SOFTWARE ENGINEERING

Seminare

- Verifikation und Testen im Modernen Software Engineering
- Automatische Detektion und Korrektur von Softwarefehlern
- Mobile Software Systems
- Advanced Programming Paradigms for Robotics
- Mobile Robotics
- Smart logistics and Manufacturing Robotics

Ziel

Praktisches Erfahren von Methoden

Meilensteine

1. **Konkretisieren des Themas**
   MS: Inhaltliche Struktur/Umfang klar - Literaturrecherche
   (bis 04.11.2015)

2. **Themenerarbeitung**
   MS: Ausarbeitung vollständig, Idee für Vortrag
   (bis 02.12.2015)

3. **Korrekturen und Verbesserungen**
   MS: Korrekturen eingearbeitet, Vortrag fertig
   (bis 13.01.2016)

4. **Vortrag**

   MS: Vortrag gehalten im Blockseminar (vmtl. 27.01.2016)

Regeln/Bewertungsmaßstäbe

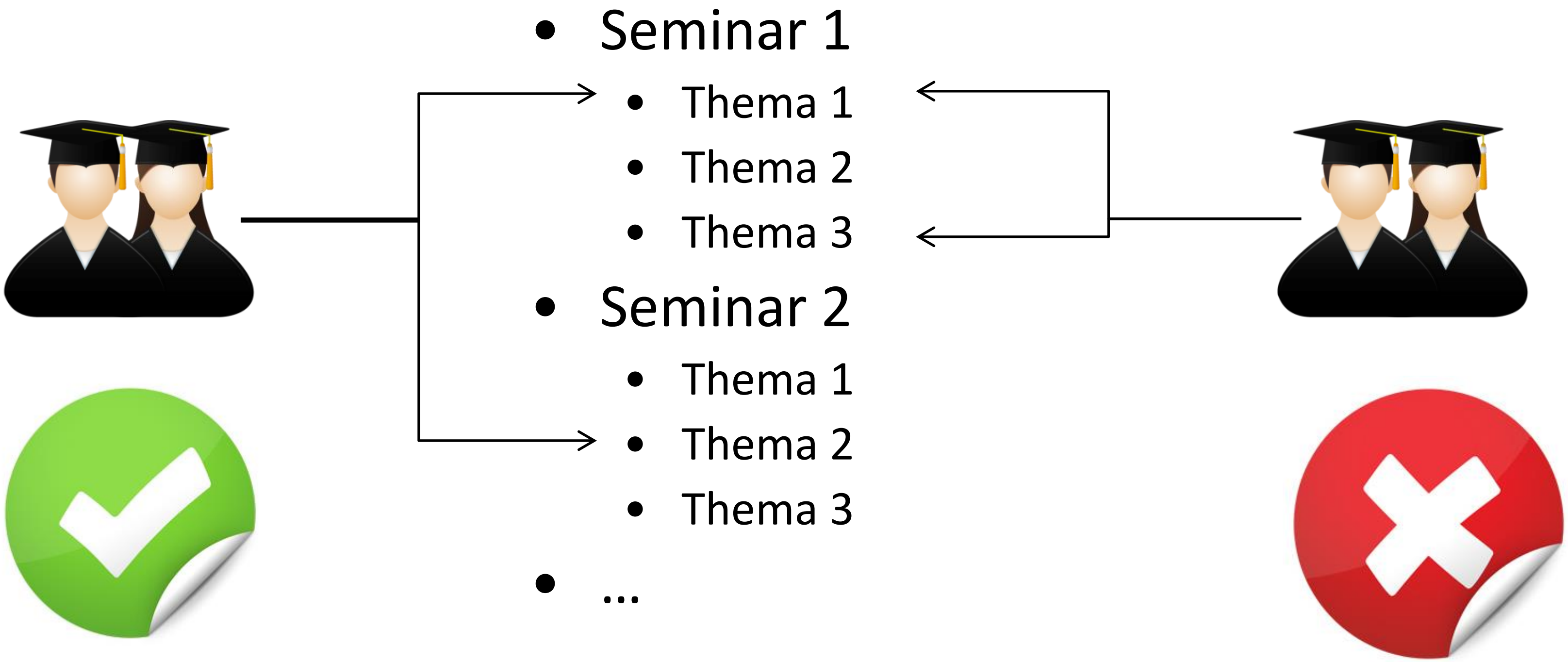
- Min. 3 Treffen mit dem Betreuer
- Ausarbeitung 14 Seiten (Springer LNCS-Style)
- Ausarbeitung folgt unserer Guideline „Wie schreibe ich ein gutes Paper“
- Inhaltliche Korrektheit der Ausarbeitung
- Ausarbeitung zwei Wochen (harte Deadline!) vor dem Blockseminar via Mail an den Betreuer
- Vortragszeit 30 min + Diskussion ~15 min
- Inhaltlich korrekter und ansprechender Vortrag
- Präsentator ist kompetent in seinem „Fachbereich“
- Aktive Teilnahme an den anderen Vorträgen wird erwartet

Seminare

Sechs Seminare zur Auswahl:

- **Verifikation und Testen im Modernen Software Engineering (VTSE)**
- **Automatische Detektion und Korrektur von Softwarefehler (ADKS)**
- **Mobile Software Systems (MSS)**
- **Advanced Programming Paradigms for Robotics**
- **Mobile Robotics (MR)**
- **Smart logistics and Manufacturing Robotics (SLMR)**

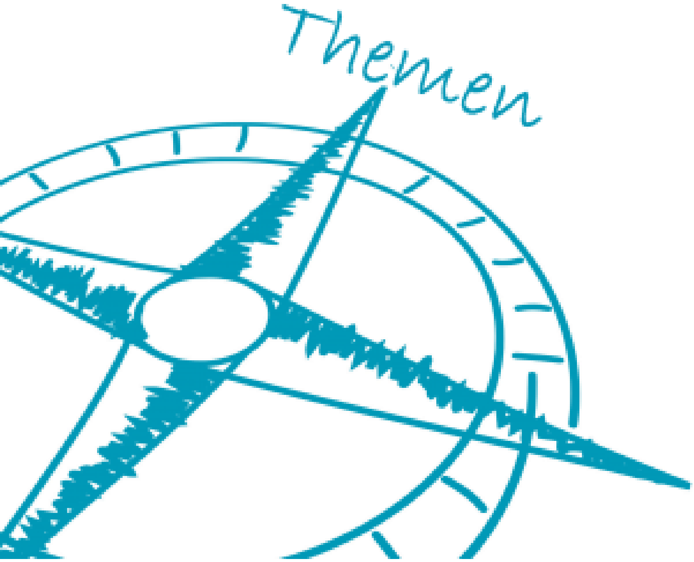
Ein Thema pro Seminar



- Seminar 1
  - Thema 1
  - Thema 2
  - Thema 3
- Seminar 2
  - Thema 1
  - Thema 2
  - Thema 3
- …

Themenvorschläge

- Vorstellung der unterschiedlichen Themen der einzelnen Seminare
- jeweils eigene Vorschläge für möglich!



Themenvorschläge

Verifikation und Testen im Modernen Software Engineering

- **Verification**
  - Model Based Verification - Bounded & Unbounded Model-Checking
  - SLAM and checking critical software behavior
  - CHESS a tool for finding Heisenbugs in Concurrent Programs
  - CBMC a bounded model checker for C-Programs
  - CPAChecker a tool for configurable software verification
  - JavaPathFinder a swiss army knife for Java verification
  - TAPAs a tool for the specification and analysis of concurrent systems
- **Program Testing**
  - The Model based Testing Approach
  - SpecExplorer and model based testing
  - Microsoft Fakes: Isolating Code Under Test
  - Microsoft Pex and Moles - Isolation and White box Unit Testing for .NET
  - Java Unittesting mit JUnit
  - Unittesting for Web applications

Themenvorschläge

Automatische Detektion und Korrektur von Softwarefehlern

- **Automatic detection of defects by analysis of**
  - source code
  - log-files
  - meta-data (e.g., performance loss)
  - data form software repositories
- **Automatic correction via**
  - check-point recovery
  - exploit redundancy in hardware and software
  - recovery shepherding
  - explicit exception handling
  - collaborative learning of software instances

Themenvorschläge

Mobile Software Systems

- **Indoor localization algorithms for smartphones**
  - Compare different algorithms for localizing unmodified smartphones
    - WiFi
    - Bluetooth
    - GAIT
    - VLC
    - Feature Tracking
- **Feature detection algorithms specialized for mobile computing**
  - Compare different algorithms
  - Evaluate different frameworks
- **Augmented advertising pillar**
  - Evaluate possibilities to build a prototype using AR Glasses
  - Project the pillar correctly in an outdoor scenario

Themenvorschläge

Advanced Programming Paradigms for Robotics (Part 1)

- **Sensors in robotics**
  - Navigation using SLAM algorithm
  - Sensor-based path planning
  - Visual servoing for grasping
- **Trajectory optimization for redundant industrial robots [theoretical work/ theoretical + practical work]**
  - What trajectory optimization problems exist? (e.g., trajectory following, planning, etc)
  - What are the state-of-the-art methods to solve them?
  - What type of smooth functions are normally used?
  - [In case of Practical work] Implement a simple smooth trajectory planning algorithm in OpenRave
- **Calibration methods for industrial robots [theoretical work]**
  - What causes imprecise robot motion?
  - What does a “good” trajectory mean?
  - What calibration ways exists?
  - How to reduce the need of often calibration?

Themenvorschläge

Advanced Programming Paradigms for Robotics (Part 2)

- **Task-specific robot design [theoretical work/ theoretical + practical work]**
  - What is the task-specific design?
  - What optimization technique are applied?
  - What type of task re-configurable robots exist?
  - [In case of Practical work] Apply one approach to optimize industrial robot structure for the sequence of tasks.
- **Combination of the Symbolic planning and Collision-free planning [theoretical work/ theoretical + practical work]**
  - What is Symbolic planning and what is collision-free planning?
  - What are the common techniques to solve these planning problems separately?
  - Why their combination is so important?
  - What are the state-of-the-art approaches for their combination?

Themenvorschläge

Mobile Robotics (Part 1)

- **Online collision-free planner.**
  Possible scenarios:
  - Crossing the road scenario in 2D (There are N parallel road lines. On every road line the cars are moving either from left to right or from right to left with different speed.)
  - Moving through a meteorite cloud in 3D

- **Offline Collision-free planning**
  Possible scenarios:
  - Find the maze exit in 2D (Dijkstra, A*)
  - Move object in the cluttered environment in 3D, e.g., piano in the cluttered room or fridge in the stairways (RRT, PRM)

Themenvorschläge

Mobile Robotics (Part 2)

- **Methods for the flocking behavior simulation in 2D/3D [theory + practice]**
  - Simulate fishes in the Aquarium
  - Add a shark with a repulsive force

- **UAV path planning and simulation using ROSPlan.**

  Possible scenarios:
  - Delivery of goods
  - Area monitoring
  - Inspection of different structures

Themenvorschläge

Smart logistics and Manufacturing Robotics (Part 1)

- **Cutting stock problem and its variants [theoretical work]**
  - Find as many Cutting-stock-like problems as possible (e.g., Knapsack problem, Bin packing problem, etc.)
  - Classify them, i.e., find commonalities and differences, e.g., input, output, constraints.
  - What are solution methods?
- **Travelling salesman problem and its variants [theoretical work]**
  - Find as many TSP-like problems as possible (e.g., Chinese postman problem, Canadian traveler problem, Vehicle Touting Problem)
  - Classify them, i.e., find commonalities and differences, e.g., input, output, constraints.
  - What are solution methods?
- **Meta-heuristics in Robotics [theory and practice]**
  - What are meta-heuristics?
  - What is the difference to heuristics or to hyper-heuristics?
  - In which robotics domain they are involved? Why?
  - Give a classification of meta-heuristics

Themenvorschläge

Smart logistics and Manufacturing Robotics (Part 2)

- **Hyper-heuristic [theory and/or practice]**
  - Advantages of applying hyper-heuristics
  - Apply one of the hyper-heuristics (by your choice) to the travelling salesman problem

- **Hybrid heuristic [theory and practice]**
  - Advantages of hybrid heuristics in comparison with simple heuristics
  - Techniques used in hybrid heuristics, their advantages and application scenarios
  - Apply one hybrid heuristic to the vehicle routing problem, job shop scheduling, bin packing problem or multiple coverage path planning

Themenvorschläge

Smart logistics and Manufacturing Robotics (Part 3)

- **Coverage path planning in T-Space**
  - The boustrophedon cellular decomposition
  - Morse-based cellular decomposition
  - Grid-based methods

- **Coverage path planning (CPP) in C-Space**
  - CPP with kinematics
  - Sampling-based CPP approaches
  - Optimal collision-free CPP