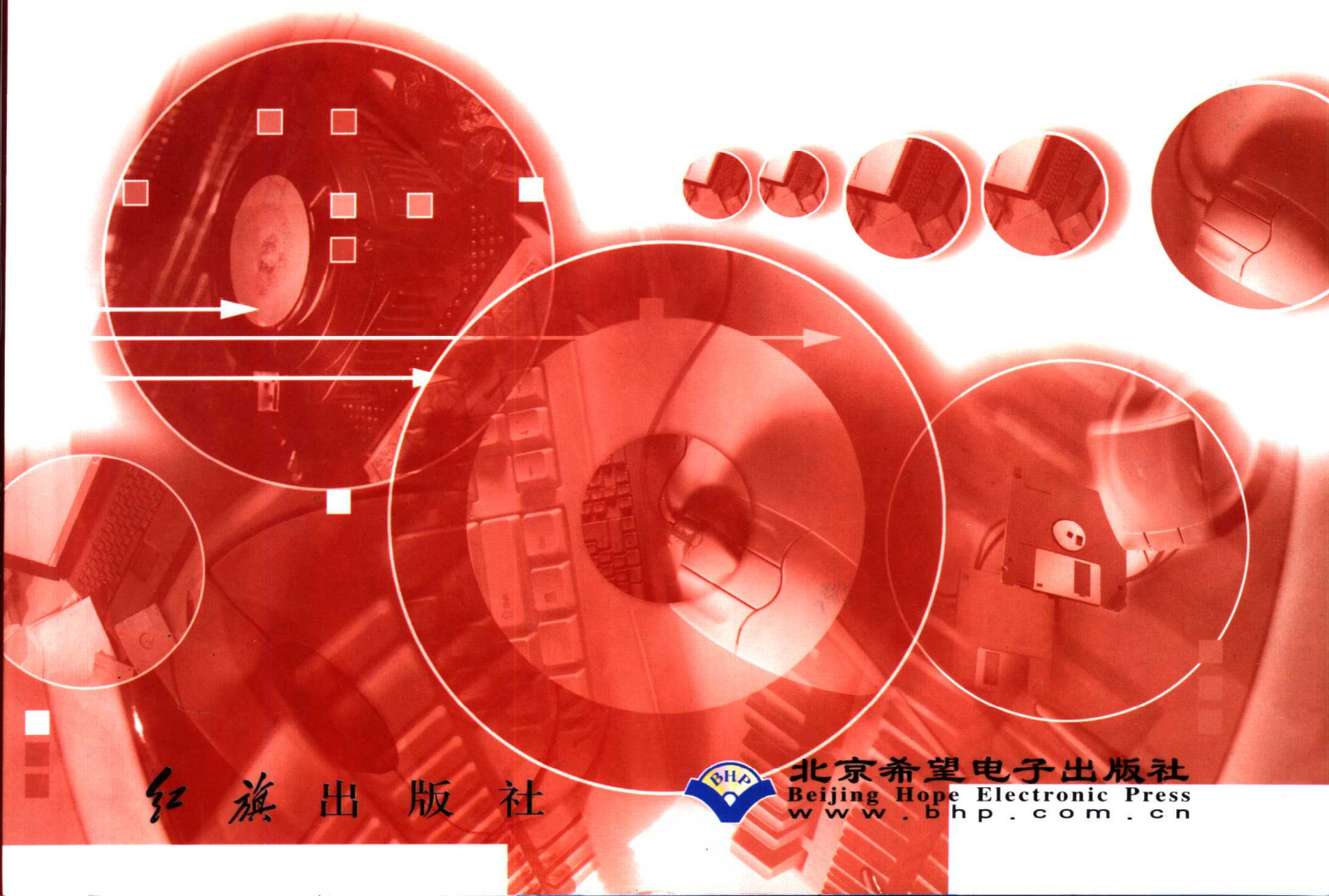
.NET 高级技术培训教材系列

使用 Visual C#.NET 开发 Windows 应用

北京希望电子出版社　总策划
苑　旭　董民辉　杨洪振　编　著

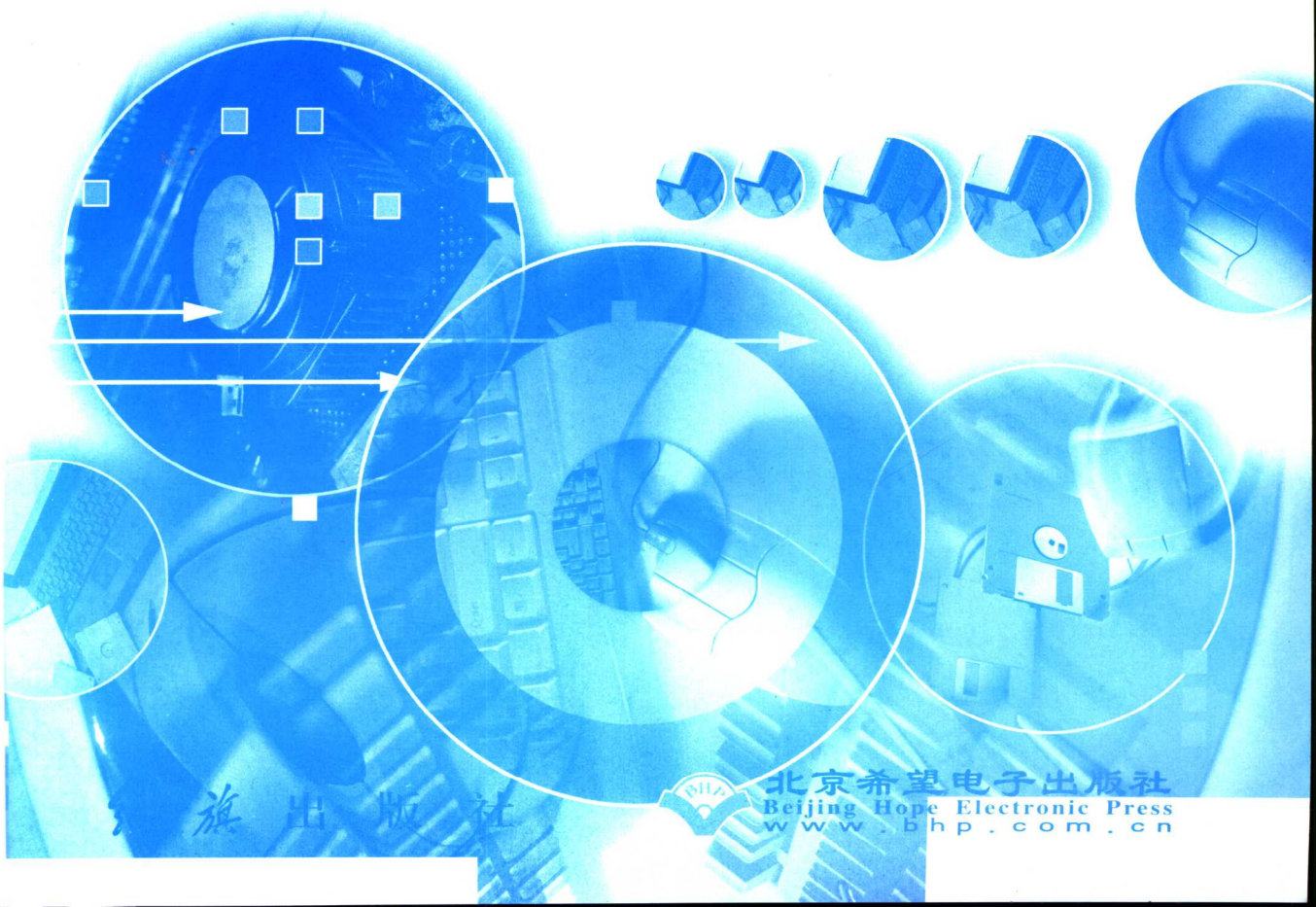


红旗出版社
北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

.NET 高级技术培训教材系列

使用 Visual C#.NET 开发 Windows 应用

北京希望电子出版社　总策划
苑　旭　董民辉　杨洪振　编　著



北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

图书在版编目（CIP）数据

使用 Visual C#.NET 开发 Windows 应用 / 苑旭，董民辉，杨洪振编著. —北京：红旗出版社，2005.2
ISBN 7-5051-1102-7

Ⅰ.使… Ⅱ.①苑…②董…③杨…Ⅲ.C 语言—窗口软件—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2004）第 120470 号

内 容 简 介

本书介绍利用 Visual C#.NET 开发 Windows 应用程序。全书内容主要讲述：.NET 介绍，Windows Forms 简介，准备 VS 开发环境，C#基础，控件基础，处理 XML，ADO.NET，提高程序的可访问性，异步编程，部署.NET 程序，Windows 应用实例等方面的内容，在每一章开始都简单介绍本章的主要内容，以及本章的学习目标，以便读者在学习过程中作为参照。同时在每章的开头均有重点介绍，每章结尾均包含模拟试题和分析。

本书适用于 C#.NET 专业编程人员，广大编程爱好者，以及高校相关专业师生，此外还可作为备考微软 MCAD/MCSD 考试的参考书，考试号 70-316。

需要本书或需要得到技术支持的读者，请与北京中关村 083 信箱（邮编 100080）发行部联系，电话：010-82702660，82702658，62978181 转 103 或 238，62978181（总机）传真：010-82702698，E-mail：tbd@bhp.com.cn。

系 列 名	.NET 高级技术培训教材系列
书 名	使用 Visual C#.NET 开发 Windows 应用
文本著作者	苑旭 董民辉 杨洪振
责 任 编 辑	范二朋 雷铎
出版、发行	红旗出版社 北京希望电子出版社
地 址	红旗出版社 北京市沙滩北街 2 号 北京希望电子出版社 北京海淀区上地 3 街 9 号金隅嘉华大厦 C 座 610
经 销	各地新华书店、软件连锁店
排 版	希望图书输出中心 孙 红
文本印刷者	北京双青印刷厂
开本 / 规格	787 毫米×1092 毫米 1/16 30.875 印张 720 千字
版次 / 印次	2005 年 3 月第 1 版 2005 年 3 月第 1 次印刷
印 数	0001~5000 册
书 号	ISBN 7-5051-1102-7
定 价	52.00 元

出版说明

.NET 开发战略是有史以来最伟大的一次开发革命，与以往的开发方式相比较，其更为严谨，也更为合理，当然，对程序员也同时提出了更高的要求：不再像以往那样仅仅需要学习一门开发语言就可以完成开发，我们需要更广泛的了解.NET 开发平台，了解我们作为.NET 程序员应该了解的一切。

为此，我们特组织优秀的微软认证讲师（MCT）和前沿程序员编写了本套微软认证高级技术培训教材。

全套教材共 6 本：

序号	书　名
1	使用 Visual Basic.NET 开发 Web 应用
2	使用 Visual Basic.NET 开发 Windows 应用
3	使用 VB.NET 开发 XML Web Services 和 Server Components
4	使用 Visual C#.NET 开发 Web 应用
5	使用 Visual C#.NET 开发 Windows 应用
6	使用 Visual C#开发 XML Web Services 和 Server Components

本套培训教材都由第一线的微软认证高级技术培训中心讲师（MCT）和走在技术最前沿的程序员编写，凝聚了多年教学和开发经验，教材的每章都有学习重点，整套教材对.NET 平台技术的阐述相互交织但又独立成书。

在必要章节附有实验，供学员练习。本套教材还包括大量模拟试题，创新性的利用模拟试题的试题分析和知识点解析以适应读者对所学知识的消化使用。同时也适用于 MCSD，MCAD 考试。本丛书对应目前.NET 开发平台最流行的两种开发语言 VB 或 C#。

本套教材既可供广大网络、数据库技术人员和爱好者学习、参考使用，也可以作为微软认证高级技术培训教材和自学教材。

编　者

编 者 序

对于大多数软件开发企业和 IT 应用部门来说，工作就意味着与时间赛跑：开发出最快的运行软件、用最快的时间进行开发……因此选择什么样的开发工具、语言环境，对于软件开发人员来说有着至关重要的意义。

作为微软.NET 核心组件之一，Visual Studio.NET 就是微软在其.NET 框架下为开发者提供的一个开发环境，或者说是一个开发工具集。Visual Studio.NET 简化了开发 Web 应用程序的过程。共享 HTML、XML 和样式表单编辑器使开发人员能很容易地利用任何 Visual Studio 语言，包括新的 C#开发 Web 应用程序。

XML 是下一代产品的关键组成因素。微软最浩大的.NET 战略就是依存于 XML 的，就像它以前的产品依赖于图形界面一样。微软将把 XML 变成整个业界的标准，而它自己的.NET 实现会成为最好的 XML 实现案例，就像过去 Windows 是图形用户界面最好的实现一样。

随着 Visual Studio.NET 这个微软历史上最重要的集成开发环境的正式发表。.NET 的开发也不再停留在概念阶段，而是正式进入工业开发领域。通过这个开发环境的广泛使用，相信，不久的将来，XML、.NET 战略所确定的目标将会走进我们的生活。

本书全面介绍 Visual Studio.NET 和 Visual C#.NET、XML、ADO.NET 等.NET 框架内需要学习的知识，同时涉及到的全新的相关 MCSD/MCAD 考试内容。

通过本书的学习，可以让读者对 Visual Studio.NET 有一个系统的、全面地认识；同时，让读者对微软公司的 MCSD/MCAD 考试（课程号：70-316）内容有全面了解。全书注重理论联系实际，采用图文并茂的形式，帮助读者理解，每一张截图都是作者从实际开发环境中精心准备得来；每章中，都精心设计了实践操作部分，让读者不仅仅可以学到理论知识，而且能在书中的指导下，进行实践操作。本书是系统全面学习 Visual Studio.NET 的开发人员，技术爱好者的很好的辅助教材，其典型模拟试题的分析更是希望参加微软 Visual Studio.NET 开发工程师（MCSD/MCAD）认证考试（课程号：70-316）的很好的参考资料。

书中我们介绍了一些使用经验和心得，难免有不当之处，或者还有更好的方法，欢迎赐教。如果有需要交流的地方，请发 email 联系：yanghongzhen@126.com，欢迎与您的真诚交流。

一件作品的完成，是众人智慧与努力的结晶，在此特别感谢北京希望电子出版社。感谢编辑栾大成先生的努力工作，他的幽默风趣的言谈，踏实的工作态度让我记忆深刻。没有他用专业的眼光和细节的关注，这一系列的出版不能如此顺利。同时感谢在本书的编写过程中曾给予我帮助的朋友们：我的挚友刘春田、李明清以及身边的众多给予帮助的朋友，你们带给了我启发和欢笑，愿你们可以完成心中所愿。 给予我心灵上支持的杨蓉：带给你祝福，坚持下去，你的愿望一定会实现。 谢谢你们的大力支持。

MCT 杨洪振

作者简介

杨洪振

资深 MCT（Microsoft Certified Trainer），MCSE，MCDBA MLC 讲师，曾长时间从事系统技术支持、系统集成项目以及讲授相关产品课程等工作，对于 Microsoft 系列产品有深入研究，在基于微软平台的网络系统，服务器应用以及网络安全等方面具有丰富经验，包括 Windows 2000、Exchange Server，ISA Server，SharePoint Portal Server，BizTalk Server，Small Business Server 并是国内第一批使用和研究 Windows Servers 2003，Exchange Server 2003 的技术人员。

曾主持设计了多个网络系统方案集成和开发，如中国联通河北分公司网络基础架构，办公系统集成，邮件提醒平台，高可靠性群集方案，北京佳诚无限科技有限公司 Exchange 邮件系统平台，Portal 平台的建设，同时为其他众多知名公司提供信息化咨询。其教学经验丰富：曾为多个知名企业进行 IT 培训，如建设银行、河北省委办公厅、中国联通、中国网通、河北电力公司等企业进行了多种 IT 系统方面的培训。其幽默风趣的授课风格，丰富的理论知识和经验，得到了大家的认可。

董民辉

浙江海洋学院图书馆技术主管。长期负责图书馆数据库开发和更新维护，最早使用 Visual Studio.NET 开发图书管理系统并成功运行于实际环境的程序员。对.NET 框架下的数据库技术有非常深刻的理解，利用.NET 成功提高了图书馆的管理效率和运营成本，受到全校师生好评。

苑旭

某 IT 公司软件开发部经理。长期从事于 Microsoft 开发产品的研究工作，精通于微软的各种开发平台，是国内最早接触研究 Visual Studio.NET 的开发人员之一。对 Visual C#，Visual Basic.NET 有深入研究，同时，苑先生在开发实践中的经验十分丰富，组织策划了多个大型项目的开发，如中国平安保险手机投保项目，河北省委移动办公项目。从大型企业的基于.NET 平台的 Web Service 应用，到基于智能终端，手机应用软件的开发都曾有过辉煌的成绩。

目 录

第 1 章　.NET 介绍

- 什么是.NET
- .NET 的意义
- .NET Framework 架构
- .NET 与其他技术的联系
- .NET SDK 提供的免费开发工具
- 走向标准化的.NET

本书主要讨论 C#，但是不能孤立地讨论 C#，必须和.NET Framework 一起考虑。C#代码总是在.NET Framework 中运行，所以 C#反映了.NET Framework 的基础方法。例如 C#的单继承，以值和引用传递的类型系统等都与.NET Framework 一致；另一方面，C#的许多特性取决于.NET framework，如 C#中的数据类型等。

由于这种依赖性，在讲解 C#之前首先介绍一下.NET Framework，这对以后的学习非常有益。

本章包含 70-316 考试中关于.NET 基础的知识。这些知识包括.NET Framework 的组成、CLR、IL、JIT 等。

学习目标：

- 了解.NET 的定义
- 了解.NET 对于各种人员的意义
- 掌握.NET Framework 的组成
- 了解.NET Framework 与其他技术的关系
- 了解 C#的标准化进程

1.1　什么是.NET

随着网络经济的到来，微软公司希望帮助用户能够在任何时候、任何地方、利用任何工具都可以获得网络上的信息，并享受网络通信所带来的快乐。.NET 战略就是为着实现这样的目标而设立的。

微软公开宣布，今后将着重于网络服务和网络资源共享的开发工作，并称，将会为公众提供更加丰富、有用的网络资源与服务。

微软新一代平台的正式名称叫做“新一代 Windows 服务”（NGWS），现在微软已经给这个平台注册了正式的商标——MicroSoft.NET。在.NET 环境中，微软不仅仅是平台和产品的开发者，并且还将作为架构服务提供商、应用程序提供商，开展全方位的 Internet 服务。在谈及这个平台中使用的新技术，微软透露，它将在.NET 环境中提供更多新产品和一揽子的全套服务。

Microsoft .NET 平台的基本思想是：

侧重点从连接到互联网的单一网站或设备上，转移到计算机、设备和服务群组上，使其通力合作，提供更广泛更丰富的解决方案。用户将能够控制信息的传送方式、时间和内容。计算机、设备和服务将能够相辅相成，从而提供丰富的服务，而不是像孤岛那样，由用户提供惟一的集成。企业可以提供一种方式，允许用户将它们的产品和服务无缝地嵌入自己的电子构架中。这种思路将扩展二十世纪八十年代首先由 PC 赋予的个人权限。

Microsoft .NET 将开创互联网的新局面，基于 HTML 的显示信息将通过可编程的基于 XML 的信息得到增强。XML 是经“万维网联盟”定义的受到广泛支持的行业标准，Web 浏览器标准也是由该组织创建的。微软公司为开发它投入了大量精力，但它并不是 MicroSoft 的专有技术。XML 提供了一种从数据的演示视图分离出实际数据的方式。这是新一代互联网的关键，提供了开启信息的方式，以便对信息进行组织、编程和编辑；可以更有效地将数据分布到不同的数字设备；允许各站点进行合作，提供一组可以相互作用的“Web 服务”。

Microsoft .NET 是构建、执行与体验下一代分布式应用程序的平台。它的内容涵括了客户端、服务器与开发者工具，包括：

- .NET Framework 程序设计模型，使开发人员可以建置 Web 应用程序、智能型客户端应用程序以及延伸式标记语言（XML，Extensible Markup Language）Web Service 应用程序，它使用 SOAP、延伸式标记语言（XML，Extensible Markup Language）与 HTTP 等标准通讯协议将其功能以程序方式公开于网络上。
- 开发者工具，例如 Visual Studio® .NET，它在使用 .NET Framework 做程序设计时，提供快速的应用程序整合开发环境。
- 完整的服务器套件包含 Windows® 2000、SQL Server™与 BizTalk™ Server ，可以整合、执行、作业并管理 XML Web Service 与应用程序。
- 客户端软件，例如 Windows XP、Windows CE 与 Microsoft Office XP，可以让开发人员在针对全系列的装置与现有产品进行开发作业时，融合顺畅便捷的使用者体验。

1.2 Microsoft.NET 平台的重要意义

我们来看一下 MicroSoft .NET 对开发人员、IT 专业人员、以及企业应用的巨大意义。

1.2.1 对于开发人员的意义

MicroSoft .NET 的策略是将互联网本身作为构建新一代操作系统的基础，对互联网和操作系统的设计思想进行合理延伸。这样，开发人员必将创建出摆脱设备硬件束缚的应用程序，以便轻松实现互联网连接。MicroSoft .NET 无疑是当今计算机技术通向计算时代的一个非常重要的里程碑。

.NET 的核心组件有：

- 一组用于创建互联网操作系统的构建块，其中包括 Passport.NET（用于用户认证）以及用于文件存储的服务、用户首选项管理、日历管理以及众多的其他任务。
- 构建和管理新一代服务的基本结构和工具，包括 Visual Studio.NET、.NET 企业服

务器、.NET 框架和 Windows .NET。

- 能够启用新型智能互联网设备的.NET 设备软件。
- .NET 用户体验。

.NET 对最终用户来说非常重要，因为计算机的功能将会得到大幅度提升，同时计算机操作也会变得非常简单。特别地，用户将完全摆脱人为的硬件束缚：用户可以自由冲浪于互联网的多维时空，而不是束缚在便携式电脑的方寸空间——可通过任何桌面系统、任何便携式电脑、任何移动电话或 PDA 进行访问，并可对其进行跨应用程序的集成。

.NET 可使用户轻松进行互联网连接，并轻松完成那些在当今看来十分费时而且费力的事务，它们往往要求用户进行数据重输入并需运行几个小时才能完成。通过将多项安全数据流合并到单一的用户界面（或者甚至是可编程决策引擎），.NET 架构将用户从充斥于当今 Web 的数据竖井的束缚中解脱出来。用户可以自由访问、自由查看、自由使用他们的数据。

.NET 对开发人员来说也十分重要，因为它不但会改变开发人员的开发应用程序的方式，而且使得开发人员能创建出全新的各种应用程序。新型开发范例的核心是 Web 服务这个概念的引入。Web 服务是一种通过简单对象访问协议（SOAP），在互联网上展露其功能性的、极为公开的服务。SOAP 是一种基于可扩展标记语言（XML）制定的协议。

在过去，开发人员通过集成本地系统服务来构建应用程序。在这种模型下，开发人员可以访问丰富的开发资源并能严格控制应用程序的行为。

如今，开发人员已在很大程度上挣脱了这种模型的束缚，致力于构建具有复杂结构的 n 层化系统，这种系统能将网络上众多的应用程序一并进行集成，大大提升了应用程序的价值。这样，开发人员便可把精力集中在充分挖掘软件独特的商业价值，而不是构建基本结构上。可喜的局面将应运而生：软件投放市场的时间大大缩短、开发人员的编程效率明显提高，最终把质量上乘的软件呈现给用户。

我们正在进入一个崭新的计算时代——一个由互联网（尤其是 Internet 核心技术 XML）实现的时代。利用 XML，能够创建出可供任何人从任何地方使用的、功能非常强大的应用程序。它极大地拓展了应用程序的功能，并实现了软件的动态提供。在这种情况下，软件已不完全指那些从光盘进行安装的程序，而是演变成了一种服务——类似于 ID 调用程序或按收看次数进行收费的电视——人们可通过通信媒体订购的服务。

n 层计算技术具有能够大幅度提高生产力、紧密耦合的特点，而 Web 概念具有面向消息、松散耦合的特点，将二者有机地糅合在一起，实现了上述构想。将这种计算风格称为 Web 服务，它的出现标志着人类已经迈入应用程序开发技术的新纪元。Web 服务是一种应用程序，它可以通过编程并使用标准的 Internet 协议，像超文本传输协议（HTTP）和 XML，将功能展示在互联网和企业内部网上。还可将 Web 服务视作 Web 上的组件编程。

从理论上讲，开发人员可通过调用 Web 应用编程接口（API），将 Web 服务集成到应用程序中。其调用方法与调用本地服务类似，不同的是 Web API 调用可通过互联网发送给位于远程系统中的某一服务。例如，Microsoft Passport（Passport）服务使得开发人员能够对应用程序进行认证。通过对 Passport 服务编程，开发人员可以充分利用 Passport 的基本结构，通过运行 Passport 来维护用户数据库，以确保其正常运行、定期备份等等。

.NET 正是根据这种 Web 服务原则而创建的，微软目前正着手提供这个基本结构，以便通过.NET 平台的每一部分来实现这种新型的 Web 服务。而 Visual Studio.NET、.NET 框架、Windows.NET 和.NET 企业服务器，正是为进行基于 Web 服务模型的应用程序开发而度身定做的新一代开发工具和基本结构。.NET 构建块服务、新增的.NET 设备支持以及即将到来的.NET 用户体验，将为人们彻底攻克这一难题划上一个圆满的句号，使人们能够充分利用 Web 服务模型，如愿以偿地开发出新一代应用程序。

1.2.2 .NET 对 IT 专业人员的重要意义

目前，IT 专业人员能够利用与构建.NET 平台相同的技术。

.NET Enterprise Servers 和 Windows 2000 操作系统，为创建具有高度可管理性的、能迅速投入市场的应用程序提供了坚实基础。它们利用的是可扩展标记语言（XML），因此随着 Web 体系结构的革新，在此平台上创建的程序依然很有价值。

.NET 平台的核心是，采用有效的、分门别类的方式来构建应用程序，达到其前所未有的规模。该平台上的 Web 服务模型指的是：企业应用程序的中心业务要素通常由本地管理，而支持它们的服务（如用户认证、文件存储、用户首选项管理、日历、邮件等等）却无须本地管理，可以被无缝订购。为了存储用户文件和邮件，IT 专业人员往往在服务器上安装新的独立磁盘冗余阵列（RAID 阵列），而有了.NET，他们在这一方面将会花费较少的精力，而更多地致力于怎样为公司增加效益。

该 Web 服务模型还将动态配置新软件的发布和更新。用户将以极其紧密的连接方式工作，因此更易于管理。而简化的管理又可使 IT 专业人员更能适应变幻莫测的业务需求。

开发应用程序的.NET Web 服务模型将为企业应用程序的创建开辟一条新路。通过企业内外多种服务的联合，很容易把企业内部数据和客户及合作伙伴的相关数据结合在一起，大大简化了应用程序的创建过程。这就为最终用户发掘了空前的功能涵盖性。例如，利用某公司的雇员福利程序，可以从其 HR 数据库订购信息，通过 Web 订购福利管理公司的服务、订购工资管理公司的服务。终端用户可以在简单、直观的界面下操作，而这个界面可以显示他们的累积休假时间、个人所得福利以及上次工资额。

1.2.3 .NET 对企业的重要意义

Microsoft .NET 平台将从根本上改善计算机和用户之间进行交互的方式，最大限度地发挥电子商务中计算技术的重要作用。首先，让我们来分析一下当前商务计算世界的现状：

人与计算机进行交互的手段极为有限——通常使用键盘和鼠标进行输入，使用监视器监控输出。

用户信息基本上是本地信息，如果从另一台机器进行登录，则无法获取用户的个人首选项设置、数据及应用程序。

用户必须亲自处理信息，而通过设置智能选项代表用户自动进行操作，则无异于是纸上谈兵。

同一用户存放于不同应用程序和站点的数据，很难（或根本不可能）进行自动合并和关联，用户无法统一进行查看。

想在家里或在路上工作的用户，不能方便地访问办公室电脑中的应用程序和数据。这无疑成为一道阻止人们获得更高工作效率的鸿沟。

不能使用其他设备访问专为特定设备设计的数据（这些设备包括 PC、寻呼机、移动电话以及 PDA 等）；最多可以定期进行同步。

.NET 将保证完全消除当今计算技术中的所有缺陷。.NET 定能实现确保用户从任何地点、任何设备都可访问其个人数据和应用程序的宏伟蓝图。除此之外，.NET 技术还可实现多个应用程序在逻辑上的松散耦合链接和紧密耦合链接。

用户可以通过手写、语音和图像技术与其个人数据进行交互。这些数据将安全地存放在互联网上，用户通过办公室（或家庭）PC，还可以通过移动电话或寻呼机、PDA、甚至是新发明的寻呼机——移动电话——PDA——PC 联合设备访问这些数据。应用程序可进行灵活的功能调整，以适应用户所用设备的功能状况。应用程序可根据用户预定义的选项集和指令集，完全代替用户自动执行相应的操作。

上述功能将协同作用，以便大幅度地提高用户使用计算技术的生产效率。根据设计，.NET 使得用户无需在如何与计算机进行交互上劳神，从而全身心地投入到使计算机自动执行任务、实现最终目标的工作中。通过使用 XML 行业标准，可将用户数据进行跨站点和应用程序的链接，从而轻松实现当前很难实现的操作。比如：对用户在数家不同银行、信用卡公司以及计费代理商那里的数据进行集中处理；这样，用户便可依据处理后的数据支付帐单，将费用明细报告归档。

.NET 把雇员、客户和商务应用程序整和成一个协调的、能进行智能交互的整体，而各公司无疑将是这场效率和生产力革命的最大受益者。简言之，.NET 承诺为人类创造一个消除任何沟鸿的商务世界。

1.3 .NET Framework 架构

.NET Framework 是.NET 平台的程序设计模型，用以建构、部署与执行使用 SOAP、XML 与 HTTP 等标准通讯协议将功能以程序方式公开于网络上的 Web 应用程序、智能型客户端应用程序（Smart Client Applications），和可延伸标记语言（XML）Web Service 应用程序。.NET Framework 提供高效能、标准化的环境，可以将现有投资与下一代的应用程序与服务加以整合，更具备应付部署与操作因特网级应用程序各项问题的灵敏度。.NET Framework 是由两个主要部分所组成（图 1-1.NETFramework）：Common Language Runtime 以及一组统一的类别库，后者包含用于 Web 应用程序与 XML Web Service 的 Microsoft ASP.NET、用于智能型客户端应用程序的 Windows Form，以及用于弹性数据存取的 Microsoft ADO.NET。

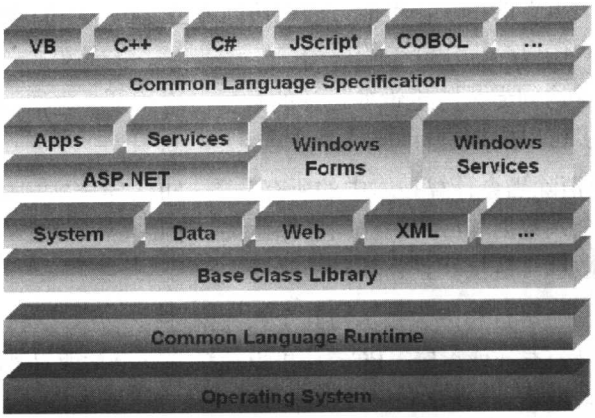


图 1-1 .NET framework

1.3.1 Common Language Runtime

公共语言运行时（Common Language Runtime）调入并运行用任何.NET 编程语言所写的代码。以 CLR 为目标的代码被称为受控（managed）代码，受控代码只是意味着在内部可执行代码与运行自身间存在已定义好的合作契约。对于像生成对象、调用方法等这样的任务，被委托给了公共语言运行时，这使得在公共语言运行时能为可执行代码增加额外的服务。

CLR 以交叉语言集成、自描述组件、简单配置和版本化及集成安全服务为特点。下面将快速查看一下每一特点。

公共语言运行时使用一种新的能表达大部分现代编程语言语义的通用类型系统，通用类型系统定义了一套标准类型及生成新标准的规则。公共语言运行时知道怎样生成、执行这些类型。编译器和解释器使用公共语言运行时服务定义类型、管理对象、进行方法调用，而不是使用工具或特定于语言的方法。

类型系统（CTS）的主要设计目的是使多种语言能深度集成。用一种语言所写的代码能继承用另一种语言所写的类的实现，用一种语言所写的代码抛出的异常能被用另一种语言写的代码捕获，像调试和剖析之类的操作会在完全封闭下工作，而不用考虑代码所用的语言。这就意味着编写可重用类库的开发者，不再需要为每一种编程语言或编译器生成一个版本，并且使用类库的开发者不再受到为他们使用的编程语言开发的库的限制。

自描述组件——现在 Microsoft .NET 框架上已成为可能——简化了开发和配置，并提高了系统的可靠性。许多由公共语言运行时提供的服务是由元数据及用于补充可执行代码的信息所驱动。因为所有的信息都储存在一起，只有可执行的（代码）才被称为自描述组件。

自描述组件的一个主要优点是，使用它们并不需要其他文件。类的定义不需要单独的头文件；通过检查元数据对类的定义可以从组件自身获得。跨语言或过程边界访问组件并不需要各自的 IDL 文件、类型文件或 proxy/stubs；所必需的信息已存在于元数据之中。为识别开发者请示的服务属性，并不需要展开各自的配置信息。 最主要的是，由于元数据是在编译过程中由源代码生成，并与可执行代码储存在一起，它将永远和可执行部分同步。

除了改善对单个组件的配置，Microsoft .NET 框架定义了一个应用程序配置模板，以解

决定制应用程序安装和 DLL 版本化（通常被称为“DLL Hell”）这一复杂过程的问题，公共语言运行时提供了支持这个模板的服务。

Microsoft .NET 框架 引入了装配件的概念。一个装配件是一组资源和类型，并包括有关这些资源和类型的元数据，也就是被作为一个单元配置的。元数据被称为装配件的清单，它包含像类型和资源表之类能被装配件外看得见的信息，这个清单也包括有关从属关系之类的信息，例如装配件建立时的版本号。开发人员可以指定版本策略，以指示公共语言运行时是否装入系统上已安装的依赖于装配件的最新版本，装入一指定版本，或在编译时使用的版本。

某软件组件的多个拷贝总可以存在于同样的操作系统上，然而，通常说来，只有其中的一个拷贝能被操作系统注册、调入内存、执行。对系统来说，定位和调入内存的策略是全局性的。.NET Framework common language runtime 增加了所必须的体系架构以支持管理组件定位和调入的每个应用程序策略，这通常被称为并行配置。

装配件可以被一个应用程序私有，或被多个应用程序共享。一个装配件的多个版本可以同时配置在同一台机器上。应用程序配置信息定义了到何处去查找装配件，这样 CLR 就能为同时运行的两个不同的应用程序装入同一装配件的不同版本。这就消除了由组件版本的不兼容性引起的问题，提高了系统整体的稳定性。如果必要，管理员可以为配置时刻的装配件增加配置信息，例如一个不同的版本策略，但是编译时提供的原始信息永远不会丢失。

因为装配件是自描述的，所以并不需要在系统上进行显式注册。应用程序的配置简单到只需将文件拷贝到目录中既可（如果为了使应用程序能够运行，必须安装未经组织过的组件的话，情况会稍微复杂一点）。配置信息保存在可被任何文本编辑器编辑的 XML 文件中。

最后，公共语言运行时也提供完整的、普遍深入的安全服务，以确保未经授权的用户不能访问机器上的资源，并且代码不会执行未经允许的动作。这就提高了系统整体的安全性可靠性。 由于公共语言运行时用于装入代码、生成对象、执行方法调用，所以当受控代码装入内存、执行时，公共语言运行时能进行安全检查，强化安全策略。

Microsoft .NET 框架不仅规定代码访问安全，还规定基于角色的安全。通过代码访问安全机制，开发人员能为应用程序指定完成工作所必需的权限。例如，代码或许需要写文件或访问环境变量的权力。这类信息和有关代码标志的信息一起存储在配置级上的。当代码装入内存及执行方法调用时，公共语言运行时验证是否能给予代码所要求的权限。如果不能，将记录一条安全冲突信息。给予权限的策略，这被称为信任策略，是由系统管理员建立的，并且是建立在关于代码的证据基础之上，比如：代码是谁发布的，是从什么地方获得的，以及在装配件中找到的代码标志和它要求的权限。开发人员可以指定他们显然不需要的权限，以防止其他人恶意使用他们的代码。如果所需要的权限依赖直到运行时刻才会知道的信息，那么就可写入纲领性的安全检查。

除了代码访问安全，公共语言运行时还支持基于角色的安全。基于角色的安全建立同代码访问安全一样的权限模板，只是这些权限是建立在用户的身份之上，而不是建立在代码的标志之上。角色表明了用户所属的类，并且可以在开发和配置阶段定义。给予权限的策略被分配到每个预定义的角色。在运行时刻，用户的身份被确定，代码将代表这个身份运行。公共语言运行时决定用户是哪个角色的成员，然后给予基于这个角色的权限。

1.3.2 Base Class Library

.NET Framework 类库提供了几乎所有应用程序都需要的公共代码。与在 Windows 和它的 SDK 中发送的代码库一样，类库使得开发者能将精力集中于编写他们的应用程序所独有的代码，而不必一再重复编写类似读写文件这样经常使用的功能的代码。类库还解决了当前的 Windows 代码库中存在的一个问题：当微软向 Windows 绑缚新功能时，API 和 SDK 之间就会出现混乱。（各种“版本”中的 Java 库试图解决非 Windows OS 上类似的、甚或更严重的问题，即：来自各种不同的厂商，用于诸如文件输入/输出和消息收发等基本功能的使人困惑的 API。）

所有的可控制代码都组织在称为类的（这就是类库的来源）逻辑组中，这些类按照称作域名空间的分级制度排列。

.NET Framework 类库在域名空间“System”之下。为了方便查找开发者所需要的类，该域名空间按照功能区的分级制度进行排列。

例如，System 中有 System. Data，它包含用于 ADO .NET 数据访问 API 的类。System.Data 本身包括一组称作 System.Data.Oledb 的类，它用来支持访问数据库以及其他拥有 Windows OLE DB .NET 数据供应器的数据源。System .Data 还包括 System .Data .SQLclient，它支持对 SQL Server 的访问。

一些 System 类提供的功能以前可以通过 Windows API 或者特定语言的功能获得，但其他类则是全新的，例如 System .Reflection，它使可控制代码能够获得来自汇编的信息，例如它的元数据。

开发者可以通过扩展类库中的类来定义自己的类，还可以定义全局惟一的专用域名空间，以使他们的类不会和微软或其他任何组织的类相冲突。

类库主要的好处是它们将核心 Win32 API 的最常用的功能和外挂 SDK 的功能封装到了一个统一的包中。采用清晰而有条理的方式对类库进行了分组和描述，这样开发者能更容易地找到他们的应用程序所需的大多数功能。

相反，在过去几年中，新功能要么被“绑缚”到 Win32 API 上，要由通过独立的 API（例如用于图形的 Directx，或者用于 XML 和 SOAP 的不同的 SDK）来提供。对它们惟一能做的逻辑分组就是按照字母顺序进行排序。结果，使用 Win32 API 和各种 SDK 经常使人晕头转向，而开发者必须判断几个类似的 API 中哪一个最适合他们特定的要求。

.NET 开发平台提供的 API 被组织安排到了一组带有逻辑名的分级域名空间中。这和 Win32 API 形成了尖锐对比，Win32 API 只是一个简单的功能名的长列表，顶多可以按字母排序。有了分级制度以后开发者就可以更加快捷地定位所需的功能，而且添加新的 API 也不会与已有的发生冲突。

类建立在基础类（底层）之上，基础类具有下述能力：文本处理（System.Text）网络访问（System .NET），以及存储列表和其他数据集（System .Collections）等。

基础类之上是更复杂的类，例如数据访问（System .Data），它包括 ADO.NET 和 XML 处理（System .XML）。

顶层是用户接口库。Windows 表单和 Drawing 库（分别是 System.Win 表单和

System .Drawing）提供了封装后的 Windows 用户接口，包括 GDI+和 DirectX。

开发者通过将他们自己的应用程序源代码和来自.NET 类库的代码相结合，创建了可控制代码。这些类库可能包括.NET 开发平台包含的类库（例如 ADO.NET 和 Windows 表单），以及来自第三方的类库。

接下来.NET 编译器将此代码从人可读的格式翻译成 Microsoft Intermediate Language（MSIL）。不管采用何种语言，所有的.NET 开发平台编译器均生成 MSIL 结果。

除了 MSIL，.NET 编译器还产生元数据，它描述了弥补代码的组件。Common Language Runtime（CLR）（通用语言运行环境）使用这种元数据来增强安全性，并确保获得它所需的任何组件的正确版本（减少组件冲突，或者“DLL Hell”）。

Visual Studio .NET 和其他工具自动将 MSIL 代码封装到 CLR 中使用的汇编中。几个 MSIL 文件可以被组合成一个单一的汇编。

1.3.3 ADO.NET

为了提供对数据的访问，服务框架包括 ActiveX Data Objects.NET（ADO.NET）类库。如同名字所暗示地那样，ADO.NET 由 ADO 发展而来。ADO.NET 被设计为基于网络的可扩展的应用程序和服务提供数据访问服务。ADO.NET 为连接的指针风格的数据访问，同时也为更适合于把数据返回到客户端应用程序的无连接的数据模板提供高性能的 APIs 流。

ADO.NET 定义了那些链接数据仓库、 对数据仓库发送命令及从中获取结果的类。这些类由受控数据提供者（managed data provider）实现。ADO.NET 中链接和命令对象看上去和 ADO 中的是一样的，并且一个名为 DataReader 的新类提供了通过高性能 API 流获取结果的能力。DataReader 在功能上同前向、只读的 ADO 记录集(Recordset)是等同的，但是 DataReader 被设计用来最小化内存中生成的对象的数量，以提高性能，避免垃圾积累。在.NET Framework 中包含了针对 Microsoft SQL Server?的受控数据提供者以及可通过 OLE DB 访问的任何数据仓库。

ADO.NET 的一个主要创新是引入了数据集（Dataset）。一个数据集是内存中提供数据关系图的高速缓冲区。数据集对数据源一无所知，它们可以由程序或通过从数据仓库中调入数据而被生成、填充，并且它使用相同的潜在的数据缓冲区。

受控数据提供者为数据仓库和数据集公开一名为 DataSetCommand 的接口对象。DataSetCommand 使用 ADO.NET 链接和命令以从数据仓库中填充数据集，并把在数据集中发生的变化解析到数据仓库中。

所有的数据都可被看作 XML，所以开发人员可以为任何数据使用转换和确认服务。ADO.NET 定义了一个消费 DataNavigator、生成一个新的 XmlReader 的通用转换体系。.NET Framework 提供了一个支持 W3C XSL Transformations（XSLT）细则的特殊转换组件。ADO.NET 同时提供了一使用 XML 架构确认 XmlReader 的确认引擎。ADO.NET 支持通过 DTDs， XSD 或 XDR 定义的架构。

1.3.4 ASP.NET

ASP.NET 对 Active Server Page（ASP）进行了很大的改变，不仅使开发者更易于创建动