Praxis Betriebssysteme

Modul OPSY1 Betriebssysteme

Teilgebiet Praxis Betriebssysteme

Prof. Dr. I. Brunner
Staatliche Studienakademie Leipzig

Inhalt

1. Einführung
   - Geschichte
   - Betriebssystem-Aufgaben
   - Struktur von Betriebssystemen
   - Betriebssystem-Konzepte
2. Grundlegende UNIX -Befehle
3. Aufbau Partionstabellen
4. Der vi Editor
5. Die Shell
6. Shell Scripte

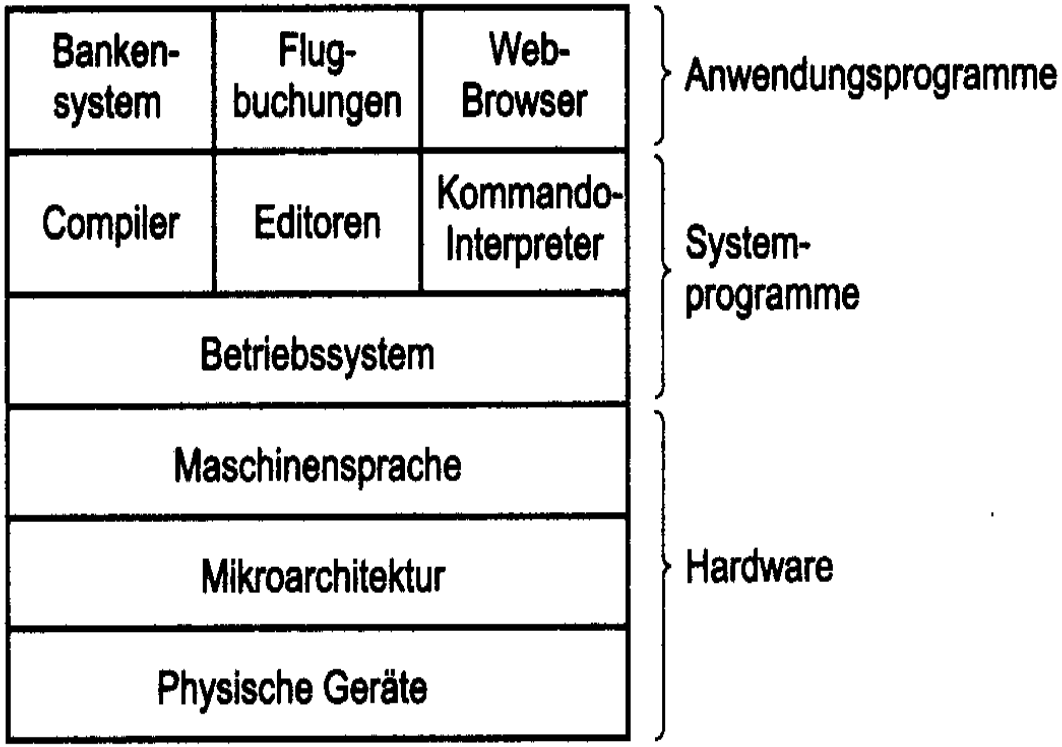
Literatur

- Adrew S. Tannenbaum: Moderne Betriebssysteme. 2. überarb. Auflage, Pearson Studium 2002, ISBN 3827370191

- Winfried Kalfa: Betriebssysteme. 1988, ISBN: 3055004779.

Rechensystem

- Hardware
- Systemprogramme
- Anwendungsprogramme



Geschichte der Betriebssysteme

- 1. Digitalrechner: „Analytische Maschine“ von Charles Babbage (1792-1871)

- Babbage erkannte die Notwendigkeit von Software
  - Für „Software“ stellte er Ada Lovelace ein, die damit faktisch die 1. Programmiererin ist
  - Die Programmiersprache Ada ist nach ihr benannt

Geschichte der Betriebssysteme

1. Generation 1945-1955 – Elektronenröhren
2. Generation 1955-1965 – Transistoren
    - Stapelverarbeitungssystem
3. Generation 1965 – 1980
    - Jobverarbeitung
    - Time Sharing
    - UNIX
4. Generation 1980 – heute - Large Scale Integration
    - CP/M
    - DOS
    - GUI
    - MS Windows

UNIX Geschichte

(Quelle: http://www.unix.org/what_is_unix/history_timeline.html)

1969	**The Beginning**	The history of UNIX starts back in 1969, when Ken Thompson, Dennis Ritchie and others started working on the "little-used PDP-7 in a corner" at Bell Labs and what was to become UNIX.
1971	**First Edition**	It had a assembler for a PDP-11/20, file system, fork(), roff and ed. It was used for text processing of patent documents.
1973	**Fourth Edition**	It was rewritten in C. This made it portable and changed the history of OS's.
1975	**Sixth Edition**	UNIX leaves home. Also widely known as Version 6, this is the first to be widely available out side of Bell Labs. The first BSD version (1.x) was derived from V6.
1979	**Seventh Edition**	It was a "improvement over all preceding and following Unices" [Bourne]. It had C, UUCP and the Bourne shell. It was ported to the VAX and the kernel was more than 40 Kilobytes (K).
1980	**Xenix**	Microsoft introduces Xenix. 32V and 4BSD introduced.
1982	**System III**	AT&T's UNIX System Group (USG) release System III, the first public release outside Bell Laboratories. SunOS 1.0 ships. HP-UX introduced. Ultrix-11 Introduced.
1983	**System V**	Computer Research Group (CRG), UNIX System Group (USG) and a third group merge to become UNIX System Development Lab. AT&T announces UNIX System V, the first supported release. Installed base 45,000.
1984	**4.2BSD**	University of California at Berkeley releases 4.2BSD, includes TCP/IP, new signals and much more. X/Open formed.
1984	**SVR2**	System V Release 2 introduced. At this time there are 100,000 UNIX installations around the world.
1986	**4.3BSD**	4.3BSD released, including internet name server. SVID introduced. NFS shipped. AIX announced. Installed base 250,000.
1987	**SVR3**	System V Release 3 including STREAMS, TLI, RFS. At this time there are 750,000 UNIX installations around the world. IRIX introduced.
1988		POSIX.1 published. Open Software Foundation (OSF) and UNIX International (UI) formed. Ultrix 4.2 ships.
1989		AT&T UNIX Software Operation formed in preparation for spinoff of USL. Motif 1.0 ships.
1989	**SVR4**	UNIX System V Release 4 ships, unifying System V, BSD and Xenix. Installed base 1.2 million.

1989	**SVR4**	UNIX System V Release 4 ships, unifying System V, BSD and Xenix. Installed base 1.2 million.
1990	**XPG3**	X/Open launches XPG3 Brand. OSF/1 debuts. Plan 9 from Bell Labs ships.
1991		UNIX System Laboratories (USL) becomes a company - majority-owned by AT&T. Linus Torvalds commences Linux development. Solaris 1.0 debuts.
1992	**SVR4.2**	USL releases UNIX System V Release 4.2 (Destiny). October - XPG4 Brand launched by X/Open. December 22nd Novell announces intent to acquire USL. Solaris 2.0 ships.
1993	**4.4BSD**	4.4BSD the final release from Berkeley. June 16 Novell acquires USL
Late 1993	**SVR4.2MP**	Novell transfers rights to the "UNIX" trademark and the Single UNIX Specification to X/Open. COSE initiative delivers "Spec 1170" to X/Open for fasttrack. In December Novell ships SVR4.2MP , the final USL OEM release of System V
1994	**Single UNIX Specification**	BSD 4.4-Lite eliminated all code claimed to infringe on USL/Novell. As the new owner of the UNIX trademark, X/Open introduces the Single UNIX Specification (formerly Spec 1170), separating the UNIX trademark from any actual code stream.
1995	**UNIX 95**	X/Open introduces the UNIX 95 branding programme for implementations of the Single UNIX Specification. Novell sells UnixWare business line to SCO. Digital UNIX introduced. UnixWare 2.0 ships. OpenServer 5.0 debuts.
1996		The Open Group forms as a merger of OSF and X/Open.
1997	**Single UNIX Specification, Version 2**	The Open Group introduces Version 2 of the Single UNIX Specification, including support for realtime, threads and 64-bit and larger processors. The specification is made freely available on the web. IRIX 6.4, AIX 4.3 and HP-UX 11 ship.
1998	**UNIX 98**	The Open Group introduces the UNIX 98 family of brands, including Base, Workstation and Server. First UNIX 98 registered products shipped by Sun, IBM and NCR. The Open Source movement starts to take off with announcements from Netscape and IBM. UnixWare 7 and IRIX 6.5 ship.
1999	**UNIX at 30**	The UNIX system reaches its 30th anniversary. Linux 2.2 kernel released. The Open Group and the IEEE commence joint development of a revision to POSIX and the Single UNIX Specification. First LinuxWorld conferences. Dot com fever on the stock markets. Tru64 UNIX ships.
2001	**Single UNIX Specification, Version 3**	Version 3 of the Single UNIX Specification unites IEEE POSIX, The Open Group and the industry efforts. Linux 2.4 kernel released. IT stocks face a hard time at the markets. The value of procurements for the UNIX brand exceeds $25 billion. AIX 5L ships.

2003	**ISO/IEC 9945:2003**	The core volumes of Version 3 of the Single UNIX Specification are approved as an international standard. The "Westwood" test suite ship for the UNIX 03 brand. Solaris 9.0 E ships. Linux 2.6 kernel released.
2007		Apple Mac OS X certified to UNIX 03.
2008	**ISO/IEC 9945:2008**	Latest revision of the UNIX API set formally standardized at ISO/IEC, IEEE and The Open Group. Adds further APIs
2009	**UNIX at 40**	IDC on UNIX market -- says UNIX $69 billion in 2008, predicts UNIX $74 billion in 2013
2010	**UNIX on the Desktop**	Apple reports 50 million desktops and growing -- these are Certified UNIX systems.

Arten von Betriebssystemen

- Mainframe OS
  - Batchverarbeitung, Transaktionsverarbeitung, Zeitaufteilungsverfahren, z.B. OS/390
- Server-OS
- Multiprozessor-OS
- OS für PC
  - „Manche Leute wissen teilweise überhaupt nicht, dass es noch andere Systeme gibt“ *(Andrew S. Tanenbaum)*
- Echtzeit-OS
- OS für eingebettete Systeme
- OS für Chipkarten

BIOS

Basic Input Output System

- POST (Power On Self Test)
- Konfiguration von Rechnerkomponenten (Plug & Play)
- Laden des OS
- Stellt OS Konfigurationsdaten zur Verfügung
- Grundlegende Programmschnittstellen zur Hardware

Aufgaben des Betriebssystems

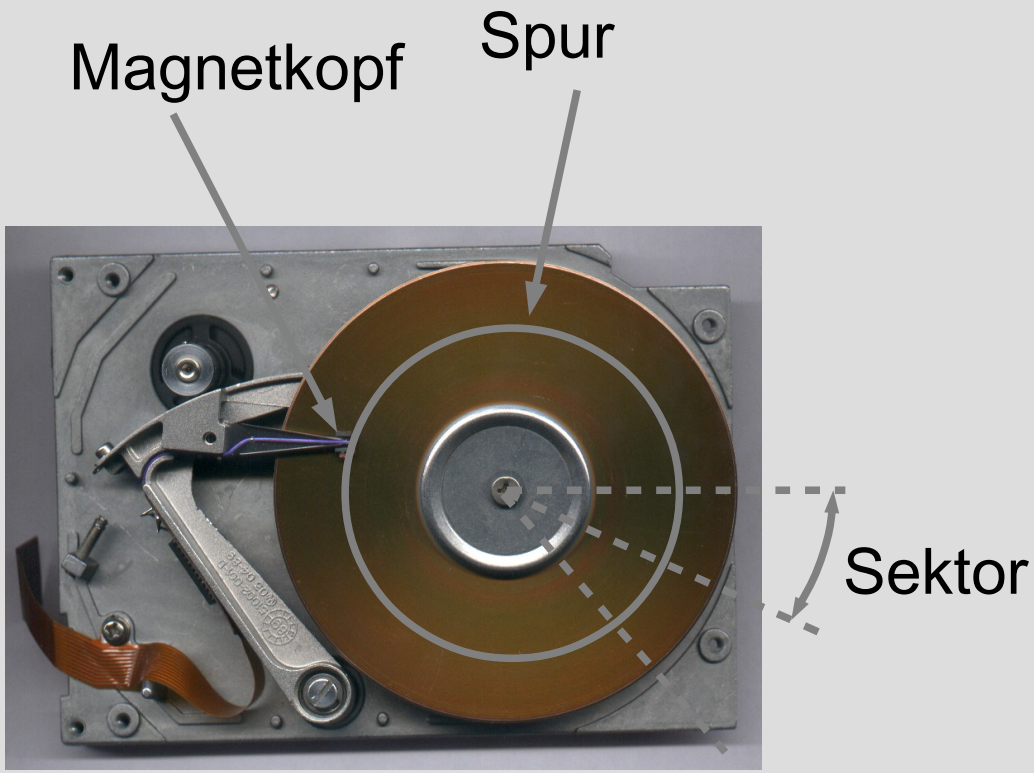
- Booten des Rechners
- Prüfung der Zugangsberechtigung (login) und Rechteverwaltung
- Verbindung zwischen Rechnersystem und Benutzer (Benutzer: Mensch, Programm)
    - Kommandointerpretation
    - Shell
    - Benutzeroberfläche
- Verwaltung von Daten in Form von Dateien (Files)
    - Zugriff auf Dateien auf Massenspeicher (Schreiben, Lesen, Kopieren, Löschen, Benennen, Ordnen)
    - File enthält Programm: OS startet Ausführung d. Programms

Aufgaben des Betriebssystems (2)

- Ressourcenverwaltung
  - Hardware: CPU, Speicherplatz, FD, HD, Netzwerk, Drucker
  - Software: Programme, Prozesse, Tabellen
  - Beispiel Mainframe: Jedem Benutzer wird die CPU regelmäßig für eine begrenzte Zeit zugeteilt, scheinbar „gleichzeitige“ Bearbeitung (Multitasking, Time Sharing)
- Erhebung von Abrechnungsdaten (vor allem für Mehrbenutzersysteme)
- Komplexität der Hardware wird vor dem Benutzer verborgen (Abstraktion, Virtualisierung)

Aufgaben des Betriebssystems (3)

- Gerätezugriff wird vereinfacht (vgl. Abstraktion)
  - Beispiel Festplatte (Floppy Disk ähnlich)



IBM PC Floppy Disk

Anzahl der Zylinder	40
Spuren pro Zylinder	2
Sektoren pro Spur	9
Bytes pro Sektor	512
Sektoren insgesamt	720
Bytes insgesamt	368640

Einordnung OS

- Top-down-Sicht:
  - OS präsentiert den Benutzern eine erweiterte bzw. virtuelle Maschine, die leichter als die darunterliegende Hardware zu programmieren ist

- Bottom-up-Sicht
  - OS übernimmt die Verwaltung aller Bestandteile eines komplexen Systems (Verwaltung CPU-Verwendung durch Programme, Drucker-Pipeline)

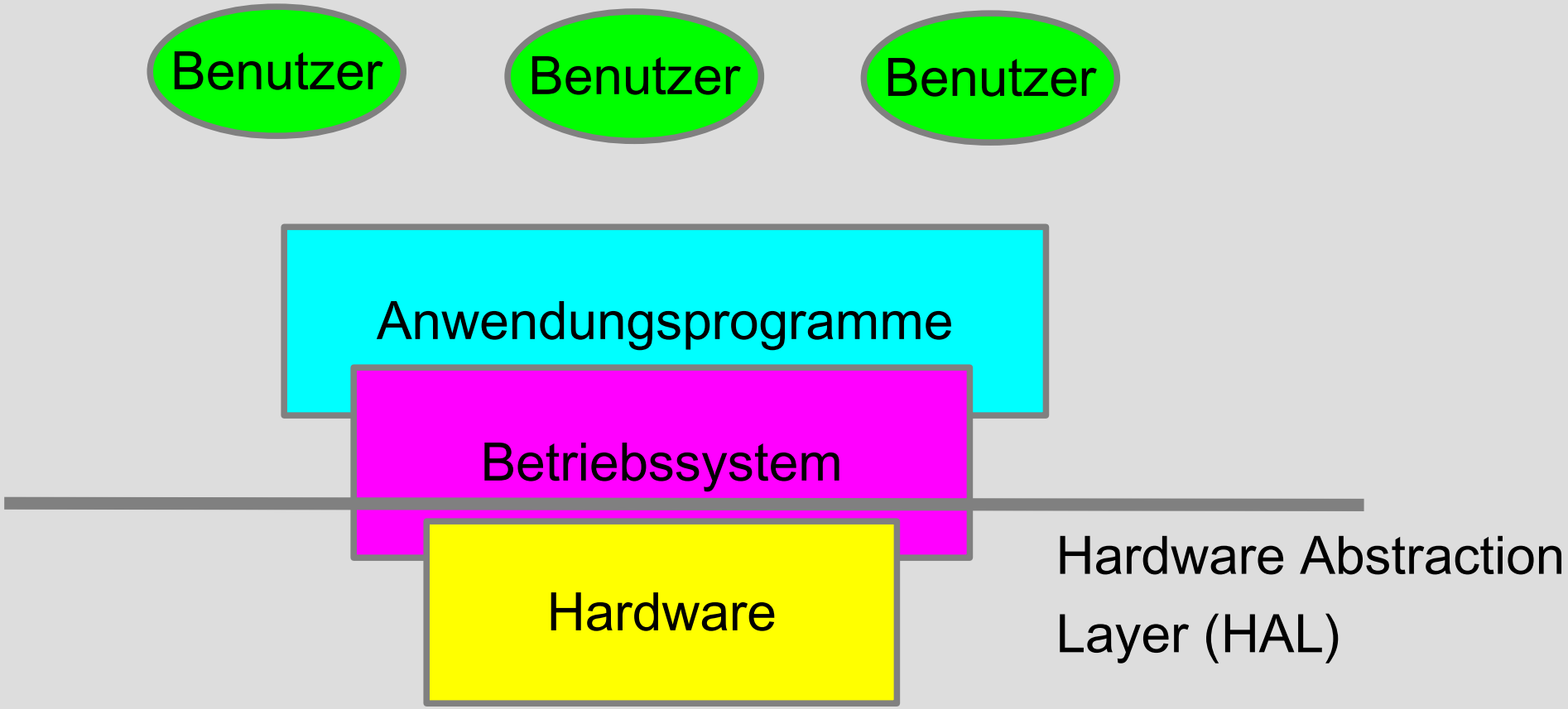
Mehrere Programme im Speicher

- Probleme:
  1. Schutz der Programme voreinander und Schutz des Kernels
  2. Umgang mit dem Laden der Programme an unterschiedlichen Adressen im Speicher

- ➔ Alle Lösungen erfordern spezielle Hardware der CPU
  - ➔ Einfachste Lösung: Basis-Register + Limit-Register (MMU Memory Management Unit)

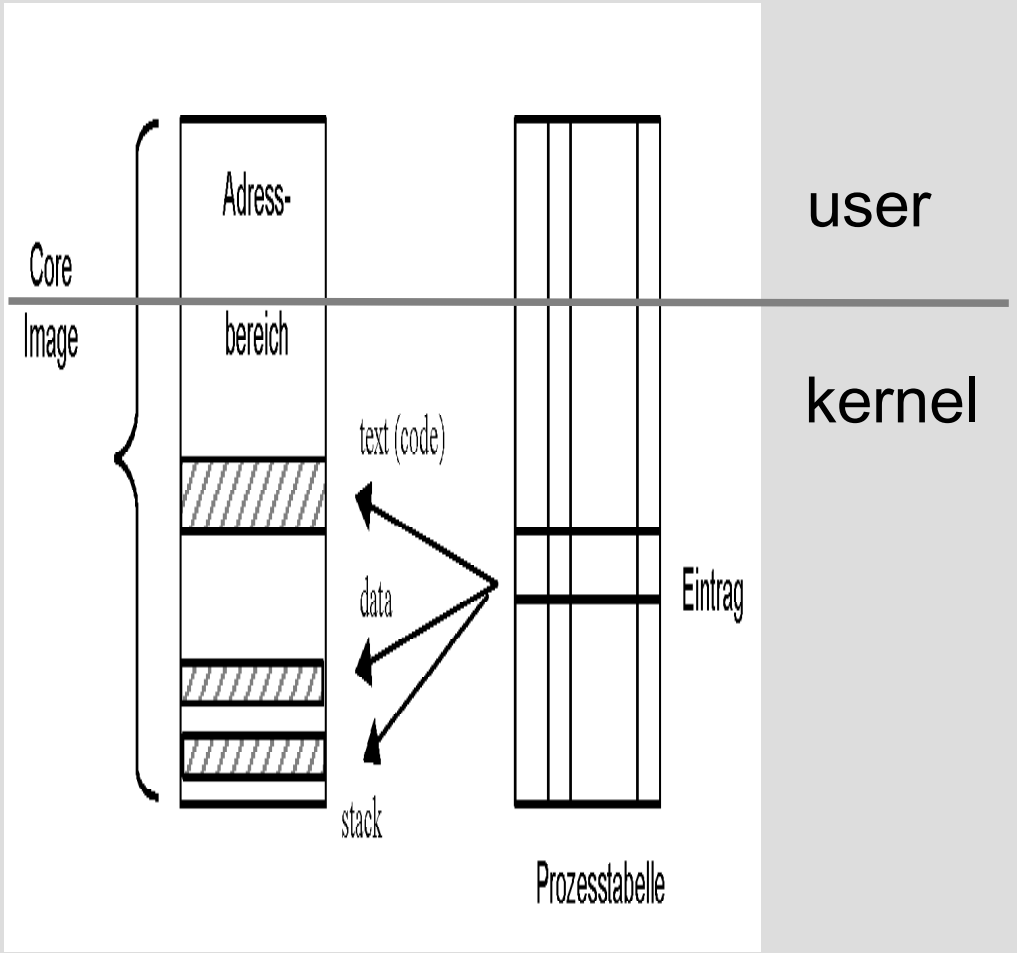
- Resultat: Prozesswechsel kostet relativ viel Zeit

Einordnung eines Betriebssystems in die Systemarchitektur



Monolithische Betriebssystemstruktur

- Aufteilung in Benutzer-ebene (user) und Systemebene (kernel)
- Ablauf:
  1. Anwender-Programm benötigt OS-Service: System Call
  2. Parameter in Übergabebereich plaziert
  3. Steuerung an Systemkern übergeben: Kernel Call
  4. Kernel identifiziert Service-Routine und ruft sie auf
  5. Service-Routine läuft ab ung gibt Ergebnis an das Anwender-Programm zurück
- Teilung in user – kernel ergibt ungenügende Strukturierung

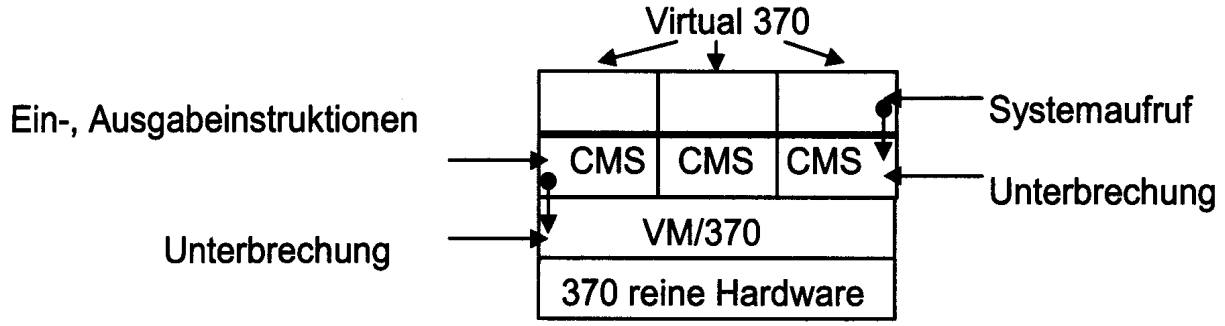


Geschichtete Betriebssystemstruktur

- 3 Ebenen:
  - User
  - Service Routinen
  - Basisdienste
- Trennung wird oft nicht strikt eingehalten

Virtuelle Maschinen

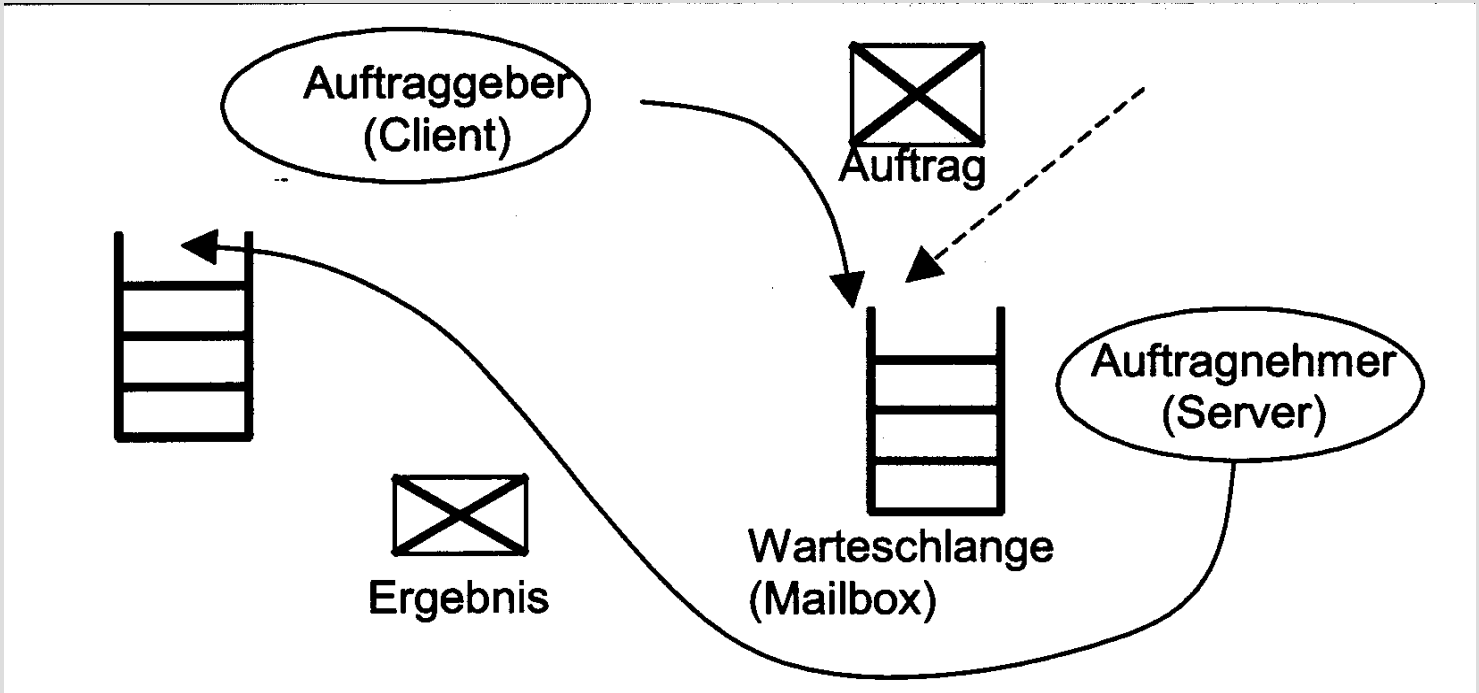
- Zweiteilung der OS-Aufgaben mit der Einführung von Time-Sharing:
  1. Mehrfachnutzung der Hardware
  2. Anhebung der Hardware-Schnittstelle: extendet machine
- Beispiel IBM/370:
  - Virtual Machine Monitor (VM/370) vervielfacht Hardware durch exakte Replikation
  - Auf der vereinfachten Hardware-Schnittstelle setzt ein (oder mehrere) Single User OS auf



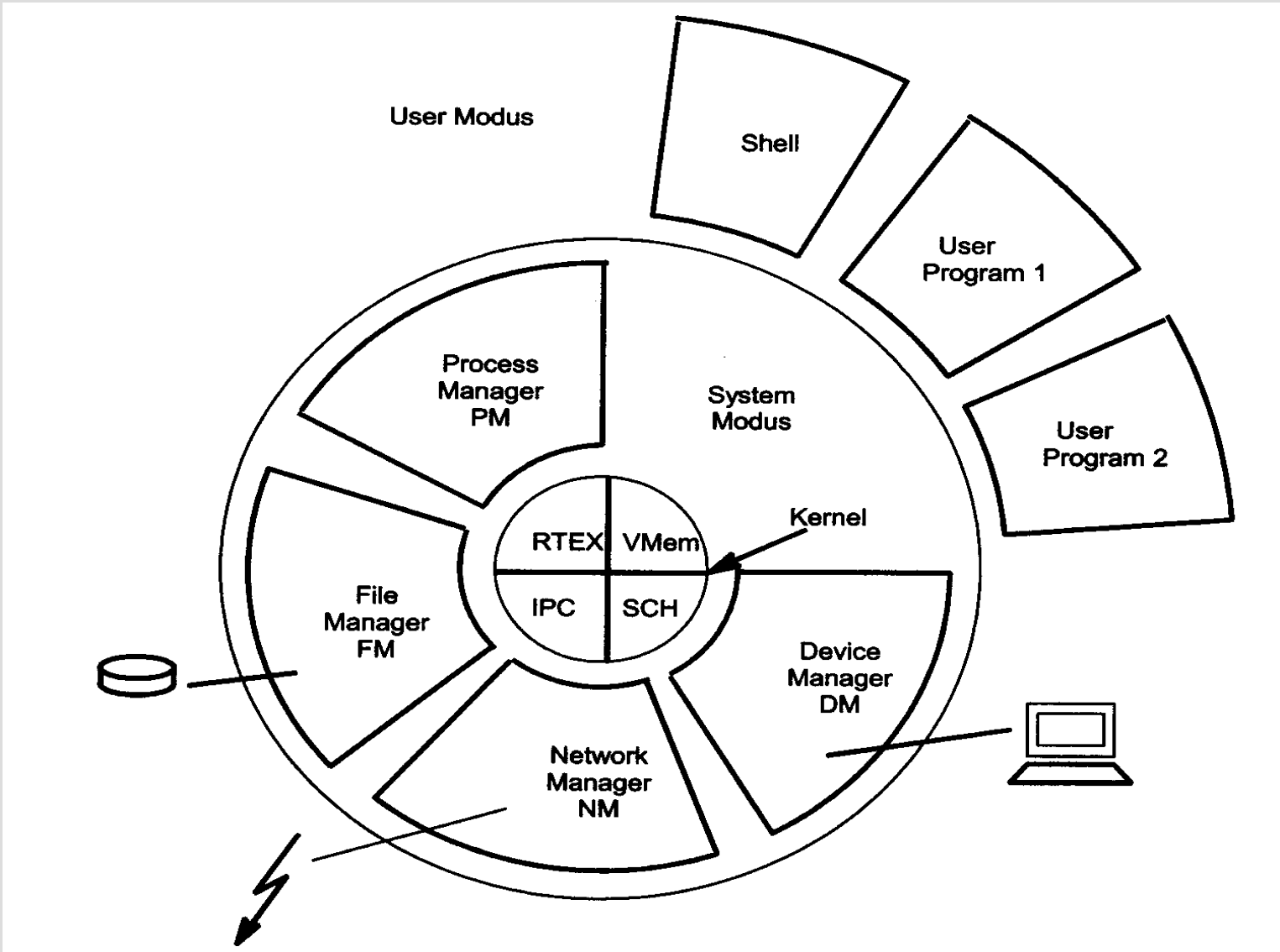
Struktur von VM/370 mit CMS

Client-Server-Modell eines OS

- Prinzipieller Mechanismus: Nachrichtenaustausch (message passing), Mailboxes, Warteschlangen
- Trend zu Microkernel, nur Basisdienste, Services in eine höhere (benutzernähere) Ebene verlagert



UNIX - Client-Server-Struktur



Gerätetreiber

- Controller sind verschieden, das erfordert eine spezielle Software die Kommandos an Controller sendet und die Antworten empfängt – den **Gerätetreiber**
- Gerätetreiber werden meist im Kernelmodus ausgeführt (selten im Benutzermodus)
- Integration in Kernel:
  1. Treiber in Kern einbinden und Kern durch Neustart laden (z.B. UNIX)
  2. OS lädt Treiber beim Hochfahren aus einer Datei (MS Windows)
  3. Nachladen zur Laufzeit (z.B. für Hot-Plug-Geräte)

Aufgaben des UNIX Kernels

- CPU-Verwaltung (Echtzeitaufgaben, Real Time Executive) und Scheduling
- Virtuelle Speicherverwaltung
- Inter-Prozess-Kommunikation
- Dienste des OS werden durch sogenannte Manager wahrgenommen:
    - File Manager
    - Network Manager
    - Process Manager
    - Terminal Manager
- Systemprogramme (Utilities) gehören zur Anwenderebene (User): Shell, Editor, GUI usw.

Eigenschaften Client-Server OS

- Kernel stellt Mechanismen zur Verfügung, ohne über deren Nutzung informiert zu sein. Diese Mechanismen werden von den Managern zur Durchführung verschiedener Aufgaben genutzt
- Vorteile:
  - Modularer Aufbau
  - Relativ einfache Kernelportierung
  - Austausch / Weglassen von Modulen möglich
  - Verteilbar auf mehrere CPUs
  - Trennung in Funktion (Schnittstelle) und Durchführung(Implementierung)
- Nachteil:
  - Zeitaufwand für IPC

OS-Konzepte

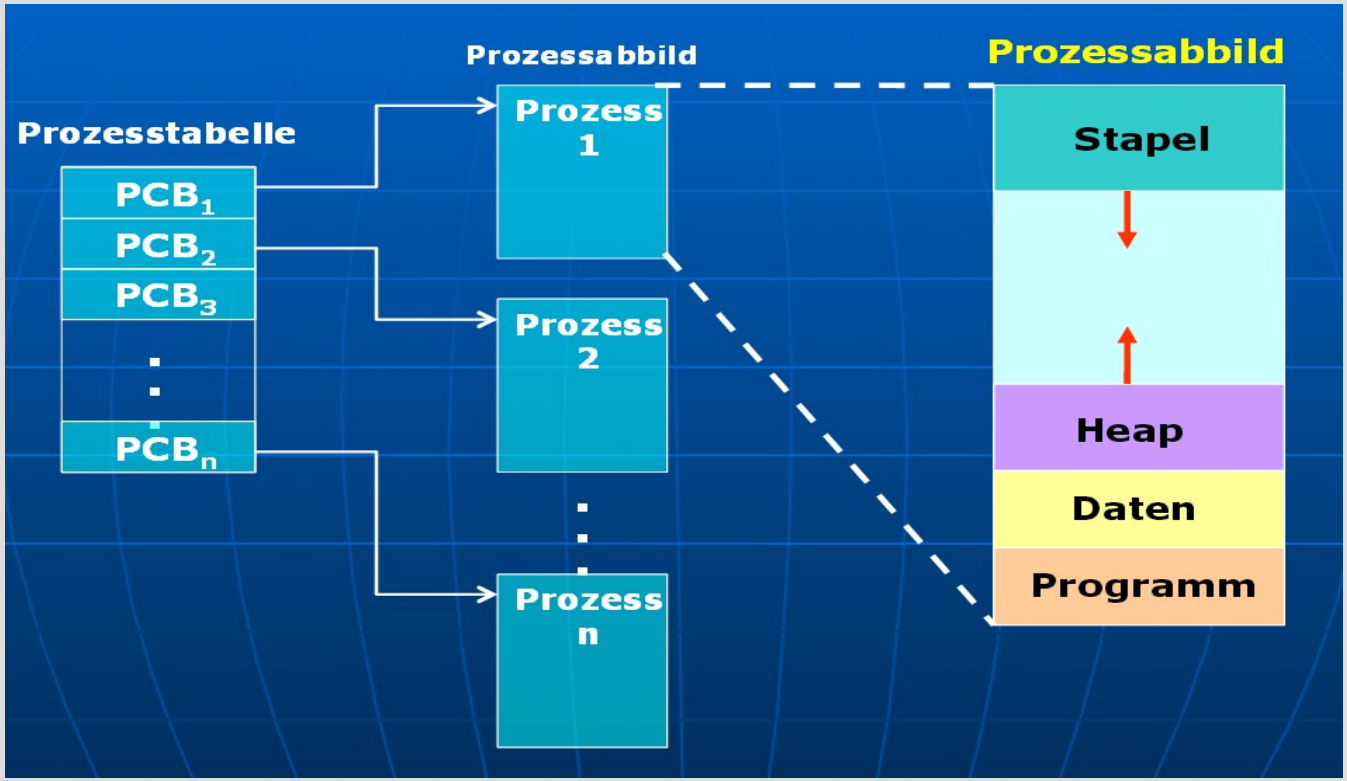
- OS bietet dem Anwender (Programm) eine erweiterte Maschinenschnittstelle (virtuelle Maschine)
    - Erweiterung des Befehlssatzes der Hardware um System-Aufrufe (system calls)
- Wichtige Abstraktionen eines OS:
    - Prozesse
    - Dateien (Files)

Prozesse

- Prozess: ausgeführtes Programm mit seiner Umgebung (Prozessor-Zustand, processor state)
- Umgebung:
    - Program Counter
    - Program Stack, Stack Pointer
    - Data Stack, Stack Pointer
    - Registersatz, evtl. Schattenregister, Flags
    - Filepointer, PID, Priorität etc.
- Bei Programm-Unterbrechung ist „Rettung“ der Umgebung notwendig (für spätere Fortsetzung)
- Identifikation:
    - Process ID (pid)
    - User ID (uid)

Prozesse

- Prozess-Tabelle: Speicherstruktur zur Aufnahme der Umgebung aller Prozesse sowie anderer Informationen

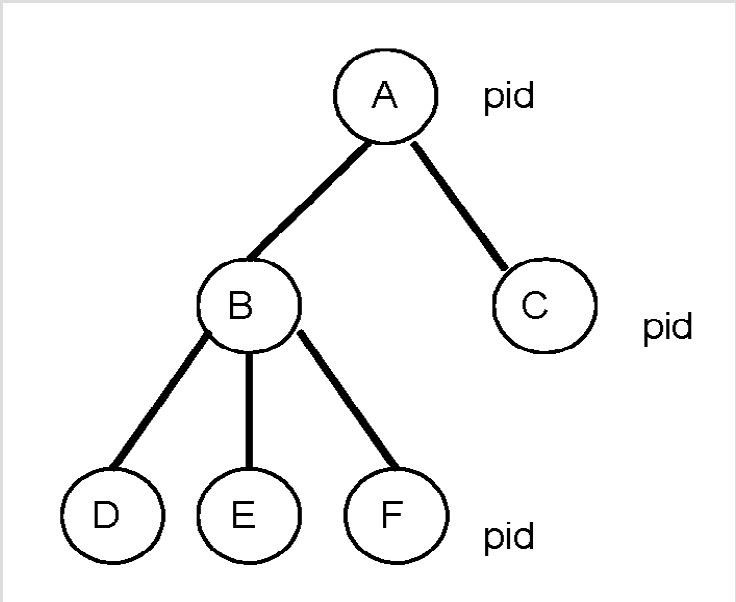


Prozess = Core Image + Eintrag in Prozesstabelle

Quelle: http://www.inf.fu-berlin.de/lehre/WS11/OS/slides/OS_V5_Prozessverwaltung_.pdf

Erzeugen von Prozessen

- Prozesserzeugung erfolgt durch *pid = fork()*
- *fork* erzeugt neuen Prozess (Kind), welcher identisch mit dem aufrufenden Prozess (Vater) ist
- Abarbeitung wird mit dem auf *fork* folgenden Prozess fortgesetzt
- Mittels *exec* kann sich ein Befehl in einen anderen transformieren



Beenden von Prozessen

- Selbst: `exit(status)`
  - `Status` meldet dem Vater-Prozess, ob der Kind-Prozess ordnungsgemäß beendet wurde
- Fremd: über Signale, z. B. Via Shell-Kommando

```
$ kill -9 pid
```