Technology
Arts Sciences
TH Köln

Lehrveranstaltung
BVS2 - Betriebssysteme und Verteilte Systeme 2

Version: 5 | Letzte Änderung: 01.04.2022 09:46 | Entwurf: 0 | Status: vom verantwortlichen Dozent freigegeben

Allgemeine Informationen

Langname	Betriebssysteme und Verteilte Systeme 2
Anerkennende LModule	BVS2_BaET, BVS2_BaTIN
Verantwortlich	Prof. Dr. Cartsten Vogt Professor Fakultät IME
Niveau	Bachelor
Semester im Jahr	Sommersemester
Dauer	Semester
Stunden im Selbststudium	60
ECTS	5
Dozenten	Prof. Dr. Cartsten Vogt Professor Fakultät IME
Voraussetzungen	prozedurale Programmierung Architektur von Digitalrechnern (Grundkenntnisse) Internetprotokolle (Grundkenntnisse) Sämtliche Inhalte von BVS1
Unterrichtssprache	englisch
separate Abschlussprüfung	Ja

Abschlussprüfung

Details

Die Studierenden sollen in einer schriftlichen Klausur folgende Kompetenzen nachweisen: 1.) Sicherer Umgang mit grundlegenden Begrifflichkeiten, Konzepten und Techniken, 2.) Erkennung und Bewertung von Auswirkungen strategischer Entscheidungen bei der Implementierung und Ausführung von Systemsoftware, 3.) Anwendung programmiersprachlicher und abstrakterer Konstrukte zur Lösung von Anwendungsproblemen bei der nebenläufigen und verteilten Programmierung. Typische Aufgabenformen zu 1.) sind Multiple-Choice-Fragen, Lückentexte, Bewertung von Aussagen

hinsichtlich ihrer Korrektheit, zu 2.) das Durchspielen typischer Szenarien unter bestimmten Annahmen über die Systemsoftware mit daraus abgeleiteten Bewertungen und zu 3.) Lösung kleinerer umgangssprachlich formulierter Probleme durch Programmstücke oder in abstrakterer Form.

Mindeststandard

Mindestens 50% der möglichen Gesamtpunktzahl.

Prüfungstyp

Die Studierenden sollen in einer schriftlichen Klausur folgende Kompetenzen nachweisen: 1.) Sicherer Umgang mit grundlegenden Begrifflichkeiten, Konzepten und Techniken, 2.) Erkennung und Bewertung von Auswirkungen strategischer Entscheidungen bei der Implementierung und Ausführung von Systemsoftware, 3.) Anwendung programmiersprachlicher und abstrakterer Konstrukte zur Lösung von Anwendungsproblemen bei der nebenläufigen und verteilten Programmierung. Typische Aufgabenformen zu 1.) sind Multiple-Choice-Fragen, Lückentexte, Bewertung von Aussagen hinsichtlich ihrer Korrektheit, zu 2.) das Durchspielen typischer Szenarien unter bestimmten Annahmen über die Systemsoftware mit daraus abgeleiteten Bewertungen und zu 3.) Lösung kleinerer umgangssprachlich formulierter Probleme durch Programmstücke oder in abstrakterer Form.

Vorlesung / Übungen

Lernziele

Kenntnisse

Kooperation
Client-Server-Modell
Beispiele: Namens- und Dateidienste
geschichtete Architekturen
Peer-to-Peer-Modell
prozedurale Kooperation: Remote Procedure Call
objektorientierte Kooperation
Remote Method Invocation
objektorientierte Middleware
Web-basierte Dienste
dynamische Web-Seiten
Web Services

Implementierung von Software-Nebenläufigkeit
Verwaltung und Steuerung von Prozessen
Dispatching und Scheduling
Exceptions und Interrupts
Speicherkonzepte
Komponenten der Speicherhierarchie
Swapping
Virtueller Speicher
Prozesse in Verteilten Systemen
Lastverteilung, Fehlertoleranz, Synchronisation

Dateisysteme
logische und reale Strukturen
lokale Dateisysteme
Implementierung von Verzeichnissen
Organisation der Festplatte

Leistungssteigerung und Fehlertoleranz
verteilte Dateisysteme
File Server und Name Server
Verteilte Dateibäume
Caching und Replikation

Dienste in verteilten Systemen
Grundlagen von Cloud Computing und Web Services
Apache-basierte Systeme
kommerziell verfügbare Systeme

Fertigkeiten

Beurteilung verschiedener Verfahren und Techniken zum Prozessor-Scheduling, zur Verwaltung von Speicherhierarchien, zur Implementierung lokaler und verteilter Dateisysteme

---

Programmierung von und mit Diensten in lokalen und verteilten Systemen

Aufwand Präsenzlehre

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

Separate Prüfung

keine

Praktikum

Lernziele

Kenntnisse

C-Funktionen der UNIX/Linux-Programmierschnittstelle zur Kommunikation und Kooperation lokal und im Internet
durch Nutzung von Shared Memory, Message Queues und Sockets
durch Remote Procedure Call

Java-Techniken zur Kommunikation und Kooperation

Web Services: SOAP, REST

ggf. andere (wird kurzfristig festgelegt)

Fertigkeiten

Anwendung der unter "Kenntnisse (fachliche Inhalte)" genannten Aspekte auf praxisbezogene Szenarien durch selbstständige Arbeit in kleinem Team.

Aufwand Präsenzlehre

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

Separate Prüfung

Prüfungstyp

praxisnahes Szenario bearbeiten (z.B. im Praktikum)

Details

Die Studierenden schließen sich zu Kleingruppen zusammen. Jede Kleingruppe absolviert mehrere "Praktikumsrunden" mit zugewiesenen Laborterminen. In jeder Runde werden Programmieraufgaben gelöst.
Zur Vorbereitung eines Labortermins muss ein "Vorbereitungsblatt" praktisch gelöst werden. Die dabei erworbenen Kenntnisse werden zu Beginn des Termins geprüft (kurzer schriftlicher Eingangstest, persönliches Gespräch mit dem Betreuer). Wird diese Prüfung nicht bestanden, so muss ein Folgetermin wahrgenommen werden; im Wiederholungsfall führt dies zum Nichtbestehen des Praktikums. Im Erfolgsfall wird ein "Laborarbeitsblatt" mit weiteren Aufgaben unter Aufsicht (und ggf. mit Hilfestellung) bearbeitet.

Mindeststandard

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