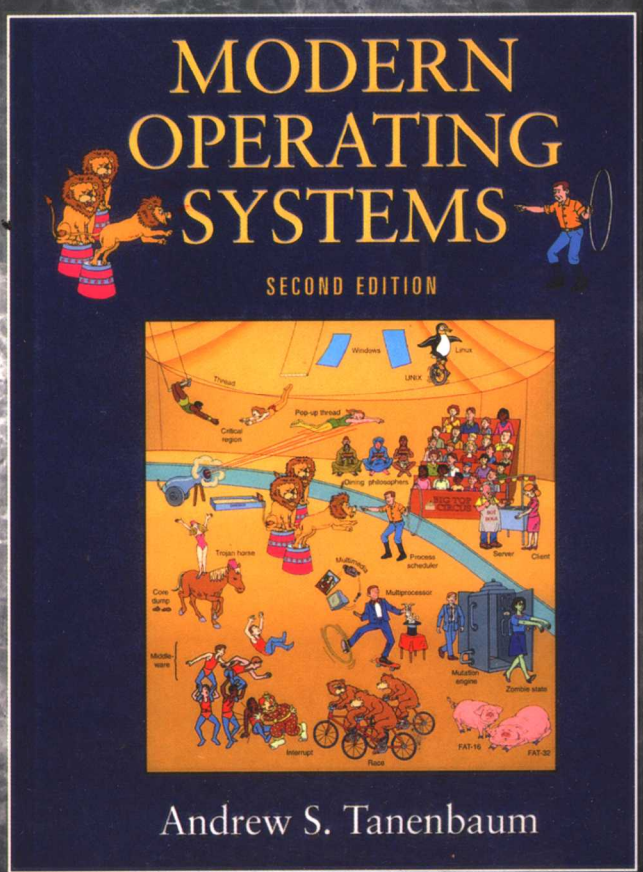
PEARSON Prentice Hall

计 算 机 科 学 丛 书

第2版

现代操作系统

(荷) Andrew S. Tanenbaum 著　陈向群　马洪兵　等译



Modern Operating Systems

Second Edition

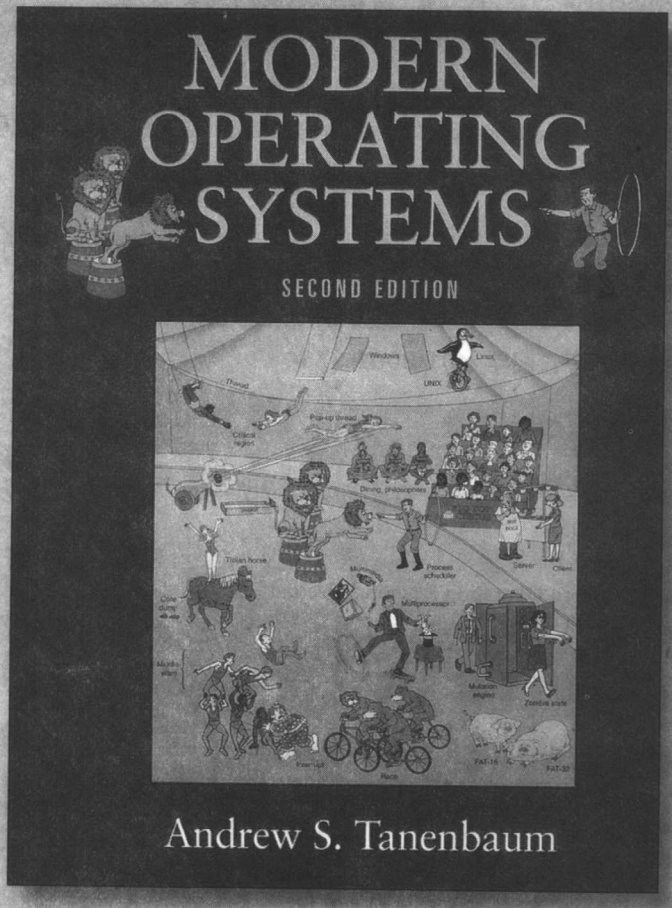
机械工业出版社
China Machine Press

计 算 机 科 学 丛 书

第2版

现代操作系统

（荷）Andrew S. Tanenbaum 著　陈向群　马洪兵　等译



Modern Operating Systems

Second Edition

机械工业出版社
China Machine Press

本书是操作系统领域的经典之作，与第1版相比有较大的变化。书中集中讨论了操作系统的基本原理，除了重点放在单处理机操作系统之外，还包含了有关计算机安全、多媒体操作系统、UNIX、Windows 2000以及操作系统设计等方面的内容。书中涉及的主题包括图形用户界面、多处理机操作系统、笔记本电脑电源管理、可信系统、病毒、网络终端、CD-ROM文件系统、互斥信号量、RAID、软定时器、稳定存储器以及新的页面置换算法等。此外，书中还增加了大量习题，方便教学。

本书适合作为高等院校计算机科学与技术专业操作系统课程教材，也是设计、开发操作系统的重要参考书。

本书版权登记号：图字：01-2001-3754

图书在版编目（CIP）数据

现代操作系统（第2版）/（荷）坦尼鲍姆（Tanenbaum, A. S.）著；陈向群等译. -北京：机械工业出版社，2005. 6

（计算机科学丛书）

书名原文：Modern Operating Systems, Second Edition

ISBN 7-111-16511-X

Ⅰ. 现… Ⅱ. ① 坦… ② 陈… Ⅲ. 操作系统 Ⅳ. TP316

中国版本图书馆CIP数据核字（2005）第046669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：杨海玲

北京瑞德印刷有限公司印刷 · 新华书店北京发行所发行

2005年6月第2版第1次印刷

787mm × 1092mm 1/16 · 35.5印张

印数：0 001-5000册

定价：55.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线（010）68326294

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，还揭橥了学术的源变，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、从业人员较少的现状下，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图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray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、研究及庋藏。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，有的还专诚为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近百个品种，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，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步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步入一个新的阶段。为此，华章公司将加大引进教材的力度，在“华章教育”的总规划之下出版三个系列的计算机教材：除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这三套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

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，但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。教材的出版只是我们的后续服务的起点。华章公司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：

电子邮件：hzedu@hzbook.com
联系电话：(010) 68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周立柱	周克定	周傲英	孟小峰	岳丽华
范　明	郑国梁	施伯乐	钟玉琢	唐世渭
袁崇义	高传善	梅　宏	程　旭	程时端
谢希仁	裘宗燕	戴　葵		

译 者 序

操作系统是计算机科学与技术专业的一门专业基础课，是该专业学生的必修课程。在学习过程中，教材起着相当大的作用。因此，要学好这门课，首先应该选择一本好的教材。

本书的作者Andrew S. Tanenbaum是一位经验丰富的教授，从事计算机教学多年，在操作系统、计算机网络方面颇有研究，出版了多本教材。Andrew S. Tanenbaum教授的每一本教材都受到广泛的好评。

作为计算机技术的重要基础学科，操作系统近年来在概念和技术上都有很快的发展。操作系统的教材当然也应该及时反映这种发展。而本书正是作者Andrew S. Tanenbaum这种努力的成果。

本书与第1版相比有较大的变化。第2版集中讨论了操作系统的基本原理，不再用很大的篇幅介绍分布式操作系统。书中新添加了有关计算机安全、多媒体操作系统、Windows 2000以及操作系统设计等多章。本书的另一个重要特色是在许多章节内都介绍了一些相关研究工作的参考文献，供感兴趣的读者参考。另外，书中附有大量的习题，其中相当一部分习题是新的或重新编写过的。

本书体系完整、结构合理、内容丰富、叙述清晰、适用面广，美国许多重点大学的计算机软件及相关专业都采用本书作为教材或主要教学参考书。

通过本书的学习，相信学生一定会对操作系统的功能、实现技术有全面的了解。

本书在翻译的过程中，得到很多人的支持帮助。参加翻译、校对、审阅工作的还有张乃琳、殷晓田、康洁刚、顿楠、涂欣、朱伟、王会娣、钟诚、胡建钧、王俊、余啸海、欧阳图、陈国辉、刘晓敏、黄亚峰、胡新婕、徐冬、王悦、张晓薇、戴蓉、滕启明、赵霞、雷志勇、黄晓晨、李锴。

特别感谢机械工业出版社华章分社的杨海玲编辑，她的辛勤工作使得本书得以顺利完成。在此致以衷心的感谢。

本书适合作为高等院校计算机科学与技术专业和电子工程类相关专业的操作系统课程教材，也是一本用于设计、开发操作系统的重要参考书。

由于译者的水平有限，译文可能未能完全表达出原书作者的意图，希望广大专家和读者提出宝贵意见。

译 者

2005年2月

前　言

自从本书第1版在1992年面世以来，整个世界已经发生了巨大的变化。各种计算机网络和分布式系统已经非常普遍。现在，儿童们在因特网上漫游，而过去只有计算机专业人员才可能使用因特网。因此，这本书也必须有较大的修改。

最明显的变化是，在第1版中约有一半内容是有关单处理机操作系统的，而另一半内容则是有关分布式系统的。在1991年我选择这样编排是因为当时几乎没有大学开设分布式系统的课程，要想让学生们学习分布式系统，有关内容就不得不安排在操作系统课程中，本书也正是为了这一意图。现在，多数大学安排有单独的分布式系统课程，所以就没有必要再把这两个主题放在同一门课程和同一本书中了。本书是为操作系统的第一门课程编写的，所以，内容主要集中于传统的单处理机系统。

我已经合作出版了另外两本有关操作系统的书籍，这样就有两种可能的课程安排。

面向实践的课程序列：

1)《操作系统设计与实现》，Tanenbaum与Woodhull。

2)《分布式系统》，Tanenbaum与Van Steen。

传统的课程序列：

1)《现代操作系统》，Tanenbaum。

2)《分布式系统》，Tanenbaum与 Van Steen。

第一个课程序列采用了MINIX，要求学生在有关的实验室中进行MINIX实验作为第一门课程的补充。后一个课程序列不使用MINIX，代之以一些小型模拟程序，安排在使用本书的首个课程里用于学生练习。这些模拟程序可以在作者的Web页面www.cs.vu.nl/~ast/上，通过点击Software and supplementary material for my books得到。

本书除了重点放在单处理机操作系统之外，其他的主要修改包括添加了有关计算机安全、多媒体操作系统以及Windows 2000的几章内容，这些都是重要而适时的主题。另外，还增加了有关操作系统设计的新的单独一章。

其他新的特色是，在许多章内安排了与该章主题相关的研究状况的小节。这样做的目的是向读者介绍有关进程、存储管理等领域中的当前工作。这些小节中有大量的涉及当前研究工作的参考文献，供感兴趣的读者参考。除此之外，在第13章中列出了许多入门和辅导性的参考文献。

最后，在本书中已经添加了大量的主题或者对原有主题进行了重大的修改。这些主题包括：图形用户界面、多处理机操作系统、笔记本电脑的电源管理、可信系统、病毒、网络终端、CD-ROM文件系统、互斥信号量、RAID、软定时器、稳定存储器、公平分享调度以及新的页面置换算法等。书中还添加了许多新的习题，并更新了旧的习题。习题的总数超过了450个。另外，为了使本书跟上时代，增加了约250个新的参考文献。

尽管删掉了400多页陈旧内容，但是由于添加了大量的新内容，所以本书的页数反而增加了。不过，本书依然适用于一学期的课程。对于多数大学里的半学期课程而言，本书内容就可能太多了。正因为如此，作者把本书设计成模块方式。任何有关操作系统的课程都应该包括第1章到第6章。这些内容是每一个学生必须了解的基本内容。

如果教学上还有时间，可以讲授其他章节。这些章节均假设读者已经学过了第1章到第6章，不过从第7章到第12章是自包容的，所以可以按任何次序使用它们的任何子集，这些都取决于授课教师的兴趣。作者的意见是，第7章到第12章远比前面的章节更有趣。不过教师应该告诉学生们，在得到双倍的“餐后巧克力甜点”之前，他们必须先吃掉“花椰菜”。

感谢所有那些评阅手稿的朋友们：Rida Bazzi、Riccardo Bettati、Felipe Cabrera、Richard Chapman、

John Connely、John Dickinson、John Elliott、Deborah Frincke、Chandana Gamage、Robbert Geist、David Golds、Jim Griffioen、Gary Harkin、Frans Kaashoek、Mukkai Krishnamoorthy、Monica Lam、Jussi Leiwo、Herb Mayer、Kirk McKusick、Evi Nemeth、Bill Potvin、Prasant Shenoy、Thomas Skinner、Xian-He Sun、William Terry、Robbert Van Renesse和Maarten van Steen。Jamie Hanrahan、Mark Russinovich和Dave Solomon等人在Windows 2000方面的知识极为丰富，对本书也大有帮助。特别感谢Al Woodhull的极有价值的评阅以及他对各章后面许多新习题的思考。

我的学生们也给予了有益的评论和反馈，特别感谢Staas de Jong、Jan de Vos、Niels Drost、David Fokkema、Auke Folkerts、Peter Groenewegen、Wilco Ibes、Stefan Jansen、Jeroen Ketema、Joeri Mulder、Irwin Oppenheim、Stef Post、Umar Rehman、Daniel Rijkhof、Maarten Sander、Maurits van der Schee、Rik van der Stoel、Mark van Driel、Dennis van Veen和Thomas Zeeman。

Barbara和Marvin像往常一样，保持着各自独特的美妙方式。最后，感谢Suzanne对其工作的热爱和耐心。

Andrew S. Tanenbaum

目　录

第1章 引 论

现代计算机系统由一个或多个处理器、主存、磁盘、打印机、键盘、网络接口以及其他输入/输出设备组成。一般而言，现代计算机系统是一个复杂的系统。编写监督上述所有部件的程序并正确地使用它们，加以优化，是一项非常困难的工作。所以，计算机配备了一个称为*操作系统*的软件层，它的任务是管理所有这些设备，并为用户程序提供一个较为简单的到硬件的接口。本书的主题就是操作系统。

在图1-1中给出了操作系统的定位。图的底部是硬件，在很多情形下硬件可分为两层或更多层。最底层是物理设备，包括集成电路芯片、连线、电源、阴极射线管以及类似的设备等。至于这些设备是如何构造以及如何工作的，则是电子工程师的职责范围。

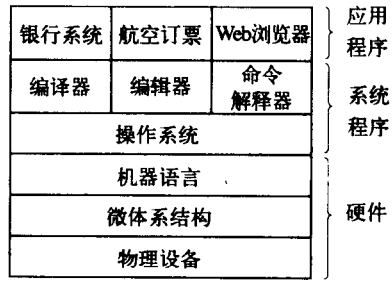


图1-1 包括硬件、系统程序以及应用程序的计算机系统

接着是*微体系结构层*，其中的物理设备分组构成了功能单元。在这层中有CPU（Central Processing Unit）的专用内部寄存器以及包含算术逻辑单元的数据通道。在每个时钟周期，CPU从寄存器中取出一个或两个操作数，并在算术逻辑单元中进行运算（如加法或逻辑与）。其结果存储在一个或多个寄存器中。在有些机器中，数据通道的操作由称为*微程序*的软件控制。在另外一些机器中，相关的操作由硬件电路直接控制。

设立数据通道的目的是执行某些指令集。其中一些指令可在数据通道的一个周期中执行；其他的指令也许需要多个数据通道周期。这些指令可以使用寄存器或其他硬件设施。综合起来，相关的硬件以及那些对汇编语言程序员可见的指令，构成了*指令集体系结构*（Instruction Set Architecture, ISA）层。这一层通常被称为*机器语言*。

典型的机器语言有50～300条指令，大多数指令在机器里从事数据传送、算术运算和值比较等操作。在这个层次上，可以通过向特定的*设备寄存器*（device register）写入值来控制对应的输入/输出设备。例如，可通过把磁盘地址、主存地址、字节数和操作类型（读/写）等值装入特定的寄存器来完成硬盘读操作。实际上，一个操作往往需要很多的参数，而操作完成后设备返回的状态也非常复杂。进一步说，对许多输入/输出（I/O）设备而言，在程序设计中，时序的作用是非常重要的。

为了隐藏设备操作的复杂性，使用了操作系统。操作系统包括一个（专门）隐藏这些硬件的软件层，并且给程序员提供一套使用更为便利的指令集。举例来说，在概念上，“从文件读数据块”比考虑“移动磁头，等待放置”之类的细节来得简单。

在操作系统的顶层是其他系统软件。其中有命令解释器（shell）、编译器、编辑器以及类似的独立于应用的程序。重要的是，尽管这些程序通常由计算机厂商提供，但是它们本身并不是操作系统的组成部分。这一点很重要，也很微妙。操作系统通常是专指在*核心态*（kernel mode）或称*管态*（supervisor mode）下运行的软件，它受硬件保护以免遭用户的篡改（这里不考虑某些老式的微处理器，它们根本没有硬件的保护）。编译器和编辑器运行在*用户态*（user mode）下。如果用户不喜欢某一个编译器，他尽可以自己重写一个，但是用户却不可以自行编写一个时钟中断处理程序，因为这是操作系统的一部分，它通常由硬件保护，以防止用户试图对它进行修改。

然而，有时在嵌入式系统（该系统没有核心态）或解释系统（如基于Java的操作系统，它采用解释方式而非硬件方式区分组件）中，上述区别是模糊的。但是对于传统的计算机而言，操作系统仍旧运行在核心态中。

这就说明了在许多系统中，一些在用户态下运行的程序协助操作系统完成特权功能。例如，经常有

一个程序供用户修改其口令之用。但是这个程序不是操作系统的一部分，也不在核心态下运行，不过它明显实现敏感的功能，并且必须以某种方式给予保护。

在某些系统中，这种想法被推向了极致，一些传统上被认为是操作系统的部分（如文件系统）在用户空间中运行。在这类系统中，很难划分出一条明显的界限。在核心态中运行的当然是操作系统的一部分，但是一些在核心态外运行的程序也有争议地被认为是操作系统的一部分，或者至少与操作系统密切相关。

最后，在系统程序的上层是应用程序。这些程序是用户买来的或者自行编写的，用于解决用户特定的问题，如文字处理、电子表单、工程计算或在数据库中存储信息等。

1.1 什么是操作系统

多数计算机用户都有一些使用操作系统的体验，但要给出操作系统的准确定义却很困难。部分原因是操作系统执行两个相对独立的任务，即扩展机器和管理资源，另外，还取决于从什么角度看待操作系统，读者多半听说过其中一个或另一个的功能。下面我们逐项进行讨论。

1.1.1 作为扩展机器的操作系统

如前所述，在机器语言一层上，多数计算机的体系结构（指令集、存储组织、I/O和总线结构）是
3 很原始的，而且编程是很困难的，尤其是输入/输出操作。要更具体地考查这一点，可以简要地考虑如何用NEC PD765控制器芯片来进行软盘I/O操作，多数基于Intel的个人计算机中使用了该控制器芯片。PD765有16条命令，每一条命令向一个设备寄存器装入长度从1字节到9字节的特定数据。这些命令用于读写数据、移动磁头臂、格式化磁道，以及初始化、检测状态、复位和校准控制器及设备等。

最基本的命令是read和write。它们均需要13个参数，所有这些13个参数被封装在9个字节中。这些参数所指定的信息有：欲读取的磁盘块的地址、每磁道的扇区数、物理介质的记录格式、扇区间隙以及对已删除数据地址标识的处理方法等。如果读者不理解这些“故弄玄虚”的语言，请不要担心；因为这正是关键所在——它们太玄秘了。当操作结束时，控制器芯片在7个字节中返回23个状态及出错字段。这样似乎还不够，软盘程序员还要注意保持电机的开关状态。如果电机关闭着，则在读写数据前要先启动它（有一段较长的启动延迟）。而电机又不能长时间处于开启状态，否则软盘片就会被磨坏了。程序员必须在较长的启动延迟和可能对软盘造成损坏（和丢失数据）之间做出权衡。

现在不用再叙述实际细节了，很显然，一般程序员并不想涉足软盘（也包括硬盘，它很复杂且与软盘不同）编程的这些具体细节。程序员需要的是一种简单的、高度抽象的处理。在磁盘的情况下，典型的抽象是包含了一组已命名文件的一个磁盘。每个文件可以被打开进行读写，然后读写完毕，最后被关闭。诸如记录是否应该使用修正的调频记录方式，以及当前电机的状态等细节，不应该出现在提供给用户的抽象描述中。

为程序员隐藏硬件的实际细节，并提供一个可以读写的、简洁的命名文件视图的程序，当然是操作系统。正是操作系统为程序员屏蔽了磁盘硬件，并提供了一个简单的、面向文件的接口，操作系统还隐藏了大量与中断、定时器、存储管理以及其他与低层特征有关的令人烦恼的细节。无论在哪种情况中，操作系统所提供的抽象都比底层硬件所能提供的更简单和更易于使用。

从上述角度看，操作系统的作用是为用户提供一台等价的扩展机器（extended machine）或称虚拟机（virtual machine），它比底层硬件更容易编程。至于操作系统是如何实现这一目标的，则说来话长，本书中将自始自终研究其细节。概括起来，操作系统提供各种类型的服务，程序可以通过使用称为系统
4 调用的特殊指令来得到这些服务。在本章的后面将考查一些较为常用的系统调用。

1.1.2 作为资源管理者的操作系统

把操作系统看作是向用户提供基本的方便接口的概念，是一种自顶向下的观点。按照另一种自底向上的观点，操作系统则用来管理一个复杂系统的各个部分。现代计算机包含处理器、存储器、计时器、磁盘、鼠标、网络接口、打印机以及许多其他设备。从这个角度看，操作系统的任务是在相互竞争的程序之间有序地控制对处理器、存储器以及其他I/O设备的分配。

假设在一台计算机上运行的三个程序试图同时在同一台打印机上输出计算结果，那么开始的几行可

能是程序1的输出，接着几行是程序2的输出，然后又是程序3的输出等等，最终结果将是一团糟。采用将打印结果送到磁盘上缓存的方法，操作系统可以把潜在的混乱有序化。在一个程序结束后，操作系统可以将其输出从其暂存的磁盘文件送到打印机输出，同时其他程序可以继续产生更多的输出结果，很明显，这些程序的输出还没有真正送至打印机。

当一个计算机（或网络）有多个用户时，管理和保护存储器、I/O设备以及其他资源的需求变得强烈起来，因为用户可能会互相干扰。另外，用户通常不仅共享硬件，还要共享信息（文件、数据库等）。简而言之，操作系统的这一种观点认为，操作系统的主要任务是记录使用资源的情况、对资源请求进行授权、计算使用费用，并且为不同的程序和用户协调互相冲突的资源请求。

资源管理包括用以下两种方式实现的复用（共享）资源：在时间上复用和在空间上复用。当一种资源时间复用时，不同的程序或用户轮流使用它。先是第一个获得资源的使用，然后下一个，以此类推。例如，若在系统中只有一个CPU，而多个程序需要在该CPU上运行，操作系统则首先把该CPU分配给某一个程序，在它运行了足够长的时间之后，另一个程序得到CPU，然后是下一个，如此进行下去，最终，第一个程序轮到再次运行。至于资源是如何实现时间复用的（谁应该是下一个以及运行多长时间等等）则是操作系统的任务。还有一个有关时间复用的例子是打印机的共享。当多个打印作业排队等待在一台打印机上打印时，必须决定下一个将被打印的是哪个作业。

另一类复用是空间复用。每个顾客都得到资源的一部分，从而取代了顾客排队。例如，通常在若干运行程序之间分割主存，这样每一个运行程序都可同时驻留在内存（例如，为了轮流使用CPU）。假设有足够的内存可以存放多个程序，那么在内存中同时存放若干个程序的效率，比把所有内存都分给一个 5
程序的效率要高得多，如果这一个程序只需要整个内存的一小部分时，结果更是这样。当然，如此的做法会引起公平、保护等问题，这有赖于操作系统解决它们。有关空间复用的其他资源还有（硬）磁盘。在许多系统中，一个磁盘同时为许多用户保存文件。分配磁盘空间并记录谁正在使用磁盘块则是操作系统资源管理的典型任务。

1.2 操作系统的历史

操作系统已经演进了许多年了。在下面的几节中，我们将简要地考查一些操作系统历史上的重要之处。因为操作系统与其所运行的计算机体系结构的联系非常密切，所以我们将考查连续几代的计算机，看看它们的操作系统是什么样的。把操作系统的分代映射到计算机的分代上有些粗糙，但是这样做确实有某些作用，否则还没有好办法说明清楚操作系统的历史。

第一台真正的数字计算机是英国数学家Charles Babbage（1792—1871）设计的。尽管Babbage花费了他几乎一生的时间和财产试图建造他的“分析机”，但是他始终未能让机器正常的运转，因为它是纯机械的，他所在时代的技术不能生产出他所需要的高精度轮子、齿轮和轮牙。毫无疑问，这台分析机没有操作系统。

有一段有趣的历史花絮，Babbage认识到他的分析机需要软件，所以他雇佣了一个名为Ada Lovelace的年轻妇女作为世界上第一个程序员，而她是著名的英国诗人Lord Byron的女儿。程序设计语言Ada则是为她命名的。

1.2.1 第一代（1945～1955）：真空管和插件板

从Babbage失败之后一直到二次世界大战，数字计算机的建造几乎没有什么进展。20世纪40年代中期，哈佛大学的Howard Aiken、普林斯顿高等研究院的John von Neumann（冯·诺依曼）、宾夕法尼亚大学的J. Presper Eckert和William Mauchley、德国的Konrad Zuse还有其他人都成功地建造了计算机器。第一批机器使用机械继电器，但是非常慢，运算周期以秒计算。继电器后来被真空管代替。这些机器极为巨大，有成上万个真空管，充满了整个房间，但仍旧百万倍地慢于现在最便宜的个人计算机。

在那个年代里，同一个小组的人设计、建造、编程、操作并维护一台机器。所有的程序设计是用纯粹 6
的机器语言完成的，常常连线到插件板上以便控制机器的基本功能。没有程序设计语言（甚至汇编语言也没有），操作系统也从来没有听说过。通常的操作的方式是程序员提前在墙上的机时表上预约一段时间，然后到机房中将他的插件板插到计算机里，在接下来的几小时里计算自己的题目，期盼着在这段时间中，二