German - Baltic Management School

Betriebssysteme

Zur Erlangung des Grades

Bachelor

im Studiengang Informatik / Computer Science / B.Sc.

an der Fakultät Informatik

der German-Baltic Management School

ABSTRACT

Betriebssysteme

Einführung und Definition

Klassifizierung und Struktur

Architektur

Prozesse und Threads

Speicherverwaltung

Input/Output

Dateisysteme

Verteilte Ressourcen

Cluster und Grid

Historische, moderne und zukünftige Betriebssysteme

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

1 EINLEITUNG UND DEFINITION

Ein Betriebssystem ist ein System, das als Vermittler zwischen Benutzer, Anwendungen und Hardware fungiert. Der Zweck eines Betriebssystems ist es eine Platform zu schaffen in der ein Benutzer seine Programme effizient ausführen kann.

Um zu begreifen was Betriebssysteme sind, muss man zuerst verstehen wie sie sich entwickelten. Von den ersten verfügbaren Systemen zu den aktuellen Mehrprogramm-Systemen und Systemen mit Timesharing. Sie sollen als Beispiele dienen, wie sich Komponenten der aktuellen Betriebssysteme als naheliegende Lösungen aus den Problemen der frühen Betriebssysteme entwickelt haben. Wenn man die Probleme bei der Entwicklung von Betriebssystemen versteht, versteht man besser welche Zwecke ein Betriebssystem verfolgt und wie sie wahrscheinlich weiter verfolgt werden.

Abbildung 1 Abakus

2 KLASSIFIZIERUNG

2.1 KLASSIFIZIERUNG NACH GENERATIONEN

Die erste Computergeneration (ca. 1945-1955) besaß kein ausgewiesenes Betriebssystem. Alle Aufgaben die das System bewältigen konnte, waren fester Bestandteil der Hardware. Sie zeichnete sich aus durch:

- Direkte Programmierung mit Steckbrett, Lochstreifen, Lochkarte
- Keine Programmiersprachen

Die zweite Generation (ca. 1955-1965) arbeitete mit Stapelverarbeitung. Ein Auftrag wird in geschlossener Form, bestehend aus Programm, Daten und Steueranweisungen, zusammengestellt. Die Resultate erhält der Benutzer erst nach Abschluss der Bearbeitung zurück (meist als Ausdruck). Der Ausdruck Stapelverarbeitung oder Stag kommt auch heute noch von dem damaligen Lochkartenstapel, den man eingelesen hat. Typische Eigenschaften sind:

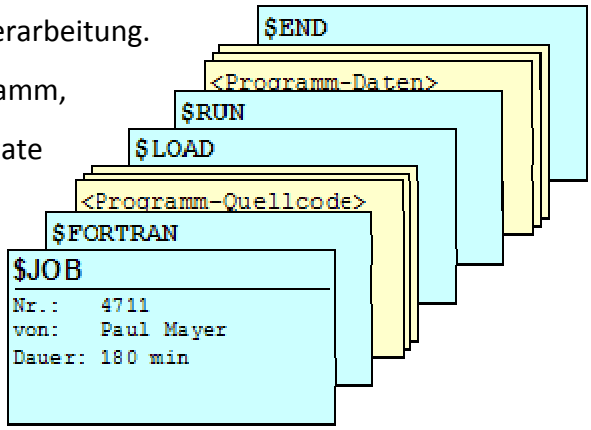


Abbildung 2 Stapelverarbeitung

- Batch-Betrieb (Lochkarten)
- Einfache Job-Control-Sprachen
- Programmiersprachen (Assembler, Fortran, etc.)
- Magnetbänder als Zwischenspeicher

Die dritte Generation (ca. 1965-1980) beherrscht Dialogverarbeitung. Der Benutzer kommuniziert mit dem Computer über Tastatur und Bildschirm, mit deren Hilfe er Programme starten, verfolgen und beeinflussen kann.

- Multiprogramming = Mehrprogrammbetrieb = Mehrere Programme gleichzeitig im Speicher quasisimultane, zeitlich verschachtelte Bearbeitung auf der Auftragsebene.
- Hauptspeicheraufteilung für mehrere Programme

Klassifizierung

- Zeitliche Verschachtelung der Programme (z.B.: Prog. A wartet auf Ausgabe, Prog. B rechnet) --> Timesharingbetrieb mit Terminals
- SPOOLING (simultaneous peripheral operation on line) direktes Speichern von Rechenaufträgen auf der Platte, "Selbstbedienung" des BS
- MULTICS als UNIX-Vorgänger
- 1969 das erste UNIX

Die vierte Generation (ab ca. 1975) ist ein Dialogsystem, wie wir es heute kennen. Zunächst erfolgte der Dialog im Textmodus über Tastatur und Textbildschirm. Später wurden grafische Benutzeroberflächen entwickeltwie GEM von Digital Research, Apple OS von Lisa und Macintosh, Windows von Microsoft und X von UNIX.

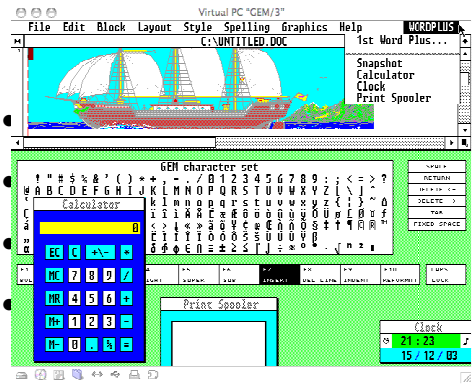


Abbildung 3 GEM von DR

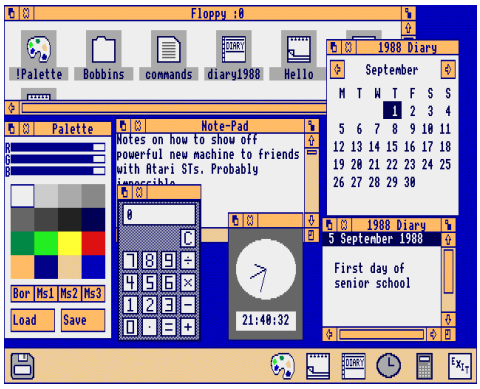


Abbildung 4 Apple OS von Mac

- UNIX und C
- Multitasking als quasisimultane Ausführung weitgehend unabhängiger Programmabschnitte innerhalb eines Auftrags.
- Personal Computer (CP/M, MS-DOS, etc.)
- Netzwerkbetriebssysteme (Kommunikation mehrerer Computer)
- verteilte Betriebssysteme (mehrere Prozessoren = Multiprocessing) Mehrere Prozessoren bilden ein Computersystem --> Mehrere Programme werden von verschiedenen Prozessoren bearbeitet oder ein Programm von mehreren Prozessoren.

Bei bestimmten Betriebssystemen spielt auch die Verarbeitungszeit eine Rolle. Bei Realzeit-(Echtzeit-)Betriebssystemen für Steuerungs- und Regelungsaufgaben (sog. Prozeßrechner)

spielt die Antwortzeit eine Rolle. Informationen werden hier zum Teil von elektronischen Meßwandlern (Sensoren) gewonnen. Das Programm reagiert auf äußere Ereignisse in angemessen kurzer Zeit (Maschinenregelung: 1 - 10 ms, Prozeßregelung: 10 - 100 ms, Prozeßsteuerung: 0,1 - 1 s). Es wird spezielle Hard- und Software benötigt.

2.2 KLASSIFIZIERUNG NACH DER BETRIEBSART DES RECHNERSYSTEMS

Stapelverarbeitungs-Betriebssysteme (batch processing) Frühe Betriebssysteme erlaubten nur den Stapelbetrieb (Lochkarten, etc.) und auch heutige Systeme besitzen vielfach die Möglichkeit, Progammabfolgen automatisch zu bearbeiten (z. B. Batch-Dateien bei DOS, Shell-Scripte bei UNIX, usw.)

Dialogbetrieb-Betriebssysteme (interctive processing, dialog processing) Der/die Benutzer bedienen den Rechner im Dialog mittels Bildschirm, Tastatur, Maus, usw.). Die Bedienoberfläche kann textorientiert oder grafisch sein.

Netzwerk-Betriebssysteme (network processing) Sie erlauben die Einbindung des Computers in ein Computernetz und so die Nutzung von Ressourcen anderer Computer. Dabei unterscheidet man zwischen Client-Server-Betrieb, bei dem Arbeitsplatzrechner auf einen Server zugreifen, und Peer-to-Peer-Netzen, bei denen jeder Rechner sowohl Serverdienste anbietet als auch als Client fungiert.

Realzeit-Betriebssysteme (realtime-processing) Hier spielt, neben anderen Faktoren, die Verarbeitungszeit eine Rolle (s. oben).

Universelle Betriebssysteme Diese Betriebssysteme erfüllen mehrere der oben aufgeführten Kriterien.

2.3 KLASSIFIZIERUNG NACH DER ANZAHL DER TASKS

In dieser Klassifikation kommt der Begriff *"Task"* vor. Alternativ kann der deutsche Begriff *"Prozeß"* Verwendung finden. Aus Anwendersicht kann an dieser Stelle auch der Begriff *"Aufgabe"*bzw. *"Auftrag"* verwendet werden.

Single Tasking (Einzelprogrammbetrieb) Ein einziges Programm läuft jeweils zu einem bestimmten Zeitpunkt. Mehrere Programme werden nacheinander ausgeführt.

Multitasking (Mehrprogrammbetrieb) Mehrere Programme werden gleichzeitig (bei mehreren CPUs) oder zeitlich verschachtelt, also quasi-parallel, bearbeitet.

2.4 KLASSIFIZIERUNG NACH DER ANZAHL DER BENUTZER:

Singleuser Mode (Einzelbenutzerbetrieb) Der Computer steht nur einem einzigen Benutzer zur Verfügung.

Multiuser Mode (Mehrbenutzerbetrieb) Mehrere Benutzer teilen sich die Computerleistung, sie sind über Terminals oder Netzwerkverbindungen mit dem Computer verbunden.

2.5 KLASSIFIZIERUNG NACH DER ANZAHL DER VERWALTETEN PROZESSOREN

Nicht die Frage wie viel Prozessoren in einem Rechner verwendet werden, sondern wie viele *Universalprozessoren* für die Verarbeitung der Daten zur Verfügung stehen ist hier entscheidend. In einem modernen Rechner gibt es mindestens einen Hauptprozessor (CPU, Central Processing Unit). Daneben gibt es einige weitere Prozessoren, z. B. den Grafikprozessor, der spezielle Eigenschaften und auch einen speziellen Befehlssatz besitzt, oder auf dem Controller für die SCSI-Schnittstelle. Somit ergibt sich folgende Unterscheidung:

Klassifizierung

Ein-Prozessor-Betriebssystem Die meisten Rechner, die auf der von-Neumann-Architektur aufgebaut sind, verfügen über nur einen Universalprozessor. Aus diesem Grund unterstützen auch die meisten Betriebssysteme für diesen Anwendungsbereich nur einen Prozessor.

Mehr-Prozessor-Betriebssystem Für diese Klassifizierung der Betriebssysteme ist noch keine Aussage über die Kopplung der einzelnen Prozessoren getroffen worden. Auch gibt es keinen quantitativen Hinweis über die Anzahl der Prozessoren, nur, dass es mehr als ein Prozessor ist. Für die Realisierung der Betriebssysteme für die Mehrprozessorsysteme gibt es zwei Herangehensweisen:

- Jedem Prozessor wird durch das Betriebssystem eine eigene Aufgabe zugeteilt. D.h., es können zu jedem Zeitpunkt nur so viele Aufgaben bearbeitet werden, wie Prozessoren zur Verfügung stehen. Es entstehen Koordinierungsprobleme, wenn die Anzahl der Aufgaben nicht gleich der Anzahl verfügbarer Prozessoren ist.

- Jede Aufgabe kann prinzipiell jedem Prozessor zugeordnet werden, die Verteilung der Aufgaben zu den Prozessoren ist nicht an die Bedingung gebunden, dass die Anzahl der Aufgaben gleich der Anzahl Prozessoren sein muss. Sind mehr Aufgaben zu bearbeiten als Prozessoren vorhanden sind, so bearbeitet ein Prozessor mehrere Ausgaben "quasi-parallel". Sind mehr Prozessoren als Aufgaben vorhanden, dann bearbeiten mehrere Prozessoren eine Aufgabe.
- Das Betriebssystem kann auch auf mehrere Prozessoren verteilt sein. Man spricht dann von "verteilten Betriebssystemen".

3 ARCHITEKTUR

Unter der Architektur versteht man eine Klassifikation der Arbeitsweise von Betriebssystemen.

3.1 MONOLITHISCHE ARCHITEKTUR

Alle Komponenten sind starr zu einem homogenen Gebilde zusammengefügt. Dadurch wird eine hohe Effizienz für spezielle Geräte erreicht, aber auf Kosten der Flexibilität.

Diese Technik findet vor allem bei Embedded Systems Verwendung.

Abbildung 5 Embedded Systems

3.2 MEHRSCHICHTIGE ARCHITEKTUR

Funktionelle Trennung des Betriebssystems in hierarchische Schichten oder Schalen, die mit unterschiedlichen Privilegien ausgestattet sind.

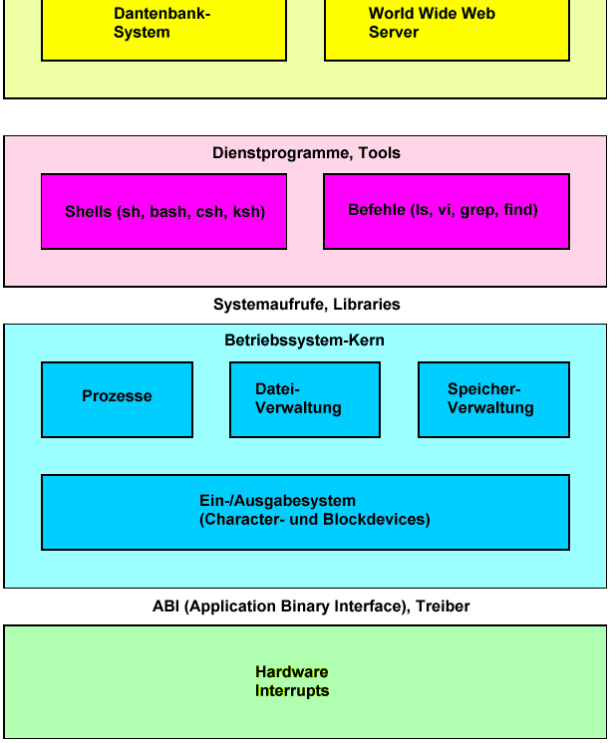


Abbildung 7 Schichtenarchitektur

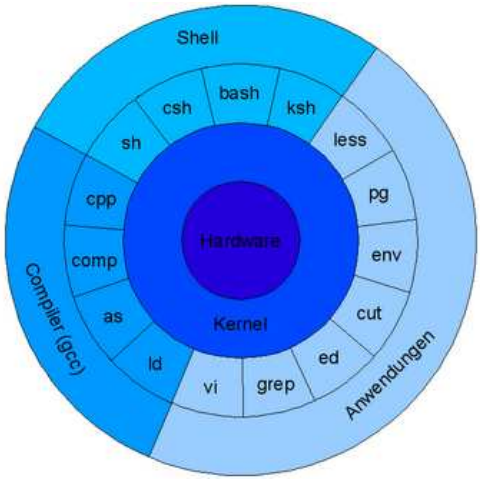


Abbildung 6 Schalenarchitektur

3.3 MIKROKERN ARCHITEKTUR

Ein **Mikrokernel** (oder auch Mikrokern) bezeichnet einen Betriebssystemkern. Dieser wird vor allem bei Echtzeitbetriebssystemen verwendet. Der Mikrokernel verfügt über weniger Funktionen als ein Monolithischer Kernel - in der Regel lediglich Funktionen zur Speicher- und Prozessverwaltung, sowie Grundfunktionen zur Synchronisation und Kommunikation.

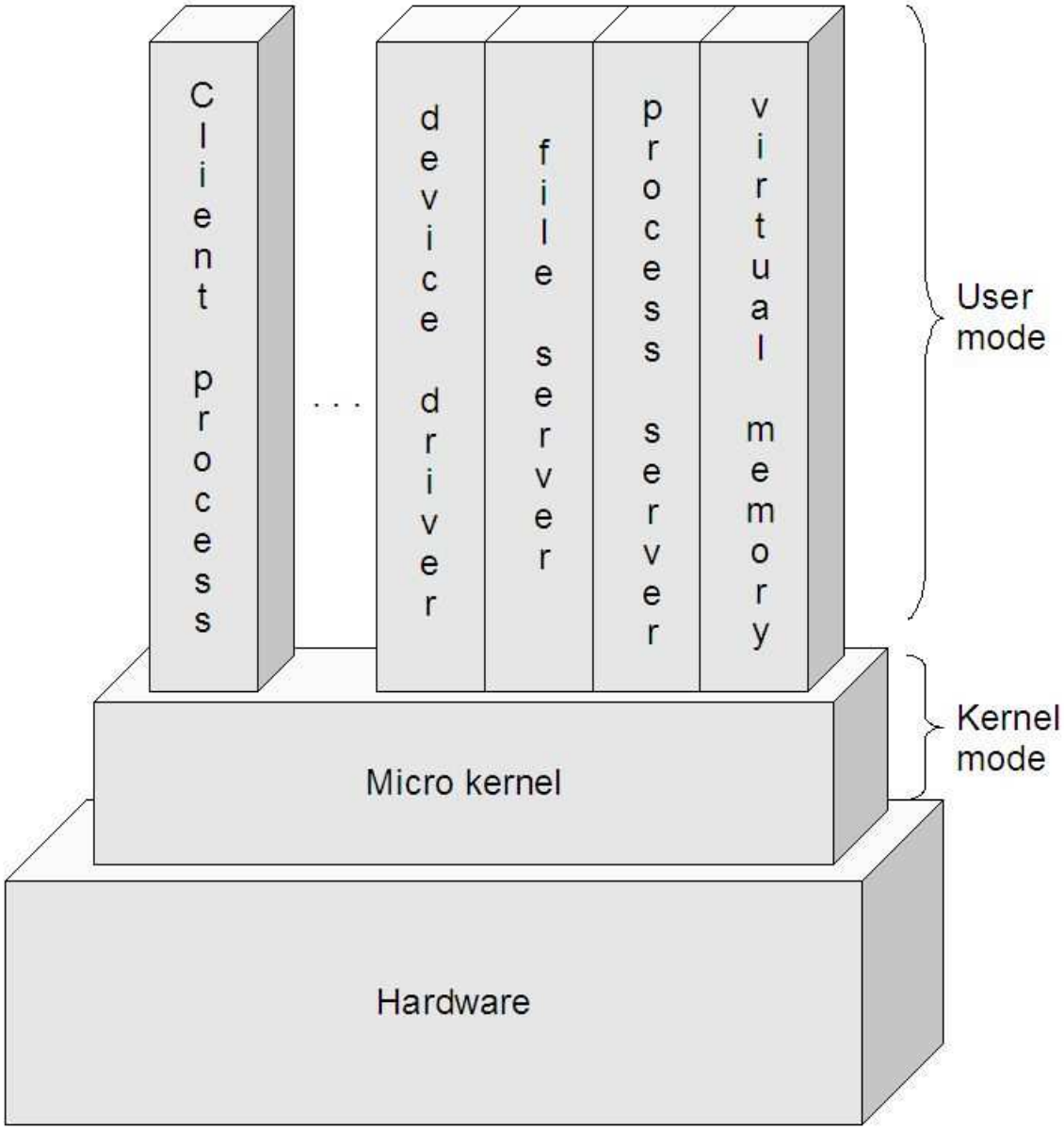


Abbildung 8 Schaubild Mikrokernel

4 PROZESSE UND THREADS

In der Informatik ist ein Prozess ein Programm. Ein Programm wiederum ist die Lösung einer Programmieraufgabe, die sich in zwei problemorientierte Teile gliedert

- Befehlen (Codebereich, Textbereich)
- Programmdaten (Datenbereich)

Ein Thread ist ein „kleines oder leichtes Programm“. Es ist eine Erweiterung des Prozessmodells. Er teilt sich mit anderen Threads die Betriebsmittel und den Prozesskontext eines Prozesses und besitzt einen eigenen Stapelspeicher (Stack).

4.1 DAS PROZESSMODELL

Ein Prozessor kann immer nur einen Prozess verarbeiten. Daher sind heutige Mehrprozessorkerne, sinnvoll eingesetzt, performanter als vermeintlich gleich schnelle Einkernprozessoren. Bei den ersten Computern wurden Programme immer nacheinander als Ganzes verarbeitet, sie konnte immer nur exklusiv ablaufen. Auch konnten mehrere Benutzer einen Computer nicht gleichzeitig verwenden. Daher wurde die Möglichkeit geschaffen, Prozesse nur teilweise auszuführen, zu unterbrechen, und später wieder aufzusetzen und fortzuführen, was im Prozessmodell beschrieben wird.

Das Prozessmodell beschreibt die wesentlichen Zustände eines Prozesses:

- WARTEND: Der Prozess wartet auf die Zuteilung eines Prozessors.
- LAUFEND: Der Prozess ist aktuell einem Prozessor zugeordnet und läuft ab.
- UNTERBROCHEN: Der Prozess wurde durch einen anderen Prozess unterbrochen.
- SUSPENDIERT: Prozesse aus dem Speicher entfernen und auslagern.

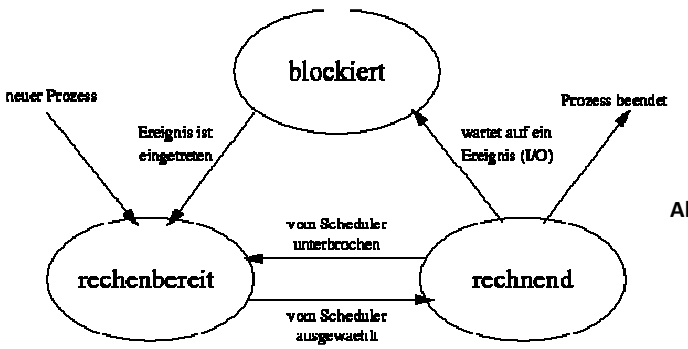


Abbildung 9 Prozessübergänge

Ein Betriebssystem benutzt dieses Modell als Strategie um den Anschein parallel ablaufender Prozesse zu simulieren. Dabei ändert das Betriebssystem eigenständig den Zustand jeden Prozesses zwischen WARTEND und LAUFEND hin und her, bis alles abgearbeitet ist. Trotzdem ist pro Zeiteinheit ist immer nur ein Prozess im Zustand LAUFEND.

Weitere vom Betriebssystem gestartete Prozesse, können einen anderen Prozess unterbrechen. Er ist dann im Zustand UNTERBROCHEN und wird vom Betriebssystem auch nicht mehr zeitweise in den Zustand LAUFEND versetzt. Erst wenn ein Prozess den Prozess wieder aufsetzt und in den Zustand WARTEND bringt, ist das wieder der Fall. So können sich zeitkritische Prozesse eine eigene kleine Echtzeitumgebung schaffen. Zusätzlich kann auch das Betriebssystem Prozessmodelle verwalten und somit einem Prozess eine Priorität zuweisen. Betriebssysteme benutzen unterschiedliche Strategien, ihren Prozessen Zeitabschnitte eines Prozessors zuzuordnen.

4.1.1 Prozess-Scheduling

Prozess-Scheduler kann man grob in unterbrechende (preemptive) und nicht unterbrechende (non preemptive, bzw. kooperativ) aufteilen. Nicht unterbrechende Scheduler lassen einen Prozess, nachdem ihm die CPU einmal zugeteilt wurde, solange laufen, bis dieser diese von sich aus wieder freigibt oder bis er blockiert. Unterbrechende Scheduler teilen die CPU von vornherein nur für eine bestimmte Zeitspanne zu und entziehen dem Prozess diese daraufhin wieder. Weiter ist eine Unterscheidung in "work-conserving" und "nicht work-conserving" Strategien möglich. Eine Scheduler-Strategie arbeitet "work-conserving", wenn das Umschalten zwischen Prozessen nur eine vernachlässigbar geringe Zeit in Anspruch nimmt.

4.1.2 Dispatcher

Der Dispatcher dient dazu bei einem Kontextwechsel dem derzeit aktiven Prozess die CPU zu entziehen und anschließend dem nächsten Prozess die CPU zuzuteilen. Die Entscheidung,

welcher Prozess der *nächste* ist, wird vom Scheduler im Rahmen der Warteschlangenorganisation getroffen.

Der Scheduler wird in der Operation *assign* (ein Thread geht vom Zustand *ready* zu *running* über) vom Dispatcher aufgerufen und trifft dann sowohl langfristige als auch kurzfristige Entscheidungen.

4.1.3 ROUND ROBIN

Weit verbreitet ist das Zeitscheibenverfahren fester Länge verbunden mit einer priorisierten Warteschlange (Vorrangwarteschlange) als Ringpuffer.

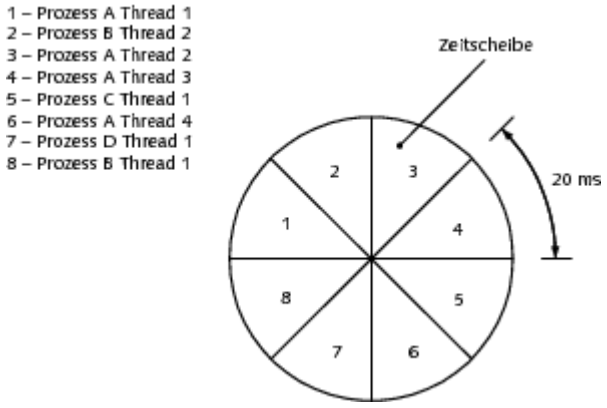


Abbildung 10 Das Round Robin Zeitscheibenverfahren

4.1.4 STRATEGIEN IN DER PRAXIS

Strategien die in der Praxis verwendet werden

- **first come, first served** Wer zuerst kommt, wird zuerst bedient. Kommen zwei Prozesse genau gleichzeitig, wird eine zufällige Auswahl getroffen.
- **Round robin** Zeitscheibenverfahren. Jeder Prozeß erhält eine feste zugeordnet. Nach Ablauf dieser Zeitspanne wird er verdrängt und der nächste Prozeß erhält die CPU --> zyklische Zuteilung. Alle Prozesse haben immer die gleiche Priorität. Die Zeitspanne kann konstant sein oder abhängig von der Prozessorbelastung variieren.

- **Prioritätssteuerung**. Jedem Prozeß wird eine Priorität zugeordnet. Ein Prozeß niedrigerer Priorität wird erst bearbeitet wenn alle Prozesse höherer Priorität abgearbeitet sind. Ein neuer Prozeß höherer Priorität verdrängt einen aktiven Prozeß niedrigerer Priorität.

4.2 PROZESSHIERARCHIE

Ein Prozeß kann einen neuen Prozeß starten (fork, spawn). Der so gestartete Prozeß heißt "Kindprozeß" (child process) und der Erzeuger-Prozeß wird "Elternprozeß" (parent process) genannt.

- Jeder Kindprozeß hat genau einen Elternprozeß
- Ein Elternprozeß kann mehrere Kindprozesse besitzen
- Eltern- und Kindprozeß können miteinander kommunizieren
- Wird ein Elternprozeß beendet, beenden sich normalerweise auch alle seine Kindprozesse

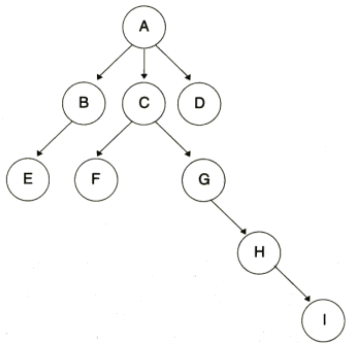


Abbildung 11Prozesshierarchie

4.3 PROZEßRECHTE

Jeder Prozeß hat im Verlauf seiner Existenz zwei Benutzeridentifikationen:

- **Die reale Benutzer-ID** bezeichnet den Benutzer, der für den Ablauf des Prozesses verantwortlich ist. Sie sollte konstant bleiben.
- **Die effektive Benutzer-ID** kann sich jedoch beliebig oft ändern und bestimmt jeweils die momentanen Rechte des Prozesses in Bezug auf Dateizugriffe und andere Mechanismen, die benutzerabhängigen Einschränkungen unterliegen.

Die Änderung der effektiven Benutzer-ID kann auf zweierlei Art erfolgen. Wenn der Prozeß ein Programm eines anderen Benutzers ausführen möchte, dann prüft das Betriebssystem über die Zugriffsrechte seine Berechtigung. Falls die Zugriffsrechte der Programmdatei das setuid-Bit enthalten, wird die effektive Benutzer-ID des Prozesses mit der

Eigentümerkennung des aufgerufenen Programms überschrieben, und der Prozeß hat die Rechte des Eigentümers des Programms.

Die zweite Möglichkeit zur Veränderung der effektiven Benutzer-ID ist der explizite Aufruf einer Systemfunktion. Sie ermöglicht die effektive Benutzer-ID so zu verändern, dass sie den Wert der realen Benutzer-ID erhält oder den Wert der effektiven Benutzer-ID, die der Prozeß bei seiner Aktivierung vom parent process geerbt hat.

4.4 PROZEßSYNCHRONISATION

In modernen Multitasking-Betriebssystemen teilen sich laufende Prozesse die vorhandenen Ressourcen. Bei gleichzeitigem Zugriff zweier Prozesse auf eine Ressource kann es zu schwer zu lokalisierenden Fehlern kommen. Wegen der vorhandenen Abhängigkeiten müssen Kooperierende Multitaskingprozesse miteinander synchronisiert (koordiniert) werden. Es lassen sich zwei Klassen von Abhängigkeiten unterscheiden:

- **Zeitkritisch** Die Prozesse konkurrieren um die Nutzung gemeinsamer exklusiver Ressourcen
- **Kritisch** Die Prozesse sind voneinander datenabhängig.

4.5 PROZESSKOMMUNIKATION

Eine intelligente und fehlerfreie Synchronisierung paralleler Prozesse ist ein wesentlicher Faktor für ein stabil arbeitendes Betriebssystem. Dazu ist eine effektive Kommunikation zwischen den Prozessen notwendig. Mögliche Arten der Kommunikation sind:

- **über gemeinsamen Speicherbereichen.** Prozesse können gemeinsame Datenbereiche, Variablen etc. anlegen und gemeinsam nutzen.
- **über gemeinsame Dateien.** Prozesse schreiben in Dateien, die von anderen Prozessen gelesen werden.

- **über Pipes.** Das sind direkte Datenkanäle zwischen zwei Prozessen. Ein Prozeß schreibt Daten in den Kanal und ein Anderer liest die Daten wieder aus.
- **über Signale.** Signale sind Ereignisse, die einen Software Interrupt erzeugen.
- **über Nachrichten** vom Betriebssystem verwaltet. Sie stellen eine für die Prozesse gemeinsam nutzbare Transportmittel (z.B. Mail) zur Verfügung. Auf diese greifen die Prozesse über bestimmte Systemaufrufe zu.
- **über Streams.** Streams sind Pipes Kommunikation über Rechnernetze. Logisch gesehen haben Streams dieselbe Aufgabe wie die lokalen Pipes.
- **über Prozedurfernaufrufe** (remote procedure call). Ein Prozeß ruft eine in einem anderen Prozeß angesiedelte Prozedur auf (also über seine Adreßgrenzen hinweg). Besonders für Client-Server-Beziehungen geeignet.
- **Ausgelöst vom Benutzer** (z.B. Eingabe)
- **Ausgelöst durch Programmfehler**
- **Ausgelöst durch andere Prozesse**

4.6 PROZESSOPERATIONEN

Es gibt einige Operationen die das Betriebssystem zur Steuerung und Kontrolle von Prozessen ausführen kann.

- Erzeugen eines Prozesses (z.B. Laden eines Programms)
- Allokieren des benötigten Speichers
- Prozess-ID vergeben
- Anlegen eines PCB in der Prozeßtabelle
- Reservieren der benötigten Ressourcen
- Vergabe einer Priorität
- Löschen eines Prozesses
- Löschen aller Einträge aus den Systemtabellen
- Freigabe von Speicher und Ressourcen (z.B. Dateien schließen)
- Löschen aller Abkömmlinge (Kindp., Enkelp., usw.)
- Suspendieren eines Prozesses.

- Wiederaufnehmen eines suspendierten Prozesses
- Blockieren eines Prozesses
- Aufwecken eines blockierten Prozesses
- CPU an einen Prozeß zuteilen
- Priorität eines Prozesses ändern

5 SPEICHERVERWALTUNG

Die Speicherverwaltung ist ein Grundbestandteil eines Betriebssystems und daher meistens vollständig im Kernel enthalten.

Seit der Zeit der ersten digitalen Rechnemaschinen bis heute kann beobachtet werden, dass der Bedarf an Arbeitsspeicher immer größer ist als verfügbare. Fast immer ist der RAM Speicher zu klein, um ein großes Programm oder mehrere Programme gleichzeitig aufzunehmen. Im Lauf der verschiedenen Rechnergenerationen hat man eine Reihe von Lösungen für dieses Problem entwickelt.

In moderneren Systemen gehen dabei von der Tatsache aus, dass zu einem Zeitpunkt (jetzt) nur auf wenige Speicherplätze tatsächlich zugegriffen wird, während die anderen nur für einen späteren Zugriff bereitstehen. Eine Erweiterung der Speicherkapazität unter Zuhilfenahme externer Massenspeicher, von denen bei Bedarf die benötigten Informationen in den Arbeitsspeicher geladen werden, ist eine Lösung. Dann muss die Speicherverwaltung in der jeweiligen Situation festlegen, welche Programmabschnitte geladen werden sollen und welche auf dem Hintergrundspeicher bleiben können.

5.1 SPEICHERVERWALTUNGSPRINZIPIEN

5.1.1 DIREKTE ADRESSIERUNG

In vielen embedded Systems wird nur ein Prozess zur gleichen Zeit ausgeführt. Dieser Prozess hat dann uneingeschränkten Zugriff auf den physischen Arbeitsspeicher und kann diesen direkt adressieren. Eine Verwaltung des Speichers ist in diesen Computern trivial und besteht darin, die angeforderte Adresse über den Datenbus zur Verfügung zu stellen.

- In **Dedizierten Systemen** werden die Programme so designt, dass sie immer in einem genau festgelegten Speicherbereich arbeiten.
- In **Universellen Systemen** übernimmt der Job-Monitor die Verwaltung der Job-Warteschlange. Ein Lader lädt die Benutzerprogramme zusammen mit einer

Programmbibliothek aus einem Hintergrundspeicher. Da alle Adressen der geladenen Programme bereits im Objektcode als absolute Adressen vorliegen, nennt man einen solchen Lader auch Absolutlader.

Sind die zu ladenden Daten von flexibler Größe (z.B. Logdateien), muss immer der Maximalaufwand an Speicher berücksichtigt werden. Das führt zu einer uneffektiven Ausnutzung des Arbeitsspeichers.

5.1.2 RELOCATION

Diese flexiblere Lösung besteht darin, die Bibliotheksroutinen so aufzubereiten, dass die Festlegung der von ihnen benötigten Speicherplätze bis zum Ladezeitpunkt aufgeschoben werden kann. Man benötigt dazu einen Relocating Loader, der dann auch gleich zum Laden der Benutzerprogramme verwendet wird.

Die Kunst des Programmierers besteht darin, die Adressteile der Maschinenberfehle als absolute bzw. relative Adressen zu kennzeichnen. Somit besteht die Aufgabe des Loaders darin, zu den relativen Adressangaben nur noch einen konstanten Größe (Offset) zu addieren. Wenn sich das Programm im Hauptspeicher befindet, kann es nicht mehr verschoben werden.

Eine weitere Steigerung der Flexibilität ist dann gegeben, wenn die Umsetzung in effektive Adressen nicht zum Ladezeitpunkt, sondern erst unmittelbar bei Ausführung des Befehls vorgenommen wird. Voraussetzung dafür ist ein Prozessor, der entsprechende Adressierungsarten der Befehle erlaubt. Dazu muss ein Adressrechenwerk im Prozessor vorhanden sein, das entweder den Inhalt eines Basisadressregisters oder den Inhalt des Program-Counters addiert (position independent code, relocatable). Die Adressangaben bei den Befehlen werden also nicht mehr vom Lader modifiziert, der Ladevorgang läuft wesentlich schneller ab. Das Programm kann nach dem Laden noch im Speicher verschoben werden.

5.1.3 OVERLAY

Schon seit den ersten Computersystemen mit Arbeitsspeicher stießen Programmierer zum auf das Problem, dass die Programme zu groß für den verfügbaren Speicher wurden. Sie lösten dieses Problem, indem sie die Programme in mehrere Overlays (Teile) spalteten.

Zunächst wurde Overlay0 ausgeführt, das dann, sobald es fertig war, das nächste Overlay1 aufrief, welches dann Overlay2 lud, usw.

Die einzelnen Overlays wurden auf der Festplatte gespeichert und nach Bedarf vom Betriebssystem dynamisch ein- und ausgelagert. Obwohl die Overlays vom Betriebssystem verwaltet wurden, musste der Programmierer das Programm selbst aufteilen. Große Programme in kleine, modulare Teile aufzuspalten, war eine zeitaufwändige und langweilige Arbeit. Diese Speicherverwaltung wurde als Overlay-Verwaltung bekannt.

5.1.4 VIRTUELLER SPEICHER

Lässt man hingegen das Betriebssystem die Arbeit der Teilung zukommen, spricht man vom virtuellen Speicher. Die Grundidee dahinter war, zu erlauben, dass der Programmcode, die Daten und der Stack zusammen größer sind als der verfügbare Hauptspeicher.

Das Betriebssystem hält die Teile des Programms, die gerade gebraucht werden, im Hauptspeicher und den Rest auf der Festplatte.

Man spricht von virtueller Speichertechnik, wenn der Speicheradressraum, des Prozessors getrennt ist vom realen Adressraum des Arbeitsspeichers, in dem sich das Programm bei der Abarbeitung befindet.

- **logischer Adressraum**: Der Adressraum, auf den sich die Programmbefehle beziehen. Er kann deutlich größer als der reale Arbeitsspeicher sein.
- **physischer Adressraum**: Der Adressraum des realen Arbeitsspeichers.

So können Programme können unabhängig vom physischen Adressraum geschrieben werden.

5.1.4.1 PAGING

Die meisten virtuellen Speichersysteme benutzen als Technik zur Speicherverwaltung das Paging. Bei diesem Verfahren ist der virtuelle Adressraum, der von den Programmen benutzt wird, in der Regel größer als der physikalischer Adressraum. Die virtuelle Adresse des Programms wird durch die MMU (Memory Management Unit) auf die physikalische Adresse umgerechnet.

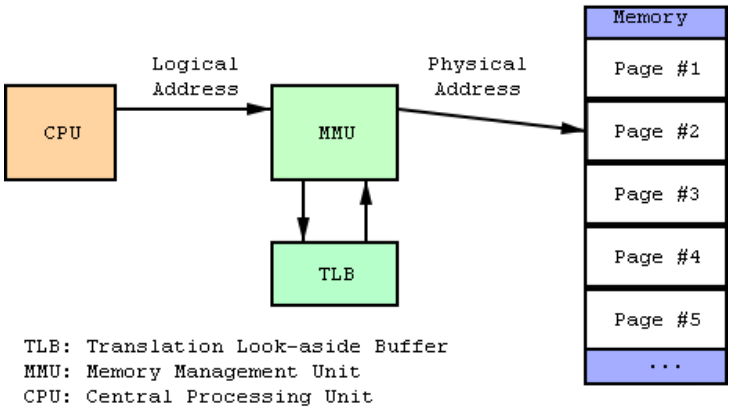


Abbildung 14 Funktionsweise Paging durch eine MMU

Abbildung 12 externer MMU Prozessor

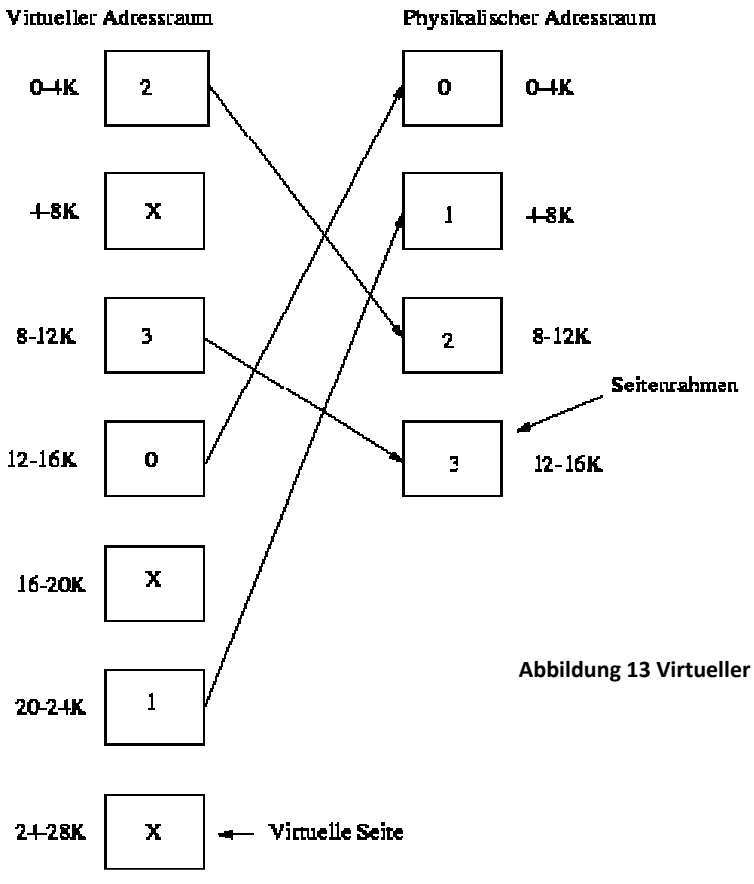


Abbildung 13 Virtueller Addressraum

Der virtuelle Adreßraum ist in Einheiten, die Seiten (Pages) unterteilt. Die dazu korrespondierenden Einheiten im physikalischen Adreßraum heißen Seitenrahmen

5.1.4.2 SEGMENTIEREN

Beim segmentieren wird der logische Adressraum in Abschnitte variabler Größe gemäß den logischen Einheiten des Programms (Unterprogramme, Datenbereiche, etc.) unterteilt. Die Abschnitte nennt man Segmente.

Die logische Adresse besteht aus 2 Teilen:

- **Segment-Nummer** (höchstwertiger Teil)
- **Wortadresse** (Offset, Adresse relativ zum Segmentanfang)

Für die in den Arbeitsspeicher geladenen Segmente ist in einer Segmenttabelle die jeweilige reale Anfangsadresse des Segments festgehalten (Basisadresse des Segments). Im Multiprogramming-Betrieb werden für die verschiedenen Jobs meist jeweils eigene Tabellen bereitgestellt damit mit den Segment-Nummern verschiedener Jobs keine Verwechslung möglich ist. Somit muss vor dem Segmenttabellenzugriff noch der Inhalt eines jobspezifischen Segmenttabellenregisters zur Segmentnummer addiert werden. Bei jedem Umschalten der Jobs muss deshalb auch das Segmenttabellenregister umgeladen werden. Die Segmenttabelle enthält auch Angaben über die Größe der Segmente. Dadurch können

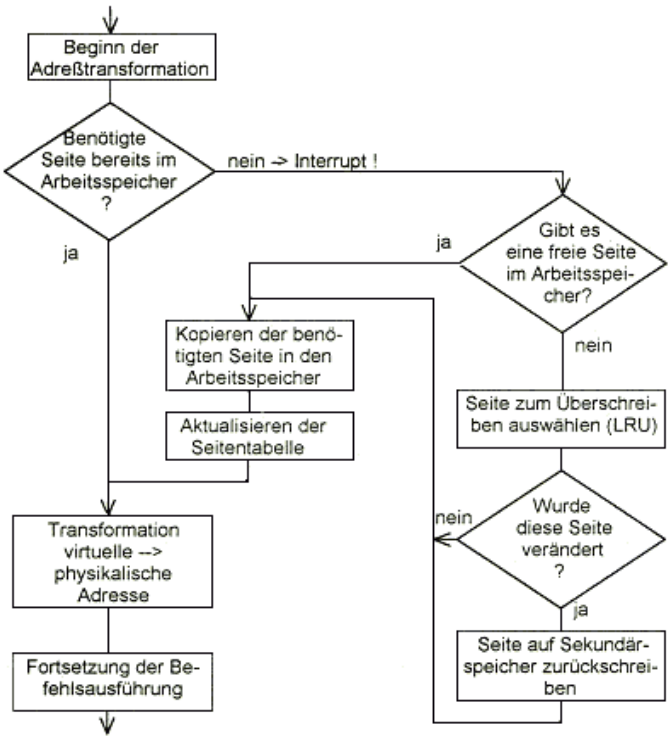


Abbildung 15 Segmentwechsel

fehlerhafte Zugriffe, die aus dem Segment herausführen, erkannt und verhindert werden.

6 EINGABE / AUSGABE

E/A-Geräte werden in der Regel in das Dateisystem eingebunden und aus Benutzersicht wie Dateien behandelt mir eventuell abweichenden Eigenschaften wie:

- Auf reine Ausgabegeräte kann nur geschrieben werden
- Von reinen Eingabegeräten kann nur gelesen werden
- Bei kombinierten E/A-Geräten kann gelesen/geschrieben werden

E/A-Geräte können grob in zwei Kategorien eingeteilt werden:

- **Block Devices** (blockorientierte Geräte), z. B. Festplatte und Band. Es verarbeitet Daten nur in Form von Datenblöcken fester Länge
- **Character Devices** (zeichenorientierte Geräte), z. B. Terminal oder serielle Schnittstelle. Es verarbeitet einen Datenstrom aus einzelnen Bytes.

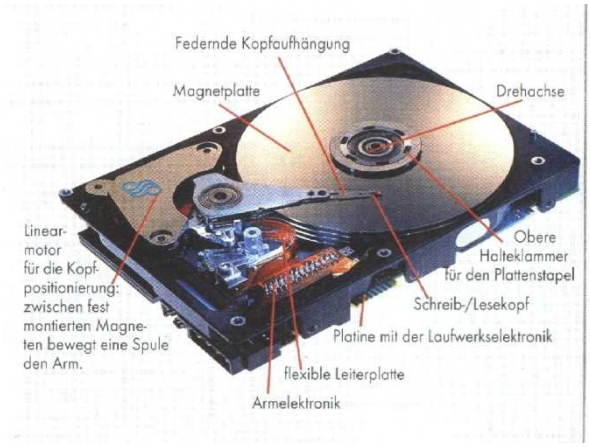


Abbildung 17 Block Device Festplatte

Abbildung 16 Character Device DEC Terminal vt52

Einige Geräte lassen sich nicht so genau einordnen. Zur Ansteuerung der Geräte ist systemnahe und z. T. hardwareabhängige Software des BS notwendig. In der Regel erhält jedes Gerät einen fest vorgegebenen Namen, unter dem es dann wie eine Datei angesprochen werden kann. Je nach Typ können Geräte gemeinsam von allen Prozessen verwendet werden (z. B. Platte) oder sie sind exklusiv für einen Prozess reserviert (z. B. Plotter). Die Software zur Ansteuerung von Geräten lässt sich in zwei Ebenen strukturieren:

Eingabe / Ausgabe

- Gerätetreiber sind Betriebssystem-Programmteile für den geräteabhängigen Code. Hier sind alle Prozeduren vereinigt, die hardwarespezifisch sind.
- Geräteunabhängige Software des Betriebssystems. In dieser Schicht wird die Schnittstelle zum Dateisystem realisiert. Sie enthält eine einheitliche Schnittstelle zum Treiber, den Gerätenamen, Pufferung der Daten, Schutz der Geräte, Vergabe exklusiver Geräte und Fehlerbehandlung.

6.1 GERÄTESTEUERUNG

Die Peripheriegerätesteuerung erfolgt auf der Hardware-Ebene über den Austausch von Statussignalen und Daten zwischen den angeschlossenen Geräten und dem Prozessor. Die Programme zur Steuerung von Peripheriegeräten werden Treiber (driver, handler) genannt. Die Komplexität der Steuerungsaufgaben ist weitgehend vom Geräteverhalten abhängig.

Alle drei Methoden des Signal- und Datenaustausches können zum Einsatz kommen (programm gesteuert, interrupt gesteuert, DMA).

6.1.1 PROGRAMM GESTEUERTE E/A

Im einfachsten Fall sind alle Grundfunktionen der Ein-/Ausgabe vollständig in die Treiberprogramme integriert. Bei der Durchführung einer Datenübertragung oder einer Steuerfunktion muss der Prozessor dann bis zu deren Beendigung explizit warten, bevor er die Bearbeitung weiterer Programmschritte aufnehmen kann.

6.1.2 UNTERBRECHUNGSGESTEUERTE E/A

Damit der Prozessor nicht durch unnötige Wartezeiten und Statusabfragen zeitlich belastet wird, erfolgt die Gerätesteuerung i. a. unter Verwendung von Unterbrechungen, wie zum Beispiel ein Tastendruck am Terminal erzeugt eine Unterbrechung. Das entsprechende Interrupt-Service-Programm liest das Zeichen vom Empfangsport der seriellen Schnittstelle und speichert es in einem Zwischenpuffer.

Eingabe / Ausgabe

Falls das Betriebssystem auf die Eingabe eines Kommandos wartet, wird gleich geprüft, ob das Eingabeende CR (Carriage Return) eingetroffen ist.

Falls das CR erkannt wird, veranlasst das Interrupt-Service-Programm eine Meldung (z. B. per Mailbox) an ein hierarchisch höher liegendes Auswerteprogramm, das den Inhalt des eingegebenen Textes analysiert, um - wieder eine Ebene höher - z. B. die gewünschte Betriebssystem-Funktion zu starten. Die Unterbrechungsignale verschiedener Geräte sollten unterschiedliche Prioritäten haben, um die dringlichen Aufgaben vorrangig bearbeiten zu können.

6.1.3 DMA-Betrieb

Um den Prozessor von ständig zu wiederholenden Formen der Datenübertragung zu entlasten, wurde das Verfahren des direkten Speicherzugriffs DMA (direct memory access) entwickelt. Die Datenübertragung zwischen dem Hauptspeicher und einem Peripheriegerät wird vollständig von der DMA-Steuerung durchgeführt und die Transferrate wird durch die Geschwindigkeitsanforderungen des beteiligten Geräts oder durch die Länge der Speicherzyklen oder die Übertragungskapazität des Busses bestimmt.

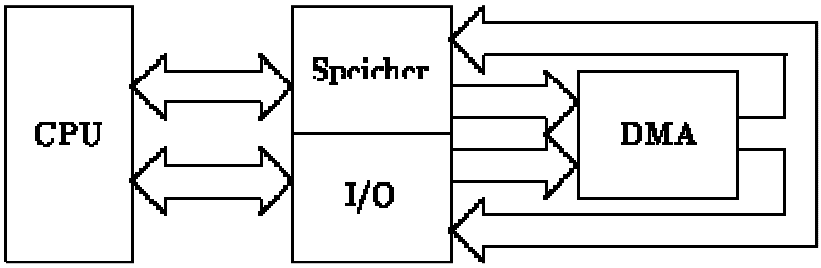


Abbildung 18 DMA

7 DATEISYSTEME

Viele Speichergeräte arbeiten blockweise, indem sie Daten als Blöcke fester Größe in Bereiche eines Datenträgers speichern. Die Hardware bietet somit dem System eine Menge von Blöcken an, die eindeutig adressierbar sind. Das Betriebssystem verwaltet alle diese Blöcke in einem Dateisystem und enthält Routinen zum Lesen und Schreiben von Blöcken für die jeweils verwendeten Gerätetypen, die sogenannten Gerätetreiber.

Das vom Betriebssystem verwaltete Dateisystem erspart dem Benutzer so das Hantieren mit lästigen Details, wie z.B. das Ausrichten des Schreib-/Lesekopfes einer Festplatte beim Zugriff auf eine Datei. Für einen Benutzer ist es viel einfacher und intuitiver, seine Daten in Dateien zu organisieren.

7.1 DATEIORGANISATION

Dateien werden in Dateisystemen organisiert. Hier gibt es lineare und hierarchische Dateisysteme.

7.1.1 LINEARE DATEISYSTEME

Die historisch ersten Dateisysteme waren lineare Dateisysteme auf Lochband oder Lochkarte sowie die noch heute für die Sicherung von Daten eingesetzten Magnetbandsysteme.

7.1.2 HIERARCHISCHE DATEISYSTEME

Jede Datei hat einen Namen und einen Inhalt. Dieser kann aus einer beliebigen Folge von Bytes bestehen. Das Betriebssystem muss eine Übersetzung zwischen den von der Hardware angebotenen Blöcken und den vom Benutzer gewünschten Dateien gewährleisten. Das Dateisystem, als zuständiger Teil des Betriebssystems, verwaltet eine Datei als Folge von Blöcken. Selbst wenn sie nur ein Byte enthält, verbraucht eine Datei

mindestens den Speicherplatz eines Blockes. Die Dateien eines Dateisystems werden in Ordnern oder Verzeichnissen organisiert. In einem Ordner oder auch Verzeichnis findet sich für jede in ihm enthaltene Datei ein Eintrag mit Informationen:

- **Dateiname** (dazu gehört ggf. auch die **Erweiterung**)
- **Dateityp** (Normaldatei, ausführbare Datei, Verzeichnisdatei)
- **Länge** in Bytes
- **zugehörige Blöcke** (meist reicht ein Verweis auf den ersten Block der Datei)
- **Zugriffsrechte** (Besitzer, ggf. Passwort, Schreib- und Leserechte)
- **Datum** (Erstellung, Änderung)

Unter Windows ist jedes Laufwerk die Wurzel eines eigenen Dateibaums, die mit einem eigenen Laufwerksbuchstaben benannt wird. Unter UNIX sind die Dateisysteme aller Laufwerke Unterbäume eines globalen Dateibaums, dessen Wurzel sich root (/) nennt.

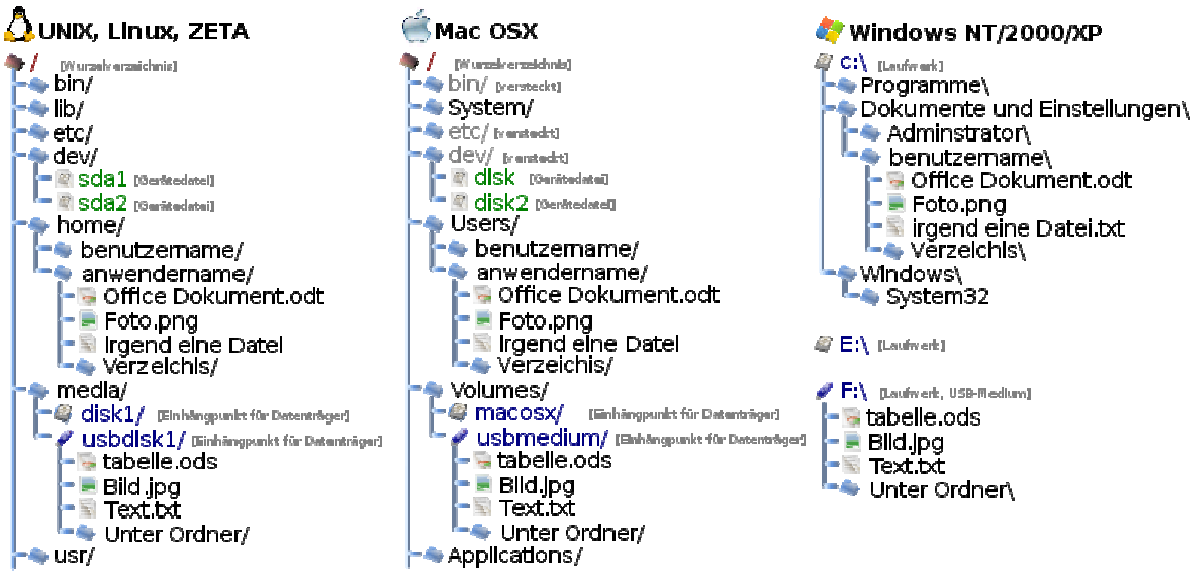


Abbildung 19 Dateibäume im Vergleich

Weiter muss das Betriebssystem Pseudodateien verwalten.

- belegte Blöcke
- freie Blöcke
- unzuverlässige Blöcke (werden nicht mehr genutzt)

Während der Bearbeitung der Dateien ändern sich die Listen dynamisch.

Es gibt ziemlich viele verschiedene Dateisysteme. Jedes ist für eine bestimmte Art von Daten, ein Betriebssystem, ein Datenmedium oder eine Aufgabe optimiert. Viele Dateisysteme stellen auch den Nachfolger aktueller Nachfolger dar.

Tabelle 1 Wichtige Dateisysteme

Dateisystem	Veröffentlicht	Entwickler	Beriebssystem
btrfs	2007	Oracle	Linux
ext2	1993	Rémy Card	Linux
ext3	2001	Stephen Tweedie	Linux
ext4	2008		Linux
JFS	1990	IBM	AIX
ReiserFS	2001	Namesys	Linux
Reiser4	2004	Namesys	Linux
XFS	1994	SGI	Unix
FAT	1980	Microsoft	MS-DOS
NTFS	1993	Microsoft	Windows
ZFS	2006	Sun	Solaris
ISO 9660	1987		CD/DVD

Eine ausführlichere Liste kann unter http://de.wikipedia.org/wiki/Liste_von_Dateisystemen

Eingesehen werden.

7.2 Implementierung von Dateien

7.2.1 Kontinuierliche Allokation

Bei der kontinuierlichen Allokation wird eine der Größe der Datei entsprechende Anzahl von Blöcken auf dem Speichermedium belegt. Dazu wird eine Tabelle angelegt (File allocation table, FAT), in der angegeben ist, ob ein Block frei, belegt oder schadhaft ist. Für einen freien Block wird eine 0 eingetragen, schadhafte Blöcke werden in MSDOS mit dem Wert FFF7 gekennzeichnet. Für einen belegten Block wird in der Tabelle angegeben, wo der nächste zu der Datei gehörende Block zu finden ist. Das Ende einer solchen Kette wird mit einem speziellen Zeichen gekennzeichnet (MSDOS: FFFF).

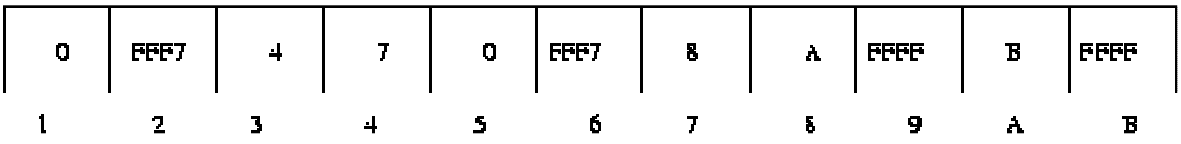


Abbildung 20 Kontinuierliche Allokation mit FAT

7.2.2 Indirekte Adressierung mittels INodes

Anders als bei einer FAT werden z.B. bei UNIX sogenannte INodes (Index Nodes) benutzt, um die Blöcke, die zu einer Datei gehören, zu adressieren. In dem INode sind außerdem Informationen über den Eigentümer, Zugriffsberechtigungen und andere Attribute enthalten.

Ein INode unter UNIX hat 13 Einträge, die zur Adressierung verwendet werden. Die ersten zehn Einträge zeigen direkt auf die ersten zehn Blöcke der Datei. Damit ist es möglich, Dateien bis zu 10 KByte direkt zu erreichen. Der elfte Eintrag zeigt auf einen weiteren INode, mit dem weitere Blöcke erreicht werden. Eintrag zwölf enthält - zur Adressierung größerer Dateien - die Adressen von weiteren INodes, welche dann die Adressen der Blöcke enthalten. Der letzte Eintrag enthält wiederum Adressen von INodes, die dem Typ aus Eintrag zwölf entsprechen.

Es zeigt sich, daß dieses System umso effektiver ist, je kleiner die Dateien sind. Der große Teil der Dateien ist jedoch kleiner als 10 KByte und kann daher direkt referenziert werden.

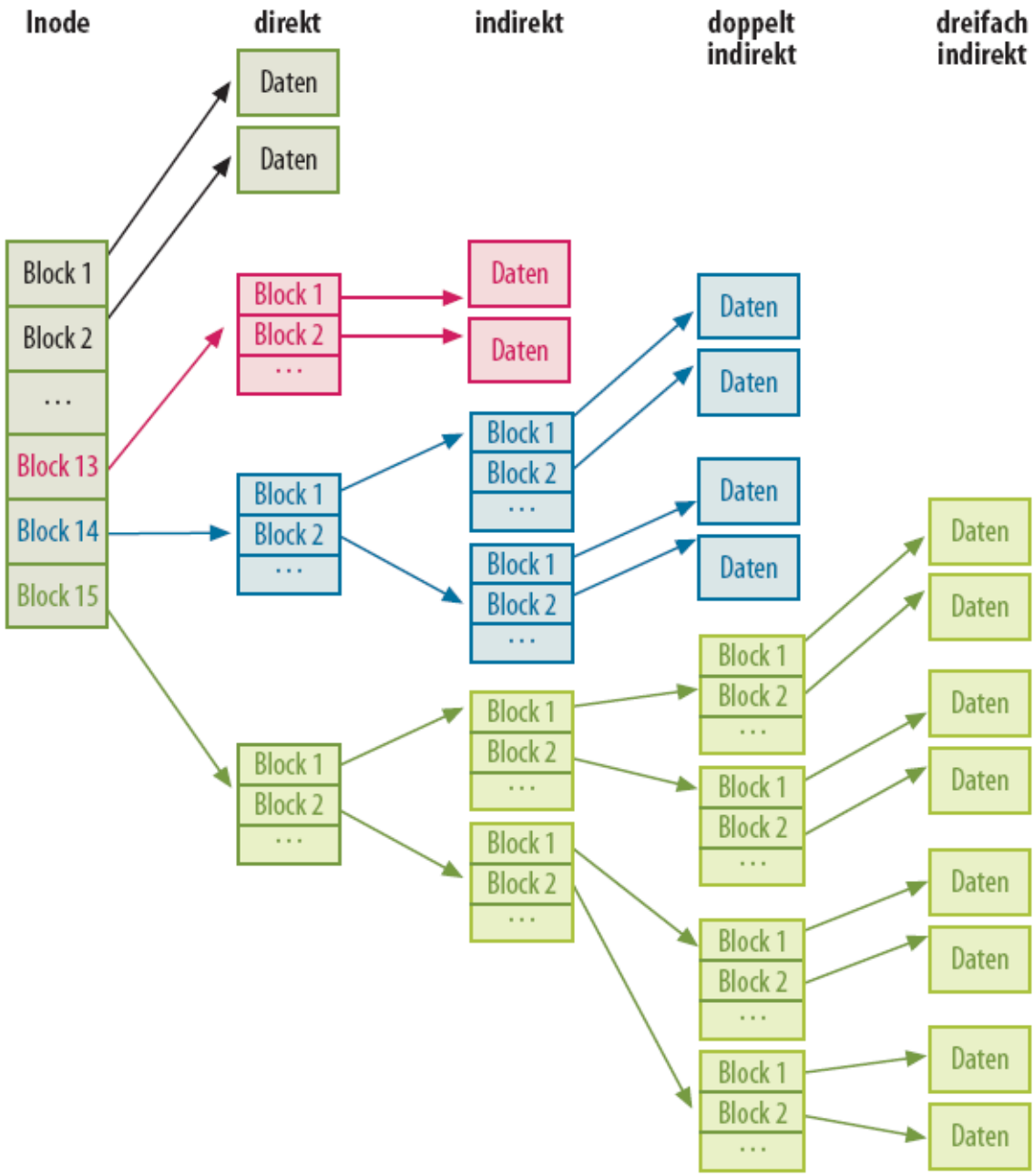


Abbildung 21 Indirekte Adressierung mittels Inodes

7.2.3 NTFS

NTFS (New Technology File System) ist mit einem Zugriffsschutz ausgestattet und hat auch keine Größenbegrenzung mehr. Ein Logfile (Journal) wird verwendet, um nach einem Systemausfall Daten rekonstruieren zu können. NTFS besitzt nur Dateien. Analog zu den I-Nodes beim Unix gibt es beim NTFS eine Master File Table, in welcher jede Datei einen Eintrag besitzt. Zusammenhängende Bereiche (extents) werden als B-Baum organisiert. Bei kleinen Dateien bzw. Verzeichnissen werden alle Attribute (incl. der Daten) innerhalb des MFT-Eintrages abgelegt (bis zu 1 bis 4 KB). Ein Eintrag in der MFT benötigt einen oder mehrere Sätze der MFT (Satzlänge ist konfigurierbar). Bei großen Dateien enthält der MFT-Eintrag den Wurzelknoten eines B-Baums, dessen "Blätter" die Verweise auf die zusammenhängenden Dateibereiche (Extent oder Lauf) enthalten.

Ähnlich wie die Inode-Tabelle beim UNIX-System stellt die MFT ein flaches Dateisystem dar. Über die Verzeichnisse wird darauf die bekannte Baumstruktur definiert.

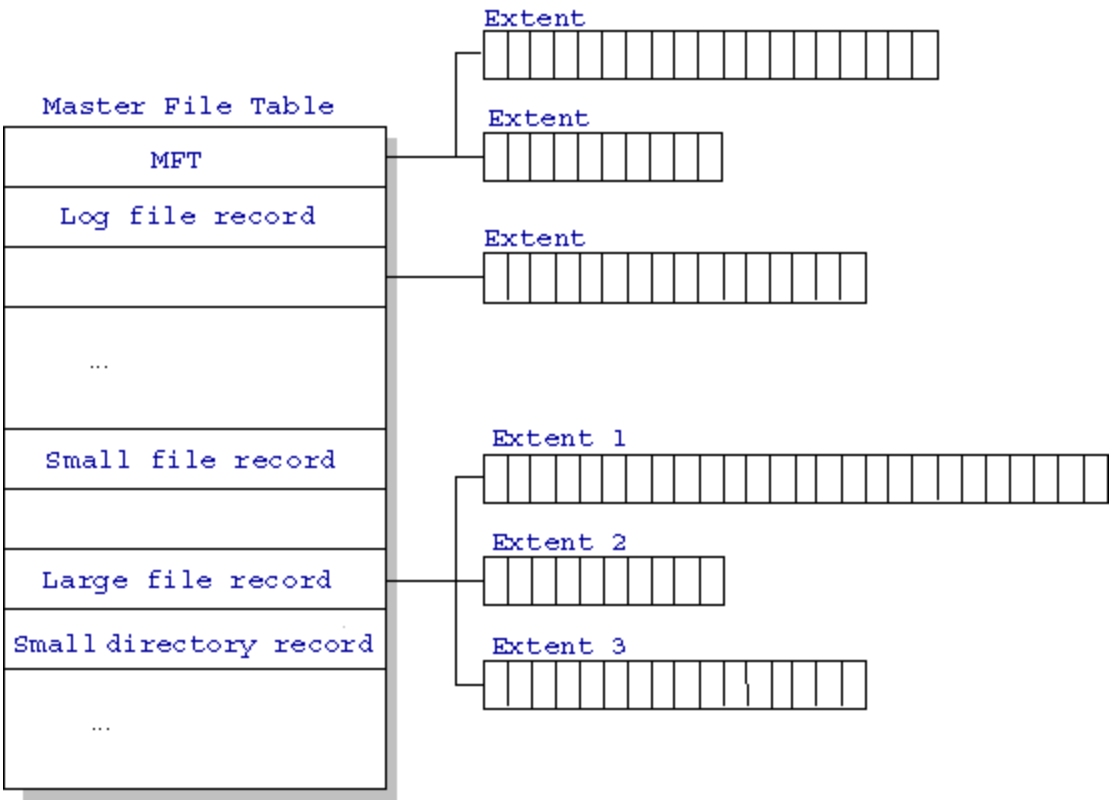


Abbildung 22 Aufbau eines NTFS

Jede Datei ist eindeutig über eine ID identifizierbar. Diese ID ist die Satznummer ihres Eintrages in der MasterFileTable (48 Bit). Zusätzlich wird eine Folgenummer (16 Bit) angehängt, die bei jedem Bezug auf den MFT-Eintrag (z.B. beim Öffnen der Datei) um 1 erhöht wird (für Konsistenzüberprüfungen)

Die Spezialdateien von NTFS

- MasterFileTable (MFT)
- MFT2: enthält die ersten 16 Einträge der MFT als Backup
- Eine Logdatei (Daten über Transaktionen zum Wiederherstellen der Daten- bzw. der Dateisystemkonsistenz)
- Datenträgerdatei enthält den Namen des Datenträgers, Versionsnummer des Dateisystems und eventuellen Verdacht auf Inkonsistenz
- Wurzelverzeichnis
- Cluster-Bitmap-Datei für belegte und freie Cluster des Datenträgers
- Bootdatei mit dem Startcode für NT
- BadClusterDatei, welche Verweise auf defekte Cluster enthält

7.3 Gemeinsame Nutzung von Daten

Wenn mehrere Benutzer die gleiche Datei verwenden wollen, kann es nützlich sein, wenn jeder Benutzer die Datei aus einem seiner Kataloge ansprechen (ohne den absoluten Pfad zu verwenden) und u. U. auch einen eigenen Namen für die Datei vergeben kann (bei Namenskonflikt mit eigenen Dateien).

Es wird dann ein Verweis auf die Datei installiert, ein Link. Durch die Einführung von Links dürfen die durch die Datei belegten Plattenblöcke nicht im Katalog gespeichert werden, da sonst alle Katalogeinträge, die auf die Datei verweisen, bei jeder Änderung an der Datei aktualisiert werden müssten. Die Lösung kann auf zwei Arten erfolgen:

- Die Verzeichnisse verweisen auf die Datei-Datenstruktur, die gesondert gespeichert ist (i-nodes bei UNIX)

- Es wird für ein Link eine spezielle Datei angelegt, die den absoluten Pfadnamen auf die gelinkte Datei enthält (symbolic linking).

Die erste Lösung kann in Mehrbenutzersystemen zu Inkonsistenzen führen. Angenommen Benutzer A ist Eigentümer einer Datei und gestattet Benutzer B, ein Link auf diese Datei zu setzen. Zu einem späteren Zeitpunkt wird der Benutzer A aus dem System gelöscht (und damit auch seine Dateien). Nun gibt es über das Link von B eine Datei, zu der es keinen Eigentümer mehr gibt. Die zweite Lösung vermeidet solche Inkonsistenzen, erfordert beim Dateizugriff jedoch einen höheren Verwaltungsaufwand (aus der Link-Datei den Pfad lesen und erst damit die Datei eröffnen). Ist die Datei nicht mehr vorhanden, erfolgt eine Fehlermeldung.

8 VERTEILTE SYSTEME

Ein Verteiltes System ist nach der Definition von Andrew Tanenbaum ein Zusammenschluss ***unabhängiger*** Computer, der sich für den Benutzer als ein einzelnes System präsentiert.

Was sind als demnach keine verteilten Systeme

- Parallelcomputer mit zentralem Speicher
- Multicore Prozessoren
- Homogene Rechnercluster

Peter Löhr definiert es etwas grundlegender als „eine Menge interagierender Prozesse (oder Prozessoren), die über keinen gemeinsamen Speicher verfügen und daher über Nachrichten miteinander kommunizieren“. Nach dem Grad der Trennung der einzelnen Systeme lassen sich zwei Gruppen bilden:

- **LCDS** (loosely coupled distributed systems) sind Rechner die nur über ein gemeinsames Netz miteinander kommunizieren.
- **TCDS** (tightly coupled distributed systems). Hier haben die einzelnen CPU's einen gemeinsamen Speicher den sie sich teilen und darüber miteinander kommunizieren können. Diese Verbindung heißt auch Prozessorknoten.

8.1 KLASSIFIZIERUNG

Um Prozesse auf mehrere Systeme zu verteilen gibt es drei grundlegende Vorgehensweisen. Jede dieser Lösungen hat Vor- und Nachteile

8.1.1 CLIENT-SERVER

Das Client-Server-Modell (oder Terminal-Server-Modell) ermöglicht, Aufgaben innerhalb eines Netzwerkes zu verteilen. Die benötigten Programme werden in Clients und Server

unterteilt. Der Client fordert eine Aufgabe beim Server an (Betriebsmittel wie Rechenleistung oder Datenbank). Der Server beantwortet die Anforderung und stellt die Betriebsmittel bereit. Meist sind Client-Server Systeme Softwarelösungen die auf eigenständigen Systemen aufbauen, wogegen Terminal-Server-Systeme auch Hardwarelösungen darstellen können, bei denen das Terminal nicht ohne Server funktionsfähig ist.

8.1.2 VERTEILTE ANWENDUNGEN

Verteilte Anwendungen sind komplexe Anwendungsprogramme, die in einem verteilten System, also auf mehreren Rechnern/Prozessoren, abläuft. Entstehen kann eine verteilte Anwendung durch horizontale Schnitte im Softwareschichtenmodell, so dass die Aufgabe des Gesamtsystems auf einzelne Softwarekomponenten aufgeteilt wird. Zur Erfüllung der Gesamtaufgabe müssen alle Komponenten der Anwendung mitwirken und untereinander kommunizieren. Für den Client erscheint das System meist wie ein einziges (transparent). Zwischen den Komponenten existieren definierte Schnittstellen.

Verteilte Anwendungen werden unter anderem für Verteiltes Rechnen verwendet.

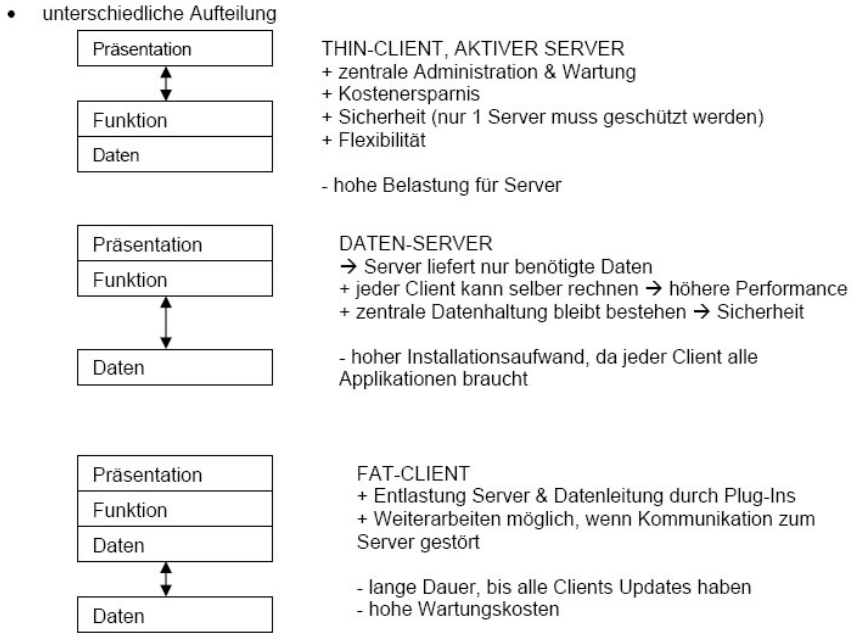


Abbildung 23 Verteilte Anwendungen

8.1.3 VERTEILTES BETRIEBSSYSTEM

Das Betriebssystem selbst ist für Benutzer und Anwendungen nicht sichtbar, verteilt. Hierzu zwei Beispiele

- **Amoeba** (engl. für Amöbe) ist ein verteiltes Betriebssystem, das von Andrew S. Tanenbaum und seinen Mitarbeitern an der Freien Universität Amsterdam entwickelt wurde. Ziel des Projekts war, jedem Benutzer die Illusion einer eigenen Maschine zu geben, obwohl das System auf vielen Rechnern verteilt ist, die eventuell auch weit entfernt voneinander, etwa in verschiedenen Ländern, stehen können. Die Programmiersprache Python wurde ursprünglich für Amoeba entwickelt.

- **Banyan VINES** (Virtual Integrated Network Service) ist ein Netzwerkbetriebssystem, welches mit proprietären Netzwerkprotokollen arbeitet, die vom *Xerox Network System (XNS)* abgeleitet sind. VINES implementiert eine Architektur nach dem Client-Server-Modell. Es kam 1984 heraus und genoss bis Mitte der 90er Jahre eine gewisse Verbreitung. Als Alleinstellungsmerkmal wurde mit VINES ein Verzeichnisdienst - *StreetTalk* - eingeführt. Er kann als legitimer Vorläufer moderner Verzeichnisdienste bezeichnet werden. Das Grundsystem VINES/386 Network Operating System ist ursprünglich ein um entsprechende Netzwerkprotokolle erweitertes UNIX-Derivat. Die Protokolle lassen sich wie folgt in das ISO-OSI-Referenzmodell einordnen:

Tabelle 2 VINES Protokollstapel im OSI Modell

<table>
<tr><th>OSI-
Schicht</th><th colspan="4">VINES-Protokollstapel</th></tr>
<tr><td>7</td><td>File Services</td><td>Print Services</td><td>StreetTalk (Verzeichnisdienst)</td><td>andere Dienste</td></tr>
<tr><td>6</td><td colspan="4" rowspan="2">Remote Procedure Calls (RPC)</td></tr>
<tr><td>5</td></tr>
<tr><td>4</td><td colspan="2">InterProcess Communications (IPC)
Datagram</td><td colspan="2">Sequenced Packet Protocol (SPP)
Stream</td></tr>
<tr><td>3</td><td colspan="3">VINES Internetwork Protocol (VIP)</td><td>Address Resolution Protocol (ARP)
Routing Table Protocol (RTP)
Internet Control Protocol (ICP)</td></tr>
<tr><td>2</td><td colspan="4" rowspan="2">Media Access Protocols:
HDLC, X.25, Token-Ring, Ethernet</td></tr>
<tr><td>1</td></tr>
</table>

8.2 KOMMUNIKATIONSMODELLE

Der Austausch von Informationen und Daten innerhalb verteilter Systeme ist einer Regelung unterworfen, die komplexer ist, als bei nicht verteilten Systemen.

8.2.1 SYNCHRON

Nachrichten selbst haben bei diesem Modell keine Laufzeit. Dies wird in der Praxis durch die Synchrone Kommunikation erreicht bei dem sich die Prozesse beim Senden oder beim Empfangen von Daten zwingend synchronisieren und warten bis die Kommunikation abgeschlossen ist. So lange ist der Prozess blockiert. Wird sowohl beim Senden als auch beim Empfangen gewartet so entspricht das einem Rendezvous der beiden beteiligten Prozesse. Das Blockieren des Prozesses wird intern durch geeignete Mechanismen zur Prozesssynchronisation erreicht.

8.2.2 ASYNCHRON

Hier haben Prozesse nur den Zustand aktiv oder passiv. Nur ein aktiver Prozess versendet Nachrichten. Ein aktiver Prozess kann jederzeit passiv werden, wohingegen ein passiver Prozess nur durch eine Nachricht reaktiviert werden kann.

8.3 BEISPIEL FÜR VERTEILTE SYSTEME

Ein recht gutes Beispiel ist hier Google. Es Betriebt mehrere Rechnerzentren mit insgesamt weit mehr als 200.000 Computern mit verschiedenen Diensten

- Load Balancers
- Proxy Servers (caches)
- Web Servers
- Index servers
- Document servers

- Data-gathering servers
- Ad servers
- Spelling servers

Trotzdem ist die Arbeit mit Google für den Anwender völlig transparent. Fällt ein Google-Server aus, wird das vom Anwender nicht wahr genommen. Er muss auch nicht entscheiden von welchem Server seine Anfrage bearbeitet wird und welcher Server die Antwort sendet.

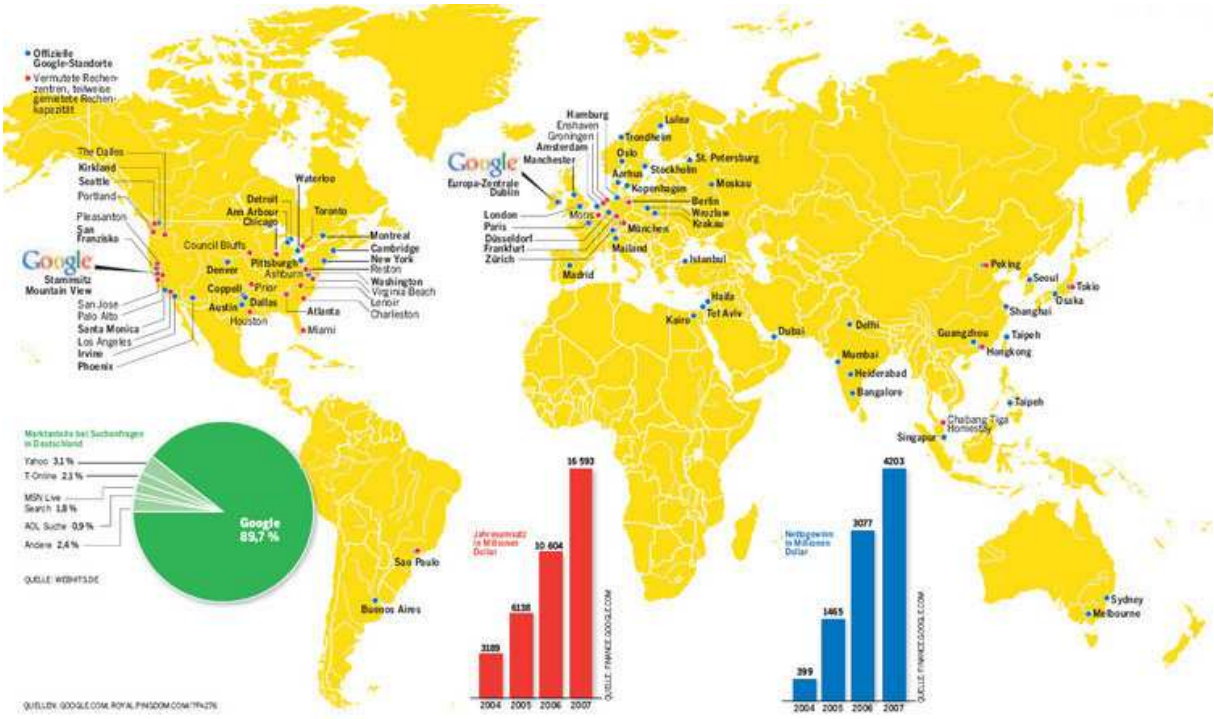


Abbildung 24 Google Standorte

9 CLUSTER COMPUTING

Das erste im Handel erhältliche Clusterprodukt war ARCnet, welches 1977 von Datapoint entwickelt wurde. Den ersten wirklichen Erfolg hatte die Firma DEC im Jahre 1983 mit der Vorstellung des Produktes VAXCluster für ihr Computersystem VAX. Das Produkt unterstützte nicht nur paralleles Rechnen auf den Clusterknoten, sondern auch die gemeinsame Nutzung von Dateisystemen und Geräten aller beteiligten Knoten.

Ein Cluster in der EDV bezeichnet eine Anzahl von vernetzten Computern, die von außen in vielen Fällen als ein Computer gesehen werden können. In der Regel sind die einzelnen Elemente eines Clusters untereinander über ein schnelles Netzwerk verbunden. Es gibt zwei grundlegende Erwartungen an einen Cluster

- Ausfallsicherheit und damit eine Hochverfügbarkeit
- Performance

9.1 HA CLUSTER

Der einfachste HA Cluster besteht aus genau 2 Mitgliedern, Clusternodes genannt. Beide Nodes sind unabhängig, es ist aber immer nur ein Node aktiv. Fällt der aktive Computer aus, übernimmt der bis dahin Passive dessen Aufgaben. Um aus zwei Computer einen HA-Cluster zu bauen ist kein verteiltes Betriebssystem nötig. Es müssen nur die dafür notwendigen Programme installiert werden. Da immer nur eine Maschine alle Daten verarbeitet, beschränkt sich die Kommunikation zwischen den Systemen auf zwei Bereiche.

- Heartbeat ist die Kommunikation zwischen den beiden Clusternodes. Über ein vom aktiven Node in genau definierten Zeitfenstern gesendetes Datenpaket kann der passive Node erkennen, dass seine Arbeit nicht benötigt wird. Erst wenn dieses Signal aus bleibt, übernimmt der passive Node die Funktion des Anderen.

- Clusterdevices sind Datenbereiche die von beiden Clusternodes verwendet werden können, damit beide Nodes immer über den gleichen Datenbestand verfügen können.

Wenn der aktive Node keine Heartbeat Pakete mehr sendet kann es trotzdem sein, dass er weiterhin erreichbar ist. Da aber der passive Node versucht die Dienste zu übernehmen kann dies zu unerwarteten Fehlern führen. Um dieser Split-Brain Problematik entgegen zu wirken sorgt der passive Node vor seiner Aktivierung dafür, dass sein Partner ab geschalten ist. Dazu wird STOHNIT (Shot the Other in the Head) verwendet.

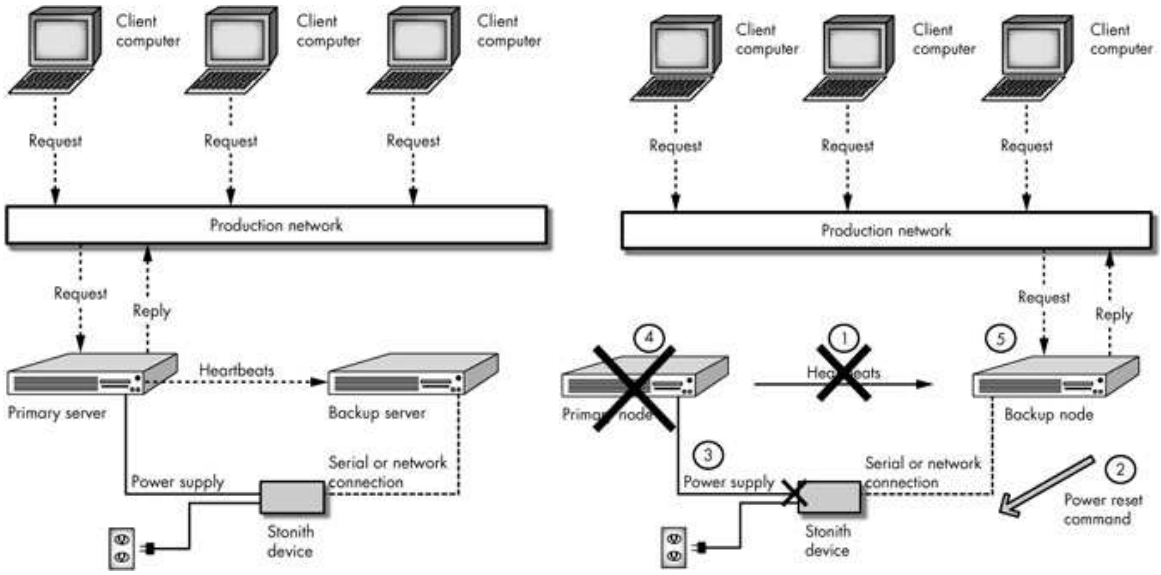


Abbildung 25 HA Cluster mit STOHNIT Device

9.2 HPC CLUSTER

High Performance Computing (HPC) Cluster dienen zur Abarbeitung von Rechenaufgaben. Diese Rechenaufgaben werden auf mehrere Knoten aufgeteilt. Entweder werden die Aufgaben in verschiedene Pakete aufgeteilt und parallel auf mehreren Knoten ausgeführt oder die Rechenaufgaben werden auf die einzelnen Knoten verteilt. Die Aufteilung der Jobs übernimmt dabei meistens ein Job Management System.

Bei HPC-Clustern wird die zu erledigende Aufgabe, der „Job“, oft mittels eines Decomposition-Programms in kleinere Teile zerlegt und dann auf die Knoten verteilt. Die Kommunikation zwischen auf verschiedenen Knoten laufenden Job-Teilen geschieht in der

Regel mittels Message Passing Interface (MPI), da eine schnelle Kommunikation zwischen einzelnen Prozessen gewünscht ist. Dazu koppelt man die Knoten mit einem schnellen Netzwerk.

Eine gängige Methode zur Verteilung von Jobs auf einen HPC-Cluster ist ein Job-Scheduling-Programm, welches eine Verteilung nach verschiedenen Kategorien vornehmen kann, wie z. B. Load Sharing Facility (LSF) oder Network Queueing System (NQS).

In jüngster Zeit reihen sich immer mehr Linux-Cluster in die TOP500 der Superrechner ein, nicht zuletzt weil sich auch für anspruchsvolle Rechenaufgaben billige COTS Hardware nutzen lässt.

Abbildung 27 Ein HPC Beowulfcluster

Abbildung 26 Der NASA HPC Cluster

9.3 LOADBALANCING CLUSTER

Im Load Balancing Cluster werden gleichartige Applikationen zur Steigerung der Gesamtperformance mehrfach angeboten. Durch sogenannte Loadbalancer wird der Auslastungsgrad der einzelnen Server ermittelt und die Clientanfrage an den Node mit der voraussichtlich besten Performance für den Clientprozess übergeben. Mit diesem Verfahren wird erreicht, dass dem Client die im Moment bestmögliche Leistung zur Verfügung gestellt wird. Die Qualität der Loadbalancer ergibt sich aus deren Fähigkeit, eine möglichst gute Lastprognose in Abhängigkeit von der Applikation zu stellen. Die Möglichkeit zur Wiederaufnahme von Sitzungen kann ein weiteres wichtiges Kriterium bei der Auswahl des

geeigneten Loadbalancers sein. Der Loadbalancer ist in einfacher Ausführung ein single point of failure. Er ist daher redundant auszuführen.

Der Ausfall eines Servers führt nicht zum Gesamtausfall des Systems. Der Ausfall wird vom Loadbalancer erkannt und entspricht in der Bewertung dem Zustand „vollständig ausgelastet“. Beeinträchtigt wird die Gesamtperformance des Systems. Ein typisches Einsatzgebiet für den Einsatz von Load Balancing Clustern ist der Betrieb von Webserver-Farmen.

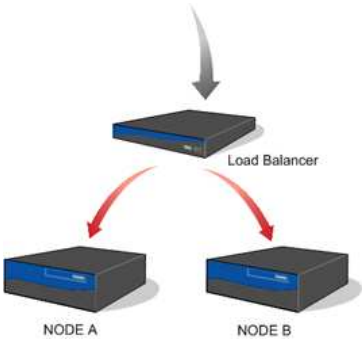


Abbildung 28 Einfacher Loadbalancer

10 GRID COMPUTING

Die Charakteristiken von Computercluster sind das Bereitstellen von reiner Rechnerleistung in einem lokalen begrenzten Bereich. Die nächste Herausforderung bestand also darin, die Rechenleistung nicht nur lokal, sondern auch global verfügbar zu machen und neben der Rechenleistung z.B. auch Daten und Applikationen bereitzustellen.

Wenn Sie das nächste Mal Ihre Arbeit kurz unterbrechen, um sich einen Kaffee oder etwas zu essen zu holen, kann Ihr Computer währenddessen weiterarbeiten ... z. B. an Berechnungen für die AIDS-Forschung, an genetischen Abgleichen für die Arzneimittelentwicklung oder an Beispielanalysen für die bessere Behandlung von Krebserkrankungen. Ihr Computer kann diese Aufgaben sogar durchführen, während Sie ihn eigentlich für andere Dinge nutzen

Dies ist möglich, wenn Sie die Zeit, in der Sie Ihren PC oder Laptop nicht nutzen, einem öffentlichen Grid zur Verfügung stellen. Beim Grid-Computing werden Tausende einzelner Computer miteinander verknüpft, aus denen ein komplexes System mit enormer Rechenleistung entsteht, ähnlich einem Supercomputer. Da die zu bewältigenden Aufgaben in unzählig viele, winzig kleine Bausteine aufgeteilt und gleichzeitig ausgeführt werden, sind für bestimmte Forschungen nicht wie bisher Jahrzehnte sondern nur noch wenige Monate nötig

Quelle: http://www-05.ibm.com/de/pov/grid/index1.html

Mitte der 1990er wurde der Begriff des Metacomputing als Möglichkeit zur Erweiterung von Paralleler Datenverarbeitung und Custer Computing eingeführt. Die Idee bestand darin, große Computersysteme über WAN-Leitungen (Wide Area Network) miteinander zu verbinden. 1997 wurden erstmals zwei Supercomputer des High Performance Computing Center Stuttgart (HLRS) und des Pittsburgh Supercomputing Centre (PSC) miteinander verbunden. Trotz der Verfügbarkeit von hohen Bandbreiten innerhalb der nationalen und internationalen Forschungsnetzwerke scheiterte das Experiment auf Grund der Latenz.

Das Metacomputing bezieht sich aber lediglich nur auf Computer (Rechenleistung) im Allgemeinen. Ian Foster und Carl Kesselman stellten 1999 ein neues erweitertes Konzept mit dem Namen Grid Computing vor, das neben Computer auch andere Arten von Ressourcen wie Software, Datenbanken, Rechenleistung, Speicherplatz oder spezielle Hardware beinhalten und miteinander vernetzen kann. Der Begriff Grid wird abgeleitet aus dem englischen Electrical Power Grid (Stromnetz). Ein Grid soll den Benutzern die Ressourcen so transparent zur Verfügung zu stellen, wie der Strom aus der Steckdose kommt. Dabei verfügt das Grid über standardisierte Schnittstellen, über die der Benutzer seine Anfragen übermitteln kann und ihm die Ressourcen dann automatisiert zugeteilt werden. Die Ressourcen sind dabei über das Internet verteilt und können unterschiedlichen ’virtuellen’ Organisationen angehören. Anhand der Schnittstellen kann der Status der Ressourcen abgefragt und diese direkt angesprochen werden. Ein entscheidender Vorteil liegt darin, dass der geographische Ort an dem sich die Ressource befindet nicht mehr von Bedeutung ist. Auf Grund des beliebigen und weltweiten Zugriffs auf Ressourcen über das Internet gilt das Grid als Generalisierung des World Wide Web. Davon abgeleitet steht die Technologie des Grid Computing somit als die Basistechnologie für die Koordination und Verarbeitung organisationsübergreifender Geschäftsprozesse und den gemeinschaftlichen Austausch und die Nutzung von Ressourcen.

Abbildung 29 GRID Computing

Das entscheidende Ziel des Grid Computings bestand also darin, Ressourcen gemeinschaftlich global zu nutzen sowie diese zu koordinieren und darüber hinaus gemeinsam Probleme institutionsübergreifend in dynamischen virtuellen Organisationen zu lösen. Genauer bedeutet dies, dass zu Beginn Formalitäten wie das Abrechnungsschema und die Zugangsrechte geklärt werden und anschließend der Zugriff auf die Ressourcen wie z.B. Rechnerleistung oder Anwendungen für die gemeinschaftliche Nutzung bereitgestellt werden. Der Begriff der virtuellen Organisation beschreibt in diesem Fall eine dynamische Allianz von Organisationen, die ein gemeinsames Interesse während der Nutzung des Grids vertreten.

10.1 Arten von Grid Computing

Je nachdem wie die Ressourcen miteinander vernetzt sind und um welches Anwendungsszenario es sich handelt, können Grids in unterschiedliche Arten unterteilt werden. Nachfolgend werden fünf unterschiedliche Arten betrachtet.

10.1.1 Computer Grids

Computer Grids werden verwendet um einem Benutzer Rechnerleistung bzw. Rechnerkapazität, die ihm in seiner eigenen Umgebung nicht zur Verfügung stehen, verteilt bereitzustellen. Das Bereitstellen kann hierbei eine derzeit nicht verwendete Ressource – z.B. eine Workstation außerhalb der Geschäftszeiten sein, oder aber auch ein Hochleistungsclustersystem.

10.1.2 Data Grids

Data Grids werden eingesetzt um große verteilte Datenmengen gemeinsam zu Nutzen und diese zu verarbeiten. Dabei wird eine sogenannte Data-Federation, eine organisationübergreifende Sicht auf alle Daten, die beispielsweise einem Projekt zugewiesen sind, definiert. Bei so einer Data-Federation handelt es sich um ein dezentral verwaltetes System, bei dem derjenige, der die Daten in dieser Umgebung zur Verfügung stellt auch die uneingeschränkte Kontrolle über diese Daten behält.

10.1.3 Application Grids

Application Grids waren der erste Ansatz um virtuelle Organisationen zu etablieren und damit die organisationsübergreifende gemeinsame Nutzung von Ressourcen voranzutreiben. Die Betreiber der Grids sollten dadurch eine höhere Auslastung und die Benutzer ein besseres Angebot erfahren. Themen, die innerhalb dieses Grids auftreten, sind sichere und schnelle Datenverbindungen, Authentifikationen, Authorisierungen und Single-Sign-On sowie Accounting und Abbrechnungsmöglichkeiten.

10.1.4 Ressource Grids

Resource Grids sind die Erweiterung der Application Grids. Diese definieren ein Rollenmodell, in dem eindeutig zwischen einem Grid Benutzer, einem Grid Provider und einem Resource Provider unterschieden wird. Die Hierarchie ist logisch geordnet. Ein Grid Benutzer verwendet die GridInfrastruktur des Grid Provider um die dort vorhandenen Ressourcen des Resource Providers zu nutzen. Für den Grid Benutzer unterscheidet sich die Funktionalität des Application- und des Resource Grids nicht. Das Konzept der beiden hat aber einen gravierenden Unterschied. Application Grids werden vertikal integriert, was bedeutet dass der Bedarf an Fremdleistungen sehr gering gehalten wird und die Komponenten individuell hinzugefügt werden. Dagegen müssen bei einem Resource Grid alle Schnittstellen definiert und offen gelegt werden, da jeder Ressource Provider über die Spezifikation der GridInfrastruktur des Grid Providers informiert sein muss um dort ggf. seine Ressourcen anbieten zu können.

11 HISTORISCHE BETRIEBSSYSTEME

Betriebssysteme – das ist nicht nur „DOS“ und „Windows“. In diesem Kapitel werden wir in Kürze einen Überblick über die Welt der „Betriebssysteme“ vor der Einführung von DOS erhalten.

11.1 WIE ALLES BEGANN

Wenn man den Begriff Betriebssystem von der EDV abkoppelt wird auch ein mechanisches Uhrwerk zu einer Art von Betriebssystem. Eine nahezu philosophische Betrachtung? Dann denken Sie einen Schritt weiter und betrachten den ersten Computer der Welt den Zuse1. Konrad Zuse konstruierte mit der Z1 das erste Rechenwerk, welches voll auf der Grundlage des Dualsystems arbeitete und nicht vorher eine Umrechnung durchführen musste. Als Schaltglieder fungieren bei der Z1, einem rein mechanischen Rechner, zwei Blechstreifen, die zueinander im 90°-Winkel stehen und einem dritten Blechstreifen, der nur dann bewegt werden kann, wenn die beiden anderen Streifen gleichzeitig bewegt werden. Dies ist die logische UND-Bedingung, die bei späteren über elektromagnetische Schaltelemente (Flip-

Abbildung 30 Das Rechnewerk der Zuse Z1

Flops) durchgeführt wurde. Die rein mechanisch arbeitende Z1, die zwischen 1935 und 1938 konstruiert wurde , funktionierte noch nicht zufrieden stellend, weshalb Zuse ein Nachfolgemodell, die Z2, entwickelte, dass die mechanischen Schaltelemente gegen Telefon-Schaltrelais ersetzte. Die Eingabe der Daten erfolgte über ein Provisorium: Zuse stanzte die Programme auf Filmstreifen, die bei den Schneidearbeiten eines Filmstudios als Abfall angefallen waren. Die Z3 mit 2600 Telefonrelais war schließlich der erste voll funktionsfähige Zuse-Rechner und 1941 fertig gestellt. Auch er wurde über Lochstreifen gesteuert, international fand er jedoch aufgrund des Krieges keine Beachtung. Die Weiterentwicklung der Z3, die Z4, wurde 1944 begonnen und von Konrad Zuse abenteuerlich durch die Wirren des Kriegsendes gerettet. 1950 wurde diese Rechenmaschine an die ETH Zürich vermietet. Ein Jahr zuvor wurde von Konrad Zuse zusammen mit Alfred Eckhard und Harro Stucken die Zuse KG in Neukirchen gegründet. Der größte Rechner, der von der Zuse KG je geschaffen worden war, ist die Z5 mit einer Gesamtlänge von 12 Metern und einer Höhe von 2,5 Metern. Die Z5 war auch gleichzeitig der letzte elektromechanische Rechner der Zuse KG. Die Z22 war der erste Röhrenrechner der Zuse KG. Mit einem Röhrenrechner konnte man in etwa tausendfach schneller rechnen als mit den älteren Relaisrechnern, zudem hatte die Z22 so viel Speicherkapazität, dass erstmals die Programme in der Maschine selbst gespeichert werden konnten und nicht mehr über Lochstreifen eingelesen werden mussten. Noch heute steht in Karlsruhe an der FH eine funktionsfähige Z22, die zunächst im universitären Einsatz genutzt und nach der Ausmusterung von zwei Mitarbeitern der FH betriebsbereit gehalten wurde. Der Einsatz von ausreichend großem Arbeitsspeicher revolutionierte auch die Möglichkeiten der Betriebssysteme.

Abbildung 31 Zuse Z22 in Berlin

Abbildung 32 funktionsfähige Zuse Z22 an der Universität Karlsruhe

11.2 DIE NÄCHSTE GENERATION

In der Zeit, als Röhren allmählich durch Transistoren ersetzt wurden, entwickelten sich auch der Aufgabenbereich der Rechenmaschinen weiter. Sie wurden nicht speziell für eine Aufgabe gebaut oder konfiguriert, sondern erhielten ihre Aufgabe in Form eines Programms beschrieben. Die Software wurde eigenständig, aber vom Betriebssystem abhängig.

Die Programme mußten zunächst in der Maschinensprache des jeweiligen Rechners geschrieben werden. Etwas später wurden dann die ersten höheren Programmiersprachen entwickelt, von denen sich FORTRAN am schnellsten und weitesten verbreitete, später im kommerziellen Bereich auch COBOL. Dadurch wurde die Programmentwicklung schneller und sicherer. Dadurch daß die Hardware zuverlässiger und das Programmieren einfacher wurde, konnten Rechner allmählich auch kommerziell eingesetzt werden, wenn auch zunächst nur in großen Betrieben und Universitäten.

Der Job des Computer-Bedieners teilte sich in zwei Teile auf

- Programmierer (Autor der Software)
- Operator (Hardware-Bedieners)

Die Eingabe erfolgte üblicherweise über Lochkarten, die Ausgabe auf Papier oder wieder auf Lochkarten. Häufig benutzte statische Daten wurden auf Magnetband gespeichert, beispielsweise Assembler, Compiler, Linker, Bibliotheken.

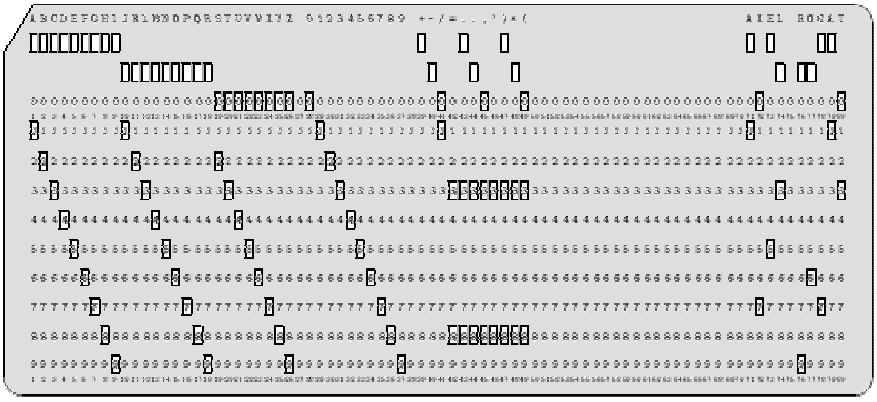


Abbildung 33 genormte Lochkarte

Eine Lochkarte stellt üblicherweise eine Zeile Programmtext (80 Zeichen) oder entsprechend viele binäre Daten dar. Die Karte ist in 12 Zeilen zu je 80 Spalten aufgeteilt. In jedes so

entstehende Feld kann ein rechteckiges Loch gestanzt werden. In jeder Spalte wird ein Zeichen oder bei Binärdaten ein binäres Datum dargestellt.

In der zeichenorientierten Darstellung (Hollerith-Code) bilden die Daten in den Zeilen 0, 11 und 12 den Zonenteil, die in den Zeilen 1 bis 9 den numerischen Teil eines Zeichens. Es darf jeweils maximal 1 Loch in den Zonenteil und maximal zwei in den numerischen Teil gestanzt werden, so daß viel Platz verschwendet wird (48 definierte Zeichen, 212=4096). Es können Großbuchstaben, Ziffern und einige Sonderzeichen dargestellt werden.

In der binären Darstellung wird die Karte in eine obere und eine untere Hälfte aufgeteilt. In jeder Spalte werden zwei 6-Bit-Binärzahlen dargestellt. Wenn diese Zahlen über einen Code wieder als Zeichen aufgefaßt werden (64 mögliche Zeichen), passen so also zwei Zeichen in eine Spalte.

11.3 Neuzeitlichere Betriebssysteme

Das **Compatible Time-Sharing System** (**CTSS**) ist das erste Anfang der 60er-Jahre am MIT entwickelte Multiuser-Betriebssystem. Jeder Nutzer bekommt reihum den Prozessor für ein kurzes Zeitsegment zugeteilt so dass er das Gefühl hat den Computer alleine zu nutzen. Es wurde mit großem Erfolg auf dem damaligen Supercomputer 7094 von IBM eingesetzt.

Das **Fortran Monitor System (FSM)** welches auch auf dem Großrechner IBM 7094 lief. Kleinere und billigere IBM 1401 Maschinen dienten als Vorrechner und bereiteten die Bänder vor. Es war somit das erste Terminalsystem

Das **Incompatible Timesharing System** (**ITS**) ist ein von den Hackern am Labor für künstliche Intelligenz des MIT geschriebenes freies Betriebssystem für die Rechner der PDP-10-Serie von DEC. ITS war bis zum Beginn der 80er-Jahre der von Hackern bevorzugte Tummelplatz unter den Betriebssystemen.

Das **Multiplexed Information and Computing Service (Multics)** ist ein Betriebssystem für Großrechner. Aus der Weiterentwicklung des MIT Compatible Time-Sharing System

entstand unter der Federführung von Fernando Corbató durch seine Arbeiten im Bereich der Multiple Access Computers **Multics** .

Es wurde ab 1963 in Zusammenarbeit von MIT General Electric und den Bell Labs von AT&T entwickelt. Die Entwicklung wurde von der ARPA finanziell gefördert. Als Programmiersprache gelangte PL/1 zum Einsatz. Ab 1969 war das erste System am MIT verfügbar. Das erste kommerzielle System wurde von **Honeywell Information Systems Inc.** Auf einer **Honeywell 6180** vertrieben bis 1986 das letzte System installiert wurde. Multics wurde zum Vorläufer für die Entwicklung von Unix

Am 30. Oktober 2000 um 17:08 wurde das letzte weltweit noch laufende Multics-System des Canadian Department of National Defence in Halifax Nova Scotia Canada heruntergefahren und außer Dienst gestellt.

12 MODERNE BETRIEBSSYSTEME

Die Aera der modernen Betriebssysteme beginnt 1969 mit der Einführung des ersten Unix Systems. 1980 begann die Auslieferung einer EDV-Revolution, dem Disketten Operating System (DOS). Diesen Betriebssystem haben wir es zu verdanken dass in den Folgejahren immer mehr Computer den Weg unter die häuslichen Weihnachtsbäumen fanden. Als dann 1990 Microsoft das Windows 3.0 auf den Markt brachte, wurde der Rechner zum PC.

12.1 UNIX UND POSIX BASIERENDE BETRIEBSSYSTEME

- AT&T Unix
  - UNIX V1-V7 (1969-1979)
  - System III kommerziell vertriebenere Quellcode von UNIX ab 1981
  - System V kommerziell vertriebenere Quellcode von UNIX ab 1983

- BSD UNIX Weiterentwicklung der USD von Unix bis 4.4BSD
  - 386BSD Portierung von Bill Jolix auf 80386-Prozessoren
  - Darwin Bestandteil von Mac OS X
  - DragonFlyBSD
  - FreeBSD modernes BSD ursprünglich für IBM PC
  - Mips OS
  - NetBSD modernes BSD für viele Prozessoren
  - OpenBSD modernes BSD für sichere
  - SunOS (Sun Operating System) UNIX der Firma Sun (bis SunOS 4.x)
  - Ultrix UNIX der Firma DEC für VAX-Computer (später auch MIPS-Workstations)

- System V –Linie
  - AIX UNIX Version der Firma IBM
  - A/UX UNIX Version der Firma Apple
  - HP-UX UNIX Version der Firma Hewlett-Packard
  - IRIX UNIX Version für Silicon Graphics Workstations
  - Open Unix
  - Sinix UNIX Version der Firma Siemens
  - SCO Unix UNIX Version der Santa Cruz Operation
  - Solaris
  - UnixWare
  - Xenix UNIX Version ursprünglich Firma Microsoft

- Unix-Derivate
  - Coherent Unix-ähnliches System der Mark Williams Company 1983
  - Linux POSIX-kompatibles Betriebssystem ursprünglich von Linus Torvalds 1991
  - HURD Mach Kernel des GNU-Projekts
  - Minix Lehrsystem von A.S.Tanenbaum 1986
  - NextStep Mach Mikrokernel
  - Tru64 ursprünglich OSF/1 1988

12.2 DOS (PC-DOS ABKÖMMLINGE)

- Caldera-DOS DOS-Alternative teilweise von Novell verwendet
- Novell-DOS DOS-Alternative der Firma Novell basierend auf DR-DOS
- MS-DOS Ur-Dos der Firma Microsoft (***MS*** –DOS)
- PC-DOS Name für MS-DOS-Version der Firma IBM
- DR-DOS DOS-Alternative der Firma Digital Research
- FreeDos freies DOS zu MS-DOS kompatibel

12.3 MS WINDOWS

Win16 –basiert (Unter MS-DOS laufend)

- Windows 1.0 erste MS-Windows-Version
- Windows 2.0 zweite Version ca. 1987
- Windows 3.x erste erfolgreiche MS-Windows Version

Win32 –basiert (Unter MS-DOS laufend)

- Windows 95
- Windows 98
- Windows ME

Windows NT – basiert

- Windows NT MACH Mikrokernel
- Windows 2000
- Windows XP
- Windows Server 2003
- Windows Vista
- Windows 7
- Windows Server 2008

Windows CE

- Windows CE Windows für Kleinrechner und Embedded Systems (Version 1-7)

12.4 MIKROKERNEL

- Mach
- Eumel
- L3
- L4

12.5 WEITERE BETRIEBSSYSTEME

- Amboss Betriebssystem für Prozessrechner der Firma Siemens
- AmigaOS Betriebssystem für Commodore Amiga Heimcomputer
- AtheOS
  - Syllable
- BeOS
  - BlueEyedOS www.blueeyedos.com
  - OpenBeOS www.openbeos.org
  - Zeta
- BS2000 Großrechnersystem der Firma Siemens AG
- CMS
- Cosmoe www.cosmoe.com
- CP/M Control Program for Microprocessors für 8080-Mikroprozessoren
  - CP/M-68k
- EPOC Betriebssystem für Handhelds
- EROS
- FDOS Lernsystem FDOS
- Flex
- GEOS Grafische Oberfläche ursprünglich für den C64
- Inferno verteiltes Betriebssystem Nachfolger von Plan 9
- MacOS
  - Mac OS X (ab Version 10)
  - Mac OS Classic (bis Version 9.2)
- MenuetOS
- Micro OS Dosähnliches Retro-Betriebssystem
- MiNT
- MorphOS
- MPE oder ***MPE/iX*** ; Extrem stabiles Betriebssystem von Hewlett Packard
- MP/M Single Processor Multitasking/ Multiuser CP/M
- MVS Großrechner-Betriebssystem der Firma IBM
  - MVS 3.8

    - MVS/SP
    - MVS/XA
    - MVS/ESA
    - OS/390
    - z/OS
- NextStep
- NetWare
- NewOS newos.sourceforge.net/
- Oberon
- OS X – Kurzform für MacOS X
- OS/2 PC-Betriebssystem von IBM
- OS/360 Großrechnersystem von IBM
    - OS/360 MFT
    - OS/360 MVT
- OS/400 Minicomputer-Betriebssystem von IBM
- OS9 Echtzeit-Betriebssystem ursprünglich für den 6809-Mikroprozessor
- OS9/68k Portierung von OS9 für den 68000 –Mikroprozessor
- PalmOS Betriebbsystem für PALM Handhelds
- Plan 9 Nachfolger von Unix
- QNX
- ReactOS
- RISC OS
- RSX-11 Echtzeit-Betriebssystem der Firma DEC für PDP-Minicomputer
- RTE
- RTOS UH Echzeit-Betriebssystem
- RT11
- SkyOS
- Symbian OS
- TENEX
- TOPS 10
- TOS Betriebssystem ursprünglich für ATARI ST-Heimcomputer
- VM Großrechner-Betriebbsystem der Firma IBM
    - VM/370
    - VM/SP
    - VM/XA
    - VM/ESA
    - z/VM
- VMS Virtual Memory System Betriebssystem der Firma DEC für VAX-Computer
    - VMS VAX ursprüngliches VMS
    - VMS Alpha Portierung von VMS auf Alpha-Prozessoren
    - OpenVMS
- VSE

Quelle: http://www.uni-protokolle.de/Lexikon/Liste_der_Betriebssysteme.html

12.6 VERGLEICH LINUX, MACOS X, MS WINDOWS

Sowohl in Serverbereich als auch auf dem Desktop-PC steht heute gerne die Frage nach dem besten Betriebssystem im Raum. Eine endgültige Entscheidung wird es wohl nicht geben. Es sind nicht nur technische Faktoren die die Qualität eines Betriebssystems ausmachen, auch wirtschaftliche und individuelle Gründe können den Vorteil ausmachen.

Mittlerweile sind sich Betriebssysteme ähnlicher denn je. Mausbedienung und Fenstertechnik sind obligatorisch. Linux und Windows können gemeinsam auf einem PC genutzt werden.

Sind Linux und Mac absolut sicher?

Der Internetalltag ist für Linux- und Mac-User tatsächlich entspannter als bei dem Windows Kollegen. Den größten Schutz bietet die immer noch geringe Verbreitung beider Betriebssysteme. Die Betreiber von Computerviren und anderen Schadprogrammen wollen möglichst schnell möglichst viele Rechner infizieren. Daher ist Windows hier die erste Wahl. Doch absolut sicher sind Linux und Mac OS nicht. Spoofing und Phishing Attacken sind von der Gutgläubigkeit des Users abhängig. Da hat das Betriebssystem kaum einen Einfluss. Aber bekannte Sicherheitslücken werden schneller gestopft als bei Windows.

Ist Linux wirklich kostenlos?

Die für eine Linuxinstallation erforderlichen Daten kann man natürlich kostenlos aus dem Internet herunterladen. Aber Neueinsteiger brauchen eine Hilfestellung per E-Mail oder Telefon. Das gibt es ab etwa 40 Euro als Kaufdistribution. Ein Systemumstieg kann dazu führen, dass Hardwareelemente gegen linuxkompatible Teile getauscht werden müssen. Die Umgewöhnung benötigt einige Zeit und im Büro oder Serverraum kommen die Betriebe nicht ohne Kosten für Schulungen aus. Fallen bei der Kostenberechnung eines Privatanwenders die Lizenzkosten für System und Software stark ins Gewicht, stellt dieser Posten für Unternehmen nur ein Teil der Gesamtkosten dar. Wann sich der Umstieg auf Linux rechnet ist von vielen Parametern abhängig.

Holt Linux mehr Leistung aus dem PC?

Jede grafische Benutzeroberfläche (GUI) macht die Betriebssysteme vergleichbar langsam. Ein leistungsfähiger Rechner und viel Arbeitsspeicher sind dafür unverzichtbar. Vergleicht man aber die aktuelle Generation von Windos und Linux, hat man bei Linux die Wahl auf eine deutlich Ressourcen schonende GUI auszuweichen oder auf Grafik ganz zu verzichten. Die dazu deutlich performantere Speicherverwaltung macht es so möglich auf altersschwacher Hardware den neuesten Linux Kernel zu betrieben.

Kann ich ein MacOS X als Server betreiben?

Ja. Wie bei seinen Konkurrenten verfügt auch MacOS über eine Servervariante. Ein Server optimiertes Betriebssystem unterscheidet sich zu einem Desktop optimierten Betriebssystem in erster Linie in der Priorisierung von Vordergrund- und Hintergrund Anwendungen. Eine typische Vordergrundanwendung ist zum Beispiel ein Textverarbeitungsprogramm. Es tritt in Interaktion mit dem Anwender. Typische Hintergrundprozesse auf einem Server, wie DNS oder DHCP sind auch auf den meisten Desktopsystemen lauffähig, werden dort aber mit geringerer Priorität verarbeitet als die Vordergrundprozesse. So kann der Benutzer ohne Wartezeit effektiver mit seinen Anwendungen arbeiten. Besonders Linux bietet zusätzlich die Möglichkeit auf ein GUI vollständig zu verzichten, nach dem Motto: Was nicht läuft, kann nicht abstürzen und verbraucht keine Ressourcen.

Ist Linux nur was für Geeks?

Die neuen Distributionen bieten zum Teil mehr Luxus als ein Windows. Die Installation und Bedienung ist vergleichbar, wenn nicht sogar besser. Die Kompatibilität mit der Windows-Welt ist besser, aber immer noch lückenhaft. Die Sicherheit dagegen ist eher höher zu bewerten. Linux hat sich vom Nischenprodukt für Computerfreaks zu einer vollwertigen Alternative entwickelt.

12.6.1 LINUX

Vorteile

- **Multiusersystem mit hoher Sicherheit als Standard**
- **legal kostenlos erhältlich**
- **Vollwertiger Unix-Ersatz**
- **Sehr flexibel und konfigurierbar**
- **Einfache Softwareverwaltung durch Paketmanager, die vorgefertigte Software-Pakete automatisiert installieren**
- **Sehr gute Hardwareerkennung**
- **Kommt auch mit leistungsschwacher Hardware aus**
- **Größtenteils ein Communityprojekt, nicht von Firmen abhängig, aktive Mitarbeit möglich**
- **Sehr hoher Aktualisierungsgrad**
- **Sehr große Auswahl an verschiedenen Distributionen für Spezialanwendungen**
- **Alle Sourcen sind frei verfügbar**
- **Seit Kernel 2.6, sehr effiziente Speicherverwaltung**

Nachteile

- **Wenig kommerzielle Software**
- **Installation von Fremdsoftware meist nur über Kompilieren möglich**
- **läuft Hardware nicht Out-of-the-Box ist ein erheblicher Rechercheaufwand von Nöten und muss im Zweifel durch linuxtaugliche Geräte ausgetauscht werden.**
- **viele Distributionen - etliche Projekte sterben nach einiger Zeit aus.**
- **Support ist am Markt nur bedingt zu bekommen**
- **Erhöhter Schulungsaufwand in Betrieben**

12.6.2 MacOS X

Vorteile

- **Vollwertiges Unix-System**
- **Sehr gute Softwareintegration**
- **Häufig genutzte kommerzielle/proprietäre Software ist vorhanden**
- **Hervorragende Einbindung in die vorgegebene Hardware**
- **Sehr gute Bedienbarkeit**
- **Besonders gute Multimedia Integration**
- **Sehr hohe Akzeptanz im Bereich Grafikdesign und Multimedia**
- **Sehr performant durch die optimierten Treiber**

Nachteile

- **Hardwarebindung, läuft aus Lizenzgründen nur auf Macintosh-Rechnern**
- **Teuer in der Anschaffung**
- **Schlechte Umsetzung von bekannten Sicherheitsmängeln in Patches**
- **Geringe Verbreitung, daher hoher Schulungsaufwand**
- **Begrenzte Menge an Applikationen**

12.6.3 MS WINDOWS

Vorteile

- **Größtes Angebot an kommerzieller Software, die sich problemlos hinzu fügen lässt**
- **Bei nahezu jedem neuen Rechner ohne Aufpreis dabei**
- **nahezu alle Consumer-Geräte (Scanner, Drucker etc.) bringen Windows-Treiber mit.**
- **Die Mehrheit der Computernutzer ist damit vertraut, was den Schulungsaufwand für Betriebe verringert**
- **Support ist am Markt einfach zu bekommen**
- **Sehr hohe Akzeptanz bei den Anwendern**

Nachteile

- **Für Sicherheit muss erst gesorgt werden**
- **Bewusst schlechte Kompatibilität zu anderen Systemen (standardmäßig kaum Unterstützung für Standardprotokolle und -dateiformate)**
- **Bei einem Wechsel des Treibermodells (z.B. von XP auf Vista) werden alte Geräte wie Scanner und Drucker zum Teil nutzlos**
- **Da es das verbreiteteste System ist, sind hierfür auch die meisten Viren und Trojaner im Umlauf**

Fazit:

Zu Windows haben Computerspieler keine Alternative – abgesehen von Spielekonsolen. Es überzeugt durch das vielfältige Angebot auch preiswerter Geräte und Programme. Mac OS ist leistungsfähig, läuft stabil, aber nur auf vergleichsweise teurer Hardware. Im Internet surft der Nutzer recht sicher. Mac OS lässt sich besonders einfach für mehrere Nutzer mit unterschiedlichen Rechten einrichten. Linux ist so sicher wie Mac OS und läuft wie Windows auch mit proprietärer Technik. Es verlangt aber Spaß am Administrieren des Systems; insbesondere, wenn der Anwender mit Zusatzgeräten wie beispielsweise Scannern experimentiert.

Welches Betriebssystem ist für mich das Beste?

Ganz einfach: Das mit dem ich mich am besten auskenne, oder den besten Support bekomme

Ein gut administriertes Windows ist allemal sicherer und stabiler als ein halbherzig installiertes Unix.

13 BETRIEBSSYSTEME DER ZUKUNFT

Aktuell werden völlig neue Anforderungen an moderne Betriebssysteme gestellt. Wo früher das Einsatzgebiet der Computer deutlich umrissen war, findet man heute in den unterschiedlichsten Geräten eine elektronische Logik, die auf eine Betriebssystem angewiesen ist. Zum Einen entwichelt sich die Anforderung wiedrt zurück zu den Anfängen der EDV: Ein Gerät, ein Programm, eine Aufgabe. Doch gibt es zunehmend einen Markt für elektronische Multitalente, deren aufgabengebiet nicht umfangreich genug sein kann. Ein einfacher MP3 Player kann nur ein Datenformat in Musik umwandeln, hat keine grafische Ausgabe und nur wenige Eingabetasten. Ein Multi Funktions Mobiltelefon kann auch MP3 abspielen, diese ist aber nur eine von unzählichen Unter- und Nebenfinktionen.

Neue Einsatzgebiete erfordern neue Funktionen.

13.1 EMBEDDED SYSTEMS

Eingebettete Systeme verrichten in einer Vielzahl von Anwendungsbereichen und Geräten meist unsichtbar ihren Dienst. Sie finden Einsatz in Geräten der Medizintechnik, Waschmaschinen, Flugzeugen, Kraftfahrzeugen, Kühlschränken, Fernsehern, DVD-Playern, Mobiltelefonen Espressomaschinen, um nur wenige Beispiele zu nennen. Im Fall von komplexen Gesamtsystemen handelt es sich dabei meist um eine Vernetzung einer Vielzahl von ansonsten autonomen, eingebetteten Systemen (z. B. im Fahrzeug oder Flugzeug).

Dabei sind Embedded Systems keine Erfindung der Neuzeit. Das erste bemerkenswerte moderne eingebettete System war der Apollo Guidance Computer, das von Charles Stark Draper zusammen mit dem MIT Instrumentation Laboratory entwickelt wurde. Jeder Flug auf den Mond hatte zwei dieser Systeme dabei, die zur Steuerung verwendet wurden. Das inertial guidance system wurde sowohl im Kommandomodul als auch in der Mondlandefähre (LEM, Lunar Excursion Module) verwendet.

Zu Beginn des Apollo-Projekts wurde genau dieses System als eine der riskantesten Komponenten des Projektes angesehen.

Die ersten eingebetteten Systeme wurden allerdings schon vorher in der Minuteman-Rakete eingesetzt und als Folge davon in Massenproduktion hergestellt. Die Anwendung war ein Wege-Such-System, das der Rakete nach einmaliger Programmierung ein unabhängiges Manövrieren ermöglichte. Die verwendeten integrierten Schaltungen wurden nach einiger Zeit, dank der Massenproduktion, für jedermann erschwinglich und damit nutzbar gemacht.

Abbildung 34 Apollo Guidance Computer

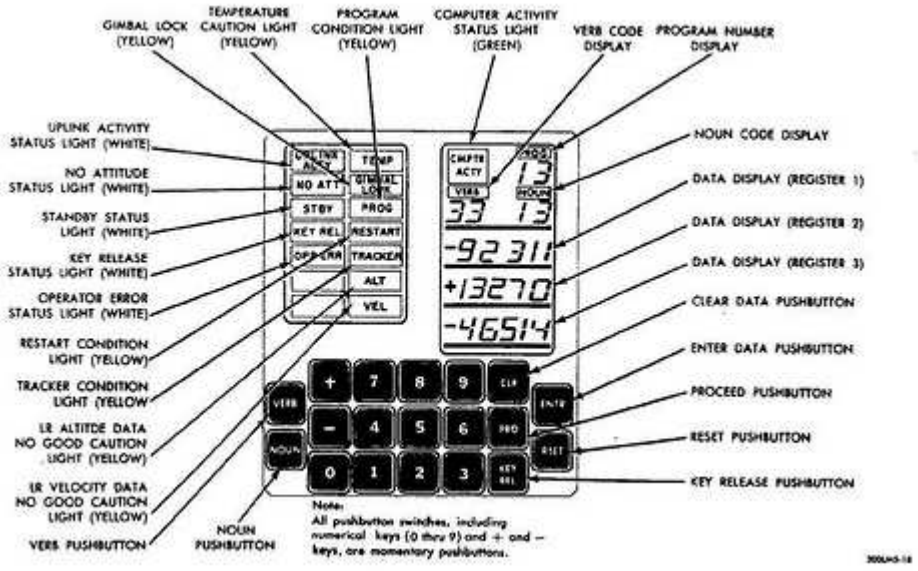


Abbildung 35 Apollo Guidance Computer Anleitung

13.1.1 PLATTFORMEN

Embedded Systems können auf vielen verschiedenen CPU-Architekturen wie zum Beispiel 8051, ARM, AVR, MIPS, PowerPC, 68k/Coldfire, Intel x86, 68HC12, C167, Renesas M16C und diverser anderer 8/16/32-Bit-CPUs oder deren Untergruppen 8051, AVR, PIC16, ARM7, PowerPC 5xx, MIPS 4k, AMD AU1000, Intel Pentium M realisiert werden.

13.1.2 ARCHITEKTUREN

Die Elektronik bildet meistens ein Mikroprozessor mit entsprechender Peripherie oder ein Mikrocontroller. Einige größere Systeme verwenden Allzweck-Mainframes oder Minicomputer. Die Verbreitesten Architekturen sind:

- ARM-Architektur
- Infineon TriCore
- MIPS-Architektur
- 68000-Architektur
- Coldfire-Prozessor
- PowerPC-Architektur
- MCS-51
- TI MSP430
- Atmel AVR
- Atmel AVR32
- PICmicro PIC und PIC32 von Microchip
- MOS Technology 6502
- Siemens Sitet

Abbildung 36 Motorola 6800 CPU

Abbildung 37 ColdFire CPU

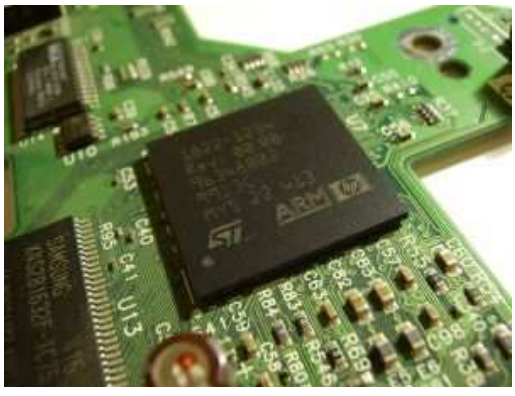


Abbildung 38 Strong ARM CPU

13.1.3 BETRIEBSSYSTEME

Wenn ein Betriebssystem eingesetzt wird, arbeitet man in Embedded Systems meistens mit sehr spezialisierten Betriebssystemen, wie zum Beispiel

- QNX
- VxWorks
- Nucleus
- OSEK, OS-9
- RTEMS

Auch spezielle eingebettete Versionen von Standardbetriebssystemen wie

- Linux (Embedded Linux)
- Android
- NetBSD
- Windows CE
- XP Embedded
- Automotive
- Embedded for PoS

finden inzwischen große Ausbreitung.

Abbildung 39 Android Mobiltelefon

Abbildung 40 PDA mit Windows CE

Abbildung 42 Registrierkasse mit Embedded XP

Abbildung 41 KFZ Bordcomputer mit Embedded Linux und Java

13.2 CLOUD COMPUTING

Das Internet revolutionierte nicht nur den Informationsaustausch der Anwender, sonder auch der Systeme. Heute ist es möglich, Anwendungen und Funktionen auszulagern. Ein Bekanntes Beispiel einer ausgelagerten Funktion ist der Emailaccount. Kaum ein Anwender betreibt einen eigenen Mailserver. Dieser Dienst wird dem Provider überlassen. Wir greifen auf unsere Mails mit einem Client oder einer Webapplikation zu.

Der Grundgedanke beim Cloud Computing ist, dass alle Anwendungen im Web laufen; von einfacher Software bis hin zu kompletten Betriebssystemen.

13.2.1 DEFINITION (LAUT WIKIPEDIA)

Der Begriff „Cloud Computing“ stammt aus dem IT-Management und wird dem Wirtschafts-Professor Ramnath K. Chellappa zugeordnet. Der Name leitet sich von dem in Strukturplänen verwendeten Wolkensymbol für das Internet ab. Es existieren eine Reihe von pragmatischen Definitionsansätzen:

- „Cloud Computing“ steht für einen Pool aus abstrahierter, hochskalierbarer und verwalteter IT-Infrastruktur, die Kundenanwendungen vorhält und falls erforderlich nach Gebrauch abgerechnet werden kann. (Quelle: Forrester Research)
- „Cloud Computing“ umfasst On-Demand-Infrastruktur (Rechner, Speicher, Netze) und On-Demand-Software (Betriebssysteme, Anwendungen, Middleware, Management- und Entwicklungs-Tools), die jeweils dynamisch an die Erfordernisse von Geschäftsprozessen angepasst werden. Dazu gehört auch die Fähigkeit, komplette Prozesse zu betreiben und zu managen. (Quelle: Saugatuck Technology)

Demzufolge geht „Cloud Computing“ über andere gegenwärtig diskutierte Ansätze (Software-as-a-Service (SaaS), „Organic Computing“) und Konzepte (Virtualisierung) hinaus. Unter der Bedingung einer öffentlichen Verfügbarkeit, ähnlich beispielsweise dem öffentlichen Telefonnetz, kann man „Cloud Computing“ je nach Architektur auch als Summe von SaaS und „Utility Computing“ ansehen.

Betriebssysteme der Zukunft

Das Unternehmen Gartner sieht in Cloud Computing gar als den Hype 2009 an und prophezeit, dass es sich in den kommenden zwei bis fünf Jahren etabliert und die Welt der Informationstechnologie grundlegend verändern wird.

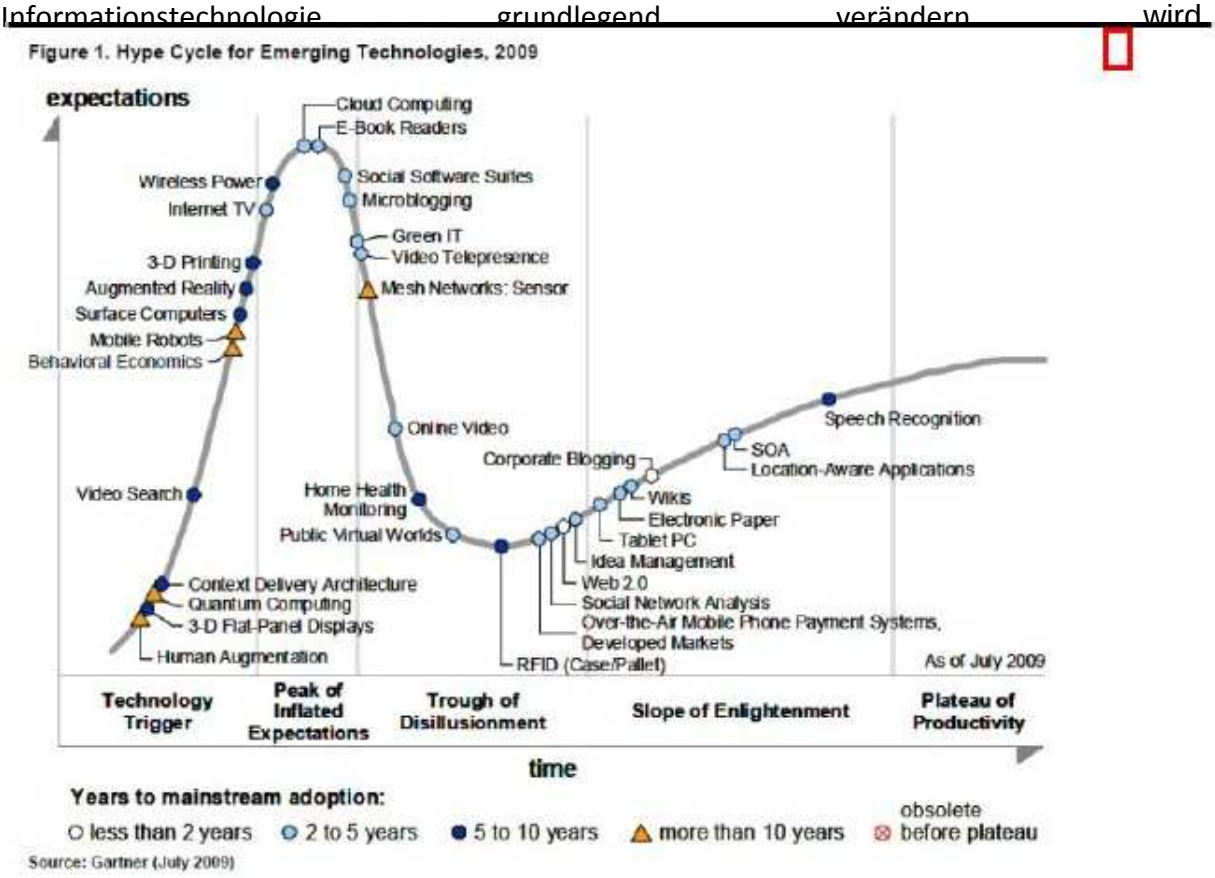


Abbildung 43 Gartner´s Hype Cycle 2009

13.2.2 TYPEN

Es werden grundlegend drei unterschiedliche Cloud Computing Typen unterschiden

13.2.2.1 PRIVATE CLOUDS

Von einer Private Cloud wird gesprochen, wenn Organisationen ihre eigenen Rechenzentren betreiben und ihre Dienste nur für Ihre eigenen Zwecke innerhalb ihrer eigenen privaten Netze verwenden und der Allgemeinheit nicht zur Verfügung stellen. Die Datensicherheit, 'Corporate Governance' und Zuverlässigkeit liegen damit in ihrem eigenen Einflussbereich. Aus diesem Grund werden Private Clouds nur indirekt zum Cloud Computing gezählt.

13.2.2.2 Public Clouds

In der Public Cloud dagegen werden die Dienste (Rechenkapazität/ Speicherplatz etc.) gegen Bezahlung oder kostenlos der Allgemeinheit zur Verfügung gestellt. Die oben genannten Aufgaben, die ein Unternehmen in der Private Cloud vornimmt, werden in der Public Cloud dann von einem Drittanbieter übernommen. Die Aufgaben und Services von unterschiedlichen Kunden werden dabei auf derselben Infrastruktur (Server, Speicher, etc.) gemeinsam gehostet und verarbeitet. Ein einzelner Kunde hat keine Kenntnis darüber, wessen Dienste ebenfalls auf derselben Infrastruktur gespeichert und verarbeitet werden.

13.2.2.3 Hybrid Clouds

Die Hybrid Cloud stellt eine Mischung aus der Private und der Public Cloud dar. Dabei verfügen Unternehmen zwar über ihre eigene Private Cloud, verwenden aber zusätzlich Dienste aus der Public Cloud von externen Anbietern. Die Attraktivität besteht vor allem darin, dass der externe Anbieter bei Bedarf schneller und kostengünstiger die benötigte Infrastruktur/ Dienste erhöhen bzw. verkleinern kann. Die Dienste werden so in die Private Cloud integriert, dass der Endanwender nicht merkt, dass er eigentlich woanders arbeitet.

13.2.3 Cloud Dienste

Innerhalb der Wolke existieren drei unterschiedliche Möglichkeiten wie Dienstleistungen bereitgestellt werden können. Da sie aufeinander aufbauen wird in diesem Zusammenhang auch von Schichten (Layers) gesprochen.

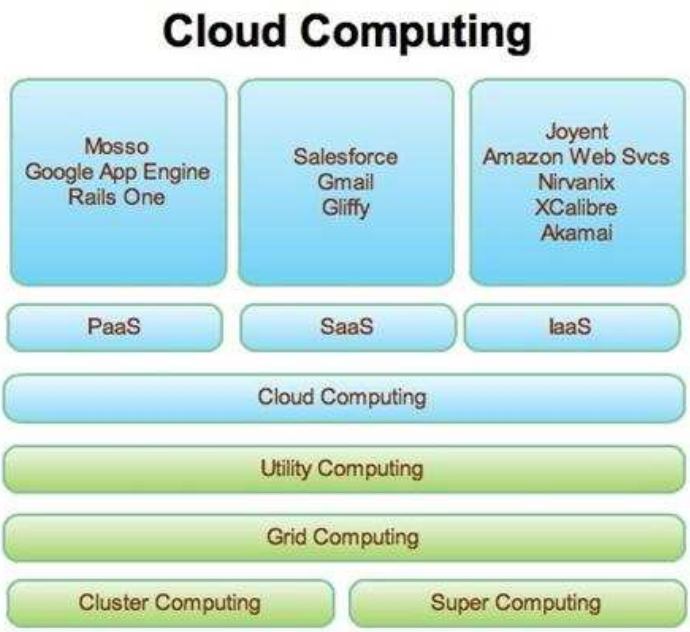


Abbildung 44 Cloud Computing Dienste

In der oberen Abbildung sieht man den blauen Bereich für Cloud Computing, der sich in die drei Bereiche PaaS, SaaS und IaaS aufteilt.

13.2.3.1 IAAS

Infrastructure as a Service (IaaS) bildet die Grundlage und stellt die grundlegenden Dienste wie Speicherplatz und Rechenkapazität bereit. In diesem Zusammenhang kann auch von Hardware as a Service (HaaS) gesprochen werden, da die gesamte Infrastuktur – Server, Speicherplatz, aber auch Router und Switches – mittels Virtualisierung bereitgestellt und gemietet (i.d.R pay per use) werden. Die gesamte Infrastruktur ist so skaliert, dass sie in Zeiten von Spitzenlast dynamisch er- weitert wird und somit unterschiedlichen Auslastungen angepasst werden kann. Bei IaaS ist der Drittanbieter lediglich für die Bereitstellung und Wartung der Hardware zuständig. Alle anderen benötigten Ressourcen wie z.B. das Betriebssystem, Anwendungen etc. obliegen dem Unternehmen

13.2.3.2 PaaS

Platform as a Service (PaaS) ist die nächste Schicht und geht einen Schritt weiter. Sie ist dafür zuständig eine transparente Entwicklungsumgebung bereitzustellen. Dabei stellt der Drittanbieter eine Plattform zur Verfügung auf der (Web)-Anwendungen entwickelt, getestet und gehostet werden können. Die Anwendungen werden anschließend auf der Infrastruktur des Anbieters ausgeführt und nutzen dessen Ressourcen. Der vollständige Lebenszyklus einer Anwendung kann darüber vollständig verwaltet werden. Über APIs können die Dienste auf der Plattform des jeweiligen Anbieters angesprochen werden. Der Vorteil besteht darin, dass vor allem kleine Unternehmen ihre Entwicklungsinfrastruktur auf ein Minimum beschränken können. Sie benötigen lediglich einen Desktop-PC, einen Web-Browser, evtl. eine lokale IDE, eine Internetverbindung und ihr Wissen, um Anwendungen zu entwickeln. Der Rest obliegt dem Drittanbieter, der für die Infrastruktur (Betriebssystem, Webserver, Entwicklungsumgebung etc.) verantwortlich ist. Auch hier erfolgt die Abrechnung mit dem Prinzip per pay use.

13.2.3.3 SaaS

Software as a Service (SaaS) stellt dem Anwender vollständige Anwendungen zur Verfügung. Sie kann als eine Art Distributionsmodell verstanden werden, bei dem die Nutzung von Software (Lizenzen) über das Internet von einem Drittanbieter angeboten wird. Der Drittanbieter übernimmt dabei u.a. die Wartung/ Aktualisierung und das Hosting der Software. Für den Anbieter besteht der Vorteil darin, dass nur eine Instanz einer Software auf den Servern bereitgestellt werden muss, welche unzählige Anwender gleichzeitig nutzen können. Wird die Software auf einen aktuellen Stand gebracht, genügt ein Update Vorgang an zentraler Stelle und die Software ist für alle Anwender gleichzeitig aktualisiert. Der Vorteil für den Anwender besteht darin, dass lediglich – wie schon bei PaaS – nur ein Desktop-PC, ein Web-Browser und eine Internetverbindung ausreicht um z.B. Dienste wie E-Mail, Office Anwendungen oder sogar ERP-Systeme nutzen zu können. Die Anschaffung und Wartung großer Serverlandschaften bzw. Softwarepakete entfällt ebenso wie das ’lästige’ Updaten der lokalen Anwendungen. Der Drittanbieter sorgt immer für einen aktuellen Stand der Software und stellt die gesamte Infrastruktur für das Hosting der Software bereit. Dazu

gehört auch das Speichern von Dateien (Dokumenten) auf den Servern des Anbieters. Der Anbieter ist demnach für alle notwendigen Bereiche des Betriebs, wie Verfügbarkeit, Backup, Redundanzen und auch die Stromversorgung verantwortlich. Auch hier erfolgt die Abrechnung wieder mit pay per use, mit dem kleinen Unterschied, dass die Kosten pro nutzenden Anwender der Software berechnet werden

13.3 ORGANIC COMPUTING

Laut der Einladung zu dem OCI-Workshop in Hannover, Dez. 2003 werden Computersysteme bei gleichen Kosten leistungsfähiger oder bei gleicher Leistung billiger und kleiner. Zudem erlaubt die drastisch gestiegene Bandbreite der Kommunikationskanäle eine flexible Verteilung von Rechnerkomponenten und Funktionen. Künftige Computersysteme werden dadurch jedoch wesentlich komplexer.

Die Beherrschung der Komplexität verlangt ein neues, mehr am Menschen als an der Technik ausgerichtetes Paradigma. Wir erwarten eine auf den Benutzer, auch auf den Informatik-Laien, zugeschnittene Entwicklung. Computer müssen sich an den Menschen anpassen, nicht umgekehrt. Ein Großteil der (lokalen) Rechenleistung wird auf die Benutzerschnittstelle verwendet werden. Die Bedienung von Computern und computerisierten Gegenständen muss weitgehend intuitiv erfolgen. Sie wird sich an unterschiedliche Benutzergruppen und Situationen anpassen und lernfähig sein. Dementsprechend werden sich neuartige Interaktionsformen an der natürlichen zwischenmenschlichen Kommunikation orientieren.

Eine sensorgestützte Kenntnis der situativen Umgebung, des Kontexts, der zumeist tragbaren Rechner wird es diesen erlauben, Informationen und Dienste im Hinblick auf die augenblickliche Situation des Anwenders zu filtern. Wir nähern uns damit dem Ideal des von Mark Weiser (Xerox PARC) geforderten „Calm Computing“, d.h. einem System zumeist unsichtbarer, in die Umgebung eingebetteter Rechner, die unauffällig arbeiten und nur bei Bedarf und auf Anforderung ihre Dienste anbieten.

Computersysteme, die diese Ziele erreichen sollen, müssen Eigenschaften besitzen, die sie lebensähnlich, organisch, erscheinen lassen.

Organische Computersysteme bestehen aus autonomen und kooperativen Teilsystemen und arbeiten, soweit möglich, selbstorganisierend. *Selbstorganisation* erlaubt adaptives und kontextabhängiges Verhalten. Zur Selbstorganisation werden u.a. die sog. self-x Eigenschaften benötigt. Hierzu gehören:

- selbst-konfigurierend
- selbst-optimierend
- selbst-heilend
- selbst-schützend
- selbst-erklärend.

13.3.1 Anwendungsszenarien für organische Computer

Die Gesellschaft für Informatik (GI) und die Informationstechnische Gesellschaft im VDE (ITG) haben eine Fülle von Anwendungsszenarien für den organischen Computer entwickelt. Sie zeigen beispielhaft, wie die Systeme der Zukunft selbst-konfigurierend, selbst-heilend, selbst-schützend und selbst-optimierend arbeiten:

Smart Factory:

Autonome Roboter können mit Hilfe spontaner Vernetzung Föderationen bilden, um anstehende Aufgaben zu erledigen. Fallen Teilsysteme aus oder sind überlastet, wird dies erkannt und die Aufgaben werden neu verteilt – selbstkonfigurierend, selbstheilend und selbstoptimierend.

Smart Warehouse:

In einem intelligenten Warenhaus lassen sich einzelne Artikel erkennen und überwachen. Schon heute können portable Geräte über Transponder mit den Etiketten der Waren kommunizieren. Bereits in wenigen Jahren werden Regale, Vorratsbehälter, Waren,

Einkaufswagen und elektronische Einkaufszettel miteinander kommunizieren, um z.B. automatisch den Inhalt des Einkaufswagens zu ermitteln und den Kunden zu noch fehlenden Artikeln zu lenken. Produkte lassen sich über die gesamte Versorgungskette Positionspapier hinweg überwachen. So kann etwa ein Chip registrieren, ob der Tiefkühlspinat immer richtig gekühlt wurde.

Smart Network/Smart Grid:

Das Internet wird Anwender im Jahr 2010 als weltweiter, heterogen aufgebauter Parallelrechner (Grid) mit Rechnerleistung versorgen. Er ist mit Hilfe intelligenter Netzwerktechnik selbst-organisierend, selbst-konfigurierend, selbst-heilend und selbst-optimierend.

Anthropomatik:

Die Lebenswissenschaften gelten derzeit als eines der spannendsten Forschungsgebiete. Sie gehen unter anderem davon aus, dass Informatik und Technik in Zukunft auf die individuellen Bedürfnisse des einzelnen Menschen abgestimmt sein werden. Das interdisziplinäre Forschungsgebiet ist schwerpunktmäßig in der Informationstechnik angesiedelt. Der allgegenwärtige und umfassende Einsatz der Informatik zielt zum Beispiel auf den Ausgleich individueller Leistungseinbußen ab, die etwa krankheits- oder altersbedingt auftreten können. Der organische Computer mit seinen lebensähnlichen Eigenschaften ist die Basis für solche Anwendungen.

Der vertrauenswürdige Computer:

Die Kombination aus Schutz durch Hardware-Maßnahmen und durch Kryptographie ist die Basis für den vertrauenswürdigen Computer. Er kommt dem Wunsch des Menschen nach informationeller Selbstbestimmung entgegen, falls die Integrität des persönlichen Datenbereichs sichergestellt und dem Benutzer glaubhaft gemacht werden kann. Ein Zugriff

von außen kann nur erfolgen, wenn eine explizite Interaktion des berechtigten Benutzers über die Benutzungsschnittstelle vorangeht. Der vertrauenswürdige Computer kann auf eine Chipkarte reduziert werden, die über biometrische Sensorik fest an einen Benutzer gekoppelt ist. Durch kryptographische Maßnahmen lässt sich eine stufenweise Erweiterung der Sicherheitshülle erreichen. Elektronische Unterschriften werden damit möglich, ohne dass sensible Information preisgegeben wird.

Roboter im Haushalt:

39% der Frauen und 25% der Männer in Deutschland wünschen sich Roboterhilfe an erster Stelle im Haushalt – so das Ergebnis einer Umfrage des VDE zur Zukunfts-IT. Heutige Geräte sind aufgrund der noch zu einfachen Sensorik (Infrarot, Ultraschall, mechanische Kontakte) und wegen der geringen Leistung der Steuersoftware dazu erst ansatzweise geeignet. Künftige Roboter sollen bei geringeren Kosten autonom werden. Sie müssen dazu selbst-konfigurierend, selbst-heilend und selbst-optimierend arbeiten. GI und ITG gehen von einem Durchbruch in diesem Gebiet in wenigen Jahren aus. Eine Schlüsseltechnologie hierfür sind Verfahren der biologisch inspirierten Informationsverarbeitung – insbesondere des Organischen Computers.

13.3.2 Die Rechner und Systemarchitekturen des Jahres 2010

GI und ITG haben eine Vision zu den Rechner- und Systemarchitekturen im Jahr 2010 entwickelt, welche den Organischen Computer ermöglichen werden. Selbstorganisation mit all ihren Facetten wird sämtliche Komponenten eines künftigen Organischen Computers bestimmen. Und Selbstorganisation kann sich nicht auf isolierte Komponenten beschränken: Sie verlangt ein übergreifendes Zusammenspiel des Gesamtsystems. Die Wissenschaftler kommen im Einzelnen zu folgender Einschätzung: Die Hardware- und Software-Basis: adaptiv und flexibel Hardware Zur Realisierung des organischen Computers sind neue Chip-Generationen erforderlich. Sie zeichnen sich durch dynamische Rekonfigurierbarkeit, hohe Anpassungsfähigkeit, Robustheit und Fehlertoleranz aus. Diese Chips werden noch immer elektronisch realisiert werden, wobei die Nanotechnologie allerdings die Integration

räumlicher Strukturen erlauben muss. Man geht davon aus, dass 3-D-Strukturen dazu beitragen werden, die Verdrahtungsprobleme der Mikroelektronik zu lösen. Rekonfigurierbare Hardware passt sich dem Bedarf an, indem Hardware-Komponenten dynamisch verschaltet werden, um gezielt bestimmte wechselnde Funktionen zu erbringen. Beim Software Defined Radio wird eine neue Hardware-Konfiguration über die Luftschnittstelle geladen, so dass sich das Gerät an neue lokale Übertragungsstandards anpassen kann. Forschungsbedarf besteht auch auf dem Gebiet der optischen Nachrichtenübertragung, um beispielsweise eine analoge Signalerfassung mit paralleler digitaler Signalverarbeitung auf engstem Raum zu koppeln. Bei Netzwerken geht es weiterhin um hohe Bandbreiten und niedrige Latenzen, die am ehesten durch optische Technologien erfüllt werden können. Der nächste Schritt dabei besteht in der Nutzung von Wellenlängen-Multiplex-(WDM)-Techniken für WAN (Wide Area Networks) und SAN (Small Area Networks). Software Im Jahr 2010 wird die Komponententechnologie etabliert sein: System-Software setzt sich aus vielen und in ihrer Größe unterschiedlichen Software-Komponenten zusammen. GI und ITG gehen davon aus, dass Software- Komponenten dazu äußerst fein strukturiert sein müssen. Die Rede ist von „leicht- bis federgewichtigen“ Strukturen der System-Software, damit die Möglichkeiten der rekonfigurierbaren Hardware voll ausgeschöpft werden. Es wird keine fest konfigurierte Standard-Software-Basis mehr geben, die identisch auf allen Knoten des organischen Computers residiert. Es ist künftig nicht mehr entscheidend, die Komponenten selbst lokal bereitzustellen. Vielmehr müssen die von ihnen erbrachten Funktionen transparent auch aus der Entfernung zugänglich gemacht werden. Organische Netzwerke werden hochgradig dynamische Software-Systeme unterstützen. Der Interoperabilität der Komponenten kommt daher zentrale Bedeutung zu. Auch in Zukunft wird es nicht den einen Standard für Betriebssystem, Laufzeitsystem und Middleware geben, wohl aber eine Vielzahl von Spezialzwecksystemen. Die meisten der heute produzierten Prozessoren (98%) werden in eingebetteten Systemen verwendet, und daran wird sich nichts ändern, da auch der organische Computer selbst ein eingebettetes System darstellt – beziehungsweise aus einer Vielzahl eingebetteter Subsysteme besteht. Standard-APIs (Application Programming Interface) werden sich durchsetzen, die heute schon die Schnittstelle zwischen Software-Basis und Anwendungen definieren. Die Software-Basis des organischen Computers unterscheidet sich im inneren Aufbau, in ihrer Struktur und Funktionsweise von der Software-Basis heutiger Systeme. Die Komponenten werden

dynamisch bei Bedarf den jeweiligen Umständen angepasst und dann eingebunden, ausgetauscht, verlagert und/oder wieder entsorgt. GI und ITG sehen hier bis zum Jahr 2010 neue Herausforderungen bei Entwurf, Implementierung, Generierung und Maßschneiderung sowie bei der semi-automatischen Konstruktion von korrekten (Sub)-Systemen aus korrekten Software-Komponenten. Energieeinsparung – auch bei Computern Die Reduktion des Energiebedarfs von Mikroprozessoren und –Controllern stellt eine zentrale Anforderung an zukünftige Rechnersysteme dar und bildet damit einen wichtigen Forschungsschwerpunkt der Informatik und Informationstechnik. Zum einen ist in den immer kleiner werdenden mobilen Geräten die durch Batterien verfügbare Energiemenge begrenzt, zum anderen erzeugen auch Hochleistungsprozessoren immer mehr Abwärme pro Chipfläche, so dass eine weitere Erhöhung der Taktraten und damit der Verarbeitungsleistung ohne geeignetes Energiemanagement auf dem Chip in Zukunft nicht mehr möglich sein wird. Energiespartechniken müssen auf vielen verschiedenen Ebenen einer Rechner- und Systemarchitektur ansetzen. Es sind dies sowohl die schaltungstechnischen Ebenen wie auch die (Mikro-)Architektur- sowie die Software und Betriebssystemebenen. Für zukünftige Rechnersysteme ist Forschung auf allen diesen Ebenen notwendig, wobei insbesondere die Wechselwirkungen zwischen den Ebenen immer wichtiger werden. Schnittstellen zwischen System und Anwender: anpassbar und Kontext-sensitiv Die Gesellschaft für Informatik (GI) und die Informationstechnische Gesellschaft im VDE (ITG) gehen davon aus, dass sich Computer in der Zukunft an den Menschen anpassen müssen, nicht umgekehrt. Daher dürfte auch ein großer Teil der lokalen Rechenleistung auf die Benutzerschnittstelle verwendet werden. Die Bedienung der Computer von morgen wird weitgehend intuitiv erfolgen. Berührungsempfindliche Bildschirme, Spracheingabe und Gestenerkennung können den Rechner zum freundlichen Partner und Problemlöser machen. Die kleinen, oftmals tragbaren Computer werden in der Lage sein, Informationen und Dienste so zu filtern, dass sie der augenblicklichen Situation des Anwenders angemessen sind. Von besonderem Vorteil ist, dass die Anwender auf diese Weise nicht mit Informationen überflutet werden. Menschen nutzen zur Informationsfilterung den lokalen Kontext: Nur die unmittelbare Umgebung ist relevant und sinnlich erfassbar. Die Eigenschaft Kontext-sensitiven „Verhaltens“ wird jetzt auch technischen Systemen vermittelt. Dazu müssen sie unter anderem eine geeignete Sensorik zur Erfassung der Kontextinformation und geeignete Mechanismen zur Informationsauswahl oder -filterung besitzen. Anpassungsfähige und kontextsensitive

Benutzerschnittstellen erlauben ein assoziatives Navigieren zum Beispiel bei der Abfrage von Dateien oder bei der Suche im Internet. Die Größe von Geräten wird auch in Zukunft durch Benutzungsschnittstellen wie Display und Tastatur bestimmt. Denn die mikroelektronischen Komponenten werden weiter miniaturisiert, nicht aber der Mensch. Um sich die wachsende Funktionsvielfalt besser zunutze machen zu können, werden heutige Eingabemedien jedoch durch Sprachsteuerung und intelligente Menüauswahltechniken ergänzt oder ersetzt. Feste aber benutzerangepasste Bedienkonzepte bilden den Anfang. Ziel ist die dynamische Anpassung der Benutzungsschnittstelle an den Nutzer und die momentane Benutzungssituation: Die adaptive Benutzungsschnittstelle führt zur Personalisierung der Systemfunktionen. Auch auf der Ausgabeseite kommen neue Verfahren auf uns zu. Mit neuen Ausgabetechniken wie dem Smart Paper lassen sich online aktualisierbare und falt- oder rollbare Zeitungen realisieren. Ein weiterer Forschungsbereich beschäftigt sich mit der entkoppelten Benutzungsschnittstelle: Der Begriff der Entkopplung bezieht sich auf die räumliche Nähe zwischen Benutzungsschnittstelle, Rechner, Speicher und Benutzer. Wird nämlich die Kommunikation zwischen Rechnerkomponenten sehr schnell und billig, dann könnte man z.B. alle seine persönlichen Datenzusammen mit einer einfachen Benutzungsschnittstelle in Form eines PDA (Personal Digital Assistant) „in der Hosentasche“ bei sich tragen. Wir erhalten so einen Personal Server. Wird größere Rechenleistung benötigt, so benutzt man einen Compute-Server im Netz. Und reicht die einfache Benutzungsschnittstelle des PDA nicht mehr aus, so koppelt sich der Personal Server mit einer gerade in der Nähe befindlichen Tastatur und einem Bildschirm, um sie temporär mitzubenutzen. Flexibel und preiswert ist bereits heute die Entkopplung mit drahtlosen Technologien wie Bluetooth oder WLAN. Als neue Technologie zeichnet sich hier Ultra Wide Band (UWB) im Gigabit/s-Bereich ab, das die Bandbreite im Nahbereich enorm erhöht. UWB ermöglicht eine grundsätzliche Entkopplung der Rechnerkomponenten und ihre flexible Ad-hoc-Konfiguration, gemäß der Vision des organischen Computers. Neue Kommunikations- und Verteilungsmechanismen In organischen Computer-Systemen wird selbst ein einzelnes Gerät eines Nutzers weitgehend transparent in eine Infrastruktur eingebunden. Es kann deren Dienste nutzen und eigene Dienste und Informationen anderen zur Verfügung stellen. GI und ITG gehen davon aus, dass dazu neuartige Verteilungs- und Kommunikationsmechanismen erforderlich sind, um etwa die vorhandenen Dienste zu ermitteln und mit ihnen in Verbindung treten zu können. Dazu ist die Fähigkeit zur

spontanen Vernetzung erforderlich. Middleware- Konzepte wie Jini oder Jxta werden dabei eine wichtige Rolle spielen. In organischen Computer-Systemen werden alle Komponenten miteinander kommunizieren können, oft wird sogar die Kommunikationsfähigkeit wichtiger als die reine Rechenleistung sein. Aus Sicht von GI und ITG stellen leitungsgebundene Netze weiterhin die Basis dar, während bei Endgeräten vielfach drahtlose Verbindungen über Infrastruktur basierte Netze wie UMTS, GSM oder WLAN und infrastrukturlose Netze überwiegen. Dem Internet dürfte die Rolle des allgemeinen Bindeglieds zukommen, das die verschiedenen zugrunde liegenden Protokollansätze – etwa für Sensornetze, Fahrzeugnetze, Kleidungsnetze – überbrückt und den meisten Anwendungen eine einheitliche Kommunikationsschnittstelle anbietet. Für besonders wichtig halten wir die Programmierbarkeit und Adaptivität in zukünftigen Netzen, die eine flexible Anpassung z. B. an unterschiedliche Kostenstrukturen oder Dienstequalitäten ermöglichen. Hohe Anforderungen an Sicherheit und Vertrauenswürdigkeit Ein hohes Maß an drahtloser Kommunikation, anpassbare und personalisierte Systeme, Verfügbarkeit zu jeder Zeit an jedem Ort: die Zukunft stellt hohe Anforderungen an die Vertraulichkeit, Vertrauenswürdigkeit und Verlässlichkeit der Informationen und verlangt Schutz vor unberechtigtem Informationszugriff. Die Forschungs- und Entwicklungsarbeiten im Bereich der IT-Sicherheit konzentrieren sich auf die Abwehr von Bedrohungen sowie die Funktionssicherheit. Public-Key-Infrastrukturen, biometrische Authentifizierungsverfahren, digitale Wasserzeichen oder weiter gehende Methoden zur Erhöhung der Netzwerksicherheit sind bereits verfügbar. Für den organischen Computer sind darüber hinaus Sicherheitskonzepte erforderlich, die zu einem vertrauenswürdigen Computer führen, der insbesondere die Fähigkeit zum Selbstschutz besitzt. Die Grundlage hierfür ist ein Konzept, das von definierten Systemgrenzen ausgeht und Maßnahmen zu deren Überwachung vorsieht. Der Import und Export von Informationen wird über vertrauenswürdige Teile des Systems laufen und mit Authentifizierungen einhergehen. Was die Funktionssicherheit betrifft, werden Systeme in der Lage sein, Fehler zu erkennen, zu tolerieren und möglichst weitgehend selbst zu reparieren.

Quelle: VERBAND DER ELEKTROTECHNIK ELEKTRONIK INFORMATIONSTECHNIK e.V.

14 ÜBUNGEN

Kapitel 1

- Welche Hauptaufgaben verfolgt ein Betriebssystem

Kapitel 2

- Erklären Sie die Computergenerationen
- Benötigen alle Computer ein ausgewiesenes Betriebssystem
- Welche Generation ist Mehrprogrammbetriebfähig
- Welche Generation ist Multitaskingfähig
- Nennen Sie die Betriebsarten der Rechnersysteme
- Was ist Batch Prozessing
- Was ist ein Task
- Welche User Mode gibt es
- Was ist ein verteiltes Betriebssystem

Kapitel 3

- Welche Architektur kommt bei Embedded Systems vorwiegend zum Einsatz
- Wird für Echtzeitbetriebssysteme eher ein Monolithischer Kernel oder ein Microkernel eingesetzt

Kapitel 4

- In welche Teile kann ein Prozess aufgeteilt werden
- Was ist der Unterschied zwischen einem Programm und einem Thread
- Welche Zustände kann ein Prozess haben
- In einem Ein-Prozessor System, wie viele Prozesse können gleichzeitig verarbeitet werden
- Was unterbricht einen Prozess üblicherweise
- Was ist ein non 91reemptive Process Scheduler

Übungen

- Was zeichnet work-conserving aus
- Welches Modul steuert bei einem Kontexwechsel die Zuteilung der CPU zum aktiven Prozess
- Was ist das Zeitscheibenverfahren
- Kommen beim first-come-fitst-served Verfahren zwei Prozesse zur genau gleichen Zeit, welcher Prozess wird zuerst verarbeitet
- Können Prozesse Prioritätsgesteuert sein
- Wie nennt man den Prozess der einen Child Prozess startet
- Wird ein Child Prozess beendet, beendet sich dann auch der Parent Prozess
- Was geschieht mit den Child Prozessen wenn der Parent Prozess beendet wird
- Nennen Sie mir die zwei möglichen Benutzeridentifikationen
- Was ist eine Pipe
- Welche Operationen kennt das Betriebssystem zur Kontrolle von Prozessen

Kapitel 5

- Was ist die Besonderheit bei der Speicherverwendung von dedizierten Systemen
- In welchen Systemen kommt der Absolutlader zum Einsatz
- Was ist ein Relocating Loader
- Für was verwendet ein Programmierer die Overlay Technik bei der Speicherverwaltung
- In welchem Adressraum befindet sich der virtuelle Speicherbereich
- Welche Komponente rechnet beim Paging die virtuelle Speicheradresse in die physikalische Speicheradresse um
- Aus welchen zwei Teilen besteht die logische Adresse bei segmentierten Speicheradressen

Kapitel 6

- Welche zwei Kategorien von E/A Geräten kennen Sie
- Welche Arten von Geräteseteuerung gibt es
- In welcher Art der Gerätesteuerung wird der Carriage Return verwendet

Übungen

Kapitel 7

- Welches übergeordnetes Dateisystem kommt bei Lochkarten und Magnetbändern zum Einsatz
- Welche 6 Einträge definiert eine Datei im Hierarchischen Dateisystem
- Welchen grundlegenden Unterschied gibt es bei den Dateisystemen von Linux/Unix und Windows
- Welche Pseudodaten kennt ein Dateisystem
- Für was steht unter MS-Dos in bei der kontinuierlichen Allokation der Wert FFFFF
- Wie funktioniert die indirekte Adressierung bei Linux/Unix Systemen
- Wo werden die Einträge kleinerer Dateien unter NTFS abgelegt
- Was ist ein Symolic Link

Kapitel 8

- Was ist die Definition eines verteilten Systems nach Tannanbaum
- Peter Löhr teilte verteilte Systeme in zwei Gruppen auf. Welche sind dies
- Was klassifiziert ein Client-Server System
- Wo läuft in verteilten Anwendungssystemen eine komplexe Anwendung ab
- Was ist der Unterschied zwischen synchroner und asynchroner Kommunikation in verteilten Betriebssystemen

Kapitel 9

- Welche zwei grundlegenden Erwartungen an einen Cluster gibt es
- Ab wie vielen Nodes kann ein HA-Cluster erstellt werden
- Wie kann in einem HA-Cluster dem Split-Brain Problem entgegen gewirkt werden.
- In einem HPC-Cluster kennt der Job-Scheduler mehrere Kategorien zur Verteilung von Programmen. Nennen Sie zwei
- Was ist ein Load Balancer

Übungen

Kapitel 10

- Nenne Sie den grundlegenden Unterschied zwischen einem Supercomputer und einem Grid
- Welchen Primären Vorteil liefert ein Computer Grid
- Kann ein privater PC Mitglied eines Grid werden?
- Was ist der größte Unterschied bei der Schnittstellenverwaltung von Application Grids und Ressource Grids

Kapitel 11

- Ist die Maschinensprache eine höhere Programmiersprache?
- Welchen erstmaligen Fortschritt seit der Entwicklung der Betriebssysteme brachte das in den 60er Jahren entwickelte CTSS
- Welches Betriebssystem wird als das erste Terminal System definiert

Kapitel 12

Nicht prüfungsrelevant

Kapitel 13

- Wo wurde das erste Embedded System eingesetzt
- Sind Embedded Systems auf Standard Hardware betreibbar
- Ist ein embedded System ein Betriebssystem
- Nennen Sie drei Betriebssysteme für embedded Systems
- Ist Cloud Computing nur für den eigenen privaten Gebrauch möglich
- Was ist beim Cloud Computing der Unterschied zwischen IaaS und HaaS
- Welche Cloud Computing Form wird benötigt wenn man eine transparente Entwicklungsumgebung benötigt.
- Welche Funktion im Cloud Computing übernehmen die API´s
- Was wird von einer SaaS Cloud zur Verfügung gestellt
- Welche Verhalten werden von selbstorganisierten Systemen verlangt (bis zu 5)

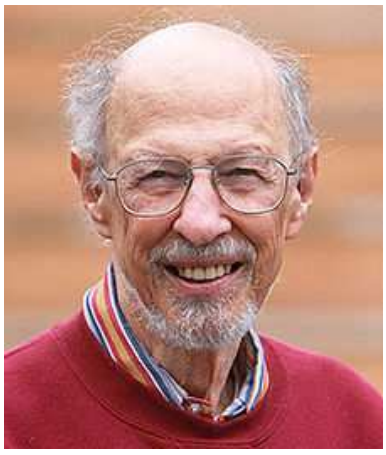
15 HALL OF FAME

Konrad Ernst Otto Zuse (* 22. Juni 1910 in Berlin; † 18. Dezember 1995 in Hünfeld bei Fulda) war ein deutscher Bauingenieur, Erfinder und Unternehmer (Zuse KG). Mit seiner Entwicklung der Z3 im Jahre 1941 baute er den ersten vollautomatischen, programmgesteuerten und frei programmierten, in binärer Gleitpunktrechnung arbeitenden Computer der Welt.

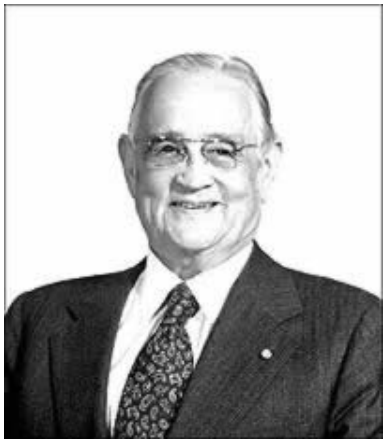
Andrew Stuart „Andy" Tanenbaum (* 1944 in New York City) ist Professor für Informatik an der Freien Universität Amsterdam (Niederlande). Tanenbaum forscht in den Bereichen Compiler, Betriebssysteme, Netzwerke sowie lokal verteilte Systeme. Bekannt wurde er vor allem als Entwickler des UNIX-artigen Systems Minix und als Autor mehrerer Fachbücher zu diversen Themen der Informatik.

Ian Foster is a Distinguished Fellow and the Associate Division Director in the Mathematics and Computer Science Division at Argonne National Laboratory, where he leads the Distributed Systems Laboratory, and he is a Professor in the Department of Computer Science at the University of Chicago. He is also involved with both the Open Grid Forum and with the Globus Alliance as an open source strategist. In 2006, he was appointed director of the Computation Institute, a joint project between the University of Chicago, and Argonne. An earlier project, Strand, received the British Computer Society Award for technical innovation.

Carl Kesselman is a project leader at the University of Southern California's Information Sciences Institute and a Research Associate Professor in Computer Science, also at the University of Southern California. Carl Kesselman is an acknowledged authority in grid computing technologies. He has co-led the Globus project, which is developing core technologies for computational grid systems in the areas of resource location, resource allocation, computer security, data communication, and data access. Most recently he received international recognition for GUSTO, the world's first high-performance computational grid. GUSTO pushes the technological envelope by using high-speed computer networks and software to provide global access to advanced supercomputers and other devices.

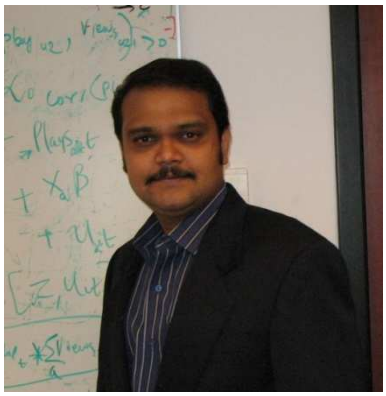


Fernando José "Corby" Corbató (born July 1, 1926 in Oakland, California) is a prominent American computer scientist, notable as a pioneer in the development of time-sharing operating systems. Amongst many awards, he received the Turing Award in 1990, "for his pioneering work in organizing the concepts and leading the development of the general-purpose, large-scale, time-sharing and resource-sharing computer systems".



Charles Stark Draper (* 2. Oktober 1901 in Windsor, Missouri; † 25. Juli 1987 in Cambridge, Massachusetts) war ein US-amerikanischer Ingenieur. Er gilt als "Vater der inertialen Navigation". Er erfand und entwickelte die Technik, die es ermöglicht, Flugzeuge, Raumfahrzeuge und Unterseeboote mittels eines Gyroskops und ähnlichen Geräten zu navigieren. Er war federführend an der Entwicklung des Navigationssystems

der Apollo-Raumfahrzeuge (Primary Guidance, Navigation and Control System) beteiligt. Charles Stark Draper wurde 1981 in die National Aviation Hall of Fame aufgenommen. Nach ihm ist der Charles-Stark-Draper-Preis benannt, der alljährlich von der United States National Academy of Engineering an verdienstvolle Ingenieure vergeben wird.



Dr. Ramnath Chellappa is currently an Associate Professor in the Information Systems & Operations Management area at the Goizueta Business School, Emory University. He was formerly a faculty member at the Marshall School of Business, University of Southern California (1997 - 2005), and also served as the Director of ebizlab (Electronic Economy Research Lab). He is also affiliated with the Center for Telecom Management at USC and the Center for Research in Electronic Commerce (formerly known as CISM) at the University of Texas at Austin. He received his PhD in Management (Information, Risk, and Operations Management) from the business school (now the McCombs School of Business) at the University of Texas at Austin in 1997-98. He has a background in engineering (Mining Engineering (1987-91) IT-BHU, India and Petroleum Engineering (1991-93), University of Texas, Austin) and has worked as a Unix and networks administrator.

William „Bill“ Henry Gates III, (* 28. Oktober 1955 in Seattle) ist ein US-amerikanischer Unternehmer und Programmierer. Bill Gates war bis zum 10. März 2010 wieder der reichste Mensch der Welt, nachdem er 2008 nur auf Platz 3 stand[1]. Seit dem 10. März 2010 steht er auf Platz 2 der reichsten Menschen. Gates gründete 1975, gemeinsam mit Paul Allen, die Microsoft Corporation. Er besitzt etwa 1,1 Milliarden Aktien von Microsoft (etwa 30 Mrd. US $), was gut 10 % des Grundkapitals entspricht, ist Aufsichtsratsvorsitzender und war bis 2006 Leiter der Entwicklungsabteilung („Chief Software Architect“) des Unternehmens. Seit Dezember 2004 ist er darüber hinaus Mitglied des Aufsichtsrats (Board of Directors) von Berkshire Hathaway. Am 12. September 2007 verabschiedete er sich auf der Unternehmensversammlung offiziell von Microsoft und der 27. Juni 2008 war sein letzter Arbeitstag dort.

Hall of Fame

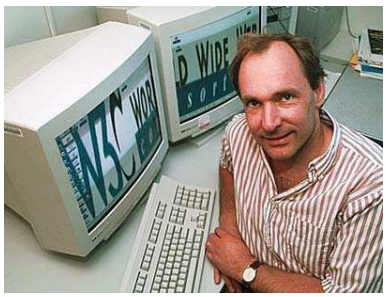
Linus Benedict Torvalds (*28. Dezember 1969 in Helsinki, Finnland) ist ein finnischer Programmierer und Initiator des Kernels von Linux, dessen Entwicklung er bis heute koordiniert.

Gary Kildall (* 19. Mai 1942 in Seattle; † 11. Juli 1994 in Monterey (Kalifornien)) war der Erfinder des Betriebssystems CP/M und Gründer von Digital Research. Kildall gilt als Schoepfer des ersten PC-Betriebssystems CP/M. Er erfand dieses nur, so Cringely in seinem Buch "Unternehmen Zukunft", weil "er es satt hatte, in die Arbeit zu fahren". Ihn stoerte, die "oede Fahrerei" zwischen seinem Zuhause in Pacific Grove ueber das Santa Cruz Gebirge ins Silicon Valley. Kildall entwickelte auch das erste BIOS (Basic Input/Output System) und ermoeglichte damit die Portierung seines CP/M-Betriebssystems auf verschiedene Hardwareplattformen.

Kenneth „Ken“ Lane Thompson (* 4. Februar 1943 in New Orleans, Louisiana, USA) und

Dennis MacAlistair Ritchie (* 9. September 1941 in Bronxville, New York) sind US-amerikanischer Informatiker. Sie entwickelten zusammen die erste Version des Unix-Betriebssystems und schrieb das erste Unix Programmer's Manual (1971). Zusammen mit Brian W. Kernighan entwickelte er die Programmiersprache C. Ritchie und Kernighan schrieben gemeinsam das Buch The C Programming Language.

Herman Hollerith (* 29. Februar 1860 in Buffalo, New York; † 17. November 1929 in Washington, D.C.) war ein US-amerikanischer Unternehmer und Ingenieur. Er ist Erfinder des nach ihm benannten Hollerith-Lochkartenverfahrens in der Datenverarbeitung und Gründer des Unternehmens IBM.



Sir Timothy John Berners-Lee, (* 8. Juni 1955 in London) ist ein britischer Informatiker. Er ist der Erfinder der HTML (Hypertext Markup Language) und der Begründer des World Wide Web. Heute steht er dem World Wide Web Consortium (W3C) vor und ist Professor am Massachusetts Institute of Technology (MIT).

Stephen Gary Wozniak (Spitzname (The) Woz oder Wizard of Woz; * 11. August 1950 in Sunnyvale, Kalifornien) ist ein US-amerikanischer Computeringenieur und Unternehmer. Er gründete 1976 zusammen mit Steve Jobs das Unternehmen Apple und war so maßgeblich am Einzug des PCs in private Haushalte beteiligt. Er erschuf unter anderem den Apple I und mit dem Apple II den letzten PC, der vollständig von einem einzelnen Entwickler entworfen wurde.[1] Außerdem erfand er das bekannte Computerspiel Breakout. Wozniak gilt wegen seiner herausragenden kreativen technischen Fähigkeiten und seiner Lebenseinstellung als typische Verkörperung des genialen Computer-Hackers. Häufig wird der Gegensatz zu seinem Weggefährten Jobs betont, der als eher erfolgs- und karriereorientiert gilt und mit dem er sich mehrfach überwarf.

William Reddington „Bill“ Hewlett (* 20. Mai 1913 in Ann Arbor, Michigan; † 12. Januar 2001 in Palo Alto, Kalifornien).

David Packard (* 7. September 1912 in Pueblo, Colorado; † 26. März 1996 in Stanford, Kalifornien) gründete zusammen mit Bill Hewlett den am 01. Januar 1939 in einer Garage in Palo Alto den US-amerikanischen Technologiekonzern Hewlett-Packard.

Matthew Gray. In Spring of 1993, he wrote the Wanderer to systematically traverse the Web and collect sites. The Wanderer was the primary tool for collection of data to measure the growth of the web. It was the first automated Web agent or "spider". The Wanderer was first functional in spring of 1993 and performed regular traversals of the Web from June 1993 to January 1996.

Sergei Michailowitsch Brin (* 21. August 1973 in Moskau) ist ein russischstämmiger Unternehmer und Mitbegründer der Suchmaschine Google

Lawrence (Larry) Edward Page (* 26. März 1973 in Ann Arbor, Michigan als Lawrence E. Page) ist ein US-amerikanischer Informatiker und Mitbegründer der Suchmaschine Google. Nach Page ist auch der PageRank-Algorithmus benannt, ein Bestandteil des Gewichtungsmechanismus dieser Suchmaschine.

16 INDEX