COM编程精彩实例

COM Programming by Example

【美】约翰·斯万科 著

徐颖 译

ng MFC,ActiveX,ATL,Ado and COM+

CMP books

中国电力出版社
www.infopower.com.cn

COM编程精彩实例

COM Programming by Example

【美】约翰·斯万科 著

徐颖 译

Using MFC,ActiveX,ATL,Ado and COM+

中国电力出版社

内 容 提 要

本书集中提供了一些能够体现 COM 最常用特性的例子，这些例子分别覆盖了使用 COM 的几个方面，即从进程内 DLL 到远程访问应用程序，从直接使用 COM API 到使用活动模板库（ATL）类来处理大部分工作。本书共分两部分，共 12 章。

本书对软件开发人员具有很高的参考价值，也适合大专院校学生阅读。

图书在版编目（CIP）数据

COM 编程精彩实例/（美）斯方科编著；徐颖译. -北京：中国电力出版社，2001

ISBN 7-5083-0608-2

Ⅰ.C… Ⅱ.①斯…②徐… Ⅲ.软件接口，COM-程序设计 Ⅳ.TP311.52

中国版本图书馆 CIP 数据核字（2001）第 23502 号

著作权合同登记号 图字：01-2000-2699

本书英文版原名：COM Programming by Example

Published by R&D Books, an imprint of Miller Freeman, Inc.

1601 West 23rd Street, Suite 200, Lawrence, KS 66046, USA

本书由美国 Miller Freeman, Inc.公司授权出版

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）

三河市实验小学印刷厂印刷

各地新华书店经售

*

2001 年 6 月第一版 2001 年 6 月北京第一次印刷

787 毫米×1092 毫米 16 开本 17.75 印张 394 千字

定价 **39.00** 元

致　谢

我直接或间接地感谢每一位曾经对本书做过贡献的人，他们是：为本书的版式而工作的 Berney Williams，做市场的 Paul Temme，以及使我听起来像 Hemingway（一个很滑稽的人）的 Michelle Dowdy 和 Kristi McAlister。我还要感谢 Mike Wallace，由于他的工作而使得本书看起来不像是小说，当然手稿不是出于 Morley's 之手。

学习一门科目并不是一件很简单的事情。我要感谢同我一起学习 COM 的人，他们是：我的兄弟 Paul、John Creaser 和 Bob Otterberg。

我还有幸认识了 Neu Vis 有限公司的一伙人，这是一家从事于应用 COM 进行创造性的工作的公司。在这里，我要感谢 Neu Vis 有限公司的 Rajiv Uppal、Jim Farrell、Devang Parikh、Jose Bescheider、RituGupta 和 Tony“EJB”Tao。如果由于本人的疏忽而没有提到谁，他可能会很沮丧。

前　言

欢迎进入 COM（构件对象模型）世界。如果说有一个以实例为最佳表现形式的专题的话，那么它就是 COM。没有哪个 COM 的 API、配置文件、类、向导和帮助程序可以单独运行——你需要一个把这一切都包括在一起的例子才能知道怎么使用这些东西。而当你有了这样一个例子并且明白自己在做什么和怎样去做的时候，在你的应用程序中加入 COM 就会不费吹灰之力了。

在这本书中，我试图提供一些能体现 COM 最常用特性的例子。我希望这个形式既能对 COM 的初学者有帮助，又能对熟练运用 COM 的专家有价值。

本书内容

本书的例子被划分成几章，它们覆盖了使用 COM 的几个不同方面——从进程内的 DLL 到远程访问应用程序，从直接使用 COM API 到使用活动模板库（ATL）类来处理大部分工作。内容上联系紧密的章组成为一个部分。

第一部分：COM 基本知识

虽然我尽量试图使这本书以例子为主，但我想还是应该先了解一点原理，当你的程序不能正常运行时它们可能会对你有所帮助。当然，你也可以跳过这一部分不看，只是在第二部分中我经常引用第一部分的内容，所以最好还是看一看。

第二部分：COM 实例

本书的第二部分由 COM 实例组成，从 COM 初期 ActiveX 控件的实例一直到现在 MTS 和 COM+实例。这些实例将着重介绍如何使用 API、MFC（微软基本类库）、ATL、VB 或 VJ++来创建和访问 COM 对象，其中又涉及到多任务、继承和回传等问题。

关于 CD

本书附有一张 CD，对书中的每个例子 CD 中都有一个相应的 VC++、VB 或 VJ++6.0 的工程。如果你想找某个例子的工程，只需在 CD 的子目录中找到这个例子的编号即可。

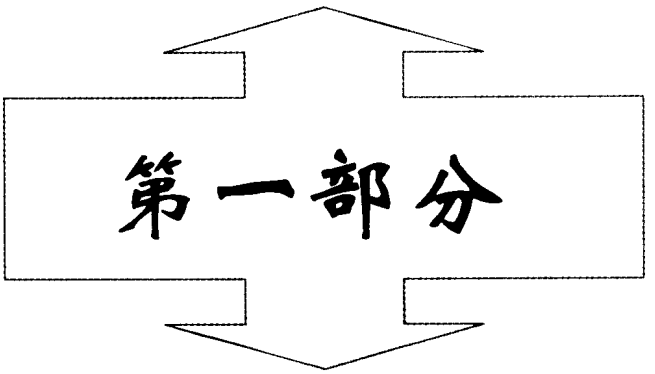
关于 SampleWizard

CD 还有一个功能即“样例向导（SampleWizard）”功能，它可以帮助你直接将书中的例子粘贴到你的应用程序中去。这个功能还可以引导你浏览书中的例子，如果你选中了其中一个，它就会详细列出在工程中添加该例所需的指令和代码。它还允许你把例子中的工程名称（如 Wzd）换成你自己的。

这个样例向导在 CD 的\SWD 目录下。它借用了各个例子下的\Wizard 目录，包含该例特殊的部分。你只需执行 SW.EXE，向导就会引导你完成剩下的步骤。

这个功能在 Developer Studio 中尤其有用。当然别忘了将这个目录设置成工程的当前路径[$(WkspDir)]，这样你需要的例子就可以直接拷贝到你的工程目录中去了。

目　录



COM 基础知识

使用 COM 其实只需要学会三件事情：创建一个 COM 对象、与 COM 对象通信以及处理当你做上述两件事时遇到的各种问题。所幸，这个“各种问题”一般都不会是什么大问题，可能是在多任务应用程序中如何使用 COM 对象、如何禁止其他用户访问你的 COM 对象等等诸如此类的问题。

这一部分的内容有以下几章：

第一章　COM 对象

本章介绍如何使用 COM API 来创建 COM 对象，以及为了使 COM API 能访问内部对象而应在 DLL 或 EXE 中做的一些准备工作。上述工作可以通过宏 COM、MFC 的 COM 类或 ATL 类来完成，其中宏 COM 直接调用 API，而后两者对 COM API 做了包装。在本章中还会介绍以 DLL 方式创建并用 MTS/COM+管理 COM 对象的优点。

第二章　COM 通信

我们在第一章中学习了如何创建 COM 对象。在本章中，我们将进一步学习如何与对象交换数据。与 COM 对象交换数据十分简单，与函数调用时的参数传递很相似。或者也可以通过网络端口自动进行数据的序列化和解构。

第三章　其他 COM 问题

在本章中，我们将学习在 COM 对象创建和通信的过程中一些常见的问题。其中包括如何继承已有 COM 对象的一些功能，在多线程环境中如何保证 COM 对象中数据的安全性以及在数据库受到攻击时 COM 对象的保密问题等。

第四章　COM+

本章将介绍 COM 最新的一些扩展。顾名思义，COM+不是整个 COM 的一个新版本，而是 COM 某一部分的扩展，叫做“微软事务服务器（MTS）”。它体现了 COM 的特点之一，即支持大规模的应用程序，潜在地鼓励用户用许多 PC 机来代替一个工作站——至少微软公司希望这样。

第一章 COM 对象

在本章中，我们将学习 COM 对象的基本知识：COM 对象是什么，如何创建 COM 对象并与之通信，如何在一个 COM 对象中加入新的功能。我们还将详细介绍有关 API 的知识以及在应用程序中使用 COM 的各种方法。

1.1 什么是 COM

简言之，COM 是一类系统应用程序接口（API），它允许你的应用程序访问其他应用程序（EXE）或动态链接库（DLL）中的数据和函数。你也许会说，不用 COM 我也能这样做，有很多机制，例如窗口消息、管道、套接字、动态数据交换（DDE）、远程过程调用（RPC）以及其他一些 API，都提供了这样的功能。

不错，COM 并没有提供什么新功能，但它提供了一种客户/服务器标准。使用其他的协议和功能，你只能处理你自己写的东西，但是如果你把你的应用程序放到一个 COM 里，其他遵守该标准的人也能访问它。

COM 是构件对象模型（Component Object Model）的简称，其基本思想是试图像搭建计算机硬件设备那样搭建软件。有人提出过这样的问题："为什么软件不能像硬件那样组装？当我在装计算机硬件的过程中需要一个总线控制芯片时，我可以去买一个，而不必从头开始设计制作，假如软件也能这样组装的话，那么当我的程序中需要一个拼写检查器时，只需买一个安装上就行了。"

这个类比也许芯片制造商们不会同意，因为芯片制造商最不希望自己的芯片和其他制造商的兼容，而软件制作公司则希望不同公司的软件能互相兼容。不过，在现实中你会发现，虽然 COM 使不同厂商的软件之间的接口变得更加简单，但这种连接仍需要有专业技术。

另外一个问题是你会发现流行一大堆和 COM 有关的术语。这些术语在很多情况下并不能帮助你理解它，而是使它更加神秘了。Marshalling、Aggragate、自由线程，还有 OLE、ActiveX、远程对象等等。恐怕没有多少人能弄清楚这些词表示什么意思。很多关于 COM 的讨论在开始时就列出了一些新的、怪异的词，然后这些词渐渐流行起来，最后变成了术语。就像一个外科医生走进一个手术室时得到一份清单一样，清单对手术室里的每件东西都重新指定了一个新的名称，然后这个外科医生用这些新的术语开始操作，我们所要应对的环境就像这样。

1.2 为什么要开发 COM

COM 的前身最初是为了解决一个特定的问题的。在字处理器中编辑一封信，信中要有一个电子表格，因此，逐渐就提出了这样一个问题：能不能不退出字处理器就完成电子表格的编辑？其实这个问题的解决方案就是对象的链接和嵌入（OLE）技术，它包含一些很复杂的 API，现在还可以通过调用这些 API 在 IE 里使用微软的 Word 编辑信件。OLE 的大部分功能都可以在 OLE32.DLL 里找到。

在 OLE 的第二版 OLE2 中，它的开发者们修改了 OLE32.DLL，使之更能体现 OLE 的基本思想，即允许一个应用程序访问另一个应用程序。这个 API 从 1993 年之后至今都没有大的改变，一般都把它认为是 COM API。

当开发这个 API 时，OLE 开发队伍所做的大部分设计都是在当时的开放系统标准之上建模的，引入这些标准的目的是使不同软件生产商开发的系统之间能够自由地交换数据和功能，至今 COM 中还残留着这些标准的影子。COM 毕竟是微软公司用来解决问题的方案，因此如果你在 UNIX 系统上没有安装 COM API，那么 COM 就不能工作。

1.3 COM 还有什么用

当 COM API 开发出来之后，微软用它来创建一些后来被称为 OLE 控件的东西，它允许用 Visual C++来写一些比较复杂的控件，比如螺旋按钮、能播放音乐的复选框等，以供 Visual Basic 的应用程序使用。这些技术后来又进一步发展成为可以使用控件从 Web 浏览器中下载，人们给它们起了一个新的名称叫做 ActiveX 控件，或者简单地叫做 ActiveX。尽管从理论上说你可以将任何功能封装在一个 ActiveX 控件中（这是在 VB 中创建一个 COM 对象的惟一方法），但一般说来实际上仅限于用户接口控件。

另一方面，微软公司鼓励所有的 C 和 C++程序员在写客户/服务器程序时直接把 COM API 作为基本框架使用，而不是自己重新写一套。因为 API 提供了一套整齐、标准以及在一定程度上透明的方法来与服务器交换数据。

另一个不太多的用途是为那些使用 DLL 的 C/C++应用程序服务的。如果程序中使用 LoadLibrary()选择性地链入 DLL，COM 对此提供了一种更强大的方法。另外，COM 也提供了一种机制，即在更新一个站点时只更新相应的 DLL，而不需要更新整个应用程序。

COM 支持多语言的共同开发（这里所说的语言是指 Visual Basic、C 等，而不是什么法语或葡萄牙语）。你可以把整个工程划分成几个 DLL 构件，在书写各个构件时可以随便选择一种语言。当然，构件本身可能对语言的选择会有一定的限制，比如你可能不会用冗长繁琐的 VB 来写一个处理数学计算的构件。

COM 的所有这些应用在今天仍然可以见到，但它们与 COM 在 IT 世界的应用相比就显得相形见绌了。在 IT 世界中，你几乎找不到一个完整的自包含的应用程序，程序都被分

为几个 COM 对象，在网络上任何一个地方运行着。我们将用单独一章“什么是 COM+”来专门介绍 COM 的这个应用。

1.4 COM 如何工作

如果你使用的语言是 VB 或 VJ++，那么你就不必去关心 COM 的工作原理，因为这些语言已经对 COM 的实现细节做了很好的屏蔽。但是如果使用 C++就没这么幸运了，C++比上面两种语言速度更快、功能更强大，因而也要求程序员对 COM 的实现机制有更深的了解。尽管近几年来情况已经有了一些好转，但是使用 C++的程序员仍会比使用其他语言的程序员更辛苦一些。如果你不准备用 C++，也不想了解 COM 是如何动态地把类实例化的，那么你可以跳过本章这部分内容。

下面是用 C++来创建一个 COM 对象的简单过程，使用其他语言也要经过以下过程，不过大部分的工作是在内部完成的，不需要程序员显式书写。

C++的未来

现在很多讲 COM 的书只有很少的篇幅讲到 C++。有人认为 C++是一种比较低级的语言。的确，VB 和 VJ++容易掌握，可以更快地编写出功能更强大的应用程序，但用这些语言写的程序的运行速度要比用 C++写的慢得多。我的一位经理人称用这些语言编程就像是“用蜡笔写程序”，因此这些语言不是未来大部分程序员的选择。如果你在 2112 年操纵一艘内部有几亿行代码的宇宙飞船，你能处理底层模块中的某个例外错误吗？实际上，现在人们正在使程序在性能上、功能上和易用这三者之间做权衡。正如 Stalin 所说：“数量就是质量。”我想未来的应用程序很可能会提供更多的功能，但会具有更少的选择。正如 Windows 资源管理器不能同时浏览两个目录，但可以详细地显示任何一种设备上的任何一级目录一样。

（1）调用 COM API 接口::CoCreateInstance()，可以直接调用或通过一个已经包装的 C++类调用。这个调用需要指定两个 ID，一个是要实例化的类的 ID，一个是对象所在的 DLL 或 EXE 文件的 ID。这里都使用惟一的 ID 标志，而不使用类名或者文件名，这样可以避免重名带来的混淆，有时候一个庞大的系统内可能有多个名字相同的 COM 对象。

（2）因为 DLL 或 EXE 文件是用 ID 给出的，所以系统要先将之转化为真实的文件名。这一步是通过检查系统注册表来完成的，因为每个 DLL 或 EXE 在装入时都会在系统注册表中有记录。

（3）获得了真实的文件名后，COM 调用 API 接口 LoadLibrary()装载需要的 DLL，或通过远程过程调用 API 启动相应的 EXE 文件。

（4）COM 根据::CoCreateInstance()中指定的另一个参数通知 DLL 或 EXE 要实例化哪

个类。在 DLL 或 EXE 创建了对象之后，::CoCreateInstance()返回该对象的指针。

（5）如果这个类在本地机器的 DLL 中，那么它的方法调用和参数传递与用类指针调用一般的 DLL 方法没什么区别，形如 p->Func（a,b）。

（6）如果对象在 EXE 中创建或在其他机器上，那么调用方法时参数要经过一个序列化过程，与客户/服务器模式相似，不同的是 COM 自动为你做了这些工作，这个过程对程序员来说几乎是透明的。

（7）消灭一个 COM 对象只需将该对象的引用计数减 1，来通知系统你不再需要这个对象。一旦对象的引用计数降至零，COM 就消灭这个对象。

（8）如果一个 DLL 为你的应用程序提供的所有对象都被释放了，COM 就会卸载这个 DLL；如果一个 EXE 为本地或网络上的任何应用程序提供的所有对象都被释放了，COM 就停止这个 EXE。

如图 1.1 所示的是 COM 的一个简化图示。

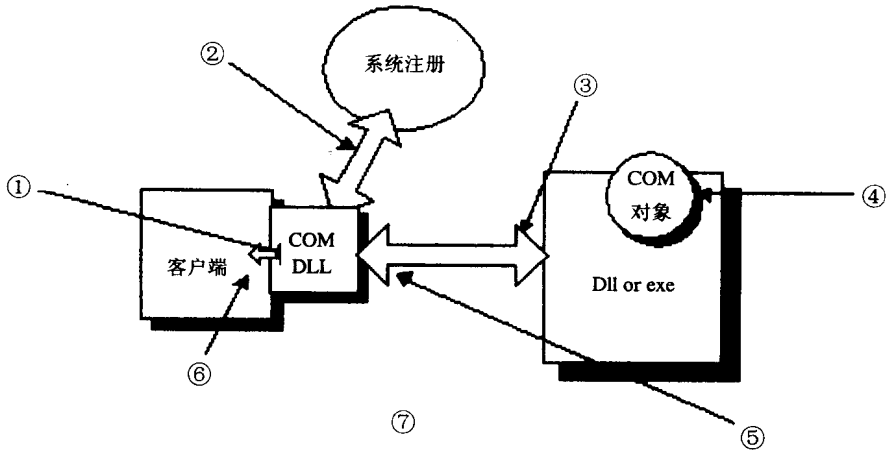


图 1.1 COM 概观

①客户端调用::CoCreateInstance()；②COM 将类 ID 映射到 DLL 或 EXE 文件名；③COM 调用 LoadLibrary()装载 DLL，或通过 RPC 启动 EXE；④COM 根据接口的类 ID 参数通知 DLL 或 EXE 创建相应的类的实例；⑤CoCreateInstance()返回对象的指针；⑥DLL 的方法调用不需要 COM 的介入，与一般的方法调用相同；EXE 的方法调用要由 COM 进行序列化，并使用 RPC；⑦客户端释放对象的指针，COM 通过卸载 DLL 或停止执行 EXE 真正消灭对象

1.5 怎样用 C++创建一个 COM 对象

用 C++创建一个 COM 对象其实要比在屏幕上创建一个窗口更容易。你不必给出一个冗长的参数表来指定窗口的位置、大小等，只告诉系统你要的对象在什么文件里、要其中的哪个对象就可以了。API 调用过程是这样的：

```
STDAPI CoCreateInstance(
REFCLSID clsid,              //Class id (what DLL or EXE file)
LPUNKNOWN pUnknown,          //Used for inheriting a COM object
DWORD dwContext,             //DLL, EXE local or remote
REFIID iid,                  //Interface id (what class in DLL or EXE)
LPVOID *ppv                  //returned pointer to created object
);
```

其中需要给出的不是类或 DLL/EXE 文件的真实名称，而是使用它们的惟一标识 GUID（全局统一 ID），从理论上来说整个宇宙都不会有两个完全相同的 GUID（关于 GUID，请参照注释）。使用 GUID 代替真实名称，是为了防止系统中有两个相同名称的对象而造成混淆。

1. GUID

开放软件组织 OSF 提出了统一惟一标识 UUID，GUID 是微软公司对 UUID 的解释。一个 guid 是一个 128 位长的数，从理论上说在整个空间和时间维上它是惟一的。为了保证它在空间维上惟一，guid 中有一些特定的位用来记录它是由哪个机器产生的，它通常是该机器的网卡地址，如果这台机器上没有网卡，会使用另一个常数。

为了保证在时间维上惟一，guid 包含了一个时间戳，用来记录它生成的时间。这个时间戳的范围是从 1490 年开始到生成该 guid 时的时间为止，以分钟为单位。另外，guid 还包含一个随机产生数。

你可以使用 MFC 提供的 GUIDGEN.EXE 来生成一个 guid。GUIDGEN.EXE 的工程和源代码也都是可以获得的，其核心是通过调用 COM API 的::CoCreateGuid()接口来生成 guid。

GUID 的格式为：

```
rrrrrrrr-tttt-tttt-oooo-aa-aa-aa-aa-aa-aa
```

其中：

rrrrrrrr：32 位长的随机数；

tttt-tttt：时间戳，其中低 16 位字在前；

oooo：与机器重启次数有关；

aa-aa-aa-aa-aa-aa：一个 6 字节长的字符串，一般是机器的网卡地址。

在不同地方 guid 可能以不同的格式出现。例如，在系统注册表中就采用这样的格式：

```
{4323CD20-2559-11d2-9BD8-00AA003D8695}
```

而在 GUID 结构中，就采用这样的格式：

```
{ 0x4323cd20, 0x2559, 0x11d2, { 0x9b, 0xd8, 0x0, 0xaa, 0x0, 0x3d, 0x86,
0x95 } };
```

采用这种格式，看起来像下面这样：