

Vorlesung

# Systemprogrammierung I

Wintersemester 2002/2003

Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[A-Org.fm, 2002-10-17 11.12]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

- 1

## A Organisatorisches

Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[A-Org.fm, 2002-10-17 11.12]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

A - 1

## 1 Dozent

- Prof. Dr. F. Hofmann
- Lehrstuhl für Verteilte Systeme und Betriebssysteme, Informatik 4 (Prof. Dr. F. Hofmann, Prof. Dr. Schröder-Preikschat)
  - ◆ E-mail: [fhofmann@informatik.uni-erlangen.de](mailto:fhofmann@informatik.uni-erlangen.de)

## 2 Übungsbetreuung

- Dipl.-Inf. Meik Felser — [felser@informatik.uni-erlangen.de](mailto:felser@informatik.uni-erlangen.de)
- Dr.-Ing. Jürgen Kleinöder — [kleinoeder@informatik.uni-erlangen.de](mailto:kleinoeder@informatik.uni-erlangen.de)
- Dipl.-Inf. Christian Wawersich — [wawersich@informatik.uni-erlangen.de](mailto:wawersich@informatik.uni-erlangen.de)
- Studentische Hilfskräfte

Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[A-Org.fm, 2002-10-17 11.12]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

A - 2

## 3 Studiengänge

- ★ Diplomstudiengang Informatik (3. Sem.)
- ★ Lehramtstudium Informatik
- ★ Bachelorstudiengang Computational Engineering (3. Sem.)
- ★ Diplomstudiengang Wirtschaftsinformatik (ab 5. Sem.)
- ★ Bachelorstudiengang Mathematik mit Schwerpunkt Informatik (3. Sem.)
- ★ Masterstudiengang Linguistische Informatik (3. Sem.)
- ★ Bachelorstudiengang Linguistische Informatik (5. Sem.)
- Vorlesung und Übung
  - ◆ Anrechenbare SWS: 4 SWS Vorlesung, 4 SWS Übungen

Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[A-Org.fm, 2002-10-17 11.12]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

A - 3

## 4 Inhalt

- Vorlesung
  - ◆ Grundlagen der Betriebssysteme (eingeschränkt auf Monoprozessoren)
  - ◆ Konzepte moderner Betriebssysteme
  - ◆ Beispielhafte Betrachtung von UNIX, Linux, Windows, Windows NT/2000
- Übungen
  - ◆ Umgang mit den in der Vorlesung vorgestellten Betriebssystemkonzepten
  - ◆ Betriebssystemschnittstelle des UNIX/Linux-Betriebssystems (POSIX)
  - ◆ Umgang mit sogenannten *System-Calls*
  - ◆ Praktische Arbeiten: Ausprogrammieren von Übungsaufgaben in der Programmiersprache C

## 5 Vorlesung (2)

- URL zur Vorlesung
  - ◆ [http://www4.informatik.uni-erlangen.de/Lehre/WS02/V\\_SP1/](http://www4.informatik.uni-erlangen.de/Lehre/WS02/V_SP1/)
  - ◆ hier findet man Termine, Folien zum Ausdrucken und Zusatzinformationen
- Literatur
  - ◆ A. Silberschatz; P. B. Galvin; G. Gagne: Operating System Concepts, Sixth Edition. John Wiley 2003.
  - ◆ A. S. Tanenbaum: *Modern Operating Systems*, Second Edition, Prentice Hall, Englewood Cliffs, NJ, 2001.
  - ◆ R. W. Stevens: *Advanced Programming in the UNIX Environment*. Addison-Wesley, 1992.

## 5 Vorlesung

- Termine: [Mo. von 10.15 bis 11.45 im H7](#)  
[Do. von 16.15 bis 17.45 im H7](#)
- Skript
  - ◆ zwei Alternativen:
    - Folien der Vorlesung werden im WWW zur Verfügung gestellt und können selbst ausgedruckt werden (Vorteil: evtl. Farbe)
    - Folien werden kopiert und vor der Vorlesung ausgegeben; Gutscheinverkauf, Kosten [5,00 EUR](#) für knapp 600 Folien auf ca. 150 S. (Vorteil: das Ausdrucken wird nicht vergessen)
  - ◆ weitergehende Informationen zum Nachlesen findet man am Besten in der angegebenen Literatur

## 5 Vorlesung (3)

- Rückmeldungen und Fragen
  - ◆ Geben Sie mir Rückmeldungen über den Stoff. Nur so kann eine gute Vorlesung entstehen.
  - ◆ Stellen Sie Fragen!
  - ◆ Machen Sie mich auf Fehler aufmerksam!
  - ◆ Nutzen Sie außerhalb der Vorlesung die Möglichkeit, elektronische Post zu versenden: [hofmann@informatik.uni-erlangen.de](mailto:hofmann@informatik.uni-erlangen.de) !

## 6 Übungen

- Übungsbeginn ist Montag, 21. Oktober 2002
- In der Woche vom 21. bis 25. Oktober 2002
  - ◆ Tafelübungen in Großgruppen
  - ◆ Termine
    - Mo. 10.15 – 11.45 im H7 (anstelle Vorlesung)
    - Di. 16.00 – 17.30 im H7
    - Do. 16.15 – 17.45 im H7 (anstelle Vorlesung)

## 6.1 Tafelübungen

- Termine

auf der Web-Seite zur Vorlesung

- Inhalt
  - ◆ Kurzeinführung in die Programmiersprache C
  - ◆ Besprechung von Übungsaufgaben
  - ◆ Klärung offener Fragen
  - ◆ Vermittlung ergänzender Informationen zur Vorlesung

## 6 Übungen (2)

- Danach
  - ◆ Beginn von Übungen in Kleingruppen
- Aufteilung
  - ◆ Tafelübung – 2 SWS
  - ◆ Rechnerübung – 2 SWS
- Anmeldung zur Übung und Einteilung in die Übungsgruppen
  - ◆ „login: span“ an allen CIP-Workstations der Informatik
  - ◆ Login-Freischaltung ab Di. 22.10.02, 9.00
  - ◆ Benötigte Eingaben: persönliche Daten, Matrikelnummer, Termin der gewünschten Tafelübungen
  - ◆ Jede(r) bekommt einen Übungsplatz!

## 6.2 Rechnerübungen

- Termine

stehen auf der Web-Seite zur Vorlesung

- Inhalt
  - ◆ Lösung der Übungsaufgaben (durch die Studierenden)
  - ◆ Raum 01.155 ist reserviert (Vorrang am Rechner für Übungsteilnehmer)
  - ◆ ein Übungsleiter steht für Fragen zur Verfügung

## 7 Studien- bzw. Prüfungsleistungen

- keine
  - ◆ Informatik Lehramt
- Schein (unbenotet)
  - ◆ Informatik (Diplom)
  - ◆ Mathematik mit Schwerpunkt Informatik (Bachelor)
  - ◆ Linguistische Informatik (Magister, Bachelor)
- studienbegleitende Prüfung
  - ◆ Computational Engineering (Bachelor)
  - ◆ Wirtschaftsinformatik (Diplom)

## 7.2 Studienbegleitende Prüfung

- ★ Keine verpflichtende Abgabe von Übungsaufgaben
  - ◆ aber **dringend** empfohlen
- ★ Klausur am Semesterende (120 min)
  - ◆ für die Note in der Klausur werden entsprechend Leistungspunkte bzw. Credit Points vergeben
  - ◆ Klausur enthält neben dem Stoff der Übungsaufgaben auch Stoff der Vorlesung, vor allem solchen, der mit den Übungsaufgaben in Zusammenhang steht.
- ▲ Klausurtermin für beide Klausurvarianten
  - ◆ Donnerstag, ??? (kurz nach Semesterende, Sondergenehmigung für CE)

## 7.1 Schein

- ★ Verpflichtende Abgabe von Übungsaufgaben
  - ◆ Abgabe erfolgt über ein spezielles Abgabeprogramm
  - ◆ Aufgaben werden auf Plausibilität geprüft und auf Abschreiben getestet
  - ◆ Aufgaben werden in Stichproben genauer analysiert
- ★ Klausur am Semesterende (60 min)
  - ◆ Zulassung zur Klausur nur wenn eine ausreichende Bearbeitung der Übungsaufgaben erfolgt ist (50%)
  - ◆ bei ausreichender Leistung in der Klausur wird der Schein vergeben
  - ◆ Bei guter Mitarbeit in den Übungsaufgaben ist die Klausur leicht zu bestehen.

## B Einführung

## 1 Warum Systemprogrammierung I?

- Rasche Einarbeitung in spezielle Systeme
  - ◆ MVS, BS2000, VM, Solaris, Unix, Windows NT, Windows 95/98, MS/DOS
- Strukturierung komplexer Programmsysteme
  - ◆ Unterteilung in interagierende Komponenten
- Konzeption und Implementierung spezialisierter Systeme
  - ◆ Eingebettete Systeme (*Embedded Systems*)
  - ◆ Automatisierungssysteme
- Erstellung fehlertoleranter Systeme
- Verständnis für Abläufe im Betriebssystem
  - ◆ Ökonomische Nutzung der Hardware
  - ◆ Laufzeitoptimierung anspruchsvoller Anwendungen

## 1.1 Phänomene der Speicherverwaltung (2)

- Beispiel: Initialisierung von großen Matrizen

- ◆ Variante 2:

```
#define DIM 6000

int main()
{
    register long i, j;
    static long matrix[DIM][DIM];

    for( j= 0; j< DIM; j++ )
        for( i= 0; i< DIM; i++ )
            matrix[i][j]= 1;

    exit(0);
}
```

- ◆ Schleifen sind vertauscht

## 1.1 Phänomene der Speicherverwaltung

- Beispiel: Initialisierung von großen Matrizen

- ◆ Variante 1:

```
#define DIM 6000

int main()
{
    register long i, j;
    static long matrix[DIM][DIM];

    for( i= 0; i< DIM; i++ )
        for( j= 0; j< DIM; j++ )
            matrix[i][j]= 1;

    exit(0);
}
```

## 1.1 Phänomene der Speicherverwaltung (3)

- Messergebnisse (Sun UltraSparc 1, 128M Hauptspeicher)

- ◆ Variante 1:

User time= 3,69 sec; System time= 1,43 sec; Gesamtzeit= 22,03 sec

- ◆ Variante 2:

User time= 21,86 sec; System time= 2,33 sec; Gesamtzeit= 86,39 sec

- Ursachen

- ◆ Variante 1 geht sequentiell durch den Speicher

- ◆ Variante 2 greift versetzt ständig auf den gesamten Speicherbereich zu

Beispiel: **matrix[4][4]** und die ersten fünf Zugriffe

Variante 1 

1	2	3	4	5										
---	---	---	---	---	--	--	--	--	--	--	--	--	--	--

Variante 2 

1	5			2				3				4		
---	---	--	--	---	--	--	--	---	--	--	--	---	--	--

## 1.1 Phänomene der Speicherverwaltung (4)

### ■ Ursachen

#### ◆ Logischer Adressraum

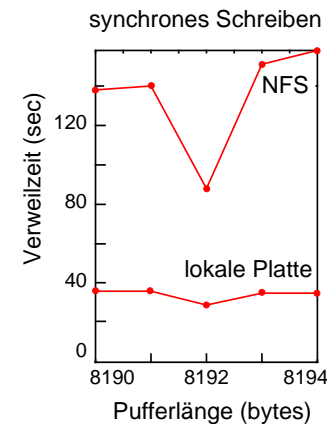
- Benutzte Adressen sind nicht die physikalischen Adressen
- Abbildung wird durch Hardware auf Seitenbasis vorgenommen (Seitenadressierung)
- Variante 2 hat weniger Lokalität, d.h. benötigt häufig wechselnde Abbildungen

#### ◆ Virtueller Speicher

- Möglicher Adressraum ist größer als physikalischer Speicher
- Auf Seitenbasis werden Teile des benötigten Speichers ein- und ausgelagert
- bei Variante 2 muss viel mehr Speicher ein- und ausgelagert werden

## 1.2 Phänomene des Dateisystems (2)

### ■ Messergebnisse



## 1.2 Phänomene des Dateisystems

### ■ Beispiel: Sequentielles Schreiben mit unterschiedlicher Pufferlänge

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#define BUFLen 8191

int main()
{
    static char buffer[BUFLen];
    int i, fd= open( "filename",
                    O_CREAT|O_TRUNC|O_WRONLY|O_SYNC,
                    S_IRUSR|S_IWUSR );

    for( i= 0; i < 1000; i++ )
        write( fd, buffer, BUFLen );

    exit(0);
}
```

## 1.2 Phänomene des Dateisystems (3)

### ■ Ursachen

- ◆ Synchrones Schreiben erfordert sofortiges Rausschreiben der Daten auf Platte (nötig beispielsweise, wenn hohe Fehlertoleranz gefordert wird – Platte ist immer auf dem neuesten Stand)
- ◆ 8192 ist ein Vielfaches der Blockgröße der Plattenblöcke
- ◆ Kleine Abweichungen von der Blockgröße erfordern bereits zusätzliche Blocktransfers

## 2 Überblick über die Vorlesung

- ★ Inhaltsübersicht
  - A. Organisation
  - B. Einführung
  - C. Dateisysteme
  - D. Prozesse und Nebenläufigkeit
  - E. Speicherverwaltung
  - F. Implementierung von Dateien
  - G. Ein-, Ausgabe
  - H. Verklemmungen
  - I. Datensicherheit und Zugriffsschutz

## 3 Was sind Betriebssysteme? (2)

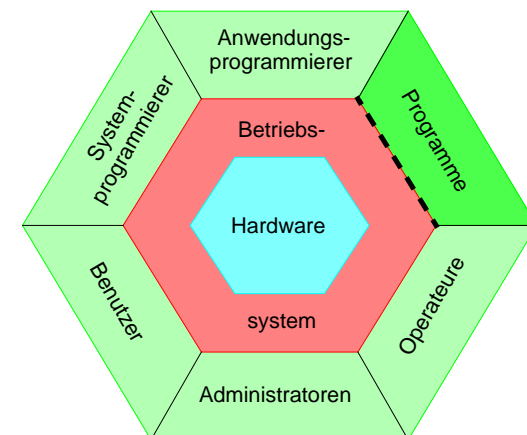
- Silberschatz/Galvin
  - ◆ „... ein Programm, das als Vermittler zwischen Rechnernutzer und Rechner-Hardware fungiert. Der Sinn des Betriebssystems ist eine Umgebung bereitzustellen, in der Benutzer bequem und effizient Programme ausführen können.“
- Brinch Hansen
  - ◆ „... der Zweck eines Betriebssystems [liegt] in der Verteilung von Betriebsmitteln auf sich bewerbende Benutzer.“
- ★ Zusammenfassung
  - ◆ Software zur Betriebsmittelverwaltung
  - ◆ Bereitstellung von Grundkonzepten zur statischen und dynamischen Strukturierung von Programmsystemen

## 3 Was sind Betriebssysteme?

- DIN 44300
  - ◆ „...die Programme eines digitalen Rechensystems, die zusammen mit den Eigenschaften der Rechenanlage die **Basis der möglichen Betriebsarten** des digitalen Rechensystems bilden und die insbesondere die **Abwicklung von Programmen steuern und überwachen**.“
- Tanenbaum
  - ◆ „...eine Software-Schicht ..., die alle Teile des Systems verwaltet und dem Benutzer eine Schnittstelle oder eine *virtuelle Maschine* anbietet, die einfacher zu verstehen und zu programmieren ist [als die nackte Hardware].“

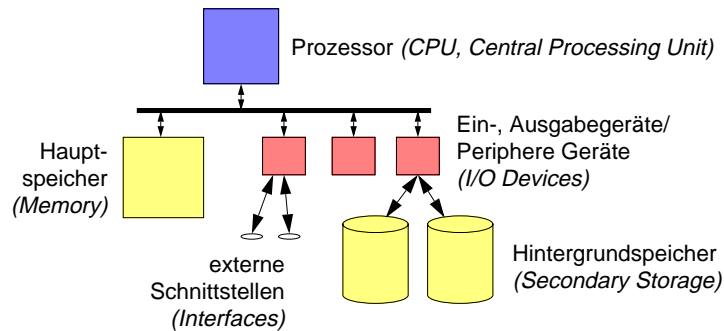
## 3 Was sind Betriebssysteme? (3)

- Strukturelle Einordnung



### 3.1 Verwaltung von Betriebsmitteln

#### ■ Betriebsmittel



### 3.2 Schnittstellen

#### ■ Betriebssystem soll Benutzervorstellungen auf die Maschinengegebenheiten abbilden

- ◆ Bereitstellung geeigneter Abstraktionen und Schnittstellen für

##### **Benutzer:**

Dialogbetrieb, graphische Benutzeroberflächen

##### **Anwendungsprogrammierer:**

Programmiersprachen, Modularisierungshilfen, Interaktionsmodelle (Programmiermodell)

##### **Systemprogrammierer:**

Werkzeuge zur Wartung und Pflege

##### **Operateure:**

Werkzeuge zur Gerätebedienung und Anpassung von Systemstrategien

### 3.1 Verwaltung von Betriebsmitteln (2)

#### ■ Resultierende Aufgaben

- ◆ Multiplexen von Betriebsmitteln für mehrere Benutzer bzw. Anwendungen
- ◆ Schaffung von Schutzumgebungen

#### ■ Ermöglichen einer koordinierten gemeinsamen Nutzung von Betriebsmitteln, klassifizierbar in

- ◆ aktive, zeitlich aufteilbare (Prozessor)
- ◆ passive, nur exklusiv nutzbare (periphere Geräte, z.B. Drucker u.Ä.)
- ◆ passive, räumlich aufteilbare (Speicher, Plattenspeicher u.Ä.)

#### ■ Unterstützung bei der Fehlererholung

### 3.2 Schnittstellen (2)

##### **Administratoren:**

Werkzeuge zur Benutzerverwaltung, langfristige Systemsteuerung

##### **Programme:**

„Supervisor Calls (SVC)“,

„Application Programmer Interface (API)“

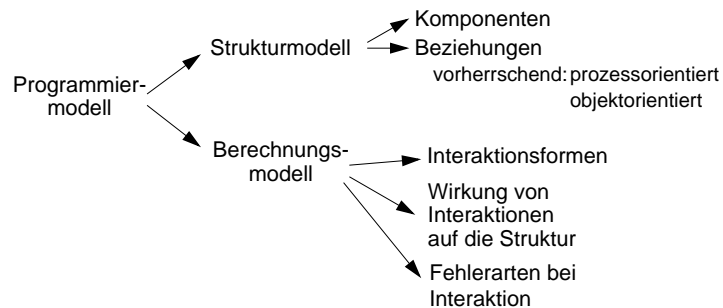
##### **Hardware:**

Gerätetreiber



### 3.3 Programmiermodelle

- Betriebssystem realisiert ein Programmiermodell
  - ◆ Keine Notwendigkeit genauer Kenntnisse über Hardwareeigenschaften und spezielle Systemsoftwarekomponenten
  - ◆ Schaffung einer begrifflichen Basis zur Strukturierung von Programmsystemen und ihrer Ablaufsteuerung



### 3.3 Programmiermodelle (3)

- Beispiele für Interaktionsformen
  - ◆ Prozedur-(Methoden-)Aufruf
  - ◆ Nachrichtenaustausch
  - ◆ Gemeinsame Speichernutzung
- Wirkung von Interaktionen auf die Struktur
  - ◆ Erzeugung und Tilgung von Prozessen
  - ◆ Instanziierung von Objekten
- Fehlerarten bei Interaktion
  - ◆ Verlust, Wiederholung oder Verspätung von Nachrichten
  - ◆ Abbruch aufgerufener Methoden, Ausnahmebehandlung

### 3.3 Programmiermodelle (2)

- Beispiele für Strukturkomponenten
  - ◆ Dateien (Behälter zur langfristigen Speicherung von Daten)
  - ◆ Prozesse (in Ausführung befindliche Programme)
  - ◆ Klassen (Vorlagen zur Bildung von Instanzen)
  - ◆ Instanzen/Objekte
  - ◆ Prozeduren
  - ◆ Sockets (Kommunikationsendpunkte, „Kommunikationssteckdosen“)
  - ◆ Pipes (Nachrichtenkanäle)
- Beispiele für Beziehungen
  - ◆ A kann B referenzieren, beauftragen, aufrufen, modifizieren
  - ◆ Pipe P verbindet A und B

### 3.4 Ablaufmodelle

- Betriebssystem realisiert eine Ablaufumgebung
- Bereitstellung von Hilfsmitteln zur Bearbeitung von Benutzerprogrammen und zur Steuerung ihrer Abläufe.
  - ◆ Laden und Starten von Programmen
  - ◆ Überwachung des Programmablaufs
  - ◆ Beenden und Eliminieren von Programmen
  - ◆ Abrechnung (*Accounting*)

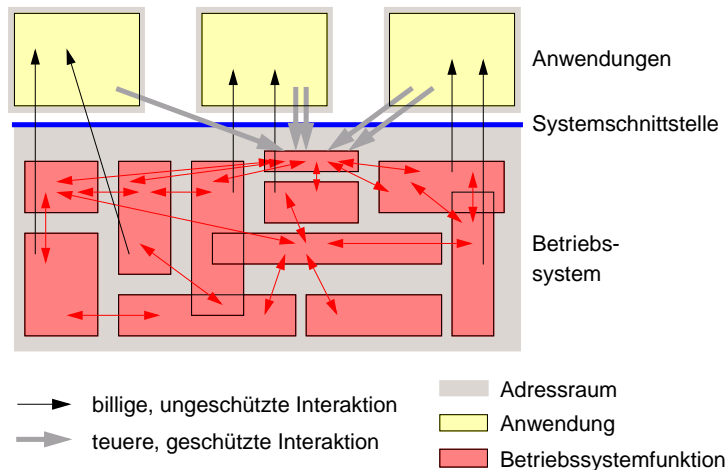
## 4 Betriebssystemarchitekturen

- Umfang zehntausende bis mehrere Millionen Befehlszeilen
  - ◆ Strukturierung hilfreich
- Verschiedene Strukturkonzepte
  - ◆ monolithische Systeme
  - ◆ geschichtete Systeme
  - ◆ Minimalkerne
  - ◆ offene objektorientierte Systeme

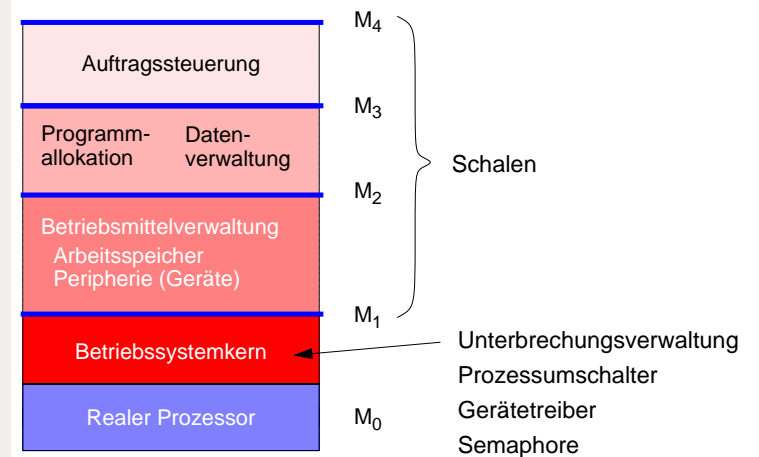
## 4.1 Monolithische Systeme (2)

- ★ Vorteile
  - ◆ Effiziente Kommunikation und effizienter Datenzugriff innerhalb des Kerns
  - ◆ Privilegierte Befehle jederzeit ausführbar
- ▲ Nachteile
  - ◆ Keine interne Strukturierung (änderungsunfreundlich, fehleranfällig)
  - ◆ Kein Schutz zwischen Kernkomponenten (Problem: zugekaufte Treiber)

## 4.1 Monolithische Systeme



## 4.2 Geschichtete Systeme



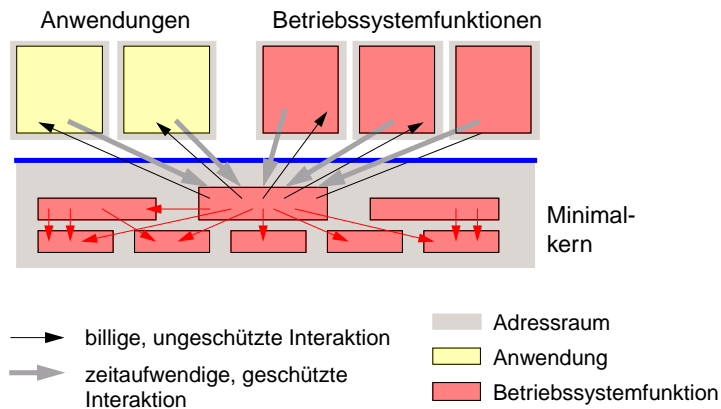
## 4.2 Geschichtete Systeme (2)

- ★ Vorteile
  - ◆ Schutz zwischen verschiedenen BS-Teilen
  - ◆ Interne Strukturierung
- ▲ Nachteile
  - ◆ Mehrfacher Schutzraumwechsel ist zeitaufwendig
  - ◆ Unflexibler und nur einseitiger Schutz (von unten nach oben)

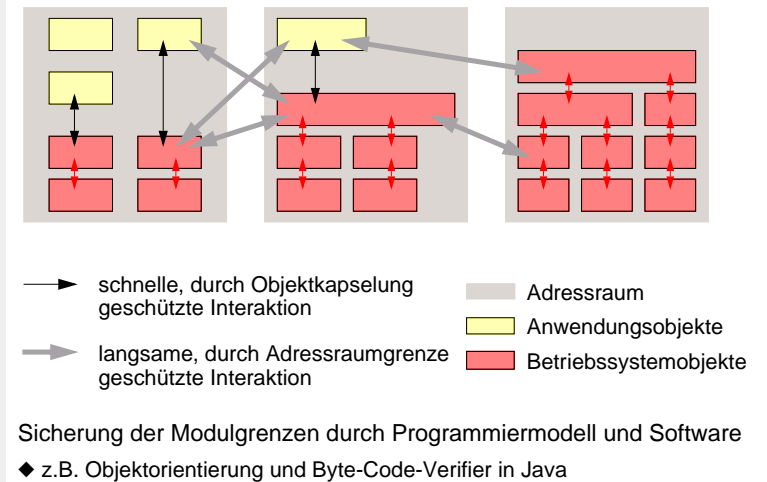
## 4.3 Minimalkerne (2)

- ★ Vorteile
  - ◆ Gute Modularisierung
  - ◆ Schutz der Komponenten voreinander
- ▲ Nachteil
  - ◆ Kommunikation zwischen Modulen ist zeitaufwendig

## 4.3 Minimalkerne



## 4.4 Objektbasierte, offene Systeme

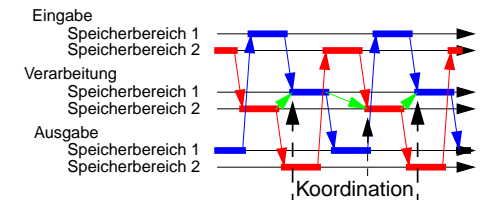


## 4.4 Objektbasierte, offene Systeme (2)

- ★ Vorteile
  - ◆ Schutz auf mehreren Ebenen (Sprache, Code-Prüfung, Adressraum)
  - ◆ Modularisierung und Effizienz möglich
- ▲ Nachteile
  - ◆ Komplexes Sicherheitsmodell

## 5.2 1960–1965

- 1960
- ◆ Autonome periphere Geräte
    - Überlappung von Programm-bearbeitung und Datentransport zwischen Arbeitsspeicher und peripheren Geräten möglich
    - Wechselpufferbetrieb (abwechselndes Nutzen zweier Puffer)



1965

## 5 Geschichtliche Entwicklung

### 5.1 1950–1960

- 1950
- ◆ Einstörmige Stapelsysteme (*Single-stream batch processing systems*)  
Aufträge zusammen mit allen Daten werden übergeben und sequentiell bearbeitet
  - ◆ Steuerung durch Auftragsabwickler (*Resident monitor, Job monitor*)  
Hilfsmittel: Assembler, Compiler, Binder und Lader, Programmbibliotheken

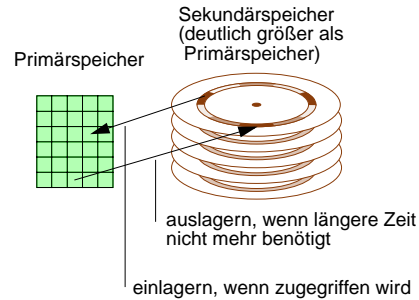
1960

### 5.2 1960-1965 (2)

- 1960
- ◆ Autonome periphere Geräte (Forts.)
    - Mehrprogrammbetrieb (*Multiprogramming*)
- 
- Spooling (*Simultaneous peripheral operation on-line*)
  - ◆ Mehrere Programme müssen gleichzeitig im Speicher sein → Auslagern von Programmen auf Sekundärspeicher
  - ◆ Programme müssen während des Ablaufs verlagerbar sein (*Relocation problem*)
  - ◆ Echtzeitdatenverarbeitung (*Real-time processing*), d.h. enge Bindung von Ein- und Ausgaben an die physikalische Zeit
- 1965

### 5.3 1965–1970

- 1965**
- OS/360
- THE
- MULTICS
- UNIX
- 1970**
- ◆ Umsetzung von Programmadressen in Speicherorte zur Laufzeit: Segmentierung, Seitenadressierung (*Paging*)
  - ◆ Virtueller Adressraum: Seitentausch (*Paging*)  
Seiten werden je nach Zugriff ein- und ausgelagert



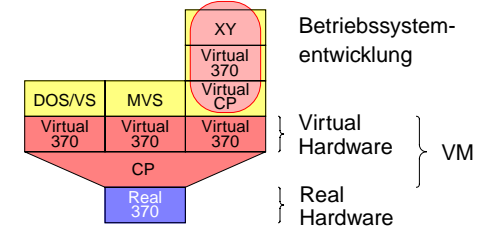
Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[B-Intro.fm, 2002-10-17 11:35]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

B – 33

### 5.4 1970–1975

- 1970**
- VM
- Hydra
- MVS
- 1975**
- ◆ Modularisierung:  
Datenkapselung, Manipulation durch Funktionen (nach Parnas)
  - ◆ Virtuelle Maschinen: Koexistenz verschiedener Betriebssysteme im gleichen Rechner



- ◆ Symmetrische Multiprozessoren: HYDRA
  - Zugangskontrolle zu Instanzen durch Capabilities
  - Trennung von Strategie und Mechanismus

Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[B-Intro.fm, 2002-10-17 11:35]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

B – 35

### 5.3 1965-1970 (2)

- 1965**
- OS/360
- THE
- MULTICS
- UNIX
- 1970**
- ◆ Interaktiver Betrieb (*Interactive processing, Dialog mode*)
  - ◆ Mehrbenutzerbetrieb, Teilnehmersysteme (*Time sharing*)
  - ◆ Problem: Kapselung von Prozessen und Dateien → geschützter Adressraum, Zugriffsschutz auf Dateien
  - ◆ Dijkstra: Programmsysteme als Menge kooperierender Prozesse (heute *Client-Server*)
  - ◆ Problem: Prozessinteraktion bei gekapselten Prozessen → Nachrichtensysteme zur Kommunikation, gemeinsamer Speicher zur Kooperation

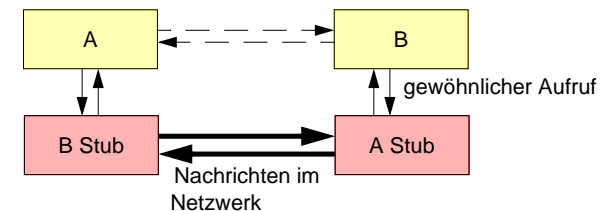
Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[B-Intro.fm, 2002-10-17 11:35]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

B – 34

### 5.5 1975–1985

- 1975**
- LOCUS
- MS-DOS
- NC
- EDEN
- 1980**
- 1985**
- ◆ Vernetzung, Protokolle (z.B. TCP/IP)
  - ◆ Verteilte Systeme
  - ◆ Newcastle Connection
  - ◆ Fernaufruf (*Remote procedure call, RPC*)

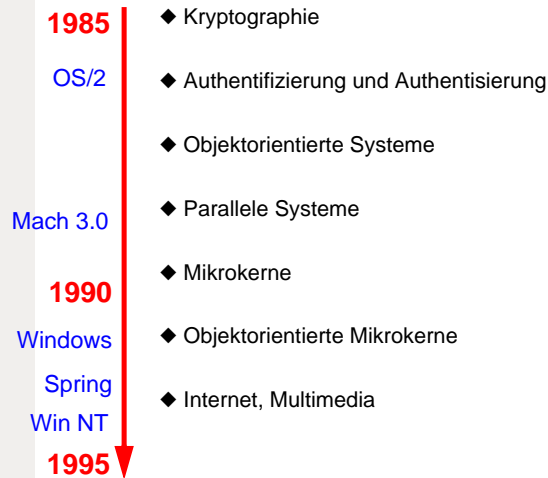


Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[B-Intro.fm, 2002-10-17 11:35]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

B – 36

## 5.6 1985–1999

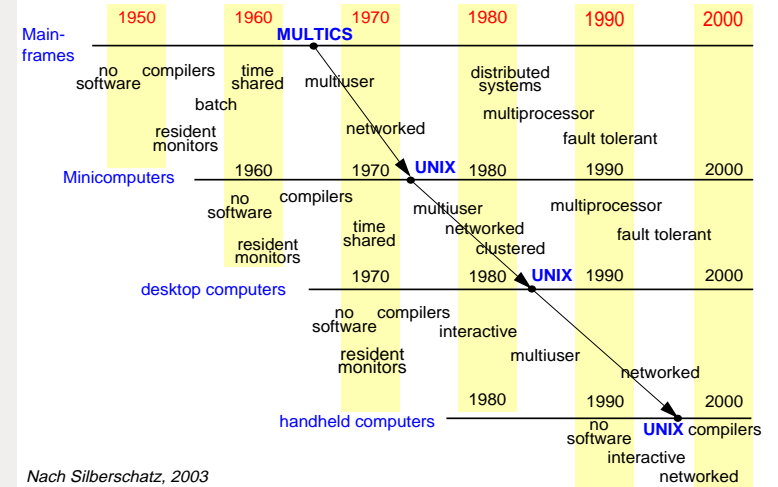


Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[B-Intro.fm, 2002-10-17 11:35]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

B – 37

## 5.8 Migration von Konzepten

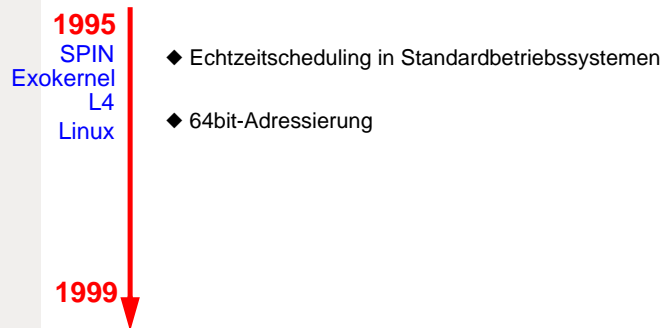


Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[B-Intro.fm, 2002-10-17 11:35]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

B – 39

## 5.7 1995–1999

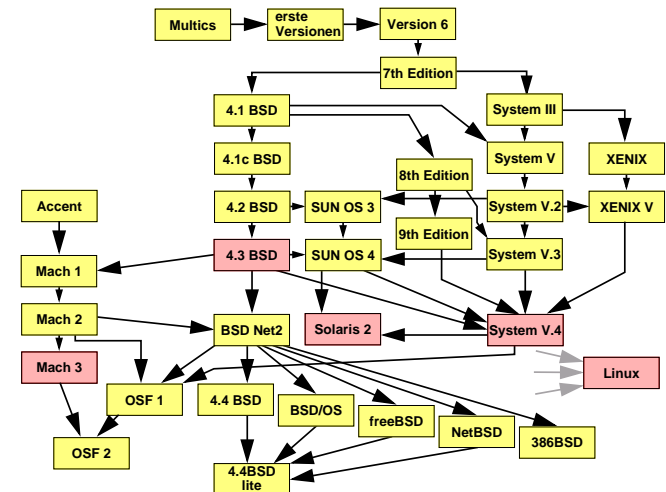


Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[B-Intro.fm, 2002-10-17 11:35]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

B – 38

## 5.9 UNIX Entwicklung



Systemprogrammierung I

© 1997-2001, Franz J. Hauck, 2002 F. Hofmann, Inf 4, Univ. Erlangen-Nürnberg[B-Intro.fm, 2002-10-17 11:35]  
Reproduktion jeder Art oder Verwendung dieser Unterlagen, außer zu Lehrzwecken an der Universität Erlangen-Nürnberg, bedarf der Zustimmung des Autors.

B – 40