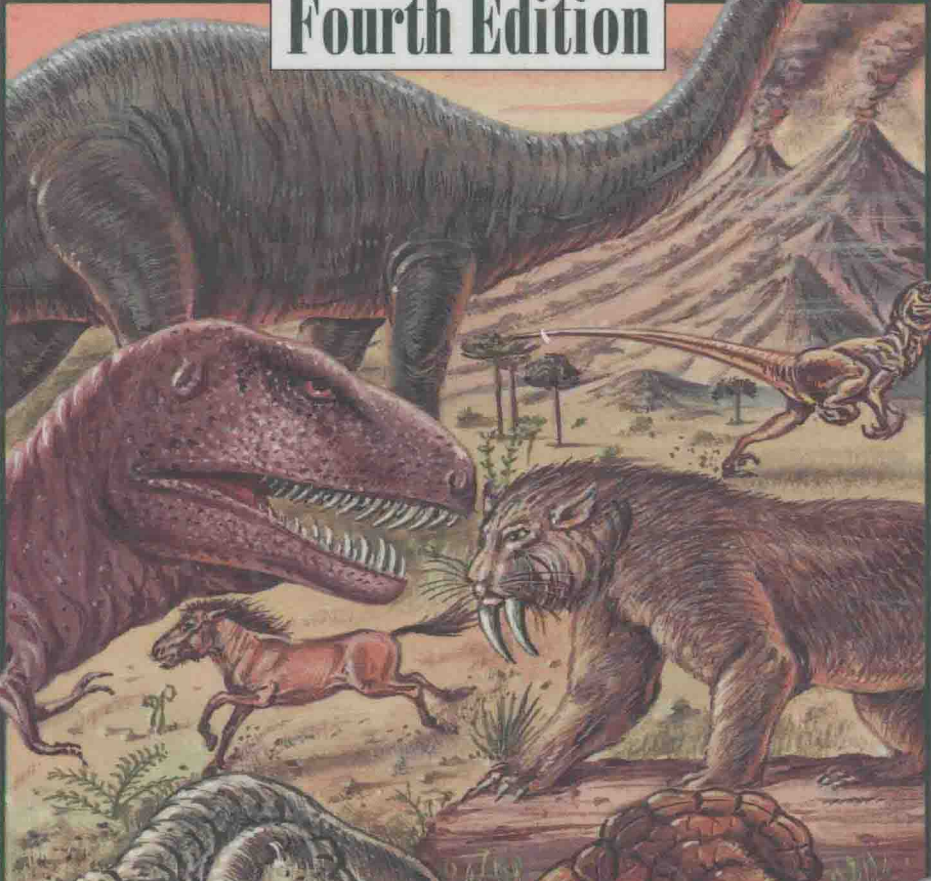


OPERATING SYSTEM CONCEPTS

Fourth Edition



SILBERSCHATZ
GALVIN

H&F

FOURTH EDITION

OPERATING SYSTEM CONCEPTS



Abraham Silberschatz

University of Texas

Peter B. Galvin

Brown University

◆ Addison-Wesley Publishing Company

Reading, Massachusetts • Menlo Park, California • New York
Don Mills, Ontario • Wokingham, England • Amsterdam • Bonn
Sydney • Singapore • Tokyo • Madrid • San Juan • Milan • Paris

Sponsoring Editor: *Deborah Lafferty*
Senior Editor: *Tom Stone*
Senior Production Supervisor: *Helen Wythe*
Marketing Manager: *Phyllis Cerys*
Technical Art Coordinator: *Susan London-Payne*
Cover and Endpaper Designer: *Howard S. Friedman*
Manufacturing Manager: *Roy Logan*

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Addison-Wesley was aware of a trademark claim, the designations have been printed in initial caps or all caps.

The procedures and applications presented in this book have been included for their instructional value. They have been tested with care but are not guaranteed for any particular purpose. The publisher does not offer any warranties or representations, nor does it accept any liabilities with respect to the programs or applications.

Library of Congress Cataloging-in-Publication Data

Silberschatz, Abraham.

Operating system concepts / Abraham Silberschatz, Peter B. Galvin.

p. cm.

Includes bibliographical references and index.

ISBN 0-201-50480-4

1. Operating systems (Computers) I. Galvin, Peter B. II. Title.

QA76.76.063S5583 1994

005.4'3--dc20

93-24415

CIP

Reproduced by Addison-Wesley from camera-ready copy supplied by the authors.

Reprinted with corrections February, 1994

Copyright © 1994 by Addison-Wesley Publishing Company, Inc.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. Printed in the United States of America.

ISBN 0-201-50480-4

2 3 4 5 6 7 8 9 10-MA-9594

*To my parents, Wira and Mietek,
my wife, Haya,
and my children, Lemor, Sivan and Aaron.*

Avi Silberschatz

To Carla and Gwendolyn.

Peter Galvin

PREFACE

Operating systems are an essential part of a computer system. Similarly, a course on operating systems is an essential part of a computer-science education. This book is intended as a text for an introductory course in operating systems at the junior or senior undergraduate level, or first-year graduate level. It provides a clear description of the *concepts* that underlie operating systems.

This book does not concentrate on any particular operating system or hardware. Instead, it discusses fundamental concepts that are applicable to a variety of systems. We do, however, present a large number of examples that pertain to UNIX and other popular operating systems. In particular, we use Sun Microsystem's Solaris 2 operating system, a version of UNIX, which recently has been transformed into a modern operating system with support for threads at the kernel and user levels, symmetric multiprocessing, and real-time scheduling. Other examples used include Microsoft MS-DOS, Windows, and Windows/NT, IBM OS/2, the Apple Macintosh Operating System, and DEC VMS and TOPS-20, among others.

Prerequisites

As prerequisites, we assume that the reader is familiar with general computer organization and a high-level language, such as PASCAL. The hardware topics required for an understanding of operating systems are included in Chapter 2. We use pseudo-PASCAL notation for code examples, but the algorithms can be understood without a thorough knowledge of PASCAL.

Content of this Book

The text is organized in six major parts:

- **Overview** (Chapters 1 to 3). These chapters explain what operating systems *are*, what they *do*, and how they are *designed* and *constructed*. They explain how the concept of an operating system has developed, what the common features of an operating system are, what an operating system does for the user, and what it does for the computer-system operator. The presentation is motivational, historical, and explanatory in nature. We have avoided a discussion of how things are done internally in these chapters. Therefore, they are suitable for individuals or lower-level classes who want to learn what an operating system is, without getting into the details of the internal algorithms. Additionally, Chapter 2 covers the hardware topics which are important to an understanding of operating systems. Readers well-versed in hardware topics, including I/O, DMA, and hard disk operation, may choose to skim or skip this chapter.
- **Process management** (Chapters 4 to 7). The process concept and concurrency are at the very heart of modern operating systems. A *process* is the unit of work in a system. Such a system consists of a collection of *concurrently* executing processes, some of which are operating-system processes (those that execute system code), and the rest of which are user processes (those that execute user code). These chapters cover various methods for process scheduling, interprocess communication, process synchronization, and deadlock handling. Also included under this topic is a discussion of threads.
- **Storage management** (Chapters 8 to 12). A process must be in main memory (at least partially) during execution. To improve both the utilization of CPU and the speed of its response to its users, the computer must keep several processes in memory. There are many different memory-management schemes. These schemes reflect various approaches to memory management, and the effectiveness of the different algorithms depends on the particular situation. Since main memory is usually too small to accommodate all data and programs and cannot store data permanently, the computer system must provide secondary storage to back up main memory. Most modern computer systems use disks as the primary on-line storage medium for information (both programs and data). The file system provides the mechanism for on-line storage of and access to both data and programs residing on the disks. These chapters deal with the classic internal algorithms and structures of storage management. They provide a firm practical understanding of the algorithms used — the properties, advantages, and disadvantages.

- **Protection and security** (Chapters 13 and 14). The various processes in an operating system must be protected from one another's activities. For that purpose, mechanisms exist that can be used to ensure that the files, memory segments, CPU, and other resources can be operated on by only those processes that have gained proper authorization from the operating system. Protection is a mechanism for controlling the access of programs, processes, or users to the resources defined by a computer system. This mechanism must provide a means for specification of the controls to be imposed, together with some means of enforcement. Security protects the information stored in the system (both data and code), as well as the physical resources of the computer system, from unauthorized access, malicious destruction or alteration, and accidental introduction of inconsistency.
- **Distributed systems** (Chapters 15 to 18). A *distributed system* is a collection of processors that do not share memory or a clock. Such a system provides the user with access to the various resources the system maintains. Access to a shared resource allows computation speedup and improved data availability and reliability. Such a system also provides the user with a distributed file system, which is a file-service system whose users, servers, and storage devices are dispersed among the various sites of a distributed system. A distributed system must provide various mechanisms for process synchronization and communication, for dealing with the deadlock problem and the variety of failures that are not encountered in a centralized system.
- **Case studies** (Chapters 19 to 21). The various concepts described in this book can be drawn together by describing real operating systems. Two UNIX-based operating systems are covered in detail — Berkeley 4.3BSD and Mach. These operating systems were chosen in part because UNIX at one time was almost small enough to understand and yet was not a toy operating system. Most of its internal algorithms were selected for *simplicity*, not for speed or sophistication. UNIX is readily available to computer-science departments, so many students have access to it. Mach provides an opportunity for us to study a modern operating system that provides compatibility with 4.3BSD but has a drastically different design and implementation. Chapter 21 briefly describes some of the most influential operating systems.
- **The Nachos System** (Appendix). A good way to gain a deeper understanding of modern operating systems concepts is for the students to get their hands dirty — to take apart the code for an operating system, to see how it works at a low level, to build significant pieces of the operating system themselves, and to observe the impact of those changes. The Nachos instructional operating system, which is briefly described in the Appendix, provides the

opportunity to see how the basic concepts introduced in this text can be used to solve real-world problems. The Nachos system was developed by Professor Thomas Anderson from the University of California at Berkeley, to complement the third edition of this text, and it is freely available in the public domain via the Internet. Reviewers, who have used the Nachos project at other universities, call it a practical and positive supplement.

The Fourth Edition

Many comments and suggestions were forwarded to us concerning our previous editions. These, together with our own observations, have prodded us to produce this fourth edition. Our basic procedure was to reorganize and rewrite the material in each chapter, adding new information, examples, and diagrams where appropriate. We also brought older material up to date and removed material that was no longer of interest. Finally, we improved the exercises and updated the references.

Substantive revisions were made in the following chapters:

- **Chapter 1.** We have condensed some of the material related to older systems and have expanded our discussion of parallel, distributed, and real-time systems.
- **Chapter 2.** We collected coverage of strictly-hardware topics from the other chapters and reorganized them here, making this material easier to skip if it is already understood, and easier to use as a reference. We also expanded discussion of I/O topics, caching, and protection.
- **Chapter 4.** This chapter introduces the process concept. The material in this chapter appeared in parts of old Chapters 4 and 5. We moved the IPC material from old Chapter 5 to Chapter 4, since we believe that the material should be covered as part of the discussion on the process concept rather than as part of the process coordination chapter. We also expanded our discussion on threads considerably, and included Solaris 2 threads as an example.
- **Chapter 5.** This chapter is a reorganized old Chapter 4. It now deals primarily with CPU scheduling issues.
- **Chapter 6.** This chapter is a reorganized old Chapter 5. We removed Eisenberg and McGuire's solution to the critical-section problem for n processes from the main text (it is now an exercise). We also condensed the discussions concerning the critical region concept. We added new material on atomic transactions, including write-ahead logging and concurrency control schemes. Synchronization in Solaris 2 is included as an example.

- **Chapters 8 and 9.** We have added new material on up-to-date computer architectures that support paging and segmentation for large address spaces. Segmentation and paging are illuminated by an OS/2 example.
- **Chapters 10, 11, and 12.** We have expanded the material and completely reorganized the presentation of the file-system concept and implementation. We now present the logical aspect of the file system in Chapter 10, the implementation issues in Chapter 11, and the underlying secondary storage system in Chapter 12. We also have added new material on swap space, stable storage, recovery, reliability and performance.
- **Chapters 13 and 14.** We have separated old Chapter 11 into two chapters — one dealing with protection issues (Chapter 13), the other dealing with security issues (Chapter 14). In each of these chapters, we have reorganized the material, and have added new information. Major expansions include coverage of the Internet Worm and viruses.
- **Chapters 15 and 16.** We have separated old Chapter 12 into two chapters — one dealing with network structures (Chapter 15), the other dealing with distributed system structure (Chapter 16). In each of these chapters, we have reorganized the material, and have added new information. Major expansions include coverage of network protocols and functionality, remote services, thread-management, and the Open Software Foundation's Distributed Computing Environment (DCE) thread package.
- **Chapter 17.** This is old Chapter 14 on distributed file systems. We have brought the material up-to-date in this rapidly changing area.
- **Chapter 18.** This is old Chapter 13 on distributed coordination. We have brought the material up-to-date and added new sections on the two-phase commit protocol and concurrency control schemes.
- **Chapter 19.** This chapter on UNIX has been updated to reflect the current state of BSD UNIX and its current implementation.
- **Chapter 20.** This is old Chapter 16 on the Mach operating system. It has been updated to describe components of Mach version 3.
- **Appendix.** This is a new Appendix, which was authored by Professor Thomas Anderson from UC Berkeley. This Appendix provides a brief tutorial introduction to the Nachos system. The Appendix presents the philosophy governing the Nachos environment as well as providing a general introduction to the Nachos operating system and the five project activities which accompany the software. The Appendix concludes with instructions for retrieving Nachos from the Internet via ftp.

Mailing List and Supplements

We now provide an environment where users can communicate among themselves and with us. We have created a mailing list consisting of users of our book with the e-mail address — `os-book@cs.utexas.edu`. If you wish to be on the list, please send a message to `avi@cs.utexas.edu` indicating your name, affiliation, and e-mail address.

For information about the teaching supplements, which complement this book, mail may be sent to `os4e@aw.com`.

Errata

We have attempted to clean up every error in this new edition, but — as happens with operating systems — there will undoubtedly still be some obscure bugs. We would appreciate it if you, the reader, would notify us of any errors or omissions in the book. Also, if you would like to suggest improvements or to contribute exercises, we would be glad to hear from you. Any correspondence should be sent to A. Silberschatz, Department of Computer Sciences, The University of Texas.

Acknowledgments

This book is derived from the previous editions, all of which were coauthored by James Peterson. Other people that have helped with the previous editions include Randy Bentson, Jeff Brumfield, Gael Buckley, Thomas Casavant, Ajoy Kumar Datta, Joe Deck, Robert Fowler, G. Scott Graham, Rebecca Hartman, Wayne Hathaway, Christopher Haynes, Richard Kieburtz, Carol Kroll, Thomas LeBlanc, John Leggett, Michael Molloy, Ed Posnak, John Quarterman, Charles Oualline, John Stankovic, Steven Stepanek, Louis Stevens, and John Werth.

Lyn Dupré copyedited the book; Cliff Wilkes provided technical copyediting; Sara Strandtman edited our text into troff format. Debbie Lafferty, Tom Stone, and Helen Wythe were helpful with book production.

Chapter 17 was derived from a paper by Levy and Silberschatz [1990]. Chapter 19 was derived from a paper by Quarterman et al. [1985]. John Quarterman helped us to convert the material on UNIX 4.2BSD to UNIX 4.3BSD. David Black worked extensively with us to update Chapter 20.

We thank the following people, who reviewed this edition of the book: Joseph Boykin, P. C. Capon, John Carpenter, Thomas Doepfner, Caleb Drake, Hans Flack, Mark Holliday, Jerrold Leichter, Ted Leung, Gary Lippman, Carolyn Miller, Yoichi Muraoka, Jim M. Ng, Boris Putanec, Adam Stauffer, Hal Stern, David Umbaugh, Steve Vinoski, and J. S. Weston.

A.S.
P.B.G.

CONTENTS

PART ONE ■ OVERVIEW

Chapter 1 Introduction

- | | |
|--|----------------------------|
| 1.1 What Is an Operating System? 3 | 1.7 Parallel Systems 20 |
| 1.2 Early Systems 6 | 1.8 Distributed Systems 22 |
| 1.3 Simple Batch Systems 7 | 1.9 Real-Time Systems 23 |
| 1.4 Multiprogrammed Batched Systems 13 | 1.10 Summary 25 |
| 1.5 Time-Sharing Systems 15 | Exercises 26 |
| 1.6 Personal-Computer Systems 17 | Bibliographic Notes 27 |

Chapter 2 Computer-System Structures

- | | |
|----------------------------------|------------------------------------|
| 2.1 Computer-System Operation 29 | 2.6 General-System Architecture 51 |
| 2.2 I/O Structure 32 | 2.7 Summary 52 |
| 2.3 Storage Structure 37 | Exercises 53 |
| 2.4 Storage Hierarchy 42 | Bibliographic Notes 55 |
| 2.5 Hardware Protection 45 | |

Chapter 3 Operating-System Structures

- | | |
|----------------------------------|-------------------------|
| 3.1 System Components 57 | 3.4 System Programs 74 |
| 3.2 Operating-System Services 63 | 3.5 System Structure 76 |
| 3.3 System Calls 65 | 3.6 Virtual Machines 82 |

3.7 System Design and Implementation	86	3.9 Summary	90
3.8 System Generation	89	Exercises	91
		Bibliographic Notes	92

PART TWO ■ PROCESS MANAGEMENT

Chapter 4 Processes

4.1 Process Concept	97	4.6 Interprocess Communication	116
4.2 Process Scheduling	100	4.7 Summary	126
4.3 Operation on Processes	105	Exercises	127
4.4 Cooperating Processes	108	Bibliographic Notes	129
4.5 Threads	111		

Chapter 5 CPU Scheduling

5.1 Basic Concepts	131	5.6 Algorithm Evaluation	152
5.2 Scheduling Criteria	135	5.7 Summary	158
5.3 Scheduling Algorithms	137	Exercises	159
5.4 Multiple-Processor Scheduling	149	Bibliographic Notes	161
5.5 Real-Time Scheduling	150		

Chapter 6 Process Synchronization

6.1 Background	163	6.7 Monitors	190
6.2 The Critical-Section Problem	165	6.8 Synchronization in Solaris 2	198
6.3 Synchronization Hardware	172	6.9 Atomic Transactions	199
6.4 Semaphores	175	6.10 Summary	208
6.5 Classical Problems of Synchronization	181	Exercises	210
6.6 Critical Regions	186	Bibliographic Notes	214

Chapter 7 Deadlocks

7.1 System Model	217	7.4 Deadlock Prevention	224
7.2 Deadlock Characterization	219	7.5 Deadlock Avoidance	227
7.3 Methods for Handling Deadlocks	223	7.6 Deadlock Detection	234
		7.7 Recovery from Deadlock	238

7.8 Combined Approach to Deadlock Handling	240	Bibliographic Notes	245
7.9 Summary	241	Exercises	242

PART THREE ■ STORAGE MANAGEMENT

Chapter 8 Memory Management

8.1 Background	249	8.6 Segmentation	283
8.2 Logical versus Physical Address Space	255	8.7 Segmentation with Paging	290
8.3 Swapping	256	8.8 Summary	294
8.4 Contiguous Allocation	259	Exercises	296
8.5 Paging	267	Bibliographic Notes	299

Chapter 9 Virtual Memory

9.1 Background	301	9.7 Thrashing	329
9.2 Demand Paging	303	9.8 Other Considerations	334
9.3 Performance of Demand Paging	309	9.9 Demand Segmentation	341
9.4 Page Replacement	312	9.10 Summary	342
9.5 Page-Replacement Algorithms	315	Exercises	343
9.6 Allocation of Frames	326	Bibliographic Notes	348

Chapter 10 File-System Interface

10.1 File Concept	349	10.5 Consistency Semantics	378
10.2 Access Methods	358	10.6 Summary	379
10.3 Directory Structure	361	Exercises	380
10.4 Protection	373	Bibliographic Notes	381

Chapter 11 File-System Implementation

11.1 File-System Structure	383	11.6 Recovery	403
11.2 Allocation Methods	387	11.7 Summary	405
11.3 Free-Space Management	397	Exercises	406
11.4 Directory Implementation	399	Bibliographic Notes	408
11.5 Efficiency and Performance	401		

Chapter 12 Secondary-Storage Structure

- 12.1 Disk Structure 409
- 12.2 Disk Scheduling 410
- 12.3 Disk Management 417
- 12.4 Swap-Space Management 419
- 12.5 Disk Reliability 422
- 12.6 Stable-Storage Implementation 424
- 12.7 Summary 425
- Exercises 426
- Bibliographic Notes 427

PART FOUR ■ PROTECTION AND SECURITY

Chapter 13 Protection

- 13.1 Goals of Protection 431
- 13.2 Domain of Protection 432
- 13.3 Access Matrix 438
- 13.4 Implementation of Access Matrix 443
- 13.5 Revocation of Access Rights 446
- 13.6 Capability-Based Systems 448
- 13.7 Language-Based Protection 451
- 13.8 Summary 455
- Exercises 455
- Bibliographic Notes 457

Chapter 14 Security

- 14.1 The Security Problem 459
- 14.2 Authentication 461
- 14.3 Program Threats 464
- 14.4 System Threats 465
- 14.5 Threat Monitoring 469
- 14.6 Encryption 471
- 14.7 Summary 473
- Exercises 473
- Bibliographic Notes 474

PART FIVE ■ DISTRIBUTED SYSTEMS

Chapter 15 Network Structures

- 15.1 Background 479
- 15.2 Motivation 481
- 15.3 Topology 482
- 15.4 Network Types 488
- 15.5 Communication 491
- 15.6 Design Strategies 498
- 15.7 Networking Example 501
- 15.8 Summary 504
- Exercises 504
- Bibliographic Notes 505

Chapter 16 Distributed-System Structures

- | | | | |
|------------------------------------|-----|---------------------|-----|
| 16.1 Network-Operating Systems | 507 | 16.5 Design Issues | 519 |
| 16.2 Distributed-Operating Systems | 509 | 16.6 Summary | 521 |
| 16.3 Remote Services | 512 | Exercises | 522 |
| 16.4 Robustness | 517 | Bibliographic Notes | 523 |

Chapter 17 Distributed-File Systems

- | | | | |
|--|-----|----------------------|-----|
| 17.1 Background | 525 | 17.6 Example Systems | 539 |
| 17.2 Naming and Transparency | 527 | 17.7 Summary | 567 |
| 17.3 Remote File Access | 531 | Exercises | 568 |
| 17.4 Stateful versus Stateless Service | 536 | Bibliographic Notes | 569 |
| 17.5 File Replication | 538 | | |

Chapter 18 Distributed Coordination

- | | | | |
|--------------------------|-----|--------------------------|-----|
| 18.1 Event Ordering | 571 | 18.6 Election Algorithms | 595 |
| 18.2 Mutual Exclusion | 574 | 18.7 Reaching Agreement | 598 |
| 18.3 Atomicity | 577 | 18.8 Summary | 600 |
| 18.4 Concurrency Control | 581 | Exercises | 601 |
| 18.5 Deadlock Handling | 586 | Bibliographic Notes | 602 |

PART SIX ■ CASE STUDIES

Chapter 19 The UNIX System

- | | | | |
|---------------------------|-----|---------------------------------|-----|
| 19.1 History | 607 | 19.7 File System | 636 |
| 19.2 Design Principles | 613 | 19.8 I/O System | 645 |
| 19.3 Programmer Interface | 615 | 19.9 Interprocess Communication | 649 |
| 19.4 User Interface | 623 | 19.10 Summary | 655 |
| 19.5 Process Management | 627 | Exercises | 655 |
| 19.6 Memory Management | 632 | Bibliographic Notes | 657 |

Chapter 20 The Mach System

- | | | | |
|---------------------------------|-----|---------------------------|-----|
| 20.1 History | 659 | 20.6 Memory Management | 679 |
| 20.2 Design Principles | 661 | 20.7 Programmer Interface | 685 |
| 20.3 System Components | 662 | 20.8 Summary | 686 |
| 20.4 Process Management | 666 | Exercises | 687 |
| 20.5 Interprocess Communication | 673 | Bibliographic Notes | 688 |

Chapter 21 Historical Perspective

21.1 Atlas	691	21.5 CTSS	695
21.2 XDS-940	692	21.6 MULTICS	696
21.3 THE	693	21.7 OS/360	696
21.4 RC 4000	694	21.8 Other Systems	698

Appendix The Nachos System

A.1 Overview	700	A.5 Conclusions	713
A.2 Nachos Software Structure	702	Bibliographic Notes	713
A.3 Sample Assignments	705		
A.4 Information on Obtaining a Copy of Nachos	711		

Bibliography 715

Credits 745

Index 747

PART ONE

OVERVIEW

An *operating system* is a program that acts as an intermediary between a user of a computer and the computer hardware. The purpose of an operating system is to provide an environment in which a user can execute programs in a *convenient* and *efficient* manner.

We trace the development of operating systems from the first hands-on systems to current multiprogrammed and time-shared systems. Understanding the reasons behind the development of operating systems gives us an appreciation for what an operating system does and how it does it.

The operating system must ensure the correct operation of the computer system. To prevent user programs from interfering with the proper operation of the system, the hardware must provide appropriate mechanisms to ensure such proper behavior. We describe the basic computer architecture that makes it possible to write a correct operating system.

The operating system provides certain services to programs and to the users of those programs in order to make the programming task easier. The specific services provided will, of course, differ from one operating system to another, but there are some common classes of services that we identify and explore.