

本书由微软TechEd2000、TechEd2001、PDC2002、MSDN DevCon等大型研讨会、Visual Studio .NET中文版巡回发布会主讲人员规划与编校：

- 由台湾Visual Basic 6.0最佳畅销教材（连续两年）作者编写
- 由入门到提高的程序设计书
- Visual Basic .NET程序员的工具书

内附光盘

Visual Basic .NET 程序设计经典

详细的Visual Basic .NET语法介绍、Console应用程序设计、Windows Form应用程序设计、最完整的控件介绍、窗口事件探讨、ASP.NET Web和Mobile Web应用程序设计、Access和SQL Server数据库存取技巧、XML Web Services、GDI+、Crystal Report报表设计、面向对象程序设计、窗口与Web用户自定义控件、多项目开发技巧应用程序的安装与部署、程序调试与异常处理技巧等21个单元，各单元都有精心设计的实用范例，用以体现设计精髓。

Microsoft Certified
Professional

Database Administrator
Systems Engineer
Solution Developer
Trainer

曹祖圣 吴明哲 黄世阳 编著
蔡文龙 林义证

本书由微软专业认证合格专家编校



科学出版社
www.sciencep.com

图字: 01-2003-6990 号

内 容 简 介

本书是探讨 Visual Basic .NET 程序设计的进阶书籍。全书共 21 章, 分为 6 个单元: Console、Windows Form、数据库、Internet 网络程序以及其他相关技术和面向对象编程。本书篇幅较大, 包含了初级及进阶内容。另外, 每章都提供了许多精心设计的范例, 以便于读者学习。

本书内容循序渐进, 范例经典, 适合初、中级程序员阅读, 也可作为本科教材。

本书繁体字版原书名为《Visual Basic .net 程式设计经典》, 由文魁资讯股份有限公司出版, 版权属曹祖圣、吴明哲、黄世阳、蔡文龙、林义证所有。本书简体字中文版由文魁资讯股份有限公司授权科学出版社独家出版。未经本书原版出版者和本书出版者书面许可, 任何单位和个人均不得以任何形式或任何手段复制或传播本书的部分或全部。

版权所有, 翻印必究。

图书在版编目(CIP)数据

Visual Basic .NET 程序设计经典/曹祖圣等编著. —北京: 科学出版社, 2003

(程序设计经典书系)

ISBN 7-03-012290-9

I.V... II.曹... III.BASIC 语言—程序设计 IV.TP312

中国版本图书馆 CIP 数据核字(2003)第 088788 号

责任编辑: 舒 立 / 责任校对: 都 岚

责任印制: 吕春珉 / 封面制作: 一克米工作室

科 学 出 版 社 出 版

北京东黄城根北街16号

邮政编码: 100717

<http://www.sciencep.com>

双青印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2003 年 11 月第 1 版 开本: 787×1092 1/16

2003 年 11 月第一次印刷 印张: 47 1/4

印数: 1—4 000 字数: 1 117 400

定价: 79.00 元 (含光盘)

(如有印装质量问题, 我社负责调换〈路通〉)

序

自从因特网 (Internet) 问世至今, 在网络上相关软件的开发技术几乎是不到两年就翻新一次, 而在微软的软件开发工具中, Visual Studio 一直是 Windows 平台上集成能力最强、功能最齐备的程序开发工具。新版的 Visual Studio .NET 更强化多项软件开发技术, 尤其是 Visual Basic .NET, 不但拥有易学易用的特性, 让初学者得以轻松进入程序设计的殿堂, 更拥有完整的面向对象机制, 这使得使用 Visual Basic .NET 来开发大型的软件系统时, 降低了软件设计与维护的复杂度。

在软件开发架构上, .NET Framework 将成为未来软件开发的通用平台, 由于 .NET Framework 容易移植到各个不同的平台上 (Windows, UNIX, PDA, 手机...), 因此 Visual Basic .NET 程序员可以直接运用 .NET Framework 中完整且丰富的类链接库, 设计出跨平台的软件系统; 另外通过通用语言运行库 (Common Language Runtime) 的接口, 在 .NET Framework 上所开发的软件, 不论使用哪一种程序语言, 都可以直接互相引用, 在多人共同开发软件系统时, 程序员就可以使用自己所专长的程序语言来参与软件开发, 直接解决了跨语言集成的问题。

微软提出的 .NET 远景 (Vision) 所要建立的信息应用环境, 就是让用户可以在任何时刻 (any time)、任何地点 (any place)、使用任何设备 (any device) 来存取所需要的信息, 而 .NET Framework 与 Visual Studio .NET 正是用来建构 .NET 远景的基础平台与开发工具, 通过开发 XML Web Service 与相关的机制, 不但可以集成各类型平台上的开发成果, 降低企业 e 化成本与缩短导入时间, 更强化了分布式系统的安全性及多功能性, 这一切正是建构 .NET 的骨与肉所不可或缺的基石。

本书是探讨 Visual Basic .NET 程序设计的进阶书籍。本书重点为 .NET 技术的介绍与实作, 内容涵盖了 .NET 程序开发的大部分领域, 尤其在数据库方面, 除了以 Access 数据库来探讨 ADO.NET 外, 在书附光盘中亦附上 SQL Server 对应范例写法, 让学习者能同时对 Access 和 SQL Server 数据库存取有深切的认识。

按照本书规划的教学大纲, 本书可作为一学期 (未规划章节可当参考书) 或一学年的教科书。同时为了方便老师上课, 采用本书的教师可通过 allok@tcts.seed.net.tw 信箱询问本书相关的 Visual Basic .NET 问题, 来信时请注明服务学校及任教科目。本书虽然经过多次精心的校对, 但难免百密一疏, 尚祈读者不吝指正, 以期再版时能更趋扎实。

笔者: 曹祖圣 吴明哲 黄世阳
蔡文龙 林义雄
2002 年 8 月

前 言

1. 本书分成六大单元

(1) Console 单元 (第 1~5 章)

本单元使程序设计初学者了解 .NET 程序架构与功能, 训练基本程序设计素养, 在 Console 模式下了解程序语言的数据类型, 学习如何编写有关流程控制、数组、过程、方法 (函数) 简短程序以及进行程序调试, 以便将这些技能应用到下面章节的范例。

(2) Windows Form 单元 (第 6~9 章)

本单元让学习者熟悉 Windows Forms 下工具箱群组中近 95% 控件的使用时机与编写方式。本单元将控件分成基本控件、常用控件、特殊控件三章, 详细介绍各控件常用属性和方法, 并说明如何在程序设计或执行阶段设置属性和使用方法, 并以具有代表性的范例作解说。

(3) 数据库单元 (第 10~12 章)

本单元让学习者熟悉如何建立和存取 Access 2002 和 SQL Server 2000 数据库。本单元以 ADO.NET 所提供的工具向导和 SQL 命令, 介绍建立和维护数据库、与数据库联机、读取数据库的记录、改动数据库、数据与控件绑定的技巧。限于篇幅所有范例以 Access 2002 数据库解说为主, 至于 SQL Server 2000 完整范例放在书附光盘中, 让学习者熟悉这两种数据库程序设计。

(4) Internet 网络程序设计单元 (第 13~15 章)

本单元让学习者熟悉目前最流行的 ASP.NET 网页数据库程序设计、Mobile Web Form 移动装置数据库程序设计, 以及如何通过 ASP.NET 来读写 XML 文件的技巧。此外, 还介绍如何设计存放在 Internet 计算机上的 XML Web Service, 以供前端用户调用。

(5) 其他相关技术单元 (第 16~19 章)

本单元包含如何使用 Crystal Reports 来制作窗口应用程序、ASP.NET 报表 XML Web Service 的报表视图程序, 多项目开发、集成 Visual Basic .NET 及 C# .NET 项目技巧, 部署与安装窗口及 Web 应用程序, GDI+ 绘图与多媒体设计, 以及使用 GDI+ 来设计不规则形状的窗口。

(6) 面向对象单元 (第 20~21 章)

本单元让学习者了解对象与类的概念, 并学会 Visual Basic .NET 面向对象程序设计技巧, 且在本单元还介绍如何设计窗口应用程序及 ASP.NET 的用户自定义控件。

2. 课程规划 (以一周三课时规划下列课程进度表)

课程进度表

章名	一学期课程	学年课程
第1章 .NET Framework 介绍	0.5 周	0.5 周
第2章 Visual Basic .NET 数据类型	1 周	1 周
第3章 流程控制	1.5 周	1.5 周
第4章 数组与字符串	1.5 周	1.5 周
第5章 过程与异常处理	1.5 周	1.5 周
第6章 Windows 窗体	1.5 周	1.5 周
第7章 常用窗体控件	1 周 (择要)	1.5 周
第8章 特殊控件	1 周 (择要)	1.5 周
第9章 窗口事件应用技巧	1 周	2 周
第10章 ADO.NET 与数据工具	1.5 周	1.5 周
第11章 ADO.NET 数据库连接与访问	1.5 周	1.5 周
第12章 ADO.NET 数据库浏览与绑定		1.5 周
第13章 ASP.NET Web 应用程序	Option	1.5 周
第14章 Mobile Web 应用程序设计		1.5 周
第15章 XML Web Service		1.5 周
第16章 Crystal Reports 报表视图程序	1 周	1 周
第17章 多项目开发		1.5 周
第18章 部署和安装应用程序	0.5 周 (择要)	1 周
第19章 GDI+ 绘图与多媒体播放技巧		1.5 周
第20章 Visual Basic .NET 面向对象技术	1 周 (前半)	2 周
第21章 用户自定义控件		Option

目 录

第 1 章 .NET Framework 介绍	1
1.1 本章概述	1
1.2 软件开发技术的演进	1
1.3 Microsoft .NET	7
1.4 .NET 架构	8
第 2 章 Visual Basic .NET 数据类型	18
2.1 变量与数据类型	18
2.2 Visual Basic .NET 运算符	24
2.3 基本输入输出使用 Console 类	28
2.4 结构	34
2.5 枚举数据类型	36
2.6 类	39
第 3 章 流程控制	42
3.1 选择结构	42
3.2 循环结构	50
第 4 章 数组与字符串	58
4.1 数组	58
4.2 ArrayList 类	63
4.3 IsArray 函数	67
4.4 字符串	67
4.5 Visual Basic 传统字符串函数	68
4.6 Visual Basic .NET 字符串类的成员	70
第 5 章 过程与异常处理	74
5.1 过程	74
5.2 常用内置函数与数学类对象	75
5.3 用户自定义过程	81
5.4 程序调试	95
5.5 异常处理	98
第 6 章 Windows 窗体	106
6.1 Windows 窗体	106
6.2 认识 Visual Basic .NET 集成开发环境	106
6.3 项目的维护	117
6.4 窗体上的控件	118

6.5	窗体和控件常用的成员	121
6.6	第一个 Windows 窗体应用程序	125
6.7	窗体的常用事件过程	133
6.8	控件的命名	133
6.9	Label 标签控件	134
6.10	Timer 计时器控件	137
6.11	LinkLabel 链接标签控件	140
6.12	ToolTip 提示控件	142
6.13	TextBox 文本框控件	145
6.14	InputBox 输入函数	148
6.15	MsgBox 函数与 MessageBox.Show 方法	150
第 7 章	常用窗体控件	160
7.1	Button 按钮控件	160
7.2	RadioButton 单选按钮控件	167
7.3	CheckBox 复选框控件	168
7.4	容器控件	169
7.5	RichTextBox 丰富文本框控件	173
7.6	列表控件	177
7.7	旋转控件	185
7.8	日期时间控件	188
7.9	滚动条控件	194
第 8 章	特殊控件	201
8.1	图形处理控件	201
8.2	Dialog 控件	208
8.3	ToolBar 控件	212
8.4	StatusBar 控件	217
8.5	菜单控件	219
8.6	ContextMenu 控件	223
8.7	选项卡控件	225
8.8	TreeView 树状视图控件	230
8.9	窗口分隔控件	240
8.10	文件控件	243
第 9 章	窗口事件应用技巧	255
9.1	键盘事件介绍	255
9.2	鼠标事件介绍	263
9.3	控件共享事件过程技巧	267
第 10 章	ADO.NET 与数据工具	277
10.1	本章概述	277
10.2	数据库的概念	277

10.3	数据库的创建步骤	280
10.4	ADO.NET 简介	280
10.5	.NET Data Provider	283
10.6	使用数据工具创建数据库应用程序	284
10.7	数据库的关系查询	296
10.8	Binding Manager	302
10.9	数据窗体向导	310
第 11 章	ADO.NET 数据库连接与访问	316
11.1	引用 ADO.NET 命名空间	316
11.2	运用 Connection 对象打开、关闭数据库的连接	317
11.3	运用 DataReader 对象读取数据库	319
11.4	如何运用 DataSet 对象来读取数据	327
11.5	SQL 语法	335
第 12 章	ADO.NET 数据库浏览与绑定	349
12.1	DataView 对象	349
12.2	数据绑定	353
12.3	数据表的关系	358
12.4	使用 BindingManagerBase 来移动数据记录	363
第 13 章	ASP.NET Web 应用程序	369
13.1	虚拟目录	369
13.2	创建 ASP.NET Web 应用程序	378
13.3	创建 ASP.NET Web 数据库应用程序	382
第 14 章	Mobile Web 应用程序设计	409
14.1	必要软件安装	409
14.2	Mobile Web 窗体开发	420
14.3	多窗体画面切换	422
14.4	数据库连接	425
第 15 章	XML Web Service	430
15.1	XML 简介	430
15.2	读写 XML 文档	433
15.3	XML Web Service	448
第 16 章	Crystal Reports 报表视图程序	472
16.1	Crystal Reports 简介	472
16.2	如何重新设置报表文件及数据库源	473
16.3	Windows 窗体的报表视图程序	478
16.4	Web 窗体的报表视图程序	497
16.5	报表 XML Web Service	503
16.6	图表、子报表的设计	508

第 17 章 多项目开发	522
17.1 解决方案与项目	522
17.2 模块 (Module)	523
17.3 多个模块的程序设计	523
17.4 如何创建多个模块的项目	526
17.5 多模块项目常用的语句与方法	530
17.6 多个项目的操作	535
17.7 调用应用程序	550
17.8 多项目开发使用 Visual Basic .NET 与 C# .NET	554
第 18 章 部署和安装应用程序	563
18.1 窗口应用程序的部署和安装	563
18.2 Web 应用程序的部署和安装	575
第 19 章 GDI+ 绘图与多媒体播放技巧	584
19.1 颜色与坐标	584
19.2 绘图对象	588
19.3 图形文件的访问	609
19.4 图形的区域与不规则窗体窗口	613
19.5 录音程序	617
19.6 媒体播放程序	623
19.7 API 播放语音函数	624
19.8 调用 Windows Media Player	631
第 20 章 Visual Basic .NET 面向对象技术	636
20.1 前言	636
20.2 对象与类	636
20.3 面向对象技术	641
20.4 面向对象程序设计流程	649
20.5 对象与类的创建	650
20.6 继承与多态性	664
20.7 接口与实现 (Interface & Implements)	679
20.8 引用外部类库	685
第 21 章 用户自定义控件	692
21.1 创建窗口应用程序的用户自定义控件	692
21.2 创建 ASP.NET Web 应用程序的用户自定义控件	706
附录 A 安装 Visual Studio.NET	713
附录 B ASCII 表	722
附录 C 使用 Access 2002 建立数据表	723
附录 D 使用 SQL Server 2000 建立数据表	732

第1章 .NET Framework 介绍

1.1 本章概述

软件系统的规划与建立，以往总需要针对不同操作系统平台、程序语言、软件技术等集成，然而这样的集成往往费时耗力，更成为增加软件复杂度的祸首之一，因此为了简化软件规划与设计成本，学术界与商业界便不断推出更加先进的软件设计技术与产品，以适应日趋复杂的软件开发需求。

本章将深入讲解整个软件开发环境，包括规划、设计、集成等领域，从过去到现在的进展，以及针对目前的软件开发困境市场上有哪些解决方案，最后将详细介绍未来的软件开发平台 .NET Framework，看看微软的 .NET 远景是如何通过 .NET Framework 构建，以及新一代 Visual Studio .NET 开发环境与 Visual Basic .NET 语言所能带来的软件开发优势。

另外，我们也将针对分布式软件开发技术，尤其是目前主流的 XML Web Services 技术，加以深入的剖析。在完成整个 Internet 软件开发技术的介绍之后，你将深刻了解到如何通过 Visual Studio .NET 软件开发工具，以及 Visual Basic .NET 程序语言，构建出以 .NET Framework 为坚实基础的软件组件与系统，再通过 XML Web Services 延伸至 Internet，最后集成 Internet 上不同平台上的不同系统以及各类型的客户端设备，勾勒出 .NET 的无限商机。

1.2 软件开发技术的演进

一项新的软件开发技术或平台的推出，往往是因为原有的技术或平台已不足以应付目前软件开发的需求。因此如果能够了解软件开发技术的演进，就更能清楚了解 .NET Framework 这个新一代的软件开发平台与技术的来源和意义。

以下分别就软件分析、程序设计、分布式技术与软件平台四个部分，说明软件开发技术的演进。

1. 软件分析方法

在整个软件系统开发流程的前期，系统分析是相当重要的一部分。在面向对象技术出现之前，结构化程序规划与设计方式一直是软件开发的主流技术。然而为了适应日益复杂的软件系统，程序设计技术出现了重大变革，那就是面向对象（Object Oriented）技术，于是以往结构化、模块化的软件分析方法已经不适用于超大型软件系统，于是面向对象分析（OOA，Object Oriented Analysis）与面向对象设计（OOD，Object Oriented Design）的方

法,在 1989 年到 1994 年之间增加到 50 多种,然而却导致任一分析方法都无法适用各类型系统分析的问题。

1997 年由 Grady Booch, James Rumbaugh, Ivar Jacobson 三位软件分析大师与微软、Digital、HP、IBM、Oracle 等公司共同制订的统一建模语言 (Unified Modeling Language) 软件分析方法,被 OMG (Object Management Group) 协会认可,正式成为主流的面向对象分析与设计的方法。

虽然通过 UML 规划的软件规范书可以节省整体规划与集成所需的时间,然而除非所有参与后期程序设计的程序员也充分了解 UML 语法,否则仍需花费时间对程序员进行 UML 的培训。

针对软件分析与设计的问题,在 Visual Studio .NET 企业架构 (VSEA) 版本中,可以直接使用 Visual Studio .NET 的方案规划工具,进行系统设计与模型构建(称为企业模版)。当然也可以将设计结果输出成标准 UML 格式,或者将 UML 文件导入到 Visual Studio .NET 环境中,因此后续的程序员只要会使用 Visual Studio .NET 开发环境,即便不了解 UML 语法,也可以直接进行程序项目的设计。

2. 程序设计方法

现在的程序员往往埋头于几十万行以上的程序代码中,竭尽全力地维持软件系统的可靠性与稳定性,然而传统的结构化程序设计技术早已无法胜任日益复杂的软件系统,于是在程序设计方法的历史演变中,面向对象 (OO, Object Oriented) 算是近代软件系统分析、规划与设计概念的一大突破,也无疑是大型软件系统的救世主。

在 OO 的领域中,除了先前提到的面向对象分析与面向对象设计之外,另一项与程序设计密不可分的技术就是面向对象程序设计 (OOP, Object Oriented Programming),所谓 OOP 就是使用面向对象的概念来编写程序。有经验的程序员甚至可以使用不支持面向对象的程序语言编写出面向对象的程序,例如汇编语言(不要怀疑,笔者就曾经写过),但经验不足的程序员也有可能使用诸如 C++ 之类支持面向对象的语言,编出完全不像面向对象的程序。

笔者经常被学生问道:如何将长度不到十行的程序代码改用面向对象重写?例如,如何将原本使用 for 循环编写的九九表程序,改用面向对象的语法重新编写?也有些负责企业 MIS 或 ERP 系统的程序员问道:如何将系统中所有的程序改成面向对象?似乎所有的人都将面向对象当成灵丹妙药,将传统结构化程序设计视为罪魁祸首。其实面向对象程序设计技术并不是用来彻底淘汰原来的程序设计方法,相反的,面向对象的目的是弥补传统结构化程序设计中,大型或复杂系统难以设计与维护的缺点,于是“如何将 for 循环编出的九九表程序,改用面向对象的语法重新编写”这样的要求,显然就太过吹毛求疵了。

曾经听过这么一个故事,有一个拥有十家分公司的跨国企业,各分公司的 MIS 系统可以统计某一月份的营销总额,然后将营销报表送交总公司备案。公司的总裁拿到这十份营销报表后,进一步想得到该月份所有分公司的营销总额,于是召开会议来商讨解决方案,会议中有人建议将目前各分公司的 MIS 系统改写,建立企业的 ERP 系统,将各分公司与总公司系统联机,以进行营销总额的统计;也有人建议委托其他软件顾问公司进行系统规划与设计;正当大家争执不休时,只见总裁的特别助理拿出计算器,将报表上十个分公司的

营销金额相加后,不到三十秒就算出该月份全公司的营销总额。

当然,在适当的地方使用适当的技术来解决问题才是软件设计中至高的法则,而这适当的技术不见得非面向对象不可,也不见得非得要用程序设计方法来解决。

3. 分布式软件技术

软件系统按功能分类,可以分成以下几种不同的层次结构:

- 表示层
在软件系统的表示层 Presentation 中,主要负责用户界面的处理,包含数据的输入与显示、图表的绘制...等与数据展现有关的工作。
- 业务逻辑层
软件系统中都包含了许多逻辑的处理,例如银行利率的计算、转帐、账号与密码的检查、数据的加密与解密等,这些工作都属于业务逻辑层(Business Logic)的工作范围。
- 数据源层
所谓数据源层(Data Source)就是软件系统中数据保存的位置,例如 Microsoft SQL Server、Oracle、Sybase、Access 等数据库,甚至 XML 数据都可以称为数据层。

以软件系统的结构(以上三个层次的组合方式)来看,大致可以分成以下类型:

- 单机结构
在单机结构(Stand Alone)中,表示层、业务逻辑层与数据源层都位于同一个软件、同一部计算机中。例如 dBase、Access、FoxPro 等开发的单机应用程序,由于整个系统不需要牵涉到与其他计算机或系统的集成,因此在设计上非常简单,在单机上的效率也不错,适合个人操作使用。然而由于数据位于软件所在的计算机上,因此如果要进行不同计算机之间的数据交换,往往需要通过文件复制来达成,非常的不方便。
假设银行的系统采用单机结构开发,那么在北京某一分行存款 10 000 元,这个存款数据存放在该办理员的计算机上,必须将该存款数据拷贝到磁盘上,然后复制到全国其他分行的所有职员计算机中,才能在其他分行中看到这笔新的存款数据。如果要进行复杂的转帐交易,单机结构就明显力不从心了。
- 客户机/服务器结构(Client/Server)
为了解决单机结构数据层分散、难以同步的问题,我们可以将数据层独立出来,如图 1.1。
于是不论使用哪一个客户端系统,其数据层都来自同一个位置,因此客户端 A 储存的数据,客户端 B 可以立即取到,省去许多数据同步的问题,而数据层通常是数据库管理系统,例如 Microsoft SQL Server。
除了后台的数据层独立出来之外,客户端当然扮演展示层的角色,然而业务逻辑在这个结构中该放在哪里呢?

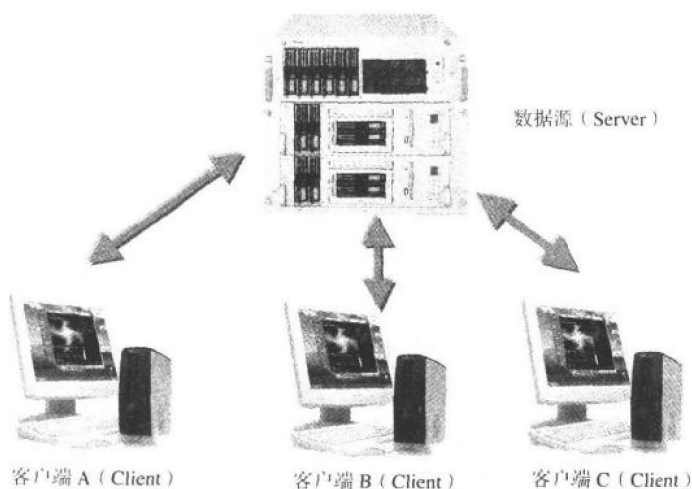


图1.1 客户机/服务器结构图

- **Heavy Client**

将业务逻辑与表示层一同放在客户端上，称为 **Heavy Client**。由于业务逻辑需要大量的运算，因此放在客户端上会造成客户端执行效率下降，再加上一旦业务逻辑有变动，就必须将新版的程序重新安装在所有的客户端上，在软件安装业务与管理上非常耗时费力。

- **Thin Client**

将业务逻辑与数据层放置在一起，客户端只做表示层工作，称为 **Thin Client**。在这个结构中由于业务逻辑放在数据层所在计算机上，因此在修改业务逻辑时比 **Heavy Client** 方便许多，缺点是会降低数据层的执行效率。通常放置在数据层的业务逻辑使用存储过程 (**Stored Procedure**) 来处理，一般中小企业都采取这样的软件结构。而存储过程是 **SQL Server** 上可调用执行的 **Server** 端程序。

- **三层结构 (3-Tiers)**

不论将业务逻辑放置在数据源层、还是客户端，两者都有优缺点。因此如果将业务逻辑独立出来，形成三层结构（如图 1.2），就可以扬长避短了。

将业务逻辑独立出来，不论是在业务逻辑的更新与维护上，还是对于客户端与数据层的执行效率上都有很大的帮助，其中负责处理业务逻辑的那部 **Server**，就是业界俗称的 **AP Server**。

- **多层结构 (N-Tiers)**

三层结构中客户端程序要修改时，例如更改操作界面，就必须将新版客户端程序安装到所有的客户端计算机上，在客户端维护上仍然非常不方便。另外业务逻辑层需要大量运算资源，往往会造成整个系统的瓶颈，因此在多层结构中（如图 1.3 所示），可以将业务逻辑层再加以分层，分层负责各类型运算处理。例如转帐交易中会使用到存款与取款的处理程序，就可以将转帐与存取款处理程序分别放在两部计算机上，将运算需求分散，借此提高整个业务逻辑的执行效率。

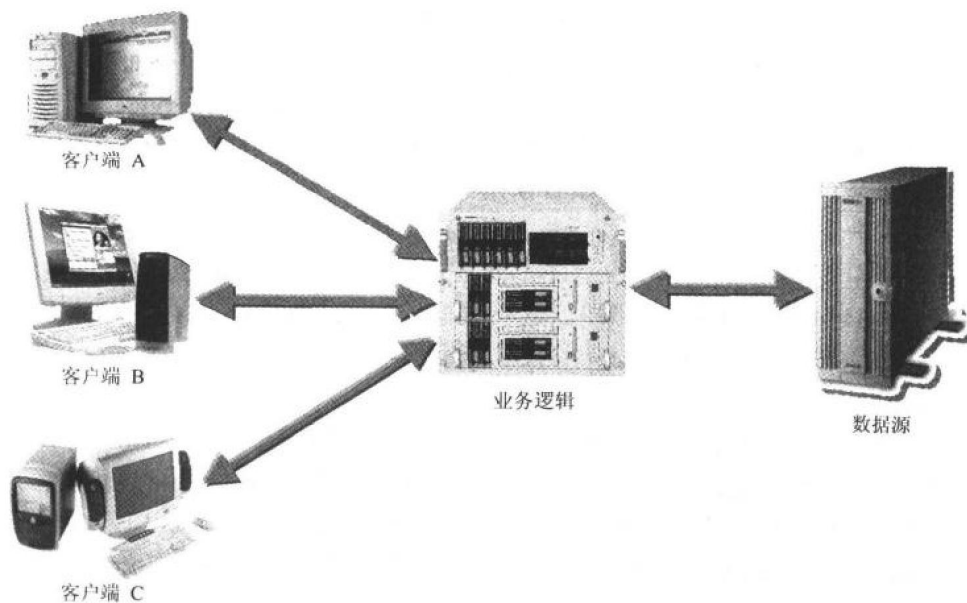


图1.2 三层结构

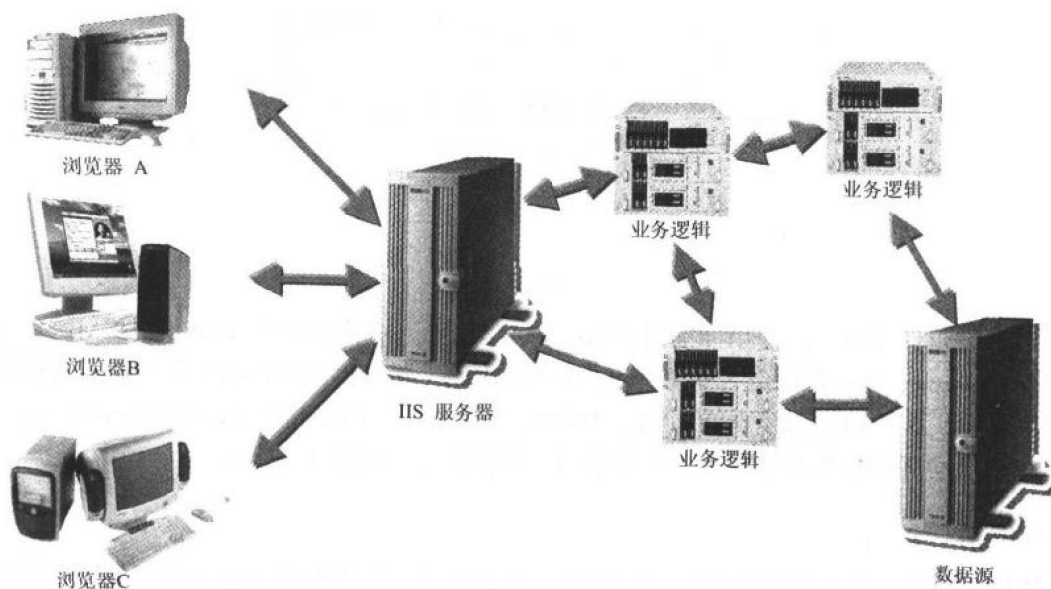


图1.3 多层结构

另外，再配合使用 ASP、PHP 等网站服务器（例如 IIS）端程序绘制客户端操作界面与处理客户端输入数据，这样客户端只要有浏览器就可以使用系统。将来如果需要修改用户界面，也只需修改 ASP 程序，用户只要单击浏览器上的刷新按钮即可得到最新的用户界面了。

- Windows DNA 2000

微软在实现 N-Tiers 结构上，提供了 Windows 分布式网络体系结构（DNA，

Distributed Internet Architecture) 2000 架构 (前身是 Windows DNA), 在这个架构中, 使用组件对象模型 (COM, Component Object Model) 做为组件设计规范, 配合 Windows 2000 上的组件服务 (Component Service), 构建出多层次结构中需要的业务逻辑组件, 后台配合 Microsoft SQL Server, 前台可以使用 Windows 应用程序, 或者使用 ASP 开发 Web 应用程序。

在这个架构中, 可以集成所有以微软为主的解决方案, 但是对于非微软的系统, 例如 IBM、Java 等, 就会有不易集成的缺点, 原因在于 COM+ 通过 DCOM (Distributed COM) 通讯协议进行数据传输, 再加上必须支持 COM+ 规范接口的存取才能调用 COM+ 组件, 其他系统未必支持。

- .NET

.NET 在支持分布式系统上, 强调使用可扩展标记语言 (XML, eXtensible Markup Language) 作为传输的标准数据格式, 并且使用 Internet 上标准的 HTTP 与简单对象访问协议 (SOAP, Simple Object Access Protocol) 等通讯协议来传输 XML 数据 (如图 1.4 所示), 因此几乎所有可以连上 Internet 的系统都支持, 十分适合用在异质系统的集成上。

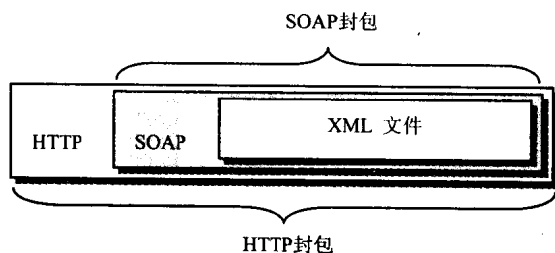


图1.4 XML数据传输

在组件规范上, 使用标准的 XML Web Service 规范将组件 Web 化, XML Web Service 可以看作是一种没有用户界面的网站, 用来提供客户端所需的业务逻辑服务, 客户端只需要支持 XML、HTTP、SOAP, 就可以和 XML Web Service 沟通, 不限任何操作系统平台、程序语言、硬件设备, 如图 1.5 所示。

4. 软件平台

软件必须在平台上开发与执行, 例如在 Windows 上使用 Visual Basic 开发窗口应用程序、在 IIS 上使用 ASP 开发 Web 应用程序或者在 UNIX 系统下用 C++ 开发系统程序, 其中的 Windows、IIS 与 UNIX 就是所谓的软件平台。由于不同软件平台的架构不同, 因此在不同平台上开发软件的方式 (例如系统调用方式) 就明显不同, 熟悉开发苹果机应用程序的人, 就不见得能够轻易转而开发 UNIX 上的应用程序, 也就是说程序员的程序开发经验中有绝大部分是与所使用的操作系统、程序语言以及类库有关, 然而这些领域上的技术或程序语法在不同的平台下往往大相径庭, 于是 Windows 平台上的程序员为了延续既有的 Windows 程序开发经验与资源, 只好继续开发 Windows 平台上的程序。

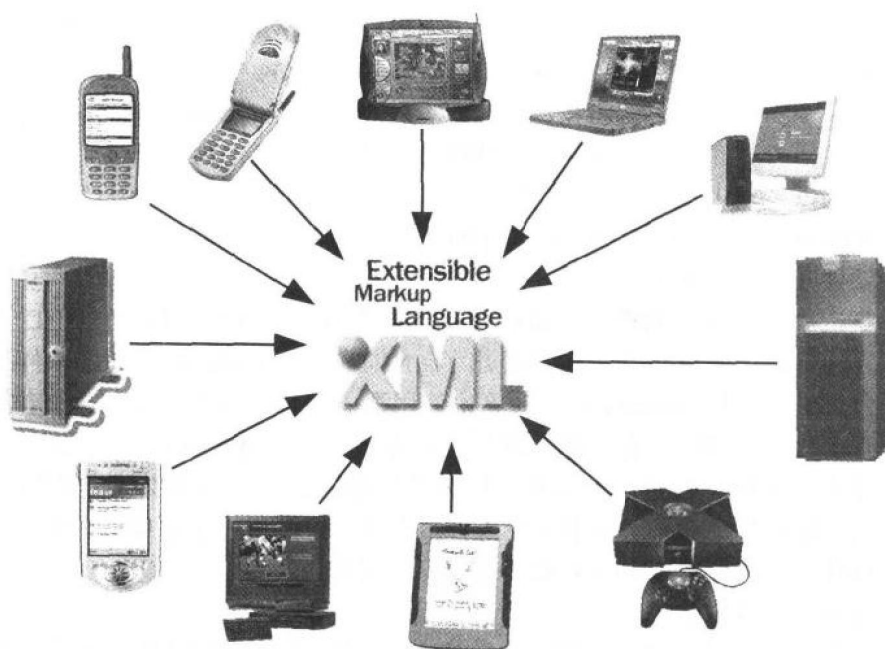


图1.5 XML Web Service示意图

这样的状况在以往单机、客户机/服务器式、多层次系统中还行得通，然而在 Internet 的时代，全世界所有类型的软件系统都悄悄连上了 Internet，我们所开发的系统不再只是企业内部封闭的系统，甚至不是使用同一种软件平台来开发的企业系统。例如现在各公司无不竭尽全力 e 化，建立公司的企业资源规划（ERP，Enterprise Resource Planning）系统，和以往企业中的物料需求计划（MRP，Material Requirement Planning）系统不同的是，传统 MRP 系统主要处理企业的某一单纯流程与数据，例如库存管理系统、薪资管理系统，而 ERP 系统着眼的是如何通过网络集成企业中所有的系统，甚至与其他公司进行必要的数据交换（或者说电子商务往来）。公司的决策者可以使用随身的手机或 PDA，通过 Internet 存取企业的 ERP 系统，实时取得各分公司的综合数据，以便下达决策，公司与企业伙伴之间也可以通过 Internet 进行合作与交易，将原本耗时的交易流程全部电子化，从下单采购、出货、货运、交货入库，全部通过 Internet 上的各个 ERP 系统集成完成。

软件平台不再局限于操作系统，现在的 Internet 就是一个软件平台，如何运用 Internet 上丰富的软件开发资源，使用适当的软件开发工具，在 Internet 上开发出适应各类型软硬件的软件系统，于是 .NET 应运而生。

1.3 Microsoft .NET

为了达到将不同平台与不同系统集成目的，微软提出 .NET（或者称为 Microsoft .NET）战略，那么什么是 .NET 呢？

- 未来生活的远景 (Vision)

大家或多或少都看过一些科幻片,例如星际大战,对于这些科幻片中出现的未来生活方式相信都印象深刻:“计算机,启动防御系统”之类的命令,或者手持一个类似 PDA 的装置,就可以对受伤的人进行诊断与医疗等,这些高科技的生活方式正是 .NET 最终的目标,如果你对 .NET 的未来生活有兴趣,可以到以下网址观看 e 时代生活影片:

http://www.microsoft.com/taiwan/products/net/net_mov.htm

- 软件开发与应用平台

要达成上述所有的理想,就必须有适当的软件来支持,因此 .NET 也是一个软件平台,在这个部分微软提出了 .NET Framework 这个基础软件架构与平台,在 .NET Framework 之上不论使用那一种程序语言或那一种操作系统,在开发软件上可以享有一致的软件开发方式(包含相同的类库),这对于软件设计人员来说无疑是一大福音,因为使用相同的开发方式甚至是相同的程序语法,就可以在不同平台下开发或移植软件,大大增加了程序员的生产力,稍后将针对 .NET Framework 做更进一步的说明。

- 超棒的用户体验

目前因特网虽然发达,但是绝大部分的应用还停留在网页的浏览与数据的查询,如果想使用因特网进行一些比较复杂的工作,就显得力不从心了。例如想上网购买基金,要先开一个浏览器连到提供基金价格的网页,查询目前哪支基金绩效最好,然后再开一个浏览器连到新闻网页,看看最新的财金新闻,了解目前市场状况,再开一个浏览器连到银行,查询目前的存款余额,如果余额足够,再回到基金网页下单购买...。这样繁复的步骤与大量窗口的切换,大大降低了用户在网络上进行交易的兴趣,更不用说这整个过程还必须有一台计算机在手边。

.NET 希望提供用户更棒的网络使用体验,让用户在任何地点(any place)、任何时间(any time)、使用手边的任何设备(any device)就可以进行网络上的相关工作并取得所需的信息。例如即使在火车上,也可以使用手机上网进行股票的买卖、机票的预定等,而这些信息可能来自于不同的公司、不同的系统,但是无所谓,运用 .NET 技术,可以将这些有用的信息集成起来,届时我们就不需要开启一大堆的浏览器,浏览一大堆无用的信息,然后在这些信息汪洋中寻找自己所需的信息,取而代之的是更聪明、更快速的操作方式。例如一个网页中包含了来自多个网站的必要信息,省去了在不同浏览器窗口中切换的不便,而这只是 .NET 提供的一小部分。接下来谈谈 .NET 架构软件开发平台,正式进入 .NET 的技术领域。

1.4 .NET 架构

图 1.6 是 .NET 架构的结构图,以及与 Visual Studio .NET 之间的关系: