12 Entwurf von Betriebssystemen

12 Entwurf von Betriebssystemen

12.1 Das Problem des Entwurfs
12.2 Schnittstellenentwurf
12.3 Implementierung
12.4 Leistungsfähigkeit
12.5 Projektverwaltung
12.6. Trends

12.1 Das Problem des Entwurfs

12.1.1 Ziele

12.1.2 Warum ist es schwierig, ein Betriebssystem zu entwerfen?

Ziele

Hauptziele von Betriebssystemen für die allgemeine Verwendung:

1. Abstraktionen definieren.
2. Primitive Operationen bereitstellen.
3. Die Isolierung sicherstellen.
4. Die Hardware verwalten.

12.2 Schnittstellenentwurf

12.2.1 Leitlinien

12.2.2 Paradigmen

12.2.3 Die Systemaufrufschnittstelle

Leitlinien

1. Einfachheit
2. Vollständigkeit
3. Effizienz

Paradigmen der Ausführung

```
main( )
{
   int ... ;


   init( );
   do_something( );
   read(...);
   do_something_else( );
   write(...);
   keep_going( );
   exit(0);
}
```

a

```
main( )
{
   mess_t msg;


   init( );
   while (get_message(&msg)) {
      switch (msg.type) {
          case 1: ... ;
          case 2: ... ;
          case 3: ... ;
      }
   }
}
```

b

Abbildung 12.1: (a) Algorithmischer Code. (b) Ereignisorientierter Code.

Tanenbaum, A. S.; Bos, H.: Moderne Betriebssysteme. Pearson Studium 2016

12.3 Implementierung

12.3.1 Systemstruktur

12.3.2 Mechanismus versus Strategie

12.3.3 Orthogonalität

12.3.4 Namensräume

12.3.5 Zeitpunkt des Bindens

12.3.6 Statische versus dynamische Strukturen

12.3.7 Top-down versus Bottom-up Implementierung

12.3.8 Synchrone versus asynchrone Kommunikation

12.3.9 Nützliche Techniken

Geschichtete Systemstruktur

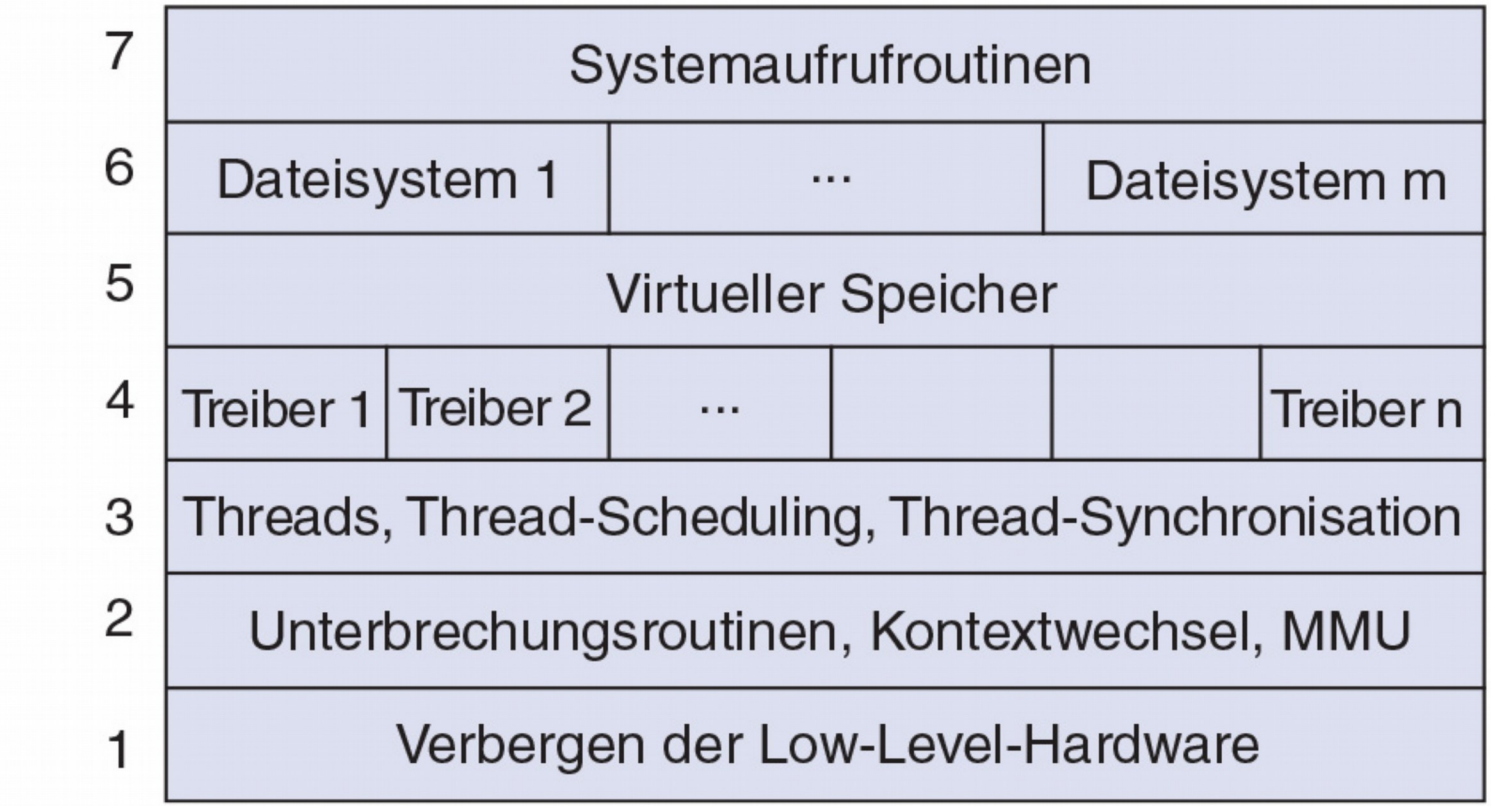

Abbildung 12.2: Möglicher Entwurf für ein modernes, geschichtetes Betriebssystem.

Tanenbaum, A. S.; Bos, H.: Moderne Betriebssysteme. Pearson Studium 2016

Mikrokern basierte Client-Server Systeme

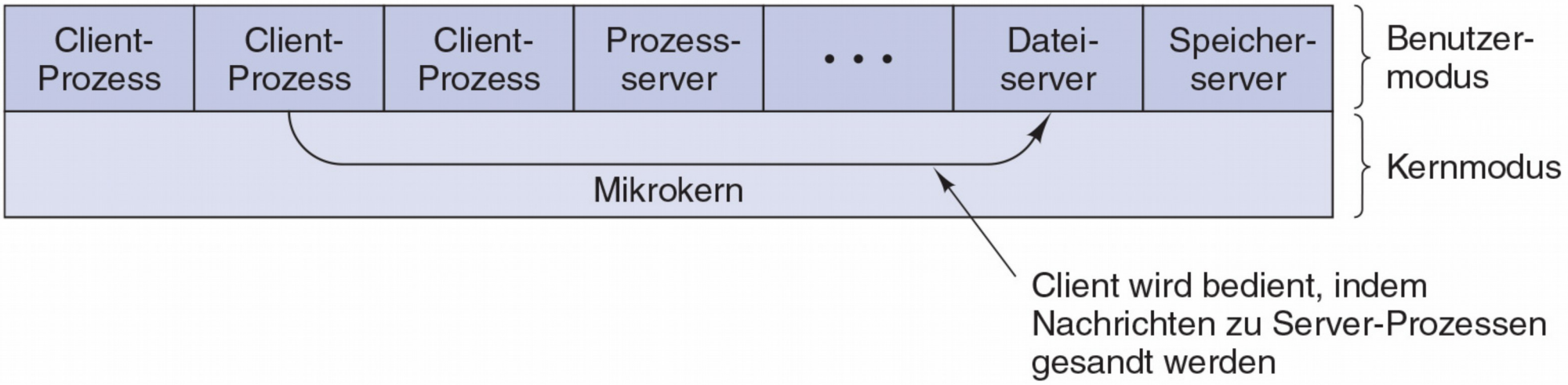

Abbildung 12.3: Client-Server-System, das auf einem Mikrokern basiert.

Tanenbaum, A. S.; Bos, H.: Moderne Betriebssysteme. Pearson Studium 2016

Namensräume

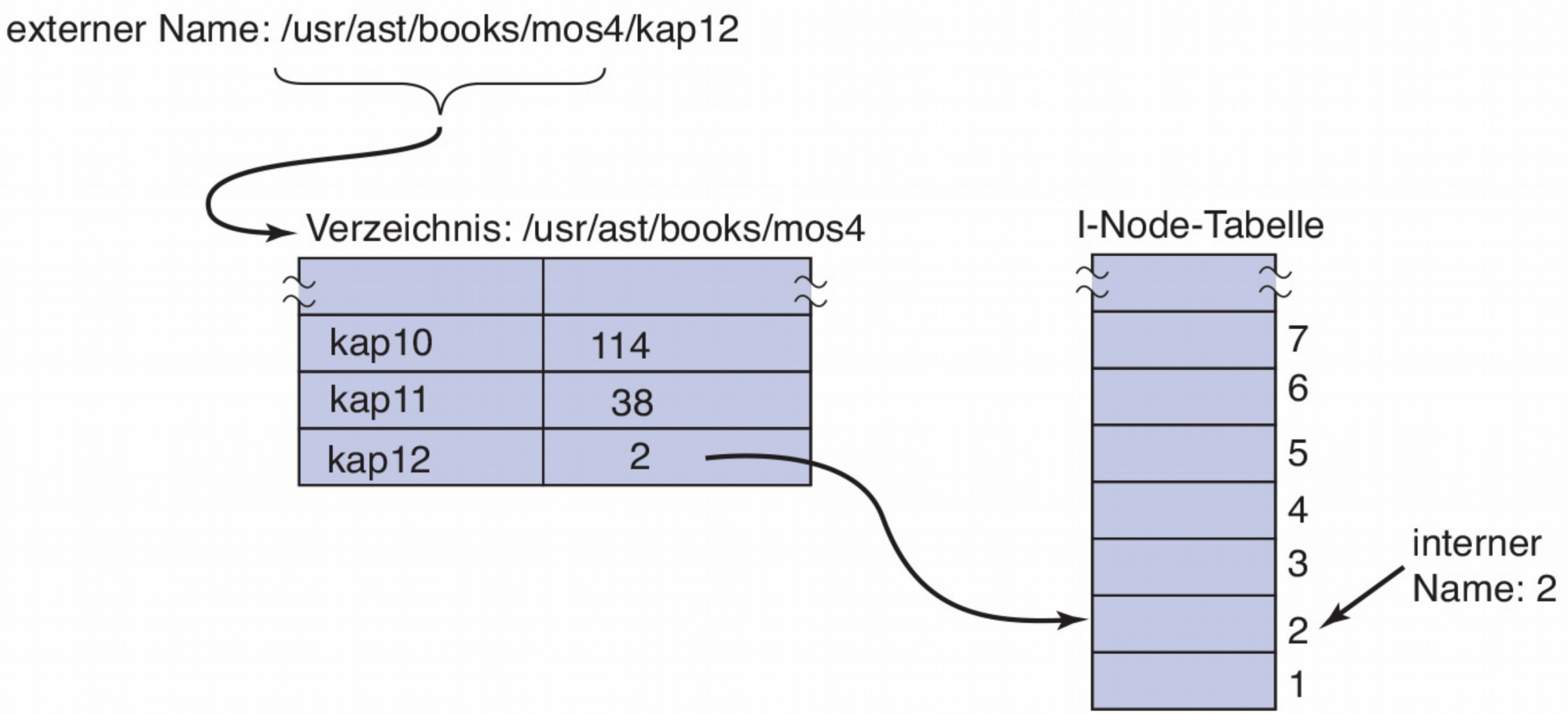

Abbildung 12.4: Verzeichnisse werden zum Abbilden der externen auf die internen Namen verwendet.

Statische versus dynamische Strukturen

```
found = 0;
for (p = &proc_table[0]; p < &proc_table[PROC_TABLE_SIZE]; p++) {
    if (p->proc_pid == pid) {
        found = 1;
        break;
    }
}
```

Abbildung 12.5: Code, um eine gegebene PID in der Prozesstabelle zu suchen.

Tanenbaum, A. S.; Bos, H.: Moderne Betriebssysteme. Pearson Studium 2016

Verbergen der Hardware

```
#define CPU IA32
#define WORD_LENGTH 32

#include "config.h"

init( )
{
#if (CPU == IA32)
/* Initialisierung für IA32 */
#endif

#if (CPU == ULTRASPARC)
/* Initialisierung für UltraSPARC */
#endif
}
```

a

```
#include "config.h"

#if (WORD_LENGTH == 32)
typedef int Register;
#endif

#if (WORD_LENGTH == 64)
typedef long Register;
#endif

Register R0, R1, R2, R3;
```

b

Abbildung 12.6: (a) CPU-abhängige bedingte Übersetzung. (b) Wortlängenabhängige bedingte Übersetzung.

Tanenbaum, A. S.; Bos, H.: Moderne Betriebssysteme. Pearson Studium 2016

12.4 Leistungsfähigkeit

12.4.1 Warum sind Betriebssysteme langsam?

12.4.2 Was sollte verbessert werden?

12.4.3 Der Zielkonflikt zwischen Laufzeit und Speicherplatz

12.4.4 Caching

12.4.5 Hints

12.4.6 Ausnutzen der Lokalität

12.4.7 Optimieren des Normalfalls

Balance zwischen Speicherplatz und Laufzeit (1)

```
#define BYTE_SIZE 8                 /* Ein Byte besteht aus 8 Bits */

int bit_count(int byte)
{                                   /* Zähle die Bits pro Byte */
     int i, count = 0;

     for (i = 0; i < BYTE_SIZE; i++)     /* Durchlaufe alle 8 */

          if ((byte >

     return(count);
}
```

```
/* Makro, um die eir
#define bit_count(b)
```

```
/* Makro zum Suchen
char bits[256] = {0,
     3, 2, 3, 3, ...
#define bit_count(b) (int) bits[b]
```

c

Abbildung 12.7: (a) Prozedur zum Zählen der Bits in einem Byte. (b) Makro zum Zählen der Bits. (c) Makro, das die Bits in einer Tabelle nachschlägt.

24 Bit

3,8,13	3,8,13	26,4,9	90,2,6
3,8,13	3,8,13	4,19,20	4,6,9
4,6,9	10,30,8	5,8,1	22,2,0
10,11,5	4,2,17	88,4,3	66,4,43

a

8 Bit

7	7	2	6
7	7	3	4
4	5	10	0
8	9	2	11

b

24 Bit

11	66,4,43
10	5,8,1
9	4,2,17
8	10,11,5
7	3,8,13
6	90,2,6
5	10,30,8
4	4,6,9
3	4,19,20
2	88,4,3
1	26,4,9
0	22,2,0

c

Abbildung 12.8: (a) Teil eines unkomprimierten Bildes mit 24 Bit pro Bildpunkt. (b) Derselbe Ausschnitt komprimiert mit GIF, mit 8 Bit pro Bildpunkt. (c) Die Farbtabelle.

Balance zwischen Speicherplatz und Laufzeit (2)

24 Bit

3,8,13	3,8,13	26,4,9	90,2,6
3,8,13	3,8,13	4,19,20	4,6,9
4,6,9	10,30,8	5,8,1	22,2,0
10,11,5	4,2,17	88,4,3	66,4,43

a

8 Bit

7	7	2	6
7	7	3	4
4	5	10	0
8	9	2	11

b

24 Bit

11	66,4,43
10	5,8,1
9	4,2,17
8	10,11,5
7	3,8,13
6	90,2,6
5	10,30,8
4	4,6,9
3	4,19,20
2	88,4,3
1	26,4,9
0	22,2,0

c

Abbildung 12.8: (a) Teil eines unkomprimierten Bildes mit 24 Bit pro Bildpunkt. (b) Derselbe Ausschnitt komprimiert mit GIF, mit 8 Bit pro Bildpunkt. (c) Die Farbtabelle.

Caching (1)

Für die Suche nach „/usr/ast/mbox“ sind folgende Festplattenzugriffe erforderlich:

1. Den i-node für das Stammverzeichnis (i-node 1) lesen.
2. Das Wurzelverzeichnis (Block 1) lesen.
3. Den i-Knoten für „/usr“ (i-node 6) lesen.
4. Das Verzeichnis „/usr“ (Block 132) lesen.
5. Den i-Knoten für „/usr/ast“ (i-node 26) lesen.
6. Das Verzeichnis „/usr/ast“ (Block 406) lesen.

Caching (2)

Pfad	I-Node-Nummer
/usr	6
/usr/ast	26
/usr/ast/mbox	60
/usr/ast/books	92
/usr/bal	45
/usr/bal/paper.ps	85

Abbildung 12.9: Teil des I-Node-Cache für ▶ *Abbildung 4.34.*

Tanenbaum, A. S.; Bos, H.: Moderne Betriebssysteme. Pearson Studium 2016

12.5 Projektverwaltung

12.5.1 Der Mythos vom Mann-Monat

12.5.2 Teamstruktur

12.5.3 Die Bedeutung der Erfahrung

12.5.4 No Silver Bullet

Stuktur des Teams

Titel	Aufgaben
Chefprogrammierer	Entwickelt das grundlegende Design und schreibt den Code
Zweiter Mann	Hilft dem Chefprogrammierer und ist Ansprechpartner
Administrator	Verwaltet die Ausstattung, Zeit, Ressourcen der gesamten Gruppe
Editor	Verbessert die Dokumentation, die vom Chefprogrammierer geschrieben werden muss
Sekretäre	Administrator und Editor brauchen Sekretäre
Programmschreiber	Verwaltet den Code und die Dokumentation
Toolverwalter	Stellt Hilfsmittel für den Chefprogrammierer bereit
Programmtester	Testet den Code des Chefprogrammierers
Sprachspezialist	Kann zeitweilig dem Chefprogrammierer bei der Sprache helfen

Abbildung 12.10: Mills Vorschlag für die Besetzung eines 10-Personen-Chefprogrammiererteams.

Tanenbaum, A. S.; Bos, H.: Moderne Betriebssysteme. Pearson Studium 2016

Die Bedeutung der Erfahrung

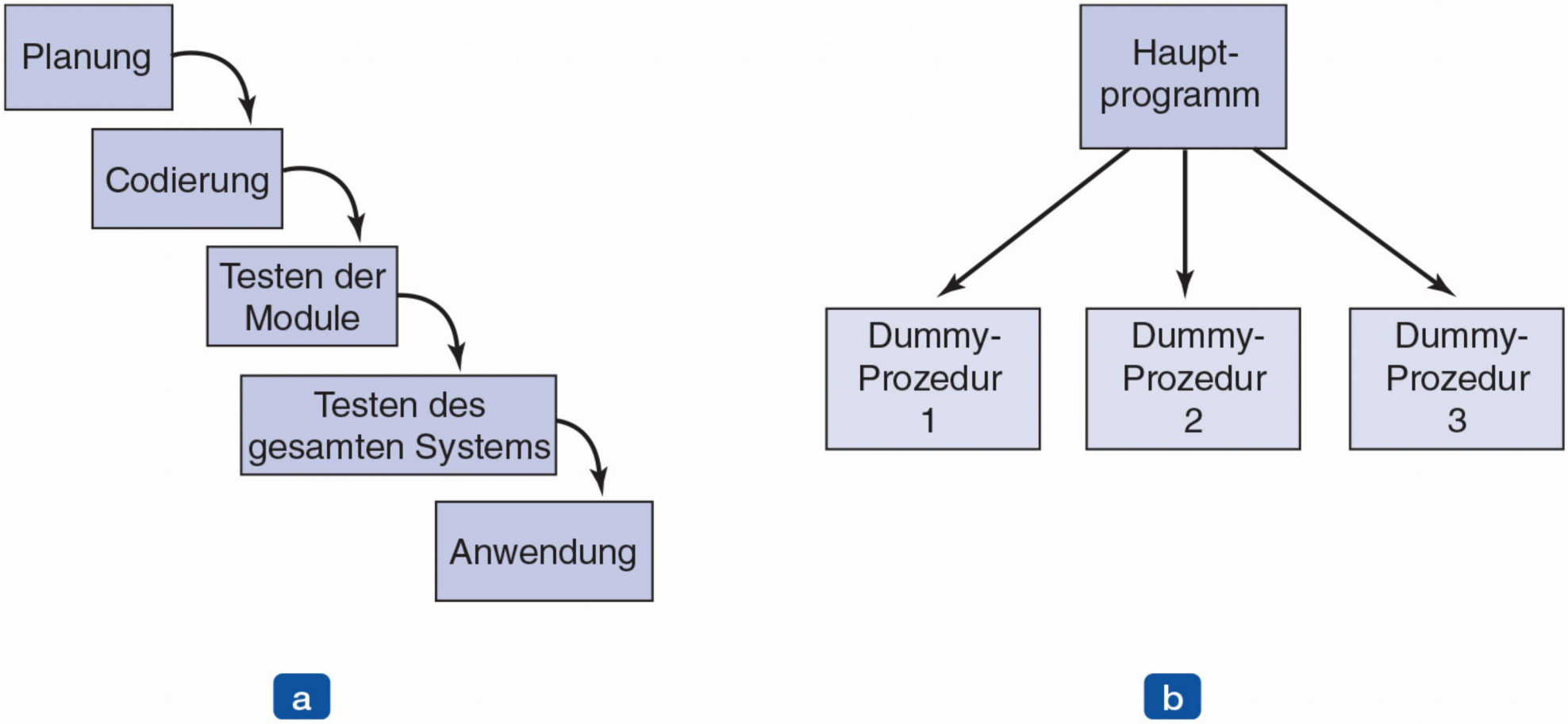

Abbildung 12.11: (a) Traditioneller Softwareentwicklungsprozess in Phasen. (b) Alternativer Entwurf, der vom ersten Tag an ein funktionierendes System (das nichts tut) produziert.

12.6 Trends beim Entwurf von Betriebsystemen

12.6.1 Virtualisierung und die Cloud

12.6.2 Vielkern-Prozessoren

12.6.3 Betriebssysteme mit großem Adressraum

12.6.4 Nahtloser Dateizugriff

12.6.5 Batteriebetriebene Computer

12.6.6 Eingebettete Systeme

Virtualisierung und die Cloud

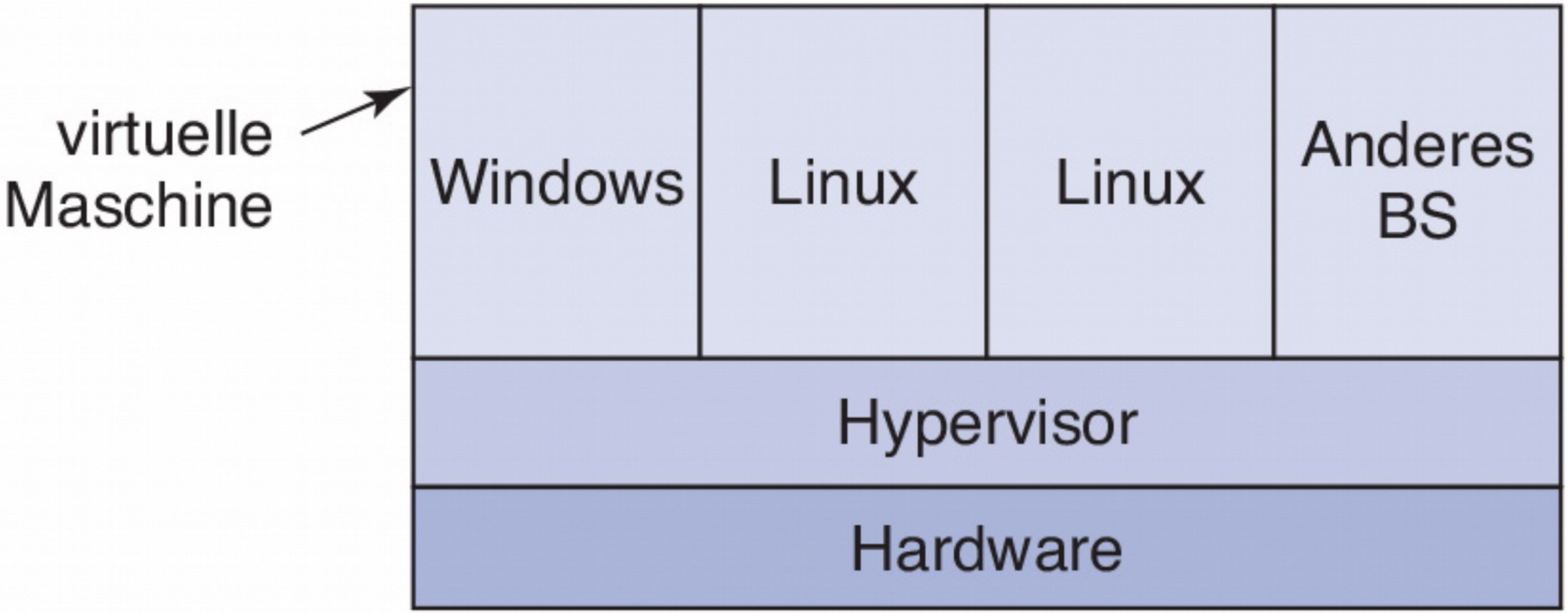

Abbildung 12.12: Ein Hypervisor, der vier virtuelle Maschinen ausführt.

Tanenbaum, A. S.; Bos, H.: Moderne Betriebssysteme. Pearson Studium 2016