Technology
Arts Sciences
TH Köln

Lehrveranstaltungshandbuch EKS

Entwicklung komplexer SW-Systeme

Version: 1 | Letzte Änderung: 03.09.2019 11:28 | Entwurf: 0 | Status: vom verantwortlichen Dozent freigegeben

Allgemeine Informationen

Langname	Entwicklung komplexer SW-Systeme
Anerkennende LModule	EKS_BaTIN
Verantwortlich	Prof. Dr. Hans Nissen Professor Fakultät IME
Gültig ab	Wintersemester 2022/23
Niveau	Bachelor
Semester im Jahr	Wintersemester
Dauer	Semester
Stunden im Selbststudium	60
ECTS	5
Dozenten	Prof. Dr. Hans Nissen Professor Fakultät IME

Literatur

E. Gamma, R. Helm, R. Johnson, J. Vlissides: Design Patterns, MITP Verlags GmbH & Co. KG, 2015.

R. C. Martin: Clean Code: A Handbook of Agile Software Craftsmanship, Prentice Hall, 2008.

S. McConnell: Code Complete, Microsoft Press, 2. Auflage, 2004.

M. Fowler: Refactoring: Improving the Design of Existing Code. Addison-Wesley Verlag, 2. Auflage, 2018.

G. Oelmann: Modularisierung mit Java 9, dpunkt Verlag, 2018.

R.S. Hull, K. Pauls, S. McCulloch, D. Savage: OSGi in Action, Manning Publications, 2011.

G. Wütherich, N. Hartmann, B. Kolb, M. Lübken: Die OSGi Service Plattform, dpunkt Verlag, 2008.

A. Spillner, T. Linz: Basiswissen Softwaretest, dpunkt Verlag, 5. Auflage, 2012

P. Liggesmeyer: Software-Qualität: Testen, Analysieren und Verifizieren von Software, Spektrum Akademischer Verlag, 2. Auflage, 2009.

H.M. Sneed, M. Winter: Testen objektorientierter Software, Hanser Verlag, 2001.

Abschlussprüfung

Voraussetzungen	Spezifikation und Modellierung von Systemen und Software mit UML, Modularisierung in Java, einfache Entwurfsmuster, grundlegende Verfahren zum Prüfen von Software, verschiedene Architekture von Systemen und Software, Grundbegriffe der Qualitätssicherung, Kenntnisse in Versionsverwaltung, sehr gute praktische und theoretische Kenntnisse der Pragrammiersprache Java
Unterrichtssprache	deutsch
separate Abschlussprüfung	Ja

Details	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, Anwendung von Entwurfsmustern auf gegebene Problemfälle, Entwurf oder Erweiterung einer modularisierten Systemarchitektur mit Gewährleicstung vorgegebenen nicht-funktionaler Eigenschaften, Erstellung geeigneter logischer Testspezifikationen und konkreter Testfälle
Mindeststandard	Mindestens 50% der möglichen Gesamtpunktzahl.
Prüfungstyp	mündliche Prüfung, strukturierte Befragung

− Vorlesung / Übungen

Lernziele

Zieltyp	Beschreibung
Kenntnisse	Entwurfsmuster
Kenntnisse	Modularisierungsprinzipien
Kenntnisse	professionelle Code-Entwicklung
Kenntnisse	fortgeschrittene Java-Konzepte
Kenntnisse	Modul-orientierte Architekturprinzipien
Kenntnisse	komplexere Testverfahren
Fertigkeiten	Entwurfsmuster anwenden und beurteilen
Fertigkeiten	Ansätze zur professionellen Code-Entwicklung anwenden und beurteilen
Fertigkeiten	Verfahren zur automatisierten Code-Anlayse anwenden und die Ergebnisse interpretieren
Fertigkeiten	modularisierte Architekturen entwerfen und realisieren
Fertigkeiten	komplexe Testverfahren einsetzen

Aufwand Präsenzlehre

Typ	Präsenzzeit (h/Wo.)
Vorlesung	2
Übungen (ganzer Kurs)	1
Übungen (geteilter Kurs)	1
Tutorium (freiwillig)	0

Besondere Voraussetzungen

keine

Begleitmaterial	elektronische Vortragsfolien zur Vorlesung , elektronische Arbeitsblätter zu Übungen
Separate Prüfung	Nein

− Praktikum

Lernziele

Zieltyp	Beschreibung
Fertigkeiten	Entwurfsmuster in Programmcode umsetzen
Fertigkeiten	modularisierte Architekturen für umfangreiche Anwendungen erstellen
Fertigkeiten	automatisierten Code-Review und statische Code-Anlayse anwenden
Fertigkeiten	Testverfahren auswählen und auf Programme anwenden

Aufwand Präsenzlehre

Typ	Präsenzzeit (h/Wo.)
Praktikum	1
Tutorium (freiwillig)	0

Besondere Voraussetzungen

keine

Begleitmaterial	elektronische Vortragsfolien zur Vorlesung , elektronische Übungsaufgabensammlung
Separate Prüfung	Ja

Separate Prüfung

Prüfungstyp	praxisnahes Szenario bearbeiten (z.B. im Praktikum)

Details	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.
Mindeststandard	Erfolgreiche Teilnahme an allen Laborterminen, d.h. insbesondere selbstständige (ggf. mit Hilfestellung) Lösung der Praktikumsaufgaben.