EIGHTH EDITION

Silberschatz
Galvin
Gagne

OPERATING SYSTEM CONCEPTS with JAVA

International Student Version

Operating System Concepts with Java

Eighth Edition

International Student Version

ABRAHAM SILBERSCHATZ
Yale University

PETER BAER GALVIN
Corporate Technologies, Inc.

GREG GAGNE
Westminster College

JOHN WILEY & SONS. INC

ISBN: 978-0-470-39879-1

Printed in Asia

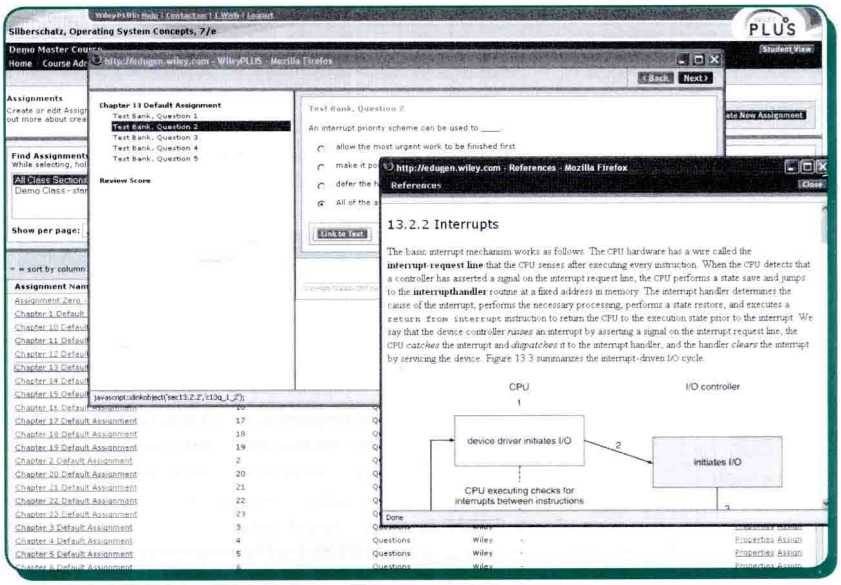
10 9 8 7 6 5 4 3 2

Why WileyPLUS for Computer Science?

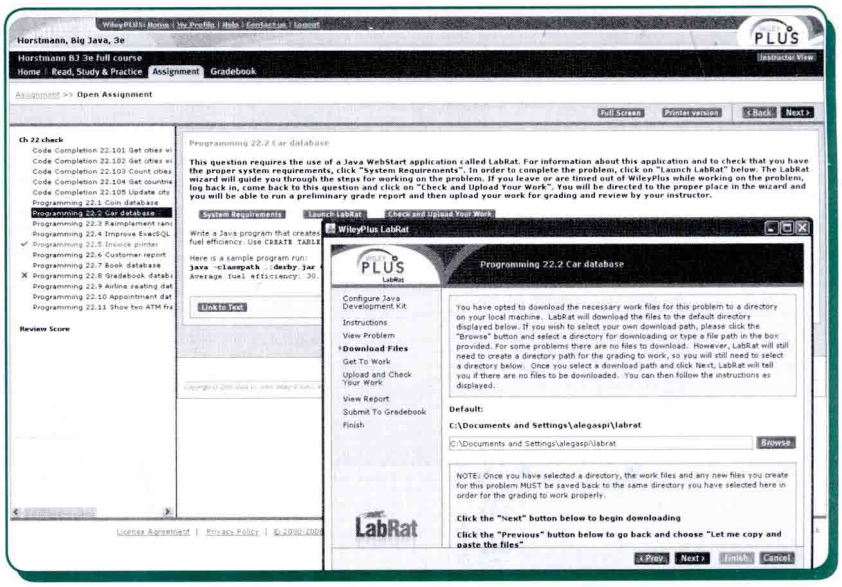
WileyPLUS for Computer Science is a dynamic online environment that motivates students to spend more time practicing the problems and explore the material they need to master in order to succeed.

"I used the homework problems to practice before quizzes and tests. I re-did the problems on my own and then checked WileyPLUS to compare answers." – Student Alexandra Leifer, Ithaca College

- ***The integrated, online text offers students a low-cost alternative to the printed text, enhanced with direct links to specific portions of the text, plus interactive animations and videos to help students visualize concepts.***



- ***Select WileyPLUS courses include LabRat, the programming assignment tool allows you to choose from hundreds of programming exercises to automate the process of assigning, completing, compiling, testing, submitting, and evaluating programming assignments. When students spend more time practicing, they come to class better prepared.***



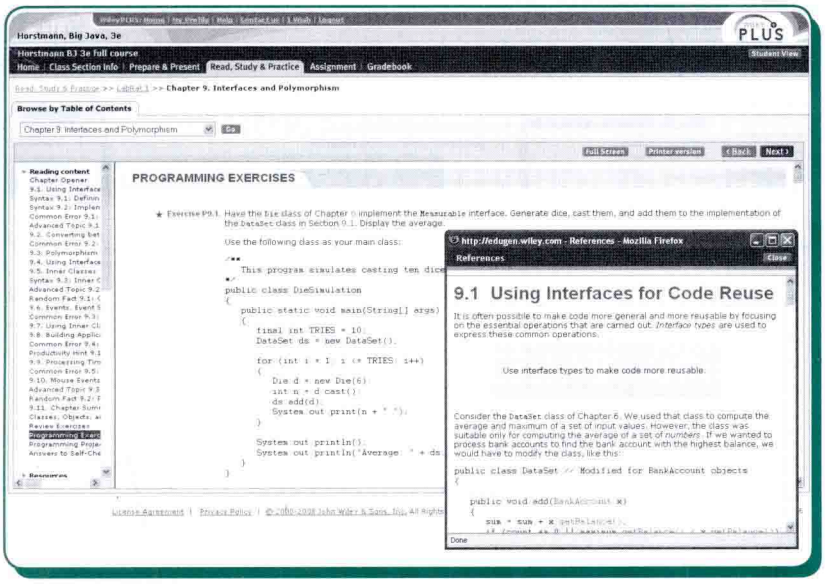
Try WileyPLUS with LabRat:
www.wileyplus.com/tours

See–and Try WileyPLUS in action!
Details and Demo: www.wileyplus.com

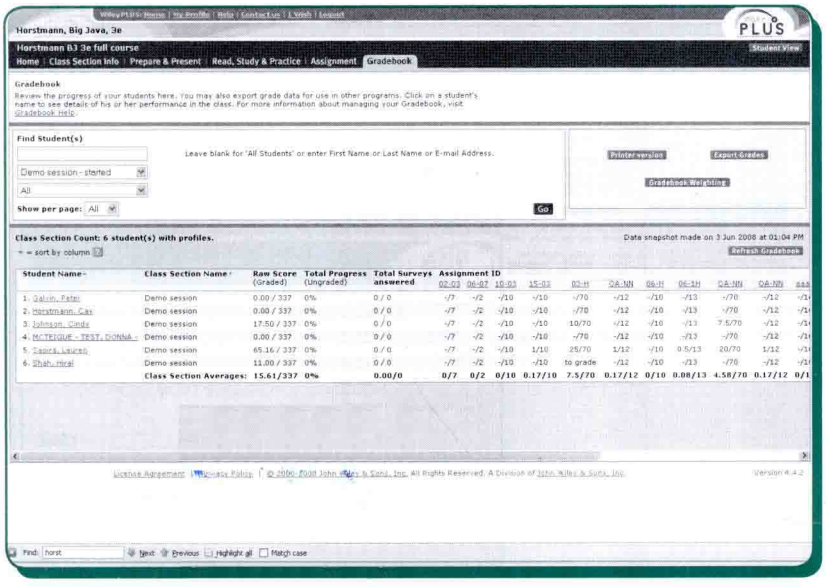
WILEY PLUS
www.wileyplus.com

WileyPLUS combines robust course management tools with the complete online text and all of the interactive teaching & learning resources you and your students need in one easy-to-use system.

"Overall the WileyPLUS package made the material more interesting than if it was just a book with no additional material other than what you need to know."
– Edin Alic, North Dakota State University



- *Students easily access source code for example problems, as well as self-study materials and all the exercises and readings you assign.*



- *All instructional materials, including PowerPoints, illustrations and visual tools, source code, exercises, solutions and gradebook organized in one easy-to-use system.*

"WileyPLUS made it a lot easier to study. I got an A!"
– Student Jeremiah Ellis Mattson, North Dakota State University

WILEY PLUS
www.wileyplus.com

To my Valerie

Avi Silberschatz

To my parents, Brendan and Ellen Galvin

Peter Baer Galvin

To Steve, Ray, and Bobbie

Greg Gagne

Abraham Silberschatz is the Sidney J. Weinberg Professor & Chair of Computer Science at Yale University. Prior to joining Yale, he was the Vice President of the Information Sciences Research Center at Bell Laboratories. Prior to that, he held a chaired professorship in the Department of Computer Sciences at the University of Texas at Austin.

Professor Silberschatz is an ACM Fellow, an IEEE Fellow, and a member of the Connecticut Academy of Science and Engineering. He received the 2002 IEEE Taylor L. Booth Education Award, the 1998 ACM Karl V. Karlstrom Outstanding Educator Award, and the 1997 ACM SIGMOD Contribution Award. In recognition of his outstanding level of innovation and technical excellence, he was awarded the Bell Laboratories President's Award for three different projects—the QTM Project (1998), the DataBlitz Project (1999), and the NetInventory Project (2004).

Professor Silberschatz' writings have appeared in numerous ACM and IEEE publications and other professional conferences and journals. He is a coauthor of the textbook *Database System Concepts*. He has also written Op-Ed articles for the New York Times, the Boston Globe, and the Hartford Courant, among others.

Peter Baer Galvin is the CTO for Corporate Technologies (www.cptech.com), a computer facility reseller and integrator. Before that, Mr. Galvin was the systems manager for Brown University's Computer Science Department. He is also Sun columnist for *;login:* magazine. Mr. Galvin has written articles for *Byte* and other magazines, and has written columns for *SunWorld* and *SysAdmin* magazines. As a consultant and trainer, he has given talks and taught tutorials on security and system administration worldwide.

Greg Gagne is chair of the Computer Science department at Westminster College in Salt Lake City where he has been teaching since 1990. In addition to teaching operating systems, he also teaches computer networks, distributed systems, and software engineering. He also provides workshops to computer science educators and industry professionals.

Preface

Operating systems are an essential part of any computer system. Similarly, a course on operating systems is an essential part of any computer-science education. This field is undergoing rapid change, as computers are now prevalent in virtually every application, from games for children through the most sophisticated planning tools for governments and multinational firms. Yet the fundamental concepts remain fairly clear, and it is on these that we base this book.

We wrote this book as a text for an introductory course in operating systems at the junior or senior undergraduate level or at the first-year graduate level. We hope that practitioners will also find it useful. It provides a clear description of the *concepts* that underlie operating systems. As prerequisites, we assume that the reader is familiar with basic data structures, computer organization, and a high-level language, preferably Java. The hardware topics required for an understanding of operating systems are included in Chapter 1. For code examples, we use predominantly Java, with some C, but the reader can still understand the algorithms without a thorough knowledge of these languages.

Concepts are presented using intuitive descriptions. Important theoretical results are covered, but formal proofs are omitted. The bibliographical notes at the end of each chapter contain pointers to research papers in which results were first presented and proved, as well as references to material for further reading. In place of proofs, figures and examples are used to suggest why we should expect the result in question to be true.

The fundamental concepts and algorithms covered in the book are often based on those used in existing commercial operating systems. Our aim is to present these concepts and algorithms in a general setting that is not tied to one particular operating system. We present a large number of examples that pertain to the most popular and the most innovative operating systems, including Sun Microsystems' Solaris; Linux; Microsoft Windows Vista, Windows 2000, and Windows XP; and Apple Mac OS X. When we refer to Windows XP as an example operating system, we are implying Windows Vista, Windows XP, and Windows 2000. If a feature exists in a specific release, we state this explicitly.

Organization of This Book

The organization of this text reflects our many years of teaching courses on operating systems. Consideration was also given to the feedback provided by the reviewers of the text, as well as comments submitted by readers of earlier editions. In addition, the content of the text corresponds to the suggestions from *Computing Curricula 2005* for teaching operating systems, published by the Joint Task Force of the IEEE Computing Society and the Association for Computing Machinery (ACM).

Online support for the text is provided by WileyPLUS. On this site, students can find sample exercises and programming problems, and instructors can assign and grade problems. In addition, on WileyPLUS, students can access new operating-system simulators, which they can use to work through exercises and hands-on lab activities. References to the simulators and associated activities appear at the ends of several chapters in the text.

Content of This Book

The text is organized in eight major parts:

- **Overview**. Chapters 1 and 2 explain what operating systems *are*, what they *do*, and how they are *designed* and *constructed*. These chapters discuss 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 and explanatory in nature. We have avoided a discussion of how things are done internally in these chapters. Therefore, they are suitable for individual readers or for students in lower-level classes who want to learn what an operating system is without getting into the details of the internal algorithms.
- **Process management**. Chapters 3 through 7 describe the process concept and concurrency as the 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 methods for process scheduling, interprocess communication, process synchronization, and deadlock handling. Also included is a discussion of threads, as well as an examination of issues related to multicore systems.
- **Memory management**. Chapters 8 and 9 deal with the management of main memory during the execution of a process. To improve both the utilization of the 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 reflecting various approaches to memory management, and the effectiveness of a particular algorithm depends on the situation.

- **Storage management**. Chapters 10 through 13 describe how the file system, mass storage, and I/O are handled in a modern computer system. The file system provides the mechanism for on-line storage of and access to both data and programs. We describe the classic internal algorithms and structures of storage management and provide a firm practical understanding of the algorithms used—their properties, advantages, and disadvantages. Our discussion of storage also includes matters related to secondary and tertiary storage. Since the I/O devices that attach to a computer vary widely, the operating system needs to provide a wide range of functionality to applications to allow them to control all aspects of these devices. We discuss system I/O in depth, including I/O system design, interfaces, and internal system structures and functions. In many ways, I/O devices are the slowest major components of the computer. Because they represent a performance bottleneck, we also examine performance issues associated with I/O devices.
- **Protection and security**. Chapters 14 and 15 discuss the mechanisms necessary for the protection and security of computer systems. The processes in an operating system must be protected from one another's activities, and to provide such protection, we must ensure that only processes that have gained proper authorization from the operating system can operate on the files, memory, CPU, and other resources of the 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 of specifying the controls to be imposed, as well as a means of enforcement. Security protects the integrity of the information stored in the system (both data and code), as well as the physical resources of the system, from unauthorized access, malicious destruction or alteration, and accidental introduction of inconsistency.
- **Case studies**. Chapters 16 through 18 in the book, and Appendices A through C (which are available on www.wiley.com/go/global/silberschatz and WileyPLUS), integrate the concepts described in the earlier chapters by describing real operating systems. These systems include Linux, Windows XP, FreeBSD, Mach, and Windows 2000. We chose Linux and FreeBSD because UNIX—at one time—was almost small enough to understand yet was not a "toy" operating system. Most of its internal algorithms were selected for *simplicity*, rather than for speed or sophistication. Both Linux and FreeBSD are readily available to computer-science departments, so many students have access to these systems. We chose Windows XP and Windows 2000 because they provide an opportunity for us to study a modern operating system with a design and implementation drastically different from those of UNIX. Chapter 18 briefly describes a few other influential operating systems.

- **Distributed systems**. Chapters 19 through 21 deal with a collection of processors that do not share memory or a clock—a *distributed system*. By providing the user with access to the various resources that it maintains, a distributed system can improve computation speed and 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 sites of a distributed system. A distributed system must provide various mechanisms for process synchronization and communication, as well as for dealing with deadlock problems and a variety of failures that are not encountered in a centralized system.
- **Special-purpose systems**. Chapters 22 and 23 deal with systems used for specific purposes, including real-time systems and multimedia systems. These systems have specific requirements that differ from those of the general-purpose systems that are the focus of the remainder of the text. Real-time systems may require not only that computed results be "correct" but also that the results be produced within a specified deadline period. Multimedia systems require quality-of-service guarantees ensuring that the multimedia data are delivered to clients within a specific time frame.

Operating-System Environments

This book uses examples of many real-world operating systems to illustrate fundamental operating-system concepts. However, particular attention is paid to the Microsoft family of operating systems (including Windows Vista, Windows 2000, and Windows XP) and various versions of UNIX (including Solaris, BSD, and Mac OS X). We also provide a significant amount of coverage of the Linux operating system, reflecting the most recent version of the kernel —Version 2.6—at the time this book was written.

The text uses Java to illustrate many operating-system concepts, such as multithreading, CPU scheduling, process synchronization, deadlock, memory and file management, security, networking, and distributed systems. Java is more a technology than a programming language, so it is an excellent vehicle for demonstrations.

Much of the Java-related material included has been developed and class-tested in undergraduate operating-systems classes. From our experience, students entering these classes lacking knowledge of Java—but with experience using C++ and basic object-oriented principles—generally have little trouble with Java. Rather, most difficulties lie in understanding such concepts as multithreading and data sharing by multiple, concurrently running threads. These concepts are systemic rather than being specific to Java; even students with a sound knowledge of Java are likely to have difficulty with them. We thus emphasize concepts of operating systems rather than concentrating on Java syntax.

All the Java programs in this text compile with versions 1.5 and 1.6 of the Java Software Development Kit (SDK). Java 1.5 provides several new features—both at the language level and at the API level—that enhance Java's usefulness for studying operating systems. We include several new additions to the Java 1.5 API throughout this text. Many programs provided in this text will not compile with earlier releases of the Java SDK, and we encourage all readers to use Java 1.5 as a minimum Java configuration.

The text also provides a few example programs written in C that are intended to run in the Windows and POSIX programming environments. POSIX (which stands for *Portable Operating System Interface*) represents a set of standards implemented primarily for UNIX-based operating systems. Although Windows Vista, Windows XP, and Windows 2000 systems can also run certain POSIX programs, our coverage of POSIX focuses primarily on UNIX and Linux systems.

The Eighth Edition

As we wrote the Eighth Edition of *Operating System Concepts with Java*, we were guided by the many comments and suggestions we received from readers of our previous editions, as well as by our own observations about the rapidly changing fields of operating systems and networking. We have rewritten material in most of the chapters by bringing older material up to date and removing material that was no longer of interest or relevance.

We have made substantive revisions and organizational changes in many of the chapters. Most importantly, we have added coverage of open-source operating systems in Chapter 1. We have also added more practice exercises for students and included solutions in WileyPLUS, which also includes new simulators to provide demonstrations of operating-system operation. Below, we provide a brief outline of the major changes to the various chapters:

- **Chapter 1, Introduction,** has been expanded to include multicore CPUs, clustered computers, and open-source operating systems.
- **Chapter 2, System Structures,** provides significantly updated coverage of virtual machines, as well as multicore CPUs, the GRUB boot loader, and operating-system debugging.
- **Chapter 4, Multithreaded Programming,** adds new coverage of programming for multicore systems and updates the coverage of Java thread states.
- **Chapter 5, Process Scheduling,** adds coverage of virtual machine scheduling and multithreaded, multicore architectures. It also includes new scheduling features in Java 1.5.
- **Chapter 6, Synchronization,** adds a discussion of mutual exclusion locks, priority inversion, and transactional memory.
- **Chapter 8, Memory-Management Strategies,** includes a discussion of NUMA.
- **Chapter 9, Virtual-Memory Management,** updates the Solaris example to include Solaris 10 memory management.
- **Chapter 10, File-System,** is updated with current technologies and capacities.
- **Chapter 11, Implementing File-Systems,** includes a full description of Sun's ZFS file system and expands the coverage of volumes and directories.
- **Chapter 12, Secondary-Storage Structure,** adds coverage of iSCSI, volumes, and ZFS pools.

- **Chapter 13, I/O Systems,** adds coverage of PCIX PCI Express, and HyperTransport.
- **Chapter 16, The Linux System,** has been updated to cover the latest version of the Linux kernel.
- **Chapter 18, Influential Operating Systems,** increases coverage of very early computers as well as TOPS-20, CP/M, MS-DOS, Windows, and the original Mac OS.
- **Chapter 19, Distributed System Structures (On WileyPlus),** adds coverage of 802.11 wireless networks.

Programming Problems and Projects

To emphasize the concepts presented in the text, we have added or modified 12 programming problems and projects using Java. In general, programming *projects* are more detailed and require a greater time commitment than programming *problems*. These problems and projects emphasize processes and interprocess communication, threads, process synchronization, virtual memory, file systems, and networking. New programming problems and projects include implementing socket programming, using Java's remote method invocation (RMI), working with multithreaded sorting programming, listing threads in the Java virtual machine (JVM), designing a process identifier management system, and managing virtual memory.

The Eighth Edition also incorporates a set of operating-system simulators designed by Steven Robbins of the University of Texas at San Antonio. The simulators are intended to model the behavior of an operating system as it performs various tasks, such as CPU and disk-head scheduling, process creation and interprocess communication, starvation, and address translation. These simulators are written in Java and will run on any computer system with Java 1.4. Students can download the simulators from WileyPLUS and observe the behavior of several operating-system concepts in various scenarios. In addition, each simulator includes several exercises that ask students to set certain parameters of the simulator, observe how the system behaves, and then explain this behavior. These exercises can be assigned through WileyPLUS. The WileyPLUS course also includes algorithmic problems and tutorials.

Teaching Supplements

The following teaching supplements are available on WileyPLUS and www.wiley.com/go/global/silberschatz: a set of slides to accompany the book, model course syllabi, all Java and C source code, up-to-date errata, three case-study appendices, and the Distributed Communication appendix. The WileyPLUS course also contains simulators and associated exercises, additional practice exercises (with solutions) not found in the text, and a test bank of additional problems. Students are encouraged to solve the practice exercises on their own and then use the solutions provided to check their own answers.

To obtain restricted supplements, such as the solution guide to the exercises in the text, contact your local John Wiley & Sons sales representative. Note that these supplements are available only to faculty who use this text. Please contact your Wiley representative for access to the instructor resources.

Contacting Us

We have attempted to clean up every error in this new edition, but—as happens with operating systems—a few obscure bugs may remain; an up-to-date errata list is accessible from the book's home page. We would appreciate hearing from you about any textual errors or omissions in the book that are not on the current list of errata.

We would be glad to receive suggestions on improvements to the book. We also welcome any contributions to the book's Web page that could be of use to other readers, such as programming exercises, project suggestions, on-line labs and tutorials, and teaching tips.

E-mail should be addressed to os-book-authors@cs.yale.edu. Any other correspondence should be sent to Avi Silberschatz, Department of Computer Science, Yale University, 51 Prospect Street, P.O. Box 208285, New Haven, CT 06520-8285 USA.

Acknowledgments

This book is derived from the previous editions, the first three of which were coauthored by James Peterson. Others who helped us with previous editions include Hamid Arabnia, Rida Bazzi, Randy Bentson, David Black, Joseph Boykin, Jeff Brumfield, Gael Buckley, Roy Campbell, P. C. Capon, John Carpenter, Gil Carrick, Thomas Casavant, Bart Childs, Ajoy Kumar Datta, Joe Deck, Sudarshan K. Dhall, Thomas Doeppner, Caleb Drake, M. Racsit Eskicioğlu, Hans Flack, Robert Fowler, G. Scott Graham, Richard Guy, Max Hailperin, Rebecca Hartman, Wayne Hathaway, Christopher Haynes, Don Heller, Bruce Hillyer, Mark Holliday, Dean Hougen, Michael Huangs, Ahmed Kamel, Richard Kieburtz, Carol Kroll, Morty Kwestel, Thomas LeBlanc, John Leggett, Jerrold Leichter, Ted Leung, Gary Lippman, Carolyn Miller, Michael Molloy, Euripides Montagne, Yoichi Muraoka, Jim M. Ng, Banu Özden, Ed Posnak, Boris Putanec, Charles Qualline, John Quarterman, Mike Reiter, Gustavo Rodriguez-Rivera, Carolyn J. C. Schauble, Thomas P. Skinner, Yannis Smaragdakis, Jesse St. Laurent, John Stankovic, Adam Stauffer, Steven Stepanek, John Sterling, Hal Stern, Louis Stevens, Pete Thomas, David Umbaugh, Steve Vinoski, Tommy Wagner, Larry L. Wear, John Werth, James M. Westall, J. S. Weston, and Yang Xiang

Parts of Chapter 12 were derived from a paper by Hillyer and Silberschatz [1996]. Parts of Chapter 17 were derived from a paper by Levy and Silberschatz [1990]. Chapter 21 was derived from an unpublished manuscript by Stephen Tweedie. Chapter 22 was derived from an unpublished manuscript by Dave Probert, Cliff Martin, and Avi Silberschatz. Appendix C was derived from

an unpublished manuscript by Cliff Martin. Cliff Martin also helped with updating the UNIX appendix to cover FreeBSD. Some of the exercises and accompanying solutions were supplied by Arvind Krishnamurthy. Marilyn Turnamian helped generate figures and presentation slides.

Mike Shapiro, Bryan Cantrill, and Jim Mauro answered several Solaris-related questions. Bryan Cantrill from Sun Microsystems helped with the ZFS coverage. Steve Robbins of the University of Texas at San Antonio designed the set of simulators that we incorporate in WileyPLUS. Reece Newman of Westminster College initially explored this set of simulators and their appropriateness for this text. Jason Belcher provided assistance with Java generics. Josh Dees and Rob Reynolds contributed coverage of Microsoft's .NET. Scott M. Pike of Texas A&M University contributed several algorithmic problems and tutorials for WileyPLUS.

Judi Paige helped generate figures and presentation slides. Mark Wogahn has made sure that the software to produce the book (such as Latex macros and fonts) works properly.

Our executive editor, Beth Golub, provided expert guidance as we prepared this edition. She was assisted by Mike Berlin, who managed many details of this project smoothly. The Senior Production Editor, Ken Santor, was instrumental in handling all the production details. Lauren Sapira has been very helpful with getting material ready and available for WileyPlus.

The cover illustrator was Susan Cyr, and the cover designer was Howard Grossman. Beverly Peavler copy-edited the manuscript. The freelance proof-reader was Katrina Avery; the freelance indexer was WordCo, Inc.

Finally, we would like to add some personal notes. Avi would like to thank Valerie for her support during the preparation of this new edition. Peter would like to thank his colleagues at Corporate Technologies. Greg would like to acknowledge three colleagues from Westminster College who were instrumental in his hiring in 1990: his dean, Ray Ownbey; program chair, Bobbie Fredsall; and academic vice president, Stephen Baar.

Abraham Silberschatz, New Haven, CT, 2009
Peter Baer Galvin, Burlington, MA, 2009
Greg Gagne, Salt Lake City, UT, 2009

Contents

Chapter 4 Multithreaded Programming

Chapter 5 Process Scheduling

Chapter 6 Synchronization

Chapter 7 Deadlocks

PART THREE ■ MEMORY MANAGEMENT

Chapter 8 Memory-Management Strategies