

COM+ 编程

A Practical Guide Using Visual C++ and ATL

Pradeep Tapadiya 著

冯延晖 刘晓铭 赵淦森 王泳 译

- * 提供能够最大限度地提高 COM/COM+ 软件的重用性和可伸缩性的实用方案
- * 为开发健壮的企业级 COM+ 应用程序提供指导和例子代码
- * 深入阐述 IDL 接口设计、COM+ 同步、事务、队列、安全、事件等
- * 详细介绍管理分步式组件环境的技术

PH
PTR



中国电力出版社

www.infopower.com.cn

COM+ 编程

A Practical Guide Using Visual C++ and ATL

Pradeep Tapadiya 著

冯延晖 刘晓铭 赵淦森 王泳 译

中国电力出版社

内 容 提 要

COM+是提供满足企业级需求服务的高级运行环境,它在软件开发这一领域得到了广泛应用。本书从一个开发员的角度来介绍了 COM+的基础机制,同时介绍了如何用 Visual C++和 ATL 创建 COM+应用程序。本书分为两个部分,12章,内容由浅及深,语言流畅,是读者掌握 COM+的最佳工具。

本书适合软件开发员和有一定基础的计算机爱好者阅读。

图书在版编目(CIP)数据

COM+编程/(美)塔波亚著;冯延晖等译. —北京:中国电力出版社,2002.8

ISBN 7-5083-1110-8

I. C... II. ①塔...②冯... III. 软件工具—程序设计
IV. TP311.56

中国版本图书馆 CIP 数据核字(2002)第 053060 号

著作权合同登记号 图字: 01-2002-0711 号

本书英文版原名: COM+ PROGRAMMING

Published by arrangement with Prentice Hall, Inc.

All rights reserved.

本书由美国培生集团授权出版

中国电力出版社出版、发行

(北京三里河路 6 号 100044 <http://www.infopower.com.cn>)

北京市地矿印刷厂印刷

各地新华书店经售

*

2002 年 11 月第一版 2002 年 11 月北京第一次印刷

787 毫米×1092 毫米 16 开本 26.5 印张 593 千字

定价 49.00 元

版 权 所 有 翻 印 必 究

(本书如有印装质量问题,我社发行部负责退换)

引 言

自从软件开发成为一门工程学科起，世界各地的软件开发小组就面临着同样的开发和运用问题。目前的开发者们关心的问题如下：

- 再用其他应用程序中使用并检测过的代码。
- 开发能根据用户需要定制的灵活应用程序，但不以开发组的过重负担为代价。
- 在应用程序发布后，解决异常，添加新特性，但避免完全重写应用程序。
- 利用第三方软件销售商开发的软件代码减少程序开发的时间。
- 以相似的方式开发分布式和非分布式的应用程序。

所有这些难题，甚至更多的难题都能用一种技术解决，即微软公司的 COM（组件对象模型）技术。COM 是开发软件组件的框架，是小的二进制可执行文件，用来为其他应用程序和组件提供服务。

Internet 大爆炸引起了信息发布方法的革命。在开发企业系统的过程中，传统的客户/服务器模型被一种用来增强 Internet 应用程序的三级程序设计模型所代替。开发这样的企业系统是一件耗时耗资源的事情，因为系统必须满足额外的企业级的要求，如可伸缩性、健壮性、安全性、事务支持等等。

COM+是一种高级的运行时期环境，它提供服务，以满足上面提到的企业级的要求。它是 Windows 2000 操作系统的主要部分。开发者可以利用 COM+提供的服务代替自己创建服务。

本书的重点是从开发者的角度理解 COM+的体系结构，并用 Microsoft Visual C++和 Active Template Library (ATL) 创建 COM+应用程序。

COM+并非从 COM 分离出来的，它只是 COM 程序设计模型演化过程中的下一个阶段。事实上，非常难分辨 COM 在哪里结束，COM+从哪里开始。要理解 COM+的基础机制和 COM+提供的服务，有必要理解 COM+核心的基本组件模型。因此，本书分为两个部分：第一部分建立基础，即 COM；第二部分解释 COM+ 1.0 提供的服务。

关于本书

编写本书的目的有两个，即帮助你：

- 理解 COM/COM+的基础机制。
- 研究 COM+ 1.0 提供的服务。

为了实现上述目标，我从开发者的视角阐述了素材，通过用 Microsoft Visual C++、Platform SDK 和 ATL 3.0 编写的简单应用程序说明关键的概念。本书提供了足够的 C++代码和工具，使读者的程序设计能力更强，能够进行进一步的研究。

在我们解释 COM+的基础机制和服务的过程中，首先提出一个或多个软件工程方面的

问题，然后讨论 COM+可以提供的解决方案。只要适用，我们就提供了代码段，以说明并解释概念。

为了集中讲述 COM+ 1.0 的体系结构和服务，我避免采用没有任何实际价值的素材。因此，我没有讨论 COM 的起源和历史，甚至没有讨论 Microsoft Transaction Server (MTS)，它是 COM+ 1.0 的前身。但只要有可能，我都提到了 COM+的未来版本 (COM+ 1.x) 中计划提供的服务。

在本书中，我对重点和有效的 COM+程序设计方法的技巧都做了标记。写字板和铅笔的图标标识重点：



灯泡的图标标记技巧：



语言的选择

COM+是一种规范，同时也是一种独立的语言，但目前大多数基于 COM 的组件都是用 C++编写的。对于所有服务器端的代码，本书都用 C++编写。客户端的代码则用 C++、VB 或其他任何适用于给定情况的语言编写。

必要条件

本书最重要的必要条件是你的学习主动性。

本书是为中到高级开发者编写的，它假定读者具备了以下知识：

- Windows 2000 操作系统。
- C++程序设计语言。
- VB 程序设计语言。
- C++模板。实际上，熟悉 C++标准模板库 (STL) 对学习本书很有帮助。
- 用 Win32 API 开发程序。
- Windows 2000 安全模型。

严格说来，这些假设中的许多都不是一定要的。例如，了解 Win32 API 或 C++模板有所帮助，但却不是必需的。在我们研究示例代码开发的过程中，我确信读者能够自动学习缺少的信息。

示例代码

本书中提供的所有示例都是精确完整的。为了简洁起见，书中有时只展示了相关的代

码段,但在本书附带的 CD 中可以找到完整的源代码。所有示例和工具都是在 Visual C++ 6.0 SP3 和 Platform SDK (2000 年 1 月)下编译的,而且用 Windows 2000 OS 的发布版本测试过。

参 考

本书常常引用其他书、Microsoft 公司的 Knowledge Based 文章、来自各种刊物的文章和 Microsoft's Developers Network (MSDN) Library。每章的结尾列出了本章的所有参考资料。每种参考资料都通过作者的姓和发布它的年份索引。例如,Don Box 所著的《Essential COM》(编者注:该书中文版《COM 本质论》已由本社出版),发布时间是 1998 年,被引用为[Box-98]。在本书中,每次引用一篇文章,我都采用参考中的关键字索引。

章节组织

本书分为两部分。第一部分“基础篇”,由三章构成。它解释了组件对象模型(COM)的基础,并展示了如何设计和开发基于 COM 的应用程序。每章都以前面一章的内容为基础。

第二部分“扩展篇”重点是 COM+ 的基础机制和 COM+ 1.0 提供的服务。这些章是彼此独立的。但第 5 章解释了 COM+ 的程序设计模型,应该在其他章节之前阅读这一章。

附带 CD

本书附带的 CD 包含以下内容:

- 本书开发的 C++ 工具类和工具的源代码。
- 根据各章安排的所有示例的完整代码。

关于该 CD 的更多信息可以在 CD 的 read-me 文件中找到。

(编者注:由于原书附带 CD 的文件容量总共只有 1.43M,为了减轻读者负担,故本书不再附带 CD,而是将 CD 文件打包放在本社站点供读者下载,网址: www.infopower.com.cn。)

致 谢

首先，要感谢 Karim Valimohamed，咨询工程师，他为本书的内容和代码提供了很大的帮助，还仔细检查了本书的手稿。

在过去十八个月（我编写本书的时间）帮助过我的人中，我格外感谢 Shaju Mathew 和 Deepak Gajare，我在 Hewlett-Packard 公司的同事，他们与 Karim 一起，为我表达自己的想法提供有价值的建议。还要感谢我在 Hewlett-Packard 公司的队友，尤其是 Doug Albright、Jeannine Klein、Paul Li 和 Anup Pant，他们仔细阅读了我的手稿，并提出了建设性的批评。

当然，如果没有我以前的导师 Doug McBride 和 Laurence Sweeny 的支持，我根本没有可能编写这本书。我要感谢这两位，感谢他们对我的信心和长期的鼓励。

我要感谢 Prentice Hall PTR 和 Hewlett-Packard Press 的编辑组，Jill Pisoni（执行编辑）、Anne Trowbridge（产品编辑）、Jim Markham（开发编辑）、Susan Wright（编辑指导，HP Press）和 Scott Suckling（MetroVoice Publishing Services）。

特别感谢 Richard Thayer，California State University 的软件工程教授，他帮助我找到了执行某些研究的学生。

最后，也是最重要的，我想感谢我的妻子 Vrushali 和我 18 个月大的儿子 Jay，他们在编写本书的整个过程中一直耐心地支持着我。

目 录

引 言

致 谢

第 1 部分 基础篇

第 1 章 组件模型	3
1.1 组件重用性	3
1.2 传统的软件重用性	4
1.3 参考硬件组件实例	5
1.4 仿真程序	6
1.5 链接程序以及操作系统装载程序的作用	8
1.6 动态链接库	9
1.7 修补一个故障	10
1.8 硬件组件模型	12
1.9 将接口从实现中分离出来	12
1.10 C++语言的一般特征	15
1.11 C++中的虚方法以及内存布局	16
1.12 一个抽象基类的接口	21
1.13 一个组件的动态选择	25
1.14 扩展接口功能	27
1.15 管理一个对象的生命周期	31
1.16 最优化	36
1.17 错误报告	37
1.18 COM 的本质	42
1.19 小结	44
1.20 参考文献	44
第 2 章 设计接口	45
2.1 为什么使用一种新的语言呢	45
2.2 IDL (接口定义语言)	46
2.3 IDL 设计目标	48
2.4 处理 IDL 文件	61
2.5 IDL 文件结构	64
2.6 IDL 语法	65

2.7 自动化.....	96
2.8 接口和列集.....	108
2.9 内存管理.....	109
2.10 组织多个 IDL 文件.....	111
2.11 将所有的接口放在一起.....	112
2.12 小结.....	115
2.13 参考文献.....	116
第 3 章 组件.....	117
3.1 从 C++到 COM—— 一个快速浏览.....	117
3.2 修订 C++代码.....	129
3.3 COM 对象的实现.....	149
3.4 “瞬间”接口.....	162
3.5 组件复用.....	162
3.6 错误报告.....	175
3.7 小结.....	182
3.8 参考文献.....	183

第 2 部分 扩展篇

第 4 章 COM+的本质.....	187
4.1 企业系统.....	187
4.2 三层体系结构.....	189
4.3 COM+, 简化了的开发.....	190
4.4 小结.....	195
4.5 参考文献.....	195
第 5 章 COM+编程模型.....	196
5.1 基于属性的编程.....	196
5.2 COM+目录.....	197
5.3 COM+环境.....	204
5.4 对象环境.....	208
5.5 调用环境.....	211
5.6 对象激发.....	211
5.7 小结.....	217
5.8 参考文献.....	218
第 6 章 并行性.....	219
6.1 多线程编程.....	219
6.2 套间.....	226

6.3 配置设定	229
6.4 套间和标准列集	233
6.5 跨环境的访问（存取）	236
6.6 开发线程安全的 COM 代码	240
6.7 小结	252
6.8 参考文献	252
第 7 章 安全性	253
7.1 COM+安全模型	253
7.2 说明性的安全性设置	261
7.3 有关服务器端安全性的编程	265
7.4 有关客户端安全性的编程	270
7.5 小结	283
7.6 参考文献	284
第 8 章 事务处理	285
8.1 股票交易	285
8.2 事务处理理论	292
8.3 COM+对事务处理的支持	293
8.4 基本结构	297
8.5 补偿资源管理器	304
8.6 小结	312
8.7 参考文献	313
第 9 章 消息队列	314
9.1 基于消息的队列	314
9.2 MSMQ 概述	316
9.3 MSMQ COM 组件	321
9.4 队列组件	332
9.5 小结	341
9.6 参考文献	342
第 10 章 事件	343
10.1 简介	343
10.2 紧密耦合事件（Tightly Coupled Events, TCE）	344
10.3 松散耦合事件（Loosely Coupled Events, LCE）	350
10.4 小结	366
10.5 参考文献	367
第 11 章 可伸缩性	368
11.1 简介	368

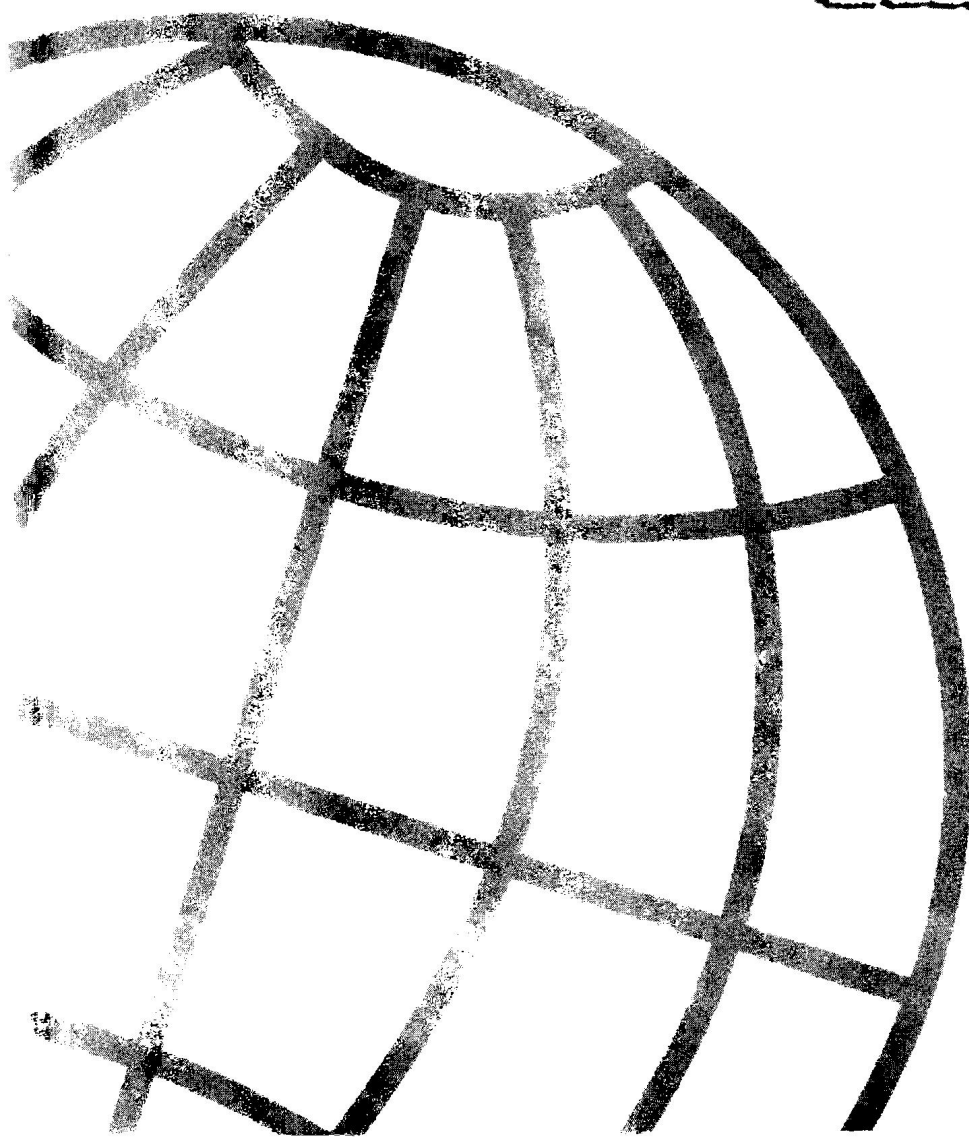
11.2 资源分配器	368
11.3 即时激活	370
11.4 对象池	374
11.5 异步方法调用	376
11.6 COM+管道	389
11.7 小结	391
11.8 参考文献	391
第 12 章 管理	392
12.1 概述	392
12.2 COM+管理对象模型	393
12.3 编程需要考虑的问题	399
12.4 一些普通的管理任务	401
12.5 未来会增强的特性	405
12.6 小结	406
12.7 参考文献	407

第一部分 基础篇

第1章 组件模型

第2章 设计接口

第3章 组件



第 1 章

组 件 模 型

计算机领域的持续性变革为用户带来了大量功能强大并且完善的应用程序。应用程序的完善性体现在流行的用户界面上，或者是对单一数据库的更新上，或者，更复杂一点的情形是：通过网络，甚至是在跨不同的硬件以及操作系统平台上，一个应用程序的不同部分能够进行通信，并且能够进行多数据库的更新。但是，这样的完善性也会给开发者们以及软件厂商带来许多的问题。开发、维护以及修改这样大型并且复杂的程序不仅困难，而且耗费大量时间。所以开发者们一直在寻找可以减少工程开发的工作量的方法，比如：重用现已存在的代码或者购买能够很容易并入他们的系统的第三方模块。他们进一步希望这样的方法能够允许他们在不中断代码原有用户的工作的条件下，扩展这种可重用的代码。

在过去的这些年中，有相当数量的范例（paradigm）以及编程方法提供给了开发者以及软件厂商，帮助他们开发可重用以及可扩展的代码。其中的一个范例就是面向对象的编程（object-oriented programming）方法学，这种方法学已经被软件界广泛接受。在这种编程思想下，特定的相关特性的集合被组合进一个单一的部件中，该部件被称为对象。这样做就创建了一个抽象层，该抽象层能够简化许多的编程任务并且为对象提供了可扩展性。

作为一门面向对象的编程语言，C++拥有大量的追随者。这门语言支持面向对象的编程原则，例如封装性（隐藏一个对象的实现细节）、继承性（在创建更新更特殊的对象时能够重用已存在的对象的能力）等等，这种支持已经导致了一个第三方 C++类库市场的出现。

尽管 C++是面向对象的，但是，在建立可重用的类库时仍然存在许多的障碍。这些障碍中的大部分是因为一个 C++编译器完成这门语言的一些特定功能的方式所造成的，这样就很难进行跨不同编译器的类库链接。本章中，我们会看到在开发作为可重用组件的 C++类库时存在的技术问题。并且，通过使用电子组件间的相互作用作为一个阐述示例，我将介绍一些编程技巧，这些技巧会阐释一些关键的技术性问题。最后，我将在遵循两个编程范例的基础上，向读者展示我们是能够开发可重用以及可扩展的软件组件的。这两个范例是动态链接和二进制代码的封装，这二者也是组成 COM（Component Object Model，组件对象模型）的基础。

1.1 组 件 重 用 性

星期五的下午快要结束了。今天也没有会议要开，于是你花了绝大部分的时间来运行

一个非常有创见性的算法，该算法将会被并入一个即将发布的产品中。一切都很顺利，这个工程也在按预定的计划被交付——这似乎太令人满意了但可能难以实现。这几个月来，你挣了薪水也忍受了难以忍受的艰难的开发工作，终于，这个周末可以好好地轻松一下了。当你靠着椅子，茫然地盯着屏幕时，你陷入了沉思：如何度过这个周末呢？这时你的项目经理走进你的办公室问道：“Dave，我们能够在我们的应用程序中，加入对网页的浏览功能吗？”你做了一个深呼吸，看着天花板，再次陷入了沉思。我们将需要做一些低级的网络界面接口的编程工作来连接到一个 Web 站点。我们将需要从语法上分析 HTML 文档，并且将用户组件显示出来，这些用户组件包括按钮、编辑框等等。似乎这还不够，我们还将需要处理在 HTML 文档中插入的脚本程序。那么，又怎样去处理防火墙以及代理服务器的问题呢？除此之外，要想跟上浏览技术的发展也不是一件容易的事。从 HTML 到 DHTML，再到 XML 以及 XSL，唉，技术的发展永无止境。于是，你回答道：“这将有許多工作要做，而且，又不是我们所擅长的。这么多问题要考虑，而我们产品的发行又是有最后期限的，我想，我们最好还是不要实现这个功能算了。”你的经理认为这不是一个令人满意的解决方法，并且，他对你的反应也很失望。你筋疲力尽，并且又觉得搅得你的经理心慌意乱让你有一种负罪感，但是，你又有什么选择呢？毕竟，你意识到了要实现这种功能的后果。那，到底有没有任何其他的选择呢？

你坐下来，靠着椅子开始思考。像网景公司（现在由 AOL 拥有的）以及微软等公司已经开发了浏览网页的技术，并且他们也一直在维护和更新这种技术。为什么我们不用一种对用户来说透明的方式来利用他们的工作呢？为什么我们不能在一起快乐地并存呢？

这也正是建立 COM 所考虑到的（除了最后一个问题外，因为这已经超出了 COM 所考虑的范围）。重用一個或者更多的软件组件来进行所有组件无缝集成的能力，以及当任何单个组件被更新时不会产生中断的能力，这些都是创建组件模型的出发点所在。

组件的重用性不是一个新概念。硬件对这个概念的支持已经很长一段时间了。例如：我最近搭建了一个家庭娱乐系统。我买了一台 Sony 的放大器/接收器、一台 GE 的电视、一台 Energy 的扬声器、一台 Pioneer 的 DVD 播放器以及一台 JVC DSS 接收器。我将这些组件的各种输入和输出挂接在一起，并且将这个系统打开。太棒了！我的电视正在播放通过 DSS 接收器接收的一个频道，扬声器正在使用 Dolby 环绕声音技术播放声音。通过遥控器，我能够方便地在卫星频道和 DVD 播放之间进行切换。

我的家庭娱乐系统就是在不管各种来自于不同制造商硬件组件的情况下，进行操作的。这些组件中的每一个组件实际上都不知道其他的组件在怎样工作，以及其他组件采用的内部电路是怎样的。这怎么可能呢？这简直是不可思议吧？

不管能不能思议，硬件就是这样工作的。我们是否也能够让软件这样工作呢？

1.2 传统的软件重用性

给一个应用程序添加网页浏览功能的老问题能够用下面的两种方法进行解决：

- 获得一个来自于第三方软件厂商的支持网页浏览的完整的源代码，编译和链接这个源代码，并且生成可执行的应用程序。但是，这种技术并不是很流行，因为，绝大多数的软件厂商都不希望将他们的源代码共享。

- 获得一个以静态链接库方式存在的预编译代码。这个厂商将提供两种信息，一个是 C++ 的头文件（例如 WebBrowser.H），该头文件包括对网页浏览类的定义，以及一个静态链接库（WebBrowser.lib）。将这个头文件包含进你的应用程序的源代码中，然后，编译这个源代码，最后使用这个静态链接库链接这个代码来生成这个应用程序的可执行文件。

在上面的任何一种情况下，软件厂商的程序设计逻辑都已经被吸收到了你的应用程序中。所以，在运行这个应用程序时，这些头文件以及这些库文件都不再是必需的了。

这种传统的软件重用性存在几个问题，如下所述：

- 让我们假想一下，在你已经将你的应用程序打好了包以后，这个软件厂商却发现他的实现中有一个缺陷。这个软件厂商是非常合作的，他将提供给你一个更新的库以及相应的 C++ 头文件。但是，你却没有办法让这个网页浏览实现自己进行现场替换（field replace）。这就迫使你用新的头文件来重新编译你的源代码，用新的库来重新链接你的代码，并且打包新的应用程序给用户。就好像买了一个 TV-VCR（电视机-盒式磁带录像机）的组合装置，每次在 VCR 不工作时，你就不得不带着这台 VCR 和 TV 一起到维修店。你不觉得痛苦吗？

- 假定这个网页浏览器的库在你的应用程序可执行文件的映像（image）中占有 50MB 的空间，而且，你的软件组是由四个应用程序组成的，每个应用程序又都与这个库链接在一起。那么，这种情况下，最终这样的一段代码占用了用户终端硬盘上一片 200MB 的大区域。更糟糕的是，当用户同时运行这四个应用程序时，这样的一段代码将被装进内存 4 次。

为了能够克服这些缺点，我们有必要仔细查看一下链接程序以及操作系统装载程序在这个过程之中所起的作用。但是，在我们进一步讨论这个问题之前，让我们先定义一下我们将在这章中使用的简单参考实例。

1.3 参考硬件组件实例

为了能够将硬件和软件联系起来，我们将经常对照两个硬件组件来进行说明——一台 TV 和一台 VCR。并且，我将只考虑在这两个组件间的视频信号的相互作用，以此来简化这个示例。

为了获取来自于 VCR 的视频信号，有必要将 VCR 的视频输出插头连接到 TV 的视频输入插口，如图 1.1 所示。

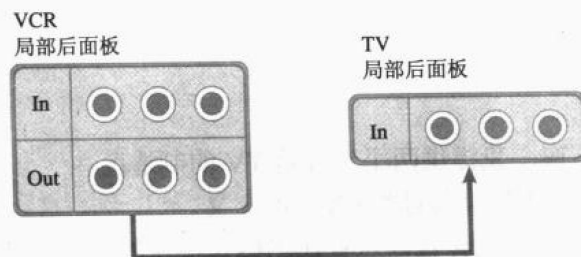


图 1.1 使用视频插头将一台 VCR 连接到一台 TV 上

使这两个硬件组件成为独立设备的优势是很明显的：如果在 VCR 中发生错误，就只需要现场替换 VCR 就可以了；当 VCR 升级时，你不用将 TV 也升级；在 TV 以及 VCR 都具

有标准的视频输入/输出插口时，任何品牌的 TV 和 VCR 都可以相匹配一起工作。

我将编写一段程序来模拟一下 TV-VCR 之间进行连接的情况。

1.4 仿 真 程 序

在我的仿真程序中，这台 VCR 的厂商已经签署了一个协议以提供一个 C++ 类 (CVcr) 给这个 TV 的制造商。这个类有一个被称为 `GetSignalValue` 的公有方法。当这个公有方法被 TV 端调用时，它将返回一个当前的视频信号值。这个信号的最大值设定为 40。

```
//vcr.h 文件
class CVcr
{
public:
    CVcr(void);

    long GetSignalValue();

private:
    long m_lCurValue;
};
```

在这个类的实现中，我让它在第一次调用这个方法时返回一个值 5。在接下来的调用中，使这个值依次加 10。如下所示：

```
//来自 vcr.cpp

CVcr::CVcr ()
{
    m_lCurValue = 5;
}

long CVcr::GetSignalValue()
{
    long lReturnValue = m_lCurValue;
    m_lCurValue += 10;
    return lReturnValue;
}
```

作为一个 VCR 的厂商，我提供两个文件给 TV 的制造商：

- 头文件 `vcr.h`，该文件包含有对类 `CVcr` 的定义。
- 库文件 `vcr.lib`，该文件包含有以对象代码格式存在的程序设计逻辑。

TV 仿真应用程序将在一个循环中获取并显示 10 次视频信号的值。其程序代码如下所示：

```
#include "VCR.h"
#include <iostream.h>
```