

Systemprogrammierung

Grundlagen von Betriebssystemen

Teil B – IV. Einleitung

15. Mai 2025

Rüdiger Kapitza

(© Wolfgang Schröder-Preikschat, Rüdiger Kapitza)



Lehrstuhl für Informatik 4
Systemsoftware



Friedrich-Alexander-Universität
Technische Fakultät

- Grundlegendes Verständnis der Rolle eines Betriebssystems
- Kennenlernen der Grundfunktionen eines Betriebssystems
- Einblick in die Entwicklungsgeschichte von Betriebssystemen
- *Hoffnung*: Verstehen, warum sich ein tieferes Verständnis von Systemsoftware und Betriebssystemen lohnt

Empfohlene Literatur (für heute)

- [1] **A. Silberschatz et al. Operating System Concepts (10th ed.).** Wiley, April 2018. ISBN 978-1-119-32091-3
- [2] **A. Tanenbaum, H. Bos : Modern Operating Systems (5th ed.).** Prentice Hall, 2023. ISBN-13: 9780137618880
- [3] **R. H. Arpaci-Dusseau, A. C. Arpaci-Dusseau: Operating Systems: Three Easy Pieces (v. 1.10)** <http://pages.cs.wisc.edu/~remzi/OSTEP/>

Einordnung

- Rolle des Betriebssystems

- Kurze Entstehungsgeschichte

- Grundbausteine

- Literaturauszüge

Zusammenfassung

Einordnung

Rolle des Betriebssystems

Kurze Entstehungsgeschichte

Grundbausteine

Literaturauszüge

Zusammenfassung

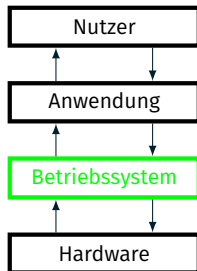
Einordnung

Rolle des Betriebssystems

Was ist ein Betriebssystem?

Ein Betriebssystem *vermittelt* zwischen Anwendungen und Hardware

- bietet eine standardisierte Schnittstelle für den Zugriff auf Ressourcen
- verwaltet die Hardware
- koordiniert die laufenden Prozesse
- behandelt Ressourcenanforderungen
- regelt die Zugriffskontrolle



- Ein Betriebssystem bietet Funktionen zum Zugriff auf Ressourcen
- Es abstrahiert von der Hardware
- Es verfügt über einheitliche Schnittstellen
 - Bspw. werden verschiedene Netzwerkkarten über dieselbe Schnittstelle angesprochen, die das Senden und Empfangen von Paketen ermöglicht

Herausforderungen

- Definition der *richtigen* Abstraktionen (z.B. auf welcher Ebene)
- Welche Hardware-Aspekte sollten offengelegt werden und in welchem Umfang?

- Ein Betriebssystem teilt die (begrenzten) Ressourcen zwischen den Anwendungen auf.
 - **Isolierung:** Anwendungen voneinander schützen
 - **Planung:** effizienten und fairen Zugang zu Ressourcen bereitstellen
 - **Beschränkung:** gemeinsamen Zugriff auf Ressourcen
- **Analogie:** Das Betriebssystem ist wie ein Kellner, der einzelne Kunden bedient. Der Kellner kennt die Speisekarte, nimmt Bestellungen auf, bringt das Essen an den richtigen Tisch und behält dabei die Rechnung im Auge.

Welche Verwaltungsfunktionen bietet ein Betriebssystem?

- **Prozessoren:** initialisiert Programmzähler/-register, teilt die Nutzung der Prozessoren auf
- **Programmspeicher:** initialisiert Prozessadressräume, lädt Programme (Code, Daten, Heap, Stack)
- **Geräte:** bspw. Lesen/Schreiben von Festplatten; Gerätetreiber sind dabei hardwarespezifisch und ermöglichen den Zugriff über einheitliche Schnittstelle

- Infrastruktursoftware für Rechensysteme
 - das Spektrum reicht von „Winzlingen“ bis hin zu „Riesen“
 - simple Prozeduren $\leftrightarrow$ komplexe Programmsysteme
 - entscheidend ist, dass Betriebssysteme nie dem Selbstzweck dienen
- jedes Rechensystem wird durch ein Betriebssystem betrieben
 - Ausnahmen bestätigen die Regel...



- Betriebssysteme sind **Handwerkszeug** der Informatik
 - mit dem umzugehen ist zur Benutzung eines Rechensystems
 - das gelegentlich zu beherrschen, anzupassen und auch anzufertigen ist



- IBM: z/VM (vormals VM/CMS), z/OS



- DEC: VAX/VMS



- DOS (16-/32-Bit)-, NT- und CE-Linie



¹Shakespeare zur Unabänderlichkeit oder Beeinflussbarkeit des Schicksals.

- funktionale und nichtfunk. Eigenschaften von Betriebssystemen werden durch die **Anwendungsdomäne** vorgegeben
 - gelegentlich passen bestehende „unspezifische“ Lösungen oder werden entsprechend angepasst (z.B. Linux)
 - häufig sind jedoch **anwendungsspezifische Lösungen** erforderlich
 - Systeme für den **Allgemeinzzweck**
 - Rechensysteme für die gängigsten Aufgaben einer Anwendungsdomäne
 - *die* Domäne der umseitig genannten Vertreter
 - Systeme für den **Spezialzweck**
 - Rechensysteme zur Steuerung oder Regelung „externer“ Prozesse
 - für gewöhnlich mit vorhersagbarem Laufzeitverhalten (**Echtzeitsysteme**)
- **Anzahl der eingebetteten Systeme** (vgl. [10])
- (das „Smartphone“ nicht mitgerechnet)

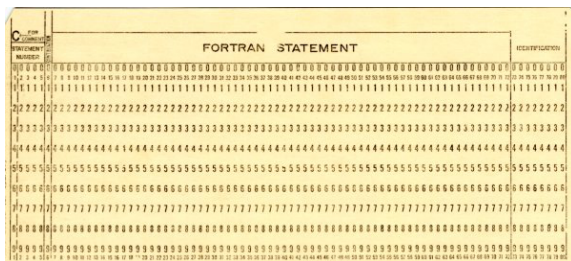


Einordnung

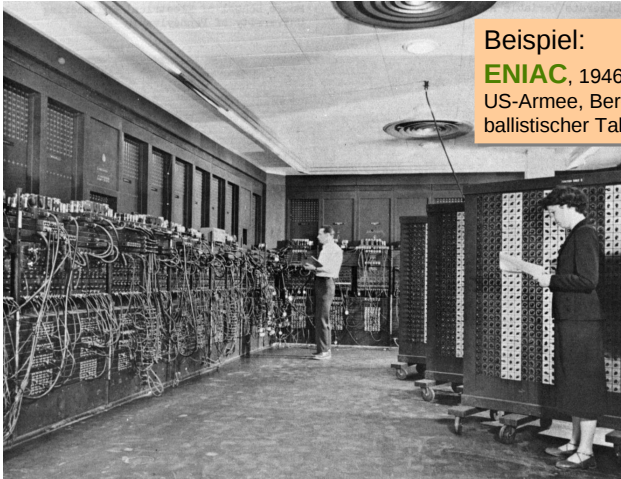
Kurze Entstehungsgeschichte

Am Anfang stand die Lochkarte

- Es gibt sie schon seit 1725 zur Webstuhlsteuerung
- Herman Hollerith nutzte sie 1890 für eine Volkszählung
 - aus seiner Firma und zwei weiteren ging später IBM hervor
- Sie wurde bis in die 70er Jahre als vielseitiger Speicher eingesetzt.



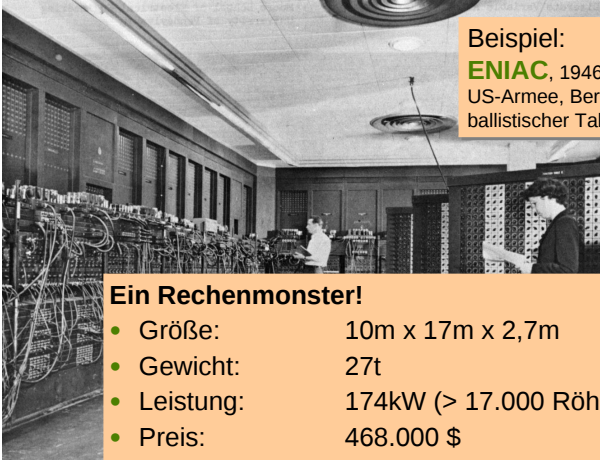
Erste elektronische Universalrechner



Beispiel:

ENIAC, 1946

US-Armee, Berechnung
ballistischer Tabellen



Beispiel:

ENIAC, 1946

US-Armee, Berechnung
ballistischer Tabellen

Ein Rechenmonster!

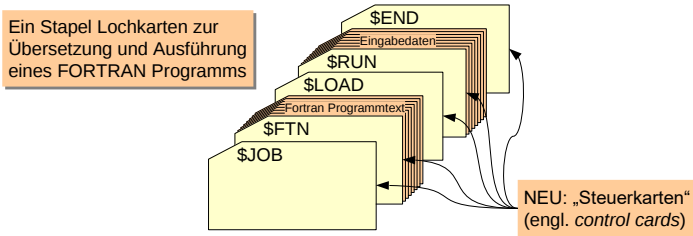
- Größe: 10m x 17m x 2,7m
- Gewicht: 27t
- Leistung: 174kW (> 17.000 Röhren!)
- Preis: 468.000 \$
- Geschwindigkeit: 500 Additionen pro Sekunde

Serielle Verarbeitung (ab 1945)

- Programmierung
 - i.d.R. in Maschinencode
 - Eingabe über Lochkartenleser
 - Ausgaben über Drucker
 - Fehleranzeige durch Kontrolllämpchen
- Rechnerzeituteilung auf Papierterminkalender
 - **Rechnerzeitverschwendung** durch zu großzügige Reservierung oder Abbruch wegen Fehler
- Minimale Auslastung der CPU
 - Die meiste Zeit verbrauchten **langsame E/A Geräte** (Lochkartenleser, Drucker)
- Erste Systemsoftware in Form wiederverwendbarer **Programmbibliotheken**
 - Binder, Lader, Debugger, Gerätetreiber, ...

Einfache Stapelsysteme (ab 1955)

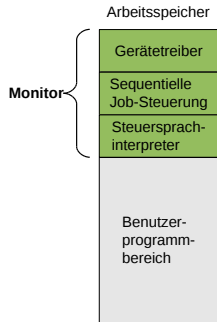
- Verringerten die Häufigkeit manueller Betriebseingriffe
- Die ersten Betriebssysteme: **residente Monitore**
 - Interpretation von *Job*-Steuerbefehlen
 - Laden und Ausführen von Programmen
 - Geräteansteuerung



Einfache Stapelsysteme (ab 1955)

Der Monitor blieb dauerhaft im Speicher, während er ein Anwendungsprogramm nach dem anderen ausführte.

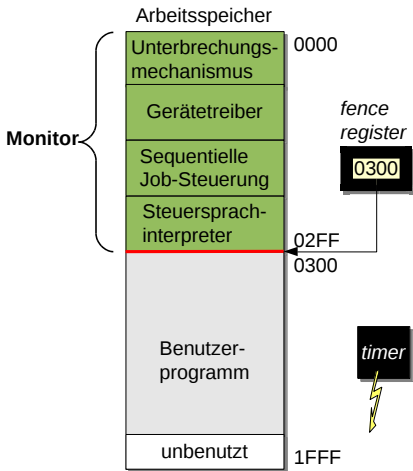
- **Probleme** durch fehlerhafte Anwendungen:
 - Programm terminiert nicht
 - Schreibt in den Speicher des residenten Monitors
 - Greift auf den Kartenleser direkt zu und interpretiert Steuerbefehle als Daten



Einfache Stapelsysteme (ab 1955)

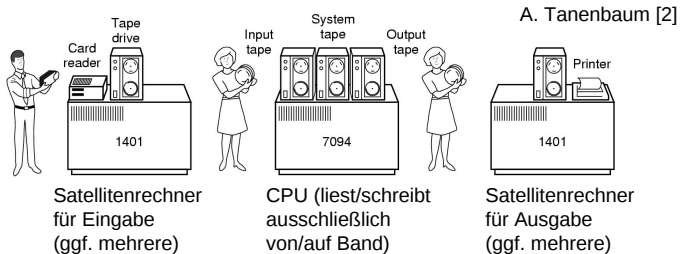
Lösungen

- Zeitgeberbaustein (*timer*) liefert **Unterbrechungen** (*interrupts*)
- **Fallen** (*traps*) für (fehlerhafte) Programme
 - Schutzgatterregister (engl. *fence register*) realisiert primitiven **Speicherschutz**
 - **Privilegierter Arbeitsmodus** der CPU (*supervisor mode*)
 - Deaktivierung des Schutzgatters
 - Ein-/Ausgabe



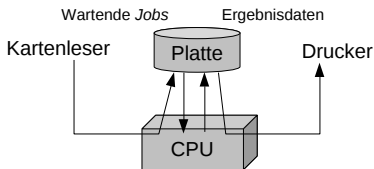
Der Ein-/Ausgabe-Flaschenhals

- **Problem:** CPU ist schneller als Kartenleser und Drucker
 - kostbare Rechenzeit wird durch (aktives) Warten verschwendet
- **Lösung 1: Off-line processing**
 - dank Bandlaufwerken
 - Parallelisierung von Ein-/Ausgaben durch mehrere **Satellitenrechner**



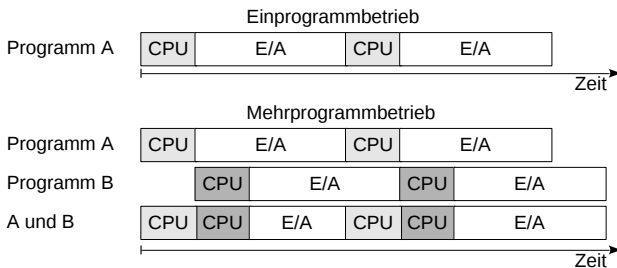
Der Ein-/Ausgabe-Flaschenhals

- **Problem:** CPU ist schneller als Kartenleser und Drucker
 - kostbare Rechenzeit wird durch (aktives) Warten verschwendet
- **Lösung 2: Spooling**
 - dank Plattenlaufwerken (wahlfreier Zugriff) und **Direct Memory Access**
 - Berechnungen und Ein-/Ausgaben werden dabei überlappt.
 - Regeln für **Prozessorzuteilung**



Mehrprogrammbetrieb (ab 1965, Teil 1)

- Trotz *spooling* nutzt ein einzelnes Programm die CPU nicht effizient.
- CPU-Stöße (*CPU bursts*) und E/A-Stöße (*I/O bursts*), bei denen die CPU warten muss, wechseln sich ab.
- Beim Mehrprogrammbetrieb bearbeitet die CPU mehrere Aufträge gleichzeitig:



;

Mehrprogrammbetrieb (ab 1965, Teil 2)

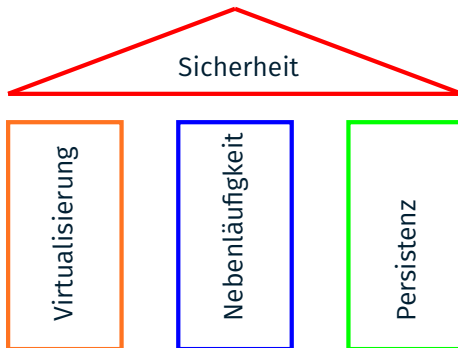
Betriebssysteme werden immer komplexer

- Umgang mit nebenläufigen E/A Aktivitäten
- **Verwaltung des Arbeitsspeichers** für mehrere Programme
- Interne Verwaltung von Programmen in Ausführung (**Prozesse**)
- **Prozessorzuteilung** (*scheduling*)
- Mehrbenutzerbetrieb: **Sicherheit** und Abrechnung (*accounting*)

Einordnung

Grundbausteine

Die Funktionalität eines Betriebssystems lässt sich in drei Säulen unterteilen, wobei Sicherheit alle Säulen umfasst.



Jede Anwendung glaubt, alle Ressourcen für sich allein zu haben

- **CPU:** unbegrenzte Anzahl von Anweisungen, kontinuierliche Ausführung
- **Speicher:** unbegrenzter Speicher verfügbar

Herausforderung: Wie können begrenzte Ressourcen gemeinsam genutzt werden?

Das Betriebssystem muss gleichzeitige Ereignisse verarbeiten und sie bei Bedarf *entwirren*

- Gleichzeitigkeit von unabhängigen Prozessen verbergen
- Gleichzeitigkeit von abhängigen Prozessen verwalten, indem Synchronisierungs- und Kommunikationsprimitive bereitgestellt werden

Herausforderung: Bereitstellung der richtigen Primitive

Lebensdauer von Informationen ist länger als die Lebensdauer eines Prozesses

- Prozesse müssen auf nichtflüchtige Informationen zugreifen
- Abstraktion der Art und Weise, wie Daten gespeichert werden (über ein Dateisystem)
- Ausfallsicherheit (z. B. bei Stromausfall)
- Zugriffskontrolle

Herausforderung: Authentifizierung und Berechtigungen

Einordnung

Literaturauszüge

*Summe derjenigen Programme, die als **residenter** Teil einer EDV-Anlage für den Betrieb der Anlage und für die Ausführung der Anwenderprogramme erforderlich ist. [7]*

Be'triebs·sys·tem <n.; -s, -e; EDV> *Programmbündel, das die Bedienung eines Computers ermöglicht. [11]*

*Der Zweck eines Betriebssystems [besteht] in der **Verteilung von Betriebsmitteln** auf sich bewerbende Benutzer. [4]*

*Eine Menge von Programmen, die die Ausführung von Benutzerprogrammen und die **Benutzung von Betriebsmitteln steuern**. [3]*

*Eine **Softwareschicht**, die alle Teile des Systems verwaltet und dem Benutzer eine Schnittstelle oder eine **virtuelle Maschine** anbietet, die einfacher zu verstehen und zu programmieren ist [als die nackte Hardware]. [9]*

*Ein Programm, das als **Vermittler** zwischen Rechnernutzer und Rechnerhardware fungiert. Der Sinn des Betriebssystems ist eine Umgebung bereitzustellen, in der Benutzer bequem und effizient Programme ausführen können. [8]*

*The operating system is itself a programm which has the functions of **shielding the bare machine** from access by users (thus protecting the system), and also of **insulating the programmer** from the many extremely intricate and messy problems of reading the program, calling a translator, running the translated program, directing the output to the proper channels at the proper time, and passing control to the next user. [5]*

*Ein Betriebssystem kennt auf jeden Fall keinen Prozessor mehr, sondern ist neutral gegen ihn, und das war es vorher noch nie. Und auf diese Weise kann man eben **jeden beliebigen Prozessor auf jedem beliebigen anderen emulieren**, wie das schöne Wort lautet. [6]*

Es ist das Betriebssystem, das die Kontrolle über das Plastik und Metall (Hardware) übernimmt und anderen Softwareprogrammen (Excel, Word, ...) eine standardisierte Arbeitsplattform (Windows, Unix, OS/2) schafft. [2]

Die Programme eines digitalen Rechensystems, die zusammen mit den Eigenschaften der Rechenanlage die Grundlage der möglichen Betriebsarten des digitalen Rechensystems bilden und insbesondere die Abwicklung von Programmen steuern und überwachen. (DIN 44300 [1]) ☺

Einordnung

Rolle des Betriebssystems

Kurze Entstehungsgeschichte

Grundbausteine

Literaturauszüge

Zusammenfassung

Resümee: Was macht ein Betriebssystem aus?

Be'triebs·sys·tem <n.; -s, -e; EDV> (*operating system*)

- eine Menge von Programmen, die
 - Programme, Anwendungen oder BenutzerInnen assistieren sollen
 - die Ausführung von Programmen überwachen und steuern
 - den Rechner für eine Anwendungsklasse betreiben
 - eine abstrakte Maschine implementieren
- verwaltet die Betriebsmittel eines Rechensystems
 - kontrolliert die Vergabe der (Software/Hardware) Ressourcen
 - verteilt diese ggf. gerecht an die mitbenutzenden Rechenprozesse
- definiert sich nicht über die Architektur, sondern über Funktionen

Resümee: Warum sich mit Betriebssystemen befassen?

- Ein Betriebssystem erstellen, modifizieren oder verwalten können
- Designentscheidungen heutiger Systemsoftware begreifen
- Systemleistung oder den Mangel davon verstehen
- Besseres Verständnis komplexer Systeme ermöglichen
- Um ein besserer (System-)Programmierer zu werden

Zusammenfassung

Bibliographie

- [1] DEUTSCHES INSTITUT FÜR NORMUNG:
Informationsverarbeitung – Begriffe.
Berlin, Köln, 1985 (DIN 44300)
- [2] EWERT, B. ; CHRISTOFFER, K. ; CHRISTOFFER, U. ; ÜNLÜ, S. :
FreeHand 10.
Galileo Design, 2001. –
ISBN 3-898-42177-5
- [3] HABERMANN, A. N.:
Introduction to Operating System Design.
Science Research Associates, 1976. –
ISBN 0-574-21075-X

- [4] HANSEN, P. B.:
Betriebssysteme.
Carl Hanser Verlag, 1977. –
ISBN 3-446-12105-6
- [5] HOFSTADTER, D. R.:
Gödel, Escher, Bach: An Eternal Golden Braid – A Metaphorical Fugue on Minds and Machines in the Spirit of Lewis Carroll.
Penguin Books, 1979. –
ISBN 0-140-05579-7
- [6] KITTLER, F. :
Interview.
<http://www.hydra.umn.edu/kittler/interview.html>, 1993

- [7] SCHNEIDER, H.-J. :
Lexikon der Informatik und Datenverarbeitung.
München, Wien : Oldenbourg-Verlag, 1997. –
ISBN 3-486-22875-7
- [8] SILBERSCHATZ, A. ; GALVIN, P. B. ; GAGNE, G. :
Operating System Concepts.
John Wiley & Sons, Inc., 2001. –
ISBN 0-471-41743-2
- [9] TANENBAUM, A. S.:
Operating Systems: Design and Implementation.
Prentice-Hall, Inc., 1997. –
ISBN 0-136-38677-6
- [10] TENNENHOUSE, D. :
Proactive Computing.
In: *Communications of the ACM* 43 (2000), Mai, Nr. 5, S. 43–50

- [11] WAHRIG-BURFEIND, R. :
Universalwörterbuch Rechtschreibung.
Deutscher Taschenbuch Verlag, 2002. –
ISBN 3-423-32524-0