Advanced Programming in the UNIX® Environment

Second Edition

W. Richard Stevens
Stephen A. Rago

PEARSON
Education

ISBN 978-81-317-0005-1

First Impression, 2006
Second Impression, 2007
Third Impression, 2008
Fourth Impression, 2008
Fifth Impression, 2009

Published by Dorling Kindersley (India) Pvt. Ltd., licensees of Pearson Education in South Asia.

Head Office: 482, F.I.E., Patparganj, Delhi 110 092, India.
Registered Office: 14 Local Shopping Centre, Panchsheel Park, New Delhi 110 017, India.

Printed in India by Baba Barkha Nath Printers.

Contents

Foreword

At some point during nearly every interview I give, as well as in question periods after talks, I get asked some variant of the same question: "Did you expect Unix to last for so long?" And of course the answer is always the same: No, we didn't quite anticipate what has happened. Even the observation that the system, in some form, has been around for well more than half the lifetime of the commercial computing industry is now dated.

The course of developments has been turbulent and complicated. Computer technology has changed greatly since the early 1970s, most notably in universal networking, ubiquitous graphics, and readily available personal computing, but the system has somehow managed to accommodate all of these phenomena. The commercial environment, although today dominated on the desktop by Microsoft and Intel, has in some ways moved from single-supplier to multiple sources and, in recent years, to increasing reliance on public standards and on freely available source.

Fortunately, Unix, considered as a phenomenon and not just a brand, has been able to move with and even lead this wave. AT&T in the 1970s and 1980s was protective of the actual Unix source code, but encouraged standardization efforts based on the system's interfaces and languages. For example, the SVID—the System V Interface Definition—was published by AT&T, and it became the basis for the POSIX work and its follow-ons. As it happened, Unix was able to adapt rather gracefully to a networked environment and, perhaps less elegantly, but still adequately, to a graphical one. And as it also happened, the basic Unix kernel interface and many of its characteristic user-level tools were incorporated into the technological foundations of the open-source movement.

It is important that papers and writings about the Unix system were always encouraged, even while the software of the system itself was proprietary, for example Maurice Bach's book, *The Design of the Unix Operating System*. In fact, I would claim that a central reason for the system's longevity has been that it has attracted remarkably talented writers to explain its beauties and mysteries. Brian Kernighan is one of these; Rich Stevens is certainly another. The first edition of this book, along with his series of books about networking, are rightfully regarded as remarkably well-crafted works of exposition, and became hugely popular.

However, the first edition of this book was published before Linux and the several open-source renditions of the Unix interface that stemmed from the Berkeley CSRG became widespread, and also at a time when many people's networking consisted of a serial modem. Steve Rago has carefully updated this book to account for the technology changes, as well as developments in various ISO and IEEE standards since its first publication. Thus his examples are fresh, and freshly tested.

It's a most worthy second edition of a classic.

Murray Hill, New Jersey
March 2005

Dennis Ritchie

Preface

Introduction

Rich Stevens and I first met through an e-mail exchange when I reported a typographical error in his first book, *UNIX Network Programming*. He used to kid me about being the person to send him his first errata notice for the book. Until his death in 1999, we exchanged e-mail irregularly, usually when one of us had a question we thought the other might be able to answer. We met for dinner at USENIX conferences and when Rich was teaching in the area.

Rich Stevens was a friend who always conducted himself as a gentleman. When I wrote *UNIX System V Network Programming* in 1993, I intended it to be a System V version of Rich's *UNIX Network Programming*. As was his nature, Rich gladly reviewed chapters for me, and treated me not as a competitor, but as a colleague. We often talked about collaborating on a STREAMS version of his *TCP/IP Illustrated* book. Had events been different, we might have actually done it, but since Rich is no longer with us, revising *Advanced Programming in the UNIX Environment* is the closest I'll ever get to writing a book with him.

When the editors at Addison-Wesley told me that they wanted to update Rich's book, I thought that there wouldn't be too much to change. Even after 13 years, Rich's work still holds up well. But the UNIX industry is vastly different today from what it was when the book was first published.

- The System V variants are slowly being replaced by Linux. The major system vendors that ship their hardware with their own versions of the UNIX System have either made Linux ports available or announced support for Linux. Solaris is perhaps the last descendant of UNIX System V Release 4 with any appreciable market share.

- After 4.4BSD was released, the Computing Science Research Group (CSRG) from the University of California at Berkeley decided to put an end to its development of the UNIX operating system, but several different groups of volunteers still maintain publicly available versions.
- The introduction of Linux, supported by thousands of volunteers, has made it possible for anyone with a computer to run an operating system similar to the UNIX System, with freely available source code for the newest hardware devices. The success of Linux is something of a curiosity, given that several free BSD alternatives are readily available.
- Continuing its trend as an innovative company, Apple Computer abandoned its old Mac operating system and replaced it with one based on Mach and FreeBSD.

Thus, I've tried to update the information presented in this book to reflect these four platforms.

After Rich wrote *Advanced Programming in the UNIX Environment* in 1992, I got rid of most of my UNIX programmer's manuals. To this day, the two books I keep closest to my desk are a dictionary and a copy of *Advanced Programming in the UNIX Environment*. I hope you find this revision equally useful.

Changes from the First Edition

Rich's work holds up well. I've tried not to change his original vision for this book, but a lot has happened in 13 years. This is especially true with the standards that affect the UNIX programming interface.

Throughout the book, I've updated interfaces that have changed from the ongoing efforts in standards organizations. This is most noticeable in Chapter 2, since its primary topic is standards. The 2001 version of the POSIX.1 standard, which we use in this revision, is much more comprehensive than the 1990 version on which the first edition of this book was based. The 1990 ISO C standard was updated in 1999, and some changes affect the interfaces in the POSIX.1 standard.

A lot more interfaces are now covered by the POSIX.1 specification. The base specifications of the Single UNIX Specification (published by The Open Group, formerly X/Open) have been merged with POSIX.1. POSIX.1 now includes several 1003.1 standards and draft standards that were formerly published separately.

Accordingly, I've added chapters to cover some new topics. Threads and multithreaded programming are important concepts because they present a cleaner way for programmers to deal with concurrency and asynchrony.

The socket interface is now part of POSIX.1. It provides a single interface to interprocess communication (IPC), regardless of the location of the process, and is a natural extension of the IPC chapters.

I've omitted most of the real-time interfaces that appear in POSIX.1. These are best treated in a text devoted to real-time programming. One such book appears in the bibliography.

I've updated the case studies in the last chapters to cover more relevant real-world examples. For example, few systems these days are connected to a PostScript printer via a serial or parallel port. Most PostScript printers today are accessed via a network, so I've changed the case study that deals with PostScript printer communication to take this into account.

The chapter on modem communication is less relevant these days. So that the original material is not lost, however, it is available on the book's Web site in two formats: PostScript (`http://www.apuebook.com/lostchapter/modem.ps`) and PDF (`http://www.apuebook.com/lostchapter/modem.pdf`).

The source code for the examples shown in this book is also available at `www.apuebook.com`. Most of the examples have been run on four platforms:

1. FreeBSD 5.2.1, a derivative of the 4.4BSD release from the Computer Systems Research Group at the University of California at Berkeley, running on an Intel Pentium processor
2. Linux 2.4.22 (the Mandrake 9.2 distribution), a free UNIX-like operating system, running on Intel Pentium processors
3. Solaris 9, a derivative of System V Release 4 from Sun Microsystems, running on a 64-bit UltraSPARC IIi processor
4. Darwin 7.4.0, an operating environment based on FreeBSD and Mach, supported by Apple Mac OS X, version 10.3, on a PowerPC processor

Acknowledgments

Rich Stevens wrote the first edition of this book on his own, and it became an instant classic.

I couldn't have updated this book without the support of my family. They put up with piles of papers scattered about the house (well, more so than usual), my monopolizing most of the computers in the house, and lots of hours with my face buried behind a computer terminal. My wife, Jeanne, even helped out by installing Linux for me on one of the test machines.

The technical reviewers suggested many improvements and helped make sure that the content was accurate. Many thanks to David Bausum, David Boreham, Keith Bostic, Mark Ellis, Phil Howard, Andrew Josey, Mukesh Kacker, Brian Kernighan, Bengt Kleberg, Ben Kuperman, Eric Raymond, and Andy Rudoff.

I'd also like to thank Andy Rudoff for answering questions about Solaris and Dennis Ritchie for digging up old papers and answering history questions. Once again, the staff at Addison-Wesley was great to work with. Thanks to Tyrrell Albaugh, Mary Franz, John Fuller, Karen Gettman, Jessica Goldstein, Noreen Regina, and John Wait. My thanks to Evelyn Pyle for the fine job of copyediting.

As Rich did, I also welcome electronic mail from any readers with comments, suggestions, or bug fixes.

Warren, New Jersey
April 2005

Stephen A. Rago
`sar@apuebook.com`

Preface to the First Edition

Introduction

This book describes the programming interface to the Unix system—the system call interface and many of the functions provided in the standard C library. It is intended for anyone writing programs that run under Unix.

Like most operating systems, Unix provides numerous services to the programs that are running—open a file, read a file, start a new program, allocate a region of memory, get the current time-of-day, and so on. This has been termed the *system call interface*. Additionally, the standard C library provides numerous functions that are used by almost every C program (format a variable's value for output, compare two strings, etc.).

The system call interface and the library routines have traditionally been described in Sections 2 and 3 of the *Unix Programmer's Manual*. This book is not a duplication of these sections. Examples and rationale are missing from the *Unix Programmer's Manual*, and that's what this book provides.

Unix Standards

The proliferation of different versions of Unix during the 1980s has been tempered by the various international standards that were started during the late 1980s. These include the ANSI standard for the C programming language, the IEEE POSIX family (still being developed), and the X/Open portability guide.

This book also describes these standards. But instead of just describing the standards by themselves, we describe them in relation to popular implementations of the standards—System V Release 4 and the forthcoming 4.4BSD. This provides a real-world description, which is often lacking from the standard itself and from books that describe only the standard.

Organization of the Book

This book is divided into six parts:

1. An overview and introduction to basic Unix programming concepts and terminology (Chapter 1), with a discussion of the various Unix standardization efforts and different Unix implementations (Chapter 2).
2. I/O—unbuffered I/O (Chapter 3), properties of files and directories (Chapter 4), the standard I/O library (Chapter 5), and the standard system data files (Chapter 6).
3. Processes—the environment of a Unix process (Chapter 7), process control (Chapter 8), the relationships between different processes (Chapter 9), and signals (Chapter 10).
4. More I/O—terminal I/O (Chapter 11), advanced I/O (Chapter 12), and daemon processes (Chapter 13).
5. IPC—Interprocess communication (Chapters 14 and 15).
6. Examples—a database library (Chapter 16), communicating with a PostScript printer (Chapter 17), a modem dialing program (Chapter 18), and using pseudo terminals (Chapter 19).

A reading familiarity with C would be beneficial as would some experience using Unix. No prior programming experience with Unix is assumed. This text is intended for programmers familiar with Unix and programmers familiar with some other operating system who wish to learn the details of the services provided by most Unix systems.

Examples in the Text

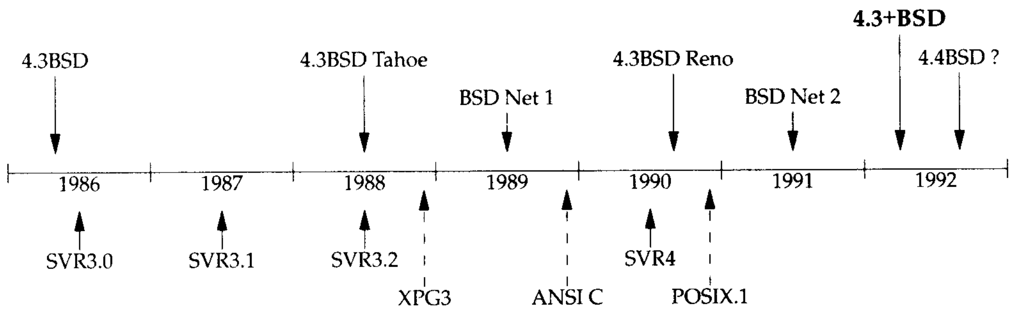
This book contains many examples—approximately 10,000 lines of source code. All the examples are in the C programming language. Furthermore, these examples are in ANSI C. You should have a copy of the *Unix Programmer's Manual* for your system handy while reading this book, since reference is made to it for some of the more esoteric and implementation-dependent features.

Almost every function and system call is demonstrated with a small, complete program. This lets us see the arguments and return values and is often easier to comprehend than the use of the function in a much larger program. But since some of the small programs are contrived examples, a few bigger examples are also included (Chapters 16, 17, 18, and 19). These larger examples demonstrate the programming techniques in larger, real-world examples.

All the examples have been included in the text directly from their source files. A machine-readable copy of all the examples is available via anonymous FTP from the Internet host `ftp.uu.net` in the file `published/books/stevens.advprog.tar.Z`. Obtaining the source code allows you to modify the programs from this text and experiment with them on your system.

Systems Used to Test the Examples

Unfortunately all operating systems are moving targets. Unix is no exception. The following diagram shows the recent evolution of the various versions of System V and 4.xBSD.



4.xBSD are the various systems from the Computer Systems Research Group at the University of California at Berkeley. This group also distributes the BSD Net 1 and BSD Net 2 releases—publicly available source code from the 4.xBSD systems. SVRx refers to System V Release x from AT&T. XPG3 is the X/Open Portability Guide, Issue 3, and ANSI C is the ANSI standard for the C programming language. POSIX.1 is the IEEE and ISO standard for the interface to a Unix-like system. We'll have more to say about these different standards and the various versions of Unix in Sections 2.2 and 2.3.

In this text we use the term *4.3+BSD* to refer to the Unix system from Berkeley that is somewhere between the BSD Net 2 release and 4.4BSD.

At the time of this writing, 4.4BSD was not released, so the system could not be called 4.4BSD. Nevertheless a simple name was needed to refer to this system and *4.3+BSD* is used throughout the text.

Most of the examples in this text have been run on four different versions of Unix:

1. Unix System V/386 Release 4.0 Version 2.0 ("vanilla SVR4") from U.H. Corp. (UHC), on an Intel 80386 processor.
2. 4.3+BSD at the Computer Systems Research Group, Computer Science Division, University of California at Berkeley, on a Hewlett Packard workstation.
3. BSD/386 (a derivative of the BSD Net 2 release) from Berkeley Software Design, Inc., on an Intel 80386 processor. This system is almost identical to what we call 4.3+BSD.
4. SunOS 4.1.1 and 4.1.2 (systems with a strong Berkeley heritage but many System V features) from Sun Microsystems, on a SPARCstation SLC.

Numerous timing tests are provided in the text and the systems used for the test are identified.

Acknowledgments

Once again I am indebted to my family for their love, support, and many lost weekends over the past year and a half. Writing a book is, in many ways, a family affair. Thank you Sally, Bill, Ellen, and David.

I am especially grateful to Brian Kernighan for his help in the book. His numerous thorough reviews of the entire manuscript and his gentle prodding for better prose hopefully show in the final result. Steve Rago was also a great resource, both in reviewing the entire manuscript and answering many questions about the details and history of System V. My thanks to the other technical reviewers used by Addison-Wesley, who provided valuable comments on various portions of the manuscript: Maury Bach, Mark Ellis, Jeff Gitlin, Peter Honeyman, John Linderman, Doug McIlroy, Evi Nemeth, Craig Partridge, Dave Presotto, Gary Wilson, and Gary Wright.

Keith Bostic and Kirk McKusick at the U.C. Berkeley CSRG provided an account that was used to test the examples on the latest BSD system. (Many thanks to Peter Salus too.) Sam Nataros and Joachim Sacksen at UHC provided the copy of SVR4 used to test the examples. Trent Hein helped obtain the alpha and beta copies of BSD/386.

Other friends have helped in many small, but significant ways over the past few years: Paul Lucchina, Joe Godsil, Jim Hogue, Ed Tankus, and Gary Wright. My editor at Addison-Wesley, John Wait, has been a great friend through it all. He never complained when the due date slipped and the page count kept increasing. A special thanks to the National Optical Astronomy Observatories (NOAO), especially Sidney Wolff, Richard Wolff, and Steve Grandi, for providing computer time.

Real Unix books are written using troff and this book follows that time-honored tradition. Camera-ready copy of the book was produced by the author using the groff package written by James Clark. Many thanks to James Clark for providing this excellent system and for his rapid response to bug fixes. Perhaps someday I will really understand troff footer traps.

I welcome electronic mail from any readers with comments, suggestions, or bug fixes.

Tucson, Arizona
April 1992

W. Richard Stevens
`rstevens@kohala.com`
`http://www.kohala.com/~rstevens`