

# DISTRIBUTED SYSTEMS

Second Edition

Andrew S. Tanenbaum  
Maarten Van Steen

*Vrije Universiteit  
Amsterdam, The Netherlands*

PEARSON

Prentice  
Hall

Upper Saddle River, NJ 07458

# CONTENTS

## PREFACE

XVII

## INTRODUCTION

|       |                                    |    |
|-------|------------------------------------|----|
| .1    | DEFINITION OF A DISTRIBUTED SYSTEM | 2  |
| 12    | GOALS                              | 3  |
| 1.2.1 | Making Resources Accessible        | 3  |
| 1.2.2 | Distribution Transparency          | 4  |
| 1.2.3 | Openness                           | 7  |
| 1.2.4 | Scalability                        | 9  |
| 1.2.5 | Pitfalls                           | 16 |
|       | TYPES OF DISTRIBUTED SYSTEMS       | 17 |
| 1.3.1 | Distributed Computing Systems      | 17 |
| 1.3.2 | Distributed Information Systems    | 20 |
| 1.3.3 | Distributed Pervasive Systems      | 24 |
|       | SUMMARY                            | 30 |

## ARCHITECTURES

33

|       |                                         |    |
|-------|-----------------------------------------|----|
| »     | ARCHITECTURAL STYLES                    | 34 |
|       | SYSTEM ARCHITECTURES                    | 36 |
| 2.2.1 | Centralized Architectures               | 36 |
| 2.2.2 | Decentralized Architectures             | 43 |
| 2.2.3 | Hybrid Architectures                    | 52 |
|       | ARCHITECTURES VERSUS MIDDLEWARE         | 54 |
| 2.3.1 | Interceptors                            | 55 |
| 2.3.2 | General Approaches to Adaptive Software | 57 |
| 2.3.3 | Discussion                              | 58 |

|            |                                                            |            |
|------------|------------------------------------------------------------|------------|
| 2.4        | SELF-MANAGEMENT IN DISTRIBUTED SYSTEMS                     | 59         |
| 2.4.1      | The Feedback Control Model                                 | 60         |
| 2.4.2      | Example: Systems Monitoring with Astrolabe                 | 61         |
| 2.4.3      | Example: Differentiating Replication Strategies in Globule | 63         |
| 2.4.4      | Example: Automatic Component Repair Management in Jade     | 65         |
| 2.5        | SUMMARY                                                    | 66         |
| <b>3</b>   | <b>PROCESSES</b>                                           | <b>69</b>  |
| 3.1        | THREADS                                                    | 70         |
| 3.1.1      | Introduction to Threads                                    | 70         |
| 3.1.2      | Threads in Distributed Systems                             | <b>75</b>  |
| <b>3.2</b> | <b>VIRTUALIZATION</b>                                      | <b>79</b>  |
| 3.2.1      | The Role of Virtualization in Distributed Systems          | 79         |
| 3.2.2      | Architectures of Virtual Machines                          | 80         |
| 3.3        | CLIENTS                                                    | 82         |
| 3.3.1      | Networked User Interfaces                                  | 82         |
| 3.3.2      | Client-Side Software for Distribution Transparency         | <b>87</b>  |
| 3.4        | SERVERS                                                    | 88         |
| 3.4.1      | General Design Issues                                      | 88         |
| 3.4.2      | Server Clusters                                            | 92         |
| 3.4.3      | Managing Server Clusters                                   | 98         |
| 3.5        | CODE MIGRATION                                             | 103        |
| 3.5.1      | Approaches to Code Migration                               | 103        |
| 3.5.2      | Migration and Local Resources                              | 107        |
| 3.5.3      | Migration in Heterogeneous Systems                         | 110        |
| 3.6        | SUMMARY                                                    | 112        |
| <b>4</b>   | <b>COMMUNICATION</b>                                       | <b>115</b> |
| 4.1        | FUNDAMENTALS                                               | 116        |
| 4.1.1      | Layered Protocols                                          | 116        |
| 4.1.2      | Types of Communication                                     | 124        |
| <b>4.2</b> | <b>REMOTE PROCEDURE CALL</b>                               | <b>125</b> |
| 4.2.1      | Basic RPC Operation                                        | 126        |
| 4.2.2      | Parameter Passing                                          | 130        |

|            |                                                 |            |
|------------|-------------------------------------------------|------------|
| 4.2.3      | Asynchronous RPC                                | 134        |
| 4.2.4      | Example: DCE RPC                                | 135        |
| <b>4.3</b> | <b>MESSAGE-ORIENTED COMMUNICATION</b>           | <b>140</b> |
| 4.3.1      | Message-Oriented Transient Communication        | 141        |
| 4.3.2      | Message-Oriented Persistent Communication       | 145        |
| 4.3.3      | Example: IBM's WebSphere Message-Queuing System | 152        |
| <b>4.4</b> | <b>STREAM-ORIENTED COMMUNICATION</b>            | <b>157</b> |
| 4.4.1      | Support for Continuous Media                    | 158        |
| 4.4.2      | Streams and Quality of Service                  | 160        |
| 4.4.3      | Stream Synchronization                          | 163        |
| <b>4.5</b> | <b>MULTICAST COMMUNICATION</b>                  | <b>166</b> |
| 4.5.1      | Application-Level Multicasting                  | 166        |
| 4.5.2      | Gossip-Based Data Dissemination                 | 170        |
| <b>4.6</b> | <b>SUMMARY</b>                                  | <b>175</b> |

## **5 NAMING 179**

|       |                                    |     |
|-------|------------------------------------|-----|
| 5.1   | NAMES, IDENTIFIERS, AND ADDRESSES  | 180 |
| 5.2   | FLAT NAMING                        | 182 |
| 5.2.1 | Simple Solutions                   | 183 |
| 5.2.2 | Home-Based Approaches              | 186 |
| 5.2.3 | Distributed Hash Tables            | 188 |
| 5.2.4 | Hierarchical Approaches            | 191 |
| 5.3   | STRUCTURED NAMING                  | 195 |
| 5.3.1 | Name Spaces                        | 195 |
| 5.3.2 | Name Resolution                    | 198 |
| 5.3.3 | The Implementation of a Name Space | 202 |
| 5.3.4 | Example: The Domain Name System    | 209 |
| 5.4   | ATTRIBUTE-BASED NAMING             | 217 |
| 5.4.1 | Directory Services                 | 217 |
| 5.4.2 | Hierarchical Implementations: LDAP | 218 |
| 5.4.3 | Decentralized Implementations      | 222 |
| 5.5   | SUMMARY                            |     |

## **6 SYNCHRONIZATION 231**

- 6.1 CLOCK SYNCHRONIZATION 232
  - 6.1.1 Physical Clocks 233
  - 6.1.2 Global Positioning System 236
  - 6.1.3 Clock Synchronization Algorithms 238
- 6.2 LOGICAL CLOCKS 244
  - 6.2.1 Lamport's Logical Clocks 244
  - 6.2.2 Vector Clocks 248
- 6.3 MUTUAL EXCLUSION 252
  - 6.3.1 Overview 252
  - 6.3.2 A Centralized Algorithm 253
  - 6.3.3 A Decentralized Algorithm 254
  - 6.3.4 A Distributed Algorithm 255
  - 6.3.5 A Token Ring Algorithm 258
  - 6.3.6 A Comparison of the Four Algorithms 259
- 6.4 GLOBAL POSITIONING OF NODES 260
- 6.5 ELECTION ALGORITHMS 263
  - 6.5.1 Traditional Election Algorithms 264
  - 6.5.2 Elections in Wireless Environments 267
  - 6.5.3 Elections in Large-Scale Systems 269
- 6.6 SUMMARY 270

## **7 CONSISTENCY AND REPLICATION 275**

- 7.1 INTRODUCTION 274
  - 7.1.1 Reasons for Replication 274
  - 7.1.2 Replication as Scaling Technique 275
- 7.2 DATA-CENTRIC CONSISTENCY MODELS 276
  - 7.2.1 Continuous Consistency 277
  - 7.2.2 Consistent Ordering of Operations 281
- 7.3 CLIENT-CENTRIC CONSISTENCY MODELS 288
  - 7.3.1 Eventual Consistency 289
  - 7.3.2 Monotonic Reads 291
  - 7.3.3 Monotonic Writes 292
  - 7.3.4 Read Your Writes 294
  - 7.3.5 Writes Follow Reads 295

|            |                                         |            |
|------------|-----------------------------------------|------------|
| <b>7.4</b> | <b>REPLICA MANAGEMENT</b>               | <b>296</b> |
| 7.4.1      | Replica-Server Placement                | 296        |
| 7.4.2      | Content Replication and Placement       | 298        |
| 7.4.3      | Content Distribution                    | 302        |
| <b>7.5</b> | <b>CONSISTENCY PROTOCOLS</b>            | <b>306</b> |
| 7.5.1      | Continuous Consistency                  | 306        |
| 7.5.2      | Primary-Based Protocols                 | 308        |
| 7.5.3      | Replicated-Write Protocols              | 311        |
| 7.5.4      | Cache-Coherence Protocols               | 313        |
| 7.5.5      | Implementing Client-Centric Consistency | 315        |
| <b>7.6</b> | <b>SUMMARY</b>                          | <b>317</b> |

## **8 FAULT TOLERANCE**

**321**

|            |                                             |            |
|------------|---------------------------------------------|------------|
| <b>8.1</b> | <b>INTRODUCTION TO FAULT TOLERANCE</b>      | <b>322</b> |
| 8.1.1      | Basic Concepts                              | 322        |
| 8.1.2      | Failure Models                              | 324        |
| 8.1.3      | Failure Masking by Redundancy               | 326        |
| <b>8.2</b> | <b>PROCESS RESILIENCE</b>                   | <b>328</b> |
| 8.2.1      | Design Issues                               | 328        |
| 8.2.2      | Failure Masking and Replication             | 330        |
| 8.2.3      | Agreement in Faulty Systems                 | 331        |
| 8.2.4      | Failure Detection                           | 335        |
| <b>8.3</b> | <b>RELIABLE CLIENT-SERVER COMMUNICATION</b> | <b>336</b> |
| 8.3.1      | Point-to-Point Communication                | 337        |
| 8.3.2      | RPC Semantics in the Presence of Failures   | 337        |
| <b>8.4</b> | <b>RELIABLE GROUP COMMUNICATION</b>         | <b>343</b> |
| 8.4.1      | Basic Reliable-Multicasting Schemes         | 343        |
| 8.4.2      | Scalability in Reliable Multicasting        | 345        |
| 8.4.3      | Atomic Multicast                            | 348        |
| <b>8.5</b> | <b>DISTRIBUTED COMMIT</b>                   | <b>355</b> |
| 8.5.1      | Two-Phase Commit                            | 355        |
| 8.5.2      | Three-Phase Commit                          | 360        |
| <b>8.6</b> | <b>RECOVERY</b>                             | <b>363</b> |
| 8.6.1      | Introduction                                | 363        |
| 8.6.2      | Checkpointing                               | 366        |

- 8.6.3 Message Logging 369
- 8.6.4 Recovery-Oriented Computing 372
- 8.7 SUMMARY 373

## **9 SECURITY 377**

- 9.1 INTRODUCTION TO SECURITY 378
  - 9.1.1 Security Threats, Policies, and Mechanisms 378
  - 9.1.2 Design Issues 384
  - 9.1.3 Cryptography 389
- 9.2 SECURE CHANNELS 396
  - 9.2.1 Authentication 397
  - 9.2.2 Message Integrity and Confidentiality 405
  - 9.2.3 Secure Group Communication 408
  - 9.2.4 Example: Kerberos 411
- 9.3 ACCESS CONTROL 413
  - 9.3.1 General Issues in Access Control 414
  - 9.3.2 Firewalls 418
  - 9.3.3 Secure Mobile Code 420
  - 9.3.4 Denial of Service 427
- 9.4 SECURITY MANAGEMENT 428
  - 9.4.1 Key Management 428
  - 9.4.2 Secure Group Management 433
  - 9.4.3 Authorization Management 434
- 9.5 SUMMARY 439

## **10 DISTRIBUTED OBJECT-BASED SYSTEMS 443**

- 10.1 ARCHITECTURE 443
  - 10.1.1 Distributed Objects 444
  - 10.1.2 Example: Enterprise Java Beans 446
  - 10.1.3 Example: Globe Distributed Shared Objects 448
- 10.2 PROCESSES 451
  - 10.2.1 Object Servers 451
  - 10.2.2 Example: The Ice Runtime System 454

|        |                                                 |     |
|--------|-------------------------------------------------|-----|
| 10.3   | COMMUNICATION                                   | 456 |
| 10.3.1 | Binding a Client to an Object                   | 456 |
| 10.3.2 | Static versus Dynamic Remote Method Invocations | 458 |
| 10.3.3 | Parameter Passing                               | 460 |
| 10.3.4 | Example: Java RMI                               | 461 |
| 10.3.5 | Object-Based Messaging                          | 464 |
| 10.4   | NAMING                                          | 466 |
| 10.4.1 | CORBA Object References                         | 467 |
| 10.4.2 | Globe Object References                         | 469 |
| 10.5   | SYNCHRONIZATION                                 | 470 |
| 10.6   | CONSISTENCY AND REPLICATION                     | 472 |
| 10.6.1 | Entry Consistency                               | 472 |
| 10.6.2 | Replicated Invocations                          | 475 |
| 10.7   | FAULT TOLERANCE                                 | 477 |
| 10.7.1 | Example: Fault-Tolerant CORBA                   | 477 |
| 10.7.2 | Example: Fault-Tolerant Java                    | 480 |
| 10.8   | SECURITY                                        | 481 |
| 10.8.1 | Example: Globe                                  | 482 |
| 10.8.2 | Security for Remote Objects                     | 486 |
| 10.9   | SUMMARY                                         | 487 |

## 11 DISTRIBUTED FILE SYSTEMS

491

|        |                                        |     |
|--------|----------------------------------------|-----|
| 11.1   | ARCHITECTURE                           | 491 |
| 11.1.1 | Client-Server Architectures            | 491 |
| 11.1.2 | Cluster-Based Distributed File Systems | 496 |
| 11.1.3 | Symmetric Architectures                | 499 |
| 11.2   | PROCESSES                              | 501 |
| 11.3   | COMMUNICATION                          | 502 |
| 11.3.1 | RPCsinNFS                              | 502 |
| 11.3.2 | The RPC2 Subsystem                     | 503 |
| 11.3.3 | File-Oriented Communication in Plan 9  | 505 |
| 11.4   | NAMING                                 | 506 |
| 11.4.1 | Naming in NFS                          | 506 |
| 11.4.2 | Constructing a Global Name Space       | 512 |



|        |                                           |     |
|--------|-------------------------------------------|-----|
| 11.5   | SYNCHRONIZATION                           | 513 |
| 11.5.1 | Semantics of File Sharing                 | 513 |
| 11.5.2 | File Locking                              | 516 |
| 11.5.3 | Sharing Files in Coda                     | 518 |
| 11.6   | CONSISTENCY AND REPLICATION               | 519 |
| 11.6.1 | Client-Side Caching                       | 520 |
| 11.6.2 | Server-Side Replication                   | 524 |
| 11.6.3 | Replication in Peer-to-Peer File Systems  | 526 |
| 11.6.4 | File Replication in Grid Systems          | 528 |
| 11.7   | FAULT TOLERANCE                           | 529 |
| 11.7.1 | Handling Byzantine Failures               | 529 |
| 11.7.2 | High Availability in Peer-to-Peer Systems | 531 |
| 11.8   | SECURITY                                  | 532 |
| 11.8.1 | Security in NFS                           | 533 |
| 11.8.2 | Decentralized Authentication              | 536 |
| 11.8.3 | Secure Peer-to-Peer File-Sharing Systems  | 539 |
| 11.9   | SUMMARY                                   | 541 |

## **12 DISTRIBUTED WEB-BASED SYSTEMS**

**545**

|        |                                     |     |
|--------|-------------------------------------|-----|
| 12.1   | ARCHITECTURE                        | 546 |
| 12.1.1 | Traditional Web-Based Systems       | 546 |
| 12.1.2 | Web Services                        | 551 |
| 12.2   | PROCESSES                           | 554 |
| 12.2.1 | Clients                             | 554 |
| 12.2.2 | The Apache Web Server               | 556 |
| 12.2.3 | Web Server Clusters                 | 558 |
| 12.3   | COMMUNICATION                       | 560 |
| 12.3.1 | Hypertext Transfer Protocol         | 560 |
| 12.3.2 | Simple Object Access Protocol       | 566 |
| 12.4   | NAMING                              | 567 |
| 12.5   | SYNCHRONIZATION                     | 569 |
| 12.6   | CONSISTENCY AND REPLICATION         | 570 |
| 12.6.1 | Web Proxy Caching                   | 571 |
| 12.6.2 | Replication for Web Hosting Systems | 573 |
| 12.6.3 | Replication of Web Applications     | 579 |

- 12.7 FAULT TOLERANCE 582
- 12.8 SECURITY 584
- 12.9 SUMMARY 585

## **13 DISTRIBUTED COORDINATION-BASED SYSTEMS 589**

- 13.1 INTRODUCTION TO COORDINATION MODELS 589
- 13.2 ARCHITECTURES 591
  - 13.2.1 Overall Approach 592
  - 13.2.2 Traditional Architectures 593
  - 13.2.3 Peer-to-Peer Architectures 596
  - 13.2.4 Mobility and Coordination 599
- 13.3 PROCESSES 601
- 13.4 COMMUNICATION 601
  - 13.4.1 Content-Based Routing 601
  - 13.4.2 Supporting Composite Subscriptions 603
- 13.5 NAMING 604
  - 13.5.1 Describing Composite Events 604
  - 13.5.2 Matching Events and Subscriptions 606
- 13.6 SYNCHRONIZATION 607
- 13.7 CONSISTENCY AND REPLICATION 607
  - 13.7.1 Static Approaches 608
  - 13.7.2 Dynamic Replication 611
- 13.8 FAULT TOLERANCE 613
  - 13.8.1 Reliable Publish-Subscribe Communication 613
  - 13.8.2 Fault Tolerance in Shared Dataspaces 616
- 13.9 SECURITY 617
  - 13.9.1 Confidentiality 618
  - 13.9.2 Secure Shared Dataspaces 620
- 13.10 SUMMARY 621