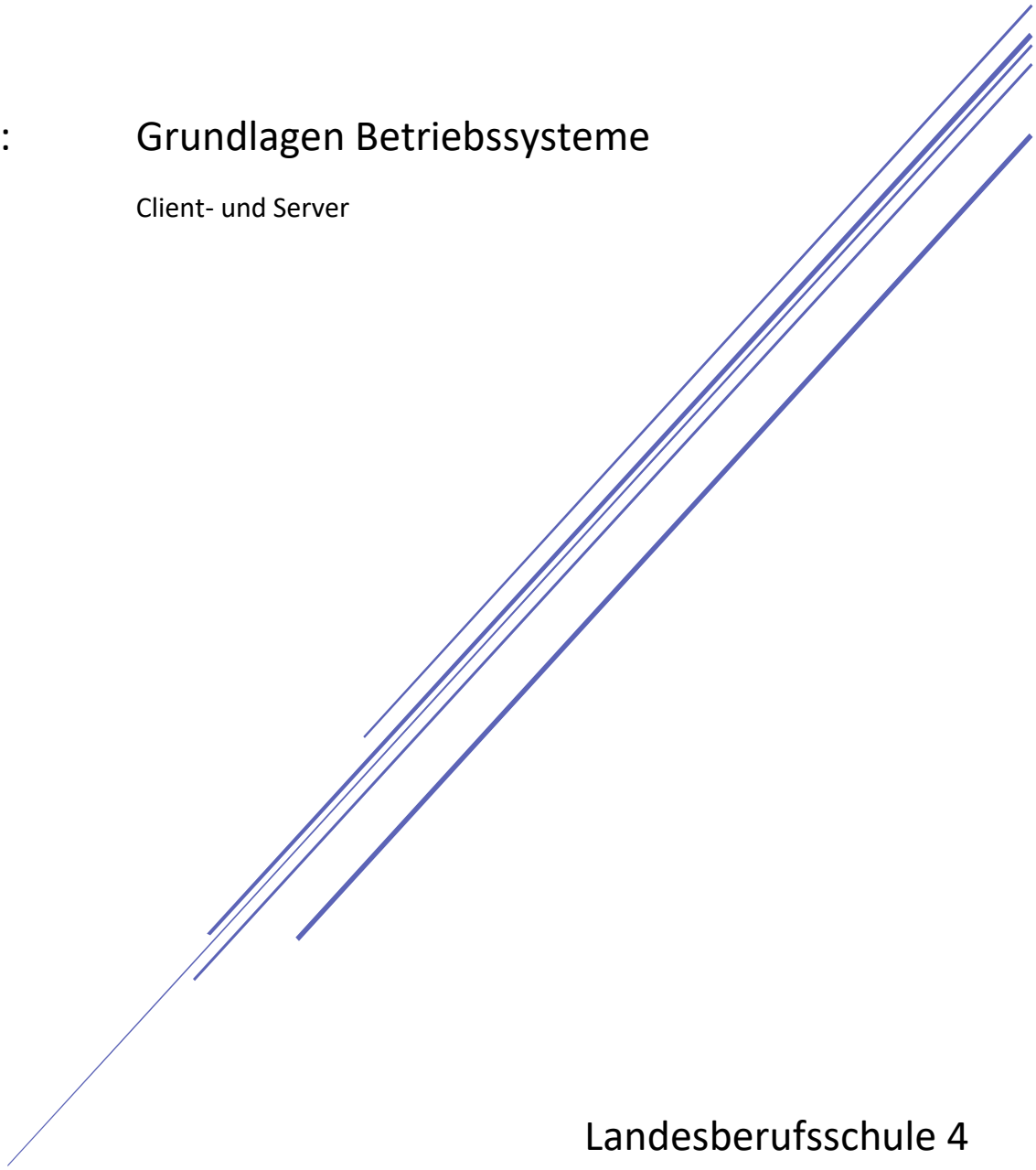
DTS 3

Thema: Grundlagen Betriebssysteme

Client- und Server



Landesberufsschule 4

Inhalt

1 Lernziele

Ich kann / kenne

- den Aufbau eines Betriebssystems erläutern
- das Schichtenmodell eines Computersystems beschreiben
- die Aufgaben eines Betriebssystems erläutern
- die möglichen Benutzeroberflächen erklären
- Arten von Betriebssystemen erläutern

2 Einleitung

Die Entwicklung von Betriebssystemen ist ganz eng mit der Entwicklung der Computer-Hardware und deren Leistungsfähigkeit verknüpft. Während die ersten Rechner (spezialisiert auf genau eine Aufgabe) noch ohne Betriebssystem auskahmen, wurde spätestens seit der Umsetzung der „Von-Neumann“ Prinzipien und der Entwicklung sogenannter "Universalrechner" eine unabhängige Schicht benötigt, die die Ressourcen verwaltet. Diese Aufgabe übernimmt das Betriebssystem.

Waren früher (zur Zeit der ”Alleinherrschaft der Großrechner”) die Betriebssysteme nur wenigen EDV-Spezialisten bekannt, sind spätestens seit dem Siegeszug der PCs und deren Vernetzung, die unterschiedlichsten Betriebssysteme (mit all Ihren Vor- und Nachteilen) auf dem Markt erhältlich. EDV-Spezialisten sollten die Kompetenz haben, die Vor- und Nachteile von unterschiedlichen Betriebssystemen beurteilen zu können.

3 Betriebssysteme

Ein Rechnersystem besteht zunächst aus physikalischen sichtbaren Elementen – der Hardware. Damit man die Hardware einsetzen kann benötigt man auch noch geeignete Software. Ein Betriebssystem realisiert die Schnittstelle zwischen dem Benutzer (Anwendungsprogrammen) und der physischen Rechenanlage. Es ist das grundlegende Computerprogramm, steuert die Hardware, koordiniert die Ressourcenzugriffe der Anwendungsprogramme und stellt dem

Benutzer Steuerungsmöglichkeiten zur Verfügung. Da aber nicht jeder Softwareentwickler sich um alle Details zur Verwaltung der einzelnen Hardwarekomponenten kümmern sollte, erscheint es sinnvoll, diese Komponenten zentral und damit allgemein verfügbar bereitzuhalten.

3.1 Mögliche Definition für Betriebssysteme

Tanenbaum: Moderne Betriebssysteme
Aufgabe eines Betriebssystems ist es Geräte zu verwalten und Benutzerprogrammen eine einfache Schnittstelle zur Hardware zur Verfügung stellen.
Den Anwendungsprogrammen statt der realen Betriebsmittel, virtuelle Betriebsmittel zur Verfügung zu stellen. (z.B.: virtueller Speicher)

Je nach Art der Hardware gibt es sehr unterschiedliche Typen von Betriebssystemen. Sie reichen von Großrechner, über Server-BS bis hin zu PC-BS und eingebetteten Systemen (Embedded-Systems)

3.2 Von Neumann Prinzip

1946 wurde von John von Neumann ein Prinzip vorgestellt um einen universellen Rechner zu gestallten. Der Rechner ist nach folgenden Prinzipien aufgebaut.

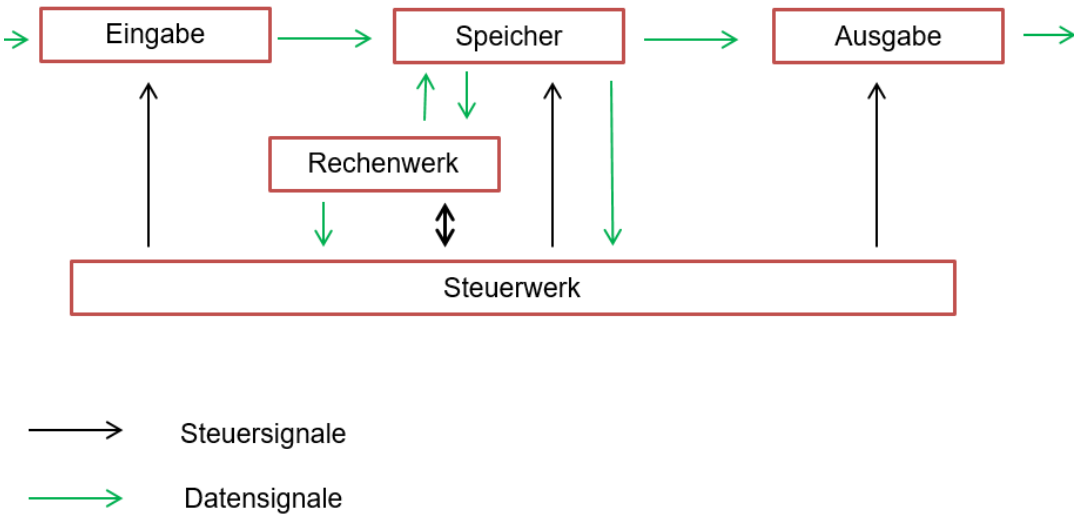


Abbildung 1: Von Neumann Prinzip

- Rechner besteht aus Eingabe, Speicher, Ausgabe, Rechenwerk und Steuerwerk

- Programme, Daten und Ergebnisse werden im gleichen Speicher abgelegt
- Der Speicher ist in gleichgroße Blöcke eingeteilt und fortlaufend nummeriert
- aufeinanderfolgende Befehle eines Programmes werden in aufeinanderfolgende Speicherblöcke abgelegt
- Durch Sprungbefehle kann von der gespeicherten Reihenfolge abgewichen werden
- Es gibt mindestens arithmetische-, logische-, und Transportbefehle
- alle Daten werden binär codiert

3.3 Schichtenmodell eines Computers

Die zunehmende Vielfalt der eingesetzten Hardware machte es erforderlich, den Programmiererinnen/Programmieren ein Hilfsmittel zur Verfügung zu stellen, damit man ohne spezielle Kenntnisse, Zugriff auf die Hardwareressourcen bekommt. Als Lösung, hat man um die reine Hardware eine Softwareschicht gelegt, die dem Anwender bzw. Entwickler die Hardware-Details verbirgt. Mittels definierter Schnittstellen werden Zugriffe auf die Hardware ermöglicht. In Abbildung 2 ist die Sicht als Benutzer zu sehen.

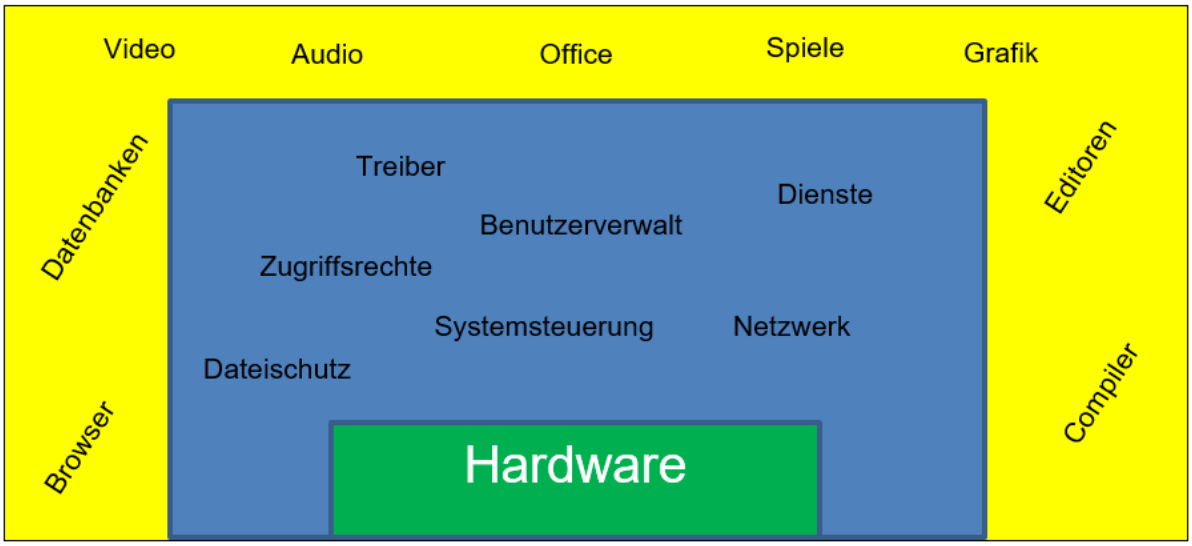


Abbildung 2: Aufbau Betriebssystem (Benutzersicht)

In der IT wird das Betriebssystem in weitere Schichten unterteilt. In Abbildung 3 ist ein mögliches Schema zu sehen.

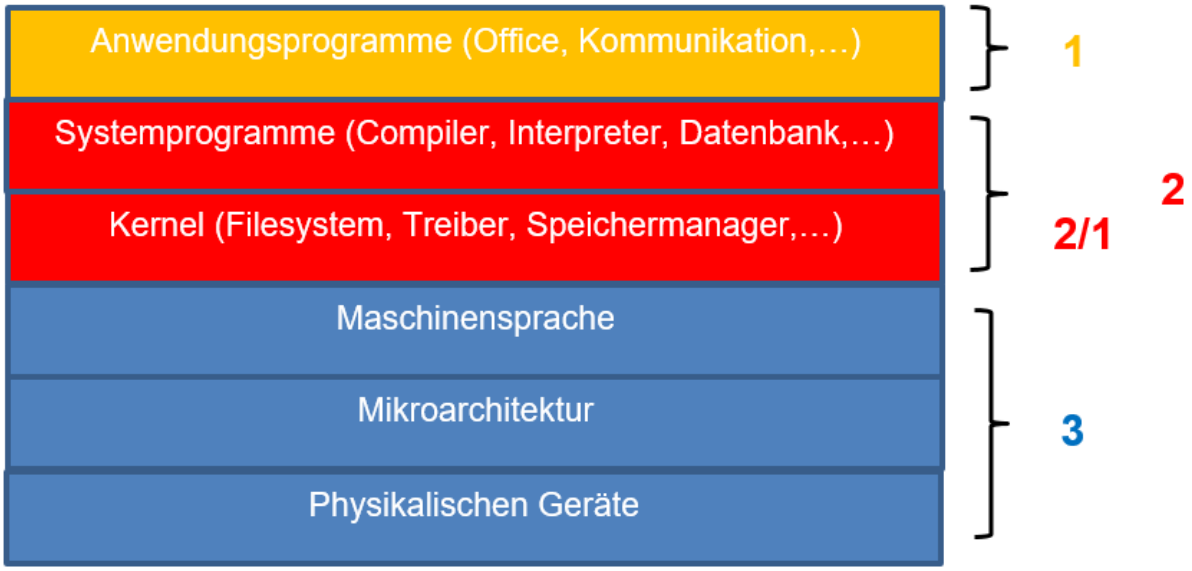


Abbildung 3: Aufbau Betriebssystem (IT-Sicht)

1 Anwendungsprogramme

2 Systemprogramme, 2/1 Betriebssystem

3 Hardware

3.3.1 Physikalische Geräte

Die unterste Schicht besteht aus den physikalischen Geräten. Sie bestehen aus Drähten, Stromversorgung, Schaltungen, usw.

Komponenten der Ebene sind:

- Prozessor
- Speicher
- Netzwerkschnittstellen
- usw.

3.3.2 Mikrocode

Der Mikrocode steuert die physikalischen Geräte direkt an. Sie werden von einem Interpreter in entsprechende Steueranweisungen übersetzt. Die Befehle der Maschinensprache werden einzeln, nacheinander ausgeführt.

Mikroprogramme werden sehr oft in ROMs abgelegt.

- ADD →addiere die Werte in den Registern eax und eax (add eax, eax)
- MOVE → setze den Wert in eax auf den Wert in eav (mov eav, eax)
- JUMP → springe zu Adresse 0x0012 (jmp 0x0012)

3.3.3 Maschinensprache

Die Menge von Instruktionen die ein Mikroprogramm ausführen kann wird als Maschinensprache bezeichnet. Maschinensprachen verfügen bis zu 300 verschiedene Befehle. Sie ist hardwarespezifisch und wird dadurch der Hardware zugeordnet.

3.3.4 Betriebssystemkern (Kernel)

Stellt die wichtigsten Funktionen des Betriebssystems dar. Der Kernel steuert unteranderem Prozess- und Speicherzugriffe und greift direkt auf die Hardware zu. Er ist die unterste Schicht des Softwaresystems und übernimmt die Kommunikation zwischen Hardware und Software.

3.3.5 Systemprogramme / Dienste

Dabei handelt es sich um Programme, die nicht zum Betriebssystem bzw. Anwendungsprogrammen gehören. Eigenschaften sind die Austauschbarkeit und zusätzlich die Systemnähe. (Interpreter, Compiler, Linker, usw.)

3.3.6 Anwenderprogramme

Stellen die Gesamtheit der Programme, die für Lösung von anwendungsspezifischen Aufgaben dienen, dar. Diese werden in Gruppen eingeteilt, wie z.B.: Office Programme, Bildverarbeitungsprogramme, CAD (Computer Aided Design) -Programme, usw.

3.4 Geschichte der Betriebssysteme

Laut Andrew S. Tanenbaum gibt es 4 verschiedene Generationen von Betriebssystemen die nachfolgend vorgestellt werden.

3.4.1 1. Generation (ca. 1945 bis 1955)

Diese Generation war durch Röhrencomputer (ca. 20000 Röhren) und der Programmierung in Maschinensprache geprägt. Ab 1950 wurde die Programmierung von Lochkarten abgelöst.

3.4.2 2. Generation (ca. 1955 bis 1965)

Computer bestanden schon aus Transistoren. Die Verarbeitung wurde mittels Stapelverarbeitung (Batch-Verarbeitung) durchgeführt. Die ersten dieser Art wurden von IBM gefertigt. Das Programm wurde von Lochkarten eingelesen, auf einem Magnetband gespeichert Und hintereinander abgearbeitet. Die Ergebnisse wurden wieder auf einem Magnetband gespeichert und anschließend ausgedruckt.

3.4.3 3. Generation (1965 bis 1980)

Erstmals wurden „Integrierte Schaltkreise“ (IC) für Rechnersysteme verwendet. Die bekanntesten waren IBM System/370 3080 und 3090, diese beherrschten Multitasking (Mehrprogrammbetrieb). Mehrere Programme liefen gleichzeitig oder „quasi-parallel“ im System ab. Während der I/O Zeiten wurde die CPU mit neuen Aufträgen versorgt. Die Jobs wurden auf Plattenspeichern zwischengespeichert. Diese Verfahren wurde Spooling genannt. Eine Weiterentwicklung war der Mehrbenutzerbetrieb (Timesharing).

3.4.4 4. Generation (1980 bis heute)

In dieser Generation wurden Personal Computer (PC) und Server entwickelt. IC’s wurden weiterentwickelt. Die Giant-Large-Scale-Integration erlaubte das Aufbringen von Millionen Transistoren auf einem Silizium Chip. Die Hardware wurde immer kleiner und dadurch schneller. Betriebssysteme waren anfangs MS-DOS, Unix, IBM OS/2.

Tabelle 1:IC-Integration

Abk.	Bezeichnung	Komplexität (Gatteräquivalente)		
		typische Interpretation	Tanenbaum	Texas Instruments
SSI	small scale integration	10	1–10	unter 12
MSI	medium scale integration	100	10–100	12–99
LSI	large scale integration	1.000	100–100.000	100–999
VLSI	very large scale integration	10.000–100.000	ab 100.000	ab 1.000
ULSI	ultra large scale integration	100.000–1.000.000	—	—
SLSI	super large scale integration	1.000.000–10.000.000	—	—
ELSI	extra large scale integration	10.000.000–100.000.000	—	—
GLSI	giant large scale integration	> 100.000.000	—	—

3.5 Anforderungen an Betriebssysteme

Betriebssysteme sind verantwortlich für den koordinierten Ablauf einer Interaktion auf einem Rechnersystem. Die nachstehenden Punkte soll ein modernes Betriebssystem erfüllen.

- Verbergen der Komplexität der Maschine
- Bereitstellen von Benutzerschnittstellen
- Bereitstellen von Programmierschnittstellen (API)
- Verwaltung der Ressourcen des Rechnersystems
- Koordination von Prozessen
- Rechteverteilung

3.6 Betriebssystemarchitekturen

Betriebssysteme sind sehr komplexe Softwaresysteme und meistens in Module bzw Komponenten strukturiert. Vom Betriebssystemkern gesehen unterscheidet man verschiedene Arten, wie Kernels aufgebaut sind. Prinzipiell gibt es zwei verschiedene Modi wie Software am Prozessor ablaufen kann.

Privilegierter-Modus: wird auch Kernelmodus genannt. Dabei werden Teile vom Betriebssystem ausgeführt die einem gewissen Schutz unterliegen. Dadurch werden Datenstrukturen und Codeteile vor Zugriff von Anwendersoftware geschützt.

Nicht-privilegierter-Modus: wird auch Benutzermodus genannt. Darin laufen üblicherweise Anwendungsprogramme ab. Diese haben keinen direkten Zugriff auf kernelspezifische Teile. Sie müssen die Schnittstellen zu Systemdienste oder Systemprogramme verwenden.

3.6.1 Monolithischer Kernel

Das Betriebssystem ist in eine Menge von Prozeduren organisiert. Jede Prozedur kann von jeder aufgerufen werden. Es sind fast alle Funktionen des Betriebssystems in den Kern implementiert. (Speicher- und Prozessverwaltung, Funktionen zur Kommunikation zwischen den Prozessen sowie Treiber für die Hardwarekomponenten)

Vorteil: sehr schnell, weil keine zusätzlichen Programme benötigt werden.

Nachteil: Monolithische Kernel sind fehleranfälliger, weil ein abgestürzter Teil nicht einfach neu gestartet werden kann.

Bekannte Betriebssysteme sind Linux und Unix. Linux hat in seiner Entwicklung die Gefahr des monolithischen Kernels erkannt und strukturiert Teile davon in Module. In Abbildung 4 ist ein Bild einer monolithischen Struktur zu sehen.

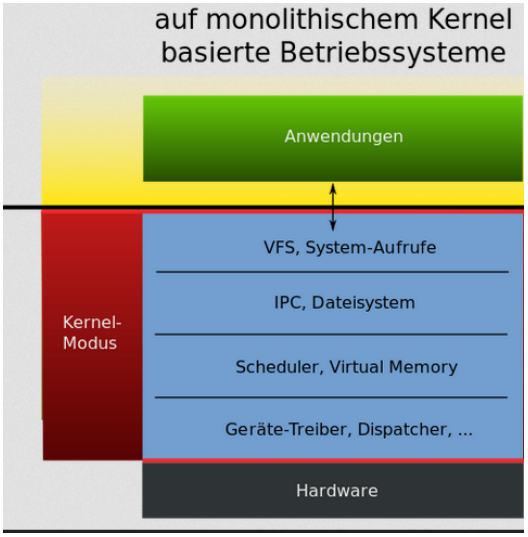


Abbildung 4:Monolithischer Kernel (Quelle: wikipedia)

3.6.2 Mikrokernel

Der Mikrokernel ist eine Weiterentwicklung des monolithischen Kernels. Dabei wird das Betriebssystem in Schichten aufgeteilt, wodurch es flexibler und überschaubarer wird. In der Regel sind nur Speicher und Prozessverwaltung, sowie Grundfunktionen der Kommunikation vorhanden. Alles Weitere wird über Programmbibliotheken implementiert.

Vorteil: einzelne Komponenten können einfach ausgetauscht werden. Die Trennung der Komponenten führt nicht zum Absturz des gesamten Systems. Gerätetreiber laufen im Benutzermodus, dadurch können Zugriffsrechte einzeln bestimmt werden.

Nachteil: Das Betriebssystem besteht aus vielen einzelnen Komponenten. Dadurch ist es langsamer als der monolithische Kern. Da der Zugriff auf die Hardware den privilegierten Modus benötigt, die Treiber aber im Benutzermodus laufen, muss ein eigenes Modul die Aufrufe umleiten.

In Abbildung 5 ist ein Bild eines Mikrokernel zu sehen. Der Mikrokern wird in verteilten Betriebssystemen eingesetzt.

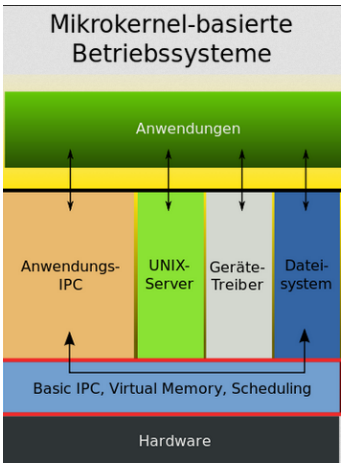


Abbildung 5: Mikrokernel (Quelle: wikipedia)

3.6.3 Hybrid-Kernel

Der Hybrid-Kernel ist ein Kompromiss aus den beiden vorherigen Varianten. Es wird versucht die Vorteile von Monolithischen- und Mikro-Kernel zu vereinen. Dabei ist nicht genau geregelt welche Komponenten bzw. Dienste im Kernel sein sollen oder müssen.

Typische Betriebssysteme sind Windows, Mac OS X und BeOS. In Abbildung 6 ist der Aufbau zu sehen.

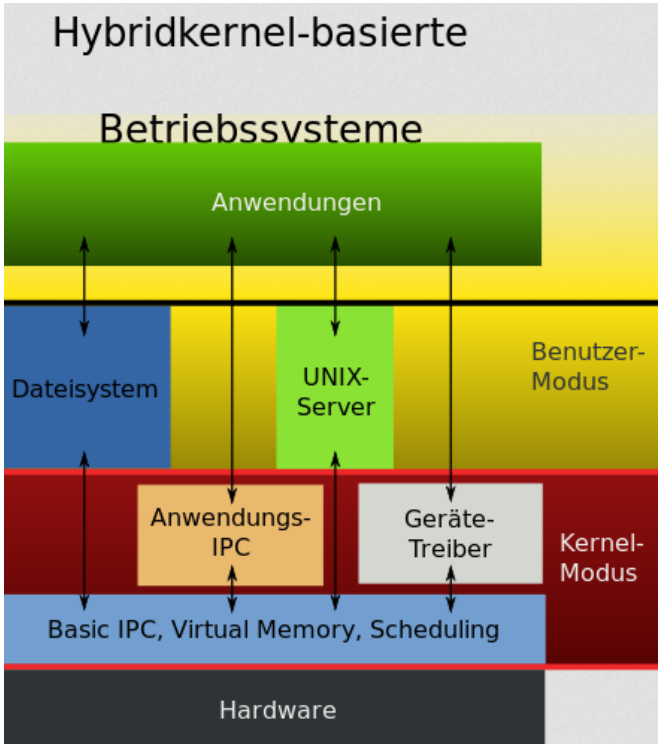


Abbildung 6: Hybridkernel (Quelle: wikipedia)

3.7 Arten von Betriebssystemen

Betriebssysteme (englisch: Operating System – OS) kann man in verschiedene Kategorien einteilen. Entweder nach Verwendungsart oder Betriebsart.

3.7.1 Mainframe-Betriebssysteme

- Betriebssysteme für Großrechenanlagen
- Verwaltung großer Datenmengen (viele Plattensysteme, oft viele GB / TB Speicherbereich)
- viele Prozesse werden gleichzeitig ausgeführt
- hoher Bedarf an schnellen Ein-/Ausgabegeräten
- drei Arten von Prozessverwaltung
- Batchbetrieb: Aufgaben ohne Benutzerinteraktion, Routineprozesse (Berichte, Logdateien, …)
- Transaktionsverarbeitung: große Anzahl kleiner Aufgaben mit vielen Benutzern (z.B.: Buchungssysteme, Shopsysteme, …)
- Zuteilungsverfahren (Timesharing): Quasi-parallele Durchführung vieler Aufgaben
- Betriebssystem ist von Hardwarehersteller abhängig.
- Einsatz: z.B.: als Webserver, Datenbankserver, …
- Beispiele sind MVS, OS/390, BS3000, …

3.7.2 Server Betriebssysteme

- Betriebssysteme, die auf Servern (leistungsfähige PCs, Workstations, Mainframes) laufen
- Sie bieten verschiedene "Dienstleistungen" im Netz an (z.B.: Druckdienste, Dateidienste, Webdienste, …)
- Die Dienste stehen vielen / allen Benutzern im Netz zur Verfügung.
- Einsatz: Firmennetzwerke, ISP, …
- Beispiele sind Windows Server, UNIX, LINUX, Novell NetWare, …

3.7.3 Multiprozessor-Betriebssysteme

- Verwaltung mehrerer parallel geschalteter Prozessoren innerhalb eines Rechners

- Meist abgeänderte Server-Betriebssysteme mit speziellen Eigenschaften für die Kommunikation und Anschlussfähigkeit
- Beispiele: Parallelcomputer, Multicomputer, Multiprozessor-Computer, …

3.7.4 Client-Betriebssysteme

- Ursprünglich zur Verwaltung von Einzelplatzsystemen (PC) konzipiert
- In Folge der technischen Weiterentwicklung um Netzwerkfunktionalitäten erweitert.
- Inzwischen keine klare Trennlinie zu Server-Betriebssystemen, sondern "fließender Übergang"
- Einsatz: Office, Programmierung, Netzwerkanwendungen, …
- Beispiele: WINDOWS, LINUX, Mac OS X, …

3.7.5 Echtzeit Betriebssysteme

- spezielle Betriebssysteme, bei denen die Zeit eine besonders wichtige Rolle bei der Verarbeitung spielt
- es muss eine definierte Antwortzeit geben – harte Zeitbedingung (Zeit zwischen Auftrag und Ergebnis)
- Beispiele: Steuerung von Maschinen, Ampelsteuerung, Robotersteuerung, …
- VxWorks, QNX, OSEK-OS

3.7.6 Eingebettete Systeme (embedded Systems)

- Computer die man nicht unmittelbar sieht
- Sehr oft Echtzeitanwendungen
- Wenig Ressourcenverbrauch
- Anwendungsbeispiele: Auto, PDA, Handy, Fernseher, Mikrowelle, …
- Beispiele: PalmOS, WINDOWS CE, embedded Linux, iOs, Android, Windows Phone

3.7.7 Chipkarten und Smardcards

- Manchmal nur für eine einzige Funktion entwickelt (z.B.: elektronisches Bezahlsystem, EC-Karte, E-Card)
- Anwendungsbeispiele: JavaCard, MULTOS

4 Aufgaben des Betriebssystems

Modern strukturierte Betriebssysteme kapseln den Zugriff auf die Betriebsmittel, dieser funktioniert nur über Betriebssystemfunktionen. Die wichtigsten Aufgaben werden nachstehend erläutert.

4.1 Prozessmanagement

Ein Prozess ist vom Grundprinzip her, ein Programm in Ausführung. Diesem Prozess können nach Bedarf verschiedene Komponenten (Betriebsmittel) des Rechensystems zugeordnet werden, um die ordnungsgemäße Programmausführung zu gewährleisten.

Jedem Prozess ist ein Adressraum zugeordnet (=Liste/Menge von Speicherstellen) den der Prozess bearbeiten (lesen und schreiben) darf. Der Adressraum enthält u.a.:

- ausführbares Programm
- Programmdaten
- Stack und Stack-Pointer
- Befehlszähler (program counter)
- Register
- Weitere Informationen, die zur Ausführung des Programms benötigt werden

Da die CPU immer nur einen Befehl zur gleichen Zeit ausführen kann, ist auf einem CPU-Kern immer nur ein Prozess aktiv – die anderen müssen warten. Die Entscheidung trifft das Betriebssystem, wann und wie lange ein Prozess eine Ressource benutzen darf. Wenn einem Prozess der Zugriff entzogen wird muss der Status gespeichert werden, damit danach wieder an derselben Stelle weitergearbeitet werden kann. Diese Verwaltung erfolgt in einer Prozesstabelle. Das Unterbrechen der Prozesse wird „Kontextwechsel“ bezeichnet.

4.1.1 Zustände

Ein Prozess kann mindestens drei verschiedene Zustände annehmen

Running der Prozess wird abgearbeitet

Ready der Prozess ist ausführbar, die CPU aber belegt

Waiting der Prozess kann nicht ausgeführt werden, weil er auf ein externes Ereignis wartet

Die Zustände bilden die Entscheidungsgrundlage für die Auswahl des Prozesswechsels. In Abbildung 7 ist ein typischer Prozesswechsel abgebildet.

1. Prozess wartet auf externes Ereignis
2. Scheduler wählt anderen Prozess aus
3. Gelb wird ausgewählt
4. Externes Ereignis ist eingetreten
5. Ein neuer Prozess wird erzeugt
6. Prozess wird terminiert

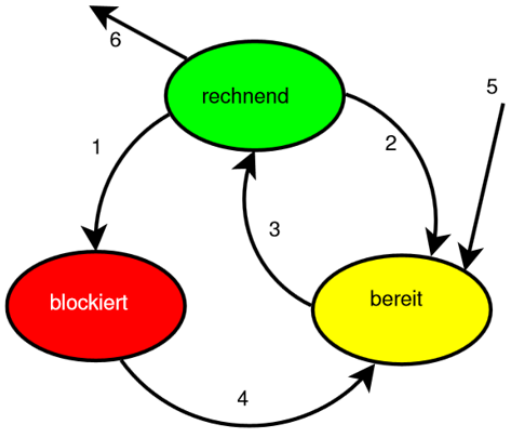


Abbildung 7:Prozesswechsel

Das Betriebssystem teilt dem Prozess (laufenden Programm) CPU-Zeit zu. Dieses Verfahren wird Scheduling genannt. Wenn nur ein Prozessorkern vorhanden ist laufen die Prozesse „quasi-parallel“. Wenn mehr als ein physikalischer Kern zur Verfügung steht laufen die Prozesse wirklich parallel ab.

Ein Sonderfall ist die Hyper-Threading Technologie. Es werden dem Betriebssystem zwei logische Kerne vorgetäuscht. Diese Prozessoren besitzen zusätzliche Pipelines und Registersätze um die Wartezeiten der CPU intelligent auszunutzen.

Jeder Prozess bekommt nach Priorität, Reihung und anderen Kriterien mehr oder weniger CPU-Zeit. Bei der Vergabe der CPU-Zeit werden nachstehende Ziele angestrebt:

- höchste Auslastung der CPU
- Fairness

- hohe Antwortzeit
- minimale Durchlaufzeit
- Durchsatz optimieren

Je nach Betriebssystem werden verschiedene Scheduling Konzepte verwendet:

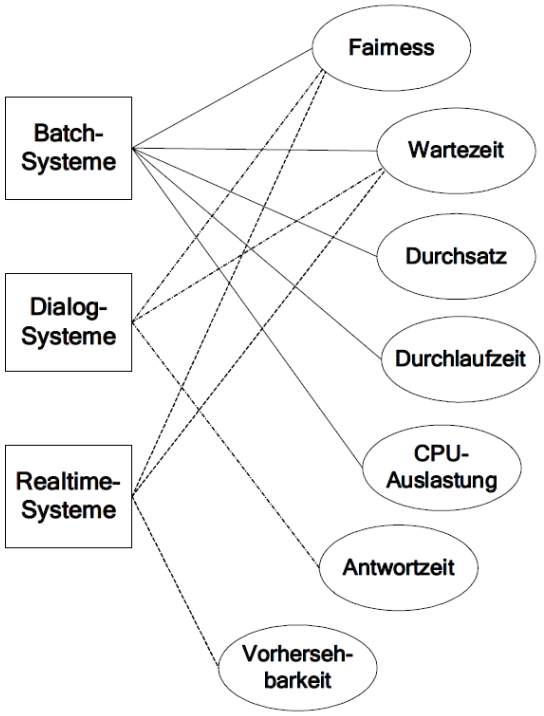


Abbildung 8: Konzepte von Betriebssystemen

Primär gibt es zwei verschiedene Scheduling-Verfahren:

präemptiv (verdrängend): Ein Prozess kann in seiner Ausführung unterbrochen werden. Dafür gibt es verschiedene Verfahren wann und wie Prozesse unterbrochen werden. Sehr oft wird eine Zeitscheibe verwendet. (Round Robin)

non-präemptiv: Ein Prozess wird nicht unterbrochen bis er seine Aufgabe ausgeführt hat.

4.1.2 Deadlock

Eine besondere Gefahr sind die sogenannten „Deadlocks (Verklemmung)“. Dabei blockieren sich Prozesse gegenseitig, weil sie auf Ereignisse von anderen warten. Das Betriebssystem muss Strategien für diese Fälle zur Verfügung stellen.

Eine Menge von Prozessen sperren sich gegenseitig, wenn jeder Prozess der Menge auf ein Ereignis wartet, das nur durch einen anderen Prozess der Menge ausgelöst werden kann.

In der realen Welt gibt es viele solcher Szenarien. Welche Beispiele für kennen Sie für einen Deadlock?

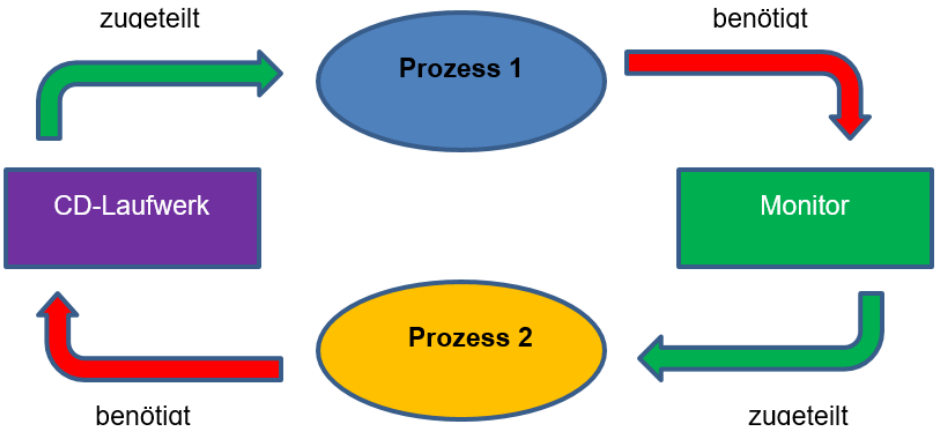


Abbildung 9: Deadlock

Prozess1 hat das CD-Laufwerk zugeteilt und benötigt den Monitor.

Prozess2 hat den Monitor zugeteilt und benötigt das CD-Laufwerk

Drei Strategien die Betriebssysteme integriert haben um Deadlocks zu vermeiden.

- Ignorieren des Problems (Vogel-Strauß-Algorithmus)
- Erkennen und beheben
- dynamische Verhinderung (vorsichtige Betriebsmittelzuteilung)

4.2 Speichermanagement

Moderne Betriebssysteme verwenden die „virtuelle Speichertechnologie“. Dabei wird der physikalische Speicher in einen logischen Adressbereich aufgeteilt. Der Grund für diese Vorgangsweise war folgende:

- Der Speicherbedarf eines Programmes sollte größer als der physikalische Hauptspeicher sein können
- Ein Programmierer sollte nur einen linearen Bereich des Speichers sehen

- Ein Prozess sollte auch ablaufen können, wenn er nur teilweise im Hauptspeicher geladen ist

Der Speicherbereich denn ein Prozess zur Verfügung hat wird virtueller Speicher oder virtueller Adressraum bezeichnet. Verwaltet wird er vom Memory-Manager. Das Betriebssystem muss für diesen Zweck mehrere Strategien implementieren.

- Abrufstrategie regelt den Zeitpunkt des Einlesens in den Hauptspeicher
- Speicherzuteilungsstrategie ermittelt den optimalen Platz
- Austauschstrategie entfernt Prozesse falls Speicher notwendig ist
- Aufräumstrategie entfernt unnötige Prozesse und Daten

Der Vorgang des Transports der Daten vom Hauptspeicher zur Festplatte wird **Paging** oder Ein-Auslagern genannt.

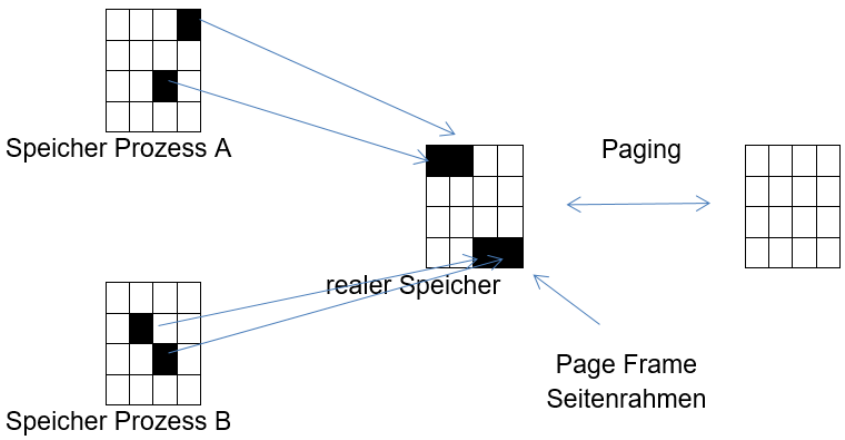


Abbildung 10:Paging Ein- und Auslagern

Der virtuelle Speicher wird in Blöcke aufgeteilt. Die einzelnen Blöcke werden als Seiten oder Pages bezeichnet. Die Blöcke des realen, physikalischen Speichers werden als Page Frame oder Seitenrahmen bezeichnet. Die einzelnen Seiten werden in einer Seitentabelle vom Memory Management Unit (MMU) verwaltet.

Die Standardgröße der Pages beträgt 4 KB. Bei dieser Größe kann ein 32 Bit Betriebssystem 2^{32} Bit Speicher verwalten. Der Memory-Manager ist in modernen Rechnersystemen in der CPU integriert. Starten Sie das Tool RAMMAP und untersuchen Sie die Speichersituation auf ihrem PC.

4.3 Ein- und Ausgabeverwaltung

Die Geräteverwaltung oder E-/A- Management dient zur optimalen Verwaltung von externen Geräten zur Ein- und Ausgabe. Kommandos zur Gerätesteuerung müssen an Geräte versendet werden. Die Anforderungen an das Betriebssystem sind die Verwaltung von Unterbrechungen und Fehlern die Geräte produzieren.

Ein Gerät besteht aus zwei Bestandteilen:

- Gerät: Hardware, mechanische und elektronische Einheit
- Controller: steuert direkt das Gerät an

Das Betriebssystem bietet zur Ansteuerung eine Schnittstelle an (Treiber). Zu den Geräten gehören Bildschirm, Tastatur, Netzwerkkarten, Festplatten, usw. Die Geräteverwaltung ist die Schnittstelle zwischen dem restlichen Betriebssystem und den physikalischen Geräten.

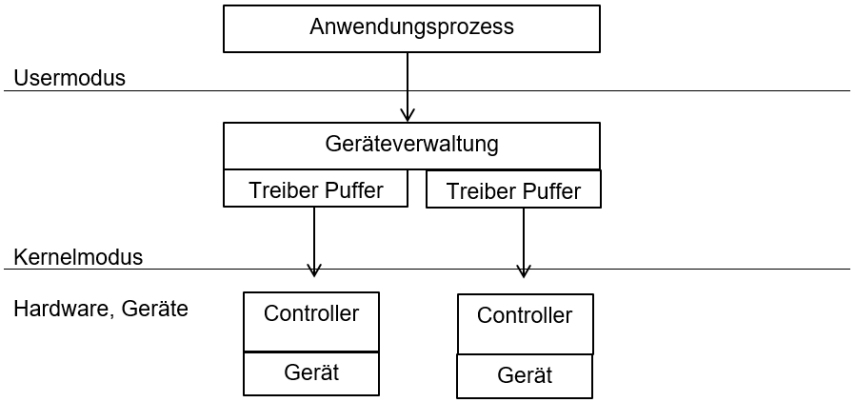


Abbildung 11:Schichten des Betriebssystems

Gerätetreiber sind somit gerätespezifische Softwarekomponenten. Gerätetreiber steuern den Datenaustausch. Wie in **Fehler! Verweisquelle konnte nicht gefunden werden.** zu sehen ist, w erden diese im Kernelmodus ausgeführt. Treiber haben folgende Aufgabe:

- macht das Gerät dem Betriebssystem bekannt
- dient zur Pufferung von Daten
- stellt ein logisches Programmiermodell bereit

Unter **UNIX/LINUX** werden Geräte logisch wie Dateien behandelt. Die Gerätetreiber binden sich als **Special-Files** ein. Eine Festplatte könnte mit dem Namen /dev/hd1 und ein Bildschirm

mit /dev/tty angesprochen werden. Der Vorteil dabei ist, dass die Geräte auch in Kommandos genutzt werden können.

```
z.B.: cp text /dev/lp
```

würde eine Datei mit dem Namen „text“ direkt an den Drucker senden.

4.4 Dateiverwaltung

Dauerhaft verwaltende Daten (persistente) werden üblicherweise in Dateien auf Massenspeichern (Festplatte, Flashspeicher) abgelegt. Die Dateiverwaltung kümmert sich um die Organisation der Dateien. Diese liegt im Schichtenmodell über der Geräteverwaltung.

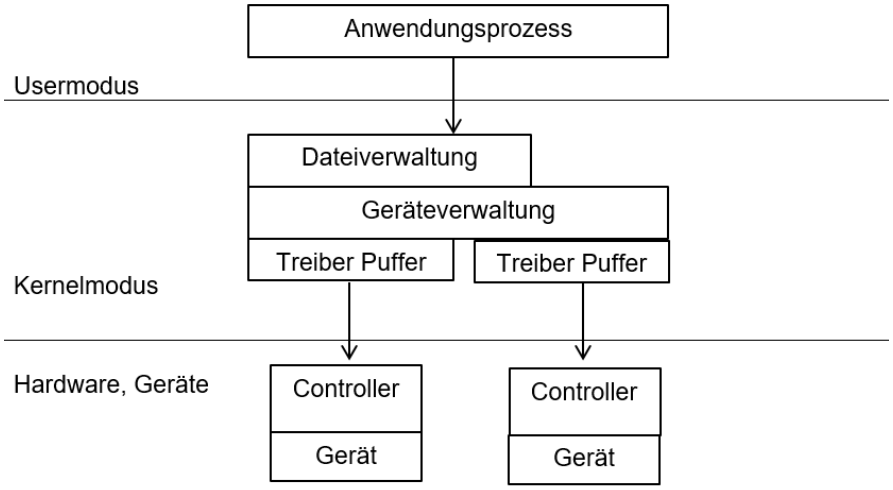


Abbildung 12:Dateiverwaltung

Dateien werden in Verzeichnissen zusammengefasst. Diese werden in Dateisystemen – das sind spezielle Datenstrukturen – verwaltet. Zu jedem Verzeichnis gibt es eine Verwaltungsinformation. Durch die hierarchische Anordnung entsteht ein umgedrehter Baum. Jede Datei wird mit Namen, Typ, Länge und Zugriffsrechte im Dateisystem abgespeichert. UNIX und Windows Systeme verwenden verschiedene Dateisysteme. Unter Windows wird FAT(32) und NTFS verwendet. FAT32 kann nur Dateien bis zu 4 GB verwalten, NTFS bis zu 17 TB. Unter UNIX/LINUX gibt es verschiedene Systeme. Ext3, ext4 und BTRFS sind zurzeit die aktuellsten und unterstützen alle Full-Journaling.

Ein weiterer Unterschied ist, dass es unter UNIX/LINUX ein systemweites Dateisystem gibt, während unter Windows für jedes Gerät ein eigenes, unabhängiges System verwaltet wird.

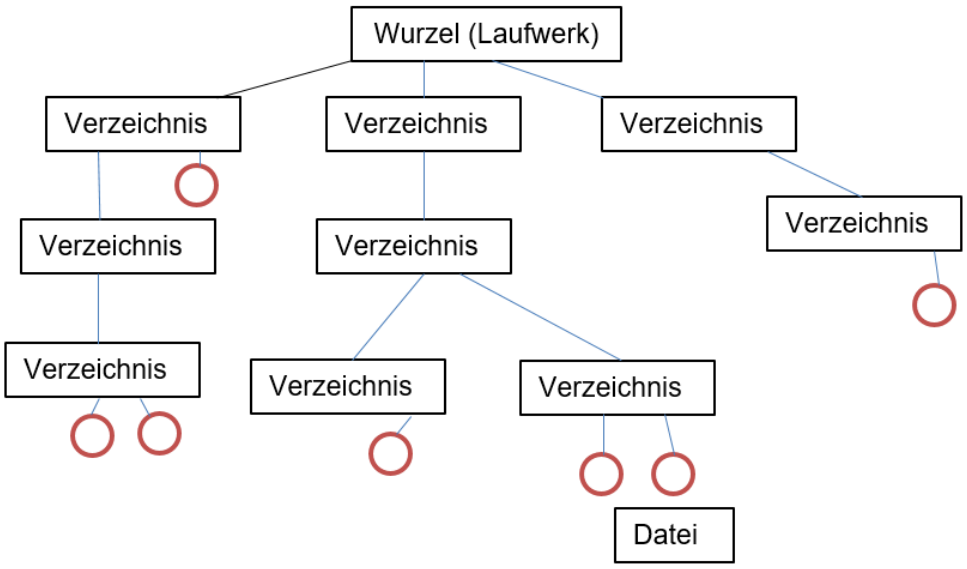


Abbildung 13: Verzeichnisbaum Windows

Für die Organisation eines UNIX Baumes gibt es Konventionen. Die Wurzel (root) wird mit einem Slash **/** abgekürzt. Im **usr** sind die Benutzerdaten und im **bin** werden die Programme gespeichert. Das **lib** Verzeichnis dient für die Libraries und im **etc** sind die Konfigurationsdateien gespeichert. Man muss sich nicht daranhalten, es hat aber einen Verwaltungsvorteil. In **Fehler! Verweisquelle konnte nicht gefunden werden.** abgebildet.

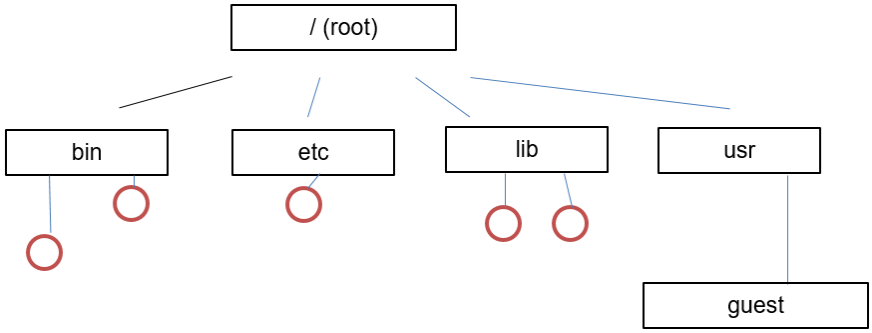


Abbildung 14:Unix Baum

4.5 Benutzeroberfläche

Die Benutzeroberfläche ist die Schnittstelle zwischen Menschen und Computer. Hier werden Daten und Befehle in den Computer eingegeben, die dann vom OS verarbeitet werden. Betriebssysteme stellen zwei verschiedene Arten von Benutzeroberflächen bereit.

4.5.1 Textbasierende Benutzeroberflächen

Sie waren die ersten Oberflächen und werden auch zeilen- oder befehlsorientierte Oberflächen genannt. Sie nicht sehr komfortabel zu bedienen. Man benötigt eine gewisse Einarbeitungszeit um die Befehle zu lernen. Typische Betriebssysteme sind Linux- oder Unix Server, DOS.

4.5.2 Grafisch-orientierte Benutzeroberflächen

Bei diesen Benutzerschnittstellen ist die Eingabe von Befehlen nicht zwingend erforderlich. Die Eingabe kann auch mittels Maus oder Touchscreen erfolgen. Sehr viele Betriebssysteme besitzen derzeit eine grafische Oberfläche. Windows, Linux oder Mac OS X.

4.6 Betriebsmittel

Eine wesentliche Aufgabe des Betriebssystems ist die Betriebsmittelverwaltung. Das ermöglicht eine koordinierte gemeinsame Nutzung der Hardware. Betriebsmitteln werden in Hardware und Software eingeteilt. Den Anwendungsprogrammen werden statt realen Ressourcen nur virtuelle bereitgestellt.

Hardware-Betriebsmittel: Massenspeicher, Arbeitsspeicher, Prozessor, Netzwerkkarten, ……

Software-Betriebsmittel: Dateien, Compiler, Dienstprogramme, Benutzerprogramme, …

Beispiel für den Übergang von realen Betriebsmitteln zu virtuellen Betriebsmitteln.

real	virtuell
Prozessor	Prozess
Arbeitsspeicher	virtueller Adressraum

4.7 Rechteverwaltung

Eine sehr wichtige Aufgabe ist die Rechteverwaltung. Das Betriebssystem muss die Sicherheit der Ressourcen gewährleisten. Das Ziel ist es, einem Benutzer nur die Betriebsmittel zur Verfügung stellen die benötigt werden und den Zugriff auf Dateien einschränken. Damit das Betriebssystem das bewerkstelligen kann, benötigt es vom Benutzer eine Identifikation, sowie die dazugehörige Authentifizierung. Rechtesysteme sind sehr oft mit Policies verankert.

Rechte können auf Benutzer, Ressourcen oder Operationen vergeben werden. Damit das Betriebssystem Benutzer verwalten kann, benötigt es die Identifikation und Authentifizierung des Benutzers:

Identifikation: eindeutiger Bezug zwischen System und Benutzer (ID, Name, ..) herstellen. Beweis der Identität

Authentifizierung: Überprüfung ob der Benutzer tatsächlich der ist, für den er sich ausgibt. (Passwort, Fingerabdruck, ..)

Beispiele für Rechte bei Betriebssystemen:

- Meta-Rechte
  - Rechte vergeben, ändern oder entziehen
- Anwendungsebene
  - lesen, schreiben ausführen
- Prozesskontrolle
  - Prozesse anzeigen, anhalten, terminieren
- Zugriff auf Ressourcen
  - CPU, Arbeitsspeicher, Netzwerk

5 Literaturverzeichnis

Brause, R. (2017). *Betriebssysteme: Grundlagen und Konzepte.* Frankfurt: Springer-Verlag.

Golftheman. (2015). *OS-Strukture.* Abgerufen am 02. 05 2018 von v

Kelling, C. (2016). *Von-Neumann-Rechner.* Abgerufen am 30. 04 2019 von http://tams-www.informatik.uni-hamburg.de/applets/baukasten/DA/VNR_Einleitung.html

Mandl, P. (2008). *grundkurs Betriebssysteme.* vieweg.de.

Schütte, A. (2014). *Prozesse.* Abgerufen am 02. 05 2019 von http://www.fbi.h-da.de/~a.schuette/Vorlesungen/Betriebssysteme/Skript/3_Prozesse/Prozesse.pdf

Tannenbaum, A. S. (2002). *Moderne Betriebssysteme.* Deutschland: Pearson.