Macmillan Computer Science Series

Consulting Editor Professor F.H. Sumner, University of Manchester

- S.T. Allworth and R.N. Zobel, *Introduction to Real-time Software Design, second edition*
Ian O. Angell and Gareth Griffith, *High-resolution Computer Graphics Using FORTRAN 77*
Ian O. Angell and Gareth Griffith, *High-resolution Computer Graphics Using Pascal*
M. Azmoodeh, *Abstract Data Types and Algorithms*
C. Bamford and P. Curran, *Data Structures, Files and Databases*
Philip Barker, *Author Languages for CAL*
A.N. Barrett and A.L. Mackay, *Spatial Structure and the Microcomputer*
R.E. Berry, B.A.E. Meekings and M.D. Soren, *A Book on C, second edition*
P. Beynon-Davies, *Information Systems Development*
G.M. Birtwistle, *Discrete Event Modelling on Simula*
B.G. Blundell, C.N. Daskalakis, N.A.E. Heyes and T.P. Hopkins, *An Introductory Guide to Silvar Lisco and Hilo Simulators*
B.G. Blundell and C.N. Daskalakis, *Using and Administering an Apollo Network*
T.B. Boffey, *Graph Theory in Operations Research*
Richard Bornat, *Understanding and Writing Compilers*
Linda E.M. Brackenbury, *Design of VLSI Systems – A Practical Introduction*
G.R. Brookes and A.J. Stewart, *Introduction to occam 2 on the Transputer*
J.K. Buckle, *Software Configuration Management*
W.D. Burnham and A.R. Hall, *Prolog Programming and Applications*
P.C. Capon and P.J. Jinks, *Compiler Engineering Using Pascal*
J.C. Cluley, *Interfacing to Microprocessors*
J.C. Cluley, *Introduction to Low-level Programming for Microprocessors*
Robert Cole, *Computer Communications, second edition*
Derek Coleman, *A Structured Programming Approach to Data*
Andrew J.T. Colin, *Fundamentals of Computer Science*
Andrew J.T. Colin, *Programming and Problem-solving in Algol 68*
S.M. Deen, *Fundamentals of Data Base Systems*
S.M. Deen, *Principles and Practice of Database Systems*
C. Delannoy, *Turbo Pascal Programming*
Tim Denvir, *Introduction to Discrete Mathematics for Software Engineering*
P.M. Dew and K.R. James, *Introduction to Numerical Computation in Pascal*
D. England et al., *A Sun User's Guide*
K.C.E. Gee, *Introduction to Local Area Computer Networks*
J.B. Gosling, *Design of Arithmetic Units for Digital Computers*
M.G. Hartley, M. Healey and P.G. Depledge, *Mini and Microcomputer Systems*
Roger Hutty, *Z80 Assembly Language Programming for Students*
Roland N. Ibbett and Nigel P. Topham, *Architecture of High Performance Computers, Volume I*
Roland N. Ibbett and Nigel P. Topham, *Architecture of High Performance Computers, Volume II*
Patrick Jaulent, *The 68000 – Hardware and Software*
P. Jaulent, L. Baticle and P. Pillot, *68020-30 Microprocessors and their Coprocessors*
J.M. King and J.P. Pardoe, *Program Design Using JSP – A Practical Introduction*
E.V. Krishnamurthy, *Introductory Theory of Computer Science*
V.P. Lane, *Security of Computer-Based Information Systems*
Graham Lee, *From Hardware to Software – An Introduction To Computers*

continued overleaf

A.M. Lister and R.D. Eager, *Fundamentals of Operating Systems, fourth edition*
 Tom Manns and Michael Coleman, *Software Quality Assurance*
 Brian Meek, *Fortran, PL/1 and the Algols*
 A. Mével and T. Guéguen, *Smalltalk-80*
 Y. Nishinuma and R. Espesser, *UNIX – First Contact*
 Pim Oets, *MS-DOS and PC-DOS – A Practical Guide, second edition*
 A.J. Pilavakis, *UNIX Workshop*
 Christian Queinnec, *LISP*
 E.J. Redfern, *Introduction to Pascal for Computational Mathematics*
 Gordon Reece, *Microcomputer Modelling by Finite Differences*
 W.P. Salman, O. Tisserand and B. Toulout, *FORTH*
 L.E. Scales, *Introduction to Non-linear Optimization*
 Peter S. Sell, *Expert Systems – A Practical Introduction*
 A.G. Sutcliffe, *Human–Computer Interface Design*
 Colin J. Theaker and Graham R. Brookes, *A Practical Course on Operating Systems*
 M.R. Tolhurst et al., *Open Systems Interconnection*
 J-M. Trio, *8086–8088 Architecture and Programming*
 A.J. Tyrrell, *COBOL from Pascal*
 M.J. Usher, *Information Theory for Information Technologists*
 B.S. Walker, *Understanding Microprocessors*
 Colin Walls, *Programming Dedicated Microprocessors*
 I.R. Wilton and A.M. Addyman, *A Practical Introduction to Pascal – with BS6192, second edition*

Non-series

Roy Anderson, *Management, Information Systems and Computers*
 I.O. Angell, *Advanced Graphics with the IBM Personal Computer*
 J.E. Bingham and G.W.P Davies, *Planning for Data Communications*
 B.V. Cordingley and D. Chamund, *Advanced BASIC Scientific Subroutines*
 N. Frude, *A Guide to SPSS/PC+*
 Barry Thomas, *A PostScript Cookbook*

UNIX WORKSHOP

Andreas J. Pilavakis

Consultant

M
MACMILLAN

© Andreas J. Pilavakis 1989

All rights reserved. No reproduction, copy or transmission of this publication may be made without written permission.

No paragraph of this publication may be reproduced, copied or transmitted save with written permission or in accordance with the provisions of the Copyright Act 1956 (as amended), or under the terms of any licence permitting limited copying issued by the Copyright Licensing Agency, 33–4 Alfred Place, London WC1E 7DP.

Any person who does any unauthorised act in relation to this publication may be liable to criminal prosecution and civil claims for damages.

First published 1989

Published by
MACMILLAN EDUCATION LTD
Houndmills, Basingstoke, Hampshire RG21 2XS
and London
Companies and representatives
throughout the world

Laserset by
Ponting–Green Publishing Services, London

British Library Cataloguing in Publication Data

Pilavakis, Andreas J.
UNIX Workshop
1. Computer Systems. Operating systems: UNIX
I. Title.
005.4'3

ISBN 978-0-333-48851-5 ISBN 978-1-349-19900-6 (eBook)
DOI 10.1007/978-1-349-19900-6

UNIX is a trademark of AT&T Bell Laboratories.

Contents

<i>Preface</i>	<i>1</i>
1 High-level introduction to UNIX	3
1.1 The kernel	4
1.2 The shell	5
1.3 Logging in	6
1.4 After logging in	6
2 The UNIX file system	8
2.1 UNIX files	8
2.2 Determine the current directory	10
2.3 Change directory	10
2.4 Create a new directory	11
2.5 Remove a directory	11
2.6 Remove files	11
2.7 List contents of a directory	12
2.8 Concatenate and display files	12
2.9 View a file	12
2.10 Print a file	13
2.11 Copy a file	13
2.12 Move a file	13
2.13 Search for files in the file system	14
2.14 Link files	14
2.15 File system protection modes	15
2.16 File creation mask	17
2.17 The UNIX file system	18
2.18 Special files and devices	18
2.19 Mount a file system	19
2.20 Unmount a file system	19
3 The Bourne shell	21
3.1 Commands	21
3.2 Flag arguments	22
3.3 Background commands	22

3.4	Command grouping and execution	22
3.5	Shell metacharacters and quoting	23
3.6	Input/output redirection	23
3.7	File name generation and metacharacters	24
3.8	Pipelines and filters	25
3.9	Shell variables	26
3.10	Command substitution	28
3.11	Useful commands	28
3.12	Signals and traps	29
3.13	Use of Here documents	31
3.14	Shell programming constructs	32
3.15	Summary by shell script examples	35
4	Modifying the shell environment	41
4.1	Environment variables	41
4.2	C program environment	42
4.3	Modifying the shell parent environment	45
5	UNIX filters and awk	47
5.2	The awk utility	47
5.2	Multiple file pattern search	53
5.3	Line count	53
5.4	Comparing two files	54
5.5	Partitioning a file	54
5.6	Selecting fields from a file	55
5.7	Translation of characters	55
5.8	Merging lines from different files	56
5.9	Octal dumping of a file	57
5.10	Sorting files	57
6	The vi editor	59
6.1	Some special characters	59
6.2	Creating a file	59
6.3	Cursor movement along the screen	60
6.4	Searching	61
6.5	Editing lines of text	61
6.6	Moving and copying text	62
6.7	Repeating and undoing commands	63
6.8	Using commands of the ed editor	63
6.9	Setting options	65
6.10	Executing shell commands	65
7	Text formatting	66
7.1	“Nroff” the text formatter	66
7.2	Common “nroff” commands	67

7.3	"Nroff" macro construction	68
7.4	"Nroff" and the "mm" package	71
8	C language program execution	79
8.1	Compiling a source file	79
8.2	Compiling several files	80
8.3	The C preprocessor and conditional compilation	80
8.4	Program structure	82
8.5	Faster execution and stripping	85
8.6	Create libraries and link objects from libraries	85
8.7	Improving program execution	86
8.8	Compiling and using shell scripts	88
9	Standard C language library functions	89
9.1	Character testing functions	89
9.2	String functions	90
10	Interprocess communication features and system calls	95
10.1	The system call	95
10.2	The exit call	95
10.3	Exec system calls	96
10.4	Exec commands and metacharacters	97
10.5	Fork and wait calls	97
10.6	Signals	99
10.7	Suspending process execution	101
10.8	Implementing timeouts	101
10.9	Resuming program execution	103
10.10	Unnamed pipes	104
10.11	Pipe functions	104
10.12	Using low-level pipes	106
10.13	Named pipes	109
10.14	Report IPC facilities status	109
10.15	Remove information about IPC facilities	110
10.16	Message queues	110
10.17	Shared memory	115
10.18	Semaphores	117
11	Project generation and maintenance tools	127
11.1	Make	127
11.2	Writing makefiles	128
11.3	Default suffix list of make	129
11.4	Makefile macros	130
11.5	The built-in rules of make	134
11.6	Source code control system (SCCS)	135
11.7	Putting programs under SCCS control	136

11.8	Special SCCS symbols	138
11.9	SCCS and make	139
11.10	Further SCCS commands	140
12	Local communication features	142
12.1	Sending mail to other users	142
12.2	Reading mail	142
12.3	Interactive communication between users	143
12.4	Disable messages to your terminal	143
12.5	Send a message to all users	144
12.6	Private reminder service	144
12.7	Inform users of current events	145
12.8	Run commands at specified times	145
12.9	Execute commands on a scheduled basis	146
13	UNIX-to-UNIX copy	149
13.1	Uucp installation	149
13.2	Connection types	149
13.3	Uucp commands	150
13.4	Uucp files	150
13.5	The “password” and “inittab” files	153
13.6	Access mode of special files	154
13.7	A practical example	154
13.8	Transferring files	156
13.9	Problems and debugging	157
13.10	Retrieve received files	157
13.11	Call another system using cu	158
14	General administrative functions	160
14.1	Booting up procedure	160
14.2	Init–getty–login–shell sequence	161
14.3	UNIX system shutdown	162
14.4	Backing up versus archiving	163
14.5	Adding new users	166
14.6	Deleting users	168
14.7	Disk utilization	169
14.8	The UNIX file system structure	171
14.9	Create a new file system	173
14.10	Maintaining the file system	173
14.11	Adding new devices to the UNIX system	176
15	Accessing information from memory	178
15.1	Files used	178
15.2	Structures used	179
15.3	Routines used	179

15.4 Preparation procedure	179
15.5 Program listing	180
15.6 Program output	184
<i>Appendixes</i>	
<i>A Shell metacharacters</i>	187
<i>B Comparison of Bourne shell and C shell constructs</i>	189
<i>C ASCII tables</i>	191
<i>Index</i>	193

Preface

UNIX was developed in the early 1970s at Bell Laboratories of AT & T by Ken Thompson, Dennis Ritchie and a number of other scientists. The development of UNIX was influenced by the Multics operating system, which was running on a GE 645 computer in the late 1960s.

The earliest version of UNIX was running on a PDP-7 and was later moved to a PDP-11 in 1971. In the search for a portable language for the new operating system, Ritchie decided to develop the C language, which was based on language B, a derivative of BCPL. Language B was an interpretive language developed by Thompson.

In 1973, UNIX was rewritten in C; this was quite an innovative idea at the time. After this, the number of UNIX installations within AT & T increased in number. However, a number of legal judgements in the United States during the 1950s prohibited the general marketing of UNIX. For example, Western Electric (the marketing company of AT & T) was not allowed to give support or maintenance or any warranty for UNIX. As a result, UNIX was initially made available mainly to non-profit organizations and educational establishments for a minimal fee.

In 1975 Version-6, which incorporated multi-programming, became the first widely available version of UNIX. By 1977, UNIX was ported to non-DEC architectures such as the Interdata 8/34. Subsequent releases of UNIX include Version-7 in 1978, System-III in 1981 and the fully AT & T supported UNIX System-V version in 1983.

UNIX has enjoyed a long gestation period, during which time it was effectively under the control of its designers, while academic and other outside sources also contributed to its development. The end result was the introduction of a set of features that greatly increased the popularity of UNIX in the commercial market; some of these features are:

- A highly portable operating system
- A simple but powerful user interface
- An easily maintainable hierarchical file system
- A dynamic development environment in which programs can be developed and ported to other computers with little or no modification

2 *UNIX Workshop*

UNIX is a true multitasking, multiuser, highly portable system that provides a powerful and hospitable program development environment. The operating system has enough commercial backing from AT & T, major microprocessor manufacturers such as Intel and Motorola, system manufacturers, software developers and users to make it the leading operating system in the market.

The book is based on the author's UNIX experience in working as a training officer, lecturer, analyst/programmer and pre-sales and post-sales consultant.

The first nine chapters of the book address the needs of undergraduate students and anyone else who wishes to acquire a good working knowledge of UNIX. The remaining chapters in the book cover a number of important UNIX utilities which are used for the generation and maintenance of software packages. Advanced interprocess communication features such as semaphores, shared memory segments and message queues are also explained. These later chapters will appeal to UNIX users who are involved in the development of packages or the conversion of packages from other operating systems to UNIX.

The book covers a very wide spectrum of UNIX topics. This has been achieved by being concise and by using a multitude of practical examples, so as to avoid many pages of theoretical explanations. Even though the book starts from the fundamentals of UNIX before covering the very advanced features of the operating system, it is expected that the reader should have a basic understanding of at least one high-level language. For the more advanced topics in the book, the reader is expected to have a working knowledge of the C programming language.

I should like to thank my brother, Kyriacos, for his assistance and encouragement during the writing of the book and for extensive proof-reading.

I should also like to thank the following people for their proof-reading work: James L. Myers, John Appleby, Peter Tintel, Martin Swash and Jean-Pierre Bertinchamps.