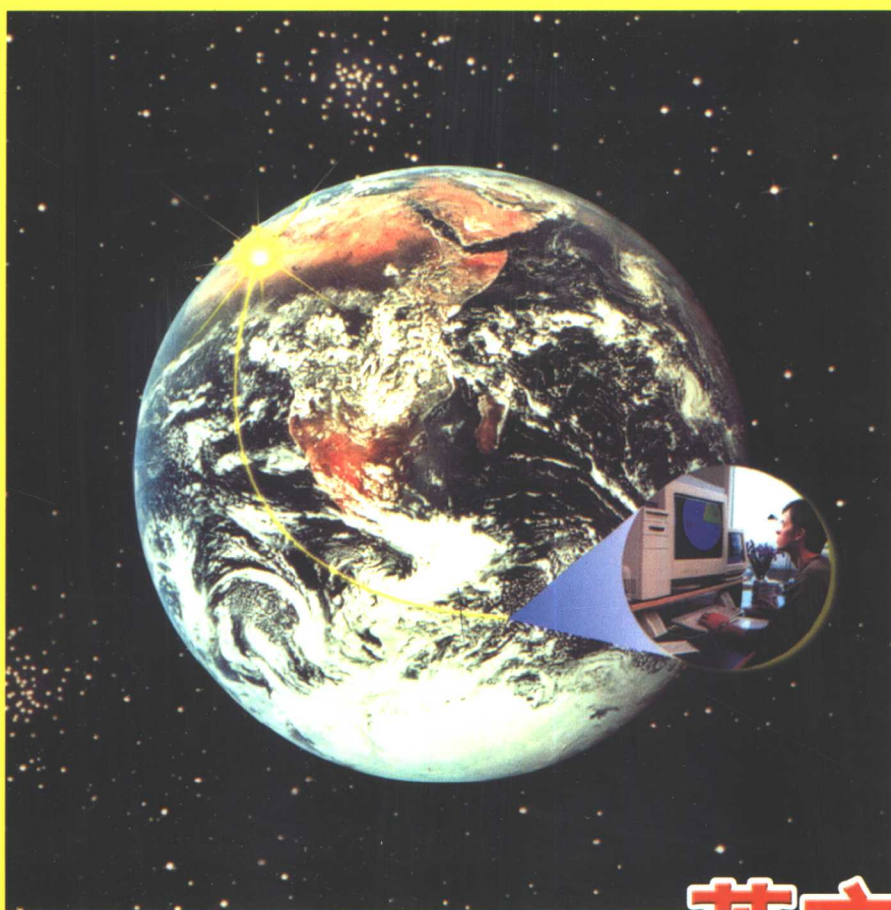


国外著名高等院校
信息科学与技术优秀教材

TCP/IP 网络互连 第3卷： 客户 / 服务器编程及应用 Linux/POSIX Sockets 版

INTERNETWORKING with **TCP/IP** Volume 3
CLIENT-SERVER PROGRAMMING AND APPLICATIONS
LINUX/POSIX Sockets Version



DOUGLAS E. COMER

DAVID L. STEVENS

英文版

人民邮电出版社
www.pptph.com.cn

PRENTICE HALL Pearson Education
出版集团


国外著名高等院校信息科学与技术优秀教材

TCP/IP 网络互连 第3卷:
客户/服务器编程及应用 Linux/POSIX Sockets 版
(英文版)

Internetworking with TCP/IP Volume 3
Client-Server Programming and Applications
Linux/POSIX Sockets Version

Douglas E. Comer

David L. Stevens

 人民邮电出版社

PRENTICE
HALL **Pearson Education** 出版集团

国外著名高等院校信息科学与技术优秀教材
TCP/IP 网络互连 第3卷:
客户/服务器编程及应用 Linux/POSIX Sockets 版
(英文版)

◆ 著 Douglas E. Comer David L. Stevens
责任编辑 陈 昇

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@pptph.com.cn
网址 <http://www.pptph.com.cn>
读者热线 010-67129212 010-67129211(传真)
北京汉魂图文设计有限公司制作
北京朝阳展望印刷厂印刷
新华书店总店北京发行所经销

◆ 开本:800×1000 1/16
印张:39.5
字数:766千字 2002年1月第1版
印数:1-3000册 2002年1月北京第1次印刷

著作权合同登记 图字:01-2001-4837号

ISBN 7-115-09921-9/TP·2654

定价:57.00元

本书如有印装质量问题,请与本社联系 电话:(010)67129223

图书在版编目 (CIP) 数据

TCP/IP 网络互连. 第 3 卷, 客户/服务器编程及应用: Linux/POSIX Sockets 版/(美) 科默 (Comer, D. E.), (美) 史蒂文斯 (Stevens, D. L.) 著. —北京: 人民邮电出版社, 2002.1

国外著名高等院校信息科学与技术优秀教材

ISBN 7-115-09921-9

I. T... II. ①科... ②史... III. 计算机网络—通信协议—高等学校—教材—英文

IV. TN915.04

中国版本图书馆 CIP 数据核字 (2001) 第 089015 号

版 权 声 明

English Reprint Edition Copyright © 2001 by PEARSON EDUCATION NORTH ASIA LIMITED and PEOPLE'S POSTS & TELECOMMUNICATIONS PUBLISHING HOUSE.

Internetworking with TCP/IP Volume 3 Client-Server Programming and Applications Linux/POSIX Sockets Version: Douglas E. Comer David L. Stevens

Copyright © 2001

All Rights Reserved.

Published by arrangement with Prentice Hall, Pearson Education, Inc.

This edition is authorized for sale only in People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macau).

著作权合同登记 图字: 01-2001-4837 号

2002/03

内 容 提 要

本书讨论了客户/服务器编程和应用，讲述了构筑所有分布式计算系统的客户/服务器计算模型的基本概念，内容包括各种不同的服务器设计方法以及用来构造客户/服务器的各种工具和技术，包括远程调用 RPC。书中包括了用来说明各种设计和工具的运行程序示例的源代码。这本书是基于 Linux/POSIX Sockets 版本编写的，内容结构合理，易于阅读，是一本关于 TCP/IP 网络互连的既经典又可读性极强的书，是任何一个想要了解网络互连技术的人所必不可少的参考书。

本书适合作为高等院校计算机专业网络相关课程的教材，也适合各类网络技术开发人员阅读。

出版说明

2001年，教育部印发了《关于“十五”期间普通高等教育教材建设与改革的意见》。该文件明确指出，“九五”期间原国家教委在“抓好重点教材，全面提高质量”方针指导下，调动了各方面的积极性，产生了一大批具有改革特色的新教材。然而随着科学技术的飞速发展，目前高校教材建设工作仍滞后于教学改革的实践，一些教材内容陈旧，不能满足按新的专业目录修订的教学计划和课程设置的需要。为此该文件明确强调，要加强国外教材的引进工作。当前，引进的重点是信息科学与技术 and 生物科学与技术两大学科的教材。要根据专业（课程）建设的需要，通过深入调查、专家论证，引进国外优秀教材。要注意引进教材的系统配套，加强对引进教材的宣传，促进引进教材的使用和推广。

邓小平同志早在1977年就明确指出：“要引进外国教材，吸收外国教材中有益的东西。”随着我国加入WTO，信息产业的国际竞争将日趋激烈，我们必须尽快培养出大批具有国际竞争能力的高水平信息技术人才。教材是一个很关键的问题，国外的一些优秀教材不但内容新，而且还提供了很多新的研究方法和思考方式。引进国外原版教材，可以促进我国教学水平的提高，提高学生的英语水平和学习能力，保证我们培养出的学生具有国际水准。

为了贯彻中央“科教兴国”的方针，配合国内高等教育教材建设的需要，人民邮电出版社约请有关专家反复论证，与国外知名的教材出版公司合作，陆续引进一些信息科学与技术优秀教材。第一批教材针对计算机专业的主干核心课程，是国外著名高等院校所采用的教材，教材的作者都是在相关领域享有盛名的专家教授。这些教材内容新，反映了计算机科学技术的最新发展，对全面提高我国信息科学与技术的教学水平必将起到巨大的推动作用。

出版国外著名高等院校信息科学与技术优秀教材的工作将是一个长期的、坚持不懈的过程，我社网站（www.pptph.com.cn）上介绍了我们首批陆续推出的图书的详细情况，后续教材的引进和出版情况我们会及时在网上发布，敬请关注。希望广大教师和学生将使用中的意见和建议及时反馈给我们，我们将根据您的反馈不断改进我们的工作，推出更多更好的引进版信息科学与技术教材。

人民邮电出版社

2001年12月

序 言

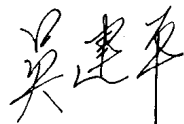
Douglas E. Comer 教授的《Internetworking with TCP/IP》是解读 TCP/IP 的经典教科书，在国内外都享有很高声誉，这已经是第四版。相信阅读过前三版的读者一定会被该书内容的丰富、原理的清晰、概念的准确、语言的明快而深深打动，这也是我非常愿意向大家推荐该书的主要原因。

计算机网络技术是当今信息领域发展最快的技术之一，在经历了实践的检验和磨砺后，当初名不见经传的 TCP/IP 以其简洁、高效、开放的优点从众多网络体系结构中脱颖而出，成为互联网络的标准。TCP/IP 的强大生命力是其成功的关键因素。

互联网络已将全世界连接在一起，并仍在以极快的速度深刻地变革着人们的生活、学习和工作方式。人们对互联网络的巨大需求更进一步促进了下一代互联网络的研究和开发，在这方面，第四版增加了许多相关内容，如：IPV6、RTP、Mobile IP 等。

培根说：“书籍是人类进步的阶梯”，下一代互联网络正在向我们阔步走来，Comer 教授的这本书是我们理解现在、迈向未来的很好的阶梯。

清华大学计算机科学与技术系教授
中国教育和科研网网络中心主任



2001 年 11 月 18 日

Foreword

It is a singular honor to introduce open source readers to the third volume of Dr. Douglas E. Comer's remarkable series: *Internetworking with TCP/IP*.

The history of open source and TCP/IP are magnificently intertwined: you can't have collaboration without a network to connect the collaborators! Further, some of the very first open source software were implementations of the TCP/IP protocols. I remember the early 80's, long before "open source" was today's media darling. In those days, there were a handful of researchers who understood the problems of network architectures and implementations. Doug was among their leaders — the principal of an extensive research program and waging a multi-front attack on the problems we faced.

I remember the early 90's, when we first began to see the push toward moving the technology to the large engineering communities that were hungering for knowledge and solutions. In those days, it was a mighty struggle for those engineers to build internet-based environments for their corporations. Doug was there to educate and inform them — making the underlying complexity accessible to them and providing them with hard-earned insights.

And now, I see the early 00's, where a new generation of designers are writing distributed applications for the Internet. In these days, we hear of many exciting Internet applications, such as napster, gnutella, and infrasearch. Surprisingly, few of today's developers have a grasp of solid network engineering principles — bluntly, they lack an understanding of the basics, and this lack of understanding inevitably leads to applications that don't scale well or that just plain do not work.

It is for this reason, that *Volume 3, Client-Server Programming and Applications*, which Doug has authored with David L. Stevens, is particularly relevant to the Internet today. It teaches us how to architect and build client-server applications, and — more importantly — how to understand what trade-offs are involved with each design decision.

My hope is that you, the reader, can benefit from Dr. Comer's wisdom as much as your humble predecessors.

Marshall T. Rose
Theorist, Implementor, and Agent Provocateur
Petaluma, California
June, 2000

Preface

The Linux operating system is soaring in popularity, and especially important in the networking community as the system for many servers. This new version of Volume 3, which uses Linux, is aimed at programmers who want to understand how to create networking applications. Broadly speaking, it examines the question, “How does application software use TCP/IP protocols to communicate across an internet?” The text focuses on the client-server paradigm, and examines algorithms for both the client and server components of a distributed program. It shows an implementation that illustrates each design, and discusses techniques including application-level gateways and tunneling. In addition, it reviews several standard application protocols, and uses them to illustrate the algorithms and implementation techniques.

Although this volume can be read and used alone, it forms part of a larger picture completed by two other volumes in the series. Volume 1 considers the question “What is a TCP/IP internet?” Volume 2 examines the question, “How does TCP/IP software work?” It presents more details, examines working code, and explores greater depth than the first volume. Thus, although a programmer can learn to create network applications from Volume 3 alone, the other volumes can be used to provide a better understanding of the underlying technologies.

This version of Volume 3 includes the latest technologies. For example, a chapter explains how a Linux program can use the POSIX thread facilities to create a concurrent server. The chapter on NFS discusses version 3, the version that is about to emerge in the Linux community. In addition, sections have been included to explain concepts behind programs like *slirp* that provide Internet access over a dialup telephone connection without requiring each computer to have a unique IP address.

Two chapters that stand out as especially timely concentrate on *streaming* and the associated technologies used to send audio and video across the Internet. Chapter 28 describes fundamental concepts such as the Real-time Transport Protocol (RTP), encoding, and jitter buffers. Chapter 29 shows an implementation of RTP that is used to receive and play MP3 audio.

The code for all examples in the text is available online. To access a copy via the Web, look for Volume 3 in the list of networking books at location:

<http://www.cs.purdue.edu/homes/comer/netbooks.html>

To access the code via FTP, use location:

`ftp://ftp.cs.purdue.edu/pub/Xinu/TCPIP-vol3.linux.dist.tar.Z`

The text is organized as follows. Beginning chapters introduce the client-server paradigm and the socket interface that application programs use to access TCP/IP protocol software. They also describe concurrent processes and the operating system functions used to create them. Chapters that follow the introductory material discuss client and server designs.

The text explains that the myriad of possible designs are not random. Instead, they follow a pattern that can be understood by considering the choice of concurrency and transport. For example, one chapter discusses a nonconcurrent server design that uses connection-oriented transport (e.g., TCP), while another discusses a similar design that uses connectionless transport (e.g., UDP).

We describe how each design fits into the space of possible implementations, but do not try to develop an abstract “theory” of client-server interactions. Instead, we emphasize practical design principles and techniques that are important to programmers. Each technique has advantages in some circumstances, and each has been used in working software. We believe that understanding the conceptual ties among the designs will help the reader appreciate the strengths and weaknesses of each approach, and will make it easier to choose among them.

The text contains example programs that show how each design operates in practice. Most of the examples implement standard TCP/IP application protocols. In each case, we tried to select an application protocol that would convey a single design idea without being too complex to understand. Thus, while few of the example programs are exciting, they each illustrate one important concept. This version of Volume 3 uses the Linux socket mechanism (i.e., socket API) in all programming examples; two other versions of the text contain many of the same examples using Microsoft’s Windows Sockets interface and AT&T’s TLI interface.

Later chapters focus on middleware. They discuss the remote procedure call concept and describe how it can be used to construct distributed programs. The chapters relate the remote procedure call technique to the client-server model, and show how software can be used to generate client and server programs from a remote procedure call description. The chapters on TELNET show how small details dominate a production program and how complex the code can become for even a simple, character-oriented protocol. The section ends with the two chapters on streaming transport described earlier.

Much of the text concentrates on concurrent processing. Many of the concepts described may seem familiar to students who have written concurrent programs because they apply to all concurrent programs, not only network applications. Students who have not written concurrent programs may find the concepts difficult.

The text is suitable for a single semester undergraduate course on “socket programming” or a beginning graduate-level course on distributed computing. Because the text concentrates on how to use an internet rather than on how it works, students need little background in networking to understand the material. No particular concept is too difficult for an undergraduate course as long as the instructor proceeds at a suitable pace. A basic course in operating systems concepts or experience with concurrent programming may provide the best background.

Students will not appreciate the material until they use it first hand. Thus, any course should have programming exercises that force the students to apply the ideas to practical programs. Undergraduates can learn the basics by repeating the designs on other application protocols. Graduate students should build more complex distributed programs that emphasize some of the subtle techniques (e.g., the concurrency management techniques in Chapter 16 and the interconnection techniques in Chapters 18 and 19).

We thank many people for their help. Members of the Internet Research Group at Purdue contributed technical information and suggestions to the original text. Michael Evangelista proofread the text and wrote the RTP code. Gustavo Rodriguez-Rivera read several chapters, ran experiments to test details, and edited Appendix 1. Dennis Brylow commented on several chapters. Christine Comer edited the entire text and improved both wording and consistency.

Douglas E. Comer

David L. Stevens

July, 2000

About The Authors

Dr. Douglas Comer is an internationally recognized expert on TCP/IP protocols and the Internet. One of the researchers who contributed to the Internet as it was being formed in the late 1970s and 1980s, he was a member of the Internet Architecture Board, the group responsible for guiding the Internet's development. He was also chairman of the CSNET technical committee and a member of the CSNET executive committee.

Comer consults for companies on the design and implementation of networks, and gives professional seminars on TCP/IP and internetworking to both technical and nontechnical audiences around the world. His operating system, Xinu, and implementation of TCP/IP protocols are documented in his books, and used in commercial products.

Comer is a professor of computer science at Purdue University, where he teaches courses and does research on computer networking, internetworking, and operating systems. In addition to writing a series of best-selling technical books, he serves as the North American editor of the journal *Software — Practice and Experience*. Comer is a Fellow of the ACM.

Additional information can be found at:

www.cs.purdue.edu/people/comer

David Stevens received his BS (1985) and MS (1993) in Computer Science from Purdue University. He has been a UNIX systems programmer working primarily on BSD UNIX kernels since 1983. He has done implementations of most of the Internet Protocol Suite and co-authored several Computer Science textbooks with Dr. Comer. His areas of professional interest are operating systems, computer networking, and large-scale software systems design.

In recent years, Stevens has worked in the area of scalable networking on high-performance multiprocessor systems for Sequent Computer Systems and the IBM Corporation. He is a member of the ACM and IEEE.

What Others Have Said About The Linux Version Of Internetworking With TCP/IP Volume 3

“This is by far the best book on the topic I have *ever* read. Thank you for making sockets easy to understand.”

*Dustin Boswell
Caltech*

“An excellent book for learning TCP/IP client-server programming. This book explains important concepts clearly and provides working example code; the combination produces an extremely effective way to learn the subject.”

*John Lin
Bell Labs*

“Your book has been extremely valuable to me — thank you very much indeed.”

Jacoby Thwaites

“I enjoy the clarity and depth!”

Rob Moloney

“Volume 3, Client-Server Programming and Applications, which Doug has authored with David L. Stevens, is particularly relevant to the Internet today. It teaches us how to architect and build client-server applications, and — more importantly — how to understand what trade-offs are involved with each design decision.”

Marshall Rose

Contents

Chapter 1 Introduction And Overview 1

- 1.1 *Internet Applications Using TCP/IP* 1
- 1.2 *Designing Applications For A Distributed Environment* 1
- 1.3 *Standard And Nonstandard Application Protocols* 2
- 1.4 *An Example Of Standard Application Protocol Use* 2
- 1.5 *An Example TELNET Connection* 3
- 1.6 *Using TELNET To Access An Alternative Service* 4
- 1.7 *Application Protocols And Software Flexibility* 5
- 1.8 *Viewing Services From The Provider's Perspective* 6
- 1.9 *The Remainder Of This Text* 6
- 1.10 *Summary* 7

Chapter 2 The Client Server Model And Software Design 9

- 2.1 *Introduction* 9
- 2.2 *Motivation* 10
- 2.3 *Terminology And Concepts* 10
 - 2.3.1 *Clients And Servers* 11
 - 2.3.2 *Privilege And Complexity* 11
 - 2.3.3 *Standard Vs. Nonstandard Client Software* 12
 - 2.3.4 *Parameterization Of Clients* 12
 - 2.3.5 *Connectionless Vs. Connection-Oriented Servers* 13
 - 2.3.6 *Stateless Vs. Stateful Servers* 14

2.3.7	<i>A Stateless File Server Example</i>	15
2.3.8	<i>A Stateful File Server Example</i>	15
2.3.9	<i>Identifying A Client</i>	16
2.3.10	<i>Statelessness Is A Protocol Issue</i>	18
2.3.11	<i>Servers As Clients</i>	19
2.4	<i>Summary</i>	20

Chapter 3 Concurrent Processing In Client-Server Software

23

3.1	<i>Introduction</i>	23
3.2	<i>Concurrency In Networks</i>	23
3.3	<i>Concurrency In Servers</i>	25
3.4	<i>Terminology And Concepts</i>	26
3.4.1	<i>The Process Concept</i>	26
3.4.2	<i>Sharing Of Local And Global Variables</i>	27
3.4.3	<i>Procedure Calls</i>	28
3.5	<i>An Example Of Concurrent Process Creation</i>	29
3.5.1	<i>A Sequential C Example</i>	29
3.5.2	<i>A Concurrent Version</i>	30
3.5.3	<i>Timeslicing</i>	31
3.5.4	<i>Singly-Threaded Process Assumption</i>	32
3.5.5	<i>Making Processes Diverge</i>	33
3.6	<i>Executing New Code</i>	34
3.7	<i>Context Switching And Protocol Software Design</i>	34
3.8	<i>Concurrency And Asynchronous I/O</i>	35
3.9	<i>Summary</i>	36

Chapter 4 Application Interface To Protocols

39

4.1	<i>Introduction</i>	39
4.2	<i>Loosely Specified Protocol Software Interface</i>	39
4.2.1	<i>Advantages And Disadvantages</i>	40
4.3	<i>Interface Functionality</i>	40
4.4	<i>Conceptual Interface Specification</i>	41
4.5	<i>System Calls</i>	42
4.6	<i>Two Basic Approaches To Network Communication</i>	43
4.7	<i>The Basic I/O Functions Available In Linux</i>	43
4.8	<i>Using Linux I/O With TCP/IP</i>	45
4.9	<i>Summary</i>	45

Chapter 5 The Socket API**47**

- 5.1 *Introduction* 47
- 5.2 *Berkeley Sockets* 47
- 5.3 *Specifying A Protocol Interface* 48
- 5.4 *The Socket Abstraction* 49
 - 5.4.1 *Socket Descriptors And File Descriptors* 49
 - 5.4.2 *System Data Structures For Sockets* 50
 - 5.4.3 *Making A Socket Active Or Passive* 51
- 5.5 *Specifying An Endpoint Address* 52
- 5.6 *A Generic Address Structure* 52
- 5.7 *Major System Calls In The Socket API* 54
 - 5.7.1 *The Socket Call* 54
 - 5.7.2 *The Connect Call* 54
 - 5.7.3 *The Send Call* 55
 - 5.7.4 *The Recv Call* 55
 - 5.7.5 *The Close Call* 55
 - 5.7.6 *The Bind Call* 56
 - 5.7.7 *The Listen Call* 56
 - 5.7.8 *The Accept Call* 56
 - 5.7.9 *Using Read And Write With Sockets* 56
 - 5.7.10 *Summary Of Socket Calls* 57
- 5.8 *Utility Routines For Integer Conversion* 58
- 5.9 *Using Socket Calls In A Program* 58
- 5.10 *Symbolic Constants For Socket Call Parameters* 59
- 5.11 *Summary* 60

Chapter 6 Algorithms And Issues In Client Software Design**63**

- 6.1 *Introduction* 63
- 6.2 *Learning Algorithms Instead Of Details* 63
- 6.3 *Client Architecture* 64
- 6.4 *Identifying The Location Of A Server* 64
- 6.5 *Parsing An Address Argument* 66
- 6.6 *Looking Up A Domain Name* 67
- 6.7 *Looking Up A Well-Known Port By Name* 68
- 6.8 *Port Numbers And Network Byte Order* 68
- 6.9 *Looking Up A Protocol By Name* 69
- 6.10 *The TCP Client Algorithm* 69
- 6.11 *Allocating A Socket* 70
- 6.12 *Choosing A Local Protocol Port Number* 71
- 6.13 *A Fundamental Problem In Choosing A Local IP Address* 71
- 6.14 *Connecting A TCP Socket To A Server* 72

6.15	<i>Communicating With The Server Using TCP</i>	72
6.16	<i>Receiving A Response From A TCP Connection</i>	73
6.17	<i>Closing A TCP Connection</i>	74
6.17.1	<i>The Need For Partial Close</i>	74
6.17.2	<i>A Partial Close Operation</i>	74
6.18	<i>Programming A UDP Client</i>	75
6.19	<i>Connected And Unconnected UDP Sockets</i>	76
6.20	<i>Using Connect With UDP</i>	76
6.21	<i>Communicating With A Server Using UDP</i>	76
6.22	<i>Closing A Socket That Uses UDP</i>	77
6.23	<i>Partial Close For UDP</i>	77
6.24	<i>A Warning About UDP Unreliability</i>	77
6.25	<i>Summary</i>	77

Chapter 7 Example Client Software

81

7.1	<i>Introduction</i>	81
7.2	<i>The Importance Of Small Examples</i>	81
7.3	<i>Hiding Details</i>	82
7.4	<i>An Example Procedure Library For Client Programs</i>	82
7.5	<i>Implementation Of ConnectTCP</i>	83
7.6	<i>Implementation Of ConnectUDP</i>	84
7.7	<i>A Procedure That Forms Connections</i>	85
7.8	<i>Using The Example Library</i>	88
7.9	<i>The DAYTIME Service</i>	88
7.10	<i>Implementation Of A TCP Client For DAYTIME</i>	89
7.11	<i>Reading From A TCP Connection</i>	90
7.12	<i>The TIME Service</i>	91
7.13	<i>Accessing The TIME Service</i>	91
7.14	<i>Accurate Times And Network Delays</i>	92
7.15	<i>A UDP Client For The TIME Service</i>	92
7.16	<i>The ECHO Service</i>	94
7.17	<i>A TCP Client For The ECHO Service</i>	94
7.18	<i>A UDP Client For The ECHO Service</i>	96
7.19	<i>Summary</i>	98

Chapter 8 Algorithms And Issues In Server Software Design

101

8.1	<i>Introduction</i>	101
8.2	<i>The Conceptual Server Algorithm</i>	101
8.3	<i>Concurrent Vs. Iterative Servers</i>	102
8.4	<i>Connection-Oriented Vs. Connectionless Access</i>	102