



Designed for



Microsoft®
Windows NT®
Windows 98

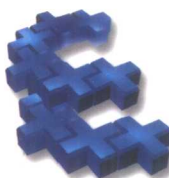


附 赠
CD-ROM

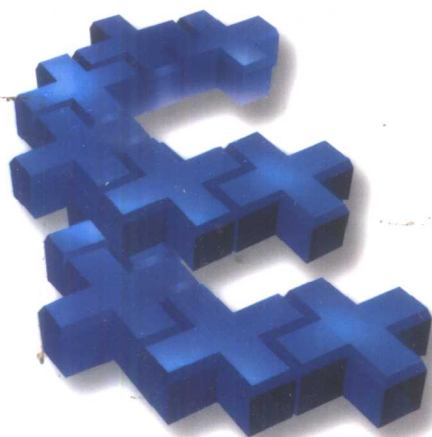
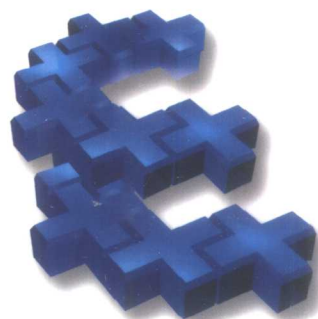
Programming Distributed
Applications with COM+ and
Microsoft Visual Basic 6.0,
Second Edition

微 软 公 司
核 心 技 术 书 库

(美) Ted Pattison 著 王新昌 陈越 费晓飞 等译



COM+ 与 Visual Basic 6



分布式应用程序设计

(第2版)



机 械 工 业 出 版 社
China Machine Press

Microsoft Press

微软公司核心技术书库

COM+与Visual Basic 6 分布式应用程序设计

(第2版)

(美) Ted Pattison 著

王新昌 陈越 费晓飞 等译

张红旗 审校



机械工业出版社
China Machine Press

本书介绍了COM和COM+的基本原理；揭示了大量COM+的技术内幕；分析讨论了COM+和Windows 2000所提供的大量服务。主要内容包括使用 Visual Basic 创建并发布组件、COM+运行时环境、本地事务与分布式事务处理、IIS和ASP最重要的新内容、MSMQ、组件队列服务、COM+事件、COM+的安全性、可伸缩应用设计等。

本书适合于希望在COM+和Windows 2000环境下进行分布式应用开发的Microsoft Visual Basic 中、高级程序员阅读，也可以用做ASP 和C++ 开发人员的参考书。

Ted Pattison : Programming Distributed Applications with COM+ and Microsoft Visual Basic 6.0, Second Edition.

Copyright 2001 © by Microsoft Corporation.

Original English language edition copyright © 2000 by Ted Pattison.

Published by arrangement with the original publisher, Microsoft Press, a division of Microsoft Corporation, Redmond, Washington, U.S.A. All rights reserved.

本书中文简体字版由美国微软出版社授权机械工业出版社出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

本书版权登记号：图字：01-2000-3119

图书在版编目（CIP）数据

COM+与Visual Basic 6分布式应用程序设计 /（美）帕蒂森（Pattison, T.）著；王新昌等译. —2版. —北京：机械工业出版社，2001.6

（微软公司核心技术书库）

书名原文：Programming Distributed Applications with COM+ and Microsoft Visual Basic 6.0, Second Edition

ISBN 7-111-08764-X

I. C… II. ①帕… ②王… III. ①软件接口，COM+—程序设计 ②BASIC语言—程序设计 IV. TP312

中国版本图书馆CIP数据核字（2001）第07667号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）

责任编辑：高智勇

北京市密云县印刷厂印刷·新华书店北京发行所发行

2001年6月第1版第1次印刷

787mm×1092mm 1/16·21.25印张

印数：0 001-5 000册

定价：55.00元（附光盘）

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

译者序

在当今的信息时代，以Microsoft Windows作为基础平台，开发分布式多层应用已成为愈来愈多人的需求。任何以Microsoft Windows作为基础平台构建多层应用的开发人员，都必须依赖于许多独立的软件组件。组件对象模型（Component Object Model, COM）是将所有这些软件组件绑定在一起的“粘合剂”。COM+——COM的扩展版本和其他一些关键的Windows服务则为开发人员创建大型多层应用提供了构件。曾几何时，Visual Basic的衰落似乎已是世所公认。但是，COM+正在终结所谓Visual Basic衰落的谎言，并且正在使它焕发出新的活力。利用COM+，Visual Basic将在多层开发领域大显身手。

本书是为希望在COM+和Windows 2000环境下进行分布式应用开发的Microsoft Visual Basic中、高级程序员而编写的，它不但教会你如何去做，还揭示了大量COM+的技术内幕。

本书介绍了COM和COM+的基本原理，讲述了基于接口、基于属性编程的概念和原理，探讨了如何使用Visual Basic创建并发布组件，分析了COM+运行时环境的体系结构以及如何利用COM+编写配置型组件，讨论了本地事务与分布式事务处理，涵盖了IIS和ASP最重要的新内容，介绍了MSMQ、组件队列服务、COM+事件，还涉及了COM+的安全性，最后讨论了影响可伸缩性和性能的应用设计问题。

毋庸置疑，COM+将使Visual Basic程序员再次成为时代的宠儿。我们为能够将本书介绍给致力于使用Visual Basic、以Microsoft Windows作为基础平台构建分布式多层应用的开发人员而感到由衷的欣慰。

参加全书翻译工作的有王新昌、陈越、费晓飞、孙凌、卢贤玲、陈性元、王希祥、吴瑞、叶慧敏、丁艳、王桂森、曹守见、杜学绘、王鲁、刘育楠、王国营、黄思国、杨艳、郭渊博、贺小川、麻信洛、季晓中、徐文军、张景生、牟书贞等，由王新昌负责统稿，由张红旗审校。

由于译者水平有限，时间仓促，错误和不妥之处在所难免，望广大读者批评指正。

2000年11月

序 言

从不同的角度讲，我们所有人都是VB程序员。

VB程序员被（正确地）描绘为：对许多人认为是非常关键的一组问题一无所知。如果这就是人们对一名VB程序员的认识的话，那就太令我——一名VB程序员黯然神伤了。我非常关心组件集成，但却对SQL Server 7.0中的T-SQL最优化全无了解。我的同事刚刚写过一篇长达400页的关于NT 安全的论文；然而，他却不能区分一个未解析的公共实体和一个内部参数实体之间的区别。在当今这个信息过载的世界上，作为一个软件开发人员，应勇于承认自己某方面的无知。

本书的第1版证实了Ted将读者的注意力集中到那些重要概念上的控制能力。与同类书籍常常是800页的庞然大物不同，Ted的书是一份易于消化的信息快餐，它可以使聪明能干的开发人员更有能力解决手头的问题。本书的第2版带来了Ted的最新故事，以及COM开发实践的现状。并且，它也将是又一本有关组件软件的力作。

Don Box
Redondo Beach, California

前 言

任何希望利用Microsoft Windows作为基础平台构建多层应用的开发者，都必须依赖于许多独立的软件组件。这些组件一些由Microsoft公司提供，一些由你或你的公司编写，而另一些则可能来自于第三方软件销售商。

组件对象模型（Component Object Model，COM）是一种重要的将所有这些软件组件绑定在一起的“粘合剂”。它可使程序开发者和公司能够有效地分发和重用他们的代码。COM+是COM的扩展版本，为运行于中间层的组件提供一种有效的运行时环境（runtime environment）。COM+和其他一些关键的Windows服务，如Microsoft Internet Information Server（IIS）、Active Server Pages（ASP）以及Microsoft Message Queuing（MSMQ），为程序开发者和公司创建大型多层应用提供了构件。

本书的读者对象

我确信，精通COM和COM+是使用Windows 2000建立多层应用的先决条件。我认为，如果对支持这些应用的基础结构没有深刻的理解，要想创建具有伸缩性的应用是不可能的。因此，本书旨在向你讲述COM+和Windows 2000中的关键部件，它们将影响你在开发分布式应用时设计和编写组件的方式。

本书是为希望在COM+和Windows 2000环境下进行应用开发的Microsoft Visual Basic 中、高级程序员而编写的。其讨论的焦点将集中在Windows平台的体系结构上。从这一点来看，它也可以用做ASP和C++开发人员的参考书。有些读者也许对COM较为陌生，而有些读者也许对COM和Microsoft Transaction Server（MTS）有一些经验。本书这一版的目标是满足上述两类读者的需要。

在过去的五年中，我经过长期而艰辛的思考精心制作了本书，使它包含最为恰当的细节。有些技术细节是十分关键的，而有些技术细节则过于深奥，了解它们将会浪费你的智力“带宽”。因此，我努力使本书涵盖重要的、而略去次要的技术细节；在保证文字尽可能简明扼要的同时，使本书覆盖更多的领域，并尽量避免不必要的复杂性，但必要时也会介绍一些低层的细节。

许多Visual Basic程序员不愿意更深入地学习COM和COM+，对于他们来说，无须任何深层技术知识，就可以在某种程度上使用Visual Basic实现平台上的功能模块。但是，他们肯定无法使用COM+、IIS、ASP和MSMQ创建大型的信息系统。

写给已熟悉本书第1版的读者

某些技术书籍的第2版与第1版相比，仅仅是更换了一个新的封面，而内容依旧，我对这种做法十分憎恶。本书第2版的一个主要目标是，为已读过第1版的读者提供增值的服务。该书第2版超过75%的文字是重新编写的。

我对传统的COM基础部分进行了重新组织，并尽可能将这些内容安排在本书的前面予以介绍。第2章讲述基于接口的编程（interface-based programming），与本书第1版的相关内容基本相同。我把分散于第1版若干章节中的有关COM基础的内容，浓缩到本版的第3章中。第4章和第5章描述了如何使用 Visual Basic 创建组件和怎样对组件进行版本控制，其中的内容比第1版有所增加，包含了用IDL建立定制类型库（Custom Type Library）和为脚本客户设计组件的新内容。对于熟悉COM和 MTS的程序员来说，第2章到第5章可作为一个快速回顾的参考材料。

第6章和第7章介绍 COM+运行时环境的体系结构。其目标是教你如何利用COM+所提供的运行时服务和线程池机制（thread pooling）编写配置型组件。第8章是关于事务处理的，它将介绍本地事务与分布式事务等方面的新资料。第9章涵盖了IIS 和ASP最重要的新内容。第10章是关于消息处理（messaging）的，与第1版类似，其内容是 MSMQ，但它增加了关于组件队列服务和COM+事件的新材料。第11章是关于安全性的，它是最近才写成的新增章节。第12章讨论了影响可伸缩性和性能的应用设计问题。

本书读者需要具有什么经验

假定你具有面向对象编程和使用类创建应用的背景。你是否了解使用 Visual Basic、C++或Java 的类倒无关紧要，重要的是应理解为什么要利用封装来设计一个类，以及明白类和对象之间的关系。

如果你具有计算机科学或用低级语言（如C语言）编程的经验，那将是十分有益的，但这不是必需的。在讲解COM时，不提及指针、存储地址、栈结构和线程这些东西是不可能的，所以需要预先学习这方面的背景知识。有时，这会花费较多时间，但你投入的时间和努力会得到更高的回报。

本书的大多数读者应做过数据库编程方面的工作，一般中级Visual Basic程序员应做过面向数据库的应用开发工作。当我讲述为 COM+ 对象编写事务时，我假定你对ActiveX Data Object（ADO）和结构化查询语言（SQL）已有一定的熟悉程度。如果你不具有这方面的背景知识，应当自己掌握它。本书出现的ADO和SQL代码并不复杂。

在关于IIS和ASP这一章，我假定你已经知道使用HTML的基本方法，能够建立简单的ASP页。鉴于大多数读者的背景，以及有许多高质量的ASP参考资料可供参考，我认为没有太多的必要讨论那些细节。

本书不包括哪些内容

本书没有包含太多的一步一步的操作指南。因此，本书对于那些只想知道做什么而不关心为什么这样做的人没有多大吸引力。建议参考微软开发者网络（MSDN）来得到关于诸如使用COM+ 或IIS管理工具的操作细节，我认为在这些方面大多数程序员都不需要帮助。我的目标是让你理解软件背后的原理。

我没有详细讲述如何通过编写脚本或定制管理应用自动进行COM+或IIS的管理。尽管COM+和 IIS 都为自动管理繁杂事务提供了丰富的对象模型，但我不在这一点上花费过多的时间。

这些话题仅在书中某些地方会涉及一点，建议通过其他途径学习有关细节。

如果你正在寻找一本包含大型样例应用的书籍，并以它作为你的学习起点的话，本书并不适合。本书所涉及的代码清单大都很短，在5至10行之间。书中的一段代码，总是尽量压缩在几行之内，以使你的注意力集中到特定的要点上。我忽略了一些无关紧要的代码，如错误处理。因此，本书不适合那些盲目地将样例拷贝并粘贴到应用代码中的人们，我的目标是教你“钓鱼”，而不是简单地给你鱼。

光盘中的内容

本书所附的光盘中包含几个Visual Basic 应用的源代码。这些应用包含了我在本书所使用的样例的代码。作为一名Visual Basic程序员，我始终感到，通过敲F5键，一行一行地单步执行整个应用，对问题的理解会有质的飞跃，我希望你也有同样的机会。Samples.htm文件将给你一个有关所有应用以及它们做什么的大纲。

光盘上的Setup目录包含了对一个Microsoft SQL Server 数据库进行安装的文件和向导，该数据库是运行Market应用时所需要的。Samples.htm文件可引导你完成这些步骤。每一个其他应用都有附带的安装向导，在其子目录下都有一个Samples.htm文件。我希望你能发现这些样例应用的价值。

运行样例应用所需的软件

如果希望运行包含于光盘中的所有应用，就必须安装Windows 2000 Server或者Windows 2000 Advanced Server。这两个版本都将COM+和IIS作为缺省安装。如果想运行第10章中的消息样例，就必须安装MSMQ，它不是Windows 2000的缺省安装部分。

你还应安装Microsoft Visual Studio 6（包括Visual Basic 6）、Microsoft Visual C++和MSDN的最新版本。在安装Visual Studio时，请确保选择Register Environment Variables 选项，其原因在第5章介绍。安装了Visual Studio后，一定要安装Visual Studio Server Pack 3或其更新的版本。要运行后续章节的样例应用，还必须安装SQL Server 7。一旦完成了SQL Server的安装，就不会影响使用最新的服务包。最后需要安装的是 Microsoft Platform SDK，请使用2000年1月或之后发布的版本，Platform SDK 提供了将在本书中描述的许多重要工具和应用程序。

更新和其他信息

作为一名研究人员、教师和作者，我不断地改进和创建新的与COM和COM+编程相关的Visual Basic应用程序。可以从我的Web站点（<http://SubliminalSystems.com>）免费下载最新的样例集。在该站点，你还将发现其他与本书有关的信息，包括bug或不准确之处的列表。如果你想给我发电子邮件，请寄到tedp@SubliminalSystems.com。希望你喜欢本书。

本书英文原书名：Programming Distributed Applications with COM+ and Microsoft Visual Basic 6.0, Second Edition。

英文原书号：ISBN 0-7356-1010-x

目 录

译者序	
序言	
前言	
第1章 COM+概述	1
1.1 为什么要使用COM+	1
1.2 Microsoft多层平台的发展历史	8
1.2.1 多层战略的基础: COM	8
1.2.2 从COM到MTS	11
1.2.3 从COM和MTS到COM+	13
1.3 理解COM+服务	14
1.3.1 Internet信息服务器	14
1.3.2 Microsoft消息队列服务	15
1.3.3 组件队列	15
1.3.4 COM+事件服务	16
1.4 使用Visual Basic创建分布式应用	17
小结	18
第2章 理解基于接口编程	20
2.1 面向对象编程的基本原理	20
2.1.1 改变类的公共接口	22
2.1.2 实现继承速成	23
2.2 分离接口与实现	26
2.2.1 继承的两种形式	27
2.2.2 利用Visual Basic使用接口	28
2.2.3 为什么要使用用户自定义接口	31
2.2.4 扩展类的定义	33
2.2.5 在应用设计中使用接口	34
2.2.6 接口与COM	35
小结	35
第3章 COM的基本原理	36
3.1 COM的产生	36
3.1.1 使用C++创建二进制组件	37
3.1.2 用虚表作为标准内存的代理	39
3.2 类型库和语言无关性	41
3.2.1 使用IDL	42
3.2.2 Visual Basic怎样映射到COM	43
3.2.3 全局唯一标识(GUID)	45
3.3 动态加载组件代码	46
3.3.1 对象激活	47
3.3.2 CLSID与Windows注册表	48
3.3.3 工作中的SCM	49
3.3.4 ProgID与CreateObject函数	51
3.4 了解IUnknown 与IDispatch	51
3.4.1 IUnknown接口	52
3.4.2 QueryInterface方法	53
3.4.3 IDispatch接口与Automation	55
3.4.4 双重接口	57
3.5 使用COM进行进程间通信	59
3.5.1 全局编排程序的作用	61
3.5.2 关于COM进程间通信	62
小结	64
第4章 创建和配置服务器	66
4.1 把组件编译到服务器中	66
4.1.1 进程内服务器	66
4.1.2 进程外服务器	67
4.1.3 为COM+封装组件	68
4.1.4 创建服务器	68
4.2 服务器配置问题	70
4.2.1 项目范围的类型库属性	71
4.2.2 组件的实例属性	71
4.2.3 修改过程属性	73
4.2.4 友元方法	75
4.2.5 使用枚举类型	75
4.2.6 使用UDT	76
4.3 为脚本客户设计组件	77

4.3.1 脚本客户与用户自定义接口	77	6.3.1 依据对象环境编程	130
4.3.2 关于脚本客户	83	6.3.2 理解调用环境	132
4.4 从服务器中捕获错误	84	6.4 测试和调试COM+组件	133
4.4.1 HRESULT	84	6.4.1 在Visual Basic调试器中运行配置型 组件	134
4.4.2 COM异常处理	87	6.4.2 其他可用调试技术	136
4.4.3 错误捕获约定	89	小结	137
4.4.4 在脚本客户中处理错误	90	第7章 在COM+应用中共享资源	138
小结	91	7.1 共享线程	138
第5章 组件版本的控制	92	7.1.1 Win32线程管理入门	139
5.1 COM中的版本控制	92	7.1.2 COM的线程管理模型	141
5.1.1 为脚本客户控制组件的版本	93	7.2 共享内存	150
5.1.2 为直接虚表绑定客户控制组件的 版本	94	7.2.1 单元与标准模块变量	150
5.2 生成用户自定义接口	102	7.2.2 共享资源管理器	151
5.2.1 使用IDL定义接口	103	7.3 共享数据库连接	155
5.2.2 使用IDL定义方法参数表	106	7.3.1 为什么需要数据库连接池机制	155
5.2.3 使用枚举型和UDT	108	7.3.2 连接池机制的工作过程	157
5.2.4 编译类型库	109	7.3.3 使用数据库连接池机制	158
5.2.5 分布和配置类型库	110	7.4 共享可装池对象	160
5.2.6 更麻烦的事情	111	7.4.1 对象池机制为什么重要	162
小结	112	7.4.2 对象池机制与数据库连接池机制	163
第6章 使用配置型组件	113	小结	164
6.1 配置型组件的基本概念	113	第8章 事务编程	165
6.1.1 COM+应用	114	8.1 什么是事务	166
6.1.2 创建和设置配置型组件	116	8.1.1 事务处理系统	167
6.1.3 库应用与服务器应用	118	8.1.2 回顾ACID规则	168
6.1.4 进程外激活	119	8.1.3 锁定与粒度	169
6.1.5 远程激活	121	8.1.4 写锁与读锁	170
6.1.6 应用代理	122	8.1.5 隔离级别	170
6.1.7 分布式废区收集	123	8.2 本地事务编程	171
6.1.8 分布式应用分类	124	8.3 分布式事务编程	173
6.2 理解环境	125	8.4 COM+与声明性事务	177
6.2.1 线程切换代理与轻量级代理	126	8.4.1 在COM+事务中创建对象	178
6.2.2 何时生成环境	127	8.4.2 三个重要标志: 满意位、完成位与 必败位	180
6.2.3 慎用New操作	128	8.4.3 SetComplete方法与SetAbort方法	182
6.2.4 环境与对象引用	128	8.4.4 IContextState接口	183
6.3 COM+编程模式	130		

8.4.5 AutoComplete属性	183	10.3.4 客户方组件队列编程	257
8.4.6 多对象事务	185	10.3.5 组件队列与异常处理	259
8.4.7 数据库连接与自动列集	188	10.4 COM+事件服务	261
8.4.8 COM+事务与事务存储过程	190	10.5 在MSMQ、QC和COM+事件间做出决定	268
8.4.9 事务对象短暂而“满意”的生命	190	小结	268
8.5 事务设计问题	192	第11章 COM+的安全性	270
8.5.1 COM+事务与本地事务	192	11.1 Windows 2000中的安全	270
8.5.2 分离商务逻辑代码与数据存取代码	193	11.1.1 当事人与授权	271
8.5.3 处理死锁	194	11.1.2 本地帐户与主域帐户	271
8.5.4 关于优化的讨论	197	11.1.3 用户帐户与组帐户	273
小结	197	11.1.4 预设帐户	273
第9章 为IIS和ASP创建组件	198	11.1.5 登录会话与令牌	275
9.1 HTTP与DCOM	198	11.1.6 资源与DACL	279
9.2 IIS/ASP体系结构	200	11.1.7 网络认证与模拟	280
9.2.1 ASP框架	201	11.2 COM+的安全特性	287
9.2.2 IIS应用	201	11.2.1 角色与授权	287
9.2.3 处理ASP请求	202	11.2.2 设置声明性安全	289
9.2.4 从ASP页创建Visual Basic对象	205	11.2.3 在运行时实施授权	290
9.2.5 把Web应用分割到独立的进程中	208	11.2.4 编程性安全	292
9.2.6 隔离的IIS应用	211	11.3 IIS中的安全	296
9.3 分离ASP页与组件之间的代码	213	11.3.1 HTTP安全标准	297
9.3.1 使用ASP页创建用户接口	213	11.3.2 证书	298
9.3.2 使用经过编译的组件创建用户接口	214	11.3.3 IIS认证模式	300
9.3.3 使用ASP页与组件的混合	217	11.4 进一步的资料	303
9.4 Web应用中的状态管理	218	小结	303
9.4.1 通过负载均衡伸缩Web应用	219	第12章 设计可伸缩的应用	305
9.4.2 缓存应用范围的状态信息	227	12.1 再论伸缩性	305
小结	228	12.2 在COM+应用中使用HTTP	307
第10章 消息处理与异步通信	230	12.2.1 分离HTTP与HTML	309
10.1 消息处理为何重要	230	12.2.2 使用HTTP与XML执行方法调用	310
10.2 MSMQ	234	12.2.3 SOAP与Web服务	313
10.2.1 MSMQ企业级模型	234	12.3 应用设计问题	314
10.2.2 MSMQ编程	235	12.3.1 使用COM传递数据	314
10.2.3 事务消息处理	245	12.3.2 使用MSMQ与QC传递数据	316
10.3 组件队列	251	12.3.3 使用HTTP传递数据	321
10.3.1 组件队列的体系结构	251	12.3.4 只读记录集与可更新记录集	324
10.3.2 设计组件队列	254	12.3.5 中间层数据缓存	326
10.3.3 配置组件队列	255	小结	327

第1章 COM+概述

一个千年已经过去，新的千年刚刚来临。作为一个专业开发人员，新的千年会给生活带来什么样的冲击呢？是的，当所有的钟表都嘀嘀嗒嗒地走向新千年的零点时，确实有个别计算机系统出现了所谓“千年虫”问题。一些过时的系统因此退出了历史的舞台，但是，从整体上看，计算机界并没有像新闻媒体所预测的那样大难临头，对于我们这些基于最后10年推出的操作系统和开发工具创建应用的人来说，更是如此。回过头来看看这一切，你也许觉得很可笑。那些声称公司的信息系统将面临灭顶之灾的流言，显然是言过其实了。

再者，新闻媒体真的了解计算机界所发生的事吗？他们忽略了一个明显的事实，那就是“千年虫”应该叫做“世纪虫”。这一问题源于某些系统使用的是两字符的日期，而不是四字符的日期。试想，如果100年之前人类具有同样的技术，那么在1900年新年到来之际我们会面临同样的问题。然而，“千年虫”听起来会感觉更好，报纸和杂志也会卖得更多。

还记得几年前的事吧，当时报刊曾预测，所有的东西将用Java重写，这包括美国宪法和联合国宪章吗？现在关于Java的闹剧已经平息了。我们已看到了Java到底是什么——一种新的、极有希望的语言，它既有优势，也有弱点。它并没有给计算机界带来暴风骤雨般的“革命”，它只是一种在某些情况下适用而在另一些情况下并不适用的语言而已。

现在每个人都在谈论XML，认为它是一种包医百病的技术。随着XML的日渐成熟，我们会发现它有更多的用场，它将解决许多非常困难、非常现实的问题。但是，XML将永远替代不了诸如COM、Corba和Java这样的技术。展望这些新技术固然十分重要，但是，当你在未来几个月内就要推出一个新应用时，不必对那些若干年后也许会解决所有问题的技术期望值过高。

1.1 为什么要使用COM+

你应该考虑使用COM+来创建分布式应用。许多公司正在使用COM+和Microsoft Windows 2000，这是因为它们提供了一个健壮的开发平台。这一平台由几种核心技术组成，这些技术为创建多层商务应用提供了基本构件。总的来说，基础平台提供的支持越多，需要用户编写和调试的代码就越少。

不言而喻，多层应用与20世纪80年代末和90年代初盛行的两层应用相比，拥有更多的优势。你的公司可能已经决定抛弃两层方法以支持多层策略。然而，在本章的开头部分，我想花点时间回顾一下两层应用中一些最重要的问题，并解释多层应用如何解决这些问题。我们也将讨论为什么多层开发带来了一些新问题和附加的复杂性。

Microsoft开发COM+的目标之一就是，在尽可能多地隐藏其内在复杂性的同时，为公司带来多层应用的益处。在过去的10年中，Microsoft公司在建立分布式应用基础结构方面已经取得了很多进展。

COM的第1版于1993年面世，从此，COM从一种年轻而复杂的技术逐步成为Microsoft多层战略的核心技术。本章将历数COM发展历程中最重要的里程碑；同时，我也将尽己所能来定义

这些年来在商界形成的所有缩写词。你也许已经听说OLE、DCOM、ActiveX和MTS，COM+和DNA是最近才添加到缩写词列表中的。但是，你是否尝试过在鸡尾酒会上向某人解释这些术语之间的区别？这可不是那么容易，是吧？它们对不同的人所代表的意义也有所不同。

本章给出了已被集成到COM+平台中的分布式服务的高层概览。任何大型的多层应用都需要诸如事务支持、集成安全性、Web服务器、消息处理机制和事件通知的传递等服务。本章将讲述这些构件是如何组装在一起的，这将从整体上给出COM+的概念并指明其用途。

从两层系统到多层系统

使用COM+的最好理由之一就是将一个公司的信息系统从两层结构转变为多层结构，这种转变需要设计和创建一个包含商务对象的中间层。商务对象通常位于客户应用和数据库服务器之间。COM+作为这类系统的平台被应用。

两层应用已被业界广泛地采用，因而，与之相关的问题也众所周知。让我们回顾一下如图1-1所示的传统两层结构具有哪些致命的弱点。

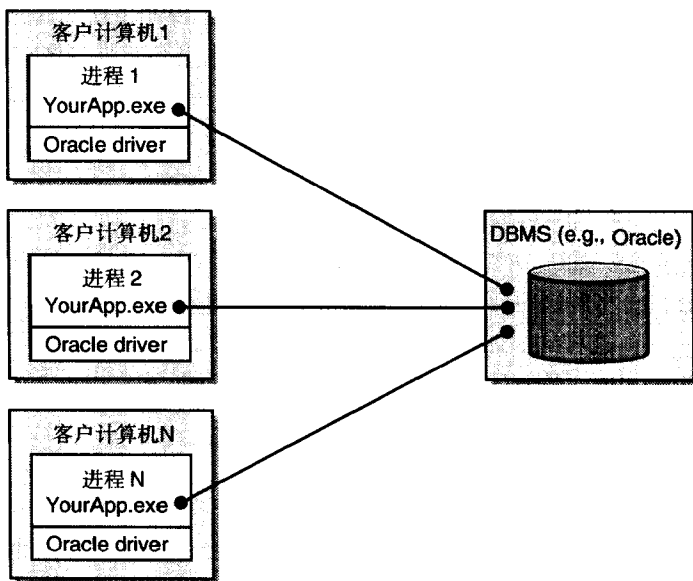


图1-1 典型的两层应用需要每个用户单独与数据库建立连接，并且，用户的计算机必须装有与特定数据库管理系统（DBMS）相应的驱动程序

- 用户接口商务逻辑以及数据存取代码相互混杂，这使得在多客户应用间重用商务逻辑和数据存取代码变得十分困难。更进一步，如果商务逻辑或采用的数据库发生了变化，则需要经常重新建立和分发客户应用。
- 每个客户应用由商务逻辑和数据存取代码来实现，都运行于各自相互独立的进程中，因此，它们不能共享进程专有资源，如线程、内存等。也不能在一个两层应用中共享数据库连接，因为每一个用户都需要单独与数据库服务器建立连接。客户应用连接到一个主机的情况也一样。这种不能在一组用户之间有效地共享资源的情况限制了系统的整体吞

吐量和可伸缩性。

- 每一个客户计算机需要一个或多个专用的驱动程序，以便与数据库服务器或一个主机应用会话，这样，就使得桌面（客户）计算机的维护和配置更加昂贵。系统管理员必须在每台计算机上安装和维护一组驱动程序，以提供对诸如开放数据库连接（ODBC）数据库等的访问。如果你需要更换后台数据库——例如将Sybase数据库更换为Oracle数据库——系统管理员就必须重新配置每一台桌面（客户）计算机。付出的代价相当昂贵。
- 客户应用从多数据源存取数据十分困难。如图1-2所示，在一个协作环境中，关键数据分布于许多不同的系统中也是十分常见的。当一个应用需要使用分布于多台计算机的数据运行商务处理时，事情会变得相当棘手，当这些数据的存储格式不同时，事情尤其难以处理。
- 客户应用通常要求与数据库服务器或主机应用在同一局域网（LAN）中运行，这样，建立跨越地理区域的分布式应用是不可能的。许多公司的雇员遍布全球，他们中的许多人需要为公司的客户和供应商建立应用，这些应用再也不能局限于像一个公司的LAN这样的受控环境之内了。
- 用户受限于特定的平台中。例如，两层系统通常要求每个客户都运行于同样的操作系统中，如32位的Windows环境。在当今任何一台计算机都有可能需要与另一台相连的世界中，这一限制越来越让人们不可接受。
- 当计算机之间断开连接时，例如，数据库服务器因需维护而离线，或者便携电脑用户在飞机上或在客户方工作而与网络断开连接，两层系统就会停止工作。因此，一旦各种计算机不能互连就会导致系统停止运转，系统的可用性就会受到严格的限制。

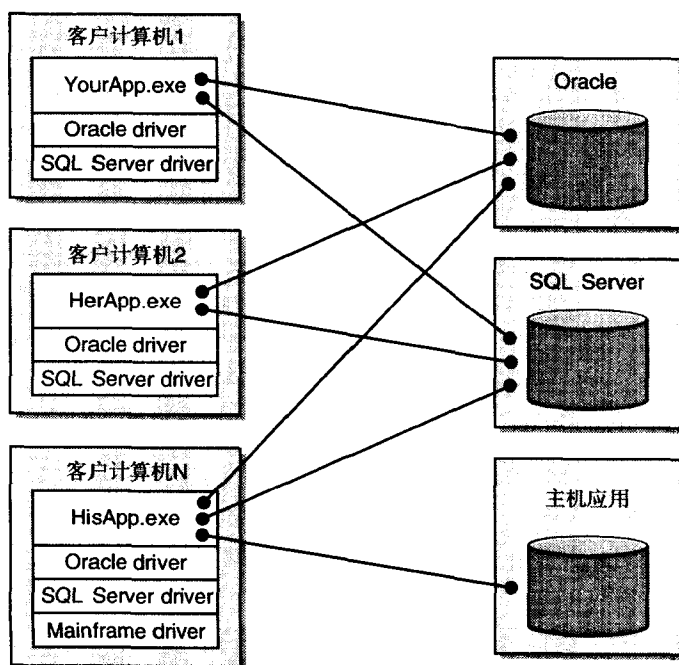


图1-2 当一个公司具有多个客户应用或多个数据库服务器时，两层策略不能很好地工作

1. 分离展示层

两层结构的这些问题可以像在计算机科学中解决其他问题那样加以解决——增加一个间接层。如图1-3所示，可以通过引入一组商务对象（business object）降低客户应用与商务逻辑及数据存取代码之间的耦合度。包含用户接口代码的客户应用通常称为展示层（presentation tier，或称为表示层）。除非另外说明；否则，本书中的“客户应用”（client application）均指包含用户接口元素和代码的一个展示层应用。

商务对象可使你的商务逻辑集中化，并可跨越多个客户应用。在基于Windows 2000和Windows NT的网络中，这些商务对象可以使用COM作为基础平台予以配置。COM 还可提供客户应用与中间层对象之间进行远程通信的基础。

COM最大的卖点之一是它允许中间件程序员更新他们的商务对象代码，而无需重新编译或重新分发客户应用。由于客户应用被屏蔽于商务对象的实现细节之外。这样，在你的组件形成产品之后，仍可以相当直接而轻松地改变商务逻辑或数据存取代码。

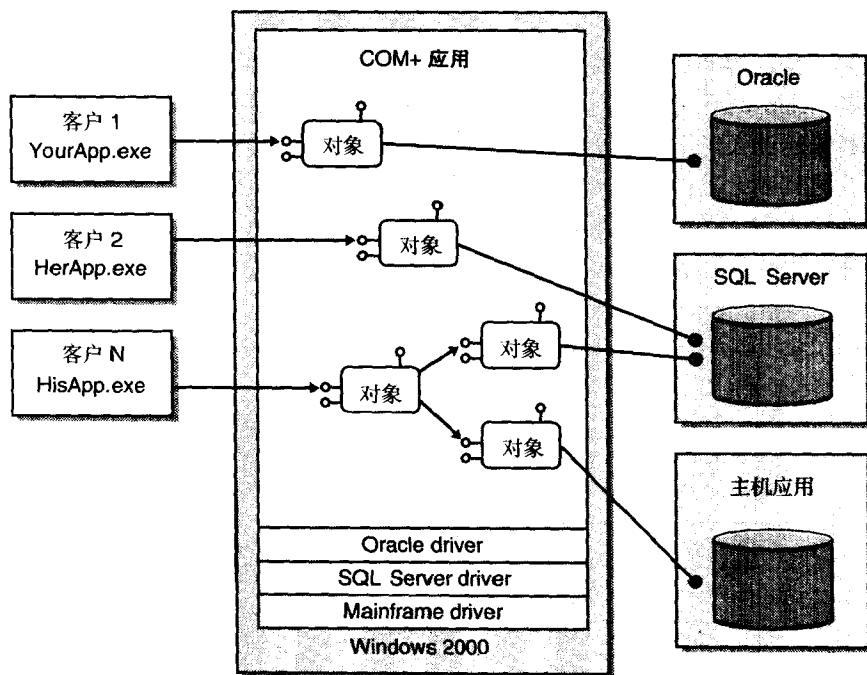


图1-3 在中间层中引入一组商务对象消除了客户应用与数据库服务器之间高代价的依赖关系

一个应用的数据可以改变存储格式，并且，新的数据库服务器可以不必遭受过度的痛苦和磨难而在线装配。你可以经常对一个单一的Microsoft Visual Basic动态链接库（DLL）做些必要的修改。重新编译一个DDL并在产品服务器的中间层替换它是容易的，而产品中的客户应用则可以保持不变。

在如图1-3所示的一个典型多层应用中，一组中间层对象代表许多用户运行。然而，所有这些对象经常运行于Windows 2000服务器的一个单一进程中，这就使得在多用户之间共享诸如线程、内存和数据库连接这样的进程专有资源成为可能。另外，与两层模型不同，运行客户应用

的计算机不需要数据库驱动程序。

2. 逻辑层与物理配置

多层开发的首要问题是每一层应配置于何处。图1-4给出了两种可能的情况。层的数量并不一定意味着计算机的数量，在一个小型的配置中，商务对象和数据库服务器可能运行于同一台计算机上，在一个较大的系统中，数据可以保存在一台或多台专用计算机上，而商务对象则运行于另一台独立的主机中。

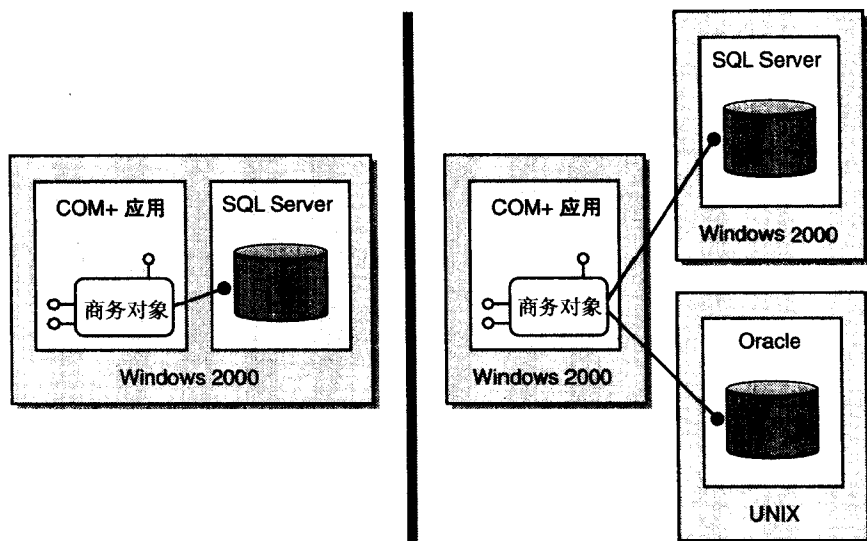


图1-4 商务对象与数据存取层之间的映射一开始可以很简单，之后，它可变得更复杂些，但这不影响客户的应用

有些物理配置模式较为容易建立，开销也不大，而另一些则提供了较高的可伸缩性、可靠性和容错性。当谈到多层系统时，一般指的是几个相互独立的逻辑层（logical tier）。而物理配置可以不同，并且当系统形成产品后，物理配置也可以方便地改变。COM的最大威力在于，它允许你改变物理配置，而无需修改和重编译任何应用代码。

一些人使用三层(Three-tier)或N层(N-tier)代替多层(multitier)这一术语。大部分情况下，这些术语的含义是一样的——系统有三层或更多的层。图1-5给出了更为复杂的一组物理上的层。在多层模型中，商务逻辑和数据存取层的复杂度可以是任意的。这一模型的最大优点是客户应用只知道商务对象的一个可视层，这些商务对象背后附加的复杂性与它们无关。因此，一个多层系统的最基本的设计目标是对组成展示层的客户应用尽可能多地隐藏这种复杂性。

COM的另一个强大特性是它允许客户应用在网络间创建和使用对象。在后台，COM使用远程过程调用(Remote Procedure Call, RPC)协议跨越进程和主机的边界进行方法调用。RPC是当你使用COM+设计分布式应用之前，应该理解的几个重要协议中的第一个。

如果你有一组均支持COM的操作系统（如Windows 2000、Windows NT和Windows 98）上的用户，则可以创建如图1-5所示的一个多层应用。运行客户应用的计算机依靠COM跨越LAN（局域网）与商务对象建立连接。一旦一个客户应用在网络间创建了商务对象，它就可以使用这

些商务对象运行事务和检索信息。

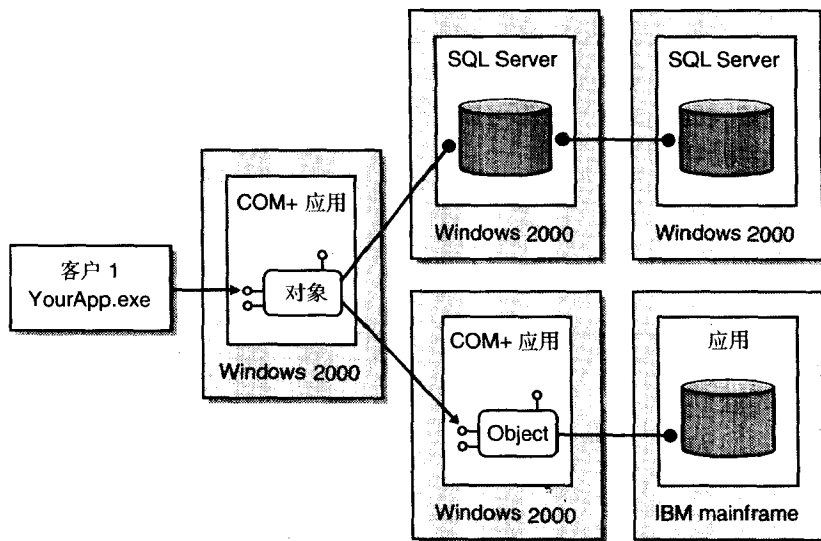


图1-5 多层结构的最大威力之一就是，它对客户应用隐蔽了公司不断演变的IT基础结构的复杂性

尽管上述方法对于某些应用工作得很好，但对另一些则并不十分适合。这种开发风格通常要求每个用户都运行于Windows的32位版本。当你的所有客户应用和商务对象都在同一LAN中运行时，它工作的依赖性就更大。然而，你可能想让那些还没有登录到一个局域网上的用户，或那些运行于Macintosh、OS/2或Unix这样的操作系统上的用户也能使用系统。可以使用一种基于Web的开发策略使用户的数量大大增加。

3. 基于Web的应用

Internet和基于Web的应用的流行刺激了业界采用多层结构。在一个基于Web的系统中，客户应用运行于浏览器中，浏览器使用一种简洁的超文本传输协议（Hypertext Transfer Protocol, HTTP）向Web服务器提交请求。在一个基于Web的应用中，展示层是用超文本标记语言（Hypertext Markup Language, HTML）建立的。

一个典型的Web应用的工作方式其实相当简单。客户向Web服务器发出一个请求，Web服务器作出相应的响应，处理这一请求，并将一个基于HTML的页返回给该用户。使用HTTP和HTML的高明之处在于各种主流平台都支持它们。这样，你的应用就具有了可以到达Internet上的任何用户的潜力。

值得注意的是，如图1-6所示，Web服务器是客户进入中间层的入口点。当客户提交一个HTTP请求时，中间层应用必须加载和运行商务对象以实现多层开发的优势。即使是组成展示层的代码和HTML页存在于服务器上时，你仍然可以实现用户接口元素、商务逻辑和数据存取代码的相互独立。正如你将在后续章节看到的那样，可以建立一组独立的商务对象，并在基于LAN的客户间以及在基于Web的客户间共享它们。