Grundzüge der Betriebssysteme

Felix Gärtner

Fachgebiet Betriebssysteme

Fachbereich Informatik

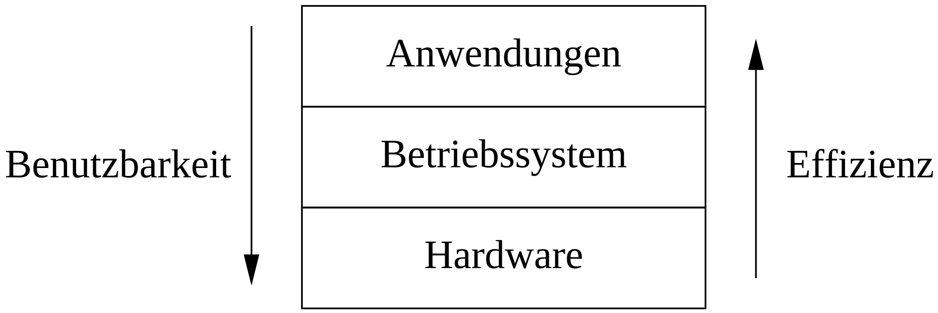
Technische Universität Darmstadt

Worum geht es hier?

1. Betriebssysteme — Grundlagen (ca. 45 Min.)
   - Was ist ein Betriebssystem?
   - Grundkonzepte von Betriebssystemen
   - Prozeßmanagement
   - Benutzer und Zugriffsrechte
   - Prozeßkontrolle
2. Planspiel — Grundlagen (ca. 15 Min.)
   - Motivation
   - Systemarchitektur
   - Spielablauf

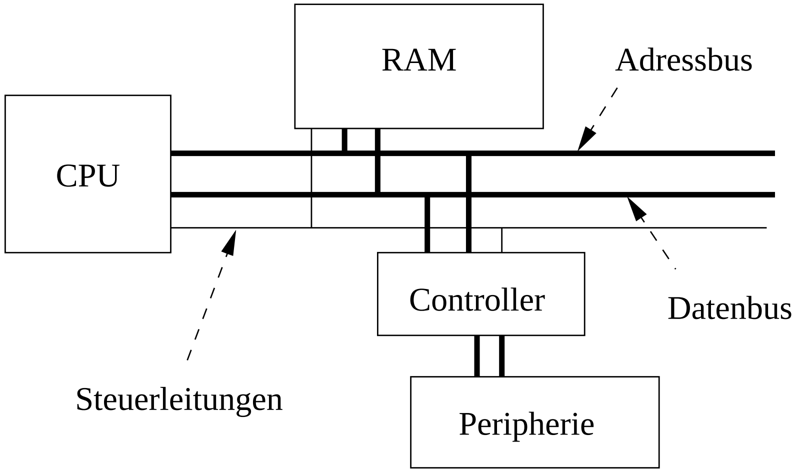
Was ist ein Betriebssystem?

- Betriebssystem vs. Anwendungsprogramme



- Betriebssystem als virtuelle Maschine (*top-down*)
- Betriebssystem als Ressourcen-Manager (*bottom-up*)
  - Schutz von Ressourcen
  - Managen von Ressourcen

Hardware-Grundlagen



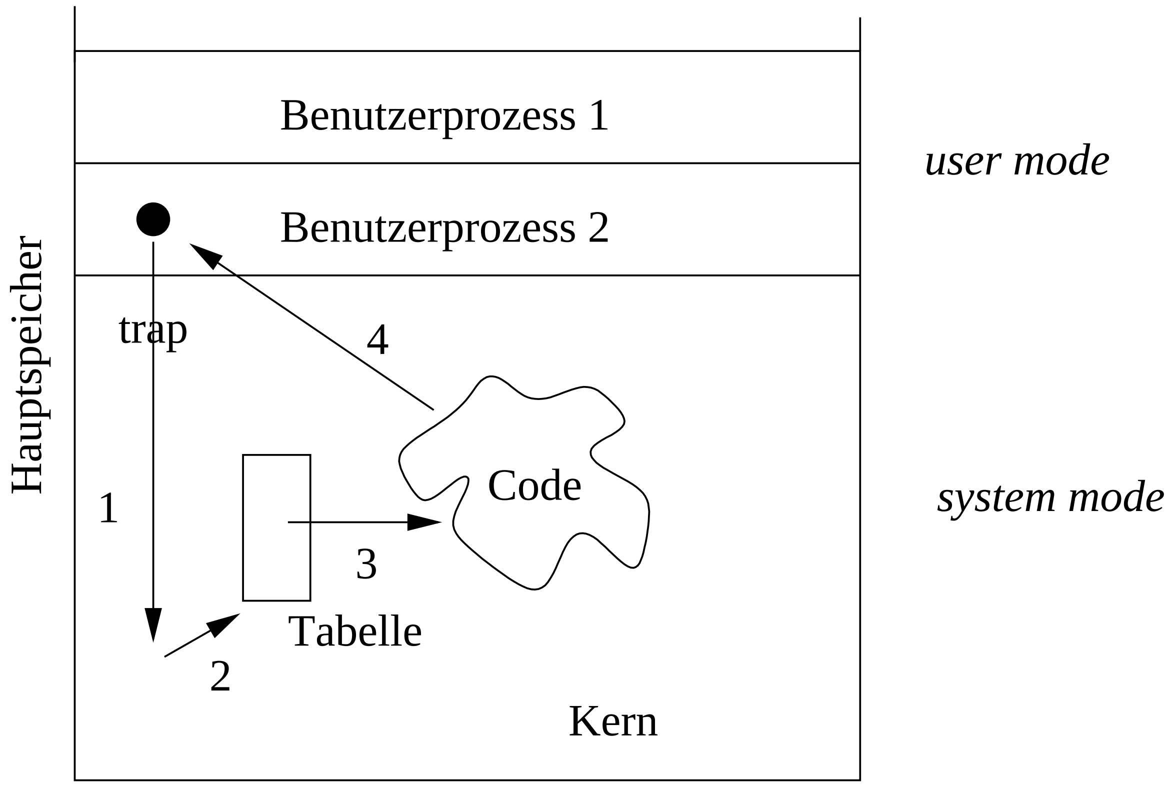
- CPU, Speicher, Peripherie (HD, FD, Drucker, Netz, etc.)
- Adreßbus, Datenbus, Steuerleitungen
- System- und Benutzermodus
- Steuerung CPU, Speicher, Controller
- Interrupts

Von *batch* zu *timesharing* (Geschichte)

- *batch*
  - ein Prozeß, Prozesse nacheinander (Stapelverarbeitung)
- *multiprogramming*
  - mehrere Prozesse "gleichzeitig", Ausnutzen der Wartezeiten bei Ein-/Ausgabe (*I/O interrupt*)
- *timesharing*
  - *multiprogramming* mit Zeitscheibe (*timer interrupt*)

Schichtbildung

- *user mode, system mode*
- Betriebssystemaufrufe (*system calls*)



Grundkonzepte

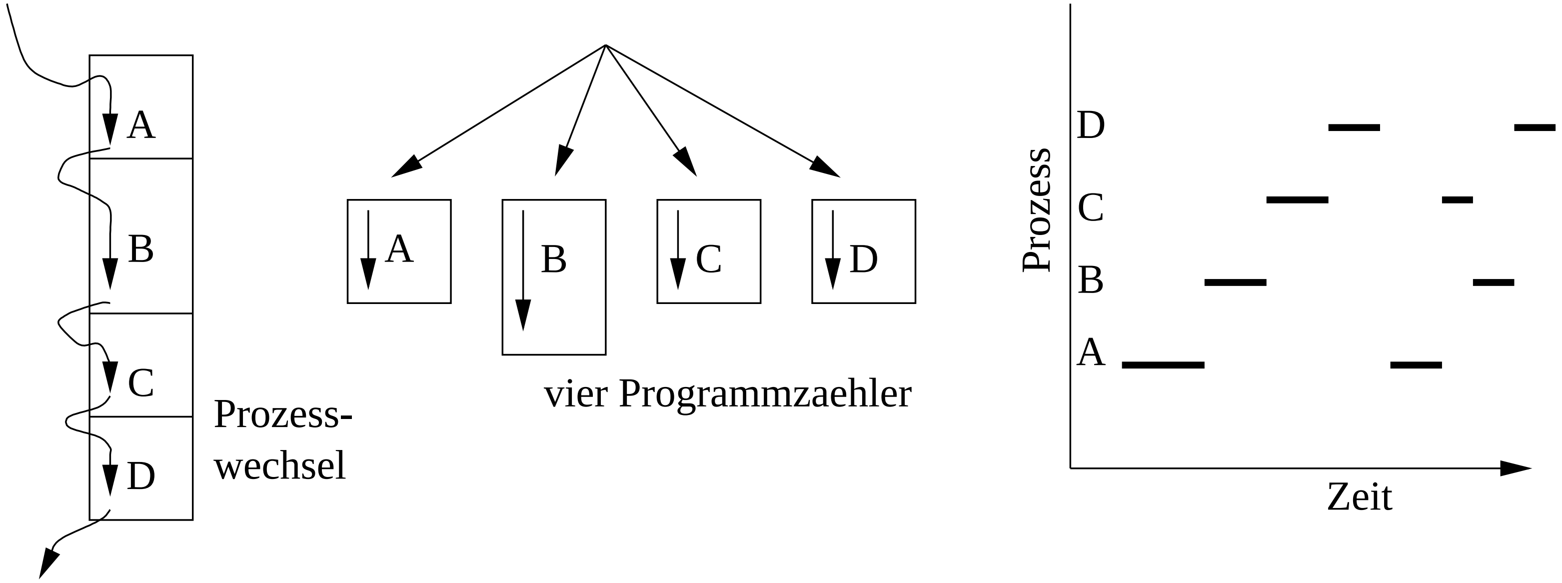
- **Prozesse**
- virtueller Speicher
- Ein-/Ausgabe

Der Prozeßbegriff

- Prozeß vs. Programm
- Prozeß = Aktivität (Programm, Zustand, E/A-Daten)
- Programm = “Algorithmus”, Code
- Kontext eines Prozesses = Programm, Daten, Stack, Register, Programmzähler
- eindeutiger Prozeßbezeichner (*process id* = PID)

Pseudoparallelität

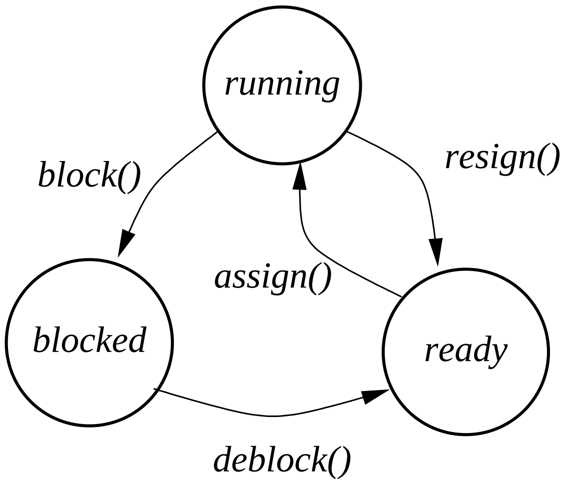
- verschiedene Sichten von *timesharing*



(a) (b) (c)

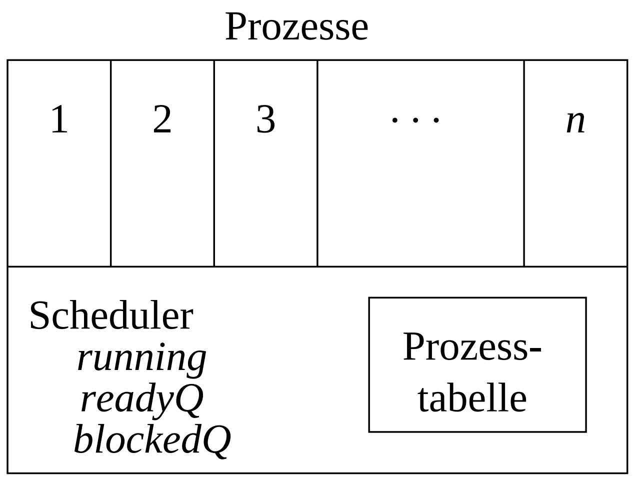
Zustände eines Prozesses

- *running* (“hat gerade den Prozessor”)
- *ready* (“will den Prozessor”)
- *blocked* (“wartet auf etwas”, z.B. I/O)



- Sinn von *blocked*?

Implementierung



- Prozesstabelle (PID, PPID, etc.), Variable *running*, Warteschlangen *blockedQ*, *readyQ*
- Implementierung der Zustandsübergänge ...
  - bei *timer interrupt*,
  - bei Ein-/Ausgabe-Anforderung und -Antwort
- Wichtig: wechselseitiger Ausschluß, Fairness

Prozeßtabelle in Linux

Ausschnitt aus `sched.h` des Linux Kernels (2.1.125, Okt. 1998):

```
struct task_struct {
  mm_segment_t addr_limit;  /* process address space:
    0-0xBFFFFFFF for user-thead, 0-0xFFFFFFFF for kernel-thread */
  /* SMP and runqueue state */
  int has_cpu;
  int processor;
  struct task_struct *next_run,  *prev_run;

  /* task state */
  struct linux_binfmt *binfmt;
  unsigned long personality;

  /* process credentials */
  uid_t uid,euid,suid,fsuid;
  gid_t gid,egid,sgid,fsgid;
  pid_t pid;
};
```

Benutzer und Prozesse

- Hochfahren: ein Logon-Prozeß pro Terminal
- Mehrbenutzersystem: Einloggen mit *id* und Passwort
- nach dem Einloggen: Kommandointerpreter/Shell
- Benutzer wird Besitzer des Prozesses (und seiner Kinder)
- Zugriffsschutz auf Ressourcen (Speicher, Dateien, etc.)
- Superuser

Prozeßkontrolle

```
ultra11 login: theedge
theedge's password:
Last login: Mon Oct 12 17:54:22 1998 on console
ultra11 $ xterm &
[1] 29866
ultra11 $ ps
    UID   PID  PPID  C    STIME TTY      TIME CMD
theedge 29839 29836  0 20:25:16 console    0:00 -bash
theedge 29867 29866  0 20:26:48 console    0:00 bash
theedge 29872 29839  0 20:27:20 console    0:00 ps
ultra11 $ top
  PID USERNAME PRI NICE  SIZE   RES STATE   TIME   WCPU    CPU COMMAND
29880 theedge   33    0 3704K 1800K cpu     0:00  0.65%  0.45% top
29836 root      33    0 2000K 1752K sleep   0:00  0.01%  0.02% sshd
29839 theedge   23    0 1944K 1680K sleep   0:00  0.02%  0.01% bash
ultra11 $ kill 29866
```

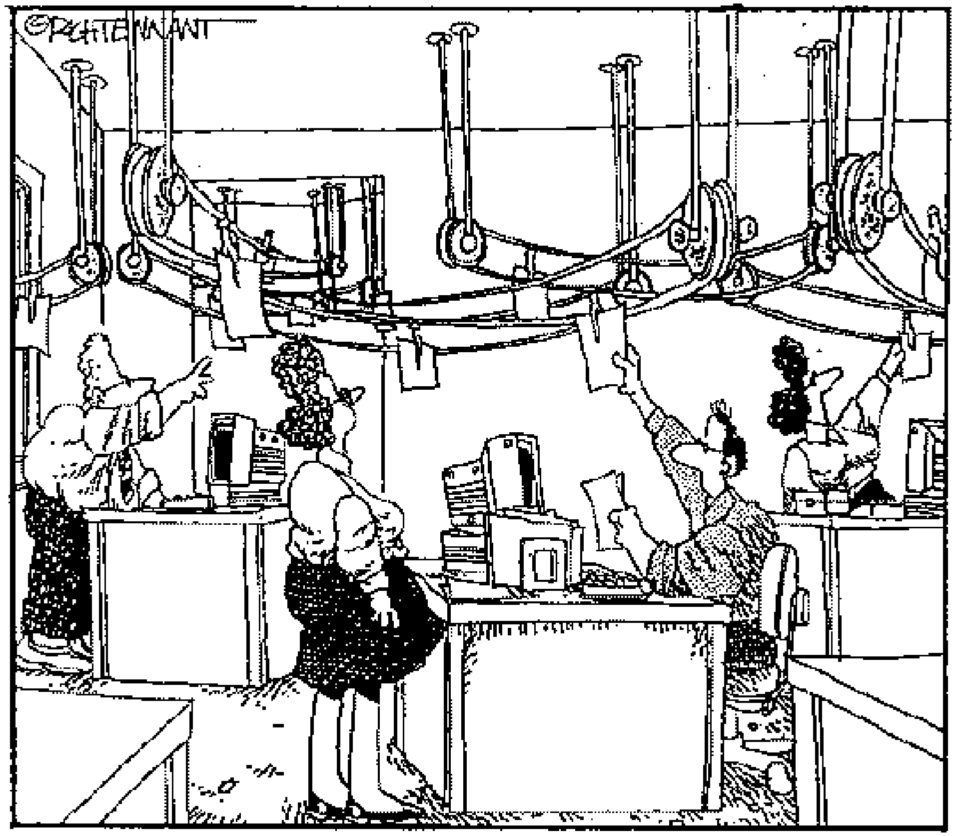
```
[1]+  Exit 15                         xterm
 ultra11 $ whoami
theedge
 ultra11 $ who
theedge     console  Oct 13 20:25
root        pts/1    Oct 13 16:15
 ultra11 $
```

- "&", ps, top, kill, who, whoami, man, rlogin

Planspiel Grundlagen

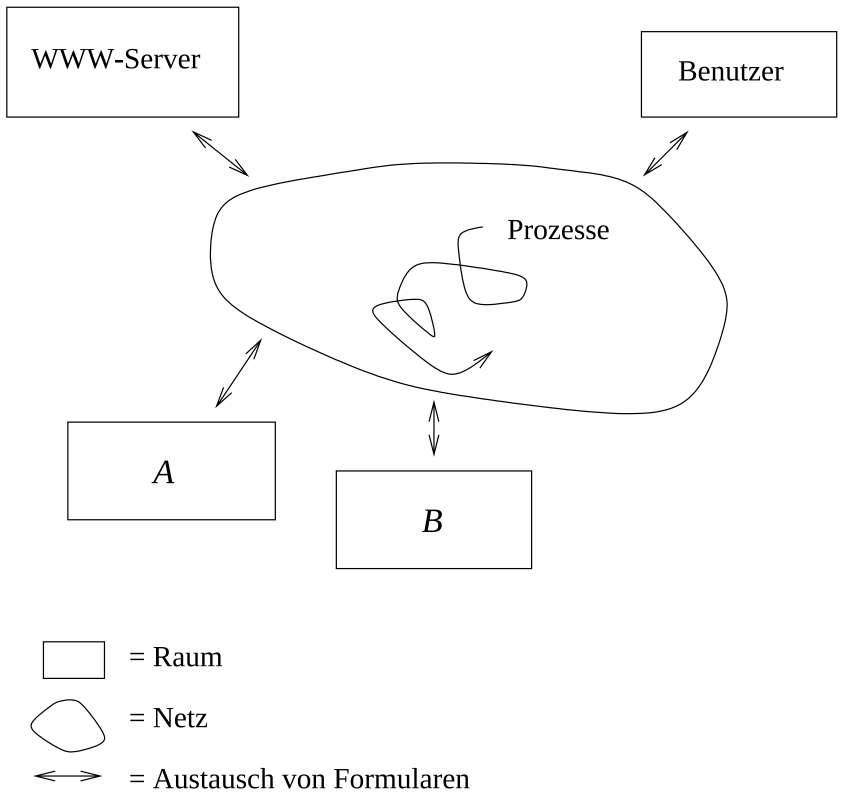
- Motivation
- Systemarchitektur
- Ressourcen
- Spielergruppen
- Spielablauf

Motivation



„DAS IST NUR VORÜBERGEHEND SO, BIS UNSER INTERNET-ANSCHLUSS WIEDER FUNKTIONIERT."

Systemarchitektur

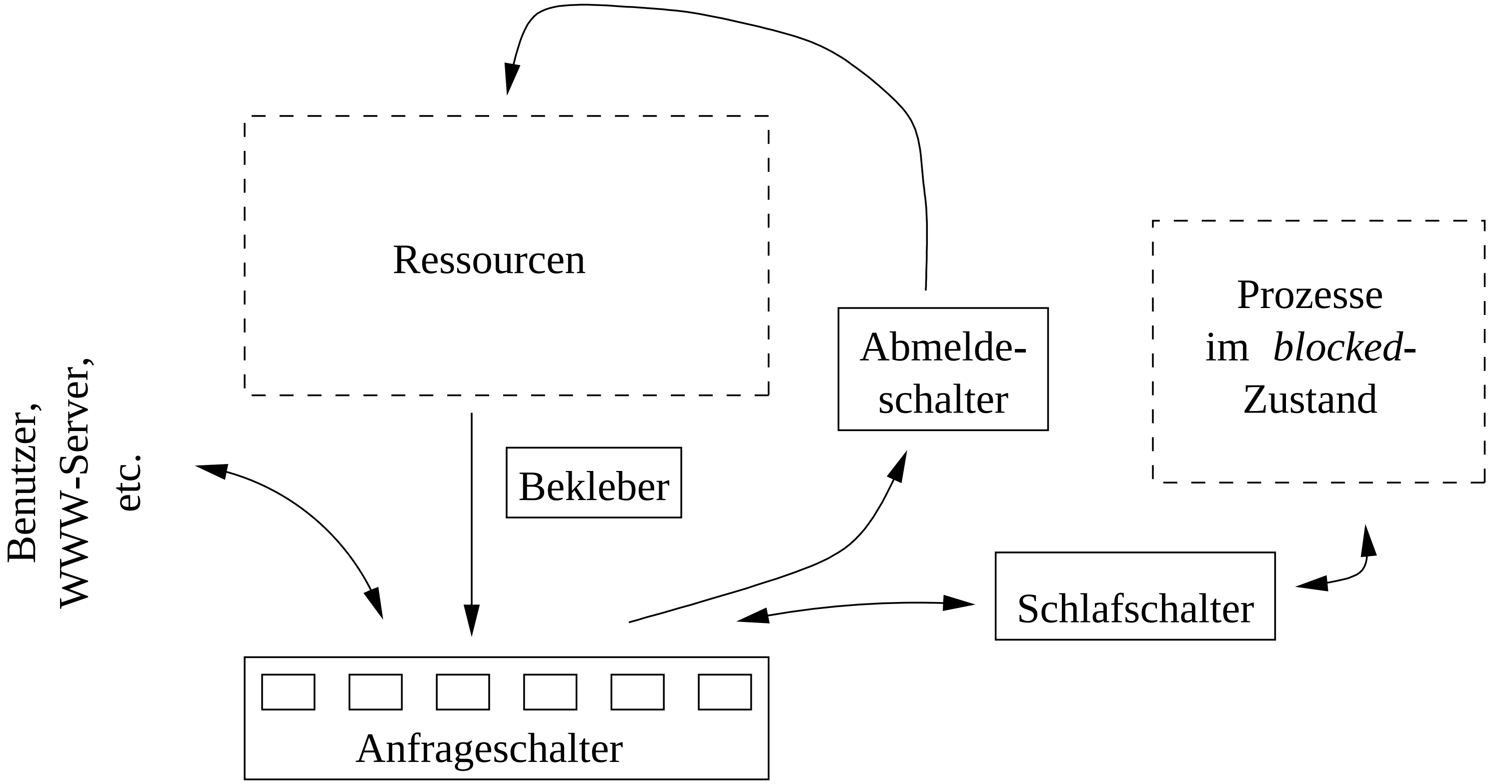


Grobüberblick I

- Benutzer:
  - loggen sich anfangs ein, “arbeiten” am Terminal
- Ressourcen:
  - “schauen überall mal rein”
- WWW-Server (eigentlich ein “*proxy*”):
  - “ich bin das WWW”’

Grobüberblick II

Die CPU (es gibt zwei):



Spielablauf

1. Einteilung in Gruppen (zu Beginn des Planspiels)
2. Einzeleinweisung der Gruppen am Ort
3. Hochfahren der Rechner
4. Einloggen der Benutzer
5. … es läuft …

Tips

- Benutzer: können Kommandos “gefahrlos” testen
- CPU: lernt am meisten
- WWW-erver: kreative Leute gefragt
- Ressource: am meisten Spaß, größte Abwechslung
- Debugging

Zusammenfassung

1. Betriebssysteme — Grundlagen
   - Was ist ein Betriebssystem?
   - Grundkonzepte
   - Prozeßverwaltung
   - Benutzer und Zugriffskontrolle
2. Planspiel — Grundlagen
   - Motivation: Wozu Planspiel?
   - Systemarchitektur
   - Spielablauf

Beginn: morgen, 13:30 Uhr, Foyer 38er