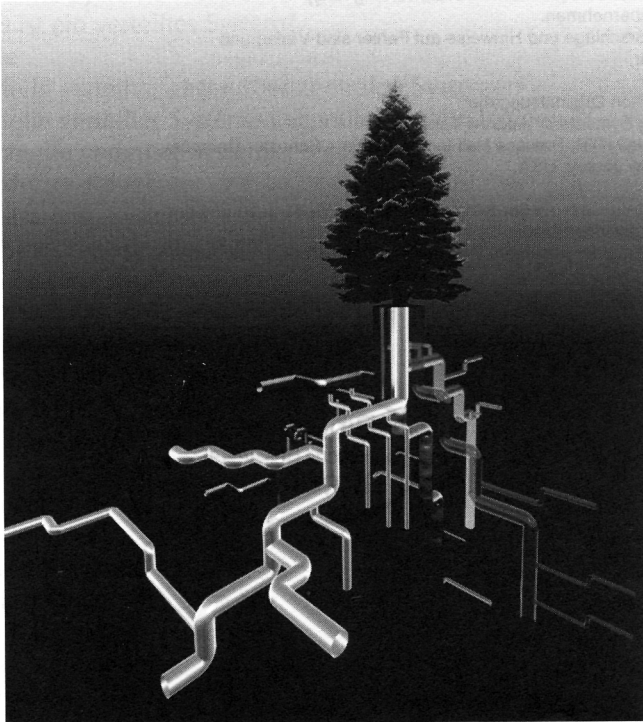


Andrew S. Tanenbaum

Verteilte Betriebssysteme



München
London
Mexiko City
New York
Singapur
Sydney
Toronto

Inhaltsverzeichnis

	Vorwort	15
1	Verteilte Systeme – Einführung	17
1.1	Was ist ein verteiltes System?	18
1.2	Ziele	19
1.2.1	Vorteile verteilter gegenüber zentralen Systemen	19
1.2.2	Vorteile verteilter Systeme gegenüber unabhängigen PCs	22
1.2.3	Nachteile verteilter Systeme	23
1.3	Hardware-Konzepte	24
1.3.1	Bus-basierte Systeme	27
1.3.2	Verbindungsorientierte Multiprozessoren	29
1.3.3	Bus-basierte Multicomputer	31
1.3.4	Verbindungsorientierte Multicomputer	31
1.4	Software-Konzepte	32
1.4.1	Netzwerk-Betriebssysteme	33
1.4.2	Echte verteilte Systeme	36
1.4.3	Multiprozessor-Timesharing-Systeme	38
1.5	Design-Aspekte	41
1.5.1	Transparenz	41
1.5.2	Flexibilität	44
1.5.3	Zuverlässigkeit	46
1.5.4	Performance	48
1.5.5	Skalierbarkeit	49
1.6	Zusammenfassung Aufgaben	51 52
2	Kommunikation in verteilten Systemen	55
2.1	Geschichtete Protokolle	55
2.1.1	Die Bitübertragungsschicht	59
2.1.2	Die Sicherungsschicht	60
2.1.3	Die Vermittlungsschicht	61
2.1.4	Die Transportschicht	62
2.1.5	Die Kommunikationssteuerungsschicht	63
2.1.6	Die Darstellungsschicht	63
2.1.7	Die Anwendungsschicht	63
2.2	Netzwerke mit asynchronem Übermittlungsverfahren	64

2.2.1	Was ist ein asynchrones Übermittlungsverfahren?	64
2.2.2	Die physikalische ATM-Schicht	67
2.2.3	Die ATM-Schicht	68
2.2.4	Die ATM-Anpassungsschicht	69
2.2.5	ATM-Vermittlung	70
2.2.6	Einige Implikationen von ATM auf verteilte Systeme	72
2.3	Das Client-Server-Modell	74
2.3.1	Clients und Server	74
2.3.2	Ein Beispiel für Client und Server	76
2.3.3	Adressierung	80
2.3.4	Blockierende und nicht-blockierende Primitiven	84
2.3.5	Puffernde und nicht-puffernde Primitiven	87
2.3.6	Zuverlässige und unzuverlässige Primitiven	89
2.3.7	Implementierung des Client-Server-Modells	91
2.4	Remote Procedure Call	95
2.4.1	Die grundsätzliche Arbeitsweise von RPC	96
2.4.2	Parameterübergabe	100
2.4.3	Dynamisches Binden	105
2.4.4	RPC-Semantik für den Fehlerfall	108
2.4.5	Implementationsaspekte	114
2.4.6	Problembereiche	126
2.5	Gruppenkommunikation	129
2.5.1	Einführung in die Gruppenkommunikation	130
2.5.2	Design-Aspekte	132
2.5.3	Gruppenkommunikation in ISIS	143
2.6	Zusammenfassung	147
	Aufgaben	148
3	Synchronisation in verteilten Systemen	151
3.1	Uhrsynchrisation	151
3.1.1	Logische Uhren	153
3.1.2	Physikalische Uhren	157
3.1.3	Algorithmen zur Uhrsynchrisierung	160
3.1.4	Die Verwendung synchronisierter Uhren	166
3.2	Gegenseitiger Ausschluß	168
3.2.1	Ein zentraler Algorithmus	168
3.2.2	Ein verteilter Algorithmus	170
3.2.3	Ein Token-Ring-Algorithmus	173
3.2.4	Ein Vergleich der drei Algorithmen	174
3.3	Wahl-Algorithmen	176
3.3.1	Der Bully-Algorithmus	176
3.3.2	Ein Ring-Algorithmus	178
3.4	Atomare Transaktionen	179

3.4.1	Einführung in atomare Transaktionen	179
3.4.2	Das Transaktionsmodell	181
3.4.3	Implementation	186
3.4.4	Nebenläufigkeitskontrolle	190
3.5	Deadlocks in verteilten Systemen	195
3.5.1	Erkennung verteilter Deadlocks	197
3.5.2	Vorbeugung verteilter Deadlocks	201
3.6	Zusammenfassung	203
	Aufgaben	204
4	Prozesse und Prozessoren in verteilten Systemen	207
4.1	Threads – Leichtgewichtsprozesse	207
4.1.1	Einführung in Threads	207
4.1.2	Die Verwendung von Threads	209
4.1.3	Design-Aspekte für Thread-Pakete	212
4.1.4	Implementierung eines Thread-Paketes	217
4.1.5	Threads und RPC	224
4.2	Systemmodelle	225
4.2.1	Das Workstation-Modell	226
4.2.2	Die Verwendung leerlaufender Workstations	229
4.2.3	Das Prozessor-Pool-Modell	234
4.2.4	Ein hybrides Modell	238
4.3	Prozessorzuteilung	239
4.3.1	Zuteilungsmodelle	239
4.3.2	Design-Aspekte für Algorithmen zur Prozessorzuteilung	241
4.3.3	Implementationsaspekte für Algorithmen zur Prozessorzuteilung	244
4.3.4	Beispiele für Algorithmen zur Prozessorzuteilung	246
4.4	Scheduling in verteilten Systemen	253
4.5	Fehlertoleranz	255
4.5.1	Komponentenfehler	255
4.5.2	Systemfehler	257
4.5.3	Synchrone und asynchrone Systeme	257
4.5.4	Ausnutzen der Redundanz	258
4.5.5	Fehlertoleranz mit Hilfe aktiver Replikation	259
4.5.6	Fehlertoleranz mit Hilfe von Primärserver-Backupserver	261
4.5.7	Übereinkunft in fehlerhaften Systemen	263
4.6	Verteilte Echtzeitsysteme	267
4.6.1	Was ist ein Echtzeitsystem?	268
4.6.2	Design-Aspekte	271
4.6.3	Echtzeit-Kommunikation	276
4.6.4	Echtzeit-Scheduling	281
	Aufgaben	288

5	Verteilte Dateisysteme	293
5.1	Entwurf verteilter Dateisysteme	294
5.1.1	Die Dateidienst-Schnittstelle	294
5.1.2	Die Verzeichnisserver-Schnittstelle	296
5.1.3	Semantik der gemeinsamen Nutzung von Dateien	301
5.2	Implementation verteilter Dateisysteme	305
5.2.1	Die Verwendung von Dateien	305
5.2.2	Systemstruktur	307
5.2.3	Caching	312
5.2.4	Replikation	318
5.2.5	Ein Beispiel: Suns Network File System	323
5.2.6	Was man daraus gelernt hat	331
5.3	Trends auf dem Gebiet der verteilten Dateisysteme	332
5.3.1	Neue Hardware	332
5.3.2	Skalierbarkeit	335
5.3.3	WANs	336
5.3.4	Mobile Benutzer	337
5.3.5	Fehlertoleranz	338
5.3.6	Multimedia	338
5.4	Zusammenfassung Aufgaben	338 339
6	Verteilter gemeinsam genutzter Speicher	343
6.1	Einführung	344
6.2	Was ist gemeinsam genutzter Speicher?	346
6.2.1	Speicher auf dem Chip	347
6.2.2	Bus-basierte Multiprozessoren	348
6.2.3	Ring-basierte Multiprozessoren	353
6.2.4	Verbindungsorientierte Multiprozessoren	356
6.2.5	NUMA-Multiprozessoren	363
6.2.6	Vergleich von Systemen mit gemeinsam genutztem Speicher	368
6.3	Konsistenz-Modelle	372
6.3.1	Strenge Konsistenz	373
6.3.2	Sequentielle Konsistenz	375
6.3.3	Kausale Konsistenz	379
6.3.4	PRAM-Konsistenz und Prozessor-Konsistenz	381
6.3.5	Schwache Konsistenz	384
6.3.6	Release-Konsistenz	387
6.3.7	Eintritts-Konsistenz	390
6.3.8	Zusammenfassung der Konsistenz-Modelle	392
6.4	Seitenbasierter verteilter gemeinsam genutzter Speicher	393
6.4.1	Das grundlegende Design	395
6.4.2	Replikation	396

6.4.3	Granularität	396
6.4.4	Die Realisierung sequentieller Konsistenz	398
6.4.5	Ermitteln des Eigentümers	400
6.4.6	Ermitteln der Kopien	403
6.4.7	Das Ersetzen von Seiten	404
6.4.8	Synchronisierung	406
6.5	Verteilter gemeinsam genutzter Speicher mit gemeinsam genutzten Variablen	407
6.5.1	Munin	407
6.5.2	Midway	416
6.6	Objekt-basierter verteilter gemeinsam genutzter Speicher	419
6.6.1	Objekte	419
6.6.2	Linda	422
6.6.3	Orca	429
6.7	Vergleich	437
6.8	Zusammenfassung Aufgaben	438 439
7	Fallstudie 1: Amoeba	443
7.1	Einführung in Amoeba	443
7.1.1	Geschichte von Amoeba	443
7.1.2	Forschungsziele	444
7.1.3	Die Amoeba-Systemarchitektur	445
7.1.4	Der Amoeba-Mikrokern	448
7.1.5	Die Amoeba-Server	450
7.2	Objekte und Capabilities in Amoeba	451
7.2.1	Capabilities	452
7.2.2	Der Schutz von Objekten	453
7.2.3	Standardoperationen	455
7.3	Prozeßverwaltung unter Amoeba	456
7.3.1	Prozesse	456
7.3.2	Threads	459
7.4	Speicherverwaltung unter Amoeba	460
7.4.1	Segmente	461
7.4.2	Abgebildete Segmente	461
7.5	Kommunikation in Amoeba	462
7.5.1	Remote Procedure Call	463
7.5.2	Gruppenkommunikation in Amoeba	467
7.5.2	FLIP – Fast Local Internet Protocol	477
7.6	Die Amoeba-Server	485
7.6.1	Der Bullet-Server	486
7.6.2	Der Verzeichnisserver	491
7.6.3	Der Replikationsserver	497

7.6.4	Der Ausführungsserver	498
7.6.5	Der Boot-Server	499
7.6.6	Der TCP/IP-Server	500
7.6.7	Andere Server	500
7.7	Zusammenfassung Aufgaben	500 501
8	Fallstudie 2: Mach	503
8.1	Einführung in Mach	503
8.1.1	Die Geschichte von Mach	503
8.1.2	Ziele von Mach	505
8.1.3	Der Mach-Mikrokern	505
8.1.4	Der Mach BSD UNIX-Server	507
8.2	Prozeßverwaltung unter Mach	508
8.2.1	Prozesse	508
8.2.2	Threads	512
8.2.3	Scheduling	516
8.3	Speicherverwaltung unter Mach	519
8.3.1	Virtueller Speicher	520
8.3.2	Gemeinsame Nutzung von Speicher	523
8.3.3	Externe Speichermanager	526
8.3.4	Verteilter gemeinsam genutzter Speicher in Mach	531
8.4	Kommunikation unter Mach	532
8.4.1	Ports	532
8.4.2	Senden und Empfangen von Nachrichten	539
8.4.3	Der Netzwerk-Nachrichtenserver	545
8.5	UNIX-Emulation unter Mach	547
8.6	Zusammenfassung Aufgaben	549 550
9	Fallstudie 3: Chorus	553
9.1	Einführung in Chorus	553
9.1.1	Geschichte von Chorus	553
9.1.2	Ziele von Chorus	555
9.1.3	Systemstruktur	555
9.1.4	Kernel-Abstraktionen	557
9.1.5	Kernelstruktur	560
9.1.6	Das UNIX-Untersystem	561
9.1.7	Das objektorientierte Untersystem	562
9.2	Prozeßverwaltung unter Chorus	562
9.2.1	Prozesse	562
9.2.2	Threads	563
9.2.3	Scheduling	565

9.2.4	Traps, Exceptions und Interrupts	566
9.2.5	Kernelaufufe für die Prozeßverwaltung	567
9.3	Speicherverwaltung unter Chorus	570
9.3.1	Bereiche und Segmente	570
9.3.2	Mapper	571
9.3.3	Verteilter gemeinsam genutzter Speicher	572
9.3.4	Kernelaufufe für die Speicherverwaltung	572
9.4	Kommunikation unter Chorus	575
9.4.1	Nachrichten	575
9.4.2	Ports	576
9.4.3	Operationen für die Kommunikation	577
9.4.4	Kernel-Aufrufe für die Kommunikation	579
9.5	UNIX-Emulation unter Chorus	581
9.5.1	Struktur eines UNIX-Prozesses	581
9.5.2	Erweiterungen für UNIX	581
9.5.3	Implementation von UNIX unter Chorus	583
9.6	COOL: Ein objektorientiertes Untersystem	589
9.6.1	Die Architektur von COOL	589
9.6.2	Die COOL-Basisschicht	590
9.6.3	Das generische Laufzeitsystem von COOL	591
9.6.4	Das Sprach-Laufzeitsystem	592
9.6.5	Implementation von COOL	592
9.7	Vergleich von Amoeba, Mach und Chorus	593
9.7.1	Philosophie	593
9.7.2	Objekte	595
9.7.3	Prozesse	596
9.7.4	Speichermodell	597
9.7.5	Kommunikation	598
9.7.6	Server	599
9.8	Zusammenfassung Aufgaben	601 602
10	Fallstudie 4: DCE	605
10.1	Einführung in DCE	605
10.1.1	Geschichte von DCE	605
10.1.2	Ziele von DCE	606
10.1.3	DCE-Komponenten	607
10.1.4	Zellen	610
10.2	Threads	612
10.2.1	Einführung in DCE-Threads	612
10.2.2	Scheduling	614
10.2.3	Synchronisierung	616
10.2.4	Thread-Aufrufe	617

10.3	Remote Procedure Call	621
10.3.1	Ziele des DCE-RPC	621
10.3.2	Schreiben von Client und Server	622
10.3.3	Binden eines Clients an einen Server	624
10.3.4	Ausführung eines RPCs	626
10.4	Zeit-Service	626
10.4.1	Das DTS-Zeitmodell	628
10.4.2	DTS-Implementation	630
10.5	Verzeichnis-Service	631
10.5.1	Namen	632
10.5.2	CDS (Cell Directory Service)	634
10.5.3	GDS (Global Directory Service)	637
10.6	Sicherheits-Service	641
10.6.1	Das Sicherheits-Modell	643
10.6.2	Sicherheits-Komponenten	645
10.6.3	Tickets und Authentifikatoren	646
10.6.4	Authentifizierte RPCs	647
10.6.5	ACLs – Access Control Lists	651
10.7	Verteiltes Dateisystem	652
10.7.1	DFS-Schnittstelle	654
10.7.2	DFS-Komponenten im Server-Kernel	656
10.7.3	DFS-Komponenten im Client-Kernel	659
10.7.4	DFS-Komponenten im Benutzer-Adreßraum Aufgaben	661 663
11	Literaturhinweise und Bibliographie	665
11.1	Empfohlene Literatur	665
11.1.1	Einführungen und allgemeine Arbeiten	665
11.1.2	Kommunikation in verteilten Systemen	666
11.1.3	Synchronisation in verteilten Systemen	667
11.1.4	Prozesse und Prozessoren in verteilten Systemen	668
11.1.5	Verteilte Dateisysteme	669
11.1.6	Verteilter gemeinsam genutzter Speicher	670
11.1.7	Fallstudie1: Amoeba	670
11.1.8	Fallstudie2: Mach	671
11.1.9	Fallstudie3: Chorus	672
11.1.10	Fallstudie4: DCE	672
11.2	Alphabetische Bibliographie	673
	Stichwortverzeichnis	695