Technology
Arts Sciences
TH Köln

# Modul
# EKS - Entwicklung komplexer Software-Systeme

Bachelor Technische Informatik 2020

Version: 1 | Letzte Änderung: 03.09.2019 11:27 | Entwurf: 0 | Status: vom Modulverantwortlichen freigegeben | Verantwortlich: Nissen

## Allgemeine Informationen

| | |
|---|---|
| **Anerkannte Lehrveranstaltungen** | EKS_Nissen |
| **Modul ist Bestandteil der Studienschwerpunkte** | NVS - Netze und Verteilte Systeme<br>SOS - Software-Systeme |
| **Dauer** | 1 Semester |
| **ECTS** | 5 |
| **Zeugnistext (de)** | Entwicklung komplexer Software-Systeme |
| **Zeugnistext (en)** | Development of Complex Software Systems |
| **Unterrichtssprache** | deutsch |
| **abschließende Modulprüfung** | Ja |

### Modulprüfung

| | |
|---|---|
| **Benotet** | Ja |
| **Frequenz** | Jedes Semester |

#### Prüfungskonzept

mündliche Prüfung, bei vielen Studenten schriftliche Klausur
Die mündliche Prüfung bzw. schriftliche Klausur stellt sicher, dass jeder Studierende auch individuell die Ziele des Learning Outcomes erreicht hat, durch Aufgaben der folgenden Typen:
Fragen zu Grundwissen über Entwurfsprinzipien, Architekturkonzepten, Testverfahren (K.3, K.9),
Anwendung von Entwurfsmustern auf gegebene Problemfälle (K.5, K.9),

Entwurf oder Erweiterung einer modularisierten Systemarchitektur mit Gewährleicstung vorgegebenen nicht-funktionaler Eigenschaften (K.1, K.3, K.4, K.5)
Erstellung geeigneter logischer Testspezifikationen und konkreter Testfälle (K.2, K.7, K.9)

## Allgemeine Informationen

### Inhaltliche Voraussetzungen

| | |
|---|---|
| **SE - Software Engineering** | Spezifikation und Modellierung von Systemen und Software mit UML, Modularisierung in Java, einfache Entwurfsmuster, grundlegende Verfahren zum Prüfen von Software, verschiedene Architekturen von Systemen und Software, Grundbegriffe der Qualitätssicherung, Kenntnisse in Versionsverwaltung |
| **PI1 - Praktische Informatik 1** | sehr gute praktische und theoretische Kenntnisse der Pragrammiersprache Java |
| **PI2 - Praktische Informatik 2** | sehr gute praktische und theoretische Kenntnisse der Pragrammiersprache Java |
| **PP - Programmierpraktikum** | sehr gute praktische und theoretische Kenntnisse der Pragrammiersprache Java |

### Kompetenzen

| Kompetenz | Ausprägung |
|---|---|
| Typische Werkzeuge, Standards und Best Practices der industriellen Praxis kennen und einsetzen | diese Kompetenz wird vermittelt |
| In Systemen denken | diese Kompetenz wird vermittelt |
| Konzepte und Methoden der Informatik, Mathematik und Technik kennen und anwenden | diese Kompetenz wird vermittelt |
| fachliche Probleme abstrahieren und formalisieren | diese Kompetenz wird vermittelt |
| Systeme entwerfen | diese Kompetenz wird vermittelt |
| Systeme realisieren | diese Kompetenz wird vermittelt |
| In vorhandene Systeme einarbeiten und vorhandene Komponenten sinnvoll nutzen | diese Kompetenz wird vermittelt |
| Systeme analysieren | diese Kompetenz wird vermittelt |
| Systeme prüfen | diese Kompetenz wird vermittelt |

| | |
|---|---|
| Informationen beschaffen und auswerten; Technische Zusammenhänge darstellen und erläutern | diese Kompetenz wird vermittelt |
| Befähigung zum lebenslangen Lernen | diese Kompetenz wird vermittelt |
| Kommunikative und interkulturelle Fähigkeiten anwenden | diese Kompetenz wird vermittelt |

## Vorlesung / Übungen

### Exemplarische inhaltliche Operationalisierung

Anhand praxisnaher Beispiele und einer Fallstudie werden typische Entwurfsmuster entwickelt und diskutiert. Zur Verbesserung der individuellen Programmierfäöhigkeiten werden Ansätze zur professionellen Code-Entwicklung vorgestellt und kritisch bewertet. Zusätzlich werden unterschideliche Verfahren zur automatierten Code-Anlayse charakterisiert, erläutert und praktisch angewendet. Unterschiedliche Ansätze zur Modularisierung von Systemen werden gegenübergestellt, deren Anwendung in Vorführungen erläutert und in Übungen praktisch angewendet.Fortgeschrittene Testverfahren werden eingeführt und in Übungen angewendet.

### Separate Prüfung

keine

## Praktikum

### Exemplarische inhaltliche Operationalisierung

Die Anwendung von Entwurfsmustern erfolgt durch die Realisierung spezifizierter Systemteile. Auf Basis einer gegebenen Spezifikation kann die Erstellung entsprechender modularisierter Architekturen erfolgen. Für ein gegebenes Programmbeispiel sollen geeignete Testverfahren ausgewählt und umgesetzt werden. Für die selbst erstellten Programme werden automatisierte Code-Reviews und statische Code-Anlaysen durchgeführt.

### Separate Prüfung

| | |
|---|---|
| **Benotet** | Nein |
| **Frequenz** | Einmal im Jahr |
| **Voraussetzung für Teilnahme an Modulprüfung** | Ja |

**Prüfungskonzept**

Die Studierenden schließen sich zu Kleingruppen zusammen.
Jede Kleingruppe absolviert mehrere Praktikumssitzungen mit zugewiesenen Laborterminen.
In jeder Sitzung werden Programmieraufgaben gelöst (K.6, K.10).
Zur Vorbereitung eines Labortermins muss ein Hausaufgabenblatt praktisch gelöst werden.
Die erarbeiteten Lösungen müssen die Studierenden vor dem Labortermin abgeben und
am Termin gegenüber dem Betreuer erläutern und verteidigen (K.16).
Wird diese Prüfung nicht bestanden, so muss eine Wiederholungsaufgabe
bis zu einem Folgetermin bearbeitet und dort präsentiert werden;
im Wiederholungsfall führt dies zum Nichtbestehen des Praktikums.
Zusätzlich wird während des Labortermins ein Anwesenheitsblatt mit weiteren Aufgaben unter
Aufsicht (und ggf. mit Hilfestellung) in einer kontrollierten Umgebung bearbeitet.
Hierdurch stellt jede Kleingruppe ihre Fähigkeit zur selbständigen Lösung
unter Beweis.
Im Praktikum werden die folgenden Typen von Aufgaben bearbeitet:
Umsetzung von Entwurfsmustern für ein gegebenes Problem (K.3, K.5, K.6),
Entwurf und Realisierung einer modularisierten Systemarchitektur für ein gegebenes Anwendungsszenario (K.1, K.3, K.4, K.5, K.6),
Anwendung fortgeschrittener Testverfahren auf Programmcode (K.3, K.7),
Analyse eines Programms unter Verwendung von automatischen Code-Review und statischer Code-Analyse (K.4, K.7) und
Entwurf und Realisierung eines Pogramms unter Verwendung fortgeschrittener Java-Konzepte (K.5, K.6).