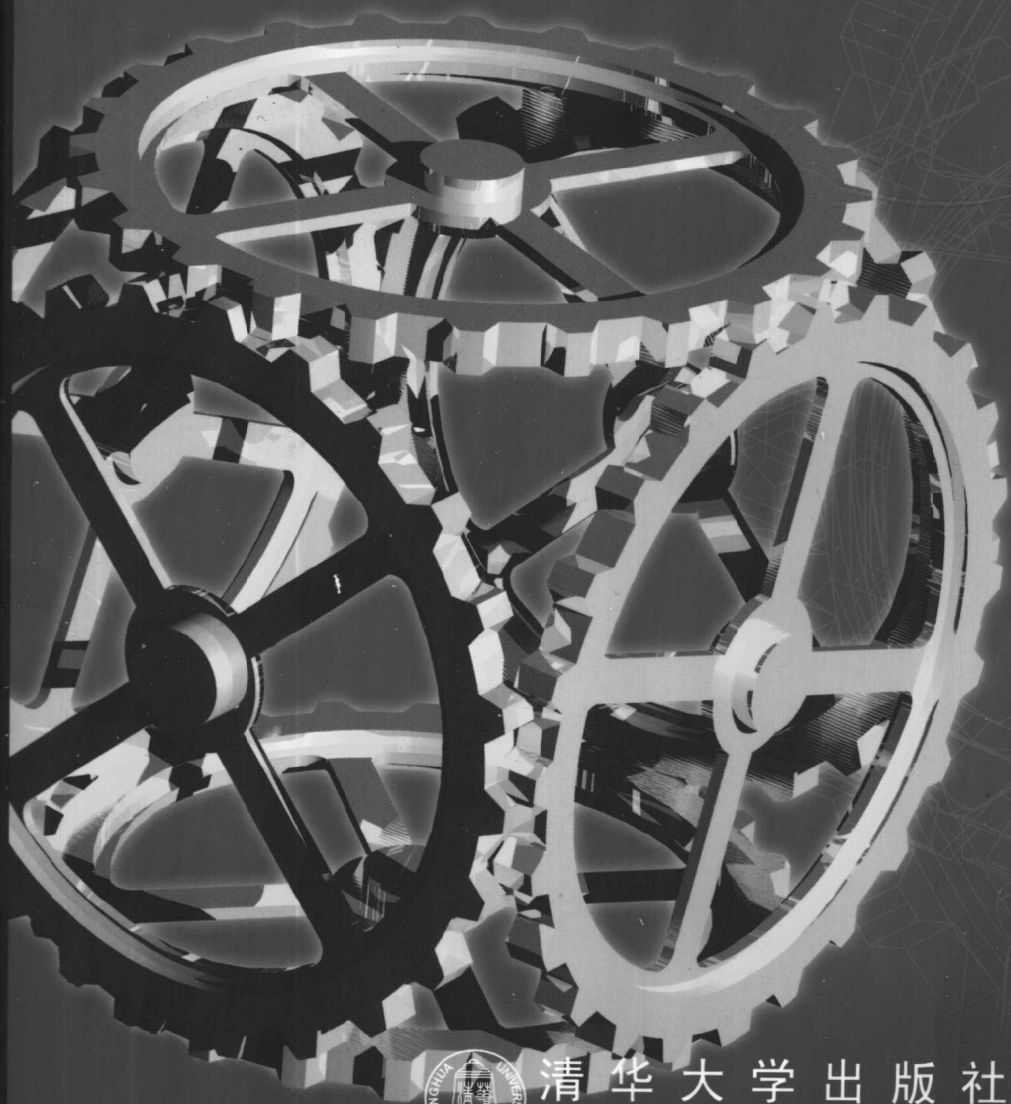
构建 Web 服务和.NET 应用程序



在分布式应用程序和 Web 服务设计中有效地使用 XML

使用 Visual Studio .NET 创建分布式的且基于 Web 的应用程序和服务

开发、调试和部署基于.NET Framework 的 Web 服务

为使用 ADO.NET 和 SQL Server 工具进行最佳的.NET 开发并提供专家级的建议

Lonnie Wall
Andrew Lader 著
康 博 译

清华大学出版社
http://www.tup.tsinghua.edu.cn

Mc Graw Hill

构建 Web 服务和.NET 应用程序

Lonnie Wall
Andrew Lader 著

康　博　译

清华大学出版社

(京) 新登字 158 号
北京市版权局著作权合同登记号：01-2002-4480

内 容 简 介

为了适应对分布式开发环境的需求，Microsoft 全力推出了功能强大的.NET 平台，它代表一种全新的基于 Web 服务模型的应用程序设计和部署方法，为我们开发 Web 服务和.NET 应用程序提供了最强有力的支持。

本书首先回顾了从独立应用程序到分布式.NET 应用程序的演变过程，然后深入介绍了 Web 服务及其相关内容、使用 XML 和 SOAP 设计并创建大型的分布式应用程序、ADO.NET，SQL Server 2000，DataSet 和.NET Framework 等知识、以及 ASP.NET 和 Windows Forms 应用程序的开发方法和过程，最后介绍了如何通过 Web 服务集成.NET 应用程序和 Java 应用程序。

本书适合于有一定开发经验并希望掌握如何创建 Web 服务和.NET 应用程序的中高级用户。

Lonnie Wall Andrew Lader : Building Web Services and .NET Applications
EISBN：0-07-213047-4

图书在版编目(CIP)数据

构建 Web 服务和.NET 应用程序/(美) 沃尔，(美)莱德著；康博译. —北京：清华大学出版社，2002
书名原文：Building Web Services and .NET Applications
ISBN 7-302-05890-3

I. 构... II. ①沃... ②莱... ③康... III. 因特网—程序设计 IV. TP393.4
中国版本图书馆 CIP 数据核字(2002)第 071456 号

出 版 者：清华大学出版社(北京清华大学学研大厦，邮编 100084)
http://www.tup.tsinghua.edu.cn
责任编辑：李阳 徐燕萍
封面设计：康博
版式设计：康博
印 刷 者：北京清华园胶印厂
发 行 者：新华书店总店北京发行所
开 本：787×1092 1/16 **印张**：27 **字数**：691 千字
版 次：2002 年 10 月第 1 版 2002 年 10 月第 1 次印刷
书 号：ISBN 7-302-05890-3/TP・3496
印 数：0001～4000
定 价：54.00 元

前　言

最近，Microsoft 全力推出一种新的基于 Web 服务模型的应用程序设计和部署方法。这种新平台被称之为.NET，它引入一组新的服务器、服务和客户应用程序。对于开发人员来说，它代表一种新方法，通过它设计应用程序可以有多种不同的选择，另外，它还意味着一种挑战，即决定如何综合利用各种功能。为此，我们编写了本书，在本书中，先深入介绍了 Web 服务，然后再详细介绍了 ASP.NET 和 Windows Forms 应用程序的开发方法和过程。

由于.NET 的基础是构建在 XML 之上的，所以我们首先讨论 XML 的概念以及如何在应用程序和 Web 服务中使用它。我们学习的重点是如何利用.NET 中的可用功能设计应用程序，而不是语言的语法和编写代码的技术细节。另外，还针对何时使用 Web 服务和如何创建大型企业应用程序，提供了指导性原则，该原则和信息都是从 30 年以来的应用程序开发经验中得来的。因此，我们讨论现实问题，并提供大量示例，而且这些示例都来自现有软件产品的应用程序。

对于使用.NET 创建应用程序，我们的经验已经被证明是正确且有趣的。希望本书能使您真正感受到利用.NET 开发应用程序是多么的简单有趣。另外还希望您在学习本书之后能够真正掌握如何使用 Web 服务和.NET 平台设计和创建应用程序。

本书读者对象

本书所包含的内容适合所有渴望学习如何设计和构建.NET 应用程序的开发人员。因此，本书在各个示例中都综合使用了多种语言，而不仅仅是某一种编程语言。学习本书您应当具备一些应用程序开发经验，但并不需要事先了解 Visual Studio .NET 开发环境。对于本书来说，您只需有一些应用程序开发经验和接受新概念的开放意识即可。

本书主要内容

第 I 部分首先回顾了从独立应用程序到分布式.NET 应用程序的演变过程。接着，深入介绍了各种有关 XML 的主题信息，这些主题信息描述了.NET 的基础。随后继续讨论了多种设计技术，并提供基准信息，用于说明 XML 是目前最优秀的分布式设计解决方案的原因。最后，概述了 Web 服务的相关内容以及如何利用该技术进行分布式设计。

第 II 部分首先全面深入地介绍了.NET Framework，其中包括公共语言运行时(CLR)、通用类型系统(CTS)和公共语言规范(CLS)。如果开发人员掌握了 CLR 的工作方式和 CTS 在.NET 应用程序开发中所扮演的角色，就可以很好地理解 CLR、CTS 和 CLS 是如何协同工作的。该部分还探讨如何利用在 SQL Server 2000 中可获得的新功能以及如何利用 ADO.NET 中的强大新功能。最后，该部分详细地介绍了 DataSet(数据集)，并探讨如何在应用程序设计中有效地使用 DataSet。

第III部分着重利用前面两部分介绍的内容创建.NET 应用程序。该部分前面两章主要介绍如何开发 ASP.NET Web 服务和 Web 窗体应用程序。然后，简单介绍了一个独立 Windows Forms(Windows 窗体)应用程序的开发过程，并将这个应用程序演化为使用 Web 服务的分布式

设计。接着，我们介绍 Windows Forms 应用程序是如何利用 Web 服务与 Java 应用程序进行交互的。另外，这一部分还介绍在.NET 中配置和使用调试功能的必要步骤，从中可以发现，.NET 中的调试非常简单，并被综合运用到应用程序的所有层中。这样，在一次会话中，不费吹灰之力就可以从客户代码执行到服务器代码并返回到客户端脚本代码。

示例代码

本书中的任何示例代码和开发的应用程序都可以从 McGraw Hill/Osborne 的 Web 站点(http://www.osborne.com)中随意免费下载。由于本书的重点不在于编写代码，而是突出设计，所以您在学习的过程中不必下载本书代码。当然，如果您希望继续使用和检查讨论的实际代码，就需要下载示例代码。

目　　录

第Ⅰ部分　XML 概 述

第 II 部分　.NET Framework

第III部分　.NET 服务和应用程序

第 I 部分　XML概述

主要目的：

- 描述 Windows 应用程序开发从客户/服务器(client/server)模型到利用.NET 的分布式服务的演变过程。
- 在深入讨论 XML 结构、XML schema(XML 模式，文档为 XSD)、XML 样式表(XSLT)、XPath 和 DOM 规范等的基础上全面理解 XML 概念。
- 描述了 DOM 对象如何映射到分层数据的实用示例，以及开发人员如何利用结构化的 XML 数据提高性能和可伸缩性。
- 掌握如何在应用程序中充分利用 SOAP，以及如何利用 ASP 和 HTML 实现 SOAP 解决方案。
- 掌握何时使用 Web 服务以及如何设计基于 Web 服务的体系结构。

第1章 Windows分布式平台

本章主要内容：

- 分布式数据
- 分布式组件
- 分布式体系结构
- 分布式服务
- 小结

最近几年，Microsoft Windows 平台已经从带有分布式数据的传统客户/服务器平台演变到支持不同分布式服务的平台。本章将高度概括这个演变过程，首先介绍客户/服务器模型中的分布式数据，最后介绍新的 Microsoft .NET 平台中的分布式服务。在介绍的过程中，我们讨论了开发人员所面对的一些问题，以及解决这些问题的办法。其中的重点集中在为支持分布式处理而演化的技术上，该技术将有利于正确定义分布式处理的概念。然后，我们介绍了 XML 和简单对象访问协议(Simple Object Access Protocol，SOAP)是如何推进从 Windows 平台到实际分布式服务平台的演变过程的。本章最后一节还概述了贯穿本书的.NET Framework 主题。

1.1 分布式数据

从 20 世纪 80 年代后期到 90 年代中期，大部分开发工作都集中在客户/服务器设计上。一个客户/服务器设计通常被视为一个带有用户接口的应用程序，客户可以通过这个接口访问服务器上的数据库引擎。需要注意的是，数据库引擎不仅表示数据，而且还能够执行各种数据操作，例如，数据的排序和过滤。对于客户/服务器设计来说，另外一个重要术语是双层(two-tiered)，这意味着在客户/服务器设计中存在两个不同的层。在实际的应用过程中，双层并不要求两个独立的计算机，单独一个计算机也能够同时作为客户和服务器。双层客户/服务器模型表示一种设计，这种设计可以通过网络将数据库引擎分布到多个客户上，如图 1-1 所示。随着 Microsoft Windows 平台的演变，人们开发了许多能够使用分布式数据的新服务。

在演变期间，个人计算机(Personal Computer，PC)的商业应用越来越广泛。许多小企业开始使用 PC 进行账目和办公管理，一些大企业也在不需要大型机的地方使用 PC。人们正在大量开发基于 PC 的客户/服务器商业应用程序，促使 PC 开始取代传统大型机系统。此外，还需要提供对数据库引擎的共享访问功能，因为目前的 PC 技术还没有真正实现这个功能。所有这些问题以及用户接口开发技术的进步，促使 Microsoft 开发了几个关键的解决方案，以解决这些要求中的一部分。这期间开发了两个关键的解决方案，分别是开放式数据库连接(Open database Connectivity，ODBC)和对象链接与嵌入(Object linking and embedding，OLE)，它们可以帮助

使用组件对象模型(Component Object Model，COM)。

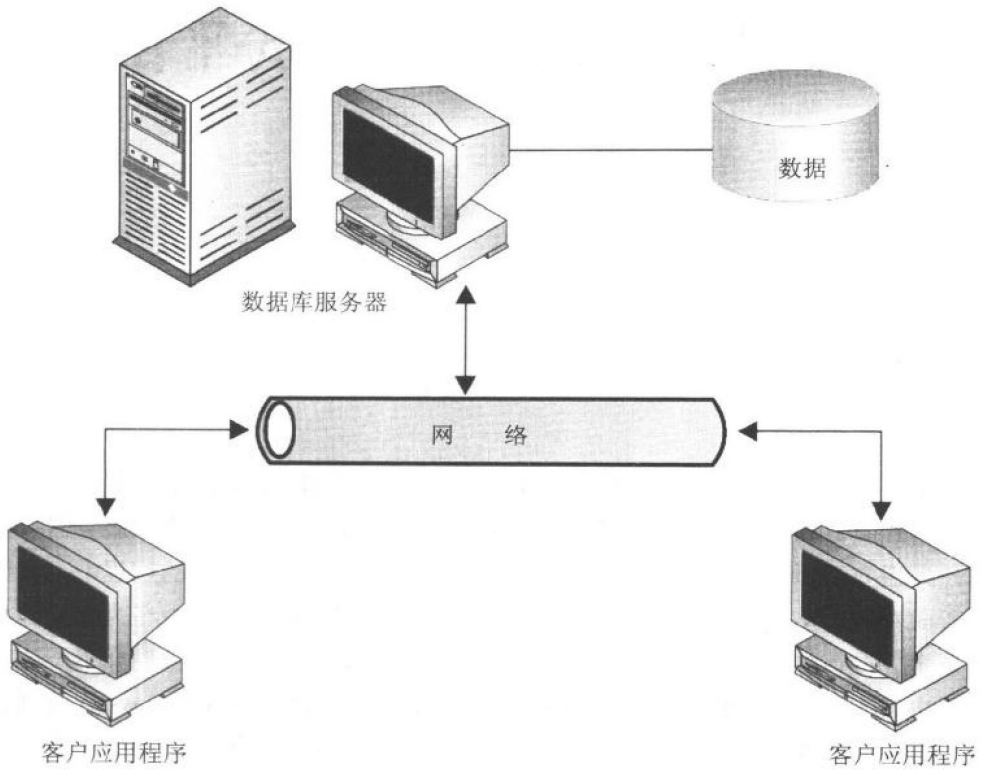


图 1-1 双层物理配置

1.1.1 开放式数据库连接(ODBC)

早期设计的客户/服务器应用程序将所有的业务逻辑都封装在应用程序中。通常，使用数据库引擎根据所执行的操作保存数据和提供不同的数据视图。每一个数据库引擎都有一个不同的应用程序编程接口(application programming interface，API)或者数据库管理系统(database management system，DBMS)，它们通常与应用程序位于同一台机器上。利用共享文件，或者为每一个数据库引擎开发专用 DBMS，可以实现共享数据的访问。但这对于开发人员来说有一点难度。如果开发人员熟悉的数据库引擎不支持所需的功能，例如对数据的共享访问，则开发人员必须学习使用新接口。每一个 DBMS 都有一个新接口，并有自己的查询语言版本，其中查询语言用于执行数据库操作。1992 年，Microsoft 推出了 ODBC，它有效地解决了大部分相关问题。

ODBC 基于 X/Open 数据管理规范(X/Open Data Management Specification)、SQL 调用级接口(Call-Level Interface，CLI)和 ISO/IEC 调用级接口规范。对于关系数据库引擎，ODBC 将结构化查询语言(structured query Language，SQL)作为公共语言使用。这可以利用为每一个 DBMS 开发的驱动程序来实现，其中驱动程序是应用程序和数据库引擎之间的公共接口。这些驱动程序将 SQL 命令翻译成 DBMS 专用的命令，并允许直接访问底层的 DBMS。在这些驱动程序的顶部有一个驱动程序管理器(driver manager)，它专门负责加载驱动程序和处理客户请求，它也处理与数据库相关的其他行为，如连接到数据库，处理命令和管理事务处理等。由于 ODBC 构