

Open distributed systems

Jon Crowcroft

University College London



Contents

<i>Preface</i>	ix
<i>Acknowledgements</i>	xiv
I What are open distributed systems and what are they for? I	
1.1 Introduction	1
1.2 The viewpoints	3
1.3 Transparencies	6
1.4 Central operating system services	8
1.5 Communications support	14
1.6 Open Communications	16
1.7 Open distributed systems	17
1.8 Objects as a modelling concept	17
1.9 Summary	22
1.10 Exercises	22
2 Modules, communication and concurrency	25
2.1 Introduction	25
2.2 Addressing, naming and routing	26
2.3 Concurrent systems	29
2.4 Interleaving and true parallelism	31
2.5 Shared resources – problems with concurrency	32
2.6 Mutual exclusion	34
2.7 Consumers, producers and critical regions	34
2.8 Monitors, semaphores and rendezvous	37
2.9 Distributed systems and concurrency	41
2.10 Worked example of networked windows	42
2.11 Remote procedure call	43
2.12 Summary	54
2.13 Exercises	54
3 Real time and reliable systems	55
3.1 Introduction	55
3.2 The object model and fault transparency	56
3.3 Conventional hardware and software reliability	58
3.4 Software in distributed systems	60
3.5 Contracting for reliability	60
3.6 Analyzing timing constraints	61
3.7 Transactions	63
3.8 Timestamps	67
3.9 More about multiple-readers/single-writer and the object model	69
3.10 Commitment, concurrency and recovery (CCR)	71
3.11 Fault tolerance	72
3.12 Object migration	75
3.13 Exception handling	75
3.14 Summary	76
3.15 Exercises	77

4	The nature of security	79
	<i>with Robert Cole, Hewlett-Packard Laboratories, Bristol</i>	
4.1	Threats and protection	80
4.2	Access control and authentication	82
4.3	Authorization	85
4.4	Access control schemes	85
4.5	Trust in a secure system	87
4.6	General models of computer security	88
4.7	Cryptography	91
4.8	Key distribution	96
4.9	Practical security approaches?	96
4.10	Summary	97
4.11	Exercises	97
5	Languages and formal methods	99
	<i>with Mark d'Inverno, Westminster University</i>	
5.1	Why protocol description?	99
5.2	Why protocol specification?	99
5.3	Format and protocol languages	100
5.4	Protocol validation	101
5.5	Language of temporal ordering specification (LOTOS)	102
5.6	Variables, values and expressions and LOTOS	105
5.7	Estelle	105
5.8	Communicating sequential processes (CSP)	107
5.9	Introduction to the specification language Z	113
5.10	A multimedia conference specification in Z	115
5.11	Summary	137
5.12	Exercises	137
6	Communications support	139
	<i>with Graham Knight, University College London</i>	
6.1	Introduction	139
6.2	Technological point of view	140
6.3	Clocks and time in distributed systems	143
6.4	Communications system modelling	144
6.5	Protocols	145
6.6	Service types	147
6.7	Relationships between services	148
6.8	The ISO reference model	149
6.9	Naming, addressing and routing	153
6.10	Connection-oriented or connectionless?	155
6.11	Programming interfaces	157
6.12	OSI application-layer support for distributed systems	158
6.13	A distributed system example	178
6.14	OSI - a critique	181
6.15	Networked windowing systems	182
6.16	Summary	184
6.17	Exercises	185

7	CORBA – An industrial approach to open distributed computing	187
	<i>Nigel Edwards HPI/APM</i>	
7.1	The Object Management Architecture (OMA)	187
7.2	The Common Object Request Broker Architecture (CORBA)	189
7.3	Interoperability	212
7.4	Common Object Services Specification	214
7.5	Common facilities	221
7.6	Some practical issues	222
7.7	Summary	226
8	Modelling and implementing distributed multimedia conferencing	227
	<i>with Mark Handley, UCL, and Ian Wakeman, University of Sussex</i>	
8.1	Introduction	227
8.2	Multicast requirements for distributed applications	227
8.3	Shared networked objects and windows	230
8.4	Classical IPC usage for multimedia conference control	238
8.5	Weak RPC	242
8.6	Event-driven approaches	243
8.7	Related work	245
8.8	The MICE design of conferencing communications channel	250
8.9	Summary	270
8.10	Exercises	270
9	Applications to network management	271
	<i>with David Lewis, University College London</i>	
9.1	Introduction	271
9.2	Functions	273
9.3	Conceptual architectures	276
9.4	Viewpoints of the architecture	278
9.6	Aspects of interoperability	280
9.7	The single managed-object view	281
9.8	Notifications	283
9.9	Behaviour	283
9.10	Specification of managed-object classes	284
9.11	Registration	284
9.12	The managed object relationships view	285
9.13	Relationships	285
9.14	Relationship definition and representation	287
9.15	The logical distribution view	287
9.16	Peer-to-peer OS relationships	289
9.17	Abstraction	289
9.18	Management hierarchies	290
9.19	Authority domains	290
9.20	The engineering viewpoint	290
9.21	The interoperable interface and the OS	290
9.22	Communication protocols	291
9.23	Management information	292
9.24	Mapping the service onto the architecture	293
9.25	Management protocols	293

CONTENTS

9.26	Applying ODP and CORBA directly to management	294
9.27	Distributed systems for managing networks	294
9.28	Embedded management functionality	295
9.29	Summary	295
9.30	Exercises	295
10	Distributed file systems	297
	<i>with Nermeen Ismail, University College London</i>	
10.1	Virtual file system model	297
10.2	Consequences of the open model	299
10.3	File (system) naming	299
10.4	Replication	300
10.5	Management	301
10.6	The network file system (NFS)	302
10.7	The Andrew file system (AFS)	307
10.8	Media file system	309
10.9	Security	313
10.10	Summary	314
10.11	Exercises	314
11	Load balancing	315
	<i>with S. Hailes, University College London</i>	
11.1	Introduction	315
11.2	ODP/ANSA migration	315
11.3	Monitoring distributed systems	315
11.4	Migration	320
11.5	Scheduling	338
11.6	Summary	354
11.7	Exercises	354
12	Future lessons and challenges	355
12.1	Introduction	355
12.2	Definition	355
12.3	Lessons and challenges	355
12.4	Micro/nano/maxi kernels – how small is beautiful?	356
12.5	Blocking, synchronous and asynchronous interfaces – easy?	357
12.6	Remote Procedure Call – latent potential?	357
12.7	Is atomic group communication useful?	357
12.8	Shared address spaces and distributed memory models	358
12.9	Client–server paradigm	358
12.10	Unix – just another program?	358
12.11	Is caching at the server good?	359
12.12	Fileserver replication – how many, when?	359
12.13	Message passing – too hard for application programmers?	359
12.14	Client caching – is it a good idea?	359
12.15	Atomicity – useful, but too expensive?	360
12.16	Causal ordering in multi-party communication	360
12.17	Threads and processes – kernel and user space	360
12.18	Thread versus process	361
12.21	Log structured file systems (LFSs)	362
	<i>References</i>	363
	<i>Index</i>	369