

Betriebssysteme

Vorlesungsskript WS 2001/2002

Jürgen Schönwälder

Version vom 15. Oktober 2001

Fachbereich Mathematik/Informatik

Universität Osnabrück

Vorwort

Betriebssysteme sind notwendig, um auf einer Rechnerhardware geordnet Programme ausführen zu können. Sie stellen Applikationen eine logische Sicht auf die zur Verfügung stehenden Betriebsmittel bereit und realisieren grundlegende Abstraktionen (wie z.B. Dateien), ohne die informationsverarbeitende Systeme heute nicht mehr denkbar wären.

Betriebssysteme unterscheiden sich dabei recht deutlich von anderen Programmen. Sie haben in der Regel einen ganz eigenen Kontrollfluß und müssen besonderen Anforderungen hinsichtlich Performanz, Robustheit und Sicherheit genügen. In diesem Zusammenhang sind viele grundlegende Konzepte entwickelt und über die Jahre optimiert worden. Schließlich sind Problemlösungen aus dem Bereich der Betriebssysteme (Behandlung von Nebenläufigkeit) auch oftmals zur Behandlung ähnlicher Situationen in Applikationen geeignet.

Das vorliegende Dokument ist aus Vorlesungen mit dem Titel „Einführung in Betriebssysteme und Netze“ an der TU Braunschweig entstanden, die ich im Jahr 1997 entworfen habe und seither als Wahlpflichtveranstaltung im vierten Semester angeboten wird. Im wesentlichen wurde der Anteil, der sich mit Grundlagen von Kommunikationsnetzen beschäftigt, entfernt und dafür konkrete Beispiele aus dem Linux Betriebssystem eingearbeitet.

Jürgen Schönwälder

Inhaltsverzeichnis

1	Einführung	1
1.1	Betriebssystem-Dienste	2
1.2	Soft- und Hardwareschichten eines Rechensystems	3
1.3	Betriebssysteme als geschichtete abstrakte Maschinen	4
1.4	Verarbeitungsmodelle	5
1.4.1	Stapelverarbeitung (batch processing)	5
1.4.2	Dialogverarbeitung (time sharing)	6
1.4.3	Echtzeitverarbeitung (real time)	6
1.5	Benutzermode versus Betriebssystemmode	6
1.6	Trennung von Mechanismums und Strategie	6
1.7	Architekturen von Betriebssystemen	6
1.8	Geschichte der Betriebssysteme	6
1.9	Standardisierung	7
1.9.1	POSIX Standards	7

Kapitel 1

Einführung

Begriffsdefinition:

- „An operating system is similar to a government... Like a government, the operating system performs no useful function by itself.“ [5]
- Die Programme eines digitalen Rechensystems, die zusammen mit den Eigenschaften dieser Rechenanlage die Basis der möglichen Betriebsarten des digitalen Rechensystems bilden und die insbesondere die Abwicklung von Programmen steuern und überwachen. (DIN 44300)

Ziele eines Betriebssystems:

- Anpassung der Benutzerwelt an die Maschinenwelt
- Organisation und Koordination des Betriebsablaufs
- Steuerung und Protokollierung des Betriebsablaufs

Randbedingungen:

- Die vorhandenen Betriebsmittel sind so einzusetzen, daß Engpässe bzw. Überlastsituationen vermieden werden.
- Das Betriebssystem soll für die Durchführung seiner Aufgaben einen geringen Rechenaufwand benötigen. (Effizienz)
- Ein Betriebssystem hat robust gegen fehlerhafte Benutzerprogramme zu sein. (Robustheit)
- Daten und Programme sollten gegen unerlaubten Zugriff und gelegentliche Ausfälle der Datenträger abgesichert sein. (Informationssicherheit)

Die Ziele und Randbedingungen sind nicht alle gleichzeitig erfüllbar!

Einige heute gebräuchliche Betriebssysteme:

- Unix in vielen verschiedenen Varianten. Die freien Versionen stammen in der Regel aus der Linux Linie oder aus der BSD Linie (Berkeley Software Distribution). Wichtige kommerzielle Unix Systeme sind Solaris (Sun Microsystems), HP-UX (Hewlett Packard) und AIX (IBM).
- Die Familie der Microsoft Betriebssysteme Windows 95, Windows 98, Windows NT, Windows 2000 und Windows XP.
- Klassische Großrechner-Betriebssysteme wie z.B. MVS (IBM) und BS 2000 (Siemens).
- Plan9 (AT&T) und Inferno (Lucent) sind zwei neuere Betriebssysteme, die in den Bell Labs (Murray Hill, NJ) von den Leuten entwickelt wurden, die ursprünglich Unix geschaffen haben.

1.1 Betriebssystem-Dienste

Dienste für Anwendungsprogramme:

- Laden von Programmen
- Erzeugen von neuen Prozessen
- Abwicklung von Ein- und Ausgabeoperationen
- Operationen auf einem logischen Dateisystem (read, write, create, delete)
- Kommunikation zwischen Prozessen (lokal oder über Kommunikationnetze)
- Steuerung von Geräten (Terminals, Drucker, Plotter, ...)
- Abwicklung von Aufträgen im Hintergrund

Dienste für den Betreiber des Systems:

- Erfassung von Verbrauchsdaten (Accounting)
- Sicherheitsüberwachung (Zugangskontrolle, Zugriffsschutz, Verschlüsselung)
- Operationen zur Administration (z.B. Abbruch von Prozessen)
- Testfunktionen (z.B. Entdeckung fehlerhafter Spuren auf Festplatten)
- Funktionen zur Initialisierung des Systems

1.2 Soft- und Hardwareschichten eines Rechensystems

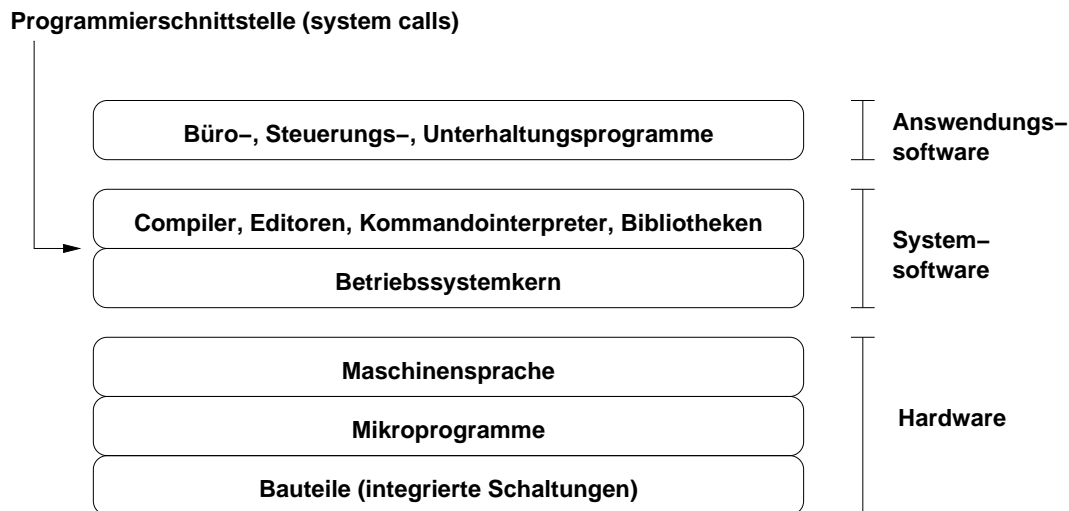


Abbildung 1.1: Soft- und Hardwareschichten eines Rechensystems

Abbildung 1.2 zeigt die typischen Schichten eines Rechensystems:

- Die *Hardware* auf der untersten Schicht besteht aus elektronischen Bauteilen (integrierte Schaltungen) mit elementaren Funktionen. In manchen Architekturen laufen auf dieser Hardware sogenannte Mikroprogramme, die die auf der Schnittstelle zur nächsten Schicht sichtbaren Befehle der Maschinensprache realisieren.
- Die *Systemsoftware* unterteilt sich in den eigentlichen Betriebssystem(kern) und eine Reihe von Hilfsprogrammen, wie z.B. Compiler, Editoren, Kommandointerpreter und Bibliotheken. Diese Vorlesung beschäftigt sich mit dem Betriebssystemkern und nicht mit den anderen Komponenten der Systemsoftware.
- Die *Anwendungssoftware* benutzt die Dienstleistungen der Systemsoftware (und indirekt der Hardware) um Probleme zu lösen (oder einfach nur zu unterhalten).

Motivation des Schichtungsprinzips:

- Schichtung dient der Beherrschung der Komplexität durch Abstraktion und macht Systeme flexibler und anpassungsfähiger.
- Schichten werden gelegentlich auch als abstrakte oder virtuelle Maschinen bezeichnet.
- Schichten erzeugen allerdings oftmals Leistungsverluste (overhead), was beim Entwurf entsprechend berücksichtigt werden muß.

Eigenschaften der Systemsoftware:

- Der Betriebssystemkern (kernel) ist ein Programm, das die ganze Zeit aktiv ist und Anwendungs- und Systemprozesse unterstützt.
- Ein Betriebssystemkern ist oftmals ein sehr großes und komplexes Softwaresystem, das in der Regel aus Komponenten und Subsystemen besteht.
- Betriebssystemkerne sind überwiegend in C geschrieben mit kleinen Teilen in Assembler.
- Systemprogramme wie z.B. Linker und Kommandointerpreter werden bei Bedarf geladen.
- Ständig existierende Hilfsprozesse werden als Dämonen (daemons) bezeichnet. Sie realisieren meist Server-Funktionen wie z.B. das Ausführen von Programmen zu vorher festgelegten Zeitpunkten oder die Verwaltung/Abarbeitung von Warteschlangen (Drucker).
- Dämonen werden entweder durch Ereignisse aufgeweckt („Datei zum Drucken eingetroffen“) oder sie schauen selbst von Zeit zu Zeit nach, ob Arbeit anliegt.

1.3 Betriebssysteme als geschichtete abstrakte Maschinen

Ein Betriebssystem stellt eine virtuelle Ablaufumgebung (abstrakte Maschine) bereit, die durch geeignete Abstraktionen die Anwendungsprogramme von der Komplexität der vorhandenen Hardware abschirmt.

Beispiele:

- Ein Betriebssystem bietet Funktionen zur Manipulation eines logischen Dateisystems, die die Applikation von den Details der Hardware (Adressen, Zylinder, Sektoren, Datenblockgrößen) abschirmen.
- Ein Betriebssystem stellt Applikationen einen linearen Adreßraum zur Verfügung und verbirgt die interne Organisation des physikalisch vorhandenen Hauptspeichers.
- Ein Betriebssystem stellt Applikationen verlässliche Kommunikationsendpunkte zum Datenaustausch bereit, ohne die erforderlichen Sicherungsmechanismen sichtbar zu machen.

Schichtungsarten:

- *Funktionelle Ersetzung:*
Der Funktionssatz F_i einer abstrakten Maschine M_x wird komplett durch den Funktionssatz F_j der abstrakten Maschine M_{x+1} ersetzt.

- *Funktionelle Erweiterung:*
Aus einer abstrakten Maschine M_x mit dem Funktionssatz F_i wird eine abstrakte Maschine M_{x+1} konstruiert, die den Funktionssatz $F_i \cup F_j$ bereitstellt.
- *Virtualisierung:*
Die Funktionalität einer abstrakten Maschine M_x bleibt in M_{x+1} unverändert erhalten. Funktional nicht erfaßbare Eigenschaften werden verbessert bzw. verallgemeinert.

Durch die konsequente Anwendung des Schichtungsprinzips lassen sich *virtuelle Maschinen* konstruieren, die eine beliebige virtuelle Ablaufumgebung bereitstellen. Einige bekannte virtuelle Maschinen sind:

- VMware virtualisiert die klassische PC Hardware und ermöglicht dadurch den gleichzeitigen Betrieb von verschiedenen Betriebssystemen.
- VM/SP ist eine virtuelle Maschine für IBM Mainframes (370/390/ESA), die Anfang der 80er Jahre auf diesen Geräten weit verbreitet war. VM/SP ermöglichte den gleichzeitigen Betrieb verschiedener Betriebssysteme (und der darauf laufenden Anwendungen) auf einer IBM Hardware.
- Java's virtuelle Maschine ist nicht an konkreter Hardware orientiert, sondern stellt vielmehr einen Interpreter für eine abstrakte Ablaufumgebung bereit. Hochsprachen werden mit Hilfe von Übersetzern (Compiler) in die Zwischensprache der Java virtuellen Maschine übersetzt, die schließlich von einem Interpreter ausgeführt wird.

1.4 Verarbeitungsmodelle

1.4.1 Stapelverarbeitung (batch processing)

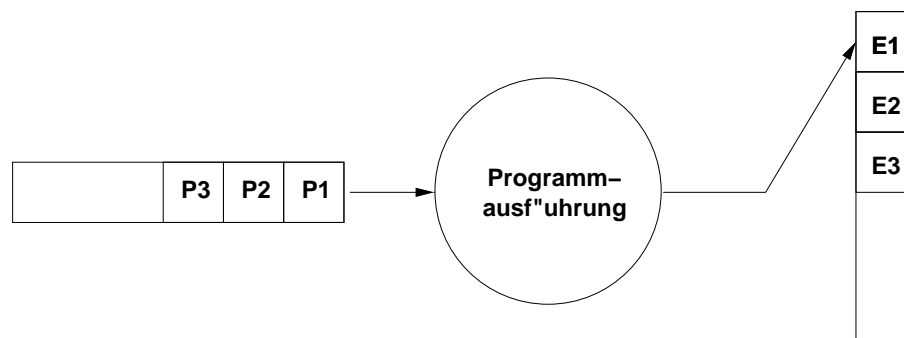


Abbildung 1.2: Stapelverarbeitung (batch processing)

Funktionsprinzip:

- Alle Aufträge werden in eine Warteschlange eingereiht.
- Aufträge werden ausgeführt, sobald sie den Kopf der Warteschlange erreichen und ein Prozessor verfügbar ist.
- Ergebnisse werden in Dateien protokolliert und ggf. ausgedruckt.
- Keine Interaktion zwischen dem Benutzer und dem Programm während der Ausführung.

1.4.2 Dialogverarbeitung (time sharing)

Funktionsprinzip:

- Mehrere Programme werden quasi-parallel ausgeführt (multi-programming, multi-tasking).
- Mehrere Benutzer können Programme gleichzeitig starten und benutzen (multi-user).
- Benutzer können ihre Programme während der Ausführung interaktiv beeinflussen.
- Die Ausführungszeiten werden in „kleine Portionen“, den sogenannten Zeitscheiben, zugeteilt. Jedes in Ausführung befindliche Programm bekommt also nur einen Teil der Hardware-Leistung zugeteilt.
- Durch eine geschickte Verzahnung lassen sich auftretende Wartezeiten (Warten auf den Abschluß eine E/A-Operation) geschickt für die Berechnungen anderer Programme nutzen.

1.4.3 Echtzeitverarbeitung (real time)

1.5 Benutzermode versus Betriebssystemmode

1.6 Trennung von Mechanismums und Strategie

1.7 Architekturen von Betriebssystemen

1.8 Geschichte der Betriebssysteme

- 1. Generation (1945 - 1955) [Röhren]
 - Manuelle Bedienung, keine Betriebssysteme
- 2. Generation (1955 - 1965) [Transistoren]

- Batchsysteme zur automatischen Abarbeitung von Aufträgen (jobs)
- 3. Generation (1965 - 1980) [Integrierte Schaltungen]
 - Spooling (Simultaneous Peripheral Operation On Line)
 - Multiprogramming
 - Time-Sharing
 - OS/360 (IBM), Multics (MIT, Bell Labs, General Electric)
- 4. Generation (1980 - 1990) [VLSI & Personal Computer]
 - Personal Computer and Workstation Betriebssysteme (CP/M, MS-DOS, Windows, OS/2, MacOS, UNIX)
 - Netzwerk Betriebssysteme (UNIX, Windows NT)
 - Verteilte Betriebssysteme (Amoeba, Mach, V)

1.9 Standardisierung

Offene Betriebssysteme benötigen Standards, die den Funktionsumfang und die Schnittstelle des Betriebssystems definieren. Proprietäre Systeme bedürfen normalerweise keiner Standardisierung, da der Funktionsumfang vom Hersteller bestimmt wird und durch die Versionsnummer identifiziert werden kann. Einige relevante Standards:

AT&T	System V Interface Definition (SVID)	1985
OSF	Distributed Computing Environment (DCE) 1.1	1995
OSF	Motif 2.0 Graphical User Interface Toolkit	1994
X/OPEN	Portability Guide (XPG-1, ..., XPG-4)	1984
IEEE	Portable Operating System Interface (POSIX)	1989
OpenGroup	Single Unix Specification Version 2	1997
OpenGroup	Common Desktop Environment (CDE) 1.0	1996
ANSI/ISO	Programmiersprache C (ISO/IEC 9899:1999)	1999
ANSI/ISO	Programmiersprache C++ (ISO/IEC 14882:1998)	1998

Die Open Software Foundation (OSF) ist später in der OpenGroup aufgegangen.

1.9.1 POSIX Standards

Literaturhinweise

Klassische Lehrbücher zum Thema Betriebssysteme sind [6, 7, 5, 2, 1, 3].

Die meisten Bücher orientieren sich stark am Unix Betriebssystem. Das Buch von Stallings [6] enthält auch einiges an Informationen über Windows Betriebssysteme. Eine gute Übersicht über die ersten 25 Jahre Unix-Entwicklung findet man in [4].

Literaturverzeichnis

- [1] G. Coulouris, J. Dollimore, and T. Kindberg. *Distributed Systems: Concepts and Design*. Addison Wesley, 1994.
- [2] M. K. McKusick, K. Bostic, M. J. Karels, and J. S. Quarterman. *The Design and Implementation of the 4.4 BSD Operating System*. Addison Wesley, 1996.
- [3] J. Nehmer and P. Sturm. *Systemsoftware*. dpunkt, 1999.
- [4] P. H. Salus. *A Quarter Century of UNIX*. Addison Wesley, 1994.
- [5] A. Silberschatz and P. Galvin. *Operating System Concepts*. Addison Wesley, 5 edition, 1998.
- [6] W. Stallings. *Operating Systems*. Prentice Hall, 4 edition, 2001.
- [7] A. S. Tanenbaum. *Modern Operating Systems*. Prentice Hall, 1992.