraic methods and Petri-nets, and how work applications based on that models in practice.

The basic concepts of the course are processes, computational processes, parallelism, operations of processes, compositions of processes and properties of processes (liveness, deadlock-free, etc.). The theory of Petri-nets is explored more partially with many modelling example. The behavioural and structural properties, methods of analysis, famed subclasses and relationships between these subclasses are investigated. We define theorems about liveness, safetyyness and reachability and present transformation, which preserve these properties. The course introduces the Petri-boxes, a special class of Petri-nets, which help us to model the program structures (sequences, branches and loops). Some tools for simulation and analysis of Petri-nets are also investigated. The second part of the course introduces the theory of algebraic models through a given example. The properties of the models, the methods of descriptions of processes and the possible compositions are examined. The denotational, operational and axiomatic semantics of the model is given and the relationships of these different descriptions are investigated. Teaching methods: There will be lectures introducing the formal specification and properties of Petri nets and algebraic models and exercises where the students will create concrete examples. There will be also programming exercises where the students can use the learned methods.

**Literature:**
Complex information systems

A short description of the course, topics:

- Concept of Information Systems
- Methodologies for Analysing and Designing Information Systems
- Concept of Enterprise Resource Systems
- Logistics as Business Process of Enterprises
- Human Resources Management
- WFMS (Workflow Management Systems)

Cloud Computing

Literature:


Introduction to data science

A short description of the course, topics:

- clustering;
- frequent pattern mining;
- linear classification and regression model: model parameters and hyper-parameters, validation, overfitting-underfitting and the bias-variance trade-off;
- introduction to prediction techniques (as black-box functions);
- data quality and pre-processing: noise, missing values, data transformation, normalization;
- the CRISP-DM process;
- recommendation techniques;
Literature:

- Jiawei Han, Micheline Kamber, Jian Pei (2011). Data Mining: Concepts and Techniques. Morgan Kaufmann.

Design of Distributed Systems

A short description of the course, topics:
Students will be able to express and verify the properties of the distributed programs using formal methods, apply different ways to create advanced compositions of simple programs, and solutions for interesting and difficult problems in a distributed way.
Dining/drinking philosophers, formal specification of distributed problems, properties of distributed systems, safety and progress properties of distributed programs, verification of safety critical properties, program compositions from components with proven properties, computing the value of an associative function, message channels, pipelined networks programming exercises where the students apply the learned methods in the practice.

Literature:


Formal semantics

A short description of the course, topics:
Introduction: motivation, approaches to semantics definitions
Translational semantics, attribute grammars and their applications
Denotational and operational semantics of expressions
Natural semantics of imperative statements
Structural operational semantics of imperative statements
Semantics of abort, nondeterministic and parallel execution
Denotational semantics of imperative statements
Domain and fixed point theory
Semantics of functional language elements
Modeling blocks and procedures
Modeling exceptions
Full abstraction

Literature:

- Kenneth Slonneger and Barry L. Kurtz: Formal Syntax and Semantics of Programming Languages (Addison Wesley Longman, 1995)
- John C. Reynolds: Theories of Programming Languages (Cambridge University Press, 1998)
**Functional languages**

**A short description of the course, topics:**
Algebraic types, type classes.
Higher-order types, existential types.
Uniqueness typing.
Dynamics, generic programming.
Purely functional data structures.
Parallel and distributed programming.
Combinators, combinator libraries.
Monadic programming.
Interactive programs, Functional Reactive Programming.
Embedded domain-specific languages.

**Literature:**

**Recommended literature:**

---

**Interactive media design and development**

**A short description of the course, topics:**
The course introduces Human–Computer Interaction (HCI) involving the study, planning, and design of the interaction between people (users) and computers.
Its aim is to understand the theoretical basics of Perception, Multimedia design, Information Visualization, Interaction Design, the Virtual Continuum, Serious Games, Tangible, Collaborative, Location-based, and Gesture-based technologies, etc.) and recent innovations in these areas.
Activities involve the exploration of emerging interactive technologies designed for demonstration, education, entertainment, navigation, narrative, support …etc. purposes and their variety of creative applications in different disciplines and user interest groups.
Students from different disciplines form groups to design and implement a specified innovative project that could well serve the basis of an industrial entrepreneurship.

**Recommended literature:**
- Horizon Reports: [http://www.nmc.org/horizonproject](http://www.nmc.org/horizonproject)
- Papers submitted to conferences:
  - iED: [http://europe.immersiveeducation.org/events/ied-europe-summit-2012](http://europe.immersiveeducation.org/events/ied-europe-summit-2012)
**Models of computation**

**A short description of the course, topics:**

The aim of the course is to provide a better understanding of the concept of computation and computational modelling by presenting different computational models. We discuss basic classical models as finite automata, pushdown automata, Turing machines and their variants (for example, register machines), partial recursive functions, random access machines, circuits, cellular automata, Petri nets. We also survey some emergent models of computation, as membrane systems and some models from DNA computing. We provide information on the computational power and efficiency of these constructs, examine their computational and descriptional complexity, and compare the different models with each other. We also discuss how these models can be used in solving theoretical and practical problems.

**Literature:**

- M. Fernandez, Models of Computation: An Introduction to Computability Theory (Undergraduate Topics in Computer Science), Springer, 2009

**Recommended literature:**


---

**Scalable enterprise applications**

**A short description of the course, topics:**

The course presents some important application domains for distributed programming, with special regard to present software industry challenges and scientific computations. After the completion of the course the students will not only understand the theoretical issues of distributed computing, but they will...
also be capable of designing and implementing distributed applications in general, and distributed object systems in particular. They will also learn common technologies used in the software industry. The following topics will be addressed (related technologies that can be used for illustration purposes are in parentheses).

Multi-tier application model: Modularization of large software systems, optimal use of distributed architectures in the design of the components (with respect to efficiency and high availability). Transactional applications backed by information systems. (Java EE, JDBC, JPA, JTA)

Remote Procedure Call: (Java RMI, EJB)
Message-based communication: (JMS, PVM/MPI)
Web-programming: Web-applications (Java servlet, JSP, JSF) , web-services (JAX-WS)
Component lookup: (JNDI, Jini).
Code mobility: (Java applet)
Grid systems: fulfilling high computational requirements.
Aspect-oriented programming: Used in the implementation of the above technologies. (AspectJ)

Literature:

Service Science

A short description of the course, topics:
Concepts and standards of Enterprise, Information and Software Architecture
Foundations of Service
Electronic Services
Service Innovation
Service Design
Which known methods and techniques are available to design services?
Service Semantics
Service Analytics
Service Optimization
Service Co-creation
Service Markets
Service Research
SOA – Service Oriented Architecture

Literature:
- Martin Op ‘t Land, Erik Proper, Maarten Waage, Jeroen Cloo, Claudia Steghuis, Enterprise
Software quality and testing

A short description of the course, topics:
Fundamentals of software testing
Fundamental test process
Testing throughout the software life cycle, Test levels,
Static techniques
Test design techniques
Specification-based or black-box techniques
State Transition Testing, Use case Testing
Structure-based or white-box techniques
Experience-based techniques
Test management
Risks and Testing
Tool support for testing
Case study

Literature:

Software Technology

A short description of the course, topics:
Purpose:
The course gives a broad overview of the process and methodologies of software development and its execution.
We cover all phases of development from requirements to maintenance and quality assurance with emphasize on architectural design.
The course tries to deliver a balanced mixture of theoretical knowledge and practical skills with currently used technologies.

Competencies delivered:
Students completing the class will understand software development process, its different strategies and methodologies.
They will be able to make sensible architectural decisions and plans well in advance using the acquired mixture of theoretical and hands-on skills.

Prerequisites:
- advanced knowledge of at least one object oriented programming language
- understanding of web technologies full stack (client, database, server...)
- (optional) project experience

Literature:
Theory of programming

**A short description of the course, topics:**

**Literature:**

Web engineering

**A short description of the course, topics:**
This curriculum introduces the students with the modern, state-of-the-art client and server side web technologies, methodologies of web engineering, the programming and design patterns, especially with the web service oriented architectures. By the end of the course the student has a global overview of the up-to-date web trends and technologies, and, with the help of them, is able to develop a web application and web information systems.

Introduction to Web Technologies and Web Engineering: specialties, characteristics, categories of web applications.
Web Architectures: multi-tier, data-centric architectures,
Requirement Analysis of Web Applications
Specialties of Large Enterprise and Small and Medium Enterprise Web Applications
Development Process of Web Applications
Model-Based Web Application Design and Development, WebML
Testing, Quality Management.
Design of Web 2.0 and Enterprise 2.0 Applications
Web Business Models
Web project management
Design of Mobile Web Applications
Semantic Web Applications, integration to Web Information Systems
Web Application Models, Cloud computing
Service Oriented Architectures, Web Information Systems

**Literature:**