25 Cambridge Computer Science Texts

The Principles of Computer Networking

D. Russell

Cambridge Computer Science Texts

The Principles of Computer Networking

D. Russell
Computing Laboratory, University of Newcastle upon Tyne

The right of the University of Cambridge to print and sell all manner of books was granted by Henry VIII in 1534. The University has printed and published continuously since 1584.

Cambridge University Press

Cambridge
New York Port Chester Melbourne Sydney

Published by the Press Syndicate of the University of Cambridge
The Pitt Building, Trumpington Street, Cambridge CB2 1RP
40 West 20th Street, New York, NY 10011, USA
10 Stamford Road, Oakleigh, Melbourne 3166, Australia

First published 1989

Printed in Great Britain at the University Press, Cambridge

British Library cataloguing in publication data available

Library of Congress cataloguing in publication data available

ISBN 0 521 32795 4 hard covers
ISBN 0 521 33992 8 paperback

Also in this series

1 An Introduction to Logical Design of Digital Circuits
C. M. Reeves 1972

2 Information Representation and Manipulation in a Computer
E. S. Page and L. B. Wilson, Second Edition 1978

3 Computer Simulation of Continuous Systems
R. J. Ord-Smith and J. Stephenson 1975

4 Macro Processors
A. J. Cole, Second Edition 1981

5 An Introduction to the Uses of Computers
Murray Laver 1976

6 Computing Systems Hardware
M. Wells 1976

7 An Introduction to the Study of Programming Languages
D. W. Barron 1977

8 ALGOL 68 – A first and second course
A. D. McGettrick 1978

9 An Introduction to Computational Combinatorics
E. S. Page and L. B. Wilson 1979

10 Computers and Social Change
Murray Laver 1980

11 The Definition of Programming Languages
A. D. McGettrick 1980

12 Programming via Pascal
J. S. Rohl and H. J. Barrett 1980

13 Program Verification using Ada
A. D. McGettrick 1982

14 Simulation Techniques for Discrete Event Systems
I. Mitrani 1982

15 Information Representation and Manipulation using Pascal
E. S. Page and L. B. Wilson 1983

16 Writing Pascal Programs
J. S. Rohl 1983

17 An Introduction to APL
S. Pommier 1983

18 Computer Mathematics
D. J. Cooke and H. E. Bez 1984

19 Recursion via Pascal
J. S. Rohl 1984

20 Text Processing
A. Colin Day 1984

21 Introduction to Computer Systems
Brian Molinari 1985

22 Program Construction
R. G. Stone and D. J. Cooke 1987

23 A Practical Introduction to Denotational Semantics
Lloyd Allison 1987

24 Modelling of Computer and Communication Systems
I. Mitrani 1987

Preface

This book is intended to cover the whole of the field of computer communications. Even with such a wide ranging ambition, there must be limits to the coverage. Roughly these are drawn at the lower end of the spectrum by assuming the properties of transmission media, and at the top end by stopping short of discussing truly distributed processing. In between, the aim has been to give an overall understanding of the principles involved.

Computer communications is such a vast and fast moving field that it is quite impossible to cover the details of any complete architecture within the confines of one book. However, that was never my intention. The real aim is to try to extract some of the *principles* that emerge in computer networking, sometimes over and over again. One prime example is the topic of flow control. The principles of flow control can be extracted independently of context, and this book devotes Chapter 4 to just that. Similarly, in Chapter 3 we look at what is at first sight a bewildering range of ways of sharing a medium. However, it soon emerges that most of their number can be reduced by asking two orthogonal questions—is the topology a bus or a ring, and is the access by contention or by token? Once these two principles have been understood, the other aspects are of secondary importance. Again, the addressing and routing principles that we discuss in Chapter 6 in relation to the techniques of providing the network service, crop up again when looking at network mail in Chapter 10, and Gateways in Chapter 15.

In overall plan the book starts at the bottom and works upwards. Thus, the early chapters through Chapter 6 explain how raw, error-prone communications media can be built into a reliable end-to-end communications service. However, the subject of communications has only just started by this time. For successful communication to take place, agreements have to be made about what the bits on this reliable pipe actually mean. This brings us to the consideration of presentation issues which are concerned with a consistent representation of the same semantic information across diverse systems. We also go into considerable detail in describing how various applications of computer connections actually work, including computer mail, terminal support, and file transfer and access. The application level tools provided by ISO, and in particular the remote operations service are explained.

As well as the "traditional" kinds of protocol with which this book is mainly concerned, a chapter is devoted to looking at so-called *lightweight protocols*, and the performance and system issues involved in the efficient implementation of communications architectures. In addition, since networks have grown so big, they need to be managed, and some of the management issues are also discussed.

Perhaps unusually, a whole chapter is devoted to Security, Authentication, and Encryption. This topic was given such treatment because the author perceives that current attitudes show a huge ignorance of what can and can't be done in this area. Computer networks are insecure, and becoming more so. However, many people either seem to be very ignorant of what can be achieved by encryption, or, naively, seem to be ready to put complete trust into encryption techniques. In 1978, Needham and Schroeder showed how, through the use of encryption techniques, two mutually unknown network entities could authenticate themselves, one to the other through the agency of a third, trusted, authentication server. This is independent of "hostile" agents observing all the messages, and inserting, corrupting, or replaying messages. Most people are very surprised by this ability, and it is surprisingly poorly known even 10 years after its first publication. On the other hand, some people put an unquestioning trust in encryption, apparently oblivious of the consistent history of broken cyphers. In addition, the interesting properties of public key encryption, together with their present disappointing position is also presented. Chapter 14 aims to give an up-to-date review of what can and cannot be done in this area.

Finally, Chapter 16 tries to summarise the need for standards, and the processes, political and technical, by which standards are produced and imposed. In addition to discussing the various political bodies that produce standards, Chapter 16 also discusses some of the techniques by which standards are described and analysed. Natural human language is inadequate and leads to ambiguity and misunderstanding, and Chapter 16 indicates some of the improved tools that are beginning to appear.

Throughout the book, the principles are illustrated by examples taken from real computer architectures. The emphasis is always on generally agreed standards, and so the examples come mainly from the ISO and ARPANET suites of protocols. The attempt has been to avoid proprietary architectures wherever possible, and the choice of the ISO and ARPANET suites is for two main reasons. One is that they are publicly and widely available, and the other is that they often have quite different approaches to solving the problems in hand. When different approaches are used on the same problem, then a careful analysis of where differences lie often illuminates the real character of the problem and reveals the principles involved.

A book like this owes much to the efforts of others. Perhaps the most help have been those with whom the author has come into contact over the long years that he has worked in designing and implementing networks. I have been privileged to encounter many luminaries, and it would be invidious to mention any individually since at least ten times as many would of necessity not be mentioned. However, I hope I may be allowed the indulgence of mentioning just two. I was privileged to know Bob Husak of the Merit Computer Network. Bob had a deep and encyclopaedic knowledge of networks at all levels, and was a good friend. His early death saddened all who knew him. In my undergraduate life as a physicist, Thomas Littlefield taught me how to get a feel for complex physical processes by employing simple mental pictures. It is an approach I have valued ever since.

Directly involved in the production of this book have been Harry Whitfield who fooled me into starting the project in the first place, and pointed out many silly mistakes, Quentin Campbell, Jill Foster, Isi Mitrani, and especially Ian Doak have read various drafts and given suggestions. From CUP, Ernest Kirkwood, Tim Bradshaw, and David Tranah have supported me during the preparation of this book.

I shall not break with the worthy tradition of thanking the typist Denis Russell for typing and revising endless revisions of the manuscript, with only the occasional hint of dissatisfaction with his lot. In addition, he typeset the text using TEX†, produced all the drawings in PostScript‡, and produced camera ready copy. Of course this leaves even less room than normal for the author to disclaim responsibility for errors. More to the point, various tools were used including text editors on several operating systems (even including UNIX§), micros, workstations and mainframes too numerous to mention, and communications systems including most of those described between these covers.

Finally, and most of all, I would like to thank my wife Marion, and the kids for putting up both with me and without me during the excessively protracted gestation period of this book. I can only hope it was worth their efforts.

† TEXis a trademark of the American Mathematical Society
‡ PostScript is a trademark of Adobe Systems Incorporated
§ UNIX is a trademark of Bell Laboratories

Contents

1

Data Transmission

This chapter examines the ways in which data is transmitted along communications channels or lines. The subject of data transmission is a large and complex area of electronic engineering, and a chapter of this size is intended only to give a flavour of some of the principles involved. The intention is first to describe how characters can be represented as bits, and then see how bits can be transmitted down communications channels. We shall touch lightly on Fourier Analysis and the fundamental results of Nyquist and Shannon. After a look at this theoretical basis, the chapter proceeds to examine some of the more common methods of transmitting data along communications lines. This leads to a discussion of the types of errors that we can expect to encounter. It will also be the grounding to which we shall refer when we come to discuss some of the various sorts of shared medium in Chapter 3.

1.1 Character Representation

The simplest form of communication between computers and human beings is by means of characters that are assembled into text of some sort. Other means of communication, such as graphs, sound, mice, pointing, joysticks, colour, and so on have increasingly been employed. However, in simple computer communications, text is still the most important currency.

It is normal to represent text inside a computer by choosing a set of symbols, the character set, and allocating each one to a unique number. These unique numbers are then stored in locations called bytes. A byte is a group of eight bits within a computer, and is thus capable of representing a number in the range 0 to 255 inclusive. Bytes are sometimes called "octets".

Fig 1.1 shows the ASCII character set. The letters "ASCII" stand for the American Standard Code for Information Interchange. (There are many other national and international codes that are either identical to ASCII, or are very close and differ only in the representation of a few graphic symbols. There are also others, such as ISO 8859-1 that are more extensive, and will probably replace ASCII. However, at the time of writing, ASCII is the most widely used character set of the many we might have chosen.) We can see from the table that there are 128 characters arranged in eight columns of 16. The columns are numbered 0 to 7 from left to right, and the rows 0 to 15 from top to bottom. To find the binary representation of a particular character, we multiply the column number by 16 and add the row number. Thus, for example, the letter "K" is in the 11^{th} row of the 4^{th} column. $4 \times 16 + 11 = 75$ (decimal) = 1001011 (binary). It is simpler to use hexadecimal numbering. Then "K" is in the B^{th} row of the 4^{th} column, and we may immediately write the hexadecimal number as #4B. (#nn is the notation used in this book to denote hexadecimal numbers, base sixteen)

		0	1	2	3	4	5	6	7
0	0	NUL	DLE		0	@	P	`	p
1	1	SOH	DC1	!	1	A	Q	a	q
2	2	STX	DC2	"	2	B	R	b	r
3	3	ETX	DC3	#	3	C	S	c	s
4	4	EOT	DC4	$	4	D	T	d	t
5	5	ENQ	NAK	%	5	E	U	e	u
6	6	ACK	SYN	&	6	F	V	f	v
7	7	BEL	ETB	'	7	G	W	g	w
8	8	BS	CAN	(	8	H	X	h	x
9	9	HT	EM	)	9	I	Y	i	y
10	A	LF	SUB	*	:	J	Z	j	z
11	B	VT	ESC	+	;	K	[	k	{
12	C	FF	FS	,	<	L	\	l	|
13	D	CR	GS	-	=	M	]	m	}
14	E	SO	RS	.	>	N	^	n	~
15	F	SI	US	/	?	O	_	o	DEL

Fig 1.1. The ASCII Character Set

The most important thing that we need to observe is the technique of representing a set of characters by a corresponding set of small binary patterns that fit into bytes. Thus, depending upon the circumstances, a given bit pattern in a byte may be interpreted either as a small integer, or as a particular graphic character. The text of this book has all at various times been represented as ASCII characters within the memory of several different computers (as well as various other representations).

There are 94 printable characters, plus the space character, in columns 2 to 7. These are the ones with the single character entry in the table. There are another 33 locations that do not represent printable characters, but have mysterious two- or three-character "names". These are the so-called *control characters.* The control characters are codes that are reserved for special functions. Thus, #0D represents the *Carriage Return*—CR function, #0A represents the *Line Feed*—LF function, and so on. These types of functions are directly related to the basic operations that are performed on simple character display devices.

For example, this text was originally typed at a simple video display terminal. On this terminal, the CR function returns the cursor to the left hand side of the screen, and the LF function moves the cursor down one line. These types of terminal and the simple means of communication that they employ are directly descended from the telex and telegraph machines that were developed over many years for sending character information over great distances. Computer communications has adopted and adapted this technology for its own uses.

In these early applications simple character machines communicated directly with each other over long distances. A typical teleprinter for example, consisted of a keyboard, a printing mechanism for printing on rolls of paper, and frequently a paper tape reader and punch for reading or punching holes in paper tape. In the early days of computing, programs were often prepared on rolls of paper tape on such machines, and then fed into the computer as ASCII encoded characters read directly off the paper tapes. Knowing this historical background makes it easier to appreciate some of the otherwise surprising aspects of character codes such as ASCII.

Codes that betray this history are such things as "ENQ"—enquire of the machine its *answerback code*. This code was a unique code on such machines as the Teletype† which was set as radial legs on a bakelite wheel, and was sent in response to the ENQ as a way of automatic self-identification. DC1, DC2, DC3, DC4 were device control codes that were sent to a device with a paper tape reader or punch, and caused this device to switch on or off.

Of course, there are many machines without paper tape equipment and thus the meanings of some of the control codes change with time. For example, DC1 and DC3 can be used to control other ancillary devices, such as cassette tapes. Alternatively, they now almost universally have the meaning not of controlling the function of an ancillary device, but of controlling the flow of the actual data. Thus, it is now the common convention that DC3 means *"Stop the flow of data to me for a while"*, and DC1 means *"OK you can start sending again now"*. Common alternative names for these control characters are "X-off" and "X-on" respectively.

The punched paper tape background neatly explains the strange character "DEL" at position #7F. The paper tapes were often prepared manually, ready for being sent automatically some time later. Human typing is notoriously error prone, and it essential that some form of

† Teletype is a trademark of the Teletype corporation