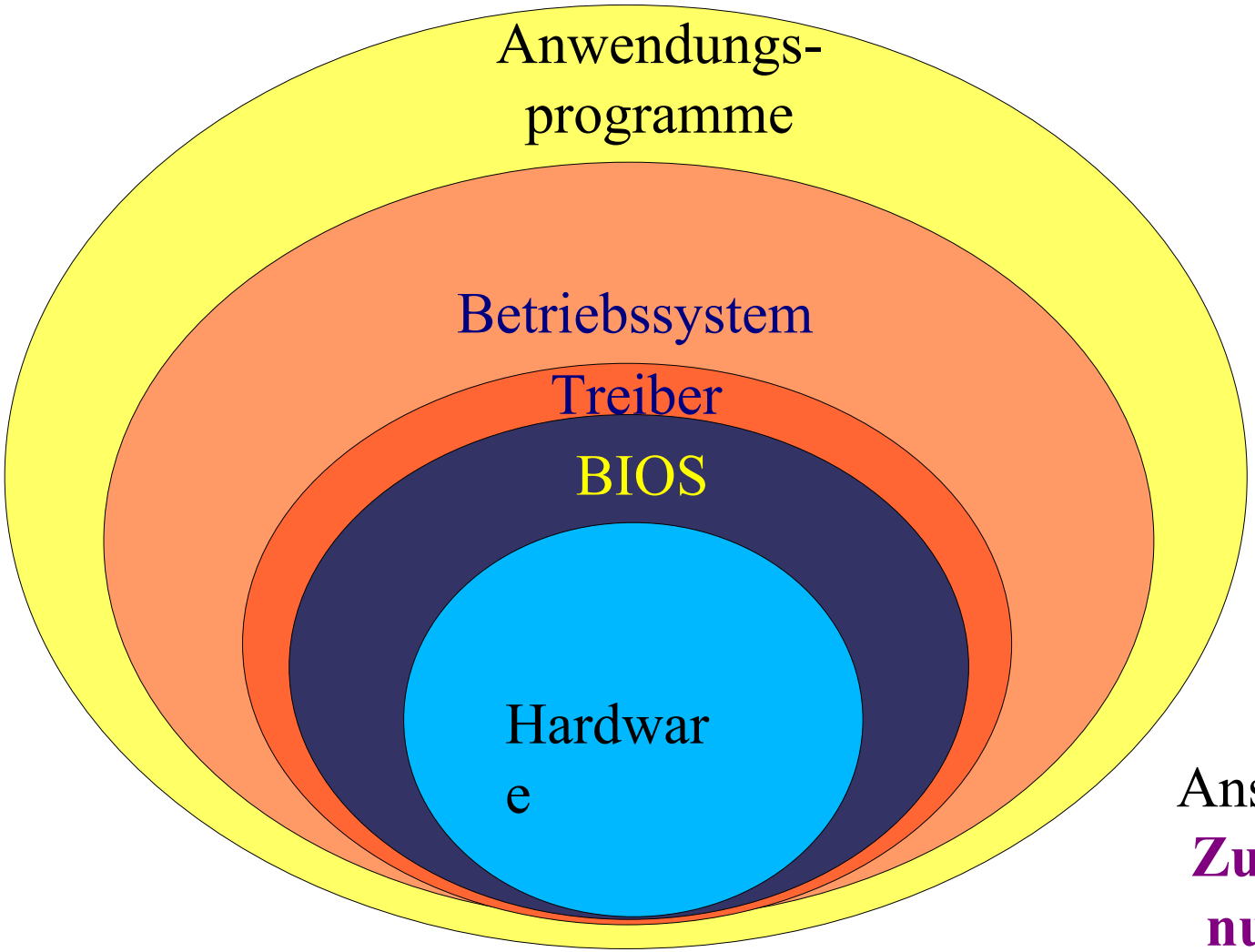
Betriebssysteme

Betriebssysteme



Direkter Zugriff von Anwenderprogrammen **auf Hardware**
nur in Ausnahmefällen sinnvoll / möglich:
- ein-Prozess-Betrieb
  (z.B. Spiele!)
- Echtzeit-Anwendungen

Ansonsten:
Zugriff auf Hardware nur über Betriebssystem

Aufgaben des Betriebssystems:

- Verwalten und Schützen der Ressourcen der Maschine (CPU, Speicher, I/O)
- Verbergen der Komplexität der Maschine vor dem Anwender („Abstraktion“)
- Bereitstellung einer normierten (Software-)Schnittstelle für Programme
- ggf. Compiler, Linker, Editor, Entwicklungsumgebung, ...
- Bereitstellung einer Nutzerschnittstelle (Shell, Command-Interpreter, graphische Oberfläche, ...)

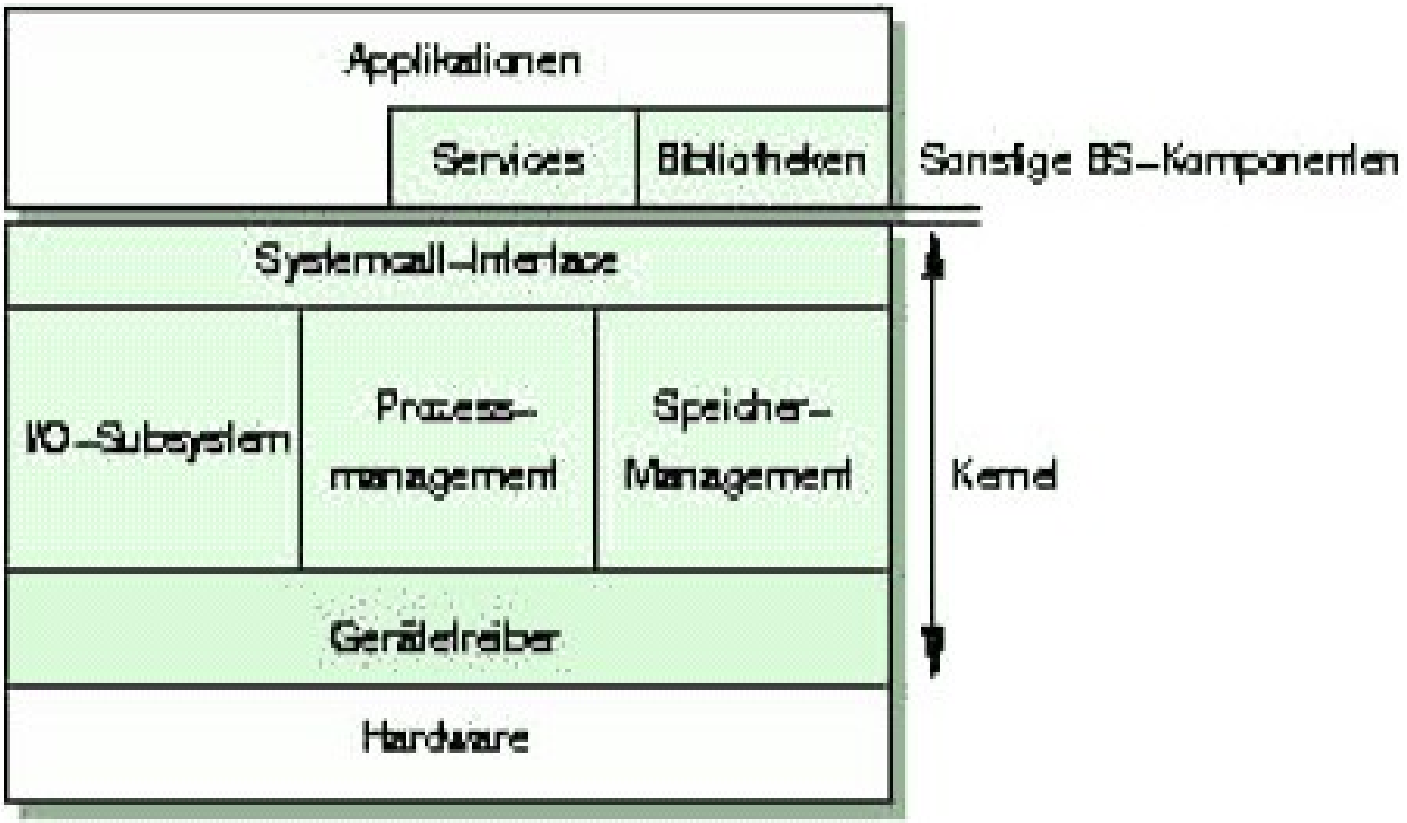
Steuerung und Überwachung von Nutzerprogrammen, evtl. Multi-Prozess und Multi-User-Betrieb, Ressourcen-Zuteilung

Unterscheidung zwischen Betriebssystem-Komponenten und Hilfsprogrammen nicht immer eindeutig

Betriebssysteme – historische Entwicklung

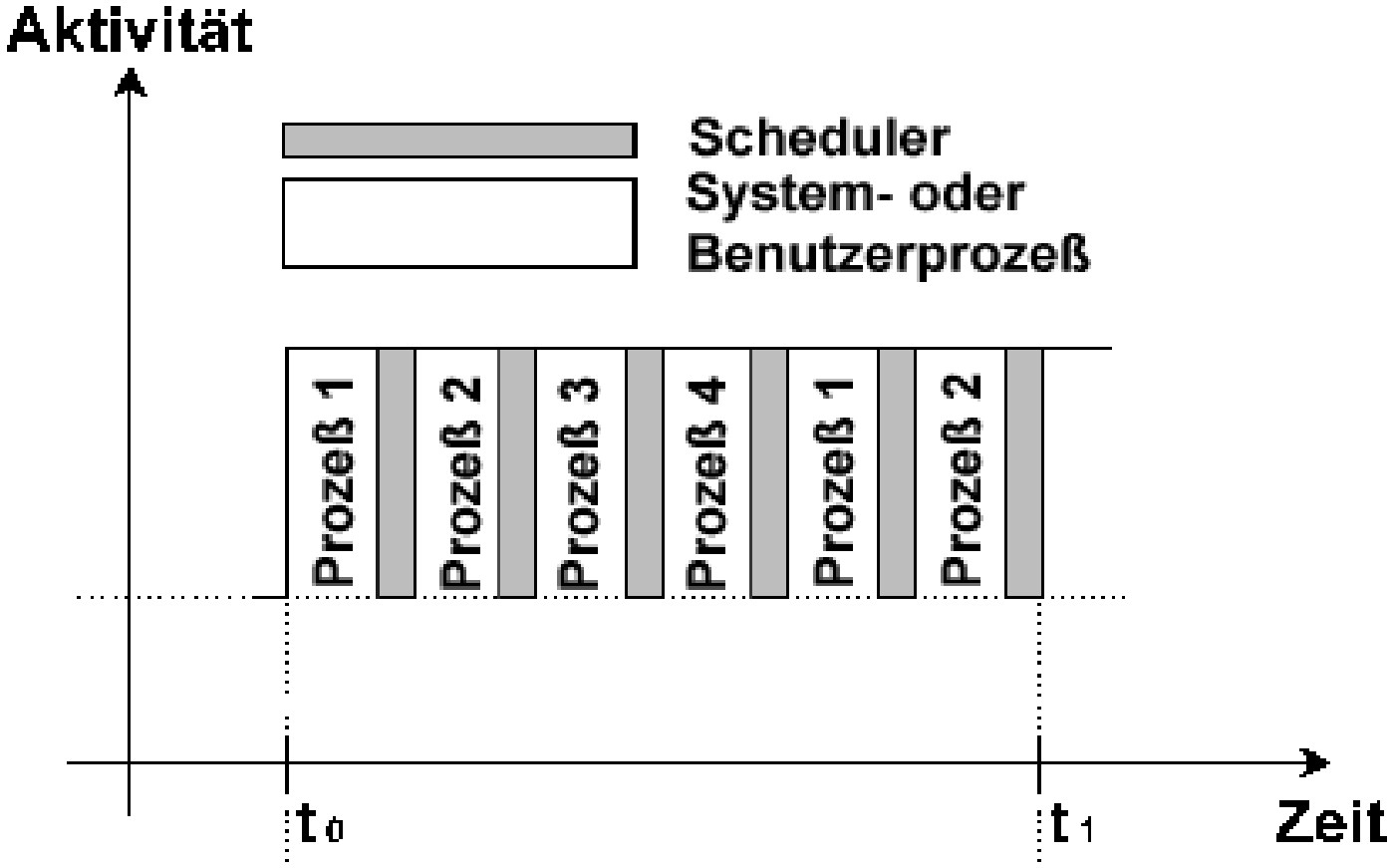
- Erste Rechner hatten gar kein“ Betriebssystem“
- Bis ca. 1965 ausschließlich „Stapelverarbeitung“ (urspr. Lochkarten-Stapel)
- Seitdem „Dialog-Betrieb“ mit Nutzer zusätzlich möglich
  quasi-simultane, zeitlich verschachtelte Ausführung von Aufgaben
  (Time-Sharing)
- 1969: das erste UNIX
- Seit etwa 1975 Dateisysteme nach heutiger Art
- Kommunikation von Computern im Netzwerk
- Verteilte Betriebssysteme:
  mehrere Programme werden von verschiedenen Prozessoren
  oder ein Programm von verschiedenen Prozessoren bearbeitet.
- 1984: der IBM-PC mit dem Betriebssystem DOS (=Disk Operation System)
- Seit Ende der 80ger: graphische Oberflächen
  (MAC-OS, Windows, X-Windows)
- Seit ca. 2000: Grid – viele Prozesse auf weltweit vernetzten Rechenzentren

Betriebssysteme – Architektur



Betriebssysteme – Multi-Tasking

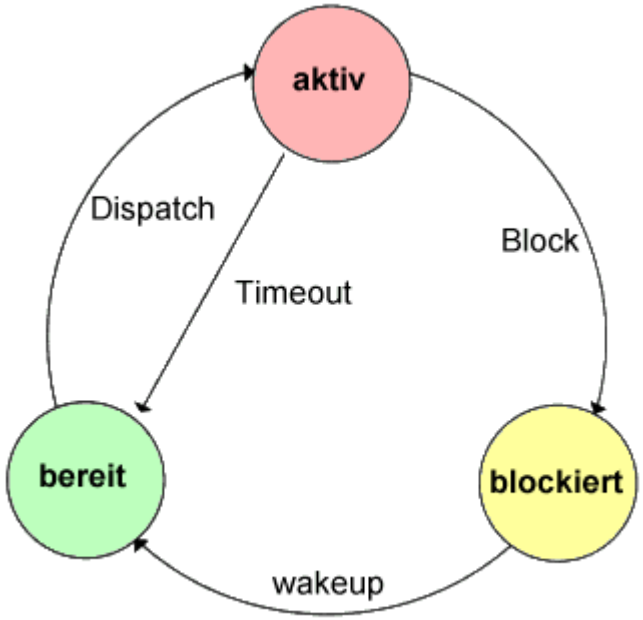
Alle modernen Betriebssysteme führen mehrere Prozesse („tasks“) quasi-parallel aus, indem abwechselnd jedem Prozess CPU-Zeit zugeteilt wird



Vorteil: Wartezeiten eines Prozesses auf I/O können von anderen Prozessen sinnvoll genutzt werden

Betriebssysteme – Prozessverwaltung

Scheduler sorgt für gerechte und
effiziente Zuteilung von CPU-Zeit
- Kooperatives Multitasking:
  Prozess gibt CPU frei
- Verdrängendes Multitasking:
  Prozess wird durch Timer-Interrupt
  unterbrochen

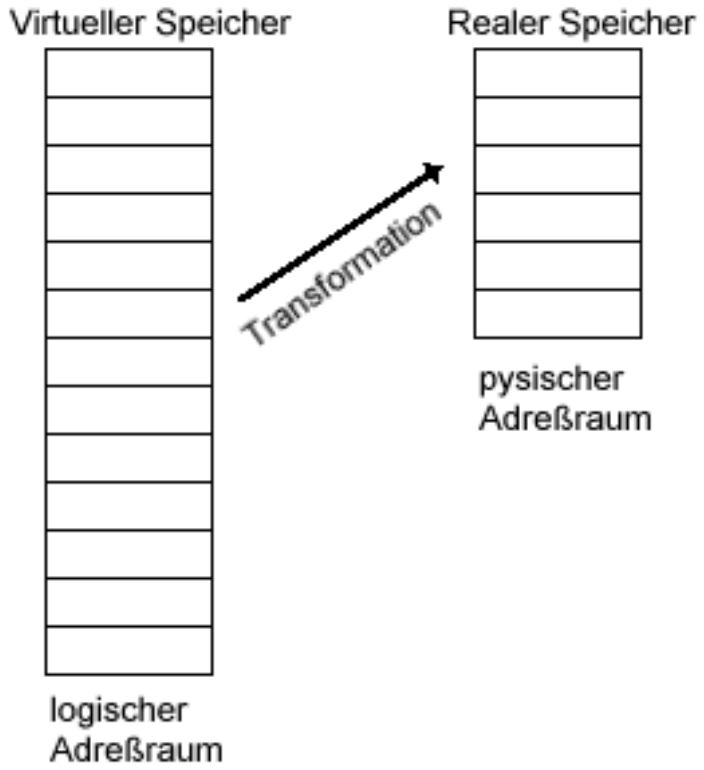


Prozess (task, thread) wird beschrieben durch:
- Zugeordnete Speicherbereiche: Programm-, Daten-, Stack-Segment
- Statusinformationen: Registerinhalte, Zustand
- Zugeordnete Ressourcen (Dateien, I/O)
- ID, Benutzer, Priorität

Ein Problem bei Multitasking: Synchronisation von Prozessen

Virtueller Speicher:

- Großer logischer Adressraum wird auf begrenzten physikalischen Speicher abgebildet
- Verwendung von logischen Adressen im Code
  -> Code ist unabhängig von der Position des Programms im Speicher
- Vergrößerung des Speichers durch Hintergrundspeicher (Festplatte)



Einteilung des Speichers in

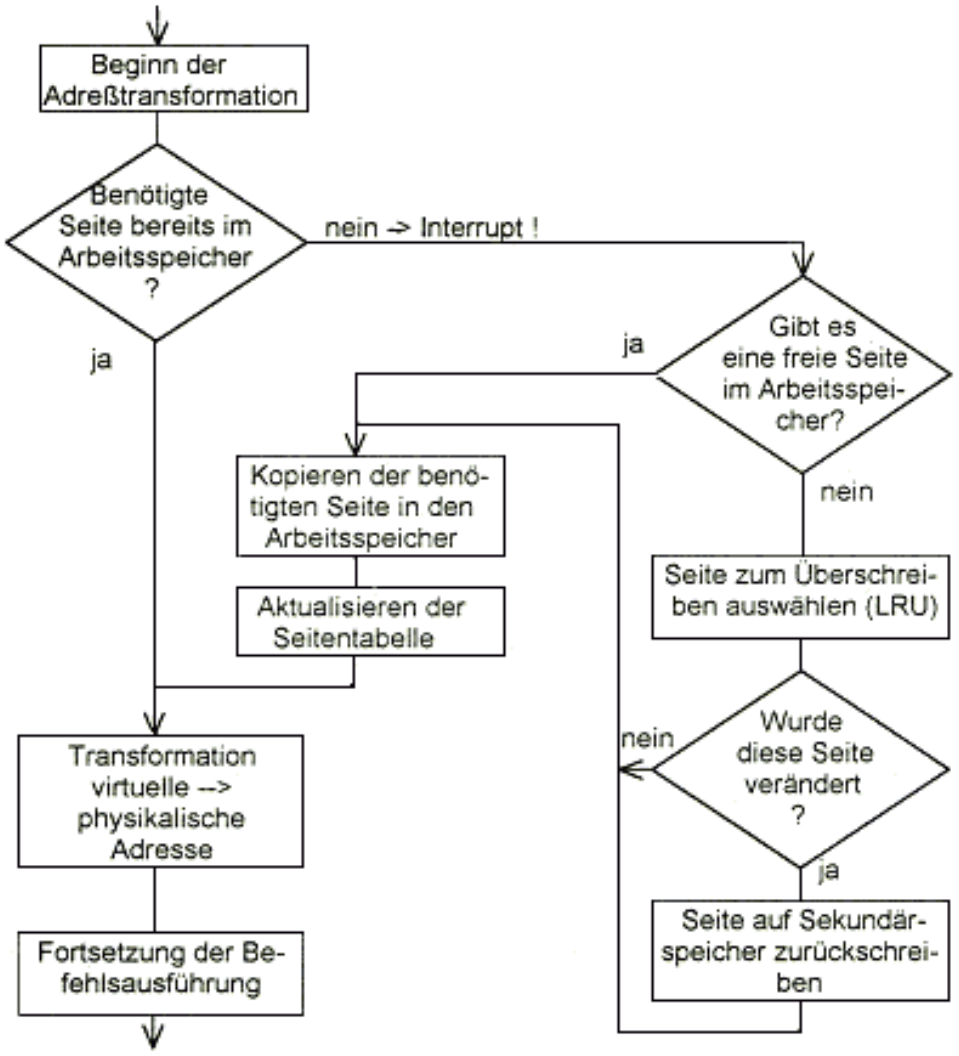
- Segmente variabler Größe
  (Problem: Externe Fragmentierung)
- Seiten fester Größe
  (Problem: Interne Fragmentierung)

Betriebssysteme – Speicherverwaltung, Swapping

CPU (memory management unit, MMU) nimmt Übersetzung von logischer in physikalische Adresse vor:

- Falls Seite nicht im Speicher, wird „Page Fault“-Interrupt ausgelöst und das OS liest die Seite vom Hintergrundspeicher
- OS entscheidet, welche Seite ersetzt wird (least recently used, dirty-bit)

- Schutz vor unbefugten Speicherzugriffen („Protection“-Interrupt)



Betriebssysteme – Peripherieverwaltung: Datenträger

Dateisystem abstrahiert von Hardwaredetails:
Speichereinheiten auf dem Datenträger (Spur/Sektor) -> Dateien

Meist realisiert als Baumstruktur von Verzeichnissen

Bereitstellung von Datei-Informationen:
- Name
- Typ
- Größe
- Eigentümer
- Zugriffsrechte
- Datum von Erstellung / letzter Änderung / letzter Lesezugriff

Unix/Linux

Linux ist eine Re-Implementierung (Open Source) von Unix
(begonnen Anfang der 90er von Linus Torvalds)

LINUX ist (wie UNIX) ein portables (in C programmiertes), recht einfach aufgebautes Betriebssystem, mit folgenden wesentlichen Eigenschaften:

-Multitasking (Prozess-Scheduling, Prozess-Kommunikation)
- Multiuser (Zugangskontrolle und Abrechnung)
- dialogorientiert
- ein Werkzeugkasten mit viele hundert Dienstprogramme
- (mehrere) Shell(s) mit mächtiger Script-Sprache;
  Scripte laufen als Unterprozesse des startenden Prozesses
- Dateisystem mit Baum-artiger Struktur, Rechtekontrolle für Datei-Zugriff
- graphische Oberfläche X-Windows (X11)
- volle Netzwerkunterstützung, incl. Netzwerk-Laufwerken, remote shells etc.