OS/2结构分析与设计

H

中国科学院希望高级电脑技术公司

OS／2 结构分析与设计

中国科学院希望高级电脑技术公司

内　容　摘　要

OS／2注定会成为一份非常重要的软件。在今后的十年中，数以百万计的程序员和用户将使用这个系统。本书详细描述了OS/2系统的总体设计思想；描述了构成OS/2的基本结构模块，并且讨论了这些模块如何满足正在到来的办公室自动化革命中的已预见和未预见到的需求；集中讨论了所有OS/2程序将遇到的问题、困难及其解决方法。全书共分三部分共21章。第一部分规划，包括第一章至第三章。第一章规划的历史；第二章目标和兼容性问题；第三章OS/2宗教。第二部分结构，包括第四章至第二十章。第四章为任务；第五章微进程和调度程序/优先级；第六章用户接口；第七章动态连接；第八章文件系统名字空间；第九章存贮管理；第十章环境串；第十一章进程间通讯；第十二章信号；第十三章图象管理程序和VIO；第十四章交互式程序；第十五章文件系统；第十六章设备管理、数据一致性及定时设施；第十七章设备驱动器和硬错误；第十八章I/O特权和排错机制；第十九章3x Box；第二十章API族。第三部分包括第二十一章未来。书后有一附录：词汇表。

本书是了解认识OS/2体系结构及其设计思想的较好的参考书，是使用OS/2的基础。

编审者

1988.10.

目　录

绪论……（1）
第一部分　规划……（4）
第一章　规划的历史……（4）
1.1　MS-DOS 1.0 版本……（5）
1.2　MS-DOS 2.0 版本……（5）
1.3　MS-DOS 3.0 版本……（6）
1.4　MS-DOS 4.0 版本……（6）
第二章　目标和兼容性问题……（7）
2.1　目标……（7）
2.1.1　绘图的用户接口……（8）
2.1.2　多任务……（10）
2.1.3　存贮管理……（10）
2.1.4　保护……（11）
2.1.5　密封……（12）
2.1.6　进程间通讯（IPC）……（12）
2.1.7　直接设备存取……（13）
2.2　兼容性问题……（13）
2.2.1　实地址方式和保护地址方式……（13）
2.2.2　在实地址（兼容）方式下运行应用程序……（14）
2.2.3　通用功能可兼容性……（16）
2.2.4　向下的兼容性……（16）
第三章　OS／2 宗教……（18）
3.1　最大灵活性……（18）
3.2　一个稳定的环境……（20）
3.2.1　存贮保护……（20）
3.2.2　副作用保护……（22）
3.3　错误定位……（23）
3.4　软件工具方法……（24）
第二部分　结构……（26）
第四章　多任务……（26）
4.1　子任务模型……（27）
4.1.1　标准文件指针……（28）
4.1.2　匿名管道……（31）
4.1.3　细节，再细节……（33）

4.2 PID 命令子树……（36）
4.3 Dos ExecPgm ……（38）
4.4 DosCwait ……（41）
4.5 子任务和命令子树的控制……（43）
4.5.1 Doskill Process ……（44）
4.5.2 Dos Set Prty……（44）
第五章 微进程和调度程序／优先级……（45）
5.1 微进程……（45）
5.1.1 微进程堆栈……（45）
5.1.2 微进程的用途……（46）
5.1.3 互锁……（47）
5.1.4 微进程 1 ……（49）
5.1.5 微进程死亡……（49）
5.1.6 性能特征……（49）
5.2 调度程序／优先级……（50）
5.2.1 一般优先级类别……（51）
5.2.2 时间—临界优先级类别……（53）
5.2.3 低优先级类别……（54）
5.2 4 设置进程／微进程优先级……（54）
第六章 用户接口……（56）
6.1 VIO 用户接口……（56）
6.2 显示管理程序用户接口……（56）
6.3 显示管理程序和 VIO 的兼容性……（57）
第七章 动态连接……（58）
7.1 静态连接……（58）
7.2 装载时动态连接……（59）
7.3 运行时动态连接……（61）
7.4 动态连接，进程和微进程……（62）
7.5 数据……（62）
7.5.1 暂用数据……（64）
7.5.2 全程数据……（65）
7.6 作为子例程的动态连接包……（65）
7.7 子系统……（66）
7.7.1 特殊的子系统支持……（67）
7.8 动态连接用作与其他进程的接口……（68）
7.9 动态连接用于与核心的接口……（69）
7.10 动态连接在结构上的作用……（70）
7.11 执行过程细节……（71）
7.11.1 动态连接数据保护……（71）

7.11.2 动态连接的生存、死亡以及共享……(72)
7.11.3 动态连接副作用……(74)
7.12 动态连接的名字……(75)
第八章 文件系统名字空间……(77)
8.1 文件名……(77)
8.2 网络存取……(78)
8.3 名字的生成和兼容性……(78)
8.4 许可……(79)
8.5 在文件系统名字空间中的其他目标……(79)
第九章 存贮管理……(81)
9.1 保护模型……(81)
9.2 存贮管理 API……(82)
9.2.1 共享存贮器……(82)
9.2.2 巨存贮器……(84)
9.2.3 运行数据段内的代码……(85)
9.2.4 存贮器子分配……(86)
9.3 段交换……(89)
9.3.1 交换杂注……(92)
9.4 状态和信息……(93)
第十章 环境串……(95)
第十一章 进程间通讯……(98)
11.1 共享内存……(98)
11.2 信号量……(98)
11.2.1 信号量恢复……(100)
11.2.2 信号量调度……(102)
11.3 命名管道……(102)
11.4 队列……(105)
11.5 动态数据交换(DDE)……(107)
11.6 信号……(107)
11.7 几种 IPC 形式的结合……(108)
第十二章 信号……(109)
第十三章 图象管理程序和 VIO……(112)
13.1 在 PM 和 VIO 之间选择……(113)
13.2 后台 I/O……(114)
13.3 在 VIO 支持下的图形……(114)
第十四章 交互式程序……(118)
14.1 I/O 结构……(118)
14.2 Ctrl-c 和 Ctrl-Break 的处理……(121)
第十五章 文件系统……(124)

15.1 OS/2 文件系统……（125）
15.2 存贮媒体文件卷的管理……（125）
15.3 I/O 效率……（128）
第十六章 设备管理，数据一致性及其定时设施……（129）
16.1 设备管理……（129）
16.2 数据完整性……（131）
16.2.1 信号量……（133）
16.2.2 Dos Buf Reset……（134）
16.2.3 Write throughs……（135）
16.3 记时服务……（135）
第十七章 设备驱动器和硬错误……（138）
17.1 设备驱动器……（138）
17.1.1 设备驱动器和 OS/2 通讯……（138）
17.1.2 设备驱动器程序设计模型……（140）
17.1.3 设备管理……（144）
17.1.4 双重方式……（144）
17.2 硬错误……（145）
17.2.1 硬错误处理进程……（146）
17.2.2 应用程序硬错误处理……（147）
第十八章 I/O 特权和排错机制……（148）
18.1 I/O 特权机制……（148）
18.2 排错机制……（149）
第十九章 3x Box……（152）
第二十章 API 族……（156）
第三部分
第二十一章 未来……（160）
21.1 文件系统……（160）
21.2 80386……（163）
21.2.1 分段较大……（163）
21.2.2 多个实际模式环境……（164）
21.2.3 完全保护的能力……（164）
21.2.4 其它特征……（164）
21.3 下一个十年……（165）

绪　论

技术上的突破与渐进是两种完全不同的发展方式。渐进是对现有产品的某项改进，简单明了，不会使人感到惊奇。一项改进措施出现了，人们看到了这种改进，知道它将怎样为他们服务，然后便开始使用它。

与先前无紧密关联的重大进展——技术上的重大突破——遵从一种不同的方式。通讯领域是个好的例子。本世纪初，有一个庞大的机构来支持个人间的通讯。每天发送几次邮件，同时还有各种传递消息的部门。商人把消息口授给他的秘书，再由秘书把消息送到信使部门。这个部门将消息传送到它附近的预定地点，在那里由专人将消息发送给接收者。

在这种环境里出现了一个技术上的突破——电话。电话的发明是一项突破，而不是渐进，因为它提供了一种全新的通讯方式，而不是在现有方法上的某种改进。正是突破性发展阻碍了它自身得到认可。大多数商人认为电话只是一种新玩具，没有什么实际的用处。“它能对我有什么好处呢？当我口授了消息，由秘书写下来送到邮局，他们再给收信人处的邮局打电话，然后消息被抄录下来——恐怕还要搞错呢，再分送给收信人的秘书。这和由送信人传递不是一样快嘛！我所有的通信人都在附近，再说有信使在，我就不用付钱雇人一整天守在电话机旁等消息了。”

这是突破性技术发展的最初阶段的一个典型例子——潜在的用户试图将它套入现存的工作模式来评价它的好坏。在我们的例子中，商人没有意识到他们已不用再将消息写下来，而且人们也已不必再在收信处抄录消息了。商人也没有意识到他的通信人都在附近的原因是他们必须在信使传递的合理范围之内。电话消除了这种必要条件，允许利用工厂和原材料附近更为有效的或办公空间花费更少的位置。但是，只有首先使用了电话，我们才能认识到这些优越性。

对于突破性技术应用来说，另一个障碍是还没有必需的基础设施。如果你要找的顾客没有电话的话，那么电话将变得毫无用处。电话的特性决定了它需要一种规范；只有建立了这种规范，你的客户才可能拥有一部电话，但这部电话也有可能被连接到你打不到的网络上去。进一步讲，由于这项技术尚处在初级阶段，它的设备还是很粗糙的。

这些障碍并不是无法逾越的。一部分人对通讯的迫切需求使得他们愿意进行新的发明并能够容忍这些初级阶段所特有的困难。另一部分人，凭着他们的勇气和抱负，使用了这个新装置去扩充他们现有的体系。终因这种技术是强有力的，一些人开始利用它来增强现有的技术。例如，某个信使部门会建立几个彼此由电话联结的办公室，利用电话来加快短信的传递。它们通过电话把短信传到离接收地点最近的办公室，在那儿，消息被记录下来，然后再按通常的方式传递出去。这样使用电话是浪费的，但对旧的服务机构的需求足够高的场合，任何改进，即使有“浪费”，还是受欢迎的。

突破性的技术一旦跨进门槛，其发展将是不可阻挡的。一段时间过后，规范被建立，障碍被排除，而且更重要的是，工具改变了它的使用者。一旦电话达到可用，产业

和个人的实践活动就以全新的方式得到发展，这种方式是以前所没有想到的，因为在过去这是不可能的。过去，信使服务是足够快的，但这仅仅是由于信使服务在电话之前是可行的最快的技术。正是电话改变了它的使用者的生活方式。

这种人类活动结构上的变化说明了为什么一个聪明人会这样说："电话是愚蠢的小玩艺儿"，而几年后却又说："电话是必不可少的"。工具使用者的这种改变——是由工具本身引起的——也使得人们难以预见这个新技术的长远影响。从现有的趋势来推断是极其不精确的，因为新的工具破坏了许多现有的实践活动并产生了完全难以预见的新的活动。阅读早期关于未来生活的明显愚蠢的预言并嘲笑这些预言者是非常有趣的事情，但实际上，这些预言者往往是具有高超智力和受过教育的。他们唯一的错误就是把这些新的发展当成渐进而不是当做突破来对待。他们只看到这种新的发展怎样改进了他们现有的实践活动，而没有看到它是怎样取代那些活动的。

数字计算机是一项明显的突破，并已经历了典型的三阶段模式："外来的玩具"、"有限的利用"和"必不可少"。在商业和科学计算的环境下，主体计算机已经走完了整个过程。IBM 对于计算机市场的最初估计是几十台。但是，随着这项技术及其基础设施的发展，随着人们产生了适应于计算机的工作方式，计算机的使用不断增长——从人口统计局到人寿保险公司，到工资发放系统以及 MIS（管理信息科学）系统和飞机订票网络这类全新的功能。

微型计算机也经历了类似的过程。"外来玩具"阶段已经让位给"有限的利用"阶段。我们正着手发展它的规范以及基础设施，离"必不可少"阶段已要不了几年时间了。由于期待着这一阶段的到来，Microsoft 担负起了 OS／2 的设计和开发工作。

尽管对主体计算机革命的研究有助于我们预见微型计算机革命的道路，但微型计算机不单单是"便宜的主体计算机"。微机革命将遵从技术突破的传统方式，产生出新的需求和新的用途，而这些需求和用途是无法只通过研究主体计算机所发生的情形来主观预见的。

本书是根据微型计算机作为技术突破的性质以及正在到来的第二次工业革命的影响。OS／2 的设计者们在可能的最大范围内做好预先的准备工作，以满足当工具——个人计算机——与工具的使用者之间达到了新的平衡时对系统的需求。仅靠有关 MS-DOS 的知识以及对 OS／2 参考手册的仔细阅读还不足以弄清 OS／2 支持的编程环境。这不仅是因为产品的复杂性，而且也是因为许多设计元素被选来提供某种服务，而这些服务从技术突破的开始阶段来预见似乎是不重要的，解决的是还没出现的问题。

其它书中提供了参考信息和详尽的编写 OS／2 程序的指导。本书描述构成 OS／2 的基本结构模块，并且讨论这些模块将如何满足正在到来的办公室自动化革命中的已预见到和未预见到的需求。本书集中讨论所有 OS／2 程序将遇到的问题和困难及其解决方法，而不考虑程序员所用的编程和接口模块如何。

正如技术讨论中常见的那样，OS／2 中的每一部分都以某种方式与其余的各部分相联结。关于胫骨的讨论自然地要引向对股骨的讨论，然后又将是更深的一层。本书的作者和编辑花了很大的力量试图把所有的内容组织进一个逻辑的序列中去，但这些内容的特性决定了不可能完全成功地做到这一点。事实上，我们经常从某个不同的观点或侧重点重复某些内容。由于这些原因，本书的索引注明了条目或主题的每一出处，尽管这

仅仅是外围的。过多的参考索引（包括几个并不值得的）总比过少的要好得多。当你要查找一个特定的条目时，建议你首先查阅目录页来确定主要讨论的位置，然后再细读索引以检出此条目可能出现的某些预料不到之处。

第一部分　规划

第一章　规划的历史

成立 Microsoft 的宗旨是要实现每个办公桌上都有一台计算机的设想。——一个第二次工业革命的设想。第一次工业革命实现了体力劳动的机械化。在十八世纪以前，几乎所有的产品都是由人的双手一次一件地创造和制成的。几乎所有的动力都来自人的肌肉的能量，仅有很少的例外，如畜力耕地和运货。第二次工业革命将实现日常的智力劳动的机械化。在接近于这次革命的今天，人们仍然进行着一次一件的“思维劳动”。

某些任务——那些规模很大，并且能被严格描述的，如工资额的计算——已经实现了自动化，但大多数的“思维劳动”仍然是由人而不是计算机来完成的。我们已能使计算机成为耕地骡马的替代物，但我们还没能使计算机代替电钻或洗衣机。

当然，计算机不能取代初始的思考和创造力（至少在近期内还不行），就如同物质领域内机器不能替代设计和创造一样。但是，白领办公室主要工作的绝大部分包括了信息的常规操作。第二次工业革命将使我们从“烦恼疲劳的工作”——常规的数据操作、分析及决策——中解脱出来，得以去处理那些需要人类判断力的工作。

许多人没有认识到第二次工业革命的必然性。因为他们的工作结构是在没有计算机的情况下形成的，所以他们不能理解一台计算机怎么能完成他们的75%的工作。但是，遵从技术突破的应有模式，微型计算机的巨大效用将改变它的使用者及其工作方式。

例如，有大量的工作难于用计算机来完成，这是因为输入信息是写在纸上的，需要花费过多的时间把它们敲入计算机。十年前，计算机的倡导者们设想了“无纸办公”作为解决这个问题的办法：所有的资料均由计算机生成，然后依靠电子技术手段或供助磁盘传到其它的计算机中去。办公室也确实正变得越来越不需要纸张，而且强有力的网络系统的出现将加快这一过程，但是纸仍然是一个很有用的媒介，因此，最近几年已在另一方面有了进展——把纸作为一种计算机的输入和输出设备结合进来，功能强大的激光打印机，台式出版系统，以及光学扫描器和光学字符识别将使以纸作媒质进行输入输出的方法更加实用。

尽管 Microsoft 的创建者完全意识到了第二次工业革命的影响，但是谁也无法从细节上预见这次革命将如何展开。Microsoft 是根据两类目标来进行日常的决策的：短期目标，这是大家都清楚的；长期目标，是自动化办公的理想，每一决策都要符合我们的短期目标，同时也要与我们的长期目标相一致，而这个理想也随着这次革命的进展将变得更加明确。

当 16 位微处理器问世的时候，Microsoft 知道这时“铁”已是足够坚硬的了，可以开始实现理想了。但是一个功能强大的计算机环境同时要有坚硬的铁和高级的操作系统。铁是可用的，但是过去作为八位微机规范的操作系统却不适用。这就是 Microsoft

进入操作系统开发工作的时间和原因：我们明白，为实现我们的理想需要有一个强有力的操作系统，而保证这个操作系统的延续性和适用性的唯一途径就是由自己来编写。

1.1 MS—DOS 1.0版本

MS—DOS 的开发始于 IBM 请 Microsoft 为其研制中的新产品——IBM 个人计算机（PC）设计一个磁盘操作系统。当时 Microsoft 仅有的操作系统是 XENIX，它是 AT&T 的UNIX（R）的一个许可版本。XENIX/UNIX 要求处理器具有存贮管理和保护设备的功能。由于 8086／8088 不具备这二者，同时也由于 XENIX／UNIX 对存贮器的需求——对于当时的小型机标准是适合的——对微型机标准是过大的，因此必须设计一种与此不同的操作系统。

当时，由 Digital Researeh（R）Incoporated（DRI）设计的 CP／M—80 一直是一个作为标准的8位机操作系统。那时绝大部分微型机软件都可以在 CP／M—80 上使用。鉴于这个原因，Microsoft 决定使 MS-DOS 1.0版本尽可能地与 CP／M-80 兼容。8088 处理器无法运行现有的为 8080 处理器编写的程序，但因为 8080 的程序可以很方便、半自动地转换为可在 8088 上运行的程序，所以 Microsoft 认为通过减少操作系统的非兼容性来减小配合上的混乱会加速 IBM-PC 对 MS-DOS 的采用。

一个重要的软件产品将需要大量的研制时间，而 IBM 又急于推出它的 PC 机。因此，Microsoft决定寻找一个合适的软件作为形成 MS-DOS 1.0 版本的基础。这样的软件产品在 Seattle Cowputer Products 中被找到了。Tim Paterson 是那里的工程师，他制作了 CP／M-80 的“无性系”，称为 SCP-DOS，这是可在 8088 处理器上运行的操作系统。Microsoft 买下了这个产品及其源码的使用权，以此作为开发 MS-DOS 1.0 版本的起点。

MS-DOS 1.0 版本于 1981 年8月推出。它仅适用于 IBM-PC，由 4000 条汇编语言源码组成，占有 8KB 的存贮器。1982又推年出了 MS-DOS 1.1 版本，它支持双面 320KB 的软盘。

Microsoft 的目标是使 MS-DOS 1.0 版本对 CP／M 高度兼容，而 MS-DOS 1.0 版本也确实满足了这一点。具有讽刺意味的是，它显然比 ORI 自己的 8088 产品 CP/M-86的兼容性更强。我们在后面将会看到，这种对 CP／M 的兼容性在当时是必要的，但后来却给 Microsoft 的工程师们带来了许多麻烦。

1.2 MS-DOS 2.0版本

1982 年初，IBM 向Microsoft 透露了它正在研制一个支持硬磁盘的个人计算机，即 IBM XT。Microsoft 于是开始了 MS-DOS 2.0 版本的设计工作以对这种新的磁盘硬件提供支持。由于一直保持着兼容 CP／M-80 传统的 MS-DOS 是为软盘环境设计的，因此必须对它进行修改。当时，一个磁盘仅包含一个目录，而且这个目录最多只能容纳 64 个文件。这种决策在起初是合理的，因为软盘仅能容纳 180KB 的数据。

但是对于硬盘来说，64 个文件的限度确实太小了，而且仅用一个目录来管理可能的上百个文件也很不灵活。因此，MS-DOS 2.0 版本的设计者 Mark Zbikowski, Aaron Raynolds, Chris Peters, 和 Nancy Panners——增加了一个分级的文件系统。在一个

分级的文件系统中，一个目录可以包含其他的目录和文件。同样，被包含的这些目录也可以既包含文件又含有新的目录，依此类推。一个分级设计的系统起始于主目录或称“根”目录，它可以拥有一个树结构的文件和目录的集合（见图1-1）。

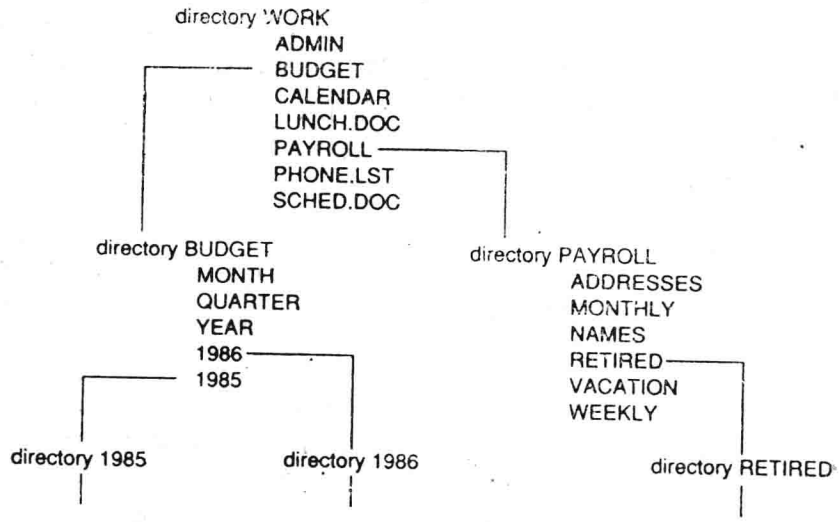

Figure 1-1.

图 1—1

一个树形分级的目录。在 WORK 目录下有五个文件（ADMID，CALENDA R，LONCH；DOC，PHONE；LST、SCHED；DOC）和两个子目录（BUGET，PAYROLL），每个子目录下又有其自己的子目录。

1.3 MS-DOS 3.0 版本

1984 年8月，IBM推出了 IBM PC／AT，同时也推出了 MS-DOS 3.0 版本。AT机具有一个 80286 处理器，但当运行 DOS 的时候，它使用植入芯片的 8086 仿真方式，就如同运行一个“快速的 8086”一样。芯片的扩展的地址范围和保护方式结构都没有得到利用。[1]

MS-DOS 3.1 版本于 1984 年11月推出，它具有网络支持功能。1986 年1月又推出了 MS-DOS 3.2 版本，这是一个较小的版本，它支持 3.5 英寸的软盘并将其格式化功能集成到外设驱动器中。接着，1987 年出现了 MS-DOS 3.3 版本，这次推出仍主要改进是支持 IBM PS／2 及其兼容的硬件。

1.4 MS-DOS 4.0 版本

1983 年1月，Microsoft 开始了研制 MS-DOS 的多任务版本的工作，当时它在内部被称作 MS-DOS 3.0 版本。当一个单任务的 MS-DOS 的新版本被安上了 MS-DOS 3.0 的名字时，这个多任务版本又改了名，在内部被称为 MS-DOS 4.0 版本。因为 MS-DOS 4.0 版本仅在实地址方式下运行，所以它既可以用于 80286 的机器也可

用于 8088 和 8086 的机器。实地址方式环境的限制使得 MS-DOS4,0版本成为一个专用的产品。尽管 MS-DOS 4.0 版本支持完全优先抢占的多任务系统，但系统内存限制在无交换 [2] 的 640KB 实地址方式的空间内。这意味着所有的进程都只能装入仅有的 640KB 存贮空间中。这样就只能运行与 MS-DOS 3.X 版本兼容的实地址方式的应用程序；其他的进程必须是特定的 MS-DOS 4.0 版本的、能适应其环境且与操作系统相配合、能和单任务的 MS-DOS 3.X 版本实地址方式的应用程序和平共处。

由于这些限制，并不打算对 MS-DOS 4.0 版本作常规的发行，而将它作为专用的 OEM 的台架去支持扩展的 PC 结构。例如，一个高效的电话管理系统可通过利用以 MS-DOS 4.0 版本为背景的特殊的进程植入一台 PC 机中去控制电话设备。这样，这台机器就可以被称作具有“具有机内超级电话的兼容于 MS-DOS 3 版不的 PC 机”。

尽管 MS-DOS 4.0 版本是作为特殊的 OEM 产品推出的，但这一规划——现被称作 MS-DOS 5.0 版本——还在继续进行。我们的目标是利用 80286 的保护方式的优点来提供不受限制的完全通用的多任务处理——如同在 MS-DOS 4.0 版本中见到的那样——具有一个实地址方式的单一环境。此后不久，IBM 和 Microsoft 签署了面向 MS-DOS 5.0 版本（现被称作 CP／DOS）的设计和开发的联合开发协议。协议内容是多方面的，但主要是针对联合开发以及随后的共同所有权，双方都具有对所形成的产品的各种权利。

随着这个项目接近结束，销售人员看到 CP／DOS，某某 DOS 5，某某 DOS 4，某某 DOS 3 一个接一个的出现，它们认为有必要……你可能会猜到……改变一下名字。因此，本书的以后部分将讨论一个叫作 OS／2 的操作系统的设计和功能。

注释：

〔1〕类似于 Microsoft XENIX／UNIX 的产品可使用这个处理器的保护方式在 PC／AT 及其兼容机上运行。这样做的原因是 XENIX／UNIX 以及类似的系统没有预先已存在的实地址方式的应用程序需要支持。

〔2〕要支持一般目的的交换，没有存贮管理硬件是行不通的，而存贮管理硬件是 8086 实地址方式所不具备的。

第二章　目标和兼容性问题

OS／2在许多方面类似于传统的多任务操作系统：具有多任务处理、调度，磁盘管理和存贮管理等功能。但它还有许多不同之处，这是因为一台个人计算机和一台多用户的小型机是有很大区别的。OS／2 的设计者根据两个清单着手工作：一是目标清单，一是兼容性问题清单。本章描述了这些目标和兼容性问题，为以后关于设计本身的讨论提供依据。

2.1　目标

OS／2的基本目标是成为理想的办公自动化的操作系统。朝着这个总的目标，设计

者们制定了以下这些表面上看起来互相矛盾的中间目标：

（1）在不引入显著的开销的前提下，提供独立于设备的图形驱动程序。

（2）允许直接寻址高带宽的外设，同时保持对这些外设的使用进行虚拟化或实行分配的能力。

（3）在不低于单任务系统所能达到的性能和响应的前提下，提供多任务服务。

（4）为每一个程序及其子程序提供一个规格化的环境，同时也要提供一个不受系统中其它程序影响的标准环境。

（5）提供保护环境以确保系统的稳定性，同时又不限制应用程序在无保护系统下的兼容性。

2.1.1 绘图的用户接口

人们接收信息的最快和最简便的方式是通过眼睛。我们天生就是有视觉的生物。我们的眼睛可以迅速地获取信息；它们也可以去“寻找”想要得到的信息，通过眼部肌肉微小而快速的动作将它们的视线移向和移离目标。人类大脑的很大一部分是专门用以处理视觉信息的。人们从视觉材料中提取数据和含义——从文本到图形，到动画——比从其它任何材料中提取信息的过程要快数百倍。

因此，如果一个办公自动化系统要迅速地而且是以易于接受的形式来提供大量信息的话，那么强有力的图解性能是必不可少的。由于高分辨率的显示需要开销很大的内存和计算功率，因此这种性能在早期的小型机操作系统中是不多见的。现在的微型机已具有能容纳显示信息的内存空间，它们也具有可产生并操作这些信息的 CPV 能力，而且对于这样能力的最好的利用莫过于去支持强有力的，易于使用的绘图应用程序。

图形可以呈现为许多形式——图象、表格、绘画，曲线图——也可能是有彩色甚至是配成动画的。所有这些是对字母和数字文本显示的有力补充。绘图应用程序并不是一定要用到图表和图象的。WYSIWYG（所见即所得）排版程序仅能显示文本，但如果把文本表用图形方式表示，则屏幕可显示出各种字体、各种大小的字号和排版的疏密情况等等。

OS／2 的屏幕绘图部分应是独立于设备的；即：对于任何应用程序都能在不依赖于任何特定的图形显示接口板的专有特性的条件下，显示出正确的图形“画面”。显示的手段每年都有进一步的提高；把应用程序和一个特殊的显示板连系在一起是目光短浅的作法，无论这块显示板多么好，不出两年都会成为过时的东西。

这种想法要求每一个设备都带有一个被称作设备驱动程序的软件包，以此来封闭面向设备的专用代码。应用程序对一个一般的设备发出控制指令，然后由设备驱动程序把这些命令译成适合于实际设备的特性的指令代码。这样做的结果是，新的图形显示板的制造厂家需要编写一个相应的设备驱动程序，将它和图形显示板一起提供给客户。应用程序毋需了解设备的实际情况，设备驱动程序也不必考虑具体应用程序的要求，而只需考虑它们共享的通用接口的特性。这个通用的接口描述的是一个虚拟的显示设备；一般地，用一简单、标准的接口来掩盖一个复杂的真实情形的技术，称为“虚拟化”。

图 2—1 给出了传统的操作系统设备驱动程序的结构。应用程序不直接访问驱动程序，因为驱动程序需要在处理器特权状态运行才能操纵它们的设备；而要访问驱动程

序的应用程序的应用程序只能运行于非特权运行状态。在 80286/80386 系列处理器的语言中，特权状态被称为 ring 0，非特权状态被称为 ring 3 。操作系统的作用类似于一个中间人：接受请求，使其生效，处理当仅有一台设备但有多个使用者的情况下出现的问题，以及把请求送到设备驱动程序中。设备驱动程序的响应或返回的数据沿着相反的路径，经过操作系统，又回到应用程序当中。

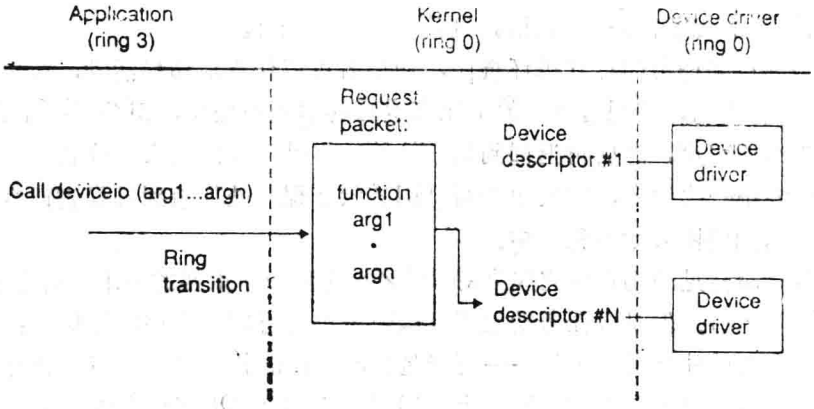

Figure 2-1

图 2--1

传统的设备驱动程序结构。当一个应用程序想要进行设备 I／O 时，它就调用操作系统，操作系统建立一个设备请求包，确定目标设备并运送这个包。设备驱动程序的响应沿着相应反的路径穿过核心返回到应用程序。

这种方法解决了设备虚拟化的问题，但在性能上有一定的损失。应用程序和设备驱动程序之间的接口是狭窄的；也就是说，信息所能采用的形式通常受到限制。人们一般希望应用程序能建立一个申请块，在申请块内装有设备驱动程序为该申请服务所需的全部信息和数据；对操作系统的实际要求很简单："把这个申请块传给设备驱动程序"。但建立这样的块需要花费时间，而且在设备驱动程序中将它分解开也要花费时间。回答还要花费更多的时间：设备驱动程序先建立回答块，再由操作系统拷贝过去，应用程序再将它分解。还有，更多的时间花费在请求通过操作系统内层，检验和复制申请块，选择到相应的设备驱动程序的路径，等等。最后，在 ring 态（特权状态和非特权状态）间的转换也是要消耗时间的，这类转换有两次——进入特权状态和返回非特权状态。

这种性能上的损失在无图形支持的系统中是可以接受的，因为，典型情况下，屏幕的一次完全更新仅需 1920 字节（或更少）的数据。而目前的图形设备每次更新需要 256,000 字节或更多的数据，而且将来的设备在这方面的需求量会变得更大。进一步讲，用户可能会希望在一秒钟内数次更新这些高分辨率的屏幕。〔1〕

OS／2 需要强有力的独立于设备的图形显示支持，要求具有宽而且高效率的用户接口——不包含 ring 状态转换、操作系统或其他不必要开销的图形支持。我们在后面将会看到，OS／2 利用被称作动态连接的手段满足了这种要求。

2.1.2 多任务

要成为真正的有用工具，个人计算机必须能同时处理多项工作——被称作多任务的功能。我们人类总是同时进行多项工作的。例如，你可能同时参加三个项目的工作，还可能一面阅读一本小说而另一面又要听西班牙语课。你依次选则一项任务，干上一段时间，然后又决下这项任务去干点别的事情。这种方式叫做串行多任务。人们还可以同时做某些事，比如一边开车一面谈话。这就叫并行多任务。

在一个串行多任务的计算机环境中，使用者可以任意地切换活动，在每一个任务上工作一段时间。例如，使用者可以放下正在进行的字处理程序，但并不终止它，而转去查询一个图表绘制程序，然后再返回到正在等待的字处理程序上来。或者，如果某人打来电话，请求安排一次约会，则使用者可从图表绘制程序处切换到日程表程序中，查询日历，然后再返回到图表绘制程序中。

多任务显而易见的功用使它成为对 OS／2 的另一个重要的要求：许多程序及应用程序应能同时运行。但是多任务功能不仅仅限于可进行应用程序间的切换；并行多任务使得应用程序可以独自进行工作——可能是打印一份很长的文件或者重新计算一张大的图表——而同时用户可以去运行另一个应用程序。由于 OS／2 支持完全的多任务，因此，如果提供先进的设备，比如无中断或者对用户工作无干扰的网络通讯，则 OS/2 除了用户正在运行的应用程序外，还可以运行其他的程序。〔2〕

2.1.3 存贮管理

多任务还算是易于实现的。所必需的就是一个周期的硬件中断源，比如一个时钟电路，其作用就是使操作系统能够作用于一个“范围开关”或进行再调度。尽管如此，要想使多任务系统成为一个可用的系统，还需要一个高效率的存贮管理系统。例如，用户想要在一个系统上运行两个程序，每一个都起始于内存中尽可能低的位置，以使可用的内存空间最大。不幸的是，如果系统支持多任务而且用户想要同时运行这两个程序，而每一个程序都企图使用同一存贮单元，那么就会产生程序间的相互破坏。

存贮管理系统利用植入 80286/80386 处理器（如 IBM PC／AT 及其兼容机和 80386 支持的微机）的特殊硬件设备解决了这个问题〔4〕。存贮管理系统利用这个硬件将实际的存贮器虚拟化，这样每一个程序就如象独自占有整个存贮空间一样。

存贮管理不仅用于避免程序间的冲突，还要跟踪每个存贮页片的拥有者和使用者，以便当某个存贮片不再使用的时候及时地将其收回，即使它原来的拥有者没有明确地释放它们。一些操作系统回避了这项工作，它们的办法是假定每一个应用程序在运行完的时候都会记着归还占用的内存，或者是通过检查内存中的内容并从内容来确认存贮器是否仍被占用（这叫做“废品收集”）。这两种方法 OS／2 都没有采用。因为 OS／2 将运行由众多软件商编写的大量的程序，想通过检查来确认空闲的内存空间是不可能的，而且通过应用程序自身来保证正确性也是不明智的。跟踪内存目标的拥有和使用情况是件复杂的工作，我们在有关动态连接程序库的讨论中将会看到这一点。

最后，存贮管理系统还必须管理内存的超载。OS／2 的多任务功能允许许多应用程序同时运行；于是，RAM 必须能容纳所有这些程序及其数据。尽管 RAM 的价格