COMPUTER NETWORKS

Prentice/Hall International editions

ANDREW S. TANENBAUM

COMPUTER NETWORKS

ANDREW S. TANENBAUM

Vrije Universiteit
Amsterdam, The Netherlands

PHI

Prentice/Hall International, Inc.

Library of Congress Cataloging in Publication Data

Tanenbaum, Andrew S. (date)
Computer networks.

(Prentice-Hall software series)
Bibliography: p.
Includes index.
1. Computer networks. I. Title II. Series.
TK5105.5.T36 001.64'404 80-23283
ISBN 0-13-164699-0

PRENTICE-HALL SOFTWARE SERIES
Brian W. Kernighan, advisor

10 9 8 7 6 5 4 3

Printed in the United States of America

Editorial/production supervision: Nancy Milnamow
Manufacturing buyer: Joyce Levatino

PRENTICE-HALL INTERNATIONAL, INC., *London*
PRENTICE-HALL OF AUSTRALIA PTY. LTD., *Sydney*
EDITORA PRENTICE-HALL DO BRASIL, LTDA, *Rio de Janeiro*
PRENTICE-HALL OF CANADA, LTD., *Toronto*
PRENTICE-HALL OF INDIA PRIVATE LTD., *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
PRENTICE-HALL OF SOUTHEAST ASIA PTE. LTD., *Singapore*
WHITEHALL BOOKS LTD., *Wellington, New Zealand*
PRENTICE-HALL, INC., *Englewood Cliffs, New Jersey*

COMPUTER NETWORKS

To Suzanne, Barbara, Marvin, and Sweetie π

PREFACE

As computers have become smaller, cheaper, and more numerous, people have become more and more interested in connecting them together to form networks and distributed systems. At first these connections were made in an ad hoc way, typically with each computer regarding the other ones as terminals. However, in the past decade, a substantial body of knowledge has developed on the subject of computer networking, so that future networks and distributed systems can be designed systematically.

The key to designing a computer network was first enunciated by Julius Caesar: Divide and Conquer. The idea is to design the network as a sequence of layers, or abstract machines, each one based upon the previous one. By reducing the study of the whole to the study of its parts, the subject becomes more manageable. This book uses a model in which networks are divided into seven layers. The structure of the book follows the structure of the model to a considerable extent.

Chapter 1 provides an introduction to the subject of computer networks in general and layered protocols in particular. Chapter 2 treats a number of algorithms and heuristics for designing the topology of a network. Chapter 3 begins the study of the seven-layer model with the bottom layer (the physical layer); it covers the architecture of data communication systems, including the telephone system and satellites. Chapter 4 is primarily concerned with data link layer protocols, algorithms for reliably transmitting data over unreliable lines. Chapters 5, 6, and 7 all deal with the network layer; Chap. 5 treats point-to-point networks, Chap. 6 treats satellite and packet radio networks, and Chap. 7 treats local networks. Chapter 8 studies the transport and session layers, in particular, end-to-end protocols and internetworking. Chapter 9 is about the presentation layer, including cryptography, text compression, virtual terminal protocols, and file transfer protocols. Chapter 10 provides an introduction to some application layer issues, primarily distributed data bases and distributed operating systems. Chapter 11 contains a reading list and bibliography.

The book is intended as a text for juniors, seniors, and graduate students in computer science, electrical engineering, and related disciplines. The only prerequisites are a general familiarity with computer systems and programming, although a little knowledge of elementary calculus and elementary probability theory is useful, but not essential. Since the amount of material in the book may be too much for a one semester course, depending on the level of the students, I have made a serious attempt to make each chapter relatively independent of the other ones. This way an instructor is free to skip whichever chapters he chooses. Chapter 2, in particular, which is long and highly technical, may be omitted without loss of continuity by students who are not concerned with the topology design problem. However, if Chap. 2 is omitted, I would recommend having the students look at Eq. (2-9) and Eq (2-10), since they are used in a few places in subsequent chapters to analyze network performance.

People who have no formal computer science background, but who have had some industrial experience with minicomputers, assembly languages, system programming languages, operating systems, or data communication should have little trouble understanding the book. Even programmers or managers unfamiliar with these areas can probably understand a substantial part of the book.*

Many people have helped me with this book. I would first like to thank Wiebren de Jonge for tens, if not hundreds, of hours of questions, discussions, arguments, and polemics about nearly every page of the manuscript. His impatience with my occasional indolence has shamed me into rewriting sections over and over until I finally got them right.

I would also like to thank Peter Apers, Dick Binder, Imrich Chlamtac, Dave Clark, George Conant, Ira Cotton, Rudy Cypser, Yogen Dalal, Dixon Doll, Phil Enslow, Howard Frank, Bill Franta, James P. Gray, Paul Green, Jan Hajek, Doug Jensen, Steve Johnson, Haim Kilov, Leonard Kleinrock, Simon Lam, Tony Lauck, Mike Liu, Alex McKenzie, Dave Morgan, Bob Morris, Holger Opderbeck, Robert Ryan, Paul Santos, Phil and Debbie Scherrer, Mischa Schwartz, John Shoch, Johan Stevenson, Carl Sunshine, Jim van Keulen, Stu Wecker, Barry Wessler, and Sylvia Wilbur for their help. Their collective assistance has greatly improved the book in many ways. My students, especially Dick Biekart, Herman Gerbscheid, and Jan de Ruiter have also been very helpful.

To all the members of the Computing Science Research Center at Bell

*A manual with the problem solutions is available from Prentice-Hall.

Laboratories, thanks for many stimulating discussions during my visit there in the summer of 1979. I would especially like to thank Sandy Fraser for arranging things, Brian Kernighan and Mike Lesk for initiating me into the mysteries of the CAT, and Lorinda Cherry for causing a certain large minicomputer to comment extensively about my diction.

At this point authors normally thank their typists. Not wanting to break with this worthy tradition, I would like to thank Andy Tanenbaum for his fast and accurate typing of innumerable versions of the manuscript. Somewhat more to the point, I would like to thank Ken Thompson, Dennis Ritchie, and the other members of the Computing Science Research Center at Bell Laboratories for developing the UNIX† operating system. The UNIX text processing tools made the typing and retyping of the 1,231,788 characters comprising the final manuscript a pleasure instead of a chore.

IBM and DEC have permitted me to use material from their SNA and DECNET protocol manuals, respectively, for which I am grateful.

Last but not least, I would like to thank Suzanne, Barbara, Marvin, and Sweetie π, Suzanne for her patience, especially since she knew what she was getting in for this time, Barbara for keeping my terminal free of peanut butter, Marvin for arriving at a propitious moment, and Sweetie π for being very quiet while I was writing.

ANDREW S. TANENBAUM

COMPUTER NETWORKS

CONTENTS

1

INTRODUCTION

Each of the past three centuries has been dominated by a single technology. The eighteenth century was the time of the great mechanical systems accompanying the industrial revolution. The nineteenth century was the age of the steam engine. During the twentieth century, the key technology has been information gathering, processing, and distribution. Among other developments we have seen the installation of worldwide telephone networks, the invention of radio and television, the birth and unprecedented growth of the computer industry, and the launching of communication satellites.

As we move toward the final years of this century, these areas are rapidly converging, and the differences between collecting, transporting, storing, and processing information are quickly disappearing. Organizations with hundreds of offices spread over a wide geographical area routinely expect to be able to inspect the current status of even their most remote outpost at the push of a button. As our capability to gather, process, and distribute information grows, the demand for even more sophisticated information processing grows even faster.

Although the computer industry is young compared to other industries (e.g., automobiles and air transportation), computers have made spectacular progress in a short time. During the first two decades of their existence, computer systems were highly localized, usually within a single large room. Not infrequently, this room had glass walls, through which visitors could gawk at the great electronic wonder inside. A medium size company or university might have had one or two computers, with large ones having had at most a few dozen. The idea that within 20 years equally powerful computers smaller than postage stamps would be mass produced by the millions was pure science fiction.

The merging of computers and communications has had a profound influence on the way computer systems are organized. The concept of the "computer center" as a room with a large computer to which users bring their work for processing is rapidly becoming obsolete. This model has not one, but two flaws in it: the concept of a single large computer doing all the work, and the idea of users bringing work to the computer, instead of bringing the computer to the users.

The old model a single computer serving all of the organization's computational needs is rapidly being replaced by one in which a large number of separate but interconnected computers do the job. These systems are called **computer networks**. The design and analysis of these networks is the subject of this book.

Throughout the book we will use the term "computer network" to mean an *interconnected* collection of *autonomous* computers. Two computers are said to be interconnected if they are capable of exchanging information. The connection need not be via a copper wire; lasers, microwaves, and earth satellites can also be used. By requiring that the computers be autonomous, we wish to exclude from our definition systems in which there is a clear master/slave relation. If one computer can forcibly start, stop, or control another one, the computers are not autonomous. An ILLIAC IV type of system, with one control unit and many slaves, is not a network. Nor is a large computer with remote card readers, printers, and terminals a network.

There is considerable confusion in the literature between a computer network and a **distributed system**. Enslow's (1978) definition requires a distributed system to have a *system-wide* operating system, with services requested by name, and not by location. In other words, the user of a distributed system should not be aware that there are multiple processors; it should look like a virtual uniprocessor. Allocation of jobs to processors, processor scheduling, allocation of files to disks, movement of files between where they are stored and where they are needed, and all other system function must be automatic.

On the other hand, Liebowitz and Carson (1978) have said: "A distributed system is one in which the computing functions are dispersed among several physical computing elements." Obviously this definition includes many systems that Enslow's excludes.

In our view a distributed system is a special case of a network, one with a high degree of cohesiveness and transparency. In essence a network may or may not be a distributed system, depending on how it is used. However, in view of the lack of any generally accepted nomenclature, we will use the term "computer network" in a generic sense, to cover both computer networks and distributed systems.