

Inhaltsverzeichnis

1	Einleitung	1
1.1	Dezentrale Computer-Systeme und verteilte Anwendungen	1
1.2	Zuverlässige Anwendungsverarbeitung	2
1.3	Einordnung der Arbeit	3
1.4	Übersicht über die Arbeit	5
2	Unterstützung verteilter Anwendungen auf gemeinsamen Ressourcen	7
2.1	Grundlagen	7
2.2	Problemstellung: Zuverlässige Abwicklung verteilter Anwendungen	12
2.3	Zielsetzung	17
2.4	Betriebssystem-Ansatz	19
2.5	Kooperationsmodelle und Kommunikationsmechanismen	24
2.6	Verteilte Programmiersprachen	27
2.7	Das OSF Distributed Computing Environment	31
2.8	Verteilte Datenbanken	35
2.9	Transaktionssysteme	37
2.10	Zusammenfassung	42
3	Transaktionsorientierte Anwendungsverarbeitung	49
3.1	Klassische Transaktionsverarbeitung in zentralisierten Datenbanksystemen	49
3.2	Stärken und Schwächen klassischer Transaktionen	53
3.3	Erweiterungen des klassischen Transaktionskonzeptes	59
3.4	Verallgemeinerter Kontrollmechanismus	69
4	Modellierung und Abwicklung langlebiger Anwendungen mit ConTracts	73
4.1	Überblick über das ConTract-Programmiermodell	74
4.2	Modellierung einer verteilten Anwendung als ConTract	75
4.3	Ablaufsteuerung einer ConTract-Ausführung	88
4.4	Zusammenfassung und Diskussion	89

Realisierung der ConTract-Mechanismen	93
5.1 Skript- und Step-Ausführung	94
5.2 Robuste Kontrollflußabwicklung	97
5.3 Kompensation	98
5.4 Kontextverwaltung und -adressierung	100
5.5 Synchronisation mit Isolationsprädikaten	104
5.6 Konfliktbehandlung	107
5.7 Zusammenfassung	110
Architektur- und Betriebsmodell eines ConTract-Systems	113
6.1 Allgemeine Entwurfsziele	113
6.2 Verteilte Transaktionsverarbeitung	114
6.3 Diskussion des X/Open DTP-Modells	124
6.4 Komponenten einer ConTract-Architektur	127
6.5 Verarbeitungsmodell der ConTract-Abwicklung	131
6.6 Prozeßarchitektur und Kommunikationsmodell	149
6.7 Sicherheitsaspekte	153
6.8 Zusammenfassung	155
Zuverlässige ConTract-Abwicklung	157
7.1 Auswirkungen der verschiedenen Fehlerarten auf die ConTract-Verarbeitung	157
7.2 Wiederanlauf eines ConTract-Systems	163
7.3 Vorwärts-Recovery unterbrochener ConTracts	167
7.4 Architektur zur Tolerierung von Knotenausfällen	171
7.5 Überwachung und Vertretung unsicherer Knoten	178
7.6 Überwachung und Vertretung zuverlässiger Knoten	180
7.7 Zusammenfassung	210
APRICOTS – Eine prototypische Implementierung	211
8.1 Organisatorisches und technisches Umfeld	211
8.2 Prototypische Realisierung des ConTract-Ansatzes	212
8.3 Fazit	215
Zusammenfassung und Ausblick	217
Syntax-Definition der Skript-Sprache	221
Beispiel-Skript „Reisebuchung“	225
Literatur	229

Abbildungsverzeichnis

2.1	<i>Modell eines verteilten Systems.</i>	8
2.2	<i>Beispiel für die Struktur einer verteilten Anwendung.</i>	9
2.3	<i>Netzwerkbetriebssystem.</i>	20
2.4	<i>Verteiltes Betriebssystem.</i>	21
2.5	<i>Programm-zu-Programm-Kommunikation nach LU6.2.</i>	24
2.6	<i>Kommunikation nach dem Client-Server-Modell.</i>	25
2.7	<i>Architektur verteilter Programmiersprachen.</i>	28
2.8	<i>Architektur eines verteilten Datenbanksystems.</i>	35
2.9	<i>Verteilte Transaktionsverarbeitung mit TP-Monitoren.</i>	38
2.10	<i>Ausfallsichere Weiterleitung von Aufträgen und Ergebnissen über stabile Warteschlangen.</i>	40
2.11	<i>Verarbeitungsschema bei der Benutzung stabiler Warteschlangen.</i>	40
2.12	<i>Fehlertoleranz durch Zustandssicherung und Wiederanlaufverfahren.</i>	45
2.13	<i>Übernahme der Berechnung durch eine passive Reservekomponente.</i>	46
2.14	<i>Übernahme der Berechnung durch eine aktive Reservekomponente.</i>	46
2.15	<i>Redundante Ausführung auf mehreren Ausführungsinstanzen.</i>	47
3.1	<i>Transaktionen bilden Zustandsänderungen der realen Welt auf die DB ab.</i>	50
3.2	<i>Transaktionsprogramm zur Überweisung eines Geldbetrages.</i>	51
3.3	<i>Ablauf einer Reisebuchung (schematisch).</i>	53
3.4	<i>Isolationsproblematik bei langdauernden Transaktionen.</i>	57
3.5	<i>Geschachtelte Überweisungs-Transaktion.</i>	61
3.6	<i>Pseudocode des Programms „Zinsgutschrift“ als Mini-Batch.</i>	65
4.1	<i>Das ConTract-Programmiermodell.</i>	74
4.2	<i>ConTract „Reisebuchung“.</i>	75
4.3	<i>Code-Fragment des Programmschrittes „Flugbuchung“.</i>	76
4.4	<i>Beispiel-Skript für den ConTract „Reisebuchung“ (Ausschnitt).</i>	78
4.5	<i>Kontextvariablen im Skript „Reisebuchung“.</i>	79
4.6	<i>Synchronisation mit Isolationsprädikaten und Konfliktbehandlung.</i>	86
4.7	<i>Kompensationen für die Programmschritte der Reisebuchung.</i>	87
4.8	<i>ConTract-Zustandsdiagramm.</i>	89

5.1	Realisierung einer Schleife.	95
5.2	Implementierung eines Skriptes als Prädikat-Transitionsnetz (Ausschnitt). .	96
5.3	Systemroutinen zum Lesen von Kontextvariablen aus der Kontext-DB. . .	103
5.4	Realisierung der Zugriffe auf Kontextwerte beim Step-Aufruf.	104
6.1	Verteilte Buchungstransaktion.	115
6.2	X/Open-Referenzarchitektur für die verteilte Transaktionsverarbeitung. .	116
6.3	Ablauf einer verteilten Überweisungstransaktion im XA-Modell.	118
6.4	Ablauf des Zweiphasen-Commit-Protokolls.	120
6.5	a) - c): Prozeß- und Kommunikationsmodell des DTP-Standards.	123
6.6	Zentrale Komponenten der ConTract-Verarbeitung nach dem DTP-Modell.	128
6.7	Schichtenarchitektur eines ConTract-Systems.	130
6.8	Ablauf der Skript-Abwicklung durch den ConTract-Manager.	132
6.9	Attribute eines Step-Bearbeitungsauftrags.	133
6.10	Kommunikation zwischen ConTract-Manager und Step-Server über stabile Warteschlangen.	135
6.11	Verarbeitung der Step-Rückmeldungen durch den ConTract-Manager. . . .	135
6.12	Abarbeitung der Auftragswarteschlange durch einen Step-Server.	137
6.13	Verarbeitung der Step-Rückmeldungen durch einen ConTract-Manager. . .	140
6.14	Protokoll zur Migration eines ConTract von Knoten A nach Knoten B. . .	145
6.15	Logische Beziehungen der Komponenten eines ConTract-Systems.	148
6.16	Abbildung der ConTract-Komponenten auf Betriebssystemprozesse.	151
7.1	Transaktions- und ConTract-Recovery beim Wiederanlauf eines ConTract-Systems mit Hilfe der Protokolldatei.	164
7.2	Beispiel-Skript mit Kontrollflußereignissen.	169
7.3	ConTract-Verarbeitung auf zuverlässigen und unsicheren Knoten.	176
7.4	Zweiphasiges Protokoll zur Reorganisation eines Überwachungsringes. . . .	183
7.5	Protokoll zur Ringüberwachung und Rekonfiguration: Deklarationen. . . .	185
7.6	Überwachungsmodus des Ringprotokolls.	186
7.7	Rekonfigurationsprotokoll Phase 1: Zustandsabfrage.	187
7.8	Rekonfigurationsprotokoll Phase 2: Ausfallentscheidung treffen.	188
7.9	Beispiel eines Überwachungsringes.	189
7.10	Überwachungsring nach Ausfall von Knoten B.	191
7.11	Rekonfiguration des Rings nach einer Kommunikationsstörung.	192
7.12	Beispiel eines Überwachungsringes nach einer Partitionierung.	194
7.13	Behandlung von Netzwerkpartitionen in der Prozedur Rekonfigurieren. . . .	195
7.14	Verschobene Ausfallentscheidung nachholen bei neuem Lebenszeichen. . .	196
7.15	Vergleich mit anderen Ansätzen.	208