1.1 Anwendung/Einführung

1.1.1 Was ist ein Betriebssystem, was macht ein Betriebssystem?

Die Definition von Betriebssystem (=*operating system*, auch OS, BS, Betriebsprogramm, Systemsoftware) in einem Satz ist schwierig. Am einfachsten kann man beschreiben, was es *nicht* ist. Man kann es abgrenzen von der

- Hardware (alle nicht programmierbaren Bausteine),
- Entwicklungssoftware (Programme zum Schreiben neuer Programme),
- Anwendersoftware (Programme, die vom Anwender eines Systems willentlich aufgerufen und beendet werden).

Eine Zusammenstellung der Aufgaben eines Betriebssystems kann am ehesten definieren, was ein Betriebssystem ist:

- Betriebsmittel (=Ressourcen, z.B. Prozessor, RAM, Platz auf der Festplatte, Netzwerkkarte, Drucker) bereitstellen, verwalten und zuteilen,
- Für Programme (und Programmierer) eine Möglichkeit des Hardware-Zugriffs ohne genaue Kenntnis der Hardware zur Verfügung stellen (Hardware-Abstraktion),
- Prozesse (Programme) auf Benutzerwunsch starten und beenden,
- Informationen und Eingriffsmöglichkeiten für Benutzer zur Verfügung stellen.

Manche Betriebssysteme erfüllen nicht alle diese Aufgaben, so hatte z.B. MS-DOS nur eingeschränkte Möglichkeiten der Prozessor-Zuteilung. Ebenso gibt es Systeme, die überhaupt kein Betriebssystem brauchen, da sie entweder nur ein Programm dauerhaft betreiben (z.B. *embedded systems*) oder ein Mini-Betriebssystem bereits im Programm enthalten ist (z.B. frühe Versionen des MS-Flugsimulators).

1.1.2 Warum soll man etwas über Betriebssysteme lernen?

Erstes Argument: Betriebssystem kommen und gehen. MS-Windows 10 wird vermutlich nicht das letzte Betriebssystem der Geschichte sein. Zweites Argument: IT-Systeme unterliegen dem technischen Wandel. Desktop-PCs, Notebooks und Smartphones sind vermutlich nicht die letzten Massen-Computer der Geschichte. Parallel zum Aufkommen neuer Rechnertypen wurden andere Typen verdrängt, die im Vergleich zu groß und zu teuer waren, deren (ebenfalls gesteigerte) Leistungsfähigkeit aber nicht mehr gebraucht wurde. Desktop-PCs und Notebooks können verdrängt werden – oder wurden bereits verdrängt – durch:

- Neue PCs mit modernerer, daher inkompatibler Hardware (der Siegeszug der ARM-Prozessoren weist in diese Richtung)
- PC-on-a-Chip zu geringem Preis mit On-Chip-Festplatten-Ersatz (SSDs weisen in diese Richtung)
- massenhaften Einsatz billiger *embedded systems* für ursprüngliche PC-Aufgaben (der Einsatz zahlloser *embedded systems* (meist mit Linux) für Routing, DSL-Einwahl, Monitor-Ansteuerung, Druckerserver, Bluetooth-Lautsprecher, ... ist ein Beispiel)
- Tablet-PCs und Smartphones
- IT-Systeme, die man am Körper oder in der Kleidung trägt (*wearables*)

Es lassen sich auch Argumente finden, warum man nichts über Betriebssysteme zu lernen braucht:

- Betriebssystem-Kurse verkommen leicht zu einfachen Produktschulungen. Diese sind aber Aufgabe der Betriebssystem-Lieferanten
- Durch neue Versionen ändert sich der Umgang mit Betriebssystemen laufend. Das Betriebssystem-Wissen veraltet zu schnell
- Viele Betriebssysteme (z.B. Windows) lassen keinen Einblick in Interna zu; insofern kann man kein tieferes Wissen erlangen, sondern wird bewusst dumm gehalten (und fährt teils gut damit)
- Tipps- und Tricks-Literatur existiert in Massen und hilft bei Problemen schnell weiter
- Allgemeine Theorie über Betriebssysteme ist oft abstrakt und hilft bei praktischen Problemen nicht weiter
- zudem hinken aktuelle Betriebssysteme hinter dieser Theorie oft um viele Jahre hinterher, so dass sich dieses Wissen oft erst nach vielen Jahren auszahlt (wenn überhaupt)

1.1.3 Einordnung von Betriebssystemen

1.1.3.1 Technik Zunächst kann man Betriebssysteme nach (technischer) Ähnlichkeit zueinander einteilen:

- UNIX-ähnliche Systeme
  - AT&T UNIX, IBM AIX, SUN Solaris, HP-UX: kommerzielle Systeme für große Anlagen
  - MINIX, GNU/Linux, Android (baut auf Linux auf): Open Source
  - BSD UNIX: Extrem sichere Systeme für Netzwerke (FreeBSD, OpenBSD, NetBSD)
  - Apple MacOS X, NextStep
- Windows-ähnliche Systeme
  - MS-DOS 1.0-6.2 (ggf. mit Windows 1.0–3.11)
  - MS-Windows 7/8/10 und Server20xx
- Sonstige
  - Apple MacOS (vor Version X)

1.1.3.2 Gleichzeitigkeit Ein Aspekt für die Unterteilung von Betriebssystemen ist die Gleichzeitigkeit; davon werden die technischen Möglichkeiten des Systems beeinflusst:

- Mehrbenutzersystem (Multiuser-): mehrere Benutzer gleichzeitig
- Mehrprozesssystem (Multitasking-): mehrere Prozesse gleichzeitig

Tabelle 1 zeigt die Möglichkeiten.

Arten von Betriebssystemen

System	Multiuser	Singleuser
Multitasking	Unix, Linux, MacOS, Windows 10	Windows ME, 2000
Singletasking	—	MS-DOS

Tabelle 1: Arten von Betriebssystemen

1.1.3.3 Was darf der Benutzer? Ein anderer Aspekt ist die Frage nach den rechtlichen Möglichkeiten im Umgang mit dem Betriebssystem: Kann der Quelltext eingesehen oder sogar verändert werden? Werden Nutzungsgebühren für jeden Computer (Windows) oder sogar für jeden Benutzer (SCO-Unix) erhoben? Ist nur eine bestimmte Gruppe zu Änderungen berechtigt, oder kann jeder Änderungen einbringen?

- MS-Windows, IBM OS/2, ... : Source nicht-öffentlich, nicht veränderbar
- Unix: Source nicht-öffentlich, aber veränderbar
- Free BSD: Source öffentlich und veränderbar; geschlossene Entwicklergruppe
- Linux: Source öffentlich und veränderbar; offene Entwicklergruppe

1.1.3.4 Verfügbarkeit Ein dritter Aspekt ist die Verfügbarkeit von Betriebssystemen auf unterschiedlicher Hardware. Betriebssysteme, bei denen dies berücksichtigt wurde, lassen sich erfahrungsgemäß schneller an neue Hardware anpassen. Bei kommerziellen Betriebssystemen dient die Unterstützung spezieller Hardware oft nur firmen-strategischen Zielen.

- MacOS X – Apple-Hardware, PCs (Desktop und Notebook) mit Intel-artiger CPU
- Linux – fast alle Hardware-Architekturen (ab 32-Bit-CPUs)
- Windows 10 – normale PCs (Desktop und Notebook) mit Intel-artiger CPU

1.1.4 Aufbau eines Betriebssystems

Die vier genannten typischen Aufgaben eines Betriebssystems

- Betriebsmittelverwaltung (CPU-Zeit, RAM usw.),
- Hardware-Abstraktion für Programme,
- Programm-Start-Möglichkeit für Benutzer,
- Informationen und Hilfen für Benutzer,

finden sich auch in seinem Aufbau wieder (Abbildung 1).

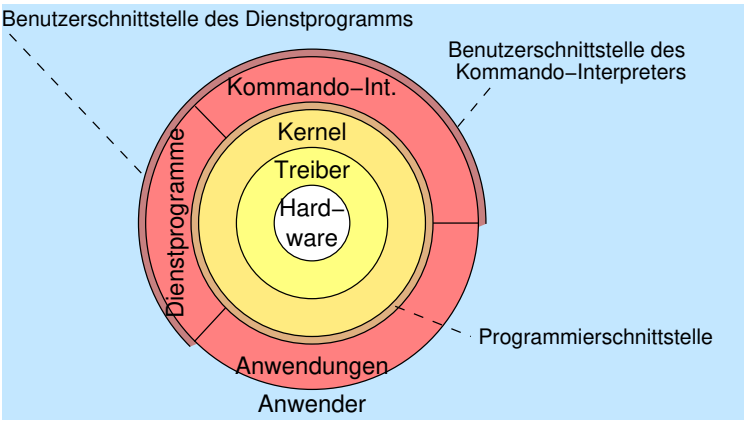

Abbildung 1: Aufbau eines Betriebssystems

1.1.4.1 Kernel Der Kernel dient zusammen mit den Treibern zur Abschirmung der Hardware-Eigenheiten, zur Organisation der Massenspeicher in Form eines Dateisystems (Dateien und Verzeichnisse) zum Festlegen von Zugriffsschutzmechanismen (Dateiberechtigungen und Dateieigentum) und zur Organisation der Prozessor- und Speicherressourcen.

Der Kernel kann entweder ein einzelnes Programm sein (Windows, Linux) oder aus einem Micro-Kernel und vielen davon unabhängigen Modulen bestehen (MacOS X, NextStep, Minix 3) — in diesem Fall kann jedes Modul auch dann noch arbeiten, wenn in einem anderen Modul ein Problem aufgetreten ist. Dies bietet höhere Sicherheit.

An der Außenseite des Kernels sieht man die Programmierschnittstelle für Anwenderprogramme (=API, *application program interface*). Wenn zwei Betriebssysteme die gleiche API benutzen, dann sind im Prinzip auch die entsprechenden Anwenderprogramme zueinander kompatibel. Der Anwendungsprogrammierer sollte sie kennen, der System- und der Netzwerkprogrammierer muss diese Schnittstelle beherrschen.

1.1.4.2 Betriebssystemoberfläche: CLI und GUI Für den Benutzer gibt es einen weiteren Betriebssystemteil, der für ihn das Betriebssystem schlechthin zu sein scheint: Die Benutzeroberfläche des Betriebssystem.

Bei den meisten Betriebssystemen für Desktop-PCs und Notebooks handelt es sich um ein Graphik-Programm, welches es erlaubt, mit Hilfe einer Zeigemöglichkeit (Maus, Trackball, Touchscreen, ...) bestimmte Prozesse zu starten (z. B. durch Doppelklick) und zu beenden (z. B. Anklicken eines Symbols im Fenster des Prozesses). Dieses Programm ist dann das *graphical user interface* (abgekürzt GUI-Programm) des Betriebssystems.

Bei Windows ist dieses Programm eine Instanz des *Windows-Explorers* (mit entsprechenden Parametern aufgerufen). Bei Linux nennt man dieses Programm den *file manager*.

Daneben gibt es bei nahezu allen Systemen noch eine zweite Steuerungsmöglichkeit, das *command line interface* (abgekürzt CLI), auch Kommandointerpreter genannt. Es ist textorientiert und erlaubt es, gezielt konkrete Aufträge an das System abzugeben.

Bei Windows dient dazu das Programm `CMD.EXE`[1], bei Unix und Linux im Textmodus nennt man das entsprechende Programm *shell*. Dieses Programm liest jeweils eine Zeile von der Tastatur, interpretiert deren Inhalt und versucht ihm dann Folge zu leisten. Dieser Kommandointerpreter ermöglicht es nun dem Benutzer, beliebige Programme zu starten; nach ihrem Ende kehrt der Benutzer wieder zum Kommandointerpreter zurück.

Für den gelegentlichen Benutzer eines Systems ist ein GUI besser geeignet. Vielnutzer und Systemadministroren schätzen dagegen oft die schnelle und gezielte Arbeitsweise eines CLI/Kommandointerpreters.

1.1.4.3 Dienstprogramme Die Informationen und Hilfen für den Benutzer (Wie viel Platz ist auf meiner Platte noch frei? Wer ist noch auf diesem System eingeloggt? Ist Post für mich da? Ich möchte gern alle meine Dateien löschen!) sind oft in vielen kleinen so genannten Dienstprogrammen (*utilities*) zusammengefasst. Jedes dieser Programme ist eigenständig und kann vom Benutzer gezielt aufgerufen werden. Manche Dienstprogramme werden von externen Firmen angeboten, so z. B. Virenscanner.

[1] und die so genannte *Power-Shell*