Betriebssysteme

Eine allgemeine Einführung

erstellt durch:

Name:	Karl Wohlrab
Telefon:	09281 / 409-279
Fax:	09281 / 409-55279
EMail:	mailto: Karl.Wohlrab@fhvr-aiv.de

Inhaltsverzeichnis

1. Allgemeine Grundlagen

1.1 Vorbemerkungen / Allgemeines

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 auskommen, wird spätestens seit der Umsetzung der Von-Neumannschen Prinzipien (siehe später) und der Entwicklung so genannter "Universalrechner" eine neutrale Instanz zum Betrieb der Rechenmaschine benötigt. Diese Aufgabe übernimmt das Betriebssystem.

Waren früher (zur Zeit der "Alleinherrschaft der Großrechner") die Betriebssysteme nur wenigen DV-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. Der versierte Anwender (zumindest diejenigen, die als DV-Betreuer tätig sind) sollten hier auch die Kompetenz haben, die Vor- und Nachteile von unterschiedlichen Betriebssystemen beurteilen zu können.

Aufgrund der offenen und klaren Struktur baut die vorliegende Veranstaltung weitgehend auf dem Betriebssystem LINUX auf, ohne jedoch auf exemplarische Verweise auf andere Betriebssysteme (insbesondere WINDOWS-Derivate) zu verzichten.

1.2 Aufgaben eines Betriebssystems

1.2.1 Was ist ein Betriebssystem?

Literaturhinweise:

[Tann02] Seite 13 ff.

Ein Rechner besteht zunächst aus den sichtbaren (und greifbaren) Komponenten – der Hardware.

Damit aber mit den Rechner auch gearbeitet werden kann, benötigt er auch noch geeignete Programme. Nach den Von-Neumannschen Prinzipien ist ein Rechner als "Universalrechner" konstruiert; die Erledigung unterschiedlicher Aufgaben wird ausschließlich über Programme – der Software gesteuert.

Da aber nicht jeder Softwareentwickler sich um alle Details zur Verwaltung der einzelnen Hardwarekomponenten (z.B.: Steuerungsinformationen für Plattenspeicher, etc.) kümmern sollte (Anm. Dies wäre eine zu große Verschwendung der Arbeitszeit), erscheint es sinnvoll, diese Komponenten zentral und damit allgemein verfügbar bereitzuhalten.

Damit erreicht man dann, dass die Verwaltungsprogramme nur einmal erstellt werden müssen und dann allen Anwendern / Anwenderprogramme zur Verfügung gestellt werden,

1.2.1.1 Mögliche Definitionen für „Betriebssystem“

DIN 44300:
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.

DUDEN:
Zusammenfassende Bezeichnung für alle Programme, die die Ausführung der Benutzerprogramme, die Verteilung der Betriebsmittel auf die einzelnen Benutzerprogramme und die Aufrechterhaltung der Betriebsart steuern und überwachen.

Rembold, Einführung in die Informatik :
Der Zweck eines Betriebssystems liegt darin, Fähigkeiten zur Verfügung zu stellen, um eine Rechenanlage möglichst durch mehrere Anwender nutzen zu können. Diese Nutzung soll einfach, zuverlässig und wirtschaftlich sein.

Abbildung 1: Was ist ein Betriebssystem? (1)

Bic/Shaw, Betriebssysteme. Eine moderne Einführung:
Betriebssysteme haben zwei Hauptaufgaben: Sie stellen Dienste bereit, die die Aufgaben der Benutzer vereinfachen, und sie verwalten Betriebsmittel um einen wirkungsvollen Rechnerbetrieb sicherzustellen.
Aus Benutzersicht erscheint das Betriebssystem als eine virtuelle Maschine mit der Menge ihrer Kommandos als Maschinensprache.

Tannenbaum, Moderne Betriebssysteme:
Aufgabe eines Betriebssystems ist es, Geräte zu verwalten und Benutzerprogrammen eine einfache Schnittstelle zur Hardware zur Verfügung zu stellen.

Abbildung 2: Was ist ein Betriebssystem? (2)

Ein Betriebssystem ist quasi der Mittler (Schnittstelle) zwischen Benutzer und Hardware. Dadurch wird die Komplexität der darunter liegenden Systemarchitektur verborgen und dem Anwender eine verständliche und handhabbare Schnittstelle angeboten. So braucht sich beispielsweise ein Anwendungsentwickler nicht um technische Details und die Steueranweisungen der jeweils vorhandenen Hardwarekomponenten zu kümmern, sondern kann sich auf die Implementierung seiner Funktion konzentrieren.

Weiterhin verwaltet das Betriebssystem die Ressourcen des Rechners. Das Betriebssystem verwaltet alle physikalischen Geräte, wie Prozessoren, Speicher, Platten, Terminals usw., des Gesamtsystems. Damit ist insbesondere eine sinnvolle Zuteilung der Ressourcen an die darum konkurrierenden Programme gemeint. Ohne eine solche Steuerung durch das Betriebssystem wäre eine vernünftige Benutzung heutiger Computer nicht möglich.

Eine genaue Definition "Was ist ein Betriebssystem?" ist nicht umfassend möglich, da auch in der Fachwelt hier unterschiedliche Meinungen und Sichtweisen existieren. Die Hauptsichtweisen sollen im Folgenden kurz vorgestellt werden.

Nähere Informationen werden hierzu im folgenden Kapitel noch genauer beschrieben.

Literaturhinweise:

[Muck02] Seite 17

1.2.2 Die Von-Neumann-Prinzipien

Im Jahre 1946 wurde von John von Neumann das folgende Konzept vorgeschlagen, um einen universellen Rechner, der technischen, wissenschaftlichen und kommerziellen Anforderungen gerecht wird zu gestalten. Von einigen wenigen Ausnahmefällen ... abgesehen, orientieren sich die heutigen Rechenanlagen an der Struktur dieses klassischen Universalrechners. Der Rechner ist nach folgenden Prinzipien ... aufgebaut:"

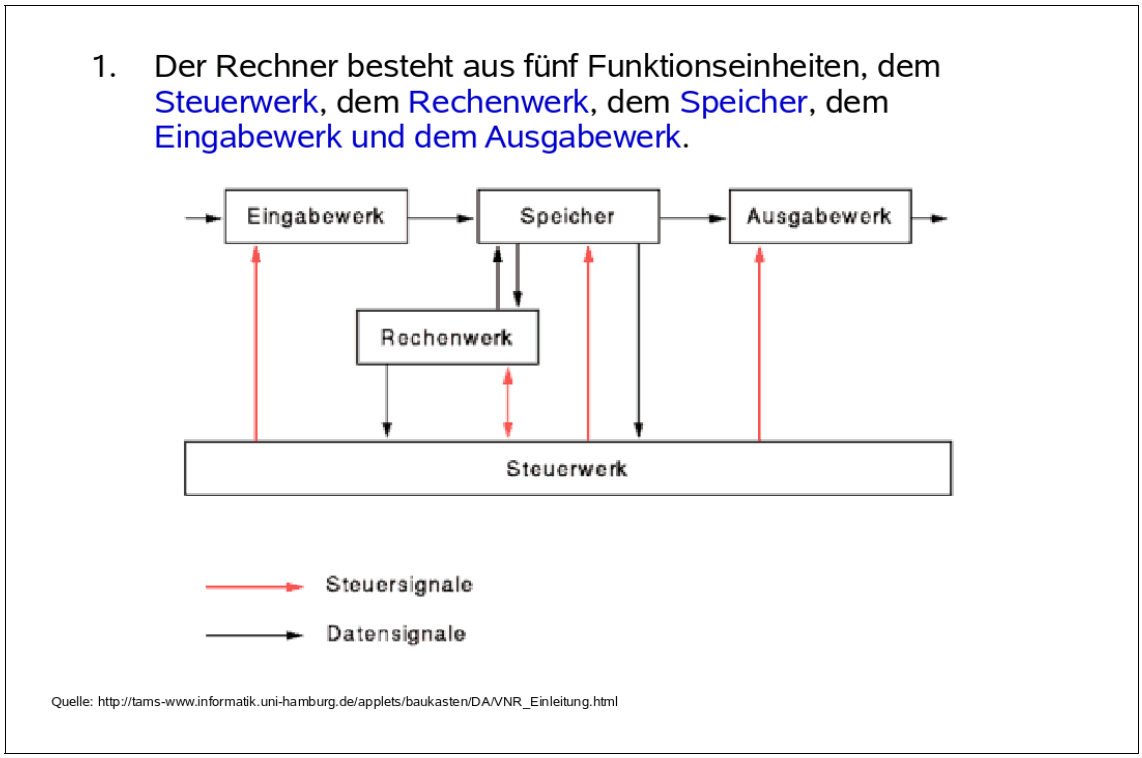

Abbildung 3: Von-Neumann-Prinzipien (1)

2. Die Struktur (Aufbau) des Von-Neumann-Rechners ist unabhängig von den zu bearbeitenden Problemen. Zur Lösung eines Problems muss von außen eine Bearbeitungsvorschrift, das Programm, eingegeben und im Speicher abgelegt werden. Ohne dieses Programm ist die Maschine nicht arbeitsfähig.

3. Programme, Daten, Zwischen- und Endergebnisse werden in demselben Speicher abgelegt.

4. Der Speicher ist in gleichgroße Zellen unterteilt, die fortlaufend durchnumeriert sind. Über die Nummer (=Adresse) einer Speicherzelle kann deren Inhalt abgerufen oder verändert werden.

   RAM = Random Access Memory
   (= Speicher mit wahlfreien Zugriff)

Abbildung 4: Von-Neumann-Prinzipien (2-4)

5. Aufeinanderfolgende Befehle eines Programms werden in aufeinanderfolgenden Speicherzellen abgelegt. Das Ansprechen des nächsten Befehls geschieht vom Steuerwerk aus durch Erhöhen der Befehlsadresse um Eins.

6. Durch Sprungbefehle kann von der Bearbeitung der Befehle in der gespeicherten Reihenfolge abgewichen werden.

Abbildung 5: Von-Neumann-Prinzipien (5-6)

7. Es gibt zumindest
   - arithmetische Befehle wie Addieren, Multiplizieren, Konstanten laden usw.;
   - logische Befehle wie Vergleiche, logisches NICHT, UND, ODER usw.;
   - Transportbefehle, z.B. vom Speicher zum Rechenwerk und für die Ein-/Ausgabe;
   - bedingte Sprünge;
   - sonstige Befehle wie Schieben, Unterbrechen, Warten usw.

   Alle diese Befehle können in verschiedenen Adressierungsarten ausgeführt werden,

8. Alle Daten (Befehle, Adressen usw.) werden binär codiert. Geeignete Schaltwerke im Steuerwerk und an anderen Stellen sorgen für die richtige Entschlüsselung (Decodierung).

Abbildung 6: Von-Neumann-Prinzipien (7-8)

1.2.3 Das Schichtenmodell eines Computersystems

Literaturhinweise: [Muck02] Seite 17

Die zunehmende Komplexität und Vielfalt der eingesetzten Hardware machte es erforderlich, den Programmierern Hilfsmittel an die Hand zugeben, mit deren Hilfe er seine Anwendungen leichter entwickeln kann, ohne dabei spezifische Detailkenntnisse über die jeweils eingesetzten Komponenten zu benötigen. Auch die Erfordernis zur Übernahme bestehender Anwendungen auf andere Hardwarekonfigurationen (Portierbarkeit) erfordert eine weitestgehend hardwareunabhängige Entwicklung.

Als Lösungsmöglichkeit hat man deshalb um die reine Hardware eine "Softwareschicht" (je nach Sichtweise auch mehrere Schichten) gelegt, die dem Entwickler bzw. Anwender die Hardware-Details verbirgt und ihm über definierte Schnittstellen den Zugriff auf die Hardware trotzdem ermöglicht. Aus der zunächst nur einen Softwareschicht sind zwischenzeitlich mehrere Schichten (oder Schalen) geworden. Einige Sichtweisen werden im Folgenden dargestellt:

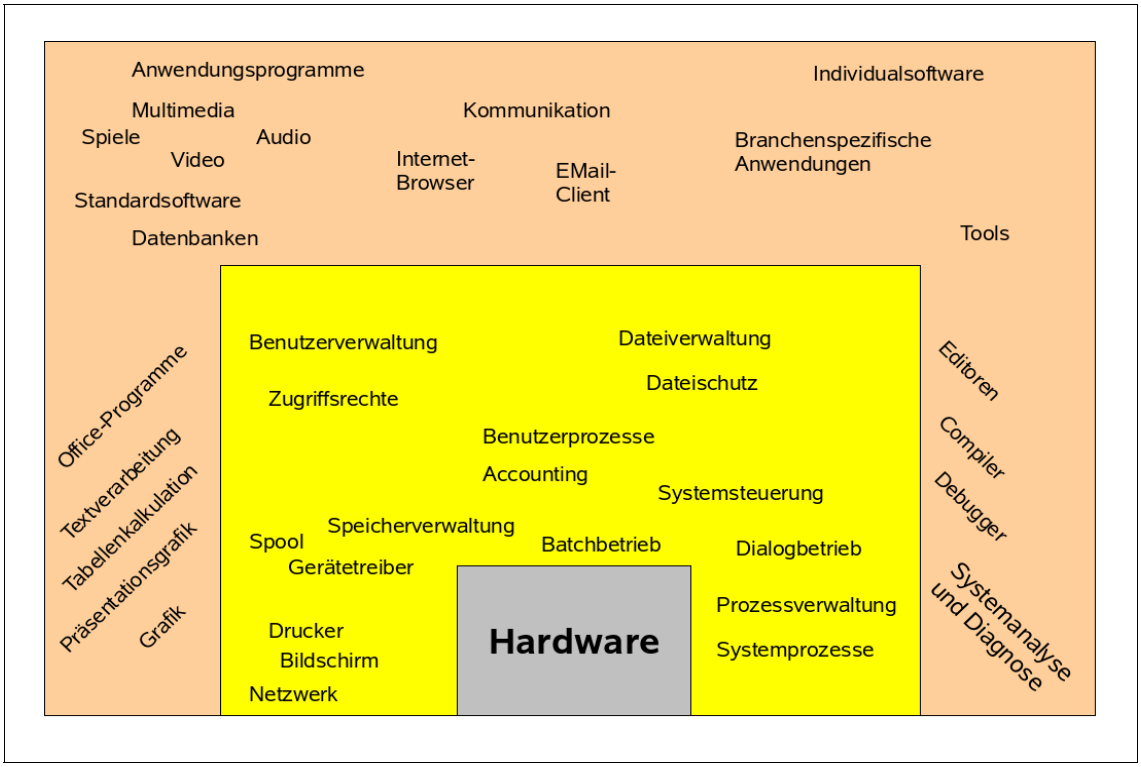

Abbildung 7: Aufbau eines Betriebssystems (Anwendersicht)

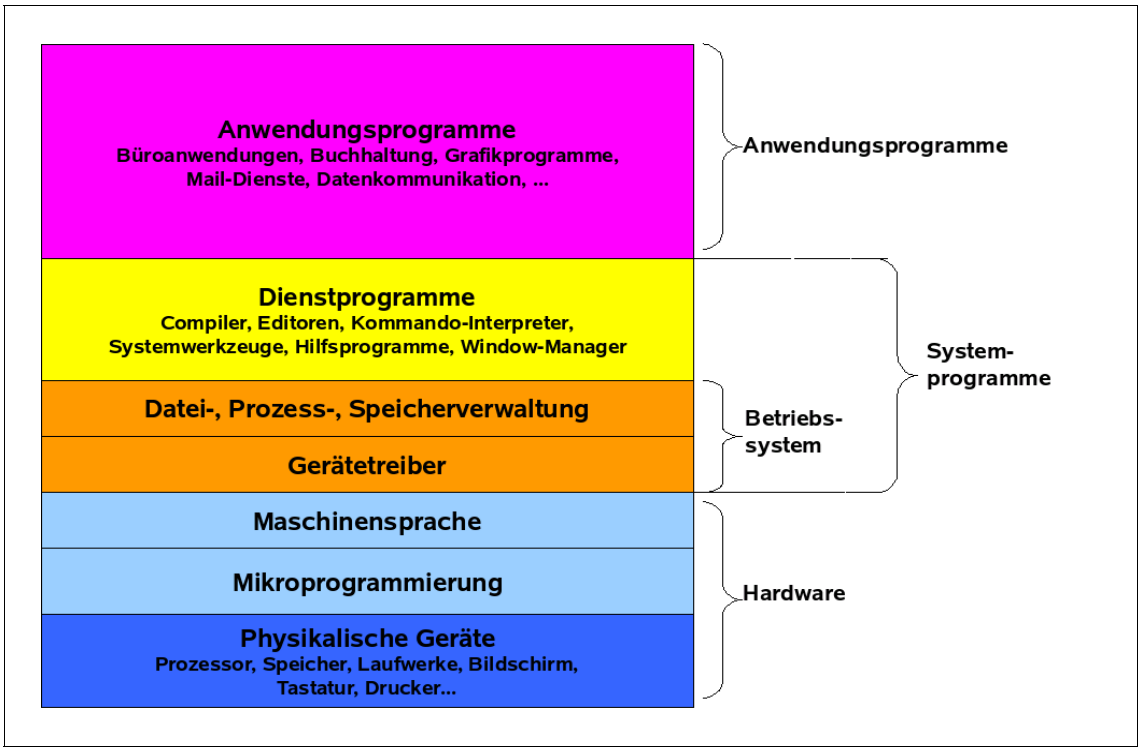

Abbildung 8: Aufbau eines Betriebssystems (IT-Sicht)

1.2.3.1 Physikalische Geräte

Die unterste Schicht enthält die physikalischen Geräte. Sie besteht aus integrierten Schaltungen, Drähten, Stromversorgung, etc.
Komponenten dieser Ebene sind beispielsweise

Prozessor,

Speicherchips des Arbeitsspeichers (Kernspeicher)

Bussystem(e)

Plattenspeicher

Netzwerkschnittstellen

usw.

1.2.3.2 Mikrocode

Aufgabe des Mikrocodes ist die direkte Steuerung der physikalischen Geräte. In Mikrocode entworfene Programme werden von einem Interpreter in entsprechende Steueranweisungen übersetzt und an die Geräte durch gereicht. Die Instruktionen der Maschinensprache (z.B.: ADD, MOVE, JUMP, ...) werden als Folge einzelner kleiner Schritte ausgeführt (siehe Befehlszyklus im Prozessormodell)

Die Mikroprogramme werden normalerweise in ROMs abgelegt; in früheren Rechnern wurden Mikroprogramme häufig in "Wire-Wrap"-Technik realisiert. Mikroprogramme werden deshalb der Hardware zugerechnet.

1.2.3.3 Maschinensprache

Die Menge von Instruktionen die das Mikroprogramm ausführen kann wird als Maschinensprache bezeichnet.

Die Maschinensprache besteht aus zahlreichen elementaren Maschinenbefehlen (z.B. zur Berechnung von arithmetischen Ausdrücken, zum Vergleich von Werten), die in Form von Mikrocode-Anweisungen ausgeführt werden. Heutige Maschinensprachen verfügen über ca. 50-300 unterschiedliche Instruktionen.

Da die Maschinensprache hardwarespezifisch ist wird sie noch der Hardware zugeordnet.

1.2.3.4 Betriebssystem

Eine der Hauptaufgaben des Betriebssystems ist es, die Komplexität der Hardware und deren Verwaltung zu verstecken und dem Programmierer eine angemessene Menge an Instruktionen zur Verfügung zu stellen, mit denen er effizienter arbeiten kann.

Die Aufgaben eines Betriebssystems, bestehen im Wesentlichen aus der Implementierung von Schnittstellen (meist in Form von Gerätetreibern) und die Verwaltung der Ressourcen (z.B.: Speicher, Prozessornutzung, etc.). Das vorliegende Scriptum befasst sich weitgehend mit dem Betriebssystem.

1.2.3.5 Systemprogramme / Dienstprogramme

Hierbei handelt es sich um Programme, die weder zum Betriebssystem (obwohl sie meistens mit vom Betriebssystemhersteller ausgeliefert werden) noch zu den Anwenderprogrammen gehören. Von ersterem unterscheiden sie sich durch ihre Austauschbarkeit, von letzterem durch ihre Systemnähe. Beispiele sind Editoren, Kommando-Interpreter, Compiler, Linker, Überwachungsprogramme, ...)

Das vorliegende Scriptum befasst sich weitgehend mit dem Betriebssystem und ausgewählten Dienstprogrammen.

1.2.3.6 Anwenderprogramme

Diese oberste Schicht stellt den Benutzern des Systems Anwendungssoftware zur Verfügung. Die Anwendungsprogramme setzen entweder auf den System- oder Dienstprogramme bzw. unmittelbar auf dem Betriebssystem auf. Als Beispiel könnte man einen Web-Browser, eine Datenbank- oder Office-Anwendung nennen.

1.2.4 Das Betriebssystem als erweiterte Maschine

Literaturhinweise: [Tann02] Seite 15 ff.

Die Architektur (Befehlssatz, I/O-Organisation, Speicherorganisation, Busstruktur, ...) der meisten Rechner ist auf der Maschinenebene sehr einfach (d.h. mit wenig Bedienkomfort) ausgestattet. Die Nutzung der Komponenten ist (zumindest auf dieser Ebene) für die Anwender sehr komplex.

Wegen der relativ einfachen Struktur der einzelnen Anweisungen ist die Programmierung auf Maschinenebene zwar meist sehr effizient, aber auch für geübte Programmierer sehr unkomfortabel. Selbst DV-Spezialisten versuchen deshalb, die Arbeit auf der Maschinenebene, soweit irgend möglich zu vermeiden.

1.2.4.1 Praxisbeispiel Floppy-I/O

Siehe [Tann02] Seite 15 ff.

- Ein Floppy-Controller (PD765) hat einen Befehlssatz von 16 Befehlen mit denen je nach Aufgabe zwischen 1 bis 9 Bytes in ein Geräteregister übertragen werden.
  Die Befehle verrichten folgende Aufgaben:
  - Lesen und Schreiben von Daten
  - Bewegung des Plattenarms
  - Formatierung der Spuren
  - Initialisierung
  - Statusabfrage
  - Zurücksetzen und Neukalibrierung des Controllers und der Geräte
- Jeder der Befehle hat mehrere Parameter (z.B. READ oder WRITE haben je 13 Parameter (in 9 Bytes gepackt)) Die Parameter enthalten z.B.: folgende Angaben:
  - Adresse des zu lesenden Plattenblocks
  - Anzahl der Sektoren pro Spur
  - Verwendeten Aufzeichnungsmodus des physikalischen Mediums
  - Lückengröße zwischen den Sektoren
  - Verarbeitung der "delete-data-adress"-Marke
- Wenn die Verarbeitung abgeschlossen ist, liefert der Controller 23 Status- und Fehlerfelder (in 7 Bytes gepackt) zurück.

- Des Weiteren muss vor dem Zugriff auf die Floppy-Disk geprüft werden, ob der Motor eingeschaltet ist. Falls dieser erst noch eingeschaltet werden muss, ist noch eine (nicht unerhebliche) Zeitspanne als Anlaufverzögerung beachtet werden, bevor Daten gelesen bzw. geschrieben werden können.
- Im Umkehrschluss sollte der Motor nicht übermäßig lange laufen, um einen vorzeitigen Verschleiß oder Beschädigung der Floppy-Disk zu vermeiden; d.h. bei längeren Nichtgebrauch sollte der Motor auch wieder abgeschaltet werden.

Es ist eigentlich offensichtlich, dass es für die Entwicklung von Anwendungsprogrammen unsinnig ist, jedes mal die Zugriffe erneut zu programmieren. Zudem könnte das Anwendungsprogramm dann nur mit Floppy-Laufwerken arbeiten, die genau diesen spezifischen Floppy-Controller verwenden; für alle anderen Controller-Typen wäre das Programm nicht geeignet.

Ein Programm, das die realen Eigenschaften der Hardware vor dem Anwender / Programmierer verbirgt und ihm stattdessen einen vergleichsweise einfachen und komfortablen Zugang zu den einzelnen Komponenten ermöglicht nennt man Betriebssystem. Die einzelnen Komponenten werden dem Programmierer bzw. Anwender gegenüber als Dateien repräsentiert; die im Umgang eher unangenehmen Aufgaben (z.B.: Unterbrechung; konkurrierende Zugriffe, Speicherverwaltung, ...) bleiben verborgen.

Die Aufgabe eines Betriebssystems ist es, dem Benutzer ein Äquivalent einer erweiterten Maschine, bzw. virtuelle Maschine zu präsentieren, die leichter zu programmieren ist, als die darunter liegende Hardware.

1.2.5 Das Betriebssystem als Betriebsmittelverwalter

Literaturhinweise: [Tann02] Kap. 1.1.2, Seite16 ff

Die Verwendung teurer Einzelkomponenten erfordert eine Strategie zur Mehrfachnutzung durch mehrere Anwender / Programme. So macht es z.B.: keinen Sinn, ein teureres Magnetbandlaufwerk exklusiv für einen einzelnen Anwender bereitzustellen: vielmehr sollte das teuere Gerät von mehreren Anwendern genutzt werden können – und dies nach Möglichkeit so, dass keiner auf den Anderen Rücksicht nehmen muss.

Das Betriebssystem hat die Aufgabe, alle Betriebsmittel des Rechners zu verwalten.

Heutige Rechner bestehen aus einer Vielzahl von Einzelkomponenten (Prozessoren, Speicher, Festplatten, Netzwerkkarten, ...), die sowohl im Gehäuse integriert oder als externe Geräte angeschlossen werden können. Die Aufgabe des Betriebssystem besteht im Grunde darin, den Zugriff auf die einzelnen Komponenten zu regeln (kontrollierte Allokation der Komponenten).

Insbesondere in dem mittlerweile üblichen Fall, dass mehrere Programme gleichzeitig auf einem Rechner abgearbeitet werden und ggf. auch gleichzeitig auf die gleichen Komponenten zugreifen möchten, erfordert eine konsequente Verwaltung der einzelnen Komponenten und klar definierte Vergabestrategien. Auch der Schutz vor unbefugtem Zugriff durch andere Programme / Benutzer ist dabei von großer Bedeutung.

1.3 Historische Entwicklung von Betriebssystemen

Literaturhinweise: [Tann02] Kap. 1.2, Seite18 ff

Den aktuellen Stand der Betriebssystemtechnik muss man zum besseren Verständnis einiger Entwicklungen auch unter dem Gesichtspunkt der historischen Entwicklung betrachten. Die Konzepte wurden über Jahre (z.T. Jahrzehnte) entwickelt und spiegeln häufig den, zum Zeitpunkt der Entwicklung geltenden Stand der Hard- und Softwaretechnik wieder.

1. Generation (Röhrenrechner und Steckkarten)

2. Generation (Transistoren und Stapelverarbeitung)

3. Generation (IC's und Mehrprogrammbetrieb)

4. Generation (Personal-Computer, Netzwerke und Client-Server-Konzepte)

1.3.1 Die erste Rechnergeneration (1945-1955): Röhren und Steckkarten

Die ersten halbwegs funktionsfähigen elektronischen Rechensysteme entstanden etwa Mitte der 40er Jahre des 20. Jahrhunderts. Zu den Computer-Pionieren gehörten unter anderem auch Howard Aiken, John von Neumann, Alan Turing und Konrad Zuse.

Die ersten Rechner bestanden aus Tausenden von Elektronenröhren, füllten teilweise mehrere Räume, verbrauchten viel Strom und produzierten (durch die vielen Röhren) eine Menge an Abwärme. Dementsprechend hoch waren die Ausfallzeiten, da durch die Wärme die Röhren häufig kaputt gingen.

Die Programmierung (soweit man hier überhaupt von Programmierung sprechen kann) erfolgte über die Verdrahtung von Steckkarten und / oder in absoluter Maschinensprache. Die Assemblersprache war noch nicht erfunden. Eigentlich war die Programmierung mehr ein individueller Umbau des Rechners für die jeweils gewünschte Berechnung. Die Einführung von Lochkarten oder Lochstreifen ermöglichte es Anfang der 50er Jahre, die Programme dauerhaft zu speichern und bei Bedarf wieder einzulesen. Zumindest die sehr zeitaufwendige und fehlerträchtige manuelle Eingabe eines Programms in Maschinensprache (Binärcode) konnte damit erleichtert werden; alle anderen Probleme bleiben aber bestehen.

Hier bestand noch keine Erfordernis für ein Betriebssystem, da einerseits jedes Programm sowieso individuell für die Rechenmaschine erstellt wurde und andererseits immer nur ein Programmierer an Rechner arbeiten konnte – ein konkurrierender Zugriff auf Systemkomponenten fand also ebenfalls nicht statt

1.3.2 Die zweite Generation (1955-1965): Transistoren und Stapelverarbeitung

Die Erfindung und Einführung von Transistoren (ab ca. 1955 eingesetzt) konnte auf die fehleranfälligen Röhren als Schaltelemente verzichtet werden. Die Rechner wurden kleiner bzw. bei gleichem Raumbedarf leistungsfähiger, vielfältiger einsetzbar und auch zuverlässiger, da die Ausfälle wesentlich seltener wurden. Ein elektrotechnischer Umbau des Rechners war nicht mehr erforderlich; die Anpassung an die jeweils zu lösenden Probleme erfolgte ausschließlich über Programme (siehe von-Neumann'sche Prinzipien). Jetzt konnten auch die Rechner erstmals an Kunden verkauft werden.

Ab diesem Zeitpunkt musste auch zwischen unterschiedlichen Personengruppen mit unterschiedlichen Aufgaben und Interessen unterschieden werden:

Entwickler / Hersteller des Rechners,

Wartungspersonal zur Störungsbehebung und Reparatur,

Operateure zur Aufrechterhaltung des Rechenbetriebs,

Programmierer zur Erstellung von Anwendungsprogrammen und

Anwender / Auftraggeber – die eigentlichen Nutzer des Rechners.

Aufgrund der trotzdem noch hohen Wärmeentwicklung waren die Rechner in speziellen klimatisierten Räumen untergebracht; lediglich die Operatoren hatten Zugang zum System.

Programme wurden von den Programmierern (jetzt in Assemblersprache oder evtl. auch schon in FORTRAN) zunächst auf Papier entwickelt und dann auf Lochkarten gestanzt. Um ein Programm ablaufen zu lassen, musste es in einen Job aufgenommen werden, d.h. der Lochkartenstapel mit dem Programm wurde um so genannte Steuerkarten ergänzt. Auf den Steuerkarten wurden z.B.: angegeben wer die Kosten für den Programmlauf übernimmt, von welchen Geräten die Eingabedaten gelesen werden sollen oder auf welche Geräte die Ergebnisse ausgegeben werden sollen (z.B.: Drucker).

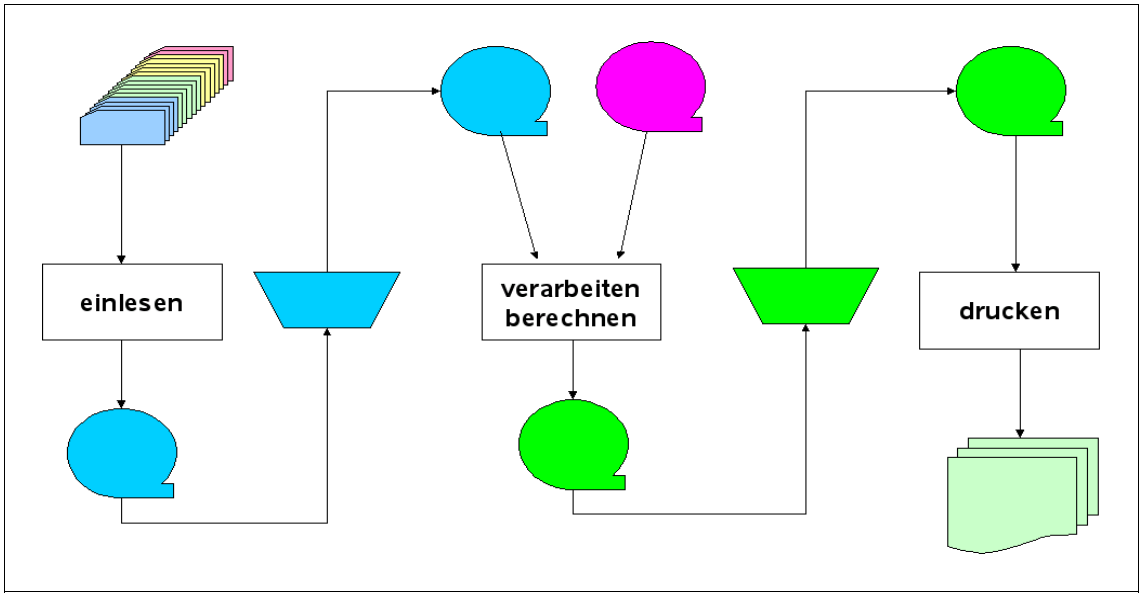

Abbildung 9: Grundprinzip der Stapelverarbeitung

Die Jobs (Lochkartenstapel) mussten dann in einem speziellen Eingaberaum an den Operator übergeben werden, dieser ließ dann den Job vom Rechner abarbeiten. Nach Ende der Verarbeitung wurde der Lochkartenstapel zusammen mit dem gedruckten Jobprotokoll und eventueller weiterer Ausdrucke im Ausgaberaum an den Auftraggeber zurückgegeben.

Auch hier war immer nur ein einziges Programm aktiv; der Rechner stand dem jeweils in Bearbeitung befindlichen Programm exklusiv zur Verfügung. Während der Zeiten die der Operator mit zwischen Eingaberaum, Rechner, Drucker und Ausgaberaum unterwegs war stand der Rechner ungenutzt herum.

Da andererseits die Kosten für die Rechner sehr hoch waren (mehrere Millionen Dollar) entstand die Forderung nach einer effizienteren Auslastung der Rechner. Eine Lösungsmöglichkeit (die sich allgemein durchsetzte) war, dass die zeitaufwendigen Ein- und Ausgaben durch einen (oder mehrere) relativ preiswerte Rechner erfolgten und lediglich die Berechnungen durch den teuren und leistungsfähigen Rechner durchgeführt wurden. Somit wurden die Lochkartenstapel und ggf. weiteren Daten mit einem kleinen (preiswerten) Rechner auf ein Magnetband eingelesen. Das Verfahren machte es möglich, mehrere Jobs auf dem Magnetband zu sammeln. Nach gewissen Zeitabständen (z.B.: stündlich) oder wenn das Band voll war, wurde das Band in den Rechnerraum gebracht und in ein Bandlaufwerk des eigentlichen Verarbeitungsrechners eingelegt. Der Operator lud ein spezielles Programm (Vorläufer eines Betriebssystems), mit dem die Jobs nacheinander vom Band gelesen und ausgeführt wurden. Die Ausgabe der einzelnen Job wurde auf einem zweiten Band (dem Ausgabeband) gesammelt. Die Druckausgabe der Ergebnisse erfolgte dann wieder "offline" auf dem preiswerteren Vorrechner.

Der Nutzungsgrad der relativ teuren Rechner konnte so deutlich erhöht werden, da ja für die Ein- und Ausgabe auch evtl. mehrere preiswerte Vorrechner eingesetzt werden konnten.

1.3.3 Die dritte Rechnergeneration (1965-1980): ICs und Mehrprogrammbetrieb

Die bisherige Entwicklung hatte dazu geführt, dass sich auf dem Markt zwei Entwicklungslinien durchgesetzt hatten. Einerseits die wortorientierten (wissenschaftlichen) Zahlenrechner ("Numbercruncher") und andererseits die zeichenorientierten kommerziellen Rechner. Wartung, Betreuung und Weiterentwicklung zweier Produktlinien war für die Hersteller (und teilweise auch für die Kunden) sehr Zeit- und kostenintensiv. Außerdem wollten die Anwender nicht bei jedem Rechnerwechsel (z.B. beim Kauf eines leistungsfähigeren Systems) alle Programme neu erstellen müssen.

Durch die Einführung der /360-Serie versuchte IBM dieses Dilemma zu lösen und führte auf dem Markt eine Rechnerfamilie ein, die sowohl für wissenschaftliche als auch kommerzielle Anwendungen geeignet war. Zudem gab es verschiedene Modellvarianten mit unterschiedlicher Leistungsfähigkeit. Da alle Modelle die gleiche Architektur hatten und auch mit dem gleichen Befehlssatz arbeiteten war auch die Portierung von Programmen auf einen anderen Rechner (der gleichen Modellreihe) möglich.

Die /360-Rechnerfamilie war die erste Rechnergeneration, die integrierte Schaltkreise (ICs) verwendete. Der daraus resultierende Vorteil beim Preis-/Leistungsverhältnis verhalf der /360-Linie zu einem großen Erfolg auf dem Rechnermarkt.

Wegen der Homogenität der Architektur sollten alle Programme auch auf allen Rechnern der Baureihe lauffähig sein – es entstand also die Forderung, dass die Software mit unterschiedlichen Rechnerkonfigurationen (z.B.: unterschiedliche Peripherieausstattung) zurecht kommen muss. Es musste also eine Software entwickelt werden, die quasi zwischen der individuell unterschiedlichen Hardwarekonfiguration und dem eigentlichen Anwendungsprogramm stand und die gegenüber dem Anwendungsprogramm immer die gleiche Verhaltensweise zeigte – das Betriebssystem.

Die Leistungsfähigkeit der Rechner ermöglichte es jetzt auch, dass mehre Programme gleichzeitig zur Ausführung kamen (Multiprogramming). Der Prozessor konnte während der Wartezeit für eine Ein- oder Ausgabeoperation für ein anderes Programm die Berechnung durchführen. Dadurch entstand die Erfordernis, den Arbeitsspeicher aufzuteilen und jedem aktiven Job einen Anteil (eine eigene Partition) zur exklusiven Nutzung zur Verfügung zu stellen. Als Folge wurde aber auch spezielle Hardware (und damit auch Software) erforderlich, die die Jobs gegen Ausspionieren oder Beschädigung des eigenen Bereichs durch andere Jobs schützt. Auch für den Zugriff auf den Prozessor (CPU) musste eine Regelung getroffen werden, die nicht einzelne Jobs zu Lasten der anderen Jobs einseitig bevorzugt.

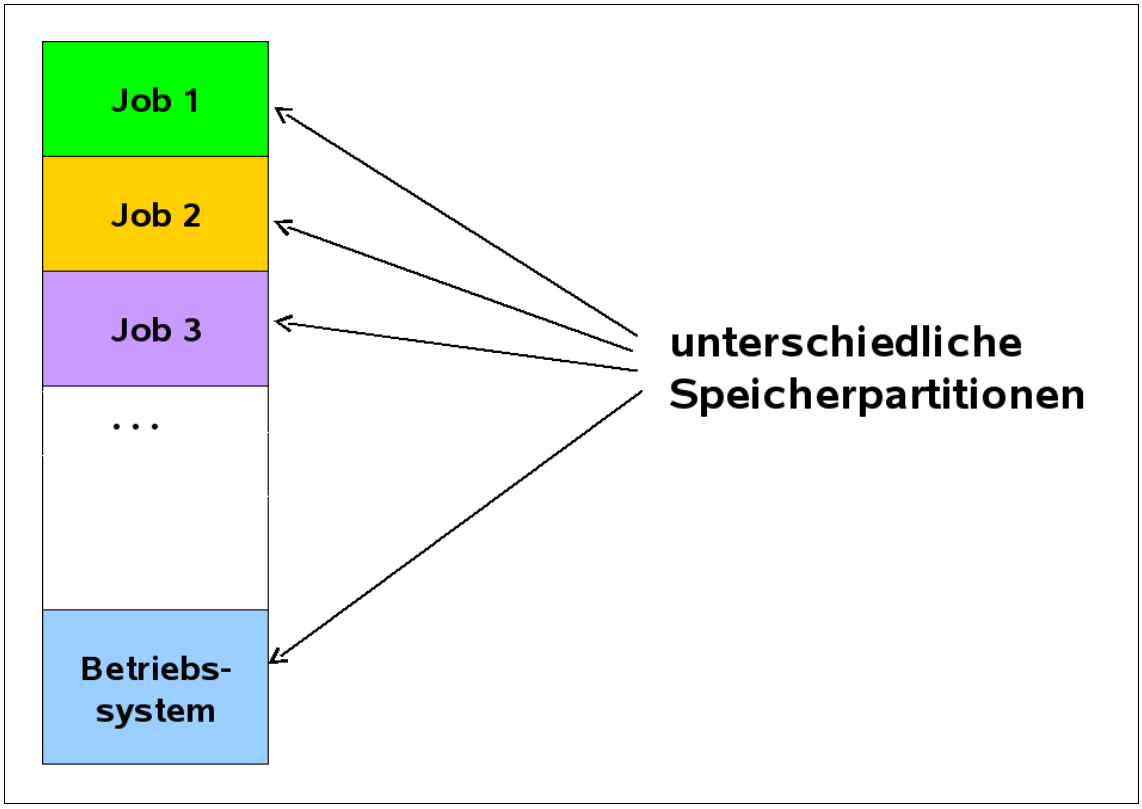

Abbildung 10: Speicheraufteilung eines Multiprogrammingsystems

Durch geeignete Vergabestrategien konnte so die Ausnutzung der CPU (dem teuersten Teil der gesamten Rechenanlage) auf nahezu 100% gesteigert werden, ohne dass die Laufzeit der einzelnen Jobs dadurch negativ beeinflusst wurde.

Als zusätzlicher Vorteil konnten Rechner der 3. Generation auch Daten (also auch ganze Jobs) auf Festplatten speichern. Sobald ein Job fertig war, konnte ein neuer Job gestartet werden bzw. das (bisher auf den Vorrechnern durchgeführte) Einlesen der Jobs und Ausgaben der Jobergebnisse konnte als eigenständige Aufgabe vom Großrechner mit übernommen werden. Das Verfahren wurde als SPOOL (**S**imultaneous **P**eripheral **O**peration **O**n **L**ine) bezeichnet – die Vorrechner und der manuelle Transport der Ein-/Ausgabebänder konnte entfallen.

Als großer Nachteil des Batchbetriebs stellte sich aber heraus, dass häufig zwischen den Abschicken eines Jobs und der Ausgabe des Ergebnisses mehrere Stunden lagen – für einen Programmierer, der ein neues Programm erstellen soll und durch Testläufe Fehler im Programm finden und beheben soll eine erhebliche Arbeitserschwernis. Es wurde deshalb schnell die Forderung nach kurzen Antwortzeiten gestellt, die eine weitere Neuerung - das Timesharing - zur Folge hatte. Hierbei wurde die CPU den aktiven Jobs / Benutzern reihum für kurze Zeit zur Verfügung gestellt und nach Ablauf der Frist wieder entzogen und dem nächsten Benutzer zugeteilt. Arbeitsschritte, die keinen CPU-Zugriff benötigen (z.B.: Denkpausen der Anwender; Ein- und Ausgabeoperationen, …) konnten somit von anderen Anwendern /Jobs genutzt werden.

1.3.4 Die vierte Rechnergeneration (1980-heute) Personal-Computer, Netzwerke und Client-Server-Konzepte

Der technische Fortschritt machte es möglich durch die Anwendung der LSI-Technologie (Large Scale Integration) tausende von Transistoren auf einem einzigen Silizium-Chip untergebracht werden konnten. Die Herstellungskosten waren vergleichbar gering. Damit konnten jetzt Personal-Computer, also Rechner für einzelne Personen hergestellt werden. Anforderungen wie Mehrbenutzerbetrieb, Optimierung der CPU-Auslastung etc. wurden jetzt zunächst einmal wieder überflüssig. Dagegen wurde aufgrund der Vielzahl unterschiedlicher Zusatzkomponenten (Festplatten, Drucker, …) die Forderung nach einer Hardware unabhängigen Programmentwicklung laut. Eine der Hauptaufgaben heutiger PC-Betriebssysteme ist es deshalb, geeignete Treiberprogramme für die unterschiedlichsten Hardwarekomponenten bereitzustellen oder vom jeweiligen Hardwarehersteller zu übernehmen.

Aufgrund der vergleichsweise geringen Kosten wurden PCs auch von „Nicht-DV-Spezialisten“ eingesetzt; hier kann jedoch die Forderung nach leichter Bedienbarkeit (möglichst mit graphischer Bedienoberfläche). Als Betriebssysteme haben sich hier insbesondere zwei Entwicklungslinien auf dem Markt durchgesetzt. MS-DOS (und später WINDOWS) und UNIX / LINUX.

Wegen der fortschreitenden Vernetzung der PCs ab etwa Mitte der 80er Jahre entstanden so genannte Netzwerkbetriebssysteme und verteilte Betriebssysteme, die den Zugang zu entfernten Rechnern steuern.

1.4 Arten von Betriebssystemen

Literaturhinweise: [Tann02] Kap. 1.3, Seite 30 ff

Derzeit unterscheidet man mehrere Kategorien von Betriebssystemen, die im Rahmen der historischen Entwicklung von DV-Anlagen und deren Anwendung entstanden sind. Die historischen Konzepte sind zum Teil noch deutlich zu erkennen.

[Tann02] unterscheidet die folgenden Systeme:

1.4.1 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
- Häufig mehrere Betriebsarten nebeneinander
  - Batchbetrieb
  - Transaktionsverarbeitung (z.B.: Buchungssysteme)
  - Zuteilungsverfahren (z.B.: Timesharing)
- Betriebssystem ist von Hardwarehersteller abhängig.
- Derzeit werden Mainframe-Betriebssysteme wieder häufiger eingesetzt (z.B.: als Webserver, Datenbankserver. B2B-Anwendungen, …)
- Beispiele sind MVS, OS/390, BS2000, …

1.4.2 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.
- Beispiele sind WINDOWS-NT, WINDOWS-2000, UNIX, LINUX, Novell NetWare, …

1.4.3 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, …

1.4.4 PC-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"
- Beispiele: WINDOWS, LINUX, Macintosh-Betriebssystem, …

1.4.5 Echtzeit-Betriebssysteme

- Spezielle Betriebssysteme, bei denen die Zeit eine besonders wichtige Rolle bei der Verarbeitung spielt
- Beispiele: Steuerung von Maschinen, Ampelsteuerung, Robotersteuerung, …

1.4.6 Betriebssysteme für eingebettete Systeme (embedded Systems)

- Spezielle Betriebssysteme, für spezielle Anwendungen.
- Anwendungsbeispiele: Palmtop. PDA, Handy, Fernseher, Mikrowelle, …
- Beispiele: PalmOS, WINDOWS CE, …

1.4.7 Betriebssysteme für Chipkarten

- Spezielle Betriebssysteme, für spezielle Anwendungen.
- Manchmal nur für eine Einzige Funktion entwickelt (z.B.: elektronisches Bezahlsystem)
- Anwendungsbeispiele: SmartCards, …

2. Grundlegende Konzepte

Literaturhinweise: [Tann02] Kap. 1.5, Seite 48 ff

Die detaillierte Realisierung / Implementierung eines Betriebssystems ist sehr eng an die Hardware des jeweiligen Rechensystems gekoppelt. Einige wichtige Aspekte sind hier (ohne Anspruch auf Vollständigkeit) zu nennen:

Abbildung 11: BS-relevante Merkmale der CPU

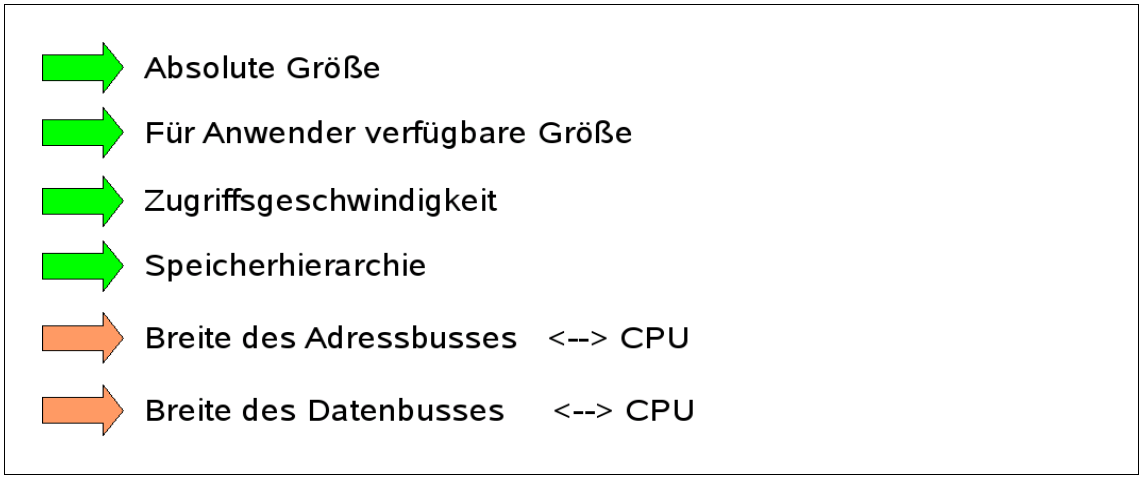

Abbildung 12: BS-relevante Merkmale des Hauptspeichers

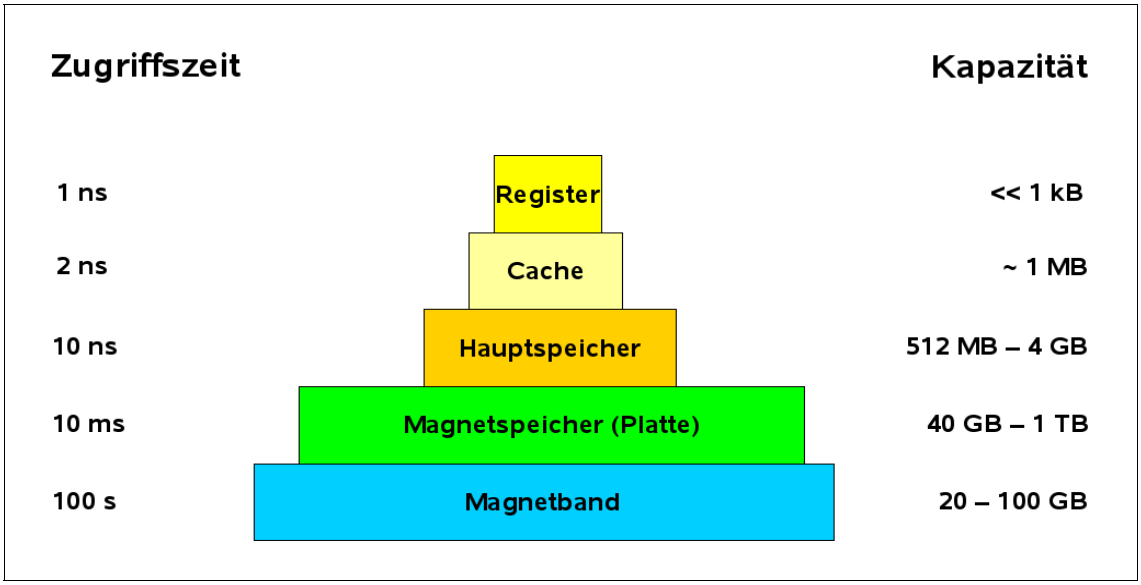

Abbildung 13: Speicherhierarchie

- Umdrehungszahl
- Schreib-/Lesegeschwindigkeit
- mittlere Zugriffszeit
- Positionierungszeit für Schreib-/Lesekopf
- Größenangaben (Sektor, Spur, Zylinder)
- Interleave-Faktor (historisch)
- Größe des Plattencaches
- BUS-Techologie (IDE, EIDE, SATA, USB, ...)

Abbildung 14: BS-relevante Merkmale von Festplatten

- Steuerung der E/A-Geräte erfolgt i.d.R. mit Hilfe gesonderter spezialisierter Hardware-Komponenten (Controller)

Abbildung 15: BS-relevante Merkmale von E/A-Geräten

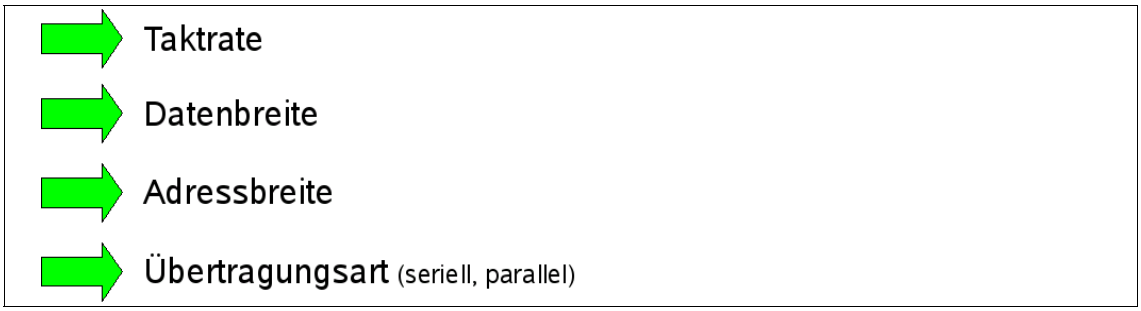

Abbildung 16: BS-relevante Merkmale von Bussystemen

2.1 Prozesse

Literaturhinweise: [Tann02] Kap. 1.5.1, Seite 48 ff

Prozesse sind ein Schlüsselkonzept aller Betriebssysteme.

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

Jedem Prozess ist ein Adressraum (adress space) 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

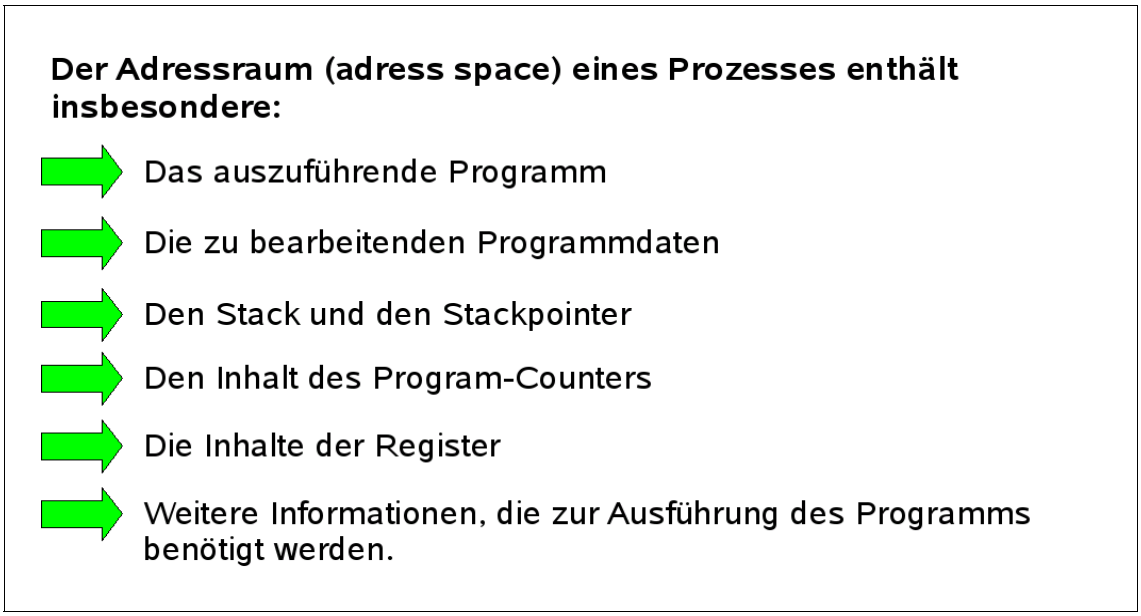

Abbildung 17: Adressraum eines Prozesses

Da die CPU ja immer "nur" genau einen Befehl bearbeiten kann (von-Neumann-Rechner), kann auch immer nur ein Prozess aktiv sein - die anderen Prozesse müssen warten. Das Betriebssystem entscheidet wann und wie lange ein Prozess eine Ressource (hier die CPU) nutzen darf und wann der Zugriff auf die Ressource dem Prozess wieder entzogen und an einen anderen Prozess übergeben wird.

- Der Zustand, in dem sich der Prozess zum Zeitpunkt des "anhaltens" befand muss gesichert werden, damit der Prozess an genau der gleichen Stelle zu einem späteren Zeitpunkt wieder fortgesetzt werden kann.

- Begriffe: Timesharing, Multiprogramming, Interrupt, …

Geöffnete Dateien müssen nach der Reaktivierung an genau der Stelle weiterbearbeitet werden können, an der die Verarbeitung vom Betriebssystem unterbrochen wurde. Hierzu werden verschiedene Dateizeiger benötigt.

Die Verwaltung von Prozessen erfolgt in vielen Betriebssystemen in Form einer Prozesstabelle. Hier werden die für jeden Prozess relevanten Daten gespeichert.

Prozesse können ggf. weitere Prozesse initiieren

- Vater-Kind-Prozess

- Prozesshierarchie

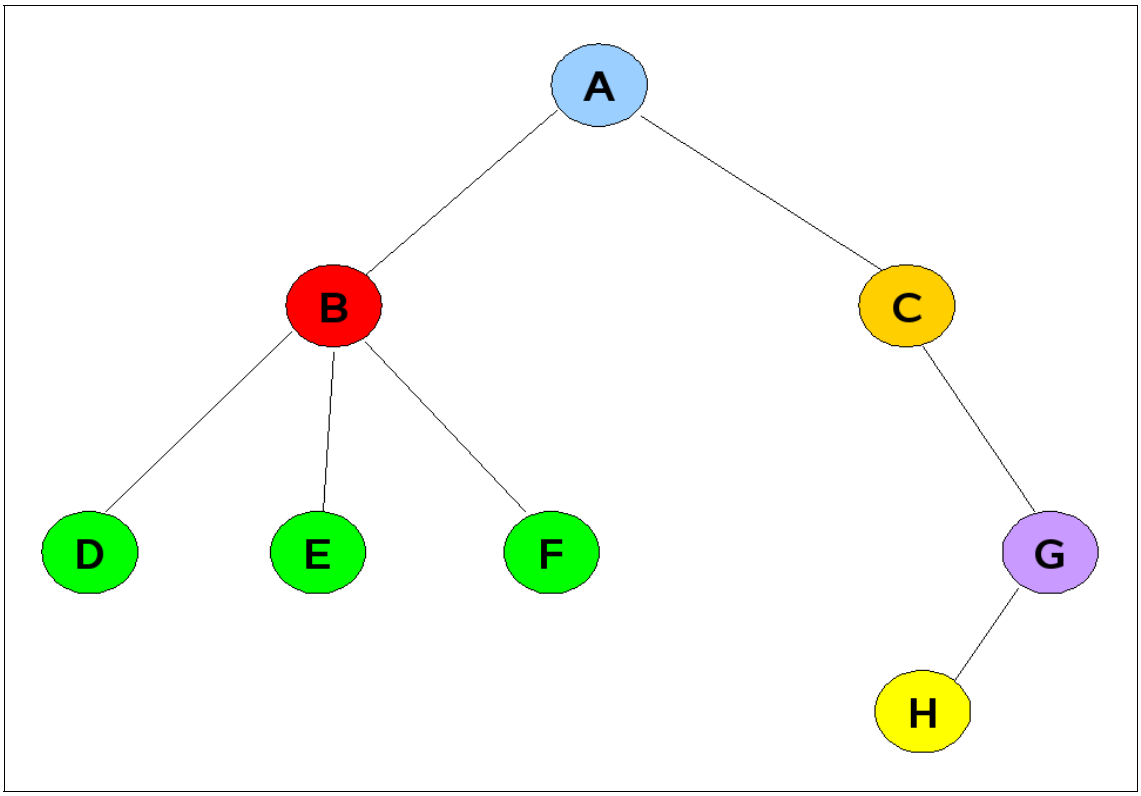

Abbildung 18: Prozesshierarchie

Prozesse, die zusammen ein Problem bearbeiten (typischerweise innerhalb einer Prozesshierarchie) müssen oft miteinander kommunizieren und ihre Aktionen gegenseitig synchronisieren. Dies nennt man **Interprozesskommunikation**.

- Beispiel: Der Verarbeitungsprozess muss auf den Abschluss der Leseoperation warten, die von einen anderen Prozess durchgeführt wird und die zu verarbeitenden Daten aus einer Datei liest.

Es kann vorkommen, dass 2 (oder mehr) Prozesse auf die Rückmeldung des jeweils anderen Prozesses warten und somit nicht weiter arbeiten können (**Deadlock**). Zur Erkennung und Auflösung derartiger Deadlock-Situationen muss das Betriebssystem entsprechende Mechanismen besitzen.

- Beispiel: Die Daten eines Bandlaufwerks sollen auf CD gespeichert werden.
  - Prozess P1 liest derzeit das Bandlaufwerk
  - Prozess P2 schreibt derzeit auf CD
  - P1 will die von Band gelesenen Daten auf CD schreiben und benötigt hierzu auch noch den CD-Zugriff. Da das CD-Laufwerk derzeit von P2 belegt ist, wartet P1 bis das CD-Laufwerk frei wird. Die Zugriffsberechtigung auf das Bandlaufwerk behält P1 solange.

- P2 will das Bandlaufwerk benutzen - dieses ist aber derzeit von P1 belegt, also wartet P2 solange, bis das Bandlaufwerk freigegeben wird, Die Zugriffsberechtigung auf das CD-Laufwerk behält P2 solange.

- Diese Deadlock-Situation muss vom Betriebssystem aufgelöst werden.

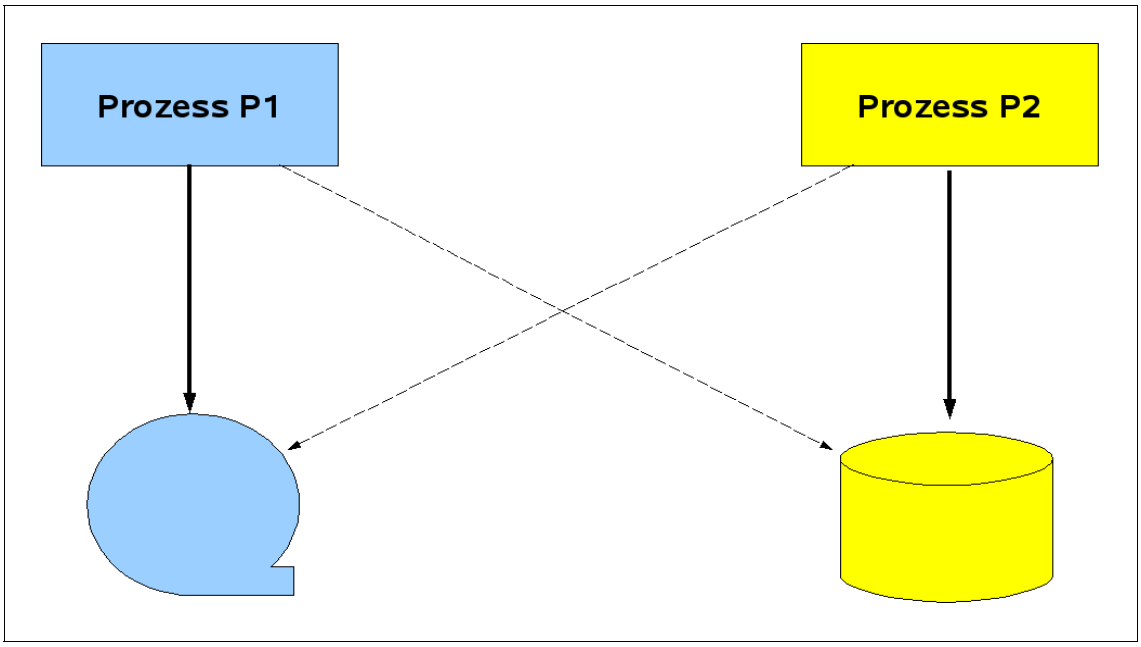

Abbildung 19: Deadlock

2.2 Speicherverwaltung

Literaturhinweise: [Tann02] Kap. 1.5.3, Seite 51

- Mehrere Prozesse müssen einen (begrenzten) Arbeitsspeicher eines Rechners nutzen, ohne sich gegenseitig zu behindern oder zu beeinflussen.

- Der Adressraum des Arbeitsspeichers ist begrenzt (z.B. [0 … 2^{32}] oder [0 … 2^{64}] Speicherzellen). Falls größere Programme oder Programme mit größeren Speicherbedarf (z.B.: Bearbeitung großer Datenmengen) ausgeführt werden sollen, sind hier zusätzliche organisatorische Maßnahmen vom Betriebssystem vorzusehen. Siehe folgende Begriffe:

  - Virtueller Speicher
  - Paging
  - Swapping
  - Absolute Adressierung
  - Relative Adressierung

2.3 Ein-/Ausgabe

Literaturhinweise: [Tann02] Kap. 1.5.4, Seite 51 ff

Ein Betriebssystem verwaltet ein eigenständiges E/A-Subsystem mit dem Ziel:

- Software soll geräteunabhängig sein
- Anpassung an spezielle E/A-Geräte mit Hilfe gerätespezifischer Software (**Geräte-Treiber**)

2.4 Dateiverwaltungssystem

Literaturhinweise: [Tann02] Kap. 1.5.5, Seite 52 ff

2.4.1 Dateien und Verzeichnisse

- Das Betriebssystem verbirgt Einzelheiten der gerätespezifischen Details. Dem Programmierer / Anwender wird ein Modell geräteunabhängiger Dateien präsentiert.
- Zum Erzeugen / Lesen / Schreiben / Löschen von Dateien werden Systemaufrufe benötigt.
- Dateien werden in den meisten Betriebssystemen in Form von Verzeichnissen gruppiert - ein Instrument zur leichteren Verwaltung.
    - Ordnen von Dateien
    - Dateihierarchie

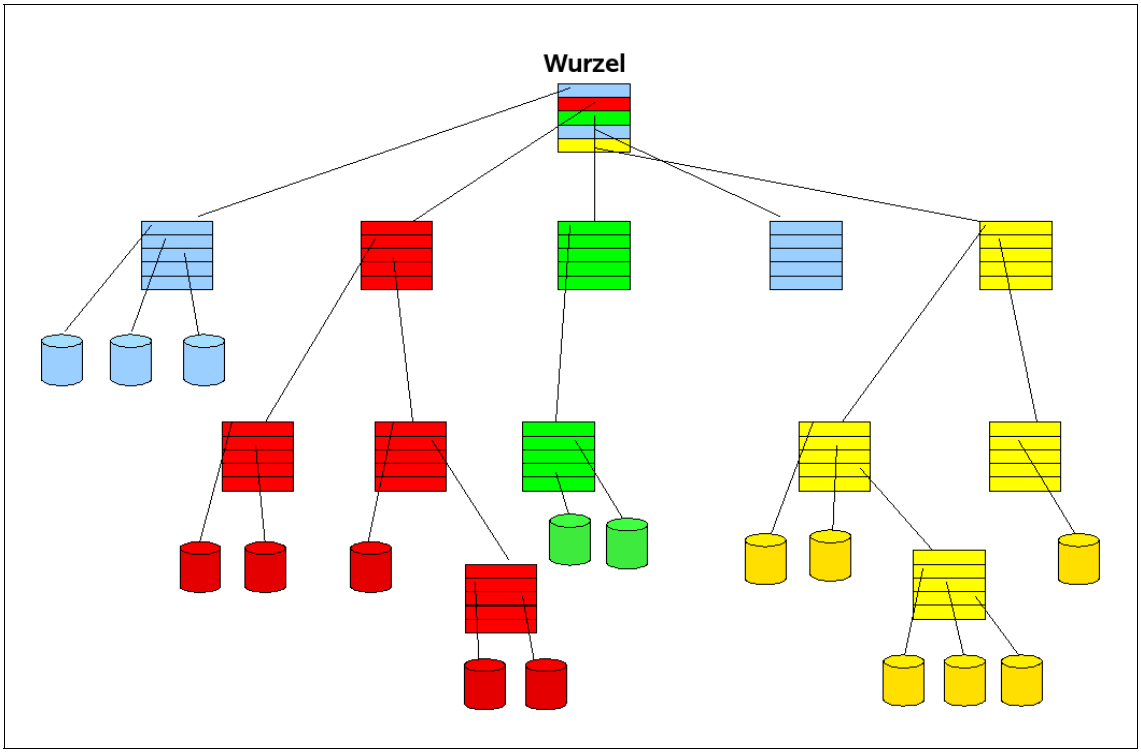

Abbildung 20: Verzeichnishierarchie

- Verzeichnishierarchien sind (im Gegensatz zu Prozesshierarchien) langlebig und häufig über mehrere (teilweise viele) Ebenen gestaffelt.
- Der Zugriff auf die jeweils gewünschte Datei erfolgt durch Angabe des Zugriffsweges (Pfad)
  - Absoluter Pfadname
  - Relativer Pfadname
- Bevor eine Datei bearbeitet (gelesen oder geschrieben) werden kann, muss sie vorher geöffnet werden und die Zugriffsrechte überprüft werden. Dies wird durch das Betriebssystem per Systemaufruf erledigt. Ist das Dateizugriff erlaubt, liefert das Betriebssystem einen **Dateideskriptor** (**File-Handle**) als Ganzzahl zurück. In den folgenden Operationen wird dann nur noch der File-Handle verwendet.
- Unter UNIX/LINUX können Dateisysteme an vielfältigen Stellen (nahezu beliebig) in den Verzeichnisbaum eingehängt werden (**mount**).

2.4.2 Spezialdateien (special files)

Spezialdateien werden häufig dazu verwendet, gerätespezifische Besonderheiten vor dem Anwender / Programmierer zu verbergen (z.B.: **device-files**). E/A-Geräte können dann wie Dateien behandelt werden (blockorientiert, zeichenorientiert).

2.4.3 Pipes (named pipes)

Pipes sind Dateien, die als Kommunikationskanal dienen und somit die Interprozesskommunikation ermöglichen. Auf Pipes müssen alle beteiligten Prozesse Zugriffsrechte besitzen.

2.5 Das Client-Server-Modell

Literaturhinweise: [Tann02] Kap. 1.7.5, Seite 77 ff

Zielsetzung der Client-Server-Architektur ist es, die Leistungen, die bei konventionellen Betriebssystemen auf einem einzigen Rechner zur Verfügung stehen, im Netzwerk für mehrere Systeme bereitzustellen. Beispielsweise die gemeinsame Nutzung eines Druckers.

Lösungsansatz:

- Aus dem eigentlichen Betriebssystem wird soviel Funktionalität wie möglich in höhere Abstraktionsebenen (~Benutzerprogramme) verlagert, bis nur noch ein minimaler **Mikrokern** übrig bleibt.
- Der größte Teil der Betriebssystemfunktionalitäten wird in Benutzerprogramme (**Dienste**) verlagert.
- Um einen Dienst anzufordern (z.B.: einen Block einer Datei lesen) sendet ein Benutzerprozess (Client-Prozess) eine Anforderung (Request) an einen anderen (speziellen) Benutzerprozess (Server-Prozess) - dieser führt die Bearbeitung des Auftrags durch und gibt eine Antwort zurück.

Konsequenzen:

- Das Betriebssystem wird in einzelne spezialisierte Teile aufgeteilt
    - File-Server
    - Prozess-Server
    - Terminal-Server
    - Print-Server
    - Backup-Server
    - …
- Die Komponenten laufen als "Server-Prozesse" auf einem gemeinsamen System oder auf verschiedenen Rechnern im Netz. Für die Anwendung macht dies im Regelfall keinen Unterschied.
- Server-Prozesse laufen im Benutzermodus. Falls ein Prozess nicht korrekt funktioniert oder gar abbricht hat der Prozess auch aufgrund der jetzt eingeschränkten Berechtigungen nicht die Möglichkeit, das gesamte System lahm zu legen.

3. Abbildungsverzeichnis

Abbildungsverzeichnis

4. Stichwortverzeichnis

Stichwortverzeichnis