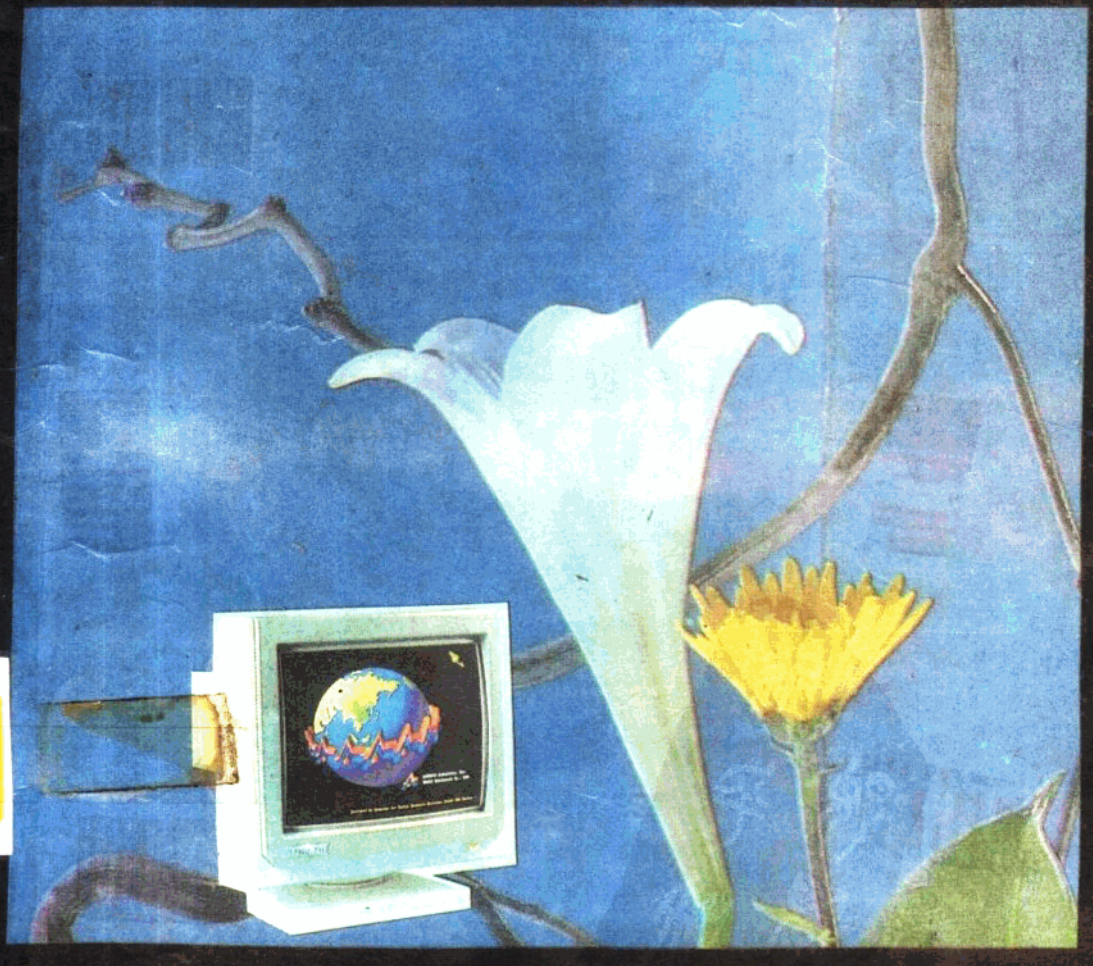
Borland C++ 环境下的 Windows 编程技术与实例

马 强 柯 源 吕永奇 编译

希望 审校



北京希望电脑公司 Borland C++ 系列丛书

海 洋 出 版 社

北京希望公司 Borland C++系列丛书

Borland C++环境下的 Windows 编程技术与实例

马 强
柯 源 编译
吕永奇

希望 审校

海洋出版社
1992. 北京

内 容 提 要

本书的目的在于帮助有经验的 C++程序员学习在 Borland C++环境下开发 Windows 应用程序，在讨论 Windows 编程技术的同时，还介绍了 Windows 的结构。

本书分为六部分。第一部分简要介绍 Windows 的发展历史；第二部分分析了一个最短的 Windows 程序，讨论 Windows 编程中的一些基本问题；第三部分讨论三种关键的用户接口对象：菜单、窗口和对话框；第四部分介绍了 Windows 的图形接口 GDI；第五部分关于用户输入，介绍了消息驱动的操作环境；第六部分介绍内存和动态链接。

本书是 Borland C++ 3.0 系列丛书之一，需要本书和整套丛书请与北京 8721 信箱资料部联系。邮政编码：100080，电话：2562329。

(京)新登字 087 号

责任编辑：刘莉蕾

Borland C++环境下的 Windows 编程技术与实例

马强　柯源　吕永奇　编译
希望　审校

海洋出版社出版发行(北京市复兴门外大街 1 号)
兰空印刷厂印刷
开本：787×1092　1/16　印张：36.69　字数：809 千字
1992 年 2 月第一版　1992 年 2 月第一次印刷
印数：1—3000 册
ISBN 7-5027-2654-3/TP・92　定价：25.00元

前　言

在推出 Peter Norton's Windows 3.0 Power Programming Techniques 几个月后，Borland 公司就为 Windows 推出了 Borland C++编译器．由于当今对 Windows 的开发都使用 C 语言，因而我们预言 C++不久将统治对 Windows 的开发．对于 C 的开发人员，C++提供了一种简单的移植方法，使 C 语言开发的系统增加面向对象的成分．基于这个原因，我们在本书中采用了 C++版本，选用 Borland C++编译器和 Borlabd Object Windows 库（OWL），因为它们能为建立应用程序提供一个好的基础．

不管你使用何种语言，Windows 编程的基础都是一样的．因此，这本书中的很多程序与 C 版本几乎相同．但也有例外:

.第二至五章中的程序用 C++重写．这反映了用 C 写的 Windows 程序和在 C++下使用 OWL 写的同样程序之间的区别．这几章中还讨论了两种基本 OWL 对象的组织：应用程序对象和窗口对象．

.所有的例子都是用 C++书写的．它们对于理解把 C++用于 Windows 编程提供了很好的帮助．

本书的目的在于帮助有经验的 C++程序员学习在 Windows 环境下编程．在学习一种新的环境时，有两种资源是很有价值的．一是利用环境特点来书写程序，另一个是使用 Help，它能提供很多环境内部和外部的信息．

本书给出了很多样本程序，用来说明对 Windows 不同特性的应用．你可以在你的 Windows 应用程序中利用这些程序．为了在 Windows 环境下编程，你需要下面一些工具:

.Borland 编程工具之一: Turbo C++ for Windows, Borland C++ for Windows 或者 Borland C++ and Application Frameworks.

.Windows 3.0 或更高版本.

在讨论 Windows 编程技术的同时，本书还介绍了 Windows 的结构.

这本书分为六个部分:

第一部分，简要介绍了 Windows 的发展历史，并提出了初次接触 Windows 编程的人员所面临的三个挑战：消息驱动编程，产生并控制图形输出，学习如何使用不同的用户接口对象.

第二部分，分析了一个最短的 Windows 程序，讨论了 Windows 编程中的一些基本问题．这一部分的核心是在于理解所有 Windows 程序所共有的基本结构．正如你会看到的，一个 OWL Windows 程序至少由两部分组成：应用程序对象和窗口对象.

第三部分，我们讨论三种关键的用户接口对象：菜单、窗口和对话框.

第四部分，向读者介绍了 Windows 的图形设备接口（GDI）．你的程序将会使用 GDI 在显示器、打印机、绘图仪上产生和设备无关的图形输出．第十章介绍了图形编程中的基本概念，诸如绘图坐标，GDI 的设备连接器，剪辑等等．其它章节讨论了用像素，实线，填充区域和文本产生的 GDI 输出.

第五部分是关于用户输入的．Windows 是一个消息驱动的操作环境，所以输入是以

消息的方式到达的．我们讨论了物理硬件传出的数据流，通过系统缓冲区，最终进入一个Windows程序．

第六部分覆盖了操作系统的其它特征．其中包含两个基本部分：内存和动态链接，Windows 3.0的一个新特点就是在内存使用方面进行了改进．为了帮助读者理解这到底意味着什么，我们介绍了Windows每一种操作模式下的内部工作情况．

总之，我们希望通过阅读这本书，读者能对Windows编程有一定的理解，并盼望阅读本书的人能给我们提出好的建议，为Windows乃至计算机软件事业的发展作出我们的贡献．

目　录

第一部分　Windows 入门

第二部分　最短的 Windows 程序

第三部分 用户接口对象

第四部分　图形设备接口介绍

第五部分　驱动消息的输入

第六部分　操作系统

附　录

第一部分

Windows 入门

第一章　Windows　概论

Microsoft Windows 是对 MS-DOS 操作系统的图形扩展，Windows 在几个方面扩展了 DOS 系统。DOS 在一个时刻只能支持一道程序的运行，而 Windows 同时能支持多道程序的运行。DOS 只支持一些简单的图形输出，而 Windows 可支持复杂的，高级的图形输出。DOS 要求每道程序都要提供自己的用户接口，也就是说，对于用户运行的每道 DOS 程序，用户要掌握不同的指令集合。不同的指令结构对应于不同的程序就像在不同的汽车中有不同的操纵装置一样。但是 Windows 有一个标准接口，使得用户经过短期训练后，就能使用 Windows 的任何程序。Windows 提供的用户接口对象集中有窗口，菜单和图标。一个普通的用户接口对象集意味“看和感受”，这样就使得 Windows 程序容易被掌握和使用。

Windows 提供了一个多任务的图形用户接口，或称为 GUI，通过它可帮助生成交互式程序，窗口环境下的程序之间有良好的交互性。窗口环境下的程序和以往大多数环境下的程序结构不同，但这种结构在很大一部分上与 Apple Macintosh 和 OS/2 Presentation Manager 环境下运行的程序结构相同，在这些环境下运行的程序是事件驱动(event driven)，也就是说，这种类型程序的结构和操作主要由用户生成的事件（如鼠标击键操作）来控制的。这些程序不像传统的应用软件而像操作系统软件和中断管理例程，传统上，应用程序是关联驱动(sequence driven)而非事件驱动，即程序规定了用户必须遵守的顺序，只有按此顺序，才能达到该程序运行的目标。相反地，对于事件驱动程序来说，为完成给定的任务，用户可以自己规定其执行步骤。从关联驱动到事件驱动的改变，要求我们用另一种方法来思考，而本书正是用来帮助我们学习这种新方法的。

1.1　Windows 的历史

所有 GUI 系统的起源要追溯到在 Xerox 进行的工作，在 1970 年，Xerox 建了 Palo Alto Research Center (PARC)。其主要是为了创造一种信息处理的新结构。在它的其它成就中，PARC 被认为发展了激光打印，局域网，图形用户接口和面向对象的编程技术。

在 PARC 的研究人员为一种取名为"Alto"的机器建造了不同的版本。经过许多年后，建成了几百台这种内部研究机器并被广泛地使用。通过对 Alto 的研究，Xerox 生成了一个商用 GUI 系统：Star 8010 工作站。Xerox 公司在 1981 年 4 月宣布了 Star 的成功，比 IBM PC 早了 4 个月。这个系统由一个鼠标和一个位图图形显示器组成，在上面可显示图标、窗口和一定比例空白的文本。虽然它的昂贵价格没有给它带来商业上的成功，但 Star 作为第一个商业可行的 GUI 系统，是一个重要的里程碑。

Steve Jobs,Apple Computer 公司的创建人之一，在 1979 年左右参观了 Xerox 公司的 PARC。他对 Alto 系统的多样性留下了深刻印象，于是，他回到公司以后，开始了发展一个同样系统的研究工作。Apple 公司 1983 年研制成功了它的第一个 GUI 系统：Apple Lisa。在这个的基

础上，又开发了第二个 GUI 系统 Apple Macintosh，这个系统是第一个成功的商用 GUI 系统。

Microsoft 公司在 1983 春季开始研究开发 Windows，在 8 年前，该公司的创建人 Bill Gates 和 Paul Allen 为世界上第一台成套部件的计算机－MITS Altair 成功地写了一个 Basic 解释器。2 年前，IBM 研制成功了个人计算机，并且正要推出 DOS 2.0 版本，以支持其新产品 IBM PC/XT 上的分级文件系统。那时，Microsoft 公司曾讨论过是否为 IBM 个人计算机建立一个 GUI 系统，但没实施任何可行计划。最主要的原因是当时的 PC 机只有两个软盘驱动器，64K 的 RAM 和一片 8088CPU。对一个 GUI 系统来说，它的建立需要更强大的硬件支持：硬盘可以快速存取，并有强大的内存来存储复杂的代码和图形数据。但是后来发生了一件事使 Microsoft 公司开始了对 GUI 系统的开发。

在 1983 年 2 月，VisiCorp 公司宣布研制成功适用于 IBM PC 的 GUI 系统 VisiOn，这件事促使了 Microsoft 公司开始着手研制自己的 GUI 系统。因为一旦 VisiOn 占领了市场，则必然导致软件开发者们损害 MS-DOS 的标准性，对于 Microsoft 公司来说，他们认为软件的标准性和一致性是微机产业成功的决定性因素。于是成立了一个开发＋小组－交互系统组（ISG），它的成员包括 Xerox PARC 的 Scott MacGregor，另外一名开发者是 Neil Konzen，他曾经为 Macintosh 开发过软件。当 Windows 的第一个版本在 1985 年 11 月研制成功时，可看出其风格受了 Xerox PARC 和 Apple Macintoin 的影响。但 Windows 是 Microsoft 的产品，因而相应为其编写一个操作系统 OS/2，Windows1.10 版在 1985 年 11 月投入市场，如图 1.1 所示，Windows 第一个版本的自动安排功能减少了用户的工作量，并且采用重叠式窗口。该版本的 Windows 被安装在带有双驱的 IBM PC 上，它有 256K RAM 和一片 8088 CPU，这也是 ISG 成员早期开发 Windows 时采用的配置。在后来，才增加了今天所采用的一些配置：硬盘和高速的局域网，因为虽然这个在某种程度上减慢了发展速度，但是 Microsoft 公司认为这些设备对于程序开发者将来在 Windows 下开发程序是必要的，在这种环境下对 Windows 的开发，也可帮助检验 Windows 在应用程序发展方面的适应性。

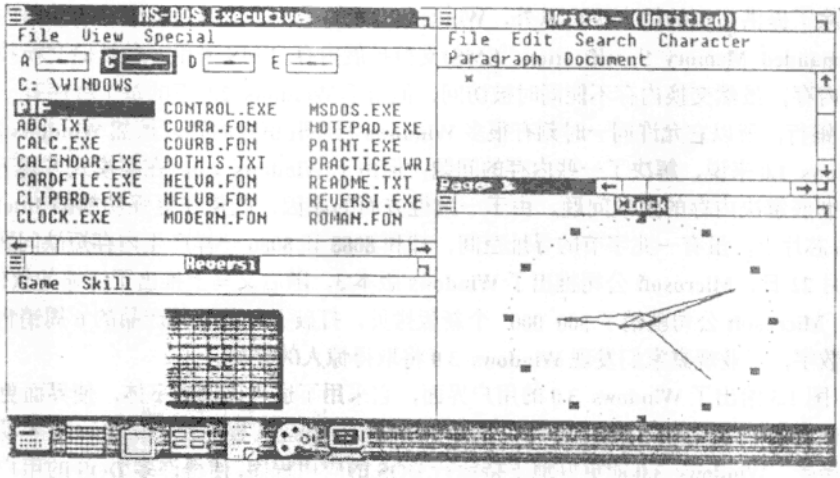


图 1.1 Windows 1.01 版本

Windows 的第二个主要版本是在 1987 年 9 月推出的 Windows 2.0。如图 1.2 所示，重叠式窗口是它的特点。Microsoft 从用户那接受到反馈信息，了解到用户并不欣赏粘贴窗口的好处，而感到粘贴是一种相当的障碍，这正是从自动粘贴改变的首要原因；其次是为了使 Windows 和另外一个那年 4 月推出的图形环境：OS/2 Presentation Manager 一致。Windows 和 OS/2 用户接口的相似性给用户从一个环境进入到另一个环境带来了方便。实际上，这两种环境下的用户接口是一个更大的 IBM 战略的一部分，这个战略称做 Systems Application Architecture。它的主要目标是使一个平台上编写的程序能容易地移到另一个环境下运行，例如，如果 SAA 得以实现，那么 OS/2 Presentation Manager 的程序可在 IBM 主机、微机和个人计算机上运行。SAA 中的用户接口寻址称作 Common User Access,简称 CUA。

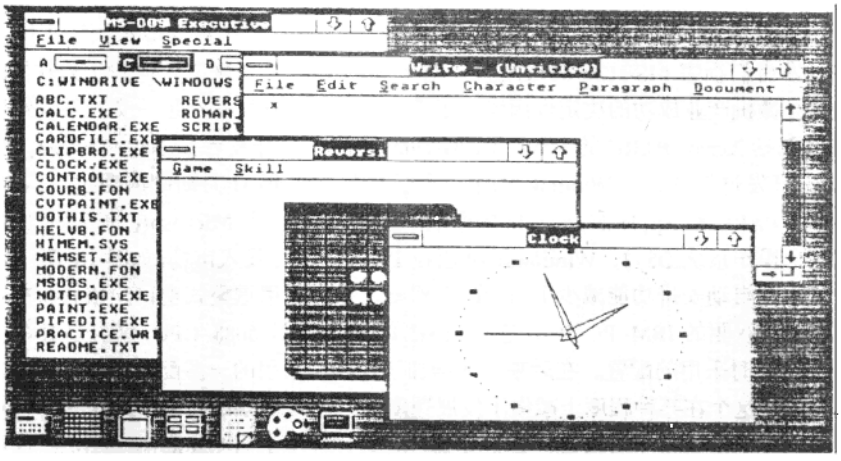


图 1.2 Windows 2.0 版本

除了提供新型的用户接口以外，Windows 2.0 的另一个成就是能更好的使用内存，它通过 Expanded Memory Specification(EMS)支持扩展内存。EMS 采用一种转储交换技术来访问附加内存，虽然交换内存不能同时被访问，但由于 Windows 2.0 下的每个程序有一个专门的 EMS 银行，所以它允许同一时刻有很多 Windows 程序驻留在内存。虽然 Windows 2.0 相对于 Windows 1.0 来说，解决了一些内存的问题，但由于 Windows 2.0。在实模式方式下运行，它并未彻底解决内存的短缺问题。由于一致性方面的原因，既使在更强功能的 Intel 80286 和 80386 芯片上，虽有一兆字节的寻址空间，却和 8088 和 8086 一样产生内存短缺问题。在 1990 年 5 月 22 日，Microsoft 公司推出了 Windows 版本 3，随后又马上推出了压缩的软件包。不到 6 周，Microsoft 公司就销了 500 000 个新版拷贝，打破了任何软件产品的 6 周销售记录。从这个数字，工业观察家们发现 Windows 3.0 将取得惊人的成功。

图 1.3 给出了 Windows 3.0 的用户界面，它采用了适当比例的字体，使界面更清洁，文本文件更容易阅读。三维投影，彩色图标和重新设计的应用程序使 Windows 更容易被广大用户所接受。Windows 3.0 能更好地支持运行 DOS 的应用程序，使得许多 DOS 的用户能很快地接受 Windows 环境。

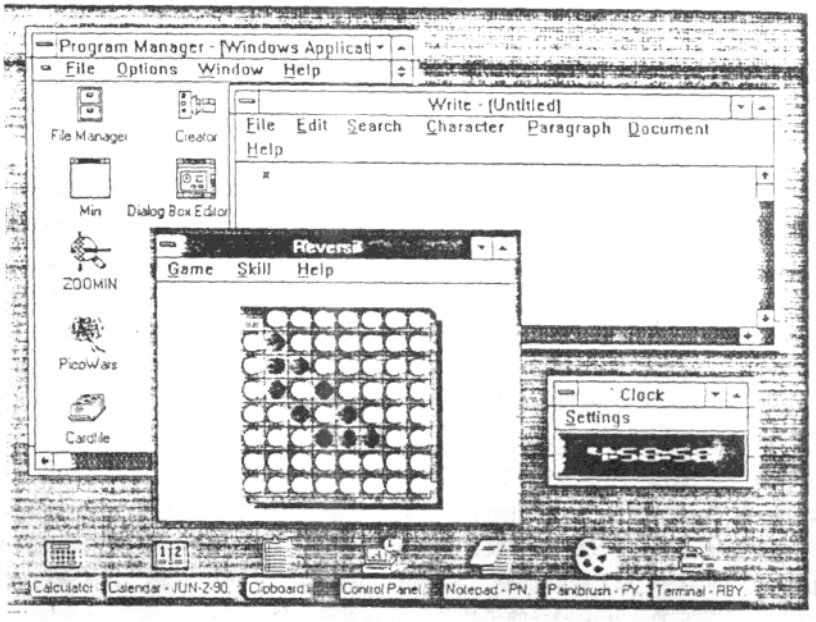


图 1.3 Windows 3.0 版本

从编程的角度来看，Microsoft 公司在用户接口中增加了更多的功能：支持用户自己设计的按键，菜单和列表框。比以往任何版本的 Windows 有了更大的适应性。在新版 Windows 下的菜单能被嵌套到程序员想要的任意深度，并且粘贴菜单使程序员可以自由地把菜单设置在他们高兴的地方。MS-DOS 的 Executive，对老版本的用户来说是很熟悉的，它被删去并被一组管理程序和文件的程序所代替:Program Manager,Task List,和 File Manager。

在内部，Windows 3.0 的一个最有意义的特点是支持扩展内存。在 Windows 3.0 环境下，WIndows 程序能访问 16MB 的 RAM。当 CPU 是 80836 或者更高档的时，Windows 通过它的内存管理功能来提供虚拟内存。在 386 增强模式下，虚拟内存是实际物理内存存储量的 4 倍。例如，如果有 16MB 物理内存，则 Windows 就可提供 64MB 的寻址空间。

比起早些时候的版本，Windows 3.0 能更好地支持网络功能，能方便地联到网络文件服务器和打印服务器上。它支持设备独立的位图格式，使得不同设备间分享彩色位图有一个标准，在支持超过 256 种颜色的设备上，应用程序甚至可以访问硬件调色板。这就意味着在一个多介质系统中可能采用的，支持最佳图形印象的功能，如今在 Windows 环境下也能实现。另一个新特点是，有一个高级的帮助设备，使你能为你编制程序的用户提供一个超级文本帮助。

当这本书即将出版之际，Microsoft 公司正在研制一个将在 1992 年推出的新型操作系统:Windows NT(New Technology)。在这个操作系统下，可支持多种类型的程序：Windows 3.x 程序，MS-DOS 程序和 Posix-compliant 程序 (Posix 是一种类似 Unix 的编程接口)。它还提供了一个 32 位的 Windows 版本，为 32 位 Windows API 所写的程序将有大量新的特征，包括访

同一个平面地址空间，信号，线和一个更强有力的 GDI。让我们来研究一下这个新操作系统的一些特色。

1.2 Windows NT

对于开发 GUI 系统的每个公司来说，都是要去建立新的和改进过的版本。Xerox 公司开始推出了 Alto，后来又推出了 Star。Apple 公司开始推出 Lisa 而后推出了 Macintosh。在推出 Windows 以后，Microsoft 公司又联合 IBM 公司推出了 OS/2 Presentation Manager。

推出 OS/2 的一个目的就在于为现有 Windows 的应用提供一条新道路。虽然 Presentation Manager 在结构上和 Windows 很相似，但它为用户提供的是完全不同的 API。每个函数的名字都不相同。例如，Presentation Manager 中的 WinCreateWindow 代替了 Windows 中的 CreateWindow。对于相似函数的参数，其顺序不是不同就是个数减少了。Presentation Manager 还提供了一个崭新的符号常量集合，如果 Windows 的开发者要把其开发的代码移植到一个新的操作系统下，那么移植到 OS/2 下和 Macintosh,X-windows 下所做的工作是相同的。

Windows NT 的推出证明了 Microsoft 公司承认它在 OS/2 上犯了一个错误。虽然 Windows API 有许多弊病，但大量的应用软件却把它们变成了我们的弊病。Windows32 位 API 由 Windows 3.0 的 16 位 API 发展而来，它能解决 Windows 的一些问题。新的 API 很容易被掌握，它保留了和旧的 API 中同样的函数名，符号常量和数据结构。

在新旧两种 API 中，原来 16 位的元素现在变成了 32 位，这看上去是一种根本性的改变，其实不然。每种 Windows 数据类型用一种简便的类型来定义，也就是说，当定义 short int 或 long int 时，可以用 HWND 和 HDC 来定义。当为这两种 API 编译程序时，只需为指定的 API 选择正确的包含文件即可，编译器就会判定 16 位和 32 位之间的区别。

所有这些特点对一个 Windows 编程人员意味着什么呢？当用户用 Borland C++编译器和 OWL 库来编制 Windows 程序时，他所编制的程序的寿命期将会很长。总的说来，既使 Windows NT 在一个无 Intel 处理器的环境中运行，用户为 Windows 3.x 编制的程序的二进制代码也适用于 Windows NT。现在，研究人员在基于 Intel 处理器的计算机和基于 Mips Risc 处理器的计算机上开发 Windows NT。

当用户对 Windows 有一定认识以后，就会对于 Windows 的编程有一定的想法，实际上，对 Windows 编程人员来说，面临的挑战是要理解 Windows 结构中的基础原理模型，一旦用户建立起 Windows 式的思考方法，那么掌握其编程技巧与掌握其它类型的编程技巧一样简单。如果用户已经有在 Apple Macintosh,OS/2 Presentation Manager 或不同的 X-windows 上的编程经验，那么就会很快熟悉在 Windows 下的编程了。

1.3 Windows 编程面临的挑战

假定用户是一个熟练的 C 编程人员，那么他所面临的三个基本问题是理解消息驱动编程，图形输出控制和掌握各种不同的用户接口对象，诸如窗口，菜单，对话框等。如果用户已了解了这几个问题中的一个或者更多，那么他就会更容易地掌握 Windows 编程。

下面我们依次来讨论用户所面临的三个挑战。

1.3.1 挑战 1：消息驱动编程

许多编程人员习惯于用关联的，过程驱动的方式来写代码。这样写成的程序，它的开始，中间和结束都被很好的定义了。例如，一个程序显示了一系列数字的屏幕输入来创建一些文本文件，它可能是一个飞机票或公司购物的顺序。在图 1.4 中的流程图描述了一个可编程序中操作的顺序。

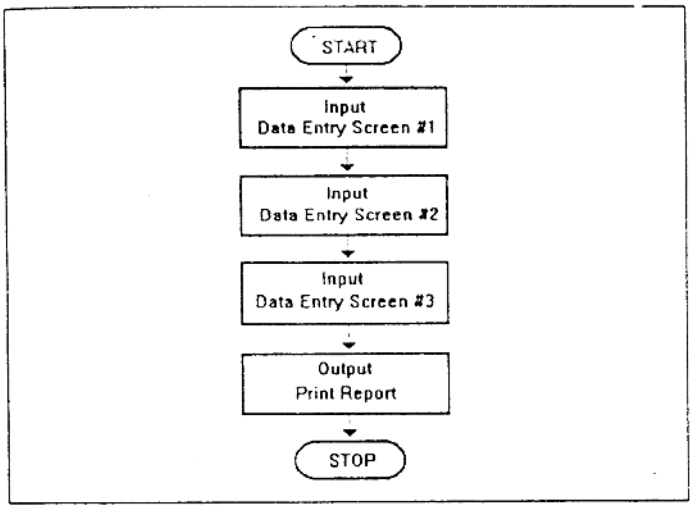


图 1.4　一个关联驱动程序

为了讨论的方便，我们说这个流程图代表了一个程序，表示代理商发售飞机票。第一次输入接受旅客信息：姓名，地址等等。第二次输入飞行信息和提供收费和日程计划信息。最后，第三次输入接受基于前面输入而制定的付账信息。每个数据输入屏幕必须正确填满，并且所有三次输入必须为发售一张票提供正确信息。

初看起来，这似乎是这样一个可执行的程序的合理方法。然而，计算机程序的工作不仅仅是售票，还包括确信正确的旅客信息已被接收和已支付当前的费用。这种方法有限制，即它是这个关联驱动方式的直接结果。

例如，因为程序执行了连续操作，一个代理人在第一次没有完整的旅客信息输入前，就不能得到第二次即提供收费和飞行信息。由于程序确信所需全部信息输入后，它不考虑现实世界中的意外情况。例如，如果代理处卖了一组机票，可能是给一个外出旅行的家庭的，代理处必须为每张卖出的票而要求输入信息。给过多次重复，程序在确信所有的必要信息已输入时，代理人在形式上却做了多余的工作。

一个事件驱动程序允许代理处以任何合理的顺序来输入数据，也许一个代理处将会选择与关联驱动程序指令的同样顺序。然而，代理却可以自由地连续地执行以适合不同旅客的要求。图 1.5 给出了一个大概思路，一个事件驱动的方法的怎样改变我们先前描述的传统的关