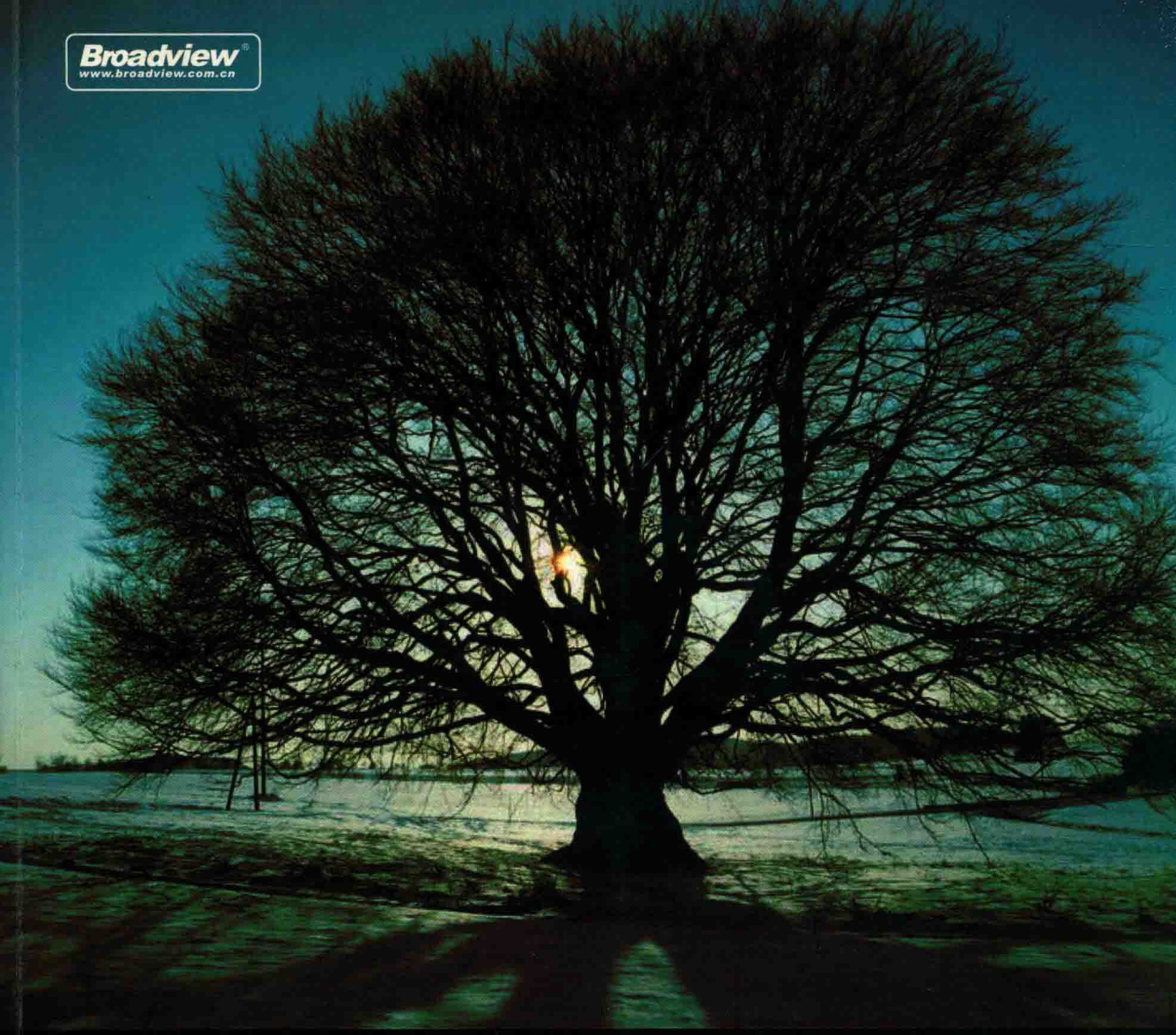
Broadview
www.broadview.com.cn



Windows 内核情景分析

——采用开源代码ReactOS

（上册）

毛德操 著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Windows 内核情景分析

——采用开源代码ReactOS

（上册）

毛德操 著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

本书通过分析具有“开源 Windows”之称的 ReactOS 的源代码，介绍 Windows 内核各个方面的结构、功能、算法与具体实现。全书从“内存管理”、“进程”、“进程间通信”、“设备驱动”等 14 个方面进行分析介绍。本书的写作仿照作者广受欢迎的《Linux 内核源代码情景分析》一书，所有的分析都有 ReactOS 的源代码（以及部分由微软公开的源代码）作为依据，使读者能深入、具体地理解 Windows 内核的方方面面，也使读者的软件开发能力和水平得到提高。

本书可以供大学有关专业的高年级学生和研究生用作教学参考，也可供广大的软件工程师，特别是从事系统软件研发的工程师用于工作参考或进修教材。

图书在版编目（CIP）数据

Windows 内核情景分析：采用开源代码 ReactOS. 上册 / 毛德操著. —北京：电子工业出版社，2009.5
ISBN 978-7-121-08114-9

I. W… II. 毛… III.窗口软件，Windows－程序设计 IV. TP316.7

中国版本图书馆 CIP 数据核字（2009）第 005799 号

责任编辑：朱沭红
印　　刷：北京东光印刷厂
装　　订：三河市皇庄路通装订厂
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：787×1092　1/16　　印张：93.25　　字数：2395 千字
印　　次：2009 年 9 月第 2 次印刷
印　　数：3001～4500 册　　定价：190.00 元（上、下册）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

序

多年来，Windows 垄断了中国桌面操作系统，中国学校中也开设了很多 Windows 课程，但令人遗憾的是，由于 Windows 的源代码不开放，这些课程往往只能使学生了解 Windows 的外特性和操作方法，却不能使学生了解和掌握 Windows 的内核。

有人会问：难道中国人真的需要深入到操作系统的内核，去掌握它的核心技术吗？有人说：微软花了上千亿美元开发出 Windows 操作系统，中国人花 1000 元人民币就能买到，为什么还要自己去研究和开发呢？一些外国咨询公司也为中国政府支招，如麦肯锡公司在 2002 年所做的《中国软件产业发展战略研究报告》就主张中国不必发展操作系统，而应像印度那样，主要发展面向出口的离岸外包业务。这样的观点前些年在中国相当流行，影响了不少人。

2008 年 10 月，有一个事件震惊了中国：微软宣称将对其认为是使用盗版 Windows 和 Office 的电脑实行了“黑屏”。用户面对“黑屏”束手无策，这才意识到，自己的电脑被微软操控了！中国信息安全的软肋由此暴露无遗。

事实上，在信息安全方面我国历来要求使用“自主可控”的软件和硬件，“黑屏”事件更使广大用户有了切肤之痛，认识到自主可控的重要性。现在，世界上许多国家也有了这种共识，如俄国、欧洲和拉丁美洲的一些国家都在发展基于开源软件 Linux 的自主操作系统。在这方面，目前中国的重大进展是起动了“核高基”重大专项，这是按照《国家中长期科学和技术发展规划纲要（2006—2020 年）》所部署的，旨在发展“核心电子器件、高端通用芯片及基础软件产品”。其中的“基础软件产品”中最重要的就是操作系统。这样，持续多年的中国要不要发展自主操作系统的争议终于有了定论。通过“核高基”专项，中国有望在今后三个五年计划内实现自主操作系统的产业化，将来中国的信息系统再不会被人“黑屏”了。

当然，中国发展自主操作系统的意义不仅仅是为了保障信息安全。作为一个有 13 亿人口的大国，中国拥有自主操作系统将产生巨大的经济效益，并带动其下游的整个软件和信息服务业的发展。

虽然当前以 Linux 为代表的开源软件为中国发展自主操作系统提供了很好的支撑，但学习和借鉴 Windows 也有重要价值。在这个时候，毛德操先生所著的《Windows 内核情景分析》出版了，它为广大读者打开了通向 Windows 内核的大门。

正如作者所说，“对于操作系统内核这么复杂的软件，是一定要结合具体的代码（哪怕只是用来描述算法的伪代码）才能说清楚的”。在 Windows 源代码不开放的情况下，作者不得不主要地基于开源的 ReactOS 来作分析。ReactOS 被誉为“开源 Windows”，它的目标是研发一个开源的 Windows 内核，这对于打破微软的技术垄断很有好处。我们认为，在当前的实际条件下，作者基于 ReactOS

源代码并辅以部分由微软公开的源代码来编写本书，是一条切实可行的途径。

与其说本书是一部单纯的学术著作，不如说这更是一篇实践经验的总结。本书正是毛德操先生近年来领导开发“兼容内核（Unified Kernel）”项目的一篇经验总结。早在 2004 年毛先生就提出了开发“兼容内核”的倡议，他提出开发一个既能支持 Linux 应用软件运行、也能支持 Windows 应用软件运行的内核，这与开源项目 Wine 有异曲同工之妙。Wine 是通过在 Linux 内核的外面加一个适配层，使得 Windows 应用软件的二进制代码可以直接在 Linux 内核上运行，但这样做难免带来性能的下降，或难以做到完全兼容。相比之下，“兼容内核”采取了更为直接的技术途径，也有可能做得更好。

不过，这些年这个倡议也受到了不少质疑。有人认为没有必要，有人怀疑做不出来，有人认为缺乏学术价值等等。尽管这些质疑似乎也有这样那样的道理，但如果为广大用户着想，那么谁也不能否认“兼容内核”的价值——一个既能运行 Windows 应用软件又能运行 Linux 应用软件的操作系统，而且又是低价的、自主可控的，这无疑是广大用户的福音！

应当感谢国家发改委和浙江省科技厅支持了“兼容内核”这个开源项目。在中国，像他们这样勇于支持软件业自主创新的行动还不是很多，因而值得大书特书。正是通过领导“兼容内核”的开发工作，作者取得了剖析 Windows 内核的实践经验，因此这本书将能很好地指导操作系统和其他许多软件的开发实践。

本书是毛德操和胡希明先生所著的《Linux 内核源代码情景分析》的姊妹篇。前一篇在短短的时间里印刷了四次，深受广大读者欢迎，因此可以预见，本书也将收到类似于前一篇的欢迎程度。

中国工程院院士

倪光南

2009 年 3 月 9 日

前　言

世上有这么一些人，这些人对于感兴趣的事物绝不满足于仅仅知其然，而非得要知其所以然才能舒服。这些人里面，如果感兴趣的是计算机操作系统、特别是 Windows 操作系统，那么从这本书里应该能获得对于许多问题的答案。其实，笔者本人也是这个人群中的一员，笔者阅读、分析过 Linux 内核的源代码，自然就也很想读一下 Windows 内核的源代码，把 Windows 内核搞搞清楚。以笔者之见，对于像操作系统内核这么复杂的软件，想要搞搞清楚，就非得要深入到程序代码中去不可。事实上，笔者有这个愿望已经很多年了，可是 Windows 内核的代码是不公开的，即便是退而求其次，要找一些深入介绍 Windows 内核的书籍，也是寥寥无几，而且看了以后也往往不得要领，觉得实在是语焉不详甚至模棱两可。确实，与程序代码相比，任何自然语言都只能说是模糊而不确切的。幸而现在有了 ReactOS 这个开源项目，这个项目的目标是要研发出一个开源的 Windows 内核。该项目的参与者们想必对 Windows 内核下了很深的功夫，在代码中极力模仿 Windows，力求忠实于 Windows，然而却又是自己的实现。读着 ReactOS 的代码，笔者常常回想起一些著作中的有关章节或片段，以前看的时候只好不求甚解，现在看到代码才真的明白了。由此又生出感慨，对于操作系统内核这么复杂的软件，是一定要结合具体的代码（哪怕只是用来描述算法的伪代码）才能说清楚的。把内核的代码封锁起来不让人研究，实际上是对于人类的“知的权利”的蔑视。而对于 ReactOS 代码的作者们，笔者则一来是感激，二来是佩服和尊敬，进而觉得应该把自己的理解和体会写出来与读者分享。本书所引的代码基本上都出自 ReactOS 的 0.3.3 版，读者可以自行下载一份代码，结合本书加以阅读。

与 ReactOS 密切相关的另一个开源项目是 Wine，这个项目的宗旨是在 Linux 内核的外面做上一个适配层，由一个服务进程和一些动态连接库相结合构成的适配层，使得 Windows 应用软件的二进制代码可以直接（不经过移植和重新编译就）在 Linux 内核上运行。一言以蔽之，就是“核内差异核外补”。Windows 应用软件本来是要在 Windows 内核上运行的，而 Windows 内核与 Linux 内核显然有着不小的差异，Wine 的目的就是在核外（用户空间）加以补偿和虚拟，使 Windows 应用软件得以在 Linux 内核上运行。在某种意义上，这甚至比 ReactOS 要做的更难，因为这是要在一个不同的基础上、不同的环境中重构一个虚拟的 Windows 内核，没有对于 Windows 内核的真正深刻的理解，这显然是不可能的。笔者曾听到（看到）一些对 ReactOS 有所怀疑的说法，说是 ReactOS 的人怎么能对 Windows 内核理解得那么深刻？是不是他们拿到了 Windows 内核的源代码，而只是在依样重画一遍葫芦？Wine 的存在和成功（也许是部分的成功）正好回答了这个问题，正好可以作为佐证。要说对于 Windows 内核的理解，Wine 的作者们丝毫不比 ReactOS 的作者们差，而 Wine 的代码与 Windows 内核却相去甚远、泾渭分明。既然 Wine 可以一行行代码从头写来，ReactOS 又有何不可？

不过，笔者以为，与其研发一个开源的 Windows 内核，还不如把 Linux 内核改造成一个“兼容内核（Unified Kernel）”，一个既能支持 Linux 应用软件运行，也能支持 Windows 应用软件运行的内核。这对于 Linux 操作系统的普及有着莫大的好处。因为许多用户已经习惯了 Windows 操作系统和 Windows 应用软件的使用，要使这些用户改用 Linux 操作系统，就得为其提供一种平滑过渡的方案，最重要的是使用户可以继续使用那些已经在上面投入了种种资源，也已经习惯了的 Windows 应用软件。虽然 Wine 在功能上也能起到相似或者基本相同的作用，但是在核外通过服务进程补偿内核差异的做法难免带来性能的下降，要避免性能的下降，就得在内核中补偿内核的差异。笔者的这个主张得到了浙大网新科技股份有限公司的支持，并为此组建了一个研发兼容内核的团队，后来还得到了国家发改委和浙江省科技厅的项目支持。作为一个开源项目，兼容内核的开发目前正在进行之中。笔者由衷地感谢浙大网新科技股份有限公司的支持。此外，无论是兼容内核的开发还是本书的写作，笔者都得到了中国开源软件促进会陆首群主席和倪光南院士的鼓励和支持，在此一并致谢。

所以，驱使笔者写作本书的动力不仅仅是对 Windows 内核的研究和介绍，更多地还来自开发兼容内核的需要。从某种意义上说，本书是兼容内核项目的副产品；如果没有兼容内核项目，恐怕就不会有这本书。事实上，起初电子工业出版社向笔者约稿时，笔者的打算是写一本五六百页的 Windows 与 Linux 的比较研究。可是，开始写了以后就觉得不妥，因为既然有了“Linux 内核源代码情景分析”，就不宜再重复那本书中写过的内容，而应该把篇幅都集中在 Windows 上。另一方面，更为重要的是，参加兼容内核研发的人对于 Linux 内核都是比较了解的，而对于 Windows 内核则往往所知甚少，因而迫切需要有一本对于 Windows 内核的情景分析。于是本书的写作计划一改再改，篇幅愈来愈大，时间愈拖愈长，最后历时三年才完成了本书的写作，而篇幅已经达到了一千多页。为此，笔者要特别感谢电子工业出版社的朱沭红、白涛两位编辑，感谢她（他）们的耐心和热情，更感谢她（他）们对本书的精心编辑。

在本书的写作过程中，微软开放了一个缩微的“Windows 研究性内核（Windows Research Kernel，即 WRK）”的源代码，供高校用于教学目的。就微软而言，这当然是个进步，值得欢迎；但是这并不说明 WRK 是开源的。首先，WRK 的公开范围只是高校，而并不面向公众，其许可证中明文规定分发的范围只是“within your educational institution”。而且，许可证虽然允许在文章、书籍中引用代码中的片段，却规定“The total amount of source code in each of your snippets should not exceed 50 lines”，即每个片段的长度不能超过 50 行。更重要的是，WRK 其实只是专供教学用的模型，而不是实用意义上的操作系统。笔者认为，实际上 WRK 离 Windows 比 ReactOS（离 Windows）更远。这样，即便不考虑许可证所加的限制，如果采用 WRK 的代码写作本书，则本书一半以上的篇幅恐怕就不能存在了。这样考虑下来，笔者决定还是采用 ReactOS 的代码。

然而，ReactOS 又毕竟不是 Windows 本身，并且 Windows 的有些功能和机制（例如文件系统中的 Notify 机制）在 ReactOS 中迄今尚未实现。所以，笔者衷心希望有朝一日微软会公开整个 Windows 内核的代码，到那时候，如果各方面的条件允许的话，笔者愿意再来写一本“正宗”的 Windows 内核源代码情景分析。

当然，我们之所以要研究 Windows 内核，并不单纯是为了满足知的欲望。操作系统内核在整个软件产业链中处于上游乃至源头的位置，对操作系统内核的深入理解对于其他软件的开发，特别是

系统软件的开发有着重要的影响。我的朋友胡希明教授常常用中医和西医来比喻两种软件开发的模式，中医当然也能治病，但是有些病却只能采用西医的方法才能奏效。而西医的最根本的基础则是对于人体的解剖。实际上，现在中医也要拍片透视照 X 光了。

搞电影的人说电影是遗憾的艺术，因为一旦拍成就不能再改了，所以常常留下遗憾。其实，留下遗憾的工作又岂止拍电影。本书尚未付印，但是笔者已经有了遗憾，有些内容本来是应该加以研究并写入本书的，但是笔者的时间和精力已经不允许了。对于 Windows 内核这么复杂和庞大的系统，笔者自觉不可能对每一个细节的理解都能正确，所以本书中谬误之处在所难免，只是笔者再没有时间和精力细细去抠了，这也是令人遗憾的。

毛德操

2009 年 4 月 28 日

于浙大网新科技股份有限公司办公楼

目　录

上　册

下　册

第1章 概述

1.1 Windows 操作系统发展简史

20 世纪 80 年代后期，当时的 UNIX 操作系统已经发展得相当成熟，例如在人机界面上已经有了 X Window，已经开始使用鼠标器，也已经有了“客户机/服务器（Client/Server）”的结构模式。当时的一些“工作站”实际上已经具备了现代桌面应用所需的大多数基本要素。同时，由于 PC 功能与性能的日渐提高，当时甚至已经有人将 UNIX 移植到 PC 上，例如当时的 Xenix 就是 PC 上的 UNIX。而微软为 PC 开发的操作系统，则还是采用命令行模式的 DOS。相比之下，当时微软的产品无论从哪一个角度说都还只能是下里巴人，与阳春白雪的 UNIX 不可同日而语。然而，那时的绝大部分公众对于计算机基本上相当于文盲，对于计算机应用的要求也还很低，所以下里巴人的 DOS 恰好获得了“和者众”的效果，而阳春白雪的 UNIX 则反倒不免曲高和寡。然后，到了 90 年代前期，在 UNIX 这一边有了类似于 Xenix 但是公开源代码的 Linux。而微软这一边，则先后有了 Windows 3.1、Windows 95，并已开始开发 Windows NT。然而相比之下当时微软所提供的操作系统产品仍只能说是下里巴人，因为 Windows 3.1 和 Windows 95 乃至 Windows 98 实际上都是基于 DOS 的，而 DOS 在严格的意义上至少称不上是现代的操作系统。所以，UNIX/Linux 阵营的人们长期以来从技术上鄙视 Windows 也是事出有因，并非全是门户之见。但是，经过这么多年的发展，微软的操作系统产品走过了一条与用户水平同步提高的过程。到了现在，如果继续从技术上鄙薄微软的操作系统产品，那就错了。事实上，现在的 Windows 与 UNIX/Linux 一样都是阳春白雪，只是现在用户的水平也已提高，不再会因为曲高而和寡，更何况用户早已熟悉了 Windows，进而竟离不开 Windows 了。回顾从 20 世纪 80 年代至今的历程，微软的操作系统产品与用户一起走过了一个从低到高同步提高，从而日益普及的历程，获得了商业上的巨大成功。

现在的 Windows 操作系统，问题不在于技术上是否先进，而在于不公开源代码，使人们无法根据第一手的资料深入了解其实现，使公众在一定程度上处于不知情的境地。不过，这种不知情主要是在具体实现方面的，而不是机理方面的。事实上，如果只从机理、技术方面考察，则现在的 Windows 与 Linux 其实是很相似的。事实上微软的操作系统产品一直在从 UNIX/Linux 吸取营养，最明显的就是 Windows 的“视窗”机制显然借鉴了 X Window。即便是在 DOS 时代，例如 DOS 命令行的输

出“重定向”和“管道”机制，就不能不令人联想起 UNIX 命令行中的相应机制，所不同者倒是 UNIX 中的这些机制功能更强、更灵活。

另一方面，正因为微软的操作系统产品一直是商品，其技术上的发展就受到一定的限制，这是因为：

- 作为商品，其更新换代的速度不能过快，技术上的步子不能迈得太大。
- 作为商品，新老产品兼容的问题更为突出。这样，如果某种机制的设计“先天不足”，后面就比较难以弥补。

尽管如此，经过二十多年的发展，现在的 Windows 操作系统从技术上说大体与现代的 UNIX/Linux 相当，在一些细节上则各有千秋。当然，要是从市场占有率上说，那就具有压倒性优势了。

在 Windows 操作系统的发展历程中，Windows NT 特别是 Windows NT 4.0 的开发有着划时代的意义。在此之前的 Windows 版本，以及与此同期的 Windows 95/98 都是基于 DOS 的，大体上只是在 DOS 外面包了一层类似于 X Window 的外壳，使它看起来像个现代的操作系统而已。微软自己也知道，沿着这条路很快就会走到尽头，必须采用较新的操作系统技术才有出路；所以从 80 年代末 90 年代初就决心要开发新一代的 Windows，并将其称为 NT，即采用“新技术（New Technology）”的 Windows。事有凑巧，当时的 DEC 公司正好有一批研发 VMS 操作系统的骨干人员想要离开，于是就一起到了微软，这里面技术上的领头人就是后来成为 Windows NT 总设计师的 David Cutler，还有后来写了《Microsoft Windows Internals》一书的 David Solomon。VMS 是 DEC 公司为其 VAX 小型机开发的操作系统，虽然没有 UNIX 和后来的 Windows 那么流行，却确实可以说是个现代意义上的操作系统。与 UNIX 大体上属于同一时代。这些人为微软带来了 VMS 的技术和经验，使 Windows NT 特别是 Windows NT 4.0 的开发取得了成功。很自然地，今天的 Windows NT 带着来自 VMS 的基因，以至于有人调侃说：把 VMS 三个字母各加上 1，就成了 WNT。但是，从另一个角度来看，这也说明所谓“新技术”其实基本上就是 VMS 和 UNIX 已经采用了的技术。

当然，Windows NT 并不是 VMS 的简单重构，在 Windows NT 里面也可以看到 MACH 的影子。MACH 是卡内基·梅隆大学开发的一个微内核操作系统，其实 MACH 也是从 UNIX 变过来的。微内核操作系统的特点是把内核尽量缩小，而改由各种服务进程来提供原本由内核提供的功能，微内核系统的最大优点就是灵活性好。所以，早期的 Windows NT 带有明显的微内核痕迹，但是后来发现由服务进程提供的服务毕竟效率比较低，所以又逐步移回到内核中，后来甚至变本加厉，把图形功能也移到了内核中。因此，现在的 Windows 内核实际上包括了两大部分，一部分是本来意义上的操作系统内核，另一部分则是移到了内核中即系统空间中的视窗服务，前者的映像是 ntoskrnl.exe，后者的映像是 win32k.sys。而在 Linux 中，则后一部分就是用户空间的视窗服务进程 X Server。有时候可以听到人们在说 Windows 的内核比 Linux 的大很多，似乎本不应该有这么大，之所以有这么大是因为设计或实现得不好。其实不然，Windows 内核之所以大，是因为把图形界面也移到了内核中，如果把 X Server 也移到 Linux 内核中，那么 Linux 内核一样也会有这么大。

在早期的 Windows 系统中，视窗应用只是三个“子系统”之一，另两个子系统是 POSIX 和 OS/2。前者提供类似于 UNIX 的应用界面，是出于政府采购的要求；后者提供与 OS/2 操作系统的兼容性，

那是因为当时的微软与 IBM 有着比较密切的合作关系。当然，微软的主要精力都集中在视窗子系统上。在后来的发展中，渐渐地另两个子系统变得不那么重要了，因而就慢慢淡出了。另一方面，早期的 Windows 操作系统并非专门针对 Intel 的 x86 系列 CPU，特别是当时还有由 DEC 公司开发的 ALPHA 处理器，可是后来几经周折 ALPHA 处理器也慢慢淡出，现在的 Windows 特别是作为桌面系统的 Windows，就变成专门针对 x86 系列的了；只有用于（小型）嵌入式系统特别是手持设备的 WinCE 是个例外。

从老一代 Windows 包括 Windows 95/98 向新一代 Windows 的过渡，到了 Windows 2000 才基本完成。许多人把 Windows 2000 称为 Windows NT 5.0，就是因为 Windows NT 4.0 是个转折点，是个新的基础。至于 Windows XP，那又是一个新的 Windows NT 版本。甚至更新的 Longhorn，其基础仍旧是 Windows NT。所以，Windows NT 4.0 之于微软的操作系统，就像 80386 之于 Intel 的微处理器一样，都是具有划时代意义的。从那以后，无论是 Intel 的微处理器还是微软的 Windows，都没有发生进一步的根本性的质的变化，虽然也在发展，但是总体上属于小修小补，属于“量变”的范畴。

1.2　用户空间和系统空间

在计算机特别是微机的发展历史上，Intel 无疑写下了浓重的一笔。应该说，Intel 80386 CPU 的出现在计算机的发展史上起着里程碑式的作用。现在，经过三十多年的发展，PC 所采用的 CPU 芯片早已今非昔比，但是基本的系统结构却还是不出 80386 的窠臼，所以人们称 Intel 这整个系列的 CPU 芯片为“x86”系列。之所以在 86 前面有个 x，是因为 Intel 早期的产品中有个 8086 CPU，后来又依次有 80186、80286、80386、80486，而 Pentium 也有人称 80586。PC 最初采用的是 16 位的 8088 CPU，也应该算是这个系列中的一员，后来则随着这个系列的发展而水涨船高；而 Intel 的不断发展壮大，也主要得益于 PC 的日益普及。在这个系列中，80286 之前的 CPU 芯片在功能上和结构上都还不能与当时的小型机 CPU 相比拟，所以还谈不上采用现代操作系统的问题。之所以说那时的 CPU 芯片还谈不上采用现代操作系统，是因为 8086、80186 CPU 在寻址技术上还只能支持“实模式（Real Mode）”，还不能支持现代意义上的内存管理，因而不能支持系统空间与用户空间的划分。

那么采用现代意义上的操作系统要有些什么条件呢？

首先，操作系统在内存中的映像必须受到保护，不会因为用户程序的不良行为而受到伤害。也就是说，必须让 CPU 有“系统态”（或称“内核态”）和“用户态”这两种不同的状态，使得当 CPU 运行于用户态时就只能访问预先配置用于用户态的内存区域，而不能访问操作系统内核所在的内存区域。这样，用户程序的不良行为或误操作就只会影响到用户程序本身，而不会因伤害了操作系统而造成整个系统的崩溃。

“用户态”和“系统态”是不对称的，二者之间不是“井水不犯河水”的关系，而是井水不能犯河水但河水可以犯井水。这是因为，一方面内核有访问用户空间的必要，要不然用户程序就无法运行，用户程序是在内核的安排和操纵下运行的；另一方面内核是专门的软件产品，其质量一般而言是有保证的，所以内核不至于伤害用户空间的程序和数据；而用户软件则五花八门，质量不能保证，因而用户程序很可能伤害内核，所以要把内核保护起来。因此，这里的第一个条件就是内核的映像必须得到保护。

其次，不同的用户程序之间也应该互相防护，不能因一个程序的不良行为或误操作而伤害另一个用户程序。这就是说，用户程序及其数据、堆栈也都应该被保护起来，与其他用户的程序、数据和堆栈隔离开来，以防受到别的用户程序的伤害。

最后，除保护之外，用户程序在物理内存中的位置应该可以浮动。在多进程的系统中，一般都会有多个应用程序同时在运行，如果这些程序都只能被装载在固定的地址上运行，就会互相牵制，整个系统的运行就受到了限制。

这里的关键显然在于“保护”。正因为如此，Intel 把符合这些要求的寻址方式称为“保护模式”。80286 之前的 Intel 微处理器之所以不能（不配）采用现代意义上的操作系统，原因就在于不支持保护模式。相比之下，当时的小型机 CPU 都是可以支持“保护模式”的，尽管当时还没有这个词。唯其如此，UNIX 才有可能诞生在当时的小型机上。但是，从 80286 开始，特别是随着 80386 的出现，PC 就具备了采用现代操作系统的条件。这是因为 80286 开始支持“保护模式”（但是还有问题），而在 80386 上这方面的技术就很成熟了，那是 80 年代中期的事。

1.3 Windows 内核

如上所述，现代操作系统的一个明显特征就是用户空间和系统空间的划分，从 UNIX 时代以来，人们一直把存在于系统空间的代码和数据的集合称为“内核（Kernel）”，因此内核是有明确边界的。空间的不同，或者说 CPU 运行模式（系统态和用户态）的不同，是不会被混淆的本质区别。可是，在 Windows 的术语中却不同，微软并不把系统空间的所有代码和数据的集合称为内核，而是把这里面的一部分，即比较低层、与硬件靠得最近因而最为核心的一部分称为“内核”，即 Kernel。实际上，这也反映了当初微软在决策上的举棋不定，因为微软称为 Kernel 的那一部分大致上相当于一个微内核。有些资料甚至据此而认定 Windows 内核为“微内核”，殊不知现今的 Windows 内核恐怕是最宏的“宏内核”，因为连图形界面和视窗机制的实现也在内核里面了。微软的文献把同在系统空间但在所谓“Kernel”以上的部分称为 Executive。国内有些资料把 Executive 译为“执行体”，其实是不妥当的。试想，有“执行体”，莫非还有“不执行体”吗？实际上，在英语中，特别是在企业管理的语境中，Executive 是“管理层”、“企业高管”的意思，所以微软其实是把内核分成了两大层，其中的低层或者说核心部分称为“内核”，而高层则称为“管理层”。管理层中有些什么呢？读者在后面将看到，里面有“对象管理”、“内存管理”、“进程/线程管理”、“I/O 管理”、“安全管理”、“进程（线程）间通信”等模块。但是，这么一来，所谓“内核”的边界就变得不很清晰了。

相比之下，还是像 UNIX/Linux 那样，以系统空间与用户空间的划分为界，把存在于系统空间的所有成分的集合统称为内核比较清晰，也更为科学（因为有明确的判定方法）。所以，在本书中，只要没有特别加以说明，“内核”就是“系统空间”的同义词，而不是特指微软所称的那个内核；而微软所称的内核，则在本书中称为内核中的“核心层”。内核中从高到低在逻辑上分成若干层次，这一点任何操作系统的内核都是如此。事实上，研究操作系统的几种观点（视野）之一就是分层模拟、分层提供服务的观点。即便是微内核的内部，也还是分出层次，其中最底层的就是“硬件抽象层（Hardware Abstraction Layer）”HAL。

不过操作系统的概念有狭义和广义之分。狭义的操作系统就是指内核，而广义的操作系统则并