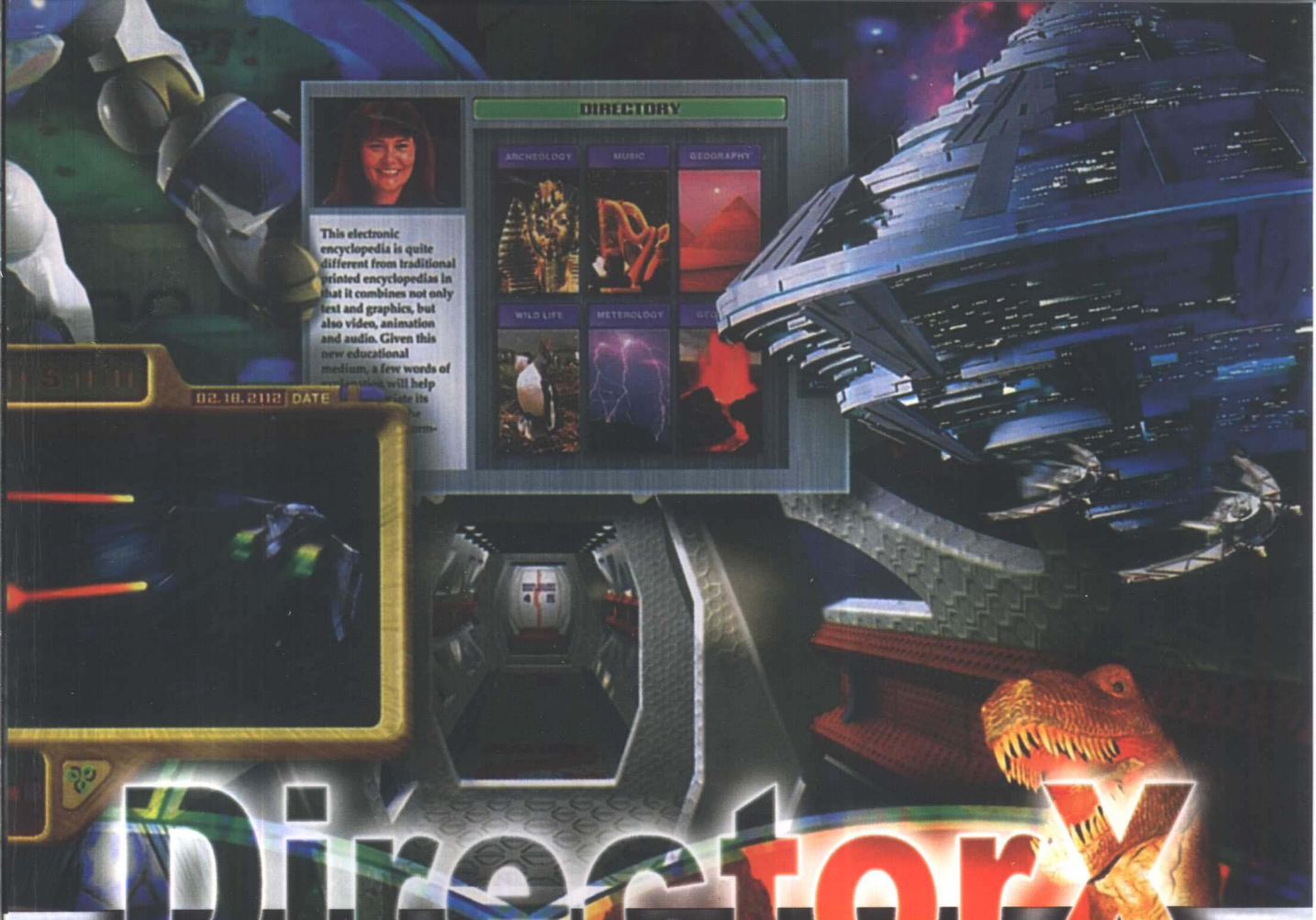


# DirectorX 7.0 高级编程



傅宇旭 编著

科学出版社

# DirectorX 7.0 高级编程

傅宇旭　编著

科学出版社

2000

## 内 容 简 介

本书主要从 DirectorX 7.0 技术的各个层面介绍了 DirectorX 7.0 开发程序的具体技术及其实现方法。它主要包括 DirectorDraw, DirectorPlay, DirectorSound, Director3D, DirectorInput & Output 和 DirectorMusic 等部分。DirectorDraw 介绍了对内存的直接读写；DirectorPlay 使游戏在调制解调器和网络之间的连接更加简单方便，并使 Director3D 和 DirectorDraw, DirectorMusic 和 DirectorX Sound, DirectorX Play 和 Winsock 之间的结合更紧密。

本书着重介绍 DirectorDraw, DirectorPlay 的编程和使用技巧，本书通过生动的实例讲解，向读者深入浅出地介绍 DirectorX 7.0 技术的各个层面，从而使读者了解和熟悉 DirectorX 7.0 的编程和应用技巧。

本书可供大专院校的学生、教师及软件开发人员应用。

**图书在版编目（CIP）数据**

DirectorX 7.0 高级编程/傅宇旭编著. -北京：科学出版社，2000
ISBN 7-03-007974-4

Ⅰ.D… Ⅱ.傅… Ⅲ.三维-动画-应用程序，DirectorX Ⅳ.TP391.41

中国版本图书馆 CIP 数据核字（2000）第 60675 号

科学出版社 出版
北京东黄城根北街 16 号
邮政编码：100717

北京双青印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2000 年 8 月第 一 版 开本：787×1092 1/16
2000 年 8 月第一次印刷 印张：15 1/2
印数：1—5 000 字数：353 000

**定价：20.00 元**

（如有印装质量问题，我社负责调换<环伟>）

# 前　言

当你醉心于千变万化的电脑游戏的时候，你可曾想过游戏是如何开发出来的？也许你梦想有一天自己也能够加入到游戏开发的行列，而不仅仅只是一个玩家，那么本书将给予你极大的帮助。在这里你将学到如何用 DirectorX 7.0 来开发流行的游戏。本书由浅入深，详细介绍了 DirectorX 7.0 的基本技术，以及用 DirectorX 7.0 的接口来开发各种应用程序。在每一章中，除介绍各种接口的应用外，包含了关于该章主题的例程，并介绍了用 Visual C++ 和 DirectorX 7.0 SDK 来构建和调试 DirectorX 7.0 应用，并提供一些小的例程，以供读者学习使用。

DirectorX 7.0 是专为 Windows 设计的游戏开发系统，它使在 Windows 环境下进行游戏开发变得更加简单，而且游戏性能更加稳定，效果更加卓越。使用 DirectorX 7.0 进行游戏开发，就像在写其他的普通应用程序一样，而不用担心底层硬件。DirectorX 7.0提供了一种即插即用的方法，从而使应用程序的开发能够独立于硬件。利用DirectorX 7.0可以开发出具有强大的实时性能的应用程序，使得用 DirectorX 7.0 开发的游戏画面更加逼真，让你有身临其境的真实感觉。

DirectorX 7.0 可以直接访问计算机中的硬件，完全不必考虑硬件究竟是如何实现的，这就是微软声称的硬件和应用程序一致的接口，这样达到了减少安装和配置的复杂性，并有可使硬件的利用达到最优。通过利用 DirectorX 7.0 提供的 API 接口，程序员不必考虑硬件的特性，只需考虑一个虚拟的用户环境，从而实现一个优秀的基于 Windows 的高性能游戏程序，并充分利用系统加速卡和各种即插即用的硬件，以实现内嵌于 Windows 的通信服务，可以想像在一个局域网内多个玩家同时作战的刺激场面！

在本书的编写过程中，得到了许多同志的无私帮助，在此一并表示感谢。成都金点创作室的汪疆先生无偿提供了《圣剑英雄传－英雄救美》的程序代码作为本书例子，在此深表感谢。

作　者

2000.5

# 目　录

# 第一章　DirectorX 7.0 发展历程和内容简介

您一定玩过游戏吧？想想 DOS 下的游戏它们是何等的精彩和离奇，相信大部分游戏玩家都对 DOS 下的游戏津津乐道。当 Windows 的大潮向我们走来时，DOS 游戏却迷失了方向——一个具有强大图形界面的操作系统却在很长的一段时间内只能支持 DOS 的游戏开发，对游戏厂商来说，这是一个两难的抉择——我选择 DOS，因为我能全部地控制它，我能利用它开发强大的游戏，但 DOS 是一艘快要沉没的帆船；我无法选择 Windows，虽然我希望选择它，因为 Windows 是一所正在起航的巨轮，但是我没拿到上船的票，Windows 还未为我们造好舱位，它所提供的，只是缓慢而又拥挤的地下室。我该何去何从？所有的程序员都望眼欲穿了——终于，我们听到了冬日里的春雷，DirectorX 出世了。

## 1.1　DirectorX 发展历程

在 Windows95 诞生之前那一段日子里，大部分电脑游戏厂商都在兢兢业业地编写 MS-DOS 游戏程序代码，痛苦万分的程序员还在同各种各样的硬件打交道，他们一边抱怨着硬件制造厂商的朝三暮四，一边又不得不修改前辈所留下几万甚至几十万行的代码，因为对于不同的硬件，原来所开发的代码是无法运行的！虽然在用户看来一模一样的游戏，但只因为用户选择了不同的硬件，程序员还必须进行许多重复开发。

MicroSoft 终于看到了程序员们的烦恼，他们又看到了赚钱的机会——他们不会放过每一个机会——他现在告诉我们，来吧，来吧，来使用 DirectorX 吧！然后他们告诉我们一大堆的理由，诸如使用 DirectorX 可以实现游戏开发平台从 DOS 向 Windows 的转移，DirectorX 是专为 Windows 设计的游戏开发系统等等。然后我们都看到了，正像 MicroSoft 的所有其他产品一样，依靠着 Windows 这艘航空母舰，微软似乎到现在为止还是战无不胜的，DirectorX 正在变成我们的主食，在对 OpenGL 之战中，他似乎赢得了越来越大的胜利。

毫无疑问的是，程序员所希望的是一个简单、稳定、卓越、强大的系统，一个能够充分利用 Windows 优势的系统，所以对微软来说，开发 DirectorX 必须要解决的首要问题就是使 Windows 环境下游戏开发变得更加简单，性能更加稳定，效果更加卓越。

要实现以上的目标，很显然的一点便是 DirectorX 必须是独立于硬件的——当然这也是 Windows 的一贯作风——程序员不要再考虑硬件。当然还有一点很重要，为了实现游戏的实时性，DirectorX 必须提供对硬件的直接访问——这是 DOS 编程的巨大优势之一——以达到鼓励游戏开发人员利用 DirectorX 进行游戏编程。

微软告诉我们的 DirectorX 的优点如下：微软开发 DirectorX 的目的在于促进 Windows 下的游戏开发，首要的目标是实现 DOS 游戏平台向 Windows 平台的迁移。DOS 游戏编程人员不得不面对一大堆不同的硬件和各种各样的加速卡。在 DirectorX 环境下，游戏编程人员可以获得各种硬件加速特性而又不用考虑硬件的组成。DirectorX 的首要目标在于提供可移植的 DOS 的人口特性，来达到或提高基于 DOS 核心的程序性能，并免除

由于用户硬件升级所带来的不便。DirectorX 还提供了一种基于 Windows 的高效、实时地对现在的和将来的硬件的访问支持。DirectorX 提供了一致的硬件和应用程序的接口，降低了安装和设置硬件的复杂性，并可以使用硬件提供的最好特性。使用 DirectorX 接口，可以不考虑硬件的实现细节而直接利用硬件所提供的卓越特性。

DirectorX 还有如下优点：可以利用专为提高效率设计的各种加速卡，各种即插即用的硬件和软件，Windows 内的各种通讯（包括 DirectorPlay）等。

不是我不明白，这世界变化快——PC 几乎每半年就升级一次——昨天的 Petium Pro、Petium II，今天已是 Petium III 了，大概过不了多久，Petium III 也该被淘汰了。PC 机升级实在太快了！各种各样的加速卡，包括从 3DFX 到 Vodoo 等的各种加速卡充盈着我们的市场，今天的时髦到明天就可能成为历史博物馆的陈列品。

如何才能使硬件升级对程序的影响减到最小呢？DirectorX 提供了一种即插即用的方法，从而使应用程序的开发能够独立于硬件。利用 DirectorX 可以开发出的具有强大的实时性能的应用程序，使得用 DirectorX 开发的游戏画面更加逼真，让你有身临其境的真实感觉。看看 Vodoo 所给你带来的清澈透明的水世界的感觉吧！想利用这种硬件特性吗？DirectorX 可以帮你完成心愿。

DirectorX 可以直接访问计算机中的硬件，你完全不必考虑硬件究竟是怎样实现的，这就是微软声称的硬件和应用程序之间的一致的接口，这样就达到了减少安装和配置的复杂性，并有可能使硬件的利用达到最优。通过利用 DirectorX 提供的 API 接口，程序员就不必考虑硬件的特性，只需考虑一个虚拟的用户环境，从而实现一个优秀的、基于 Windows 的高性能游戏程序，而且有可能充分利用系统加速卡和各种即插即用的硬件，并实现内嵌于 Windows 的通信服务，你就可以想象在一个局域网内多个玩家同时作战的刺激场面了！

DirectorX 7.0 是微软在 1999 年 9 月推出的针对新一代多媒体计算机开发游戏和多媒体应用程序的工具包。它融合了微软的最新技术，为广大的程序员提供了一整套基于 Windows 9X 和 Windows 2000 的应用程序接口 API，使程序员能够在 Windows 系统下设计出高性能实时的多媒体应用程序，实现对内存的直接管理和对硬件的直接操作。DirectorX 使运行在微软 Windows 操作系统下的游戏比运行在 MS-DOS 操作系统下的游戏具有更高的性能和更高的交互性，甚至可以和专用的游戏机相媲美。根据微软的介绍，DirectorX7.0 支持以下的新特性：

（1）提供了一个 Visual Basic 运行库的支持，从而支持在 Visual Basic 开发环境中开发 DirectorX 程序。

（2）大量地对 DirectorX Immediate Mode 进行了改进。

（3）提供了一个新的 Director3DX 功能库，它的目的是用来帮助改进使用 Director3D 和 DirectorDraw 来设计和处理 3D 场景。

（4）DirectorMusic 增加了对 Download Music Version 2.0 的支持。

（5）扩展了 DirectorInput 的功能。

（6）DirectorSound 提高了对硬件声音能力的支持。

（7）DirectorSound 支持最新的 3D 音频技术。

（8）DirectorDraw 提供了最新的双路技术以支持动态设备。

(9) DirectorPlay 的快速启动技术。

## 1.2 内容简介

DirectorX 7.0 总共包含 7 个部分,分别介绍如下:

### 1.2.1 DirectorDraw

DirectorDraw 是 DirectorX 中的视频输入输出的基本部分。

对 DOS 程序员来说,最开心的一件事大概就是能够方便地直接访问显存,对显存进行各种直接的读写和块拷贝、块移动操作。任何一幅图像,不妨假设为 320 × 200 × 256 色,程序员可以简单地在从地址 A000:0000 开始的内存区进行直接读写操作,从而可以高效地处理视频。

但在 Windows 的 32 位环境中,我建议使用 DirectorDraw。因为 DirectorDraw 可以帮助你完成相同的工作,甚至可以做得比以前更好! 只要用户的硬件支持 DirectorDraw ,那么用 DirectorDraw 可以编制出高效的视频处理程序,以给你带来完美的享受,并完成更复杂的功能。在 Windows 环境中,DirectorDraw 可以提供一块内存区供程序员直接读写,从而完成同 DOS 相似的操作。

DirectorDraw 提供的功能非常多,其中当然包括了程序员最关心的对内存的直接读写功能和内存的块移动、复制功能,它还支持访问屏外显示内存中的位图,你可以利用硬件的位块传输和缓冲区翻转等软硬件加速技术实现快速直接存取。DirectorDraw 能让你直接操作显存,支持显卡的位块操作(Hardware Blitter),硬件覆盖(Hardware OverLay),翻转内存影射的平面(Flipping Surface)等。

当然,DirectorDraw 在提供这些特性的同时,保持了与 Windows 的应用程序及设备驱动程序的兼容。DirectorDraw 是微软专为游戏提供的一个基于 Windows 的软件子系统,它支持许多种不同的显示硬件,你可以想像的最简单的 SVGA 到支持剪切、伸展和各种色彩格式的高级显示卡。DirectorDraw 提供的 API 接口不但可以使你的程序应用各种硬件特性,还可能模拟硬件尚不支持的特性。使用 DirectorDraw 只需程序员遵守一些标准硬件约定,如: RGB 或 YUV 色彩格式及解析度。其对显存的移动操作也非常简单,无须调用特殊的过程。

DirectorDraw 目的在于提供一个基于 Windows/Windows NT 游戏图形系统,它很有可能在未来的版本中与 Director 3D 合并,组成一个强大的支持 3D/2D 操作的系统。不过微软似乎也有可能在 DirectorDraw/Director3D 的基础上再建立一个层面,如 DirectorX 7.0 中新提供的 Director3DX,来完成对 3D 的支持。

### 1.2.2 DirectorSound

声音是多媒体应用软件的重要组成部分,优美动听的音乐使游戏具有更大的吸引力。但是大量声音文件的播放占用了大量的时间,音频信号仍然对整个系统的运行产生着重大影响,所以 DirectorSound 的首要目的就是提高程序的运行速度,从而提高游戏的实时性。

DirectorSound 提供低延时的声音混合系统,硬件加速并支持对声音设备的直接访

问。同时,它还允许进行属性设置,以便应用程序开发员充分利用声卡和相关的外设驱动所提供的扩展服务。DirectorSound 给程序员提供了一个独立于硬件的编程接口,使程序员能够非常有效地应用硬件所提供的特性而不必了解硬件的复杂的细节。

DirectorSound 充分利用了声卡的各种优越性能,以尽可能的提高其运行速度,从而减少 CPU 的占用时间。可以从系统获得硬件的特性——哪些是系统支持的、哪些又是当前硬件所不支持的。程序员开发的程序将既可以支持最简单的声音回放设备的特性,也可以支持包括 3D 特性在内的各种音频设备的高级特性。DirectorSound 还支持用户的自定义属性,以支持用户开发和使用硬件的各种特性。DirectorSound 支持最新的 3D 音轨、并支持音轨的获取功能。

DirectorSound 与 DirectorDraw 一样,也是与 Windows API 相兼容的,也就是说,你可以有两种选择——使用 DirectorSound 或 Windows API 来进行编程,但是,对于复杂的、大量的、需要有特殊音效的应用程序,建议使用 DirectorSound。在微软所提供的工具包中,DirectorSound 波形声音重放设备是为支持在 Windows 9X 和 Windows NT 中开发游戏和交互媒体应用程序而设计的,DirectorSound 和 Director3DSound 允许你在同一个三维空间中同时运行多个声音文件和移动声源。DirectorX 将竭尽其能地充分利用声卡等外设来改善程序运行速度和减少声音播放对 CPU 的占用。

对程序员来说,有一点是需要注意的:音轨播放将占用大量的系统资源,所以,程序员必须小心地设计程序,因为任何意外的情况都可能导致以下结果:你的计算机播放着美妙的音乐,但是它已无暇干它该干的事情了。

### 1.2.3 DirectorPlay

DirectorPlay 的目的在于简化游戏使用通讯设备的入口,从而构成游戏外设的标准接口。DirectorPlay 所提供的通讯功能自成一体,其所实现的功能独立于用户之间实现通信的介质、协议和其他的在线控制功能。

由于任何 AI 的实现均难于抵抗人类的智慧,所以实现几个游戏用户之间的竞争是游戏发展的方向之一。个人电脑提供了一种便捷的互联方法,其所实现的功能大大超过了以前的各种游戏平台,在局域网上实现游戏用户的互联,要解决的首要问题便是解决计算机的互联问题。虽然存在各种各样的协议,但是微软的 DirectorPlay 力求针对特殊的需求从而开发出高效的系统,它屏蔽了复杂的底层的物理和信号连接,以使程序员从复杂而且无聊的底层接口中解脱出来,使设计师可以更注重于游戏的创意和设计而不是游戏的实现。在当前的发展潮流中,就有可能采用此种技术而建立成各种 Client/Server 模式的多用户的游戏程序,从而建立高效的服务器和方便的连接。

微软建立了一套称为 Director Protocol 的互联协议。在 DirectorX 6.0 以后的版本中,DirectorPlay 支持各种信令——包括 TCP/IP,IPX 等,只要取得建立的服务器的认可。其一系列新的特性甚至包括了各种的通信的同步/异步控制及信令控制,并支持建立在 TCP/IP 协议基础上的防火墙。

### 1.2.4 Director3D

Director3D 的设计是为了开发基于 Windows 的多语言的游戏接口和实现互动的图形

显示。Director3D 提供了两种工作模式：保留模式（RetainedMode）和立即模式（Immediate-Mode）。保留模式使程序易于建成一个完整的三维图形系统，它是建立在即时模式基础上的，而立即模式却使用户能完全控制的着色管道，它提供了一个与设备无关的底层硬件加速系统，Director3D 的目的在于给程序员们提供一个底层的高性能 3D 接口，使得用户程序可以与硬件相结合，从而获得独立于设备的灵活性。在 DirectorX 7.0 中，微软提供了一个新的接口——Director3DX 接口，用来简化对 Director3D 设备的操作。

### 1.2.5 DirectorInput

DirectorInput 提供了基于 Windows 游戏的输入的 API 和驱动程序，它支持键盘、鼠标和游戏操纵杆，并支持反馈的游戏设备，并可能兼容将来的输入外设。当然，DirectorInput 也是基于 COM 模型的，它提供了一种利用设备驱动程序直接访问输入设备的方法，可以替代传统的 Windows 消息机制。DirectorInput 所提供的扩展服务和优越的性能是游戏开发的强大工具，并为实现模拟和实时交互系统提供了强大的支持。

### 1.2.6 DirectorSetup

DirectorSetup 提供了基于 DirectorX 应用程序的一次性安装过程，并可以对使用 DiretorPlayLobby 对象的应用程序提供自动的 Windows 注册表进行注册。DirectorSetup 还提供了用户自定义接口的能力，用户可以通过声明一个函数原型 DirectorSetupCallbackFunction 来达到对用户接口的扩展定义。该函数原型可以将当前的安装状态信息传递给安装程序，并通过自定义的用户接口接收信息以显示安装状态，DirectorSetup 提供多用户登陆的方法以运行程序。

### 1.2.7 DirectorMusic

DirectorMusic 支持 MIDI 音频，并且支持在运行时动态作曲，当然，这种作曲并不是居于算法的，而是支持人的作曲和乐谱。

# 第二章　COM 技术与展望

自从数字计算机发明以来，在短短的 50 多年的时间里，软件开发的概念从无到有，经历了一次又一次的革命性的变化。软件开发技术也随之得到了突飞猛进的发展。从 50 年代以科学计算为主要目的的机器码“游戏”，直到 20 世纪 80 年代的软件系统开发构造，我们可以看出，软件开发面临的矛盾在不断地发展演化。

软件工程作为计算机科学的一个重要方面，一直寻求着软件开发设计的工程化的方法。但是，由于软件设计完全是人的纯脑力劳动结果，因而给其度量和过程控制带来很大的困难。开发人员的经验和技巧很不容易总结，也难以推广到整个软件工业界进行共享。

## 2.1　COM 技术的发展背景

进入 90 年代以后，网络计算、信息高速公路，以及多媒体等技术的出现和发展给软件工业带来了巨大的冲击，人们对软件开发的认识从单一系统的完整性和一致性，向着群体生产率的提高、不同系统之间的灵活互联和适应性而变化。软件的非功能性需求比以往得到更大的重视。

早期的 OO(面向对象)技术经过了变形和发展，衍生到软件体系结构设计和代码设计之中，近年来出现了部件化软件和 Design Pattern 等新的技术概念。过去的 10 年里，计算机硬件和软件技术水平的持续提高，软件工业界给 PC 机用户桌面和互联网络提供了大量功能强大而结构复杂的应用程序。正是由于如此复杂的程序功能和结构，给软件开发者、用户和软件供应商带来了难以想象的困难。今天的应用程序庞大而复杂，动辄上十兆或百兆的体积，开发周期也越来越长，而且更大的问题是维护的困难和昂贵，以及功能性扩展的风险。

一个应用软件程序作为一个整体，在发布之前就集成了广泛的使用特性。但是，其中的许多特性不能独立地被删除、升级或者替代。对于其它应用程序而言，即使是使用的同一种编程语言，并且在同一台机器上运行，也很难利用该程序的数据和功能。不同的应用程序之间就好象陌生人一样，被完全地隔离开来。

操作系统也面临着同样的问题。它们的结构不够模块化，也就不容易以非常简洁的方式来重写、升级或者替换某些服务功能。无论是服务器应用程序还是客户应用程序的编写，都受到这种运行环境的极大限制，即使程序本身具有很强的开放性，也会由于操作系统和网络提供的不同方式的服务而以不同的形式展现在其它应用程序面前。软件开发的编程模型强烈地依赖于提供的服务方式，比如通过动态链接(dynamic linking)提供的相同地址空间的服务，同一机器上不同进程间的服务，从操作系统得到的服务，或者跨越网络在另一个机器上运行的服务等。

总的说来，复杂软件的设计和实现仍然非常昂贵，并且容易出现错误。许多的努力和消耗都集中于相同的设计概念和代码部件的重复开发上。而硬件结构的多样化、操作系

统的纷繁复杂和通信平台的一不致使开发出具有设计正确、可移植、高效而低代价的应用变得非常困难。

## 2.2 新的软件开发模式

可以看到,环境的发展使得旧的软件设计方法不能适应如今的需求。既然CPU如此便宜快捷,网络无处不在,为什么不将处理能力分散到最适合的地方呢?因此,强烈需要一种新的软件开发模式,它必须具有分布式的计算、客户/服务器结构、模块化和部件化的特点。具体来说,要求:

提供一套手段,能够寻找定位及使用其它应用程序、操作系统的服务(以一种对于服务的来源透明的方式),和服务提供者有效地交互通信,并且以版本兼容的方式实现服务提供程序的扩展和更新。

在操作系统和应用程序服务体系中,使用面向对象的概念,从而更好地应用各种面向对象开发工具,通过更强的模块化来克服软件高度复杂所带来的开发维护的困难,更加有效地利用已有的解决方案,设计构造出更加自持(self-sufficient)的软件部件。

通过客户/服务器的计算结构,来利用越来越强大的桌面设备、网络服务器等资源。

使用分布式计算,在与用户和应用程序保持单一的界面接口的同时,突破本地地址空间限制,充分利用网络服务环境,但不需要考虑空间分布、机器结构或实现环境的影响。

## 2.3 解决方案:部件化软件(Component Software)

面向对象软件开发技术在80年代得到了长足的进步,由此出现了许多面向对象的软件开发工具和应用程序。但是,面向对象的软件开发设计并没有发挥出它应有的最大力量,主要是因为还不存在一个标准的框架,使得不同开发商所提供的软件对象不能够在同一地址空间里互相交互合作,就更不用说是跨越线程空间、网络空间,或者机器结构的边界了。于是,面向对象的软件设计给我们带来的是一个个分离在应用程序的汪洋大海中的孤立的对象体。

解决问题的方法就是使用和开发可重复利用的软件部件。软件部件,是指在软件系统设计中能够重复使用的建筑模块[1]。部件包装了一系列互相关联的操作和服务。软件部件与其它的可重复使用的软件模块的区别在于,它既能够在设计时进行修改,也能够在二进制执行模块时修改。一个以二进制形式实现的软件部件能够有效地嵌入其它开发商开发的部件之中。

打一个比方,现在的软件工业所处的开发程度,就如同几十年前人们忙于用分离电子管或晶体管搭建的复杂电路一样。显然分离部件结构给开发和维护都带来了巨大的困难。软件部件就如同集成芯片(IC),利用少量IC就能够快速地组建一个功能相当复杂的电路版,同时也具有较大的可靠性。随着IC芯片的体积越来越小,而功能越来越复杂,一个由IC组成复杂的电路版又可以集成为一个更大的IC芯片。

部件标准(Component standards)规定了如何建造及互连各种软件部件。部件对外部世界的表现完全不受本身内部实现的约束。这种对外部接口和交互协议的强调,使部件

标准和传统通信协议之间有一定的区别。很好的部件标准应该使部件的交互、替代、升级、定制、以及组合成更大粒度的部件或应用成为可能。因此，部件标准是软件开发工作从可复用部件技术得到应有效果的必要保证。

部件软件的发展非常迅速，现在的情况就和几年前的情况大不相同。那时，几个大商家都独立推出了自己的一套部件标准，规定了部件的建造和互连机制。如今，Sun 的 JavaBeans 成为了 Microsoft 的 DCOM（Distributed Component Object Model）的头号竞争对手，而 Component Integration Laboratories 的 OpenDoc 似乎已经被挤出市场。很重要的一点是，部件软件正从最初的以桌面复合文档为中心，向着包含分布式服务部件的企业应用转化。

由于涉及分布式服务部件的使用，部件概念发展成为分布式部件模型（DOM，Distributed Component Model）。当然，它继承了部件软件的所有功能的优点，并且引入了强大的网络特性。分布式部件模型不仅会改变应用程序本身的结构，同时也会改变网络的运行和管理机制，这一点应该引起充分的重视。

从应用程序结构的角度来说，DOM 能够很好地实现模块化，以及各模块的复用和交互连接。这些功能在下文将进行更为细致的解释。而从网络的角度讲，DOM 的优势也很明显，比如客户/服务器结构模型、网络的带宽利用等。更为吸引人的是，DOM 有可能改变网络本身。

现在，网络管理功能一般是相当集中的。这一点是计算机 IP 网络和通信 ATM 网络的最大区别。在 ATM 网里，智能不是在使用者的终端，而是分布在整个网络内部之中。而 IP 网似乎本身没有什么智能，它完全靠各个终端计算机本身和几个特定的服务器。因此，网络的质量保证、调度、管理、计费等都不是很理想。

由于 DOM 技术的采用，能够将许多的智能分散于不同地点的部件之中，比如放在 switching hub 里，能够使得它越来越自动化。网管员能够在各地加载强大的网管工具，使得各个路由节点能够自动地动态调整带宽，以适应本地的网络条件的变化。不同地点的 object agent 能够对网管员的查询作出更为细致的反应，这就能够帮助问题的定位和解决。我们也发现，switched net 和 virtual LANs 的概念也和 DOM 很相似。

DOM 一个关键的思想是：让处理能力和智能在对其最恰当的地方工作。因此，很容易建立起逻辑子网，使带宽不被浪费在传送无用的消息上。DOM 作为软件部件对象，能够方便地维护状态信息，这也能够弥补 TCP/IP 的一个弱点，即 HTTP 没有状态信息，但不能维护其建立的连接。并且，DOM 的概念可以说和网络计算机的目标不谋而合。

在此基础上，分布式发展的将来，应该使操作系统并不限于连接之间的建立，而应该引入更加高层的能力，比如更好的分布安置对象、平衡对象的大小、可能需要的带宽及保证功能的实现等。而这些功能，现在只能依靠手动调整和设置。同时，引入分布式网管部件，底层部件能够自动进行自我调整，为带宽资源进行竞争，网络管理员仅仅需要调整一些高层参数。底层部件的直接对话，能够很大程度上将网络的运行管理和外界屏蔽开来。

## 2.4 分布式部件标准

ActiveX 技术建筑在微软自己的对象标准之上，其优势在于 Windows 的使用数目。

其底层技术是 Microsoft 的 COM(DCOM)标准。本文以 DCOM/COM 标准的介绍为主，因此下面对其特点进行比较详尽的描述。

COM 作为一个面向对象的编程模型，目的是提高软件的交互协同工作能力，使其而不受开发方法和语言、运行地点和环境的限制。它定义了应用程序作为软件对象(由一组以某种形式相关的功能以及与这些功能相联系的状态所组成的集合)而互相连接的实现机制。

从某种意义上说，COM 就像传统的系统服务 API，提供某些操作以使服务的客户能够和多个服务器建立连接。但是一旦建立连接，COM 就立刻退出这个通信框架，而完全由客户和服务器直接进行交互，从而避免了因为强制通过中央处理 API 产生的许多不必要的附加代价的损失，如图 2.1 所示。

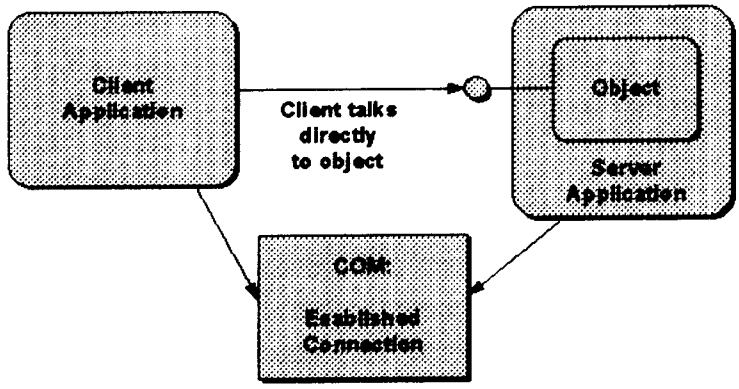


图 2.1 COM 部件模型

应该说明的是，COM 并没有规定软件的结构形式，它也并没有帮助如何更加快捷高效地实现一个复杂的功能性需求。相反，它往往还给软件编程带来一定的复杂性。因此，软件设计本身很大程度上仍然依赖于其它的软件设计方法，比如传统的结构化设计，或者新近出现的 Design Pattern 等。但是，COM 提供了一套二进制接口的标准，相对其它对象模型来说，更好地解决了软件的版本化及进化问题。它使得对象功能可以递增性地改变或进化，而无需该对象的当前客户采取同步进化措施。对象能够在提供更好或更新的接口以及和新客户进行通信的同时，保持和旧客户的通信接口。为了避免这种不必要的 overhead 的服务连接，并解决版本化问题，COM 具有以下几个显著的特点：

“部件对象”的包装使得部件能够重复创建和使用。面向对象软件编程适于编写一些灵活而强大的程序代码，并且能够被自己和其它编程者重复使用。其中的一个重要概念就是“包装”。所谓对象，是指许多函数(它们表示该对象能够做些什么)和这些函数的相关状态(即数据)的集合体。包装完成了算法的实现及对数据结构的操作以及对象的使用客户之间的屏蔽。COM 就是将客户和对象间的这种包装进行标准化。

为交互接口定义二进制标准。由于标准是二进制形式的，因此对象的编写和使用完全不受编程语言的限制，而且执行效率也相对更高一些。标准包含了各种网络协议的低层实现以及通用的通信接口，因此客户和服务器都不需要包含许多例行的通信实现函数。这使应用程序开发者能够将主要精力集中于软件功能的实现上。

传统的操作系统主要以 API SDK 的形式向应用程序提供服务。而 COM 力图将对

象化的概念延伸到系统中,零散的 API 函数被组织成逻辑上比较紧凑的形式。所有的客户、服务器和系统间都通过一组函数(在 COM 中称为方法或接口)实现交互。所谓接口(interface)就是软件部件间所必须遵守的协议,它由较少量但又有用的逻辑相关的操作组成。接口严格地说明了操作实现的功能和期望的响应,而操作间的相关性使得设计者能够顺利地以整体的方式实现所需功能。COM 使用接口具有以下优点:

(1) 客户/服务器都能独立地完成升级。通过 QueryInterface 机制,对象可以同时提供多个服务接口,并且软件开发者可以随时动态地增加新的服务接口,而不同的接口又被旧客户和新客户同时使用,从而保证了向后兼容性和版本化的问题;

(2) 对于同一地址空间地客户服务器,能够实现高效简单的对象交互,因为接口调用就是以间接的函数的指针形式实现的,而且某个接口的指针只需调用一次 QueryInterface,之后可以反复使用,所以它可以比一般的 C++ 函数继承重载效率更高;

(3) 对象设计完全实现地点的透明性,COM 能够截获对某对象的接口调用,转而以 RPC(Remote Procedure Call)实现网络传输,因此对象设计完全可以假设同一进程空间模型,同时 COM 允许用户定制调度形式(custom marshaling)以使对象在网络调用时采取特殊的处理,所以设计通过两个层次来完成:首先按自然的本地流程实现对象功能,然后再考虑网络效果,而完全不影响前一层次的设计;

(4) 不依赖任何设计语言,只要该语言能够使用指针变量并用之调用函数,比如 C, C++, Pascal, Ada, Smalltalk, 甚至 BASIC 环境等。

ActiveX 是一个真正的系统对象框架。它使用全局唯一的标识(GUIDs)-128-bit integers 来分辨不同的对象类和其接口,利用 containment/delegation 和 aggregation 机制避免传统的语言式继承而实现了代码复用,对相同进程、不同进程或不同网络地址等情形具有相同的设计模式,以参考计数(reference counting)的方式实现对象的生存期控制,并在多个层次上为对象级的安全性奠定了基础。

(5) 提供分布式计算能力,允许软件设计被分隔为多个不同的部件运行于多个机器,而对网络实现透明。

COM 协议规定了接口的二进制标准,它包含有:

- 通过 QueryInterface 实现基本接口交互
- 通过 refefence counting 机制管理对象及其资源,包括和多个客户连接
- 内存在独立开发部件间进行传递交换时的分配释放原则
- 一致性和错误诊断工具

作为二进制标准的同时,COM 也包含部分的系统代码实现,这就是 COM Library(如图 2.2 所示),它主要包含:

几个用来创建 COM 对象应用程序的底层 API。对于客户程序,它提供基本的对象创建例程,而对于服务器,它提供其对象的声明注册例程。

对象实现的定位服务。它根据唯一的类标识,通过系统注册表来确定那个服务器实现了该类和该服务器的地址。

透明的 RPC 实现,以支持在本地或远程服务器中运行的对象。

提供一套应用程序如何控制其进程空间里内存的分配和释放的标准机制。

不同的平台,分别只需 COM Library 的一个实现。比如 Microsoft 已经提供了 COM