Contents

PART I

List of Sidebars xv
Preface xix
Where to find Part II and other On-line Materials xxix
Acknowledgments xxxi

CHAPTER 1 Systems 1
Overview 2
1.1 Systems and Complexity 3
1.1.1 Common Problems of Systems in Many Fields 3
1.1.2 Systems, Components, Interfaces, and Environments 8
1.1.3 Complexity 10
1.2 Sources of Complexity 13
1.2.1 Cascading and Interacting Requirements 13
1.2.2 Maintaining High Utilization 17
1.3 Coping with Complexity I 19
1.3.1 Modularity 19
1.3.2 Abstraction 20
1.3.3 Layering 24
1.3.4 Hierarchy 25
1.3.5 Putting it Back Together: Names make Connections 26
1.4 Computer Systems are the Same but Different 27
1.4.1 Computer Systems have no Nearby Bounds on Composition 28
1.4.2 d(technology)/dt is Unprecedented 31
1.5 Coping with Complexity II 35
1.5.1 Why Modularity, Abstraction, Layering, and Hierarchy aren't Enough 36
1.5.2 Iteration 36
1.5.3 Keep it Simple 39
What the Rest of this Book is About 40
Exercises 41

CHAPTER 2 Elements of Computer System Organization 43
Overview 44
2.1 The Three Fundamental Abstractions 45
2.1.1 Memory 45

2.1.2 Interpreters 53
2.1.3 Communication Links 59
2.2 Naming in Computer Systems 60
2.2.1 The Naming Model 61
2.2.2 Default and Explicit Context References 66
2.2.3 Path Names, Naming Networks, and Recursive Name Resolution 71
2.2.4 Multiple Lookup: Searching through Layered Contexts 73
2.2.5 Comparing Names 75
2.2.6 Name Discovery 76
2.3 Organizing Computer Systems with Names and Layers 78
2.3.1 A Hardware Layer: The Bus 80
2.3.2 A Software Layer: The File Abstraction 87
2.4 *Looking Back and Ahead* 90
2.5 Case Study: UNIX® File System Layering and Naming 91
2.5.1 Application Programming Interface for the UNIX File System 91
2.5.2 The Block Layer 93
2.5.3 The File Layer 95
2.5.4 The Inode Number Layer 96
2.5.5 The File Name Layer 96
2.5.6 The Path Name Layer 98
2.5.7 Links 99
2.5.8 Renaming 101
2.5.9 The Absolute Path Name Layer 102
2.5.10 The Symbolic Link Layer 104
2.5.11 Implementing the File System API 106
2.5.12 The Shell and Implied Contexts, Search Paths, and Name Discovery 110
2.5.13 Suggestions for Further Reading 112
Exercises 112

CHAPTER 3 The Design of Naming Schemes 115
Overview 115
3.1 Considerations in the Design of Naming Schemes 116
3.1.1 Modular Sharing 116
3.1.2 Metadata and Name Overloading 120
3.1.3 Addresses: Names that Locate Objects 122
3.1.4 Generating Unique Names 124
3.1.5 Intended Audience and User-Friendly Names 127
3.1.6 Relative Lifetimes of Names, Values, and Bindings 129
3.1.7 *Looking Back and Ahead*: Names are a Basic System Component 131

3.2 Case Study: The Uniform Resource Locator (URL) 132
3.2.1 Surfing as a Referential Experience; Name Discovery 132
3.2.2 Interpretation of the URL 133
3.2.3 URL Case Sensitivity 134
3.2.4 Wrong Context References for a Partial URL 135
3.2.5 Overloading of Names in URLs 137
3.3 War Stories: Pathologies in the Use of Names 138
3.3.1 A Name Collision Eliminates Smiling Faces 139
3.3.2 Fragile Names from Overloading, and a Market Solution 139
3.3.3 More Fragile Names from Overloading, with Market Disruption 140
3.3.4 Case-Sensitivity in User-Friendly Names 141
3.3.5 Running Out of Telephone Numbers 142
Exercises 144

CHAPTER 4 Enforcing Modularity with Clients and Services 147
Overview 148
4.1 Client/Service Organization 149
4.1.1 From Soft Modularity to Enforced Modularity 149
4.1.2 Client/Service Organization 155
4.1.3 Multiple Clients and Services 163
4.1.4 Trusted Intermediaries 163
4.1.5 A Simple Example Service 165
4.2 Communication Between Client and Service 167
4.2.1 Remote Procedure Call (RPC) 167
4.2.2 RPCs are not Identical to Procedure Calls 169
4.2.3 Communicating through an Intermediary 172
4.3 Summary and The Road Ahead 173
4.4 Case Study: The Internet Domain Name System (DNS) 175
4.4.1 Name Resolution in DNS 176
4.4.2 Hierarchical Name Management 180
4.4.3 Other Features of DNS 181
4.4.4 Name Discovery in DNS 183
4.4.5 Trustworthiness of DNS Responses 184
4.5 Case Study: The Network File System (NFS) 184
4.5.1 Naming Remote Files and Directories 185
4.5.2 The NFS Remote Procedure Calls 187
4.5.3 Extending the UNIX File System to Support NFS 190
4.5.4 Coherence 192
4.5.5 NFS Version 3 and Beyond 194
Exercises 195

CHAPTER 5 Enforcing Modularity with Virtualization 199
Overview 200
5.1 Client/Server Organization within a Computer Using Virtualization 201
5.1.1 Abstractions for Virtualizing Computers 203
5.1.2 Emulation and Virtual Machines 208
5.1.3 Roadmap: Step-by-Step Virtualization 208
5.2 Virtual Links Using SEND, RECEIVE, and a Bounded Buffer 210
5.2.1 An Interface for SEND and RECEIVE with Bounded Buffers 210
5.2.2 Sequence Coordination with a Bounded Buffer 211
5.2.3 Race Conditions 214
5.2.4 Locks and Before-or-After Actions 218
5.2.5 Deadlock 221
5.2.6 Implementing ACQUIRE and RELEASE 222
5.2.7 Implementing a Before-or-After Action Using the One-Writer Principle 225
5.2.8 Coordination between Synchronous Islands with Asynchronous Connections 228
5.3 Enforcing Modularity in Memory 230
5.3.1 Enforcing Modularity with Domains 230
5.3.2 Controlled Sharing Using Several Domains 231
5.3.3 More Enforced Modularity with Kernel and User Mode 234
5.3.4 Gates and Changing Modes 235
5.3.5 Enforcing Modularity for Bounded Buffers 237
5.3.6 The Kernel 238
5.4 Virtualizing Memory 242
5.4.1 Virtualizing Addresses 243
5.4.2 Translating Addresses Using a Page Map 245
5.4.3 Virtual Address Spaces 248
5.4.4 Hardware versus Software and the Translation Look-Aside Buffer 252
5.4.5 Segments (Advanced Topic) 253
5.5 Virtualizing Processors Using Threads 255
5.5.1 Sharing a Processor Among Multiple Threads 255
5.5.2 Implementing YIELD 260
5.5.3 Creating and Terminating Threads 264
5.5.4 Enforcing Modularity with Threads: Preemptive Scheduling 269
5.5.5 Enforcing Modularity with Threads and Address Spaces 271
5.5.6 Layering Threads 271

5.6 Thread Primitives for Sequence Coordination 273
5.6.1 The Lost Notification Problem 273
5.6.2 Avoiding the Lost Notification Problem with Eventcounts and Sequencers 275
5.6.3 Implementing AWAIT, ADVANCE, TICKET, and READ (Advanced Topic) 280
5.6.4 Polling, Interrupts, and Sequence Coordination 282
5.7 Case Study: Evolution of Enforced Modularity in the Intel x86 284
5.7.1 The Early Designs: No Support for Enforced Modularity 285
5.7.2 Enforcing Modularity Using Segmentation 286
5.7.3 Page-Based Virtual Address Spaces 287
5.7.4 Summary: More Evolution 288
5.8 Application: Enforcing Modularity Using Virtual Machines 290
5.8.1 Virtual Machine Uses 290
5.8.2 Implementing Virtual Machines 291
5.8.3 Virtualizing Example 293
Exercises 294

CHAPTER 6 Performance 299
Overview 300
6.1 Designing for Performance 300
6.1.1 Performance Metrics 302
6.1.2 A Systems Approach to Designing for Performance 304
6.1.3 Reducing Latency by Exploiting Workload Properties 306
6.1.4 Reducing Latency using Concurrency 307
6.1.5 Improving Throughput: Concurrency 309
6.1.6 Queuing and Overload 311
6.1.7 Fighting Bottlenecks 313
6.1.8 An Example: The I/O Bottleneck 316
6.2 Multilevel Memories 321
6.2.1 Memory Characterization 322
6.2.2 Multilevel Memory Management using Virtual Memory 323
6.2.3 Adding Multilevel Memory Management to a Virtual Memory 327
6.2.4 Analyzing Multilevel Memory Systems 331
6.2.5 Locality of Reference and Working Sets 333
6.2.6 Multilevel Memory Management Policies 335
6.2.7 Comparative Analysis of Different Policies 340
6.2.8 Other Page-Removal Algorithms 344
6.2.9 Other Aspects of Multilevel Memory Management 346

6.3 Scheduling 347
6.3.1 Scheduling Resources 348
6.3.2 Scheduling Metrics 349
6.3.3 Scheduling Policies 352
6.3.4 Case Study: Scheduling the Disk Arm 360
Exercises 362

About Part II **369**

Appendix A: The Binary Classification Trade-off **371**

Suggestions for Further Reading **375**

Problem Sets **425**

Glossary **475**

Index of Concepts **513**

PART II [ON-LINE]

Preface to Part II

CHAPTER 7 The Network as a System and as a System Component
Overview
7.1 Interesting Properties of Networks
7.2 Getting Organized: Layers
7.3 The Link Layer
7.4 The Network Layer
7.5 The End-to-end Layer
7.6 A Network System Design Issue: Congestion Control
7.7 Wrapping up Networks
7.8 Case Study: Mapping the Internet to the Ethernet
7.9 War Stories: Surprises in Protocol Design
Exercises

CHAPTER 8 Fault Tolerance: Reliable Systems from Unreliable Components
Overview
8.1 Faults, Failures, and Fault-Tolerant Design
8.2 Measures of Reliability and Failure Tolerance
8.3 Tolerating Active Faults
8.4 Systematically Applying Redundancy

8.5 Applying Redundancy to Software and Data
8.6 Wrapping up Reliability
8.7 Application: A Fault Tolerance Model for RAM
8.8 War Stories: Fault-Tolerant Systems that Failed
Exercises

CHAPTER 9 Atomicity: All-or-nothing and Before-or-after
Overview
9.1 Atomicity
9.2 All-or-Nothing Atomicity I: Concepts
9.3 All-or-Nothing Atomicity II: Pragmatics
9.4 Before-or-After Atomicity I: Concepts
9.5 Before-or-After Atomicity II: Pragmatics
9.6 Atomicity across Layers and Multiple Sites
9.7 Case Studies: Machine Language Atomicity
9.8 A More Complete Model of Disk Failure (Advanced Topic)
Exercises

CHAPTER 10 Consistency
Overview
10.1 Constraints and Interface Consistency
10.2 Cache Coherence
10.3 Durable Storage Revisited: Geographically Separated Replicas
10.4 Reconciliation
10.5 Perspectives
Exercises

CHAPTER 11 Information Security
Overview
11.1 Introduction to Secure Systems
11.2 Authenticating Principals
11.3 Authenticating Messages
11.4 Message Confidentiality
11.5 Security Protocols
11.6 Authorization: Controlled Sharing
11.7 Reasoning about Authentication (Advanced Topic)
11.8 Summary
11.9 Cryptography as a Building Block (Advanced Topic)
11.10 Case Study: Transport Layer Security (TLS) for the Web
11.11 War Stories: Security System Breaches
Exercises

Suggestions for Further Reading

Problem Sets

Glossary

Complete Index to Parts I and II