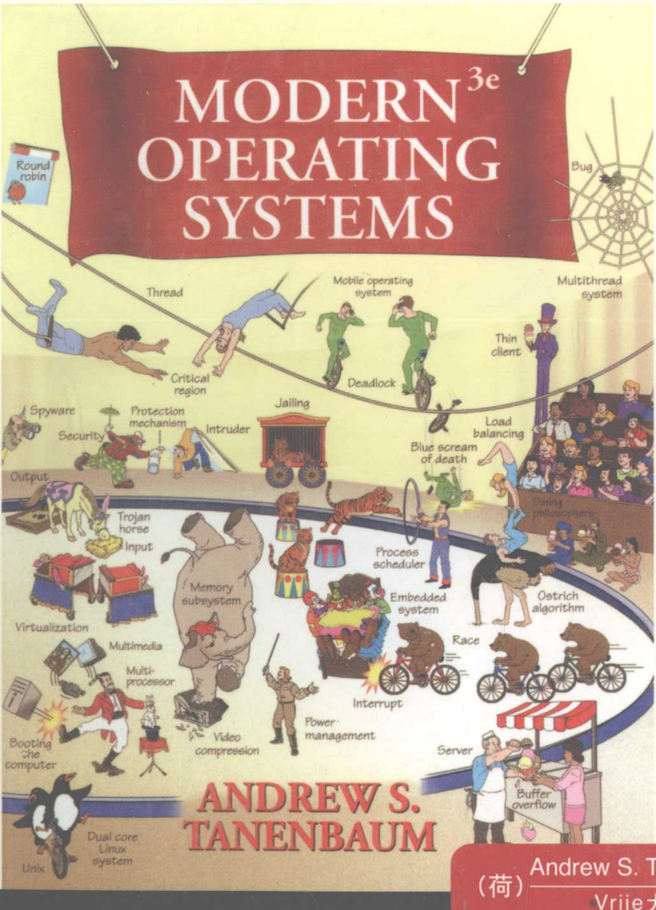
HZ BOOKS
华章教育

PEARSON
Prentice Hall

经 典 原 版 书 库

现代操作系统

(英文版 · 第3版)



(荷) Andrew S. Tanenbaum 著
Vrije大学

机械工业出版社
China Machine Press

经 典 原 版 书 库

现代操作系统

（英文版 · 第3版）

Modern Operating Systems

(Third Edition)

（荷） Andrew S. Tanenbaum 著
Vrije大学

机械工业出版社
China Machine Press

本书版权登记号：图字：01-2009-1357

图书在版编目（CIP）数据

现代操作系统（英文版·第3版）/（荷）塔嫩鲍姆（Tanenbaum, A. S.）著. —北京：机械工业出版社，2009.3
（经典原版书库）
书名原文：Modern Operating Systems, Third Edition

ISBN 978-7-111-26527-6

Ⅰ. 现…　Ⅱ. 塔…　Ⅲ. 操作系统-英文　Ⅳ. TP316

中国版本图书馆CIP数据核字（2009）第031374号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：迟振春
北京京师印务有限公司印刷
2009年3月第1版第1次印刷
150mm×214mm · 34.5印张
标准书号：ISBN 978-7-111-26527-6
定价：69.00元

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章分社较早意识到“出版要为教育服务”。自1998年开始，华章分社就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与Pearson，McGraw-Hill，Elsevier，MIT，John Wiley & Sons，Cengage等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出Andrew S. Tanenbaum，Bjarne Stroustrup，Brain W. Kernighan，Dennis Ritchie，Jim Gray，Afred V. Aho，John E. Hopcroft，Jeffrey D. Ullman，Abraham Silberschatz，William Stallings，Donald E. Knuth，John L. Hennessy，Larry L. Peterson等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工

作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章分社欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

华章网站：www.hzbook.com
电子邮件：hzjsj@hzbook.com
联系电话：(010) 88379604
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

HZ BOOKS
华章教育

PREFACE

The third edition of this book differs from the second edition in numerous ways. To start with, the chapters have been reordered to place the central material at the beginning. There is also now more of a focus on the operating system as the creator of abstractions. Chapter 1, which has been heavily updated, introduces all the concepts. Chapter 2 is about the abstraction of the CPU into multiple processes. Chapter 3 is about the abstraction of physical memory into address spaces (virtual memory). Chapter 4 is about the abstraction of the disk into files. Together, processes, virtual address spaces, and files are the key concepts that operating systems provide, so these chapters are now placed earlier than they previously had been.

Chapter 1 has been heavily modified and updated in many places. For example, an introduction to the C programming language and the C run-time model is given for readers familiar only with Java.

In Chapter 2, the discussion of threads has been revised and expanded reflecting their new importance. Among other things, there is now a section on IEEE standard Pthreads.

Chapter 3, on memory management, has been reorganized to emphasize the idea that one of the key functions of an operating system is to provide the abstraction of a virtual address space for each process. Older material on memory management in batch systems has been removed, and the material on the implementation of paging has been updated to focus on the need to make it handle the larger address spaces now common and also the need for speed.

Chapters 4–7 have been updated, with older material removed and some new material added. The sections on current research in these chapters have been rewritten from scratch. Many new problems and programming exercises have been added.

Chapter 8 has been updated, including some material on multicore systems. A whole new section on virtualization technology, hypervisors, and virtual machines, has been added with VMware used as an example.

Chapter 9 has been heavily revised and reorganized, with considerable new material on exploiting code bugs, malware, and defenses against them.

Chapter 10, on Linux, is a revision of the old Chapter 10 (on UNIX and Linux). The focus is clearly on Linux now, with a great deal of new material.

Chapter 11, on Windows Vista, is a major revision of the old Chap. 11 (on Windows 2000). It brings the treatment of Windows completely up to date.

Chapter 12 is new. I felt that embedded operating systems, such as those found on cell phones and PDAs, are neglected in most textbooks, despite the fact that there are more of them out there than there are PCs and notebooks. This edition remedies this problem, with an extended discussion of Symbian OS, which is widely used on Smart Phones.

Chapter 13, on operating system design, is largely unchanged from the second edition.

Numerous teaching aids for this book are available. Instructor supplements can be found at *www.prenhall.com/tanenbaum*. They include PowerPoint sheets, software tools for studying operating systems, lab experiments for students, simulators, and more material for use in operating systems courses. Instructors using this book in a course should definitely take a look.

In addition, instructors should examine GOAL (Gradiance Online Accelerated Learning), Pearson's premier online homework and assessment system. GOAL is designed to minimize student frustration while providing an interactive teaching experience outside the classroom. With GOAL's immediate feedback, hints, and pointers that map back to the textbook, students will have a more efficient and effective learning experience. GOAL delivers immediate assessment and feedback via two kinds of assignments: multiple choice Homework exercises and interactive Lab work.

The multiple-choice homework consists of a set of multiple choice questions designed to test student knowledge of a solved problem. When answers are graded as incorrect, students are given a hint and directed back to a specific section in the course textbook for helpful information.

The interactive Lab Projects in GOAL, unlike syntax checkers and compilers, check for both syntactic and semantic errors. GOAL determines if the student's program runs but more importantly, when checked against a hidden data set, verifies that it returns the correct result. By testing the code and providing immediate feedback, GOAL lets you know exactly which concepts the students have grasped and which ones need to be revisited.

Instructors should contact their local Pearson Sales Representative for sales and ordering information for the GOAL Student Access Code and Modern Operating Systems, 3e Value Pack (ISBN: 0135013011).

A number of people helped me with this revision. First and foremost I want to thank my editor, Tracy Dunkelberger. This is my 18th book and I have worn out a lot of editors in the process. Tracy went above and beyond the call of duty on this one, doing things like finding contributors, arranging numerous reviews, helping with all the supplements, dealing with contracts, interfacing to PH, coordinating a great deal of parallel processing, generally making sure things happened on time, and more. She also was able to get me to make and keep to a very tight schedule in order to get this book out in time. And all this while she remained chipper and cheerful, despite many other demands on her time. Thank you, Tracy. I appreciate it a lot.

Ada Gavrilovska of Georgia Tech, who is an expert on Linux internals, updated Chap. 10 from one on UNIX (with a focus on FreeBSD) to one more about Linux, although much of the chapter is still generic to all UNIX systems. Linux is more popular among students than FreeBSD, so this is a valuable change.

Dave Probert of Microsoft updated Chap. 11 from one on Windows 2000 to one on Windows Vista. While they have some similarities, they also have significant differences. Dave has a great deal of knowledge of Windows and enough vision to tell the difference between places where Microsoft got it right and where it got it wrong. The book is much better as a result of his work.

Mike Jipping of Hope College wrote the chapter on Symbian OS. Not having anything on embedded real-time systems was a serious omission in the book, and thanks to Mike that problem has been solved. Embedded real-time systems are becoming increasingly important in the world and this chapter provides an excellent introduction to the subject.

Unlike Ada, Dave, and Mike, who each focused on one chapter, Shivakant Mishra of the University of Colorado at Boulder was more like a distributed system, reading and commenting on many chapters and also supplying a substantial number of new exercises and programming problems throughout the book.

Hugh Lauer also gets a special mention. When we asked him for ideas about how to revise the second edition, we weren't expecting a report of 23 single-spaced pages, but that is what we got. Many of the changes, such as the new emphasis on the abstractions of processes, address spaces, and files are due to his input.

I would also like to thank other people who helped me in many ways, including suggesting new topics to cover, reading the manuscript carefully, making supplements, and contributing new exercises. Among them are Steve Armstrong, Jeffrey Chastine, John Connelly, Mischa Geldermans, Paul Gray, James Griffioen, Jorrit Herder, Michael Howard, Suraj Kothari, Roger Kraft, Trudy Levine, John Masiyowski, Shivakant Mishra, Rudy Pait, Xiao Qin, Mark Russinovich, Krishna Sivalingam, Leendert van Doorn, and Ken Wong.

The people at Prentice Hall have been friendly and helpful as always, especially including Irwin Zucker and Scott Disanno in production and David Alick, ReeAnne Davies, and Melinda Haggerty in editorial.

Finally, last but not least, Barbara and Marvin are still wonderful, as usual, each in a unique and special way. And of course, I would like to thank Suzanne for her love and patience, not to mention all the *druiven* and *kersen*, which have replaced the *sinaasappelsap* in recent times.

Andrew S. Tanenbaum

ABOUT THE AUTHOR

Andrew S. Tanenbaum has an S.B. degree from M.I.T. and a Ph.D. from the University of California at Berkeley. He is currently a Professor of Computer Science at the Vrije Universiteit in Amsterdam, The Netherlands, where he heads the Computer Systems Group. He was formerly Dean of the Advanced School for Computing and Imaging, an interuniversity graduate school doing research on advanced parallel, distributed, and imaging systems. He is now an Academy Professor of the Royal Netherlands Academy of Arts and Sciences, which has saved him from turning into a bureaucrat.

In the past, he has done research on compilers, operating systems, networking, local-area distributed systems and wide-area distributed systems that scale to a billion users. His main focus now is doing research on reliable and secure operating systems. These research projects have led to over 140 refereed papers in journals and conferences. Prof. Tanenbaum has also authored or co-authored five books which have now appeared in 18 editions. The books have been translated into 21 languages, ranging from Basque to Thai and are used at universities all over the world. In all, there are 130 versions (language + edition combinations).

Prof. Tanenbaum has also produced a considerable volume of software. He was the principal architect of the Amsterdam Compiler Kit, a widely-used toolkit for writing portable compilers. He was also one of the principal designers of Amoeba, an early distributed system used on a collection of workstations connected by a LAN and of Globe, a wide-area distributed system.

He is also the author of MINIX, a small UNIX clone initially intended for use in student programming labs. It was the direct inspiration for Linux and the platform on which Linux was initially developed. The current version of MINIX, called MINIX 3, is now focused on being an extremely reliable and secure operating system. Prof. Tanenbaum will consider his work done when no computer is equipped with a reset button. MINIX 3 is an on-going open-source project to which you are invited to contribute. Go to *www.minix3.org* to download a free copy and find out what is happening.

Prof. Tanenbaum's Ph.D. students have gone on to greater glory after graduating. He is very proud of them. In this respect he resembles a mother hen.

Tanenbaum is a Fellow of the ACM, a Fellow of the IEEE, and a member of the Royal Netherlands Academy of Arts and Sciences. He has also won numerous scientific prizes, including:

- the 2007 IEEE James H. Mulligan, Jr. Education Medal
- the 2003 TAA McGuffey Award for Computer Science and Engineering
- the 2002 TAA Texty Award for Computer Science and Engineering
- the 1997 ACM/SIGCSE Award for Outstanding Contributions to Computer
- the 1994 ACM Karl V. Karlstrom Outstanding Educator Award

He is also listed in *Who's Who in the World*. His home page on the World Wide Web can be found at URL *http://www.cs.vu.nl/~ast/*.

CONTENTS