BOOK REVIEWS

Editor: Wiley McKinzie, School of Computer Science and Technology, Rochester Institute of Technology, Rochester, NY 14623; Compmail, w.mckinzie; CSnet, wrm@rit

The Design of the Unix Operating System

Maurice J. Bach (Prentice-Hall, Inc., Englewood Cliffs, NJ, 1986, 471 pp., $31.95)

For years, the only publicly available description of the Unix operating system was the *Lyon's Book*, produced by the University of New South Wales. It contained a complete listing of the Unix source code plus a line-by-line commentary, and it was available to people without source licenses. Unfortunately, it only described Version 6 Unix, which by now is at least four versions out of date. Those of us who were lucky enough to have source licenses were at least able to see the source code, but it was not very understandable in the absence of a commentary.

Maurice Bach has written a book that at least alleviates our problems. Based on a course that he taught at AT&T Bell Laboratories in 1983 and 1984, the book is designed as a companion to the source code. The book explains the algorithms involved inside the kernel rather that providing a line-by-line commentary on the code. It is up-to-date in that it describes Unix System V Release 2, which is the most common AT&T variant of Unix used today. To add generality to the book, Bach also covers some features from Unix System V Release 3 and from the Berkeley Software Distributions (BSD) versions of Unix.

The book is a treasure trove of information. The first chapter consists of an overview of the Unix system for those people who are not familiar with it. The next chapter previews the rest of the book by describing first the structure of the kernel and then the process subsystem and the file subsystem. The rest of the book is divided into three sections. The first one is devoted to the file subsystem, with chapters on the buffer cache, the representation of files and the system call interface to the file system. The next section covers the process subsystem, including memory management, interprocess communications and the I/O subsystem. (A discussion of the I/O subsystem appears here instead of in the file subsystem section, where it might more logically belong. This arrangement allows Bach to discuss the issues of process control that accompany the writing of terminal drivers.) The final two chapters cover advanced topics—multiprocessor Unix systems and distributed Unix systems.

Before reading the book seriously, I skimmed it to try to get a feel for it. I found the presentation very pleasing. Bach has broken out the algorithms and presents them in a very readable pseudocode. He provides a lot of pictures describing how data structures fit together. At this first glance, the book seemed very readable. I was not disappointed.

After glancing over the table of contents, I wanted to see how good the coverage of the particular topics is. I have spent a lot of time writing device drivers and protocols, usually implemented as line disciplines, at least since the advent of System III. Unfortunately, none of the information that accompanied the release of System III said anything about the line disciplines other than that the entries for the routines belonged in the *linesw* table. Turning to the section on terminal drivers, the first thing that I saw was a bullet list of seven items describing what a line discipline had to accomplish. Here, already, was more documentation than AT&T had provided on the topic. The description is clear and succinct. It includes a step-by-step description of how to put characters on and take them off *clists*. For anyone who has ever spent time figuring out exactly how to manipulate all the pointers and counts, it's nice to see this procedure written down clearly.

From line disciplines, I went to an area that I was not so familiar with—*streams*. Streams are a new feature in System V, release 3 that were designed by Dennis Ritchie to provide more flexibility and modularity in the I/O subsystem. Bach provides a good description of what streams are and how they are intended to be used. He also provides a high-level description of how to use streams to implement windows on a terminal. This description draws on a paper that Pike wrote describing the Blit terminal. Bach concludes the section on streams with a discussion of why Ritchie implemented them the way that he did. In this discussion, he identifies some of the remaining shortcomings of the current implementation.

Bach's chapter on interprocess communications covers three distinct areas. The first is the use of the *ptrace* system call as a form of interprocess communications for debugging. He then discusses all of the System V IPC mechanisms, message queues, shared memory and semaphores as a group, due to their great similarity in interface. Lastly, he talks about network communications by concentrating on the Berkeley socket implementation.

The list of topics discussed can go on and on. As I stated earlier, the book is a real storehouse of information on Unix in particular and operating systems in general. Bach has culled out the essential pieces of Unix by describing the algorithms and data structures in a pseudocode that is easy to understand. He presents a very balanced view of Unix, often describing the shortcomings of the various features he discusses.

This book belongs on the shelf of everyone who is involved with the Unix operating system on a regular basis. Having it, you will have a much better understanding of how the system works, and therefore, how to make better use of the system. It is also useful for anyone who wants to know about operating systems. While designed solely for a course on Unix, it describes the major functions and features of operating systems, so that it could also be used for a general operating systems course.

The only complaint I have, and considering the purpose of the book it may not be a valid one, is the lack of discussion of Berkeley extensions to the Unix system. Berkeley has made extensions to Unix in three areas: the fast file system, the terminal handlers in the line discipline area, and the use of sockets for network communications. Of the three, only the last is covered to any extent. It would have been nice if the book covered the other two topics in the appropriate chapters, since no easily readable discussion of them is available anywhere.

Lorne H. Schachter
Bell Communications Research

Data Networks

Dimitri Bertsekas and Robert Gallager (Prentice-Hall, Inc., Englewood Cliffs, NJ, 1987, 486 pp., $37.33)

This book was written as a textbook in computer networking for senior undergraduates and masters level graduate students in computer networking. The teaching experience of the authors is evident in the careful development of topics in convenient, lesson-sized installments. Each chapter includes numerous problems to be worked out by the student. The authors have been careful to make these problems reasonable for the student to solve yet nontrivial in their relationship to real-world networking problems.

The book is equally suitable as a reference for those already in the field. It is well organized in the material it covers, and the index appropriately guides the reader to the relevant coverage. I would not, however, recommend this book to someone who is not already familiar with the issues and technology of computer networking. Some of the definitions used conflict with common usage. For example, the term *virtual circuit* is used in the old network-routing algorithms context rather than the higher level connection-oriented protocol context. While not a fatal flaw, such usage could confuse the naive reader.

More important than the misleading use of a few terms is the formal and stilted presentation style. The book is heavy reading. A reader unfamiliar with the jargon and acronyms of networking could find it next to impossible to comprehend. I often found myself reading passages several times in order to understand the authors' intent.

The difficult presentation style is doubly unfortunate, as the book otherwise is excellent. The authors set out to write a book which covered both the theoretical aspects of networking and the practical implementation of those theories. They succeeded on both counts and provide excellent coverage of many of the issues in low-level network protocol design, particularly those affecting performance. From queueing theory to optimal routing strategies, they develop the theory behind many of the challenges facing network designers, then build on that theory to show how the problems discussed are solved in real networks today.

The first 100 pages provide general coverage of basic networking: layering protocols, the International Standards Organization's (ISO) reference model for Open Systems Interconnection (OSI), physical aspects of communications media, error detection, error recovery, and framing. The remaining 350 pages consist of four chapters, one each on delay models, multiaccess communications, routing, and flow control. The depth of coverage is enhanced by the extensive, annotated references which appear at the end of each chapter. The following references for Section 4.2 are typical: "The Aloha network is first described in [ROB72]. The problem of stability was discussed in [MET 73], [Lak75], and [CaH75]. Binary exponential back off was developed in [MEB76]. Modern approaches to stability are treated in [HVL83] and [RIV85]." In all, the bibliography includes over 400 articles and books.

The total audience for this book is limited by the book's coverage of only the lower layers of the ISO OSI network architecture model. Given the profusion of standards available for these network layers, few practitioners need the depth of coverage provided by this book. However, for those interested in getting a better understanding of technical networking issues and an appreciation of the considerations that go into developing protocols at the lower networking layers, this book is right on target.

Vincent C. Jones

Big Blue: IBM's Use and Abuse of Power

Richard Thomas DeLamarter (Dodd Mead & Company, New York, 1986, 393 pp., $22.95)

If the purpose of a book review is to leave the review reader with a desire either to read or to ignore a book, then this review will likely meet its objective. The title of the book tells the whole story. If you hate IBM (big business, the defense budget, the Republican Party, etc.), then you will love this book. If you do not hate IBM (or any of the preceding), then you will probably dislike this book.

Even ignoring the title, the author's thesis is clear from the onset. In January, 1974, he was hired as an economist by the Department of Justice's Antitrust Division. Eight years later, he departed when the (Republican) government declared the case against IBM to be without merit. Interestingly, I too was hired at that time, but by the accused, with whom I spent the next seven years in various development and marketing positions. However, my perspective on the issues raised by the author, most of which are entirely legitimate, leads me to entirely different conclusions.

The first chapter, entitled "The Education of T.J. Watson," chronicles Watson's early days at NCR, which he left after he and other company executives were successfully prosecuted under the newly legislated antitrust laws. Throughout the book, DeLamarter incessantly refers to that period of Watson's career as his training in (monopolistic) crime by NCR's president, John Patterson. Other historians have positively characterized that period as the basis for IBM's Business Conduct Policies, the strictly enforced guidelines governing the business behavior of all IBM employees.

It is this perspective of the author that makes the book both interesting and irritating. Certainly, cynical analysts of IBM (who probably have never been employed by the company) imply that those policy guidelines are more often breached than followed. However, IBM employees know the serious consequences of such a breach. In fact, those guidelines which were so ingrained in me by my former employer underlie my own assessment of the clearly articulated arguments presented in this book.

DeLamarter leads the reader through a history of IBM products, including Hollerith's tabulating machines, unit record equipment, the first 700 series of computers, the System 360, and the System 370. His analyses of IBM mainframe, memory, and peripheral product announcement schedules and pricing policies obviously reflect the well considered and predetermined conclusions of the government's costly eight-year effort. On a page-by-page basis, I felt the compulsion to rebut those conclusions in this review.

Of particular interest to me were the conclusions drawn with respect to IBM's product policies in the mid- to late 1960's. Unlike many other computer manufacturers, IBM had no viable interactive (timesharing) product. The author contends that to avoid losing market share, IBM deliberately announced System 360-based timesharing products which were undeliverable, or did not work.

It is certainly true that IBM missed the boat in the original S360 design by not realizing the emerging importance of timesharing. But the author's simplistic conclusion (founded in his obvious predilection for seeing conspiracy under every rock) entirely ignores the technical difficulty of regressively altering this acknowledged state-of-the-art batch system in order to make it suitable for interactive use. Any IBM cus-

tomer who witnessed IBM's Herculean attempts to adapt those batch systems for interactive use, as I did as a graduate student at the end of the 1960's, will question the author's conclusions.

The book is weakened by two questionable implications. The first is the implication that IBM's competition simply had no chance against IBM's illegal practices, even when they could offer superior products. In 1967, as an employee of a large multinational timesharing company, I witnessed business practices against IBM that were at best questionable, but more likely illegal. This company, cited by the author as having a clearly superior timesharing offering, did not fail because of anything IBM did or did not do. This company failed because of weak management which was inexperienced in this new market. In those relatively early days of the computer industry, many large multinationals entered the market and withdrew a short time later. Were those failures all attributable to evildoing by IBM, or did other factors come into play?

The other questionable implication made by the author is that the marketplace had no freedom of choice in product selection. For a decade, beginning in the late 1950's, the field was wide open for both computer manufacturers and for customers. The nature of trade in the U.S. provides everyone with the freedom to choose any product or vendor. Why did so many new computer customers select IBM?

The author provides detailed analyses of IBM pricing which indicate that IBM enticed new customers with loss-leading, entry-level systems. He argues that IBM's strategy was to recoup those losses later by selling overpriced memory and peripherals to tied-in customers. In the absence of equivalent government-collected data to refute those analyses, my reaction to this admonishment is "So what?" IBM is not the first, and will certainly not be the last company to use loss leaders to attract business. And in any case, no customer is really an unwilling captive of any vendor. Company politics, followed by economics, usually determine the degree of capture felt by unsatisfied customers of any vendor. This principle is also true in the computer industry. The author portrays the computer marketplace as completely manipulable by IBM. This charge is clearly nonsense.

If you are predisposed to arguing with your colleagues about the evil monster, IBM, I strongly recommend that you study this book. It will provide you with substantial material for your assaults. On the other hand, if you are a vehement capitalist with an inclination to write a book, counterarguments to DeLamarter's work will provide you with more than enough material.

Sorel Reisman
California State University, Fullerton

Handbook of Software Quality Assurance

G. Gordon Schulmeyer and James I. McManus (eds.) (Van Nostrand Reinhold Company, New York, 1987, 454 pp., $32.95)

This handbook is a collection of 17 original papers on various aspects of software quality assurance (SQA), authored by 17 authorities in the field. And I do not use the term "authority" loosely.

The authors are all SQA professionals who deal with the implementation of software quality assurance on a daily basis. They include such noted authorities as Thomas J. McCabe ("Software Complexity Metric," *IEEE Software Engineering Transactions*, December, 1976), Robert H. Dunn (*Quality Assurance for Computer Software*, McGraw-Hill, 1982 (with Richard Ullman)), Matthew J. Fisher (*Software Quality Management*, Petrocelli Books, 1979 (with John D. Cooper)), and William E. Perry (*Effective Methods of EDP Quality Assurance*, Q.E.D. Information Sciences, 1981). In the Preface, the editors state that their objective was to produce a "collection of experiences and expectations of some of the best people in the software quality field," rather than a collection of stuffy theory written by philosophers. They have very definitely succeeded in accomplishing this objective.

The handbook is intended for use by the practicing SQA professional, but is suitable for use by the software professional needing assistance in producing a quality software product, since many of the essential aspects of the field are covered. It is logically divided in two sections, with a total of 17 chapters and two appendices.

The Preface contains a name index, a subject index, and a precis of each chapter to allow the professional to select the desired information. Each chapter also supplies footnoted references, general references, or both, for additional research.

The first section of the handbook (Chapters 1-8) "sets the stage," introducing definitions, describing the interaction of the software quality function with other software specialty areas, and providing an excellent overview of the government and industry standardization efforts, complete with outlines of the most current SQA standards and a table comparing the requirements of these different standards.

The second section (Chapters 9-17) focuses on tools, techniques, and methodologies for SQA implementation. This section includes topics such as design inspections, software configuration management, statistical methods applied to software (for software quality control), and internal standards development. In particular, Chapter 10 discusses the application of the Pareti Principle (concentrating on the vital few, not the trivial many) to SQA implementation, and Chapter 17 contains interesting and revealing results from a survey conducted by the Quality Assurance Institute of its members on software quality assurance. The chapters on software reliability and statistical methods contain some valuable new metrics.

However, do not misconstrue the comments above to imply that the nonsoftware professional could use the handbook for self-instruction. The handbook does not contain a software development primer as many of the publications on SQA do. It is designed to provide hints and experiences for SQA personnel and for those nonspecialists who are interested in the subject and are already familiar with software development methodologies. Since it is intended as a handbook and is written by a diverse group of professionals, some discontinuity occurs in the terms used (*SQAM* vs. *SQA* vs. *SQC*), and some personal bias is injected ("Over the years many individuals who possess a degree in education or the liberal arts have made the switch to software. . .They do, however, make poor SQA engineers.").

Despite these minor problems, the handbook is sufficiently self-contained, well organized, and interesting to read. Although it is not a primer on software quality, I plan to use it as a resource for a graduate level course I recently designed in Software Quality Assurance Engineering (Amber University, Garland, Texas). In addition, I have found a number of useful techniques in the handbook and intend to keep it handy.

I feel that this handbook is definitely a worthwhile addition to the library of any SQA professional—he or she can expect to gain from the expertise and experience presented.

Vincent L. Boyer, Sr.
E-Systems, Inc., Garland Division