

Das OSF Distributed Computing Environment

Grundlagen und Anwendung

Springer

Berlin

Heidelberg

New York

Barcelona

Budapest

Hongkong

London

Mailand

Paris

Santa Clara

Singapur

Tokio

Alexander Schill

Das OSF Distributed Computing Environment

Grundlagen und Anwendung

2., überarbeitete
und erweiterte Auflage

Mit 128 Abbildungen



Springer

Prof. Dr. Alexander Schill
Technische Universität Dresden
Fakultät Informatik
D-01062 Dresden

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Schill, Alexander:

Das OSF-distributed-computing-Environment : Einführung
und Grundlagen / A. Schill. - 2., überarb. und erw. Aufl. -
Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ;
Hongkong ; London ; Mailand ; Paris ; Santa Clara ; Singapur ;
Tokio : Springer, 1997

1. Aufl. u.d.T.: Schill, Alexander: DCE - das OSF-distributed-
computing-Environment
ISBN-13:978-3-642-64531-0 e-ISBN-13:978-3-642-60731-8
DOI: 10.1007/978-3-642-60731-8

ISBN-13:978-3-642-64531-0 Springer-Verlag Berlin Heidelberg New York

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

© Springer-Verlag Berlin Heidelberg 1993, 1997
Softcover reprint of the hardcover 2nd edition 1997

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, daß solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Satz: Reproduktionsfertige Vorlage vom Autor
Umschlaggestaltung: Künkel + Lopka, Heidelberg
SPIN: 10558615 33/3142-5 4 3 2 1 0 - Gedruckt auf säurefreiem Papier

Vorwort

Das *Distributed Computing Environment (DCE)* der *Open Software Foundation (OSF)* stellt eine Reihe von Softwarekomponenten bereit, um die Erstellung verteilter Anwendungsprogramme auf heterogenen Rechnernetzen zu erleichtern. Dieses Buch hat zum Ziel, dem Leser einen detaillierten, praxisnahen Überblick über das OSF DCE zu vermitteln und ihm dabei konkrete Hilfestellungen beim Einsatz dieser Systemumgebung zur Entwicklung verteilter Anwendungen zu geben. Das Buch wendet sich zum einen an Fachleute auf dem Gebiet der Informatik und Telematik, die sich besonders für den Entwurf und die Realisierung verteilter Anwendungen interessieren, also etwa an Projektleiter, Software-Ingenieure und spezialisierte Programmierer. Zum anderen richtet sich das Buch aber auch an das technische Management, das besonders an einer gut überschaubaren und anschaulich gestalteten Einführung interessiert ist, um in kurzer Zeit eine Bewertung komplexer Systemkonzepte im Hinblick auf die eigenen Anforderungen vornehmen zu können. Im Bereich der Lehre möchte das Buch Dozenten und Studenten der praktischen Informatik ansprechen, die sich mit verteilten Rechnersystemen befassen und das DCE als Beispiel für eine dedizierte Systemarchitektur näher betrachten wollen.

Das Buch gibt zunächst einen Überblick über Eigenschaften, Vorteile und Probleme verteilter Anwendungen. Daraus wird der Bedarf nach weitergehender Systemunterstützung bei der Anwendungsentwicklung abgeleitet, wie sie vom OSF DCE geboten wird. Anschließend wird die Gesamtarchitektur des DCE vorgestellt und aufgezeigt, wie das System für eine konkrete industrielle Anwendung eingesetzt werden kann.

Die nachfolgenden Schwerpunktkapitel befassen sich dann anhand dieser Anwendung mit den Teilkomponenten des DCE, ihrer Konzepte und Funktionalität, ihrer Programmierschnittstelle und ihrer internen Realisierung. Zunächst wird ausführlich die Entwicklung von Anwendungen mit dem Remote Procedure Call (RPC) des DCE, auch unter Verwendung nebenläufiger Prozesse und einer verteilten Namensverwaltung, beschrieben. Zum RPC werden auch die Ergebnisse erster Leistungsanalysen am realen System diskutiert. Anschließend werden die Details der wichtigsten unterstützenden DCE-Komponenten (insbesondere des Directory Service, des Security Service und des Distributed Time Service) vorgestellt. Danach wird die verteilte Dateiverwaltung mit dem DCE Distributed File System beschrieben und aufgezeigt, wie auch plattenlose Workstations und PCs in eine DCE-Umgebung eingebettet werden können. Das Buch schließt mit einer Bewertung der DCE-Konzepte und zeigt aktuelle Weiterentwicklungsmöglichkeiten, z.B.

in Richtung der verteilten objektorientierten Programmierung, sowie Bezüge zu Standards auf, beispielsweise ISO/OSI, Open Distributed Processing, der Object Management Group und Unix International.

Eine besondere praktische Bedeutung hat das DCE vor allem dadurch, daß es herstellerunabhängig einsetzbar ist und deshalb die Kommunikation und Datenverwaltung in heterogenen offenen Systemen ermöglicht. Außerdem bietet es sehr viel mehr als reine Kommunikationsmechanismen an; bereits oben wurden Namensverwaltung, Sicherheit und Dateiverwaltung genannt. Nicht zuletzt ist das DCE heute eines der wenigen verteilten Systeme mit dieser Funktionalität, die in Produktqualität verfügbar und damit auch kommerziell breit einsetzbar sind.

Bei der Darstellung wird großer Wert auf konzeptionelle Klarheit gelegt; aus diesem Grunde werden nicht nur die einzelnen Komponenten des OSF DCE detailliert vorgestellt, sondern es werden auch die zugrundeliegenden Konzepte beschrieben. Obwohl häufig Programmierbeispiele zur praktischen Illustration verwendet werden, kann das Buch aber nicht Programmierhandbücher zum OSF DCE vollständig ersetzen, sondern versteht sich als tutorielle Grundlage, um beim Leser die Basis für das entsprechende weiterführende Material zu schaffen. Allerdings werden dar die wichtigsten ausgewählten Bereiche der Anwendungsentwicklung mit dem DCE in ihrer Gesamtheit dargestellt, so daß Hintergrundliteratur nur für spezielle Aspekte und nicht für grundsätzliche Entwicklungsschritte herangezogen werden muß.

Bei der Beschreibung der Programmbeispiele wurde Wert auf eine verständliche, kompakte Darstellung gelegt. Dies bedeutet, daß die Kernfunktionalität jeweils vollständig vorgestellt wird, daß jedoch auf andere, in der Praxis wichtige, aber zum Grundverständnis unwesentliche Details wie etwa eine vollständige Speicher-verwaltung oder Fehlerbehandlung verzichtet wird. Ebenso werden nicht alle DCE-Funktionen ausführlich dargestellt (dies würde ein Vielfaches des Buchumfangs erfordern), sondern es erfolgt eine pragmatische Fokussierung auf die aus Sicht des Autors wesentlichsten Funktionen für gängige verteilte Anwendungen. Die Programmbeispiele wurden alle in einer realen DCE-Umgebung getestet (z.T. nach Vervollständigung um allgemeine Variablen- und Typdefinitionen, die aus Platzgründen im Buch an einigen Stellen weggelassen werden müssen). Für die Programmkommentare wurde der einfachen Darstellung halber die in C++ übliche Notation ("//...") verwendet, die allerdings nicht von üblichen C-Compilern akzeptiert wird. Für praktische Arbeiten mit den Programmstücken sind die Kommentare also entsprechend anzupassen.

Grundsätzlich beruhen die qualitativen Aussagen auf Erfahrungen mit realen Anwendungsimplementierungen, die in der eigenen Arbeitsgruppe durchgeführt wurden. Bei der verwendeten Version des OSF DCE handelt es sich um die *Revision 1.0*, die uns in der vorliegenden Form etwa Mitte 1992 verfügbar wurde. Da sich Funktionsschnittstellen, Konstantendefinitionen, Include-Files usw. im Rahmen von Folgeversionen sicherlich leicht ändern werden, kann die Funktionstüchtigkeit der beschriebenen Programmbeispiele beim Einsatz anderer DCE-Versionen beeinträchtigt werden; hierauf sei an dieser Stelle ausdrücklich hingewiesen.

Die Programmbeispiele richten sich vorwiegend an den Anwendungsentwickler. Auf den Bereich der DCE-Systemverwaltung wird aus Platzgründen meist nur in

kompakter Form auf konzeptioneller Ebene eingegangen; wo es sinnvoll scheint, wird dies durch einige Beispiele zum Einsatz von DCE-Dienstprogrammen ergänzt. Zur Erlernung der DCE-Systemverwaltung im Detail ist in jedem Fall auf die sehr umfangreiche Dokumentation des DCE Administration Guide zurückzugreifen [OSF5].

Die Darstellung der technischen Eigenschaften und der Programmierschnittstellen des DCE basiert auf der umfangreichen, aktuellen DCE-Dokumentation [OSF1-OSF6] sowie weiterführenden Quellen [LOC94, PET95, ROT93, SHM94]. Teilweise wird auf Beispiele und Konzepte aus dem vom Autor mit veröffentlichten Lehrbuch [MÜS92] zurückgegriffen. Hinsichtlich der einzelnen DCE-Komponenten wird auch auf wissenschaftliche Veröffentlichungen zu den zugrundeliegenden Systemen Bezug genommen, um die Grundkonzepte und ihre Entwicklung möglichst umfassend zu erläutern.

Das Buch entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter und Hochschulassistent am Institut für Telematik der Universität Karlsruhe. An dieser Stelle möchte ich dem Institutsleiter, Herrn Prof. Krüger ganz herzlich dafür danken, daß er dieses Buchprojekt durch die Gewährung entsprechender Freiräume bei meiner wissenschaftlichen Tätigkeit ermöglicht hat. Dem Springer-Verlag, namentlich Herrn Rossbach, sei für die Unterstützung bei der Vorbereitung des Buches ausdrücklich gedankt. Auch meinen Kollegen, namentlich Herrn Merz und Herrn Zeidler, möchte ich für ihre Hilfe beim Einsatz des DCE und vor allem bei der Durchführung des DCE-Systemmanagements meinen Dank aussprechen. Herrn Nonnenmacher danke ich für seinen Einsatz beim Test aller beschriebenen Programmstücke und bei der Durchführung von Leistungsmessungen sowie auch für wichtige inhaltliche Anmerkungen ganz herzlich. Dem Campusnahen Forschungszentrum (CEC) der Firma DEC in Karlsruhe, speziell Herrn Dr. Heuser, danke ich ganz besonders für die frühe Verfügbarkeit der Prototyp-Version des DCE. Herrn Stransky von der iXOS Software GmbH gebührt mein umfassender Dank für die Bereitstellung aktueller Informationen zum DCE sowie vor allem für die fachkundige Korrektur des Manuskripts. Nicht zuletzt gilt mein Dank natürlich auch meiner Ehefrau Ursula, deren Unterstützung ich am meisten zu schätzen wußte.

Karlsruhe, im Februar 1993

Alexander Schill

Vorwort zur zweiten Auflage

Die erste Auflage dieses Buches stieß auf erfreuliches Interesse bei Anwendern und Forschungseinrichtungen, was nun eine Neuauflage erforderlich macht. Dies gibt auch die Gelegenheit, neue Erfahrungen aus eigenen Vorlesungen, Seminaren und Praktika zu DCE mit einfließen zu lassen und aktuelle Entwicklungen zu berücksichtigen.

Mittlerweile konnte sich DCE weiter zu einem wichtigen De-facto-Standard für die verteilte Verarbeitung etablieren und erfuhr auch einige wesentliche Funktionalitätserweiterungen, auf die genauer eingegangen wird. Beispiele sind die Integration asymmetrischer Kryptoverfahren in den DCE Security Service und die Bereitstellung objektorientierter Programmierschnittstellen für DCE auf C++-Basis. Auch einschlägige Produkte zur verteilten Transaktionsverarbeitung auf DCE-Basis wie etwa Encina werden aufgrund ihrer besonderen Bedeutung für kommerzielle DCE-Anwendungen nun ausführlicher behandelt.

Mit OMG CORBA (Common Object Request Broker Architecture) existiert ferner eine mit DCE vergleichbare Lösung, die in den letzten Jahren stark an Bedeutung gewonnen hat. Aus diesem Grunde wird CORBA nun ebenfalls mehr Platz eingeräumt, um die wichtigsten Eigenschaften genauer darzustellen und einen konkreten Vergleich mit DCE zu ziehen.

An dieser Stelle möchte ich die Gelegenheit nutzen, allen Kollegen recht herzlich zu danken, die mit zum Erscheinen dieser Neuauflage beitrugen. Zunächst gilt mein ganz besonderer Dank Herrn Thoralf Reichelt, der die Erstellung und Überprüfung von Beispielen, die Informationsrecherche und die Aufbereitung des Textes unermüdlich und mit ganz außergewöhnlichem Engagement unterstützte. Ferner möchte ich Herrn Sascha Kümmel ganz herzlich für seine umfangreichen Arbeiten zur Leistungsanalyse von DCE danken. Herr Dr. Christian Mittasch und Herr Wolfgang König befaßten sich intensiv mit der Entwicklung von DCE-Anwendungen und mit der Weiterentwicklung unserer objektorientierten DCE-Erweiterung DC++, wofür Ihnen besonderer Dank gebührt. Ebenfalls möchte ich allen Mitarbeitern meiner Dresdner Forschungsgruppe für ihr großes Engagement und ihre ausgesprochen kollegiale Arbeitsweise ausdrücklich danken. Nicht zuletzt gilt mein Dank natürlich auch meiner Ehefrau Ursula, die große Geduld mit meinen außerplanmäßigen Aktivitäten wie etwa dieser Neuauflage hatte.

Inhaltsverzeichnis

1	Einführung	1
1.1	Einleitung	1
1.2	Beispiel einer verteilten Anwendung	2
1.3	Vorteile verteilter Anwendungen	5
1.4	Spezielle Probleme bei verteilten Anwendungen	6
1.5	Lösungsansätze durch das OSF DCE	7
1.6	Gliederung des Buches	9
2	Gesamtarchitektur des OSF DCE	11
2.1	Architekturmodell	11
2.2	Struktur von DCE-Cells	12
2.3	Die Teilkomponenten des DCE	13
2.3.1	Thread Service	13
2.3.2	Remote Procedure Call	15
2.3.3	Directory Service	18
2.3.4	Security Service	21
2.3.5	Distributed Time Service	24
2.3.6	Distributed File System	26
2.3.7	Diskless Support Service	28
2.4	Zusammenwirken der DCE-Komponenten	29
2.5	DCE-Systemkonfigurationen	31
2.5.1	Grundprinzipien	31
2.5.2	Beispiele für konkrete Cell-Konfigurationen	32
2.6	Einsatz am Beispiel der verteilten Anwendung	33
2.7	Bedeutung der DCE-Komponenten für Anwender	35
2.8	Bisherige Entwicklung des DCE	38
3	Der Remote Procedure Call des DCE	41
3.1	Wichtige Eigenschaften des DCE RPC	41
3.2	Grundlegender Ablauf eines DCE RPC	43
3.3	Grundfunktionalität: Programmierbeispiel	46
3.3.1	Schnittstellenbeschreibung	46
3.3.2	Implementierung des Servers	50
3.3.3	Implementierung des Clients	54
3.4	Beschreibung von RPC-Schnittstellen durch IDL	55
3.4.1	Datentyp-Definitionen	56
3.4.2	Attribute für Schnittstellen und Prozeduren	59
3.5	Die Steuernotation ACF	61

3.6	Der RPC-Bindevorgang: Details	62
3.6.1	Detaillierter Ablauf des Bindevorgangs	62
3.6.2	Datenstrukturen für das Binden: Binding Handles	64
3.6.3	Methoden des Bindens	66
3.6.4	Die RPC-Client-Schnittstelle des Directory Service	68
3.6.5	Bereitstellen und Anfordern von Ressourcen	71
3.6.6	Registrierung und Einsatz von Servergruppen	76
3.6.7	Verwendung von Directory-Suchpfaden	79
3.6.8	Bindeinformation als String für Testläufe	81
3.6.9	Statische RPC-Endpunkte	84
3.7	RPC mit Authentisierung und Autorisierung	85
3.7.1	Überblick	85
3.7.2	Schnittstellenfunktionen	86
3.8	Spezielle Laufzeitmechanismen	90
3.8.1	Fehlerbehandlung	90
3.8.2	Pipes	91
3.8.3	Rückwärtige Aufrufe	96
3.8.4	Kontext zwischen Client und Server	98
3.8.5	Einsatz von Threads	101
3.9	Leistungsmessungen zum DCE RPC	103
3.9.1	Basisexperimente	103
3.9.2	Spezielle Durchsatzuntersuchungen	105
3.10	DCE-Administrationsaufgaben	108
4	DCE-Threads als Basismechanismus	111
4.1	Wichtige Eigenschaften von DCE-Threads	111
4.2	Einsatzmöglichkeiten und Verarbeitungsmodelle	112
4.2.1	Nebenläufigkeit unabhängiger Aufträge	113
4.2.2	Auftragsinterne Nebenläufigkeit	114
4.2.3	Pipelining	115
4.3	Elementare Thread-Verwaltung	115
4.4	Thread-Synchronisation	118
4.4.1	Semaphore	118
4.4.2	Erweiterung: Rekursive Semaphore	120
4.4.3	Bedingungsvariablen	121
4.5	Thread-Attribute und Scheduling-Strategien	124
4.6	Spezielle Aspekte von DCE-Threads	127
4.6.1	Aufruf von E/A- und Betriebssystem-Routinen	127
4.6.2	Non-reentrant Software und globale Semaphore	129
4.6.3	Verklebungen	129
5	Verteilte Namensverwaltung im DCE	131
5.1	Verteilte Namensverwaltung: Übersicht	131
5.1.1	Aufgaben und Eigenschaften	131
5.1.2	Resultierende Anforderungen	132
5.2	Konzeptionelle Grundlagen von CDS und GDS	133
5.2.1	Gesamtarchitektur	133
5.2.2	Namensstruktur	134
5.2.3	Junctions zur Verbindung von Namensräumen	136
5.2.4	Der Default-Namensraum des DCE	137

5.2.5	Aliases für CDS-Namen	137
5.2.6	Struktur von Namenseinträgen	139
5.3	DCE Control Program und Browser	142
5.3.1	Überblick über die einzelnen Kommandos	143
5.3.2	Beispiele für die Verwendung der Kommandos	144
5.3.3	Einsatz des CDS-Browsers	147
5.4	XDS-Programmierschnittstelle	148
5.4.1	Übersicht	148
5.4.2	XDS- und XOM-Datentypen	150
5.4.3	XDS- und XOM-Funktionen	151
5.5	Interne Realisierung der Namensverwaltung	154
5.5.1	Interner Ablauf einer Namensinterpretation	154
5.5.2	Verteilung und Replikation von Namenseinträgen	157
5.6	Systemmanagement-Aufgaben	160
5.6.1	CDS-Management	160
5.6.2	Das DCE Control Program als Management-Werkzeug	161
5.6.3	GDS-Management	163
6	Security Service	165
6.1	Wichtige Eigenschaften des Security Service	165
6.2	Authentisierung und Verschlüsselung	167
6.2.1	Ablauf des Authentisierungsprotokolls	167
6.2.2	Cell-übergreifende Authentisierung	170
6.2.3	Erweiterung: Asymmetrische Kryptoverfahren	171
6.3	Autorisierung	173
6.3.1	Zugriffskontrolllisten: ACLs	173
6.3.2	Prüfung der Zugriffsrechte zur Laufzeit	175
6.3.3	ACLs für CDS-Directories	176
6.3.4	Der ACL-Editor	177
6.4	Administrationsaufgaben	179
6.4.1	Übersicht	179
6.4.2	Der Registry Editor	179
6.4.3	Konsistenz mit Unix-Accounts	182
6.5	Weiterführende Programmierschnittstellen	183
7	Der Distributed Time Service	185
7.1	Wichtige Eigenschaften des DTS	185
7.2	Interne Systemstruktur und Zeitformate	186
7.2.1	Architekturkonzept	186
7.2.2	Zeitformate	188
7.3	Programmierschnittstelle	190
7.3.1	Ermittlung und Konvertierung von Zeitangaben	191
7.3.2	Arithmetische Operationen mit Zeitangaben	195
7.3.3	Spezielle Operationen für Zeitintervalle	198
7.4	Management-Aufgaben	200
8	Verteilte Dateiverwaltung	201
8.1	Das Distributed File System	201
8.1.1	Funktionalität und Einordnung des DFS	201
8.1.2	Vergleich mit dem Network File System	205
8.1.3	Einsatz des DFS für verteilte Anwendungen	207

8.1.4	Interne Systemstruktur	208
8.1.5	Management-Aufgaben	211
8.1.6	Weiterführende Programmierschnittstellen	215
8.2	Diskless Support Service	216
9	Objektorientierte DCE-Erweiterungen	219
9.1	Motivation und Einführung	219
9.1.1	Charakteristische Eigenschaften	219
9.1.2	Verteilte objektorientierte Systeme	221
9.2	Unterstützung von C++ in DCE	222
9.2.1	Ziele und Charakteristika	222
9.2.2	Grundkonzepte und Anwendungsbeispiel	223
9.2.3	Interne Realisierungsaspekte	225
9.2.4	Objektorientierte Schnittstellen für weitere DCE-Komponenten	227
9.2.5	Weitere Ansätze	229
10	Online Transaction Processing	231
10.1	Einführung und Grundlagen	231
10.1.1	Grundkonzept	231
10.1.2	Beispiel	232
10.1.3	Zwei-Phasen-Commit-Protokoll	233
10.2	Systembeispiel Encina	234
10.2.1	Systemarchitektur	234
10.2.2	Anwendungsbeispiel	235
10.2.3	Migration und Host-Integration	236
10.2.4	Systemadministration und Realisierungsaspekte	237
10.3	Systemvergleich und Zusammenfassung	239
11	Gesamteinordnung und Bewertung des OSF DCE	241
11.1	CORBA	241
11.1.1	Object Management Group	241
11.1.2	Object Management Architecture	242
11.1.3	Object Request Broker	243
11.1.4	CORBA Services	249
11.1.5	Systembeispiele	251
11.1.6	Vergleich mit DCE	252
11.2	Anwendungsorientierte ISO/OSI-Standards	254
11.2.1	Standardisierung und Entwicklungstendenzen	254
11.2.2	Bezug zu den beiden oberen ISO/OSI-Schichten	254
11.2.3	Open Distributed Processing	258
11.3	SUN ONC	260
11.4	Gesamtbewertung des OSF DCE	260
11.5	Verfügbarkeit von DCE	262
11.6	Ausblick	263
	Literatur	265
	Sachverzeichnis	269
	Glossar	275

Abbildungsverzeichnis

Abb. 1-1	Beispiel einer verteilten Anwendung	2
Abb. 1-2	Grobablauf bei der Projektplanung	3
Abb. 1-3	Grobablauf bei der Fertigung	4
Abb. 1-4	Grobablauf bei der Buchhaltung	4
Abb. 2-1	Gesamtarchitektur des OSF DCE	11
Abb. 2-2	Kooperierende DCE-Cells	13
Abb. 2-3	Struktur von Threads	14
Abb. 2-4	RPC mit Unterbeauftragung	15
Abb. 2-5	Teilkomponenten des DCE RPC	17
Abb. 2-6	Beispiel eines einfachen CDS-Namensraums	18
Abb. 2-7	Beispiel eines einfachen GDS-Namensraums	19
Abb. 2-8	Konfiguration von CDS und GDS mit X.500 und DNS	20
Abb. 2-9	Authentisierung und Autorisierung	22
Abb. 2-10	Struktur der Autorisierung	24
Abb. 2-11	Time-Server-Konfiguration	25
Abb. 2-12	Struktur des Distributed File Systems	27
Abb. 2-13	Diskless-Client/Server-Konfiguration	29
Abb. 2-14	Zusammenwirken der Komponenten des DCE	30
Abb. 2-15	Beispiele für DCE-Systemkonfigurationen	32
Abb. 2-16	Einsatz von DCE in der Beispielanwendung	34
Abb. 2-17	Bedeutung der DCE-Komponenten für Anwender	36
Abb. 2-18	Eingereichte Vorschläge für die DCE-Komponenten	39
Abb. 3-1	Spezielle Eigenschaften des DCE RPC	42
Abb. 3-2	Grundlegender Ablauf eines DCE RPC	44
Abb. 3-3	Struktur des Beispiels	46
Abb. 3-4	Verarbeitungsvorgang mit dem IDL-Compiler	49
Abb. 3-5	Vorgang bei Export des Servers	51
Abb. 3-6	Komplexe Datenstrukturen mit Zeigern in IDL	57
Abb. 3-7	Variables Array innerhalb einer Datenstruktur	58
Abb. 3-8	RPC-Aufrufsemantik	60
Abb. 3-9	Bindevorgang im Detail	63
Abb. 3-10	Binding Handles eines Servers	64
Abb. 3-11	Binding Handles eines Clients	65
Abb. 3-12	Vergleich der verschiedenen Methoden zum Binden	67
Abb. 3-13	Durchführung einer Namensanfrage durch einen Client	70
Abb. 3-14	Erzeugen und Exportieren von Ressourcen-Objekten	71
Abb. 3-15	Auswahl typabhängiger Server-Implementierungen	75
Abb. 3-16	Beispiel für eine Gruppe von Drucker-Servern	77

Abb. 3-17	Beispiel für Gruppeneinträge beim Directory Service	78
Abb. 3-18	Beispiel für Profile-Einträge beim Directory Service	80
Abb. 3-19	Bindevorgang unter Verwendung von Strings	83
Abb. 3-20	Struktur der Schutzmechanismen im Rahmen des RPC	85
Abb. 3-21	Mögliche Schutzgrade beim RPC	87
Abb. 3-22	Ablauf bei der Pipe-Datenübertragung	94
Abb. 3-23	Durchführung von Rückaufrufen	97
Abb. 3-24	Verwendung von Context Handles beim Dateizugriff	100
Abb. 3-25	Nebenläufige RPC-Bearbeitung durch Threads	102
Abb. 3-26	Leistungsdaten des RPC	104
Abb. 3-27	Durchsatzergebnisse	105
Abb. 3-28	Durchsatz bei Verwendung von Pipes	106
Abb. 3-29	Durchsatz des Secure RPC	107
Abb. 3-30	Übertragung über ATM 155 MBit/s	108
Abb. 3-31	Kommandos des DCE Control Programs	109
Abb. 4-1	Spezielle Eigenschaften von DCE-Threads	111
Abb. 4-2	Ausnutzung von Parallelitätseigenschaften durch Threads	113
Abb. 4-3	Operationen zur Thread-Verwaltung	117
Abb. 4-4	Pipeline-Verarbeitung mit Bedingungsvariablen	123
Abb. 4-5	Scheduling-Strategien	124
Abb. 4-6	Operationen zur Verwaltung von Thread-Attributen	125
Abb. 4-7	Prinzip von Jacket-Routinen für E/A-Operationen	128
Abb. 4-8	Liste des wichtigsten Jacket-Routinen	128
Abb. 4-9	Beispiel für eine Verklemmung zwischen Threads	130
Abb. 5-1	Interaktion der Komponenten des Directory Service	133
Abb. 5-2	Integration von Namensräumen mittels Junctions	136
Abb. 5-3	Default-Namensraum einer Cell (oberste Ebene)	137
Abb. 5-4	Beispiele für Soft Links auf CDS-Ebene	138
Abb. 5-5	Struktur eines Namenseintrags	140
Abb. 5-6	Vordefinierte Attribute bei CDS (begrenzter Auszug)	141
Abb. 5-7	Vordefinierte Attribute bei GDS (begrenzter Auszug)	142
Abb. 5-8	Benutzerkommandos des DCE Control Programs	143
Abb. 5-9	Symbole des CDS-Browsers	147
Abb. 5-10	Bildschirmfenster des CDS-Browsers: Beispiel	147
Abb. 5-11	Konzeptionelle Struktur der XDS-Schnittstelle	149
Abb. 5-12	Namensrepräsentation durch XDS/XOM-Objekte	150
Abb. 5-13	Signatur der XDS-Funktionen in Pseudo-Notation	151
Abb. 5-14	Schematischer Ablauf der Namensinterpretation	154
Abb. 5-15	GDS-Client/Server-Konfiguration	155
Abb. 5-16	Globale Namensinterpretation als Weg-Zeit-Diagramm	156
Abb. 5-17	Server-Struktur für GDS und CDS	157
Abb. 5-18	Replikation von Namenseinträgen durch CDS-Server	158
Abb. 5-19	Interne Verkettung von Directories über Child Pointer	159
Abb. 5-20	Wichtige Management-Kommandos von dcecp	162
Abb. 5-21	Masken für das GDS-Shadow-Management	164
Abb. 6-1	Wichtige Eigenschaften des Security Service	166
Abb. 6-2	Authentisierungsprotokoll in der Login-Phase	168
Abb. 6-3	Authentisierungsprotokoll beim Server-Aufruf	169
Abb. 6-4	Surrogate für die Security Server fremder Cells	171

Abb. 6-5	Verfahren auf der Basis öffentlicher Schlüssel: Struktur	172
Abb. 6-6	Beispiel einer Zugriffskontrollliste für ein CDS-Directory	173
Abb. 6-7	Mögliche Arten von ACL-Einträgen	174
Abb. 6-8	Erweitertes Beispiel einer Zugriffskontrollliste	176
Abb. 6-9	Weitergabe von Attributen einer Creation ACL	177
Abb. 6-10	Wichtige Kommandos des ACL-Editors	177
Abb. 6-11	Import und Export von Paßwort-Information	182
Abb. 7-1	Wichtige Eigenschaften des Distributed Time Service	185
Abb. 7-2	Detaillierte Time-Server-Konfiguration	187
Abb. 7-3	Berechnung der Zeit bei der Synchronisation	188
Abb. 7-4	Konvertierungsfunktionen zwischen Zeitformaten	191
Abb. 7-5	Ablauf des Beispiels für Zeitberechnungen	197
Abb. 7-6	Wichtige Management-Kommandos von dcecp	200
Abb. 8-1	Wichtige Eigenschaften des DFS aus Anwendersicht	202
Abb. 8-2	Globaler Dateiraum des DFS	202
Abb. 8-3	Wichtige systeminterne Eigenschaften des DFS	203
Abb. 8-4	Vergleich von DFS und NFS	206
Abb. 8-5	Verteilte Anwendung als Data-Sharing-Modell	208
Abb. 8-6	Teilkomponenten des Distributed File Systems	209
Abb. 8-7	Befehle zur Verwaltung von Filesets	212
Abb. 8-8	Abläufe zwischen diskless Client und Servern	217
Abb. 9-1	Struktur einer verteilten objektorientierten Anwendung	221
Abb. 9-2	RPC-Objektmodell: Vererbungsdiagramm	223
Abb. 9-3	Aufbau einer Objektreferenz	226
Abb. 10-1	Beispiel für OLTP	232
Abb. 10-2	Zwei-Phasen-Commit-Protokoll	233
Abb. 10-3	Systembeispiel Encina	234
Abb. 10-4	Encina-Beipielszenario	235
Abb. 10-5	Beispiel: Integration mit CICS	237
Abb. 10-6	Encina-Systemadministration	238
Abb. 10-7	Sicherheitsaspekte	238
Abb. 10-8	Message Queueing	239
Abb. 11-1	Object Management Architecture	243
Abb. 11-2	CORBA-Architekturmodell	244
Abb. 11-3	Entwicklung von CORBA-Anwendungen	245
Abb. 11-4	Interne Struktur des Interface Repository	247
Abb. 11-5	Komposition von Aufrufstrukturen zur Laufzeit	248
Abb. 11-6	Funktionsweise und Aufbau des BOA	249
Abb. 11-7	CORBA Services	250
Abb. 11-8	Vergleich von CORBA und DCE	252
Abb. 11-9	Dienste der beiden oberen ISO/OSI-Schichten	255
Abb. 11-10	Grobstruktur eines Trading-Ansatzes	259

1 Einführung

1.1 Einleitung

Verteilte Rechneranwendungen gewinnen zunehmend an Bedeutung in den verschiedensten Bereichen. Sie lassen sich beschreiben als Programme aus verschiedenen Modulen, die auf unterschiedlichen Rechnern plziert sind und während ihrer Verarbeitung über ein Kommunikationsnetz interagieren [MÜS92]. Die einzelnen Module arbeiten weitgehend autonom und unterliegen insbesondere keiner zentralen Kontrolle.

Konkrete Beispiele sind verteilte Anwendungen zur Automatisierung von Bürovorgängen, zur Steuerung von Fertigungsabläufen oder zur Verwaltung von Management-Information. Diese Bereiche sind gekennzeichnet durch eine inhärente physische Verteilung ihrer Datenhaltungs- und Verarbeitungskomponenten; z.B. sind Fertigungsvorgänge auf verschiedene Maschinen verteilt, die oft dezentral kontrolliert werden.

Die Kommunikationsmechanismen sowie eine Vielzahl weiterer Konzepte der verteilten Programmierung werden durch ein *verteiltes System* unterstützt [SLK87, COD93, MUL89]; dieses setzt sich aus der Menge der beteiligten Rechnerknoten, dem Kommunikationsnetz sowie der systemnahen Software zur Realisierung der verteilten Verarbeitung zusammen. Generell läßt sich sagen, daß die Entwicklung verteilter Anwendungen umso einfacher wird, je mächtiger die Mechanismen des verteilten Systems sind.

Das *Distributed Computing Environment (DCE)* der *Open Software Foundation (OSF)* stellt ein solches verteiltes System dar, das relativ umfangreiche unterstützende Softwaremechanismen bietet. Ziel dieses Buches ist es, diese Softwareumgebung detailliert vorzustellen und dem Leser die Bewertung und den praktischen Einsatz ihrer Komponenten zu ermöglichen. Angesprochen werden dabei insbesondere Anwendungsentwickler und Endbenutzer des OSF DCE, sowie technische Manager, die mit der DCE-Umgebung in Berührung kommen. Für alle im Buch beschriebenen Systembereiche und Komponenten des DCE wird auch eine allgemeine Übersicht und Einführung mit Darstellung der Grundkonzepte gegeben, was für das Verständnis ohne spezielle Hintergrundliteratur von Bedeutung ist.

Im Februar 1996 fusionierten die OSF und die X/OPEN Company Ltd. zur *Open Group*, um gemeinsam die Entwicklung von verteilten Systemen fortzuführen.

1.2 Beispiel einer verteilten Anwendung

Abb. 1-1 zeigt ein Beispiel einer verteilten Anwendung aus dem Gesamtbereich der Büro- und Fertigungsautomatisierung.

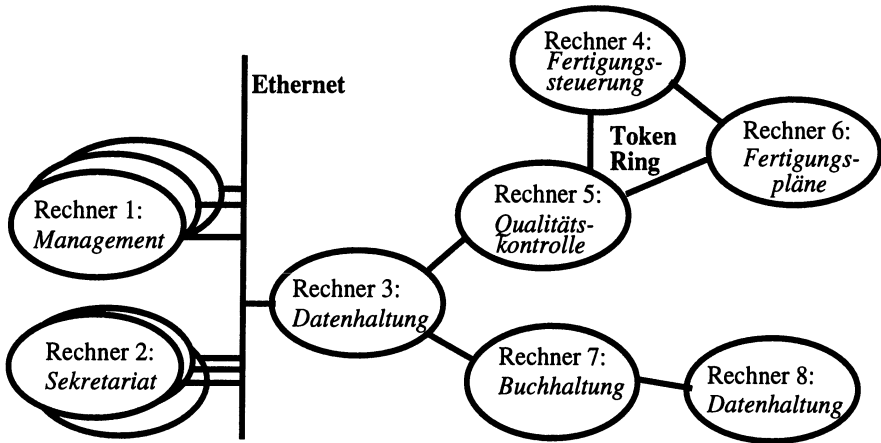


Abb. 1-1 Beispiel einer verteilten Anwendung

Struktur des verteilten Systems

Die Anwendung ist auf mehrere Rechner verteilt, die über ein Ethernet, einen Token Ring sowie über mehrere Punkt-zu-Punkt-Leitungen gekoppelt sind. Einige Rechner agieren dabei u.a. als Gateway, etwa Rechner 3, 5 und 7. Auf der Ebene des OSF DCE ist jedoch die Netzstruktur transparent; es wird grundsätzlich davon ausgegangen, daß die Rechner logisch voll vermascht sind, auch wenn das physisch nicht der Fall ist. Dadurch kann potentiell jeder Rechner Nachrichten an jeden anderen senden. Diese Funktionalität wird durch transportorientierte Protokolle wie z.B. TCP/IP erbracht, die vom DCE als existierend vorausgesetzt werden.

Abläufe innerhalb der Anwendung

Die einzelnen Komponenten der Anwendung sind nun - wie gezeigt - verteilt auf verschiedenen Rechnern plziert. Typische verteilte Verarbeitungsabläufe haben etwa folgende Form:

Projektplanung (Abb. 1-2)

Ein Manager (auf Rechner 1) beschafft sich Daten zu früheren Produkt-Entwicklungsprojekten von der entfernten Komponente zur Datenhaltung (auf Rechner 3), etwa durch Aufruf einer entfernten Datenbankoperation. Danach werden lokale Planungsaufgaben unter Einsatz eines Spreadsheet-Programms durch den Manager durchgeführt. Gegebenenfalls sendet der Manager im Verlauf dessen Informations- oder Anfragenachrichten an andere Manager auf anderen Rechnern; auf diese Weise können z.B. Koordinationsfragen entfernt geklärt werden. Nach Abschluß einer

groben Produktplanungsphase sendet der Manager die resultierenden Daten zur weiteren Überarbeitung an seine Sekretärin (Rechner 2), indem er eine entfernte Speicherprozedur dort aufruft. Danach folgen zahlreiche Interaktionen mit anderen Abteilungen zur Feinplanung, z.B. mit der technischen Entwicklungsabteilung, die hier nicht gezeigt sind.

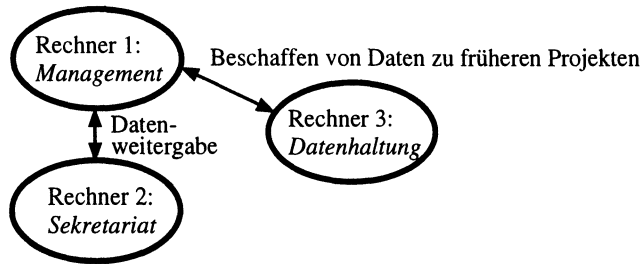


Abb. 1-2 Grobablauf bei der Projektplanung

Fertigung (Abb. 1-3)

Im Rahmen der Fertigung von Produkten wird die Fertigungssteuerung (Rechner 4) mit der Koordination beauftragt. Je nach Fertigungsauftrag beschafft sie einen zugehörigen Fertigungsplan von Rechner 6, der eine Datenbasis von Fertigungsplänen verwaltet. Anschließend werden ggf. mehrere Maschinensteuerungen mit den einzelnen Fertigungsschritten beauftragt, die wiederum auf verschiedene Rechner verteilt sein können (in der Abbildung nicht gezeigt). Schließlich nimmt die Fertigungssteuerung die Fertigmeldungen der Maschinen entgegen und beauftragt die Qualitätskontrolle (Rechner 5) mit der Abschlußprüfung des gefertigten Produkts.

Buchhaltung (Abb. 1-4)

Die Buchhaltung (Rechner 7) ist zunächst recht konventionell als eine zentrale Instanz realisiert. Die von ihr verwendeten Daten (Rechnungen, Lieferscheine etc.) werden aber z.B. wegen ihres großen Umfangs durch eine separate, entfernte Datenbank auf Rechner 8 verwaltet. Es ist daher erforderlich, dort entfernte Operationen zum Anfordern und Abspeichern von Daten aufzurufen. Außerdem greift die Buchhaltung teilweise auch auf die Dienste der Datenhaltung auf Rechner 3 zu, um z.B. bei Bedarf eine Zuordnung zwischen Rechnungstiteln und Produktbeschreibungen durchführen zu können.

Globale Interaktionen

Grundsätzlich sind neben diesen dedizierten Abläufen auch globale Interaktionen möglich, in die ggf. sehr viele Rechner involviert sind. Als Beispiel sei die Analyse eines Produkts durch einen Manager in bezug auf seine Ausschußrate bei der Fertigung und die erzielte Gewinnspanne genannt.

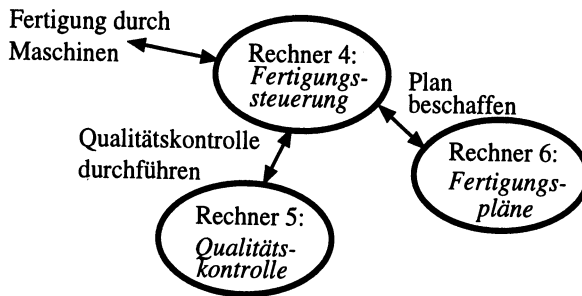


Abb. 1-3 Grobablauf bei der Fertigung

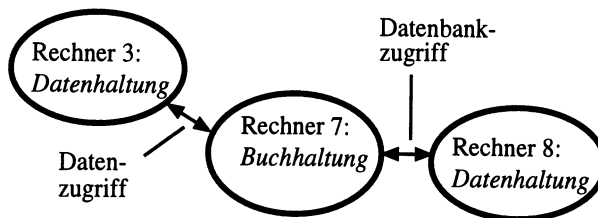


Abb. 1-4 Grobablauf bei der Buchhaltung

Wichtige Eigenschaften

Am Beispiel wurde aufgezeigt, daß eine bestimmte Plazierung von Anwendungskomponenten auf Rechnerknoten vorgenommen wird. Es ist auch offensichtlich, daß geeignete Kommunikationsmechanismen zwischen den Komponenten bereitstellen müssen.

Außerdem ist schon hier die weitergehende Entkopplung der verschiedenen Abläufe zu erkennen, die dadurch auch parallel ausgeführt werden können. Ebenfalls ist zu sehen, daß manche Dienste bzw. Betriebsmittel (z.B. die Projektdatenbank) von mehreren Komponenten verwendet werden und dadurch eine verbesserte Integration erzielt wird. Andererseits kann die Verwaltung komponentenspezifischer Daten (z.B. des Managements) stärker dezentralisiert und damit leichter durch die jeweilige Komponente kontrolliert werden. Diese und einige weitere Eigenschaften werden im folgenden vertieft.

1.3 Vorteile verteilter Anwendungen

Die Entwicklung verteilter Anwendungen ist - auch mit einer Systemumgebung wie dem OSF DCE - generell durch eine erhöhte Komplexität gekennzeichnet; daher stellt sich zunächst die Frage nach den Vorteilen solcher Ansätze, um den entsprechenden Entwicklungsaufwand technisch und wirtschaftlich zu rechtfertigen:

- *Dezentralisierung und Lokalität:* Durch die Verteilung der Module wird eine weitgehend dezentrale Verarbeitung möglich. Dadurch kann eine hohe Lokalität in bezug auf die Zuordnung von Daten und bearbeitenden Operationen erzielt werden. Dies wiederum verbessert die Laufzeiteigenschaften und erleichtert die organisatorische Verwaltung.
- *Parallele Verarbeitung:* Da jeder beteiligte Rechner eigene Betriebsmittel, insbesondere Prozessor und Speicher, bereitstellt, können die einzelnen Module stark parallel arbeiten; dadurch wird die Gesamtbearbeitungszeit eines Auftrags verkürzt.
- *Fehlertoleranz:* Die beteiligten Rechner sind weitgehend autonom; dies gilt auch für Systemausfälle; bei Ausfall eines Rechners bleibt der übrige Teil der Anwendung auf anderen Rechnern funktionsfähig. Explizite Fehlertoleranz kann zusätzlich durch Replikation von Daten und Operationen erzielt werden.
- *Gemeinsame Betriebsmittelnutzung:* Da verschiedene Rechner über ein Kommunikationsnetz gekoppelt sind, können sie auch gemeinsam auf die Ressourcen eines bestimmten Rechners zugreifen. Dadurch wird die gemeinsame Nutzung teurer Peripheriegeräte (z.B. Drucker, Plotter oder Speichermedien) sowie komplexer Softwarekomponenten (z.B. Datenbanken) ermöglicht.
- *Integration von Teilanwendungen:* Die Gesamtfunktionalität existierender Teilanwendungen kann oft durch ihre Integration zu einer verteilten Anwendung gesteigert werden; beispielsweise kann eine Management-Datenbank mit einem Spreadsheet-System gekoppelt werden, um automatisierte Datenauswertungen durchzuführen.

Gerade die beiden letztgenannten Punkte stellen in sehr vielen Fällen die wichtigste Motivation für den Einsatz verteilter Systemtechnologie in der industriellen Praxis dar. Sie lassen sich vergleichsweise einfach erreichen und können sehr rasch zu einer realen Kostenreduzierung führen, die meist auch relativ genau gegenüber dem zu erwartenden Entwicklungsaufwand abgeschätzt werden kann. Die drei anderen Punkte erfordern dagegen meist eine umfangreichere Planung bzw. Umstrukturierung von Anwendungen. Beispielsweise müssen unabhängige Verarbeitungseinheiten identifiziert oder konstruiert werden, um Parallelität echt auszunutzen. Ebenso ist der Einsatz und gegebenenfalls sogar die Entwicklung dezentraler Replikations- und Konsistenzerhaltungstechniken erforderlich, um einen hohen Grad der Fehlertoleranz zu erzielen. Die Ausnutzung von Lokaltätseigenschaften erfordert ebenso grundsätzliche Entwurfsentscheidungen im Rahmen der Anwendungsentwicklung oder -neustrukturierung. Dennoch sind diese Bereiche gerade für große Rechneranwendungen als Mittel zur Steigerung der Verarbeitungskapazität und -qualität von zentraler Bedeutung. Sie weisen meist ein sehr