Softwaresysteme 1 — SOS 1

Grundlagen von Betriebssystemen

Jürgen Kleinöder
Wolfgang Schröder-Preikschat

Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)

www4.informatik.uni-erlangen.de

...---...

Sommersemester 2006

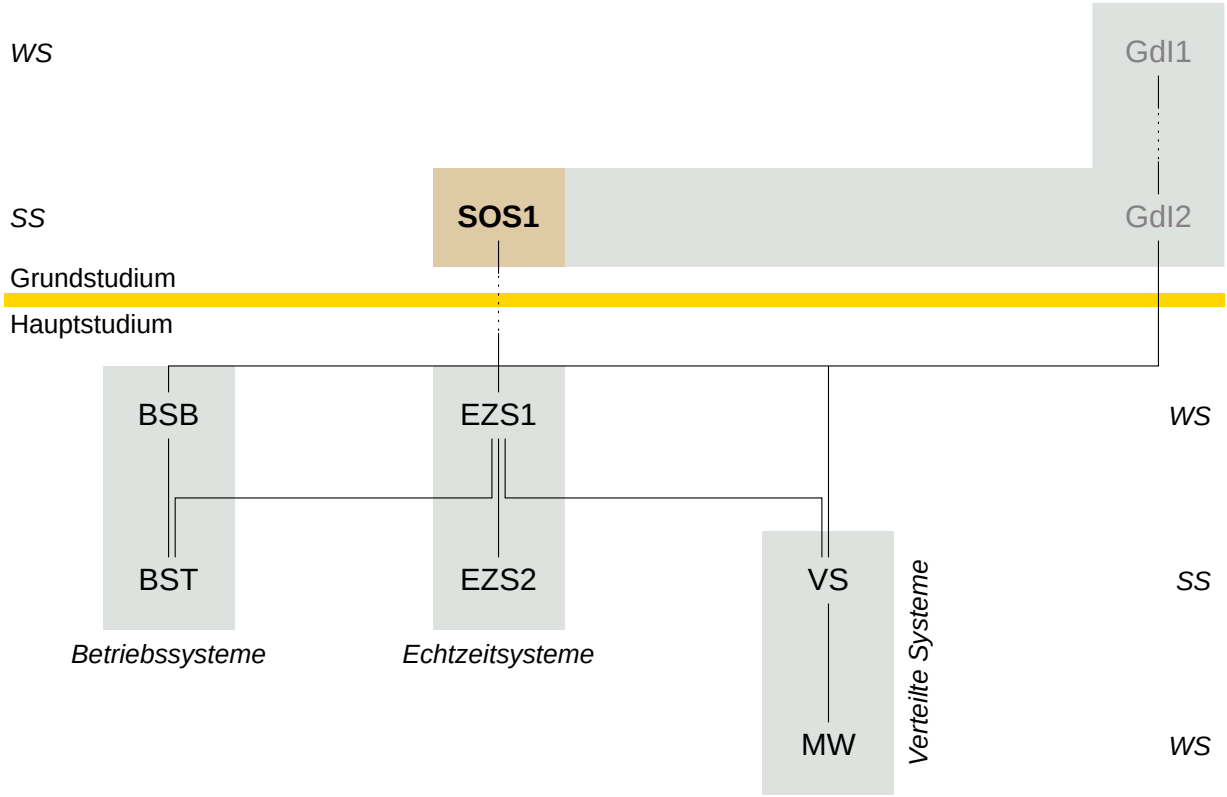
Überblick

Säule im Grundstudium Informatik, 2. – 4. Semester

SOS 1	☞	Betriebssysteme	☞	I4
SOS 2	☞	Datenbanksysteme	☞	I6
SOS 3	☞	Softwaretechnik	☞	I11

Vermittlung der Grundlagen von (großen) Softwaresystemen.

Sockel für die Kernlehrveranstaltungen von I4



Integrierte Lehrveranstaltung

$$\text{Termine} \left\{ \begin{array}{ll} \text{Vorlesung} & 2 \\ \text{Übung} & 1 \\ \text{Rechner} & 1 \end{array} \right\} \equiv 4 \times 1{,}5 = 6 \text{ Zeitstunden wöchentlich}$$

Vor-/Nacharbeit

- N Stunden wöchentlich: $0 \leq N \leq (162 - X)$
- $X \ll 162$ ist das Zeitstundenäquivalent anderer „Pflichten“

Softwaresysteme ist Kernstoff der Informatik

Diplom

- Informatik, I & K
- Mathematik (Wahlfach Informatik), Technomathematik
- Wirtschaftsinformatik

Bachelor

- Computational Engineering (CE)

Lehramt Informatik

- Gymnasium

Lernziele

Vorgänge innerhalb von Rechensystemen ganzheitlich verstehen

- Zusammenspiel
$$\left\{\begin{array}{ccc} \text{Hardware} & \leftrightarrow & \text{Software} \\ & \vdots & \\ \text{Anwendung} & \leftrightarrow & \text{Betriebssystem} \\ & \vdots & \\ \text{Treiber} & \leftrightarrow & \text{Gerät} \end{array}\right\} \text{begreifen}$$

Imperative Systemprogrammierung (in Grundzügen) selbst erleben

- im Kleinen praktizieren ☞ Dienstprogramme
- im Großen erfahren ☞ Betriebssysteme

Lehrinhalte

Vorlesung $\mapsto$ Vorstellung und detailierte Behandlung des Lehrstoffs

- Organisation (der Systemsoftware) von Rechensystemen
- Grundlagen von Betriebssystemen
- maschinennahe Programme

Übung $\mapsto$ Vertiefung, Besprechung der Übungsaufgaben, Tafelübungen

- Systemprogrammierung in C
- Systemprogramme, -aufrufe, -funktionen von UNIX

Inhaltsüberblick

Teil A $\mapsto$ SPiC[a]

1. Organisation
2. Einführung in C
3. Programm $\mapsto$ Prozess (UNIX)

Teil B $\mapsto$ Überblick

4. Einleitung
5. Rechnerorganisation
6. Betriebsarten
7. Abstraktionen (UNIX)
8. Zwischenbilanz

Teil C $\mapsto$ Vertiefung

9. Einplanung
10. Einlastung
11. Synchronisation
12. Verklemmungen
13. Adressräume
14. Arbeitsspeicher
15. Dateisysteme

[a] Systemnahe Programmierung in C

Erforderliche Grundkenntnisse

Algorithmik $\mapsto$ Grundlagen strukturierter Programmierung

- Datentypen, Kontrollkonstrukte, Prozeduren
- statische und dynamische Datenstrukturen
- „Programmierung im Kleinen“

Technische Informatik $\mapsto$ Grundlagen der Rechnerorganisation

- „von Neumann Architektur“
  - Operationsbefehle, Befehlsoperanden, Adressierungsarten
  - Unterbrechungssteuerung (Pegel kontra Flanke)
  - Assemblerprogrammierung
- CPU, DMA, FPU, IRQ, MCU, MMU, NMI, PIC, TLB

Vorlesungsbetrieb und Lehrmaterialien

Vorlesungstermine ab 24.04.

Montag	10:15 – 11:45	H7
Donnerstag	16:00 – 17:30	H7

Sondertermin am 26.04.

Mittwoch	16:00 – 17:30	H7

Handzettel (engl. *handout*) sind verfügbar wie folgt:

1. `www4.informatik.uni-erlangen.de/Lehre/SS06/V_SOS1`
   - der Verweis führt zu den Folien zum Vorlesungsstoff
2. Gutscheinverkauf zum Bezug von Folienkopien, Kosten 5 EUR
   - die Kopien werden vor der Vorlesung ausgegeben

Fachbegriffe der Informatik (Deutsch $\leftrightarrow$ Englisch)

- `http://www.babylonia.ork.uk`

Ergänzende Literatur

- B. W. Kernighan, D. M. Ritchie. *The C Programming Language*. Prentice-Hall, Inc., second edition, 1988. ISBN 0-13-110362-8 (paperback) 0-13-110370-9 (hardback).
- J. Nehmer, P. Sturm. *Systemsoftware: Grundlagen moderner Betriebssysteme*. dpunkt.Verlag GmbH, zweite Edition, 2001. ISBN 3-89864-115-5.
- A. Silberschatz, P. B. Galvin, G. Gagne. *Operating System Concepts*. John Wiley & Sons, Inc., sixth edition, 2001. ISBN 0-471-41643-2.
- A. S. Tanenbaum. *Structured Computer Organization*. Prentice-Hall, Inc., fourth edition, 1999. ISBN 0-13-095990-1.

Bedeutung von Tafel- und Rechnerübungen

Tafelübungen $\leadsto$ „*learning by exploring*“

- Besprechung der Übungsaufgaben, Skizzierung von Lösungswegen
- Vertiefung des Vorlesungsstoffes, Klärung offener Fragen

Rechnerübungen $\leadsto$ „*learning by doing*“

- selbständiges Bearbeiten der Übungsaufgaben am Rechner
- Hilfestellung beim Umgang mit den Entwicklungswerkzeugen
- der Rechner ist kein Tafelersatz, die Betreuung verläuft eher passiv

☞ „Wieso, weshalb, warum? Wer nicht fragt, bleibt dumm!“

Leistungskontrolle

unbenoteter Schein: mind. 60 % aller Übungspunkte sind zu erreichen

- obligatorisch für Informatik (Diplom, Lehramt (Gym.)), I & K, CE
- empfohlen für Wirtschaftsinformatik (Diplom)

Ex/Testat: Teil des Scheins, einer von zwei Terminen ist verpflichtend

- 1. Termin: mindestens 40 % der Punkte sind zu erreichen
- 2. Termin: wie bei 1., falls entschuldigt gefehlt, 50 % sonst

studienbegleitende Prüfung: schriftlich (Klausur)

- Zulassung mit Schein und bei bestandener Ex, sofern obligatorisch

Ex/Testat und Klausur

Ex/Testat: Mehrfachauswahl (engl. *multiple choice*), 45 Minuten

- abgefragt wird Stoff von Vorlesung und Übung (Programmbeispiel)
- Termine im Zeitraum Juni/Juli, Ankündigung eine Woche vorher

Klausur: schriftliche Prüfung, 120 Minuten

- geprüft wird Stoff von Vorlesung und Übung (Programmbeispiele)
- Klausurtermin im Zeitraum September/Oktober

aktive Mitarbeit machen Ex/Testat und Klausur „leicht“

- Programme zwar im Team entwickeln, aber selbst zum Laufen bringen

www4.informatik.uni-erlangen.de/*

Dozenten

- Jürgen Kleinöder (~jklein)
- Wolfgang Schröder-Preikschat (~wosch)

Mitarbeiter

- Christian Wawersich (~wawi)

Fragen...

?

Übungen

- Praxisnahe Vertiefung des Vorlesungsstoffs
- Systemnahe Programmierung in C am Beispiel UNIX
  - ➤ Systemprogramme
  - ➤ Systemschnittstellen
  - ➤ Funktionsbibliotheken
- 3 Ziele
  - ◆ Einführung in die prozedurale Programmierung
  - ◆ Praktische Realisierung kleinerer "echter" Anwendungsprogramme
  - ◆ Umgang mit Systemabstraktionen und -schnittstellen in der Praxis
- Die Übungen sind thematisch an den Vorlesungsverlauf angelehnt, sie sind aber keine starre Wiederholung von Vorlesungsstoff sondern gehen oft darüber hinaus

Übungen (2)

! Man kann die meisten Übungsaufgaben auch ohne das Wissen aus der Vorlesung programmieren

aber das ist nicht Informatik — das ist "Manuals wälzen und programmieren"

- ➤ das mag mühsam sein und viel Zeit kosten — ist aber aus Sicht eines Universitätsstudiums wenig interessant

➥ nur mit dem Vorlesungsstoff im Hinterkopf und durch die Reflektion zwischen Konzepten und praktischer Erfahrung kann man ein Verständnis für die Zusammenhänge entwickeln

- Beginn: 2. Mai 2006 (2. Vorlesungswoche)
- Verantwortlich:
  - ➤ Jürgen Kleinöder
  - ➤ Christian Wawersich
- Übungsleiter:
  - ➤ Stefan Kempf, Jens Schedel, Chris Schwemmer, Isabella Thomm

Bedeutung von Tafel- und Rechnerübungen

Tafelübungen

- Besprechung der Übungsaufgaben, Skizzierung von Lösungswegen
- Diskussion von Problemen
- Vertiefung des Vorlesungsstoffs, Klärung von Fragen
- Folien aus den Tafelübungen stehen auf der Übungs-Webseite

Rechnerübungen

- selbständiges Bearbeiten der Übungsaufgaben am Rechner
- Hilfestellung beim Umgang mit den Entwicklungswerkzeugen
- Übungsleiter steht für Fragen zur Verfügung
- freie Gruppenwahl

Programmieraufgaben

- Programmierung alleine oder in 2er-Gruppen
  (in der Aufgabenstellung jeweils angegeben)
- Abgabetermine sind strikt einzuhalten
- Lösungen werden automatisch auf Abschreiben überprüft
  - ➤ Strukturanalyse und Test auf strukturelle Ähnlichkeit
  - ➤ Treffer werden manuell gesichtet - abgeschriebene Lösungen = 0 Punkte
- Lösungen werden vom Übungsleiter bepunktet
- Lösungen müssen jederzeit in der Tafelübung präsentiert werden können
  - ➤ unentschuldigte Abwesenheit = 0 Punkte auf die Lösung
- In Zweifelsfällen entscheidet Rücksprache mit dem Übungsleiter

Schein, Prüfung

- Schein durch Bearbeitung der Übungsaufgaben und "Ex"/Miniklausur
- Prüfung (Klausur)
  - ➤ Termin voraussichtlich Donnerstag, 14. September 2006
  - ➤ Dauer 120 min
  - ➤ Inhalt: Fragen zum Vorlesungsstoff + Programmieraufgabe
  - ➤ Bestehensquote bislang immer > 75 %, meist > 80 %

Prüfungsvorbereitung

- Programmieren erfordert Erfahrung — Lernen alleine hilft nicht viel
  - ➤ Programmierteil ist ohne die Programmierpraxis aus den Übungen erfahrungsgemäß sehr schwer, sonst gut zu schaffen
- Alte Klausuren bearbeiten
  - ➤ alle früheren Klausuren sind auf der Web-Seite abrufbar
- Musterlösungen?
  - ➤ wir veröffentlichen keine Musterlösungen (Durchsehen von Musterlösungen bringt nicht viel)
  - ➤ wir empfehlen zur Prüfungsvorbereitung alte Klausuren selbst zu bearbeiten, im Kommilitonenkreis zu diskutieren und Probleme dann mit uns oder im Forum zu diskutieren
  - ➤ in den letzten beiden Übungswochen besprechen wir in den Übungen Fragen zu alten Klausuraufgaben
  - ➤ wir stehen auch im August und September jederzeit für Fragen zur Verfügung

Übungsanmeldung

- Beginn
  - 2. Mai (2. Vorlesungswoche)
  - Teilnehmer der Montagübungen bitte auf andere Termine ausweichen
- Termine
  - Daten auf der Übungs-Webseite sind bislang nur Vorplanungen
  - Verbindliche Termine ab Mittwoch vormittag
- Anmeldung
  - ab Mi. 26.04.2006 17:45
  - über Web-Anmeldesystem W.A.S
  - Link auf der Übungs-Webseite
  - Bei der Anmeldung Festlegung der Tafelübungsgruppe (ACHTUNG: die Partner der 2er-Gruppen bei den Programmieraufgaben müssen in der gleichen Tafelübung angemeldet sein!)
  - Ummelden ist möglich (solange Plätze in der Ziel-Gruppe frei sind)

… Verschiedenes

- spezielle Übungsgruppe für Informatiker, CE'ler, I&K'ler
  - für Leute mit viel Vorwissen
  - Gruppe T15: Montag 14-16 Uhr
- spezielle Übungsgruppe zur Besprechung von Problemen
  - ausführlichere Wiederholung und Erläuterungen
  - Gruppe T16: Freitag 14-16 Uhr
- Evaluation