TER Volume 4, Issue 12, December 15, 1997

ter

telecommunications electronic reviews

Volume 4, Issue 12, December 15, 1997

Telecommunications Electronic Reviews (TER) is a publication of the Library and Information Technology Association.

Contents:

ter issues (/lita/publications/archive/ter)

REVIEW OF: Douglas E. Comer and David L. Stevens. Internetworking with TCP/IP Volume III: Client-Server Programming and Applications- Windows Sockets Version. Upper Saddle River, NJ: Prentice-Hall, 1997.

by Deborah Bailey

This volume, the third edition of Volume III of the Internetworking with TCP/IP series, continues in the tradition of earlier volumes and editions. It does not disappoint and delivers on its stated objective, which is to "describe how to design and build distributed applications." (p. 7) The authors state that they "wrote this text to meet the demand from programmers who are building software for personal computers," and they have kept their stated audience in mind. (p. xxv)

This is not a beginner's text. It was written as a textbook for a senior or graduate level course in introductory networking, and, as such, each chapter contains suggestions for further study and exercises that are relevant to the materials just presented. Readers should have experience in programming and an understanding of basic networking principles.

The book is made up of twenty-eight chapters, two appendices, and an extensive bibliography. It is meant to be read sequentially, with later chapters building upon the foundations laid in earlier ones. The reader is made aware in the beginning of each chapter why the information presented is important and how that data is relevant to preceding and succeeding material. Summaries of important concepts are presented in indented and italicized form for emphasis.

Text, illustrations, and code are all vital parts of the book. No information is redundant, and the reader's time is not wasted with flowery, oversimplified explanations. Perhaps the only thing missing is a glossary of acronyms and terminology.

The early chapters, 1 through 7, contain important definitions of concepts and provide the background necessary to proceed. Examples of these concepts include concurrent processing and the client/ server paradigm. First, concurrent processing is described as the apparently simultaneous handling of two or more processes. In other words, a server must be able to carry out more than one operation at a time. Concurrency is virtually simultaneous because of the speed with which microprocessors can handle data.

Second, the client/server paradigm is presented as the basis for most computer communication. Within the client/server paradigm, applications are divided into two categories: one, the client, in which the application initiates communication, and one, the server, which waits for incoming communication. In this text the server is not a piece of hardware; it is a program with special coding which handles the issues of authentication, authorization, data security, privacy, and protection.

In this early portion of the text the authors begin their useful practice of pointing out the several options available to the programmer and then discussing the positive and negative aspects of each option. It is also very helpful that they distinguish between theory and practice. Several times throughout the text it is pointed out that though an idea would appear to work, in practice the realities of processing call for code to be written in a different manner. Readers are also reminded in this early section that not all code needs to be rewritten from scratch with every new application.

The second portion of the text, chapters 8 through 15, discusses the design of server software and illustrate concepts by providing complete programs that implement basic design ideas. Concepts are presented for both User Datagram Protocol (UDP) servers and Transmission Control Protocol (TCP) servers. UDP is a protocol that converts data messages into packets to be sent by IP (Internet Protocol) but does not verify that they have been delivered correctly. TCP is the protocol that breaks up data messages into packets to be sent by IP, reassembles the data, and verifies that it was received correctly.

Throughout this section algorithms are presented to provide the programmer with a basic sequence of steps to follow to create the server software. The code is written in C and is executable. It is an integral part of the text and should not be simply glanced over. Code is provided for services such as TIME, DAYTIME, and ECHO. The initial presentation of the code is simplistic but grows in complexity as new concepts are explored.

An important feature to readers is the code that is available on the Web (http://www.cs.purdue.edu/homes/comer/books.html (http://www.cs.purdue.edu/homes/comer/books.html)). You can either view it on screen or print it off so you have a continuous printout--in other words, no flipping of pages. This is a nice touch, especially since at the site you can find the table of contents and code from the other two editions of this work. This will allow users of this material to make comparisons and draw parallels between platforms.

The third portion of the text, chapters sixteen through eighteen, deals with the necessity of concurrency in client software and techniques for communication between clients and servers which may not connect directly to a TCP/IP Internet. It is necessary to become familiar with these techniques because of the relatively slow development of networking. Companies and other organizations have put together networks in a piecemeal fashion, leading to different hardware, software, and communication protocols. Two methods for handling communication across these varied networks are tunneling and gateways. Tunneling involves wrapping a data packet from one protocol in the packet of another. This procedure circumvents differing protocol restrictions. Gateways, devices which connect networks that have different communication protocols, are explained and their usefulness examined in chapter eighteen.

The fourth portion of the text, chapters nineteen through twenty-five, present to the reader External Data Representation (XDR), the Remote Procedure Call (RPC), and the Network File System (NFS). These three de facto standards have been developed by Sun. The first, XDR, specifies data formats for most of the types of data to be exchanged and how an object is encoded for those types. The second, RPC, which is also known as Open Network Computing Remote Procedure Call (ONC RPC), includes a compiler that assists programmers in building distributed programs automatically. It also defines the format of messages sent by the client to invoke a remote procedure, the format of the results to be sent, and the format of arguments. Chapters twenty-one and twenty-two focus on distributed program generation using RPC and specifically its automatic program generation tool, rpcgen. The eight steps involved in building a distributed program are applied using a dictionary application as an example.

Lastly, NFS, the Unix file structure it is derived from, and the mount procedure are examined. NFS is a mechanism which allows a machine to run a server that allows access to some or all of its files by remote applications. It essentially makes possible the same actions one would expect to implement on a local machine (e.g., read, write, open, and close). The Unix file structure was used by NFS designers while defining individual operations; therefore, in order to understand NFS, one must understand the Unix file system. The mount protocol is used by NFS for security purposes. It provides authentication for clients and allows them to find the handle for the root of a file hierarchy.

The fifth portion of the text, chapters twenty-five and twenty-six, pull together all of the concepts of the previous chapters using a Telnet client. The design of the software, its thread structure, the use of TCP for communication on the same computer, and keyboard mapping are discussed. Three finite state machines which define the interpretation and syntax of command sequences are used to control processing. The code and discussion of implementation details, such as displaying incoming data, printing status information, and responding to illegal commands, are found in chapter twenty-six.

The final two chapters deal with real world challenges for application programmers. Chapter twenty-seven deals with the common challenge of porting (rewriting software so that it will be able to run on a different computer) applications from Unix to Windows. Some of the items to be aware of include whether a piece of software operates in the background like a server, shared descriptors, detachment of the controlling terminal, changing directories, file protection, process groups, I/O (input/output) descriptors, mutual exclusion, recording of process identifiers, child process termination, and the use of syslog to record messages.

Chapter twenty-eight deals with deadlock and starvation, two basic problems in the client/server environment. Deadlock occurs when two programs are each waiting for a response from the other before continuing. This can be prevented by specifying which side should be responsible for initiating an interaction, then by either using a reliable transport protocol (TCP) or by inserting a timer mechanism which will set the maximum amount of time that a sender will wait for a response before retransmitting a request.

Starvation occurs when some clients can obtain access to a service while others cannot. This may be caused by idle connections, specification of a buffer size which is too small for the data expected, requests for the server to transmit data, or reading incoming data slowly or not at all until the buffer is full. Idle connections can be resolved by timers, but the last three problems require that the server be concurrent or avoid making calls that block. The warnings and explanations of causes for these problems, if remembered, should save time and resources for the programmer.

This text is full of information, with very little wasted space. It should be on the bookshelf of those who program applications in the Windows environment. The Web site that contains the code is of great use as well, and students should be encouraged to visit it, as there is more than just code available. Essays by the author and other information make it a worthwhile site to visit. Students using this text, with proper faculty guidance, should be well prepared for the challenges that face them.

Deborah Bailey (dbailey@choctaw.astate.edu (mailto:dbailey@choctaw.astate.edu)) is Head of Media Services at Dean B. Ellis Library, Arkansas State University.

table of contents ter issues (/lita/publications/archive/ter)

REVIEW OF: Chuck Musciano and Bill Kennedy. HTML: the Definitive Guide. 2nd ed. Cambridge, England: O'Reilly & Associates, 1997.

by Brad Eden

In our technology-driven society, having Internet connectivity and an Internet presence is almost essential in order to compete for the time, money, and interest of the public. Knowledge of Hypertext Markup Language (HTML) and how to write it is one of the many job skills that almost everyone needs to have in today's marketplace. Many people, while having the ability to use and manipulate icons and programs on the computer, do not feel comfortable or computer-literate enough to tackle the skills involved in learning and

writing HTML. The need to learn the skills to produce a home page on the Internet has driven our society to produce easy-to-understand books and software to assist people in the use of HTML. It is of interest, then, that a book can be published with the title HTML: the Definitive Guide.

According to the authors, no experience is necessary in any computer language before picking up this book. Beginning HTML writers are encouraged to read Chapter 1, "HTML and the World Wide Web," while those familiar with the Web and having some experience with HTML are encouraged to start with Chapter 2, "HTML Quick Start," where a brief overview of the most important HTML features are given. The authors state from the beginning that the book is based on Version 3.2 of HTML, and that they understand that the language is constantly being revised. They therefore give a WWW address (http://www.ora.com/info/html/ (http://www.ora.com/info/html/)) for readers to check out updates and corrections to the book.

The fifteen chapters move from a very basic to a very comprehensive understanding of the HTML language. The book is written in much the same style as computer programming books, with numerical divisions by chapter, section, and paragraphs (e.g., Chapter 3, 3.1, 3.1.2). I found the author's style to be very understandable in the early chapters, but as the book progressed into the more detailed aspects of HTML language, I had a hard time following the line of reasoning. The six appendices are a great resource for anyone using HTML, as they deal with HTML grammar, an HTML tag quick reference, a cascading style sheet properties quick reference, the HTML 3.2 DTD, character entities, and color names and values. A detachable HTML desktop quick reference guide is available at the end of the book.

Overall, this book is a definitive reference work on HTML, and should be available in most libraries. It is fairly understandable for both the beginner and advanced writer of HTML, is well written, and can show any patron how to create and maintain documents on the World Wide Web.

Dr. Brad Eden (BEDEN@MAIL.NHMCCD.CC.TX.US (mailto:BEDEN@MAIL.NHMCCD.CC.TX.US)) is the Coordinator of Technical Services/Automated Library Services at North Harris Montgomery Community College District in Houston, Texas.

table of contents ter issues (/lita/publications/archive/ter)

REVIEW OF: J.M. Ivler with Kamran Husain. CGI Developer's Resource: Web Programming in TCL and PERL. Upper Saddle River, NJ: Prentice-Hall, 1997.

by John Wynstra

The Common Gateway Interface (CGI) is a protocol that allows interaction between Web browsers and programs that reside on Web servers. J.M. Ivler's book CGI Developer's Resource: Web Programming in TCL and PERL is written for those who "want to learn about CGI programming" (p. xiii) and who "want to learn why things are done and not just how things are done." (p. xiii) CGI is not a programming language; in fact, CGI programs can be written in many different programming languages: Tcl (Tool Command Language), Perl (Practical Extraction and Report Language), Java, C, and C++, to name but a few. The

author selected Tcl and Perl because of their popularity and portability. The book was not written to teach the Tcl or Perl programming languages, but rather features Tcl and Perl programs to illustrate CGI programming techniques.

The book is 597 pages long; paperback bound; and includes three appendices, an index, and a CD-ROM, while a Web site, maintained by the author, provides updates and bug information. The CD-ROM contains the code for all of the programs that are discussed in the book as well as distributions of Perl and Tcl. The reader needs to have a background in programming, system analysis, and system design in order to understand the contents of this book. In addition, readers need to have an understanding of the server environments on which they will be using the programs.

Section One, "A Solid Foundation" (Chapters 1-4), begins with a discussion of client/server computing, TCP/IP (Transmission Control Protocol/Internet Protocol), the Internet, intranets, networks, firewalls, and security. The author takes an in-depth look at these topics, initially leaving the reader wondering whether this much detail is necessary for writing CGI programs. Ultimately, this information provides a valuable overview of the environment in which CGI programs operate. Particular attention is given to computer network security issues, which is important since poorly designed CGI programs can create a major security risk on a server.

No CGI book would be complete without an explanation of Uniform Resource Locators (URLs), Hyper Text Transfer Protocol (HTTP), Hypertext Markup Language (HTML), and other such acronyms. Chapter 2 is dedicated to sorting through these common concepts and bringing the reader up to speed on where they fit into the CGI-oriented view of the World Wide Web. One of many popular uses for CGI programs is to process data collected from forms and return relevant, current, or interactive data back to a Web browser. Chapters 3 and 4 focus on how data is passed up to the server and how the server processes that data. Topics that are covered include using the POST and GET methods, headers, server side includes (SSI), environment variables, and using cookies to simulate a stated relationship between a browser and a server for transactional activities.

Section Two, "Putting it All Together" (Chapters 5-14), contains the meat of this book--the CGI programs. One of the author's assertions near the beginning of this book was that he did not believe there was one single language that was best for writing CGI programs. It is for this reason that the author chose to showcase both Tcl and Perl. Each chapter focuses on a particular type of CGI program and contains the Tcl code and the Perl code for that program. The author, who wrote the Tcl code and most of the content of the book, chose to have Kamran Husain write the Perl code.

As a Tcl programmer by preference, I know that I wouldn't be able to do justice to the Perl code if I were to attempt to write both. I really needed someone who could write Perl, and use the features of Perl, not someone who wrote Perl that didn't make use of some of its key strengths. (p. xv)

Each chapter flows from a discussion of the design requirements for the program, to a detailed description of how the code addresses these requirements, to a full listing of the program code. The programs are designed to be general enough so that they can be customized by the reader according to their preferences. Ultimately, the programs are meant to teach the reader techniques that can be applied to writing other programs.

The following programs are included in this book: (1) a program to send email from a form on a Web browser; (2) a traditional and a not-so-traditional visitor counter program; (3) a program that redirects a browser to another location and keeps a log in order to track activity; (4) a random banner generator for

advertising purposes; (5) an events calendar; (6) a marketing program used to sell products and recruit participants; (7) an interactive company directory capable of displaying information about an individual including a picture and present availability; and (8) a systems management program used to track disk usage.

The programs are well thought out and adequately executed. The Tcl code has exceptionally well done comments. The Perl code, in comparison, does not, but it has adequate comments to be a readable program. As expected, a certain amount of tinkering is required in order to make these programs work. There are variables to change, directories and files to create, and permissions to set. The reader is made aware of most of these needed changes in both the comments contained in the code and in the discussion contained in each chapter. In many cases there are also HTML pages that need to be written. The author chose to avoid too much discussion of HTML and often left it to the reader to create the HTML forms used to interact with the programs that are contained on the CD-ROM.

Section Three, "Advanced Discussions: Server Issues" (Chapters 15-17), deals mainly with security and systems management issues related to maintaining Web servers. The discussion here, while of most interest to those who have responsibility for maintaining a Web server, is useful in understanding why a server behaves the way it does. The author discusses with some detail the various log files maintained by Web servers and the configuration files which determine how the server will perform. The section finishes with one last program titled "What's New"; a program designed to provide a customized page to each visitor based on the date of their most recent visit--reporting back to them the files that have changed since they last visited the site.

This book is a useful addition to anyone's CGI programming collection. It would not, however, be my recommendation for someone's first book about writing CGI programs or for understanding the Common Gateway Interface; there are a number of other books that cover this topic in a more focused manner. For the Tcl programmer, this book is a must-have, since I do not know of many other books, if any, that deal specifically with Tcl programming for the Common Gateway Interface.

John Wynstra (wynstra@uni.edu (mailto:wynstra@uni.edu)) is the Library Systems Project Manager at the University of Northern Iowa.

table of contents ter issues (/lita/publications/archive/ter)

REVIEW OF: Susan Maze, David Moxley, and Donna J. Smith. Neal-Schuman Authoritative Guide to Web Search Engines. New York, NY: Neal-Schuman Publishers, 1997.

by Brian K. Yost

The World Wide Web is a vast mine of information, but efficiently finding useful, relevant, and authoritative resources can be difficult. There are many attempts to remedy this situation (e.g., keyword search engines, subject directories, adapting the MARC (Machine-Readable Cataloging) format, indexing and abstracting, "push" technology, subject-specific search engines, etc.). The most popular of these by far is the keyword search engine, which is the subject of the Neal-Schuman Authoritative Guide to Web Search Engines by Susan Maze, David Moxley, and Donna J. Smith.

This is a much needed book. Each search engine has its own searching syntax, capabilities, and coverage, and it can be a daunting task to attempt to keep up with all of them. In addition, as the authors point out, the help provided by the search engine services tends to be self-promotional, vague, and sometimes inaccurate.

I found that the Neal-Schuman Authoritative Guide to Web Search Engines answered many of my questions about various search engines. For example, the use of the "pipe" command in InfoSeek--how is this different from using the Boolean "and" operator ("+" before the terms, in the case of InfoSeek)? The authors explain that the search is the same--that is, the same number of hits is retrieved-- but the ranking algorithm is altered. So while the same number of documents are retrieved, the order in which they are presented will change with the pipe command.

The book is divided into four sections. The first, a short section entitled "Essential Background," provides a brief history of the World Wide Web, an explanation of HTTP (Hypertext Transfer Protocol), overview of Web content, and an introduction to finding information on the Web. A dilemma for authors of specialized Internet books is determining how much of an introduction to provide. A comprehensive introduction to the Internet or World Wide Web can be very lengthy and tedious for more experienced readers. On the other hand, a certain basic level of knowledge is required for the user to comprehend the rest of the book. In this case, the authors handle it quite well. The section is brief and substantive enough not to bore Web veterans but still provides an essential knowledge core.

In section two, "A Model Search Tool," the authors discuss how robots discover and add documents to indexes, search engines handle user queries, and users interact with the search tools through an interface. The authors approach these processes from the perspective of a generic search engine. This big-picture approach facilitates an understanding of similarities and differences among various search engines and can be applied to search engines not covered in this text or ones not yet available. Especially interesting and useful for understanding the composition of a search engine's index is the discussion of the depth-first vs. breadth-first strategies robots use when gathering Web documents. A depth-first strategy creates a comprehensive index on a few subject areas, while a breadth-first approach creates a more diverse but more shallow database.

Section three, "Understanding and Using Seven Search Engines," is the largest section and the heart of the book. After a chapter on general searching tips, the authors review and provide instruction on seven Web search engines. The search engines covered are: WebCrawler, Lycos, InfoSeek, Open Text, AltaVista, Excite, and HotBot. Why these seven search engines? Fortunately, the authors provide explicit justification for which ones were included. In order to be included, the search engine must be popular, created by automated document gathering and indexing procedures, comprehensive in subject coverage, and free for users. After a brief history, the authors describe the process each search engine uses to add documents to its database, the search interface and capabilities, and how the search engine presents results of searches to users (e.g., relevancy rankings).

Section four is a bit of a catch-all. First, it includes a chapter on "The Economics of Web Search Tools." This chapter discusses the implications of the for-profit nature of most Web search engines and also the authors' views on the future direction of Web search tools. The second chapter of section four describes four subject directory tools: Yahoo!, Magellan, The Argus Clearinghouse, and the WWW Virtual Library. The Yahoo! description follows the format of the search engine reviews and is very informative. The sections on the other subject directories are sketchy and do little more than inform the reader of their existence and scope. Oddly, while the URLs (Uniform Resource Locators) for all the other search/index tools reviewed are provided, addresses for Argus Clearinghouse and WWW Virtual Library are missing.

In addition to these four sections, there are appendices with tables for evaluating and comparing Web search engines, a glossary, and an index.

A downside to a printed monograph on a subject as ephemeral as the World Wide Web is that it will likely be outdated within a few months. The authors recognize this and have taken several steps to combat this problem. For example, the use of a model search tool for explaining the inner workings of search engines and the general Web searching tips will be relevant for the foreseeable future. With some of the search engines, the interfaces have been tweaked, and searching options have changed from what is described in the book. The authors, however, argue that while these cosmetic aspects of search engines often change, the inner workings seldom do and most of their discussions concentrate on these more constant aspects.

Possibly because Internet searching is as much an art as a science, the authors have strong opinions on some searching techniques with which readers may or may not agree. For example, they advise avoidance of meta-tools. Normally I would agree with this advice, but a meta search can be useful for surveying what is available on a subject or when comparing how different search engines handle a common query. Also, the authors decry Alta Vista Simple Search for arguably unfair reasons: "Alta Vista's Simple queries offer no Boolean operators, only plus (+) and minus (-) symbols which provide a weak approximation of them." (p. 100) While it is accurate that plus (+) and minus (-) are not one-for-one substitutes for Boolean operators, when they are used consistently and combined with the default "or" operator, they are much better than "weak approximations."

The book is well produced and uses quality paper, attractive typefaces, and easy-to-read diagrams. There are a couple of minor typos, and it is overpriced at a list price of $49.95. While it is a high quality, well-researched book with up-to-date information, this is too much for a 178 page, 28 x 22 cm. paperback. Unfortunately, the price may discourage some information professionals and libraries from obtaining an otherwise excellent book.

Overall, the book does what the authors intended very well. It will serve as a primer for effective Web searching and as a useful reference. I recommend the Neal-Schuman Authoritative Guide to Web Search Engines for all information professionals involved with any Web searching at all levels of expertise.

Brian K. Yost (yostb@hope.edu (mailto:yostb@hope.edu)) is Technical Services/Electronic Resources Librarian at Hope College in Holland, Michigan.

table of contents ter issues (/lita/publications/archive/ter)

About TER