Technology
Arts Sciences
TH Köln

Course
BVS2 - Operating Systems and Distributed Systems 2

Version: 5 | Last Change: 01.04.2022 09:46 | Draft: 0 | Status: vom verantwortlichen Dozent freigegeben

General information

Long name	Operating Systems and Distributed Systems 2
Approving CModule	BVS2_BaET, BVS2_BaTIN
Responsible	Prof. Dr. Cartsten Vogt Professor Fakultät IME
Level	Bachelor
Semester in the year	summer semester
Duration	Semester
Hours in self-study	60
ECTS	5
Professors	Prof. Dr. Cartsten Vogt Professor Fakultät IME
Requirements	procedural programming architecture of a digital computer (basic knowledge) Internet protocols (basic knowledge) full content of BVS1
Language	English
Separate final exam	Yes

Final exam

Details

Students shall prove that they can 1.) explain and apply fundamental terms, concepts, and techniques, 2.) identify and assess the impact of strategic decisions in the implementation and execution of system software and 3.) apply programming and more abstract concepts to solve application problems in the field of concurrent and distributed programming. Typical types of assignments are 1.) multiple choice questions, fill-in-the-blank texts,

assessment of statements, 2.) play through typical scenarios under certain assumptions about the system software and evaluate the findings and 3.) write program code or develop a solution in a more abstract form to solve given problems of limited size.

Minimum standard

At least 50% of the total number of points.

Exam Type

Students shall prove that they can 1.) explain and apply fundamental terms, concepts, and techniques, 2.) identify and assess the impact of strategic decisions in the implementation and execution of system software and 3.) apply programming and more abstract concepts to solve application problems in the field of concurrent and distributed programming. Typical types of assignments are 1.) multiple choice questions, fill-in-the-blank texts, assessment of statements, 2.) play through typical scenarios under certain assumptions about the system software and evaluate the findings and 3.) write program code or develop a solution in a more abstract form to solve given problems of limited size.

Lecture / Exercises

Learning goals

Knowledge

cooperation
client-server model
examples: naming and file services
layered architectures
peer-to-peer model
procedural cooperation: remote procedure call
object-oriented cooperation
remote method invocation
object-orientierte middleware
web-based services
dynamic web pages
web services

---

implementation of software concurrency
management of processes
dispatching and scheduling
exceptions and interrupts
storage concepts
components of the storage hierarchy
swapping
virtual storage
processes in distributed systems
load distribution, fault tolerance, synchronization

---

file systems
logical and real structures
local file systems
implementation of directories
organisation of the hard disk
performance enhancement and fault tolerance

distributed file systems
file server and name server
distributed directory trees
caching and replication

Services in distributed systems
fundamentals of cloud computing and web services
Apache-based systems
commercially available systems

Skills

assess various strategies and techniques for processor scheduling, for storage hierarchy management and for the implementation of file systems in local and distributed environments

---

programming of and with services in local and distributed systems

Expenditure classroom teaching

Type	Attendance (h/Wk.)
Lecture	2
Exercises (whole course)	1
Exercises (shared course)	1
Tutorial (voluntary)	0

Separate exam

none

Practical training

Learning goals

Knowledge

C functions of the UNIX/Linux programming interface to communicate and cooperate locally and in the Internet
by using shared memory, message queues, and sockets
by using Remote Procedure Call

Java techniques for communication and cooperation

web services: SOAP, REST

others as appropriate (to be determined on short notice)

Skills

application of the aspects listed above to real-world scenarios in small teams

Expenditure classroom teaching

Type	Attendance (h/Wk.)
Practical training	1
Tutorial (voluntary)	0

Separate exam

Exam Type

working on practical scenarion (e.g. in a lab)

Details

Students work in small teams. Each team completes multiple "rounds" with assigned appointments in the lab. In each round, programming assigments are solved.

For the preparation of a laboratory appointment a "preparation sheet" has to be solved. The acquired knowledge will be tested at the beginning of the appointment (short written entrance test, interview with the supervisor). In case of failure, a follow-up appointment must be taken; in case of multiple failures, the student will be excluded from the lab. In case of success, a "laboratory work sheet" with further tasks will be worked on under supervision (and, if necessary, with assistance).

Minimum standard

Successful participation in all laboratory appointments, i.e. in particular independent solution (or with some assistance if necessary) of the programming assignments.