Advanced Software Engineering

Allgemeine Angaben

Kürzel	ASEN
Modulverantwortliche	Prof. Dr. Rohde, Prof. Dr. Ruckmann
Dozenten	Prof. Dr. Rohde, Prof. Dr. Ruckmann
Lehrsprache	Deutsch
Semester	3
ECTS-Punkte	5
Kontaktstunden	40
Selbststudium	85
Dauer	1 Semester
Art	Wahlpflicht
Häufigkeit	jedes Studienjahr
Gewichtung	5/180
Prüfungsleistung	**K**RS90

Stichwörter

- Software Design und Softwarearchitekturen
- Vorgehensmodelle zur Dokumentation (z.B. Arc42)
- Softwaremodellierung mit UML
- Entwurfsprinzipien und -methoden (z.B. Domain-Driven Design, Model-Driven Architecture)
- Architekturmuster (z.B. Ports & Adapter, MVC, CQRS, SOA, Microservices)
- Entwurfsmuster (z.B. Factory, Command, Observer, Strategy)

Zugangsvoraussetzungen

- Grundlagen der Informatik
- Programmieren I & II
- Software Engineering
- Angewandte Daten-Analyse

Verwendbarkeit

- Projekt-Module
- Das Modul kann als Vorbereitung zu Zertifizierung „iSAQB® Certified Professional for Software Architecture – Foundation Level" genutzt werden.

Qualifikations- und Kompetenzziele

Die Studierenden kennen die grundlegenden Konzepte und Vorgehensweisen der Softwarearchitektur und die Aufgaben eines Softwarearchitekten. Sie sind in der Lage, Softwarearchitekturen selbständig durch UML-Diagramme zu modellieren und geeignet zu dokumentieren. Sie kennen wesentliche Entwurfsprinzipien und -methoden, Architekturmuster und Entwurfsmuster und können diese in Software-Entwicklungsprojekten einsetzen und anwenden.

Lehr- und Lernmethoden

Unterschiedliche Lehr-/Lernumgebungen: Präsenzveranstaltungen, Eigenstudium; Wechselnde Lehr-/Lernmethoden: Individuelles und kooperatives Lernen, problemorientiertes und integratives Lernen, synchrones und asynchrones Lernen; Fallstudienarbeit, Expertenvorträge.

Inhalte

- Grundlagen zu Softwarearchitekturen
  - Begriffsbestimmung und Motivation
  - Aufgaben eines Softwarearchitekten
  - Vorgehen bei der Architekturentwicklung
  - Softwarearchitektur & Softwarequalität
- Dokumentation von Softwarearchitekturen
  - Bausteine, Schnittstellen, Sichten
  - Vorgehensmodelle (z.B. Arc42)
- Softwaremodellierung mit UML
  - UML-Strukturdiagramme (z.B. Komponenten-, Klassendiagramm und Einsatz- und Verteilungsdiagramm)
  - UML-Verhaltensdiagramme (z.B. Aktivitäts-, Zustands- und Sequenzdiagramm)
- Entwurfsprinzipien und -methoden
  - Entwurfsprinzipien, Kopplung und Kohäsion, SOLID-Prinzip
  - Technische Schulden, Clean Code, Coding Guidelines, Continuous Inspection
  - Domain-Driven Design, Model-Driven Architecture
- Architekturmuster
  - Datenfluss- und datenzentrische Systeme (z.B. Batch Sequential, Pipes & Filters)
  - Hierarchische Systeme (z.B. Layer, Ports & Adapter)
  - Interaktionsorientierte Systeme (z.B. MVC, MVVM)
  - Verteilte und ereignisbasierte Systeme (z.B. CQRS, Broker, Message Queues)
  - Serviceorientierte Architekturen und Microservices
  - Architekturmuster des Cloud Computing
- Entwurfsmuster
  - Beispiele für Erzeugungs-, Struktur- und Verhaltensmuster gemäß GoF (z.B. Factory, Command, Observer, Strategy etc.)

Grundlegende Literaturhinweise

GAMMA, E., R. HELM, R. JOHNSON und J. VLISSIDES, 2015. *Design Patterns. Entwurfsmuster als Elemente wiederverwendbarer objektorientierter Software*. Frechen: mitp.

GHARBI, M., A. KOSCHEL, A. RAUSCH und G. STARKE, 2018. *Basiswissen für Softwarearchitekten. Aus- und Weiterbildung nach iSAQB-Standard zum Certified Professional for Software Architecture - Foundation Level*. Heidelberg: dpunkt.verlag.

Ergänzende Literaturempfehlungen

LILIENTHAL, C., 2017. *Langlebige Software-Architekturen. Technische Schulden analysieren, begrenzen und abbauen*. Heidelberg: dpunkt.verlag.

TOTH, S., 2015. *Vorgehensmuster für Softwarearchitektur. Kombinierbare Praktiken in Zeiten von Agile und Lean*. München: Hanser.