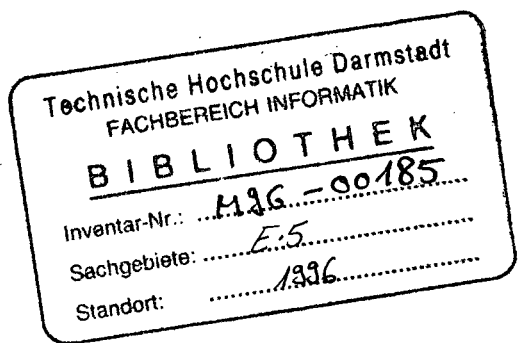


James Jay Kistler

Disconnected Operation in a Distributed File System



Springer

Table of Contents

1. Introduction	1
1.1 Distributed Computing	1
1.2 Distributed File Systems	2
1.3 Disconnected Clients	4
1.3.1 Impact of Large Scale	5
1.3.2 Impact of Mobile Computers	6
1.4 Disconnected Operation	7
1.5 The Thesis	8
1.5.1 Requirements for Masking Disconnection	9
1.5.2 Establishing the Thesis	10
1.6 Organization of this Document	10
2. Design Rationale	11
2.1 AFS Heritage	11
2.1.1 Vice/Virtue	11
2.1.2 Client Caching	14
2.1.3 Volumes	17
2.2 Coda Charter	17
2.2.1 High Availability in an AFS-Like Environment	18
2.2.2 First- versus Second-Class Replication	19
2.3 Partitioned Replica Control	20
2.3.1 Transaction Model of Computation	22
2.3.2 System-Inferred Transactions	28
2.3.3 High Availability Transaction Specification	31
2.3.4 Optimistic Enforcement of Correctness	41
2.3.5 Replica Control Summary	48
3. Architecture	51
3.1 The View from Venus	51
3.2 Decoupling from Server Replication	53
3.3 Masking Disconnection	53
3.3.1 Cache Miss Avoidance	53
3.3.2 Replica Control	54
3.3.3 Update Buffering	54

XIV Table of Contents

3.3.4	Persistence	55
3.3.5	Security	55
3.4	Computation Correctness	56
3.4.1	Intra-Partition Transaction Processing	56
3.4.2	Merging	58
3.5	Detailed Design and Implementation	60
4.	Coda Internals	61
4.1	Client Organization	61
4.1.1	Cache Manager	62
4.1.2	MiniCache	66
4.2	Server Organization	70
4.2.1	File Server	70
4.2.2	Authentication Server	74
4.2.3	Update Client/Server	75
5.	Hoarding	77
5.1	Idealized Cache Management	77
5.2	Practical Cache Management	78
5.2.1	Use of Explicit Reference Information	79
5.2.2	Combining Implicit and Explicit Information	80
5.2.3	Use of Naming Information	81
5.2.4	Whole-Object Caching	82
5.3	Detailed Design and Implementation	83
5.3.1	Hoard Database	83
5.3.2	Object Priorities, Resource Allocation, and Naming Effects	93
5.3.3	Cache Equilibrium	96
6.	Server Emulation	117
6.1	Disconnected Processing	117
6.1.1	User Transaction Processing	117
6.1.2	Hoard Walking	118
6.2	Transaction Logging	119
6.2.1	Log Organization and Record Format	120
6.2.2	Cancellation Optimizations	121
6.2.3	Store-Record Optimization	132
6.3	Data Persistence	133
6.3.1	Leveraging Off of the Local Unix File System	134
6.3.2	Using RVM for Meta-Data	136
6.4	Other Masking Responsibilities	146
6.4.1	Access Checking	146
6.4.2	Fid Generation	149
6.5	Mitigating Emulation Failures	149
6.5.1	Cache Misses	150

6.5.2	Resource Exhaustion	151
7.	Reintegration	153
7.1	Overview of the Algorithm	153
7.1.1	Prelude	153
7.1.2	Interlude	156
7.1.3	Postlude	158
7.2	Non-Certification Issues	159
7.2.1	Volume-Log Ownership	160
7.2.2	Transaction Validation	161
7.2.3	Back-Fetching	161
7.2.4	Atomicity of Reintegration	162
7.2.5	Closures	164
7.3	Certification	165
7.3.1	Version Certification	166
7.3.2	Value Certification	167
7.3.3	Hybrid Certification	168
7.3.4	Other Certification Issues	173
7.3.5	Summary	177
8.	Evaluation	183
8.1	Implementation Status and Testbed Environment	183
8.2	Qualitative Evaluation	186
8.2.1	Hoarding	186
8.2.2	Server Emulation	193
8.2.3	Reintegration	195
8.2.4	General Observations	198
8.3	Quantitative Evaluation	200
8.3.1	Client Storage Requirements	200
8.3.2	Task Latency	208
8.3.3	Reintegration Latency	211
8.3.4	Cross-Client Write-Sharing	215
9.	Related Work	221
9.1	Systems	221
9.1.1	FACE	221
9.1.2	AFS and Cedar	223
9.1.3	Tait and Duchamp	223
9.1.4	Ficus	225
9.2	Mechanisms	227
9.2.1	Replica Control	227
9.2.2	Pre-Fetching and Hoarding	228
9.2.3	Log Optimizations	230
9.2.4	Recovery in Physical File Systems	230

10. Conclusions	233
10.1 Contributions	233
10.2 Future Work	235
10.3 Final Remarks	237
Bibliography	239
Index	245