第一部分

理解 Windows NT 概念

第一章　WINDOWS——过去、现在和未来

你将开始你的个人编程生涯最激动的新时代！这是因为 Windows NT 为你提供了一个坚实的基础。Microsoft 的新一代操作系统引入了许多开发人员曾经梦想的高精技术。

例如，由于 Windows NT 是一个完整的 32 位操作系统，因此你的应用程序不再需要使用令人困惑、经常冲突的内存管理模式。加上许多新的图形函数，如 Bezier 曲线，完整的 32 位图形例程将极大地提高程序的性能。另外，由于 Windows NT 引入了新的 Unicode 标准，使得程序员开发的程序可以在世界各地的不同语言版本上运行。Microsoft 以她的承诺和业已证明的能力为软件开发做了可靠的保证。

对 Windows NT 的认可始于对这个产品目标的了解。Windows NT 操作系统是 Microsoft 公司为适应下个世纪程序员需要的高档计算平台而设计的。Windows NT 是为开发人员、最终用户、公司管理信息系统(MIS)管理员、网络服务器管理员而设计的。

这个 32 位的、抢先多任务的操作系统集易用性、Windows 图形界面和先进的操作系统技术于一体。由于它运行在基于 Intel 80x86 微处理和 RISC 结构上，Windows NT 极大地释放了先进的 PC 硬件结构所蕴含的潜力。它甚至还支持对称多处理结构。

当 Windows NT 安装到你以前购买的硬件上时，并不意味着你以前的软件投资作废。Windows NT 支持所有 MS－DOS 和 Windows 应用程序的运行。

Microsoft 对向上兼容性的承诺使 Windows NT 对未来的应用程序开发仍是一个安全的平台。Windows NT 的硬件接口使它能充分利用所有的 I/O 技术，从 CD 声频到实时视频，并且能适应未来的发展。

Microsoft Windows 的巨大成功已经证明一个真理：使人们比以前更有创造力。

1.1　过去——从 MS－DOS 到 Windows

Windows NT 无疑是这个时代最活跃的最终用户环境和开发环境。然而，要全面地了解 Windows NT，你需要回顾它的历史。下面是基于 PC 硬件和软件革命性变化的一个简略回顾。

历史回顾

我们从一个普通人对计算机的印象开始。20 世纪 40 年代，计算机还在使用硬拷贝设备(用现在的标准来衡量，它就是古董啦！)，它需要被终用户翻阅大量的打印输出。自然，这非常浪费时间，而且对于个人在成千上万张硬拷贝中查找其所需的少量数据也相当困难。另外，这些古老的输出设备在容量，产生图形输出等方面也非常落后。

为了能进行简单的图形输出，计算机使用了阴极射线管(CRT)系统，该系统最早是由 MIT(马塞诸塞州立大学)为研究飞机的控制和其稳定性而于 1950 年研制出的。之所以把这个显示设备与计算机联系起来，是由于当时需要缩短用户输入和计算机输出之间的时间。

50 年代飞机也安装了 SAGE 空中防御系统，它把雷达信号转变成粗糙的计算机产生的图

像，这个新的图形系统也是第一个用光笔在屏幕上选择符号的系统。

60年代初期，一位MIT Ph.D候选人通过开发Sketchpad画线系统改进了图形输入。这个系统允许用户在屏幕上各点之间通过光笔连线来作图。这个图形系统不仅可以画线，也可以画建筑平面多边形，简化了复制简单物体的复杂图形的生成。

早期的CRT可以在显示屏幕的任意两点间画一条直线。然而，由于图像很快就消失，它又不得不在每秒钟内重画几次。60年代，这个用来存储线的端点的存储器和用于快速重画线的硬件都非常昂贵。比如，1965年IBM推出第一个适用于这种图形显示的CRT产品。单独的显示部分就要100000美元。由此你可以理解为什么当时这种设备不普及。

三年后，Tektronix开发出了第一个存储管CRT。这种类型的CRT具有保持画面的能力，直到你不再需要它为止。由于这种显示的构造的改变，昂贵的存储器和重画所用的硬件都可以省掉，从而使显示部分的价格降到15000美元。以这个价格，Tektronix显示马上取得了成功。

70年代，由于存储器和硬件逻辑器件的费用戏剧性地降低，图形开发环境得到了巨大的推动。这些变化导致了存储器、激光扫描等快速发展，从而可以显示出真实效果的带有灰度和彩色的图像。

到了80年代，显示监视器不再是数字的。比如，来自IBM显示图形阵列(VGA)的输出就是模拟的。为保证和现有环境兼容，它可以支持所有的以前的显示模式。因此，单色、彩色图形适配器(CGA)和增强型图形适配器(EGA)模式都可以在VGA适配器上使用。

将来会出现标准的、高分辨率的、带有大彩色调色板的监视器。随着这些新技术的开发利用，也许将来写Windows NT应用程序将无须代码。

软件从BIOS 10H到Windows

要理解图形软件开发的历史过程，首先让我们花点时间讨论一下BIOS(基本输入/输出子系统)。构成任何一个小系统的是一组BIOS例程，它们被存储在ROM(只读存储器)中。这些子例程提供了计算机同其标准硬件的一个接口，这些标准硬件包括时钟、键盘、软件和硬盘以及与Windows NT相关的硬件，还有显示子系统。

显示BIOS例程由一组执行基本显示任务的简单函数组成，比如往屏幕上写字符串、刷新屏幕、改变颜色等等。

中断10H

以前，要完成任何实时图形处理，程序员必须用汇编语言编写汇编程序来访问这些BIOS例程。要访问BIOS的显示部分，使用8086系列微处理器的程序员必须发出一个10H中断。

今天，ROM BIOS支持多个在执行中断10H时被访问的显示输入/输出功能。这些功能都已经被编号；在执行中断10H之前，程序员必须先把你希望的功能的号码放到适当的寄存器中。比如，ah中。

当中断执行时，微处理器上的其它寄存器可能含有由BIOS例程传送过来的其它参数。如果被调用的中断10H返回数据给程序，则它一样是把数据放在微处理器上的一个或多个寄存器中。然而，这个寄存器的基、参数传送规程根本上还是由你用的汇编语言程序决定(注意：由于Windows NT是一个完整的操作系统。所以Windows NT应用程序要调用新函数以替代MS-DOS中断的调用)。

高级语言

很可能你第一次接触计算机图形用的是 BASIC 语言。你可能已经用过像 LINE，CIRCLE，COLOR 等这样的命令，接着你可能转向学习其它语言，比如 Pascal。这个语言会给你的程序更多的结构，但在图形能力上没什么进步。

随着 Borland 的 Turbo Pascal 的引入，程序员有了一个惊人丰富的图形环境。从 Turbo Pascal 4.0 开始，许多复杂的特征，比如视口、剪裁板、用户自定义填充模式，三维条等等都可以使用。

另一种广受欢迎的语言是 C/C++，由于它把汇编语言和高级语言结合在一起，所以它被公认为最好的语言。C/C++提供给程序员汇编语言的硬件访问能力和强大的逻辑结构。C/C++程序员现在可以访问整个用于图形应用程序的非常复杂的图形例程库。

并行处理

在这个简单的历史回顾中，我们忽略了一个方面，那就是关于并行图形应用程序的研究。在此以前编写的任何程序都独占整个显示子系统，包括所有寄存器、存储器和显示设备。

随着用户对软件和硬件和期望的提高，要求程序能够实现并行运行。用户在编辑一封信之前、不再等待数据库存储结束——他们希望这两件事能同时做。这个需求引出了像 Quarterdecks' DESQview 和 Microsoft Windows 那样的产品——它给用户多任务的能力。

比多任务能力更重要的是 Windows 提供给用户的图形设备接口(GDI)。通过使用 Windows GDI 函数，程序员可以写出一个对用户看起来很熟悉的应用程序(它有菜单、滚动条、消息框等)。另外，应用程序还可以改变其界面大小、移动、变成图标、成为背景，甚至还可以和其它程序一起运行。最后，GDI 函数还允许一个应用程序与另一个通讯，虽然许多应用程序还未允许充分利用这个强大的性能。

这个关于软件和硬件发展的简短的历史概述说明软件和硬件的发展变化是如此迅猛。今天，用户和应用程序开发人员想有一个具有多任务能力并且不依赖于硬件的面向图形的用户界面。Windows 就适合于这种潮流。

1.2 现在——WINDOWS NT

如果你过去用过 Windows 的 DOS 版本，你可以问自己一个问题：除了 Window 3.x 提供我们的东西之外，我还需要什么？和 Windows 的 DOS 版本一样，NT 仍有很多地方可以改进。

Windows NT 的历史

Windows NT 准确的设计时间是 1988 年初。开发组由 Dave Cutler 领导，并汇集了许多具有设计 Windows，UNIX，VMS 和 OS/2 经验的出色的软件工程师。Windows NT 首次成功的启动是于 1989 年末在一个 Intel i860 上实现的。1990 年初，Microsoft 决定 Windows NT 操作系统基于 Windows-based 32 位 API(应用程序界面)上运行。Microsoft 把其设计的硬件要求定位于 Intel 386/486 和 RISC 结构。

什么是 Windows NT

和传统的 MS－DOS 操作系统相比，Windows NT 操作系统为用户和程序员提供了相当多的好处。虽然三个主要性能(面向图形的用户界面、多任务和不依赖于硬件)都不是新的，但它的革新在于它把这三个性能结合在一起，构成单独一个微处理器操作系统。

图 1－1 说明了 Windows 3.1、Windows for Workgroups、Windows NT、和 Windows NT LAN Manager 之间的关系。我们可以注意到，Windows NT 包含了所有 Windows 3.1 和 Windows for Workgroups 的性能，并能运行这些环境上的应用程序。

下面的讨论将解释一些使 Windows NT 成为下世纪最好的开发平台的特点。

标准的用户界面

在 Windows NT 提供的三个主要性能中，标准的面向图形的用户界面是最引人注目的、当然、对用户来讲也是最重要的。这种统一的用户界面使用了图片或图标(icon)来表示驱动器、文件、子目录和许多操作系统命令和动作。图 1－2 显示了一个典型的 Windows NT 文件管理器。

程序通过标题条来标识，许多基本文件操作可通过鼠标点菜单来完成。多数 Windows NT 程序都有一个键盘接口和一个鼠标接口，虽然 Windows NT 程序的多数函数可以由键盘控制，但对许多任务来说，使用鼠标往往很容易。

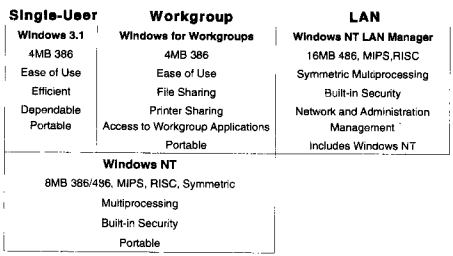

图 1－1　Windows 3.1、Windows for Workgroups、Windows NT 和 Windows NT LAN Manager 之间的内部关系

由于 Windows 程序都很"相像"，所以用户不再需要花费很长时间来学习如何使用新的应

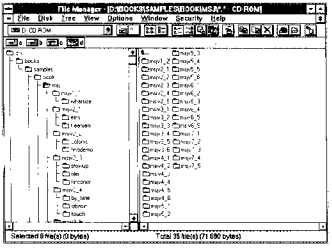

图 1－2　Windows NT 文件管理器

用程序。图 1－3(来自 Windows NT Paintbrush)和图 1－4(来自 Windows NT Notepad)说明了这种"相像"；注意看命令 File 和 Edit 以及 Maximize/minimize 按钮。

对于程序员，这种统一的界面可通过使用 Windows NT 的内部子例程来构造菜单和对话框得到。所有菜单都有相同的键盘和鼠标接口，这是因为 Windows NT 处理这些工作，而并非应用程序在管理键盘和鼠标。

多任务

一个多任务操作系统允许用户有几个应用程序或同一应用程序的几个实例并行运行。图 1－5 显示了几个 Windows NT 应用程序，每个应用程序占用了屏幕的一块矩形窗口。在任何时候，用户都可以在屏幕上移动窗口，改变窗口的大小，切换应用程序，和交换窗口间的信息。

这个例子(图 1－5)显示了并行运行 4 个程序时的情况。Windows 3.x(Win16)和 Windows NT(Win32)间最主要的差别之一就是 Windows NT 是真正的占先(prempitve)多任务系统。虽然图 1－5 在 Windows 3.x 和 Windows NT 中看起来是相同的，但其在后台所做的事情则完全不同。

使用 Windows 的 DOS 版本时，你可以装载几个应用程序，但在任意时刻他们中只有一个可以真正地使用处理器。区别并一个正在处理的任务和一个仅仅在运行的任务是很重要的。在 Windows NT 下，处理器自动地在各应用程序间切换，而无需支持正在执行的程序释放处理器控制权。

一个应用程序的其中一个状态叫活动态。一个活动的应用程序就是一个正在接受用户输入的程序。正如只有一个应用程序在给定的某一时间片内能够被运行，尽管可以运行多个并行的任务，但在某一时间内只能有一个活动的应用程序。分配处理器的时间是 Windows NT 的职责。Windows NT 通过输入和消息队列来控制处理器的共享。

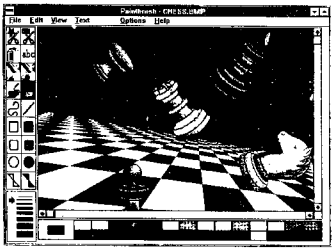

图 1－3　Windows NT Paintbrush

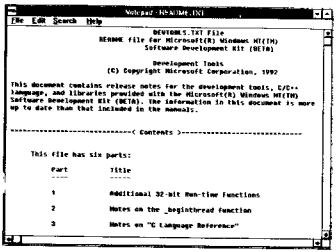

图 1－4　Windows Notpad

在多任务操作系统之前，应用程序独占计算机的所有资源，包括输入/输出设备、内存、显示器，甚至CPU本身。然而，在Windows NT下，所有这些有用的资源都必须共享。例如，一个标准的C++程序不再能够访问不被系统或程序使用的所有内存。

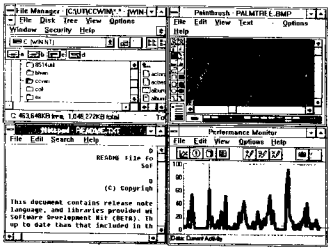

图 1—5　并行运行 4 个 Windows NT 应用程序

内存管理

内存是 Windows NT 下的最重要的共享资源之一。由于同一时期运行多个应用程序，它们之间就必须协同共享内存，以防止耗尽资源。另外，当新的程序运行和旧的程序结束时，内存变得非常零散。Windows NT 可以通过移动内存中的代码和数据块，将非常零散的空闲内存空间合并。

Windows NT 允许应用程序超过内存的限制，也就是说，一个应用程序的代码可以超过某一时刻的空闲内存大小。Windows NT 能够从内存中丢弃一部分代码并在以后再从程序的执行文件中将其重新载入内存。

在某些情况下，一个用户可以同时运行某一程序的几个实例。为了节省空间，Windows NT 共享相同的代码，运行在 Windows NT 下的应用程序甚至可以共享其它.exe 文件中的例程。这些包含共享例程的文件我们称之为动态连接库。

Windows NT 采用了一种在运行时连接程序和动态连接库中例程的机制(Windows NT 本身就是一组动态连接库集)。为了实现这个目标，Windows NT 采用了一种新的.exe 文件格式，我们称之为"new executable"格式。这些文件包含 Windows NT 所需的管理代码和数据段以及执行动态连接的信息。

输入队列

正像内存是 Windows NT 下的共享资源一样，键盘和鼠标输入也是共享的资源。C/C++ 程序不再能够通过 getchar 函数来直接读到键盘的输入。

在 Windows NT 下，一个应用程序不能直接从键盘或鼠标上读入数据。而是由 Windows NT 接受所有的键盘、鼠标和定时的输入而形成一个系统队列。这个队列的职责是将系统队列

中的输入送给相应的应用程序队列。当一个应用程序准备接受输入时，它从本身的队列中读入，并发送消息给适当的窗口。

这个输入是一种统一的格式提供的，我们称之为输入消息。所有的输入消息无非是指明系统时间，键盘的状态，按键扫描码，鼠标的位置，被按下的鼠标键，以及哪个设备产生了这个消息。

所有的键盘、鼠标，定时器消息都有相同的格式，并以相同的方式被处理。另外，对每一个消息，Windows NT 提供了一个与设备无关的虚拟键码来标识键（这与何种类型的键盘无关），与设备相关的由键盘产生的扫描码和键盘上的某些键的状态，如 NUMLOCK，ALT，SHIFT 和 CTRL。

键盘和鼠标做为一种共享资源，一个键盘和一个鼠标必须能对运行在 Windows NT 下的所有应用程序提供输入。使用键盘时所有输入消息都发送给当前的活动窗口。然而，鼠标消息则有点不同；它们发送给处于鼠标光标下的那些窗口。

定时器消息和键盘及鼠标消息非常相似。Windows NT 允许一个应用程序设置一个定时器，以便它的其中一个窗口定时地收到消息。这个消息直接送到程序的消息队列。还有一些消息进入程序的消息队列。例如当调用了某些 Windows NT 函数时会产生这些消息（我们将在例子中来分别讨论）。

消息

Windows NT 的消息系统是在多任务环境中用来传送消息的基本结构。从你的应用程序角度来讲，一个消息可以被看作一个“通知”，它告诉你，你所关心的某些事件已经发生。这些事件可以是由用户“启动”，比如点或移动鼠标，改变窗口大小，或选择一个菜单项。另外，signaled 事件也可以由应用程序产生。比如，一个基于图形的 Spreadsheet 可能已经重新计算了一次，得到了一个新的结果，这时就必须要修改饼图，以表达最新的结果，那么应用程序就要向自己发送一条“update window”消息。

Windows NT 也有可能产生一些消息，比如在“close session”时，Windows NT 必须通知每个想要关闭的应用程序。最后一个主要的消息来源可能是来自仪器监测程序，比如，一个重要的化学处理已经到了指定的温度，仪器监测程序就会发出消息，使应用程序作出相应的反映。不考虑消息的来源，你的程序也必须对消息采用适当的动作。

当你考虑消息在 Windows NT 的职责时，这有两点要记住。首先，Windows NT 之所以能够实现多任务，它就是通过消息完成。消息系统使得在 Windows NT 下不同应用程序间共享处理器。每次 Windows NT 发送一条消息给应用程序，它也占用处理器时间。

Windows NT 的第二个职责是允许应用程序响应环境中的事件，这些事件可以由应用程序本身、其它并行运行的应用程序、用户或 Windows NT 来产生。每次一个事件发生，Windows NT 都记录下来并把适当的消息送给有关的应用程序。因此，简言之，你可以认为 Windows NT 应用程序最主要的任务就是处理消息。

设备无关性

Windows NT 提供的第三个主要性能是硬件设备无关性。前面，你已经了解到，与一个应用程序相关的硬件设备的发展变化是那么迅猛。特别是，你已注意到硬件设备无关给显示系统

带来的好处。这节我们将集中讨论设备无关是如何实现的。

Windows NT 允许程序员使用各种型号的显示器、打印机、输入设备和 CPU。通常，一个非 Windows NT 应用程序在编写时必须为每一个可能使用的设备包含其设备驱动程序。显然，这是软件开发中效率低的主要原因之一。

特别是，要想使一个应用程序具有在任何打印机上打印的能力，程序员就要为不同类型的打印机提供一个对应的驱动程序。这就需要许多软件公司必须一遍一遍地写相同的设备驱动程序，比如，一个 Hewlett－Packard LaserJet 4 驱动程序是针对 WordPerfect 的，Microsoft Word 要一个，AmiPro 要一个等等。

在 Windows NT 环境下，每一个设备驱动程序——无论是针对显示器、打印机、键盘，还是鼠标——都只需写一次。替代每个软件公司都写其自己的完整集合的情况，硬件公司仅为系统写一个驱动程序。Microsoft 在 Windows NT 中包含了许多驱动程序，其它的可用驱动程序来自硬件和软件制造商。

Windows NT (hppcl. drv) 的一个很好的改进是：它支持外部软字体。比如，Hewlett－Packard LaserJet 驱动程序已经具有这种能力。当 Windows NT 被安装时，它为当前系统中的每个设备安装一个驱动程序。无论何时，当一个应用程序发出打印或绘制命令时，Windows NT 就会通过适当的驱动程序输出。把可能使用的驱动程序直接合并到系统中减少了大量的编程负担。

对于程序员来讲，应用程序开发变得更容易了。应用程序只与 Windows NT 交流，而与任何指定设备不直接接触。这就使得程序员无需了解打印机如何工作。如果应用程序发出命令要绘制一个实心矩形，Windows NT 就会在最新的“3－D Surrealistic Animation Holographic SilkScreener”上完成，每个设备驱动程序都和任何 Windows NT 应用程序一起工作。开发者节省时间，并且用户不再担心是否每个新的应用程序将支持他们喜欢的牌子的数字化仪。

Windows NT 通过指定硬件必须有的性能来完成这种设备无关。和 Software Development Kit (SDK) 一起使用，Windows NT 会提供你的应用程序一些必须的是以保证适当的例程可以执行的最低性能。

任何 Software Development Kit 例程，不考虑其复杂性，都有能力把自己分成一个设备需要的操作的最小集合。比如，并非任何绘图机都能画椭圆。作为一位应用程序开发者，无论如何，你仍可以用例程画一个椭圆，即使绘图机没有特殊的椭圆功能。由于任何同 Windows NT 相联的绘图机必须至少可以画直线，Windows NT 就可以把椭圆分成一组短线组合而成。

提到输入，Windows NT 可以提供一个确保你的应用程序将只接收有效的、预定义的输入的最小集合。Windows NT 已经定义了一个合法击键集合；换句话说，Windows NT 定义了 Windows NT 应用程序可能接收的所有击键。

这个有效集合与键盘产生的内容很相似。如果某个生产商生产的键盘所包含的键，有些不在 Windows NT 可接受键的集合中，那么生产商就应该提供把这些“非法”键转换成 Windows NT 可以接受的合法键的软件。

这种预定义的 Windows NT 输入覆盖了所有的输入设备，比如鼠标。因此，即使某人想开发一个六键鼠标，你也不必担心。生产商将再次提供给你软件，把所有的鼠标输入转换成 Windows NT 预定义的鼠标按键形式。

动态链接库

Windows NT 的多数功能都由动态链接库(DLLs)提供,通过提供一个强大的和灵活的图形用户界面增强了基本的操作系统。动态链接库包含那些当应用程序被装载时(动态地)就和应用程序链在一起的预定义好的函数,代替了在.exe 文件产生时的链接(静态地)。

Windows NT 的动态链接并非来源于动态库的思想。C/C++语言对于不同的系统非常依赖库去完成标准功能。比如,链接程序就拷贝 C/C++运行时间库的函数,比如 getchar 和 printf,到某一程序的可执行文件中。

函数库使得每位程序员不必为一个普通操作重编一个新过程,比如读一个字符或格式化输出。程序员也可以建立他们自己的包含有特别功能的库,比如改变字体和校正正文,把经常使用的函数变成一般的例程会减少你编程的负担。

就像我们前面所说,Windows NT 库是动态链接的。换句话说,链接程序不把库函数拷贝到程序的执行文件中。也就是说,当程序正在执行时,它才调用库中的函数。自然,这会节省内存。无论有多少应用程序在运行,在某一时刻随机访问存储器(RAM)中只有一个库的拷贝份。

通过改变库的格式,Windows NT 库应用程序更加广泛。当它们保持和其它 DOS 可执行文件相同的格式时(虽然他们不能使自己被执行),他们可以拥有 DOS 可执行程序能拥有的任何东西。除了函数之外,库中也能包含数据,甚至像鼠标形状和位图这样的图形资源。Windows NT 库扩展了共享资源的范围并为程序员节省了许多时间。

从技术上讲,当一个应用程序调用 Windows NT 的一个函数时,编译程序必须产生机器码来调用位于 Windows NT 众多库之一的该函数。这就出现了一个问题,因为程序真正在 Windows NT 下运行以前不知道该 Windows NT 函数的地址。

这个问题的解决办法就被称为延迟合并或动态链接。新的连接程序允许一个程序调用在连接时间内不能完全连接的函数。只有当程序被装入内存运行时,该函数才真正地同程序连接上。

包含在 Windows NT 工具中的是特殊的引入库(import libraries),它用来为一个 Windows NT 程序的动态链接做准备。这些引入库为每一个你的应用程序可以调用的 Windows NT 函数包含了一个记录。这个记录定义了含有这个函数和与这个函数相对应的序号的 Windows NT 模块。比如,一个 Windows NT 应用程序可能调用了 Windows NT 的 PostMessage 函数。当你链接程序时,链接程序在该库中查找 PostMessage 函数。链接程序得到了该函数的序号后,就把这个信息插到程序的.exe 文件中。当程序运行时,Windows NT 就根据这个序号把你程序中调用的 PostMessage 函数的真实部分链进去。

Windows 3.x 和 Windows NT 之间有何不同

Windows 3.x 从技术上讲是一个基于 DOS 运行的一个非常优秀的应用程序。从表面上就可以看出来它不是一个非常成熟的操作系统。Windows 3.x 适合于各种 x86 机器,只需要 4MB 内存。

Windows NT 不是基于 MS-DOS 的。它之所以替代了 MS-DOS,是因为它是一个能在多平台和处理器上运行的全 32 位可移植操作系统。Windows NT 有桌上型或工作组平台并可以被用作服务器。同时也向下兼容 MS-DOS 和 Windows 应用程序,Windows NT 支持运行新

的完整的 32 位应用程序。

下面将讨论 Windows NT 操作系统的最为显著的新的组成部分。

可移植性

Microsoft 为现在和将来的程序员和最终用户提供了一个激动人心的美好前景。Microsoft 针对其新的操作系统提出的主要目标之一就是系统可移植性。他们成功啦！Windows NT 的应用程序可以在多种常用的系统上移植。Portability(可移植性)使得整个操作系统只需做很少的改动就可以移到另一台机器上。

这使得开发者和最终用户在不同系统上运行不会感到很大的区别。这样，用 Windows NT，你就可以让座位外出使用他们的便携机上安装的笔记本上的 Windows 界面，在办公室使用联网的微机，而在开发实验室使用小型计算机。

对于开发者，只需设计一个产品。对于最终用户，也只需学习一套命令。皆大欢喜。很划算，对于不同系统只须做一次最初投资。

可移植性是一种比较高级的性能。实际问题不是软件是否能移植，而是它移植起来有多困难。Windows NT 就擅长轻松地移植。

首先，编写它用的不是对硬件要求严格的汇编语言，而是可在所有目标结构上运行的可移性的高级语言 C。第二，Windows NT 尽量减少直接与硬件接触的代码量。第三，由于它不可能避免使用硬件相关代码，所以它把这些例程局部化。通过在整个操作系统中不广泛使用硬件相关算法，Windows NT 移植起来就比较容易。每个目标环境都只需替代它的体系结构中特殊的那部分代码段。

特殊地是，Windows NT 把与硬件相关的代码都放在动态链接库中，这部分称之为硬件抽像层(HAL)。HAL 用这层软件解释硬件动作，比如 I/O 中断和内存访问。HAL 层关心标准 Windows NT 操作。与和硬件相关的实现间的通讯，这就使得 Windows NT 的核心在从一个体系结构到另一个体系结构的变动中保持不变。

Win32 软件开发包(SDK)

Windows NT Win32 软件开发包(SDK)允许开发者建立基于 Windows 的 32 位应用程序。现有的 Windows 3.x 应用程序也可以很容易地移到 Win32 下。下面几个重要的部分构成了 Win32 SDK：

- 完全支持基于 Windows 的 32 位应用程序。
- 完整的系统管理和开发工具：
  对话框编辑器，图像编辑器，热点(Hotspot)器，字体编辑器，DDESpy，跟踪，放大，压缩，扩大，Source Profiler，资源编辑器，Help 编辑器，Win32－兼容 Microsoft 基本类库，WinDbg(基于 Win32 的 Debugger)，Microsoft 编辑器和连接器，Microsoft 宏汇编，和 PView(允许看处理情况、线程及内存的使用)。
- C 和 C++的编译程序
- 支持 Intel 80386/80486 和 RISC 处理器
- 为 MIPS 对称多处理系统提供支持
- 为 MS－DOS 和 16 位基于 Windows 的程序提供支持

· LAN Manager—— 兼容服务器和客户代码。

Windows NT 的 Win32 SDK 在两个版本中有效：一个 CD－Only（包括所有的 postscript 格式的文件）和在带有已打印文件的标准磁盘上。

多处理器

现在我们越来越感到每个系统只有一个微处理器已很不够用。对于高级用户来说，Windows NT 可以利用多处理器体系结构。每个加到系统中的处理器都会给系统运行速度带来可观的提高。

一个 Windows NT 应用程序可以把程序中的每个线程（子任务或过程）分配给它自己的处理器。Windows NT 甚至可以贡献出自己所有处理器给系统层任务使用。这种设计使得应用程序和操作系统都得到了多处理器的好处。

所有这些的实现都无需对应用程序作出任何改变。无论何时，Windows NT 都会充分利用对称多处理方式。因此，用户立刻就可以感受到这种明显的变化。

可扩展性

可扩展性指的是一个操作系统在不同平台上的运行能力。Windows NT 可运行的机器范围很广，从笔记本式到高级服务器，到基于 Intel 和 MIPS、Alpha 系统，再到对称多处理器的机器。

分布计算

Windows NT 的另一个设计目标就是实现信息共享。现在的工厂从拥有一台巨大的主机已发展变化到拥有大量体积小、价格低的微型计算机。

这种计算机硬件的质的变化，就产生了一个为这些独立系统间的通讯开发程序的需要。Windows NT 通过在操作系统中直接加入网络的功能解决了软件和硬件间的握手。

POSIX 原则

POSIX 是"一个基于 UNIX 的可移植操作系统"的字首组合词。它包含了一组为一个 UNIX 类型操作系统定义界面的国际标准。在 80 年代中后期，美国政府开始使用 POSIX 标准。

这个标准的宗旨是：使得很容易把他们的应用程序从一个 UNIX 类型的界面上移植到另一个上。Windows NT 提供给用户的一个可选择的 POSIX 应用程序执行环境。

政府—— 可证明的可靠性

一旦数据存储和检测从一个体积大的主机移到联网的微机时，MIS 管理人员就非常担心其安全性。自然，这不只是某个公司关心的问题，它也给美国政府带来了许多困扰。政府看到了保持来自别处的某用户资源的安全和防止一个用户在某个时间得到所有系统资源的需要。Windows NT 就把美国国防部的 Class C2 安全原则做为目标。

这个标准规定了作为一个审核员要知道什么，或必须知道什么，遵守什么原则等。它也规定了一组具有特殊审核能力的标准。一个 C2 原则的操作系统允许系统的所有者决定谁可以

访问它。一个系统的监督员必须记录谁在何时访问了数据。

美国政府标准共包括从D(最低级)到A(最高级)4个安全层。Windows NT的基本结构将允许它的安全层移到较高的层次。

Windows 开放系统体系结构(WOSA)

Microsoft一直致力于实现"信息随手可得"的美好前景。通过他们的Windows开发系统体系结构(WOSA)的使用,Microsoft已经为文件共享,打印机共享,数据库访问,电子邮件和系统配置和管理定义了一个开放的网络接口标准集。WOSA允许Windows NT的最终用户访问多厂家的信息服务。

系统崩溃

Windows NT是一个已被证明很稳定的和完整的成熟的平台。通过改进内存保护方法,它能防止重要的商业应用程序的任何数据的损坏。没有单个处理器,或操作系统可以损坏一个程序,这是因为它们所给的都是它们自己的独一无二的内存空间。

Windows NT通过把每个应用程序和直接的硬件访问分隔开,使得它也能在系统崩溃时保护应用程序。Windows NT检查每一个请求的有效性及正确的特权级别。

最后,Windows NT文件系统(NTFS)提供了许多文件修复能力,比如一个完全恢复系统能很快地恢复整个文件。文件系统还保留一个处理记录,以便系统关闭时能保证磁盘结束的完整性。

NT文件系统(NTFS)

Windows NT支持多种文件系统,包括FAT(文件分区表),HPFS(高级执行文件系统),和Microsoft的新NTFS(NT文件系统)。NT文件系统具有下面六个主要优点:

- 内部安全性,包括可执行文件
- 一个系统崩溃后,可快速、直接地对磁盘上的数据进行文件系统恢复。
- 不间断地访问巨大的存储设备(2^{64}或17兆gigabyte)
- Uicode(16位)文件名,允许文件从一台计算机到另一台的国际间转移,而不会产生错误的文件名和路径。
- POSIX操作系统原则
- 内置的网络功能

虚拟内存

Windows NT有能力访问高达4GB的RAM和海量存储器。用户和开发人员能够执行需要大量内存的应用程序。这个具有64位地址访问能力的32位操作系统基本上能够消除你的应用程序在结构上的限制。

Windows NT的一个主要的目标就是能够在其上运行MS-DOS、Windows,POSIX和OS/2应用程序。这是一项非常艰巨的任务,因为每个环境对待内存都有自己独特的方法。Microsoft的解决办法是:让每个NT的环境子系统以应用程序所希望的方式来管理内存。

Windows NT采用32位平面地址或线性地址空间的虚拟内存系统。一个应用程序的虚拟

地址空间是由一个进程和它的线程组成的地址空间集。在运行时刻，虚拟内存管理器转换或映射每个虚拟地址到实际物理地址。由于操作系统验证虚拟地址和物理地址之间的转换，这样就阻止了每个进程访问别的进程空间。

Windows NT 允许每个进程的虚拟地址空间高达 4GB。Windows NT 同时还允许额外的 2GB 的空间做为程序缓冲空间。

许多运行在 Windows NT 下的系统实际上并没有 4GB 的内存。然而，当虚拟内存管理器检测完物理内存后，它自动地传输(或页交换)某些物理内存中的内容到磁盘上。页交换数据释放了一些物理内存，这就允许 Windows NT 将需要的代码和数据移入内存。当虚拟地址映射到页面地址时，Windows NT 自动装载或重装载需要的数据。

从自我保护的角度来考虑，Windows NT 的操作系统驻留在虚拟内存的高端，而用户的代码和数据则分配在虚拟内存的低端。Windows NT 还提供了通过阻止应用程序访问系统内存的死机保护。

Windows NT 同时还管理一个非页面的内存池。这个不可交换的内存包含了 Windows NT 最重要的对象和一些重要的数据结构。所有分配给应用程序的内存都是可以页交换的。

Windows NT 是如此先进，以致可以适应各种内存。这是通过动态地平衡内存以及页交换内存和文件高速缓存来实现的。

1.3 未来——“信息随手可得”

未来的信息交换应该是：语言无关的，硬件和软件融合的，文件格式透明的。未来的知识将在你的指尖；那时将没有含糊的协议、转换算法和不可达到的领域。想一想信息随手可得的美好前景吧。未来的操作系统将是：

- 各种信息的大集成
- 易于存储，检索和浏览信息
- 可以从任何地方访问信息
- 自动进行日常的事务处理
- 广泛的通讯能力

这几乎令人难以至信的梦幻就是 Microsoft 软件工程师的目标。Windows NT 仅仅是朝着未来目标奔驰途中的一个驿站。

对象的进化

新一代操作系统的目标将包含创建、使用、管理和发现信息，而不管它是什么或在哪里。这样就导致了一个对应用程序和系统信息的一致的对象表示模型。

这个可扩充的易于使用的对象模型将给用户一个个性化的、可控制并且极端灵活的个人工作空间。对所有的视觉接口的控制都有一致的感官，而不管其数据的来源是 CD、闭路电视、录像机和激光视盘。

这种信息的对象模型表示把数据的表达方式考虑进来。它同时还要考虑到哪找到这个信息(内容)，而与实现(提供者)无关。

例如，一个数字化的电话信息无论是从电话线上接收的，还是从内部其它信息中心传过来

的，都具有某些共同的特点。然而，对象模型将能够区分它们间的不同特点而辨别它们的处理方式。

这种未来的操作系统将要改写应用程序设计的未来。设计的重点将完全转移到基于对象的哲学思路上去，在那里所有的交点将只有一个模型，而不论什么类型(用户、进程、文档或设备)。

从表面上看，这个操作系统的未来将提供一个个性化的工作空间。它允许用户去选择信息的组织方式，并且可以具有很强的视觉效果。

这种感观效果是非常有必要的，因为未来的用户将可以随时访问世界各地的图书馆、艺术中心，通过 modem 查看和下载 CD，参加虚拟现场会议，到 Harrods 购物等等。

具有树形结构和新感观的先进的浏览方式将极大地提高你的查询能力。新的查询语言将引入许多特点和基于内容的查找，逐步求精模型和新的范围控制。联机条目的访问将可使用本地和网络存储、目录服务和数据库服务。

Windows NT 进化的基础

Microsoft 软件工程师已经提交了一个设计精良的、模块化的操作系统。通过采用一种近似卡内基——梅隆大学的 Match 系统的设计，Windows NT 提供了一个具有基本的操作系统功能的底层。

这个底层支持受保护的子系统，各个子系统处理 API，环境和操作系统服务。这种功能层的概念允许硬件上的发展。

1.4 开始学习你都要做些什么

这本书的目的就是逐渐地帮助有经验的 C/C++程序员过渡到可以用 Microsoft Windows NT 应用程序接口编写应用程序。文中介绍了如何使用所有常用的 Windows NT 函数、消息和数据结构。这些内容对所有 Windows NT 应用程序的执行都很有用。

C/C++编程语言是开发 Windows NT 应用程序的最佳语言。Windows NT 的许多编程特点的设计都与 C/C++程序员一致。Windows NT 应用程序还可以用 Pascal 和汇编语言开发，但同用 C/C++开发的程序相比，这些语言不可避免地遇到了一些挑战。

软件要求

要想建立本书中的 Windows NT 应用程序，你首先要在你的计算机上安装 Windows NT，并运行它，而且你还需要两个 Windows NT 开发工具：

- Microsoft C/C++编译程序(32 位版本)
- Microsoft Windows NT 软件开发包(SDK)

系统要求

Microsoft Windows NT 操作系统将运行在带有 8MB RAM 和多于 100MB 的磁盘空间的 32 位 Intel CPUs(80386 以上)的配置上，它与 RISC 系统的配置很相似。

它支持所有类型的磁盘驱动器，包括 3½英寸，CD－ROM，和 WORM；常用视频显示标

准，包括 VGA、Super VGA、XGA、8514 和 TIGA；32 位网络接口卡；磁带后备系统；和几台种常见打印机。

为了运行状态良好且容易使用，本书推荐如下最低系统配置：

上系统	486 或运行速度在 50Mh 或更高的奔腾微处理器
系统内存	最少 16MB
硬盘大小	最少 320MB(大部分用于图形应用程序开发)
软盘类型	1.44MB 3½英寸
图形适配器	ATI Ultra Graphics Pro(2MB)或局域总线图形适配器
鼠标	Microsoft 鼠标
语音板	Microsoft Sound System Sound Blaster Pro
CD－ROM	NEC Intersect CD－ROM Reader

1.5 下面该做什么

在这章里，已经介绍了 Windows NT 的发展史和 Microsoft 操作系统的美好前景。第二章将解释开发 Windows NT 应用程序所需的概念和词汇。