

.NET Web 服务入门经典

——VB.NET 编程篇

Joseph Bustos 著

Karli Watson

耿艳楼 译

田玉卓

清华大学出版社

北京市版权局著作权合同登记号：01-2002-3194

内 容 简 介

在.NET中, Web服务是最重要的特征之一, 它允许我们使用HTTP和SOAP等标准协议在Internet或者intranet上的任一位置直接进行方法调用。本书以Visual Basic.NET为编程语言, 全面介绍Web服务的相关概念和技术内幕, 并具体介绍创建Web服务框架的核心内容以及部署、发布和使用Web服务的方法。

本书适用于希望掌握利用Visual Basic.NET创建功能强大、健壮且用户友好的Web服务的开发人员。对于所有希望全面掌握Visual Basic.NET和Web服务的编程人员, 本书也是不错的参考读物。

Joseph Bustos, Karli Watson : Beginning .NET Web Services using Visual Basic .NET

EISBN: 1-86100-725-6

Copyright©2002 by Wrox Press Ltd.

Authorized translation from the English language edition published by Wrox Press Ltd.

All rights reserved.

Chinese simplified language edition published by Tsinghua University Press.

本书中文简体字版由英国乐思出版公司授权清华大学出版社出版。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, 翻印必究。

本书封面贴有清华大学出版社激光防伪标签, 无标签者不得销售。

图书在版编目(CIP)数据

.NET Web 服务入门经典——VB.NET 编程篇/(美)巴斯托斯著, (美)沃特森著; 耿艳楼, 田玉卓译.
—北京: 清华大学出版社, 2002

书名原文: Beginning .NET Web Services using Visual Basic .NET

ISBN 7-302-06198-X

I. N... II. ①巴...②沃...③耿...④田... III. BASIC 语言—程序设计 IV. TP312
中国版本图书馆 CIP 数据核字(2002)第 106240 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.com.cn>

责任编辑: 郭东青

封面设计: 康博

版式设计: 康博

印 刷 者: 北京牛山世兴印刷厂

发 行 者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 21.75 字数: 556 千字

版 次: 2003 年 1 月第 1 版 2003 年 1 月第 1 次印刷

书 号: ISBN 7-302-06198-X/TP·3705

印 数: 0001~4000

定 价: 45.00 元

前 言

本书读者对象

本书适用于希望学习如何利用 Visual Basic .NET 创建功能强大、健壮且使用方便的 Web 服务的人员。读者学习本书时，需具备基本的 Visual Basic .NET 编程技能，并对数据库的概念有所了解。如果读者使用过 Visual Studio .NET IDE，则有助于对本书的学习。

本书主要内容

本书主要介绍以下内容：

第 1 章，“Web 服务基础知识”。本章定义了什么是 Web 服务；介绍了与 Web 服务相关的一些标准；最后还介绍了 Web 服务的具体内容，其中包括创建并运行一个简单的 Web 服务。

第 2 章，“Web 服务数据”。本章介绍了 XML 基础知识、如何表示数据结构以及表示数据结构的最佳方式，同时还介绍了使开发人员标准化数据表示方式的命名空间与架构；最后，介绍了如何用 .NET Framework 处理 XML。

第 3 章，“描述 Web 服务”。通过说明 Web 服务能够交换什么数据以及要将数据转换成什么格式来说明如何描述 Web 服务。本章还介绍了 Web 服务描述语言(Web Services Description Language, 简称 WSDL)、WSDL 文档的结构以及如何生成 WSDL 文档。最后，介绍了通用描述、发现和集成(Universal Description, Discovery, and Integration, 简称 UDDI)协议。该协议用于发布位于中央目录的 Web 服务信息。

第 4 章，“Web 服务活动与通信”。本章介绍了如何在 Web 服务与服务的客户之间进行通信；如何利用 HTTP 和 SOAP 访问 Web 服务，以及 SOAP 的功能及其语法。

第 5 章，“.NET Web 服务”。本章介绍了如何在 .NET 中创建、使用 Web 服务。本章首先介绍如何用 ASP.NET 创建 Web 服务，然后介绍如何用 VS.NET 创建 Web 服务；还介绍了利用 .NET Framework 实现的更为高级的 Web 服务技术，其中包括 HTTP 和 SOAP 头、SOAP 扩展等。最后，介绍了 XML 串行化以及如何定制串行化的行为等。

第 6、7 章，“数据用作 Web 服务(I)”、“数据用作 Web 服务(II)”。这两章介绍



了 Web 服务的主要应用之一：从数据库提取数据，并将数据作为 Web 服务响应传递。

第 8 章，“Web 服务实际应用(I)”。本章介绍创建 Web 服务框架的核心内容。介绍了一系列电子商务 Web 服务的设计与实现，其中包括通过创建数据库与存储过程实现的各设计阶段。

第 9 章，“Web 服务实际应用(II)”。本章介绍了 Web 服务的部署与发布功能，还介绍了如何使用 Web 服务。

学习本书需要的准备工作

学习本书需要一些相关软件与知识。

1. 需要的软件包括：

- Microsoft Windows 2000 或 XP。
- Visual Studio .NET。可以是完全版，也可以是 Visual VB.NET Standard 包版。
- SQL Server 2000 或 MSDE。

2. 需要的知识包括：

- VB.NET 语言方面的知识
- SQL 方面的知识

用户支持

我们一贯重视读者的意见，并想知道每位读者对本书的看法，包括读者喜欢和不喜欢的内容，以及读者希望我们下一次完善的地方。您可以通过发送电子邮件(地址为 feedback@wrox.com)来向我们反馈意见。请确保反馈信息提到本书的书名。

如何下载本书的示例代码

当您访问 Wrox 公司站点(地址为 <http://www.wrox.com/>)时，通过 Search 工具或书名列表，可以方便地定位需要的书目。然后，单击 Code 列中的 Download 超链接，或者单击本书的详细页面中的 Download Code 超链接，就可以下载相应的示例代码。

当您单击并下载本书中的代码时，将会看到带有以下三个选项的 Web 页面：

- 如果您是 Wrox Developer Community 的成员(即如果您已经在 ASPToday、C#Today 或者 Wroxbase 上注册)，就可以使用一贯的用户名和密码进行登录以

下载代码。

- 如果不是它们的成员，则会向您询问是否愿意注册为会员，以便可以免费下载代码。此外，也可以从 Wrox Press 下载免费的文章。注册为会员后，可以得到本书升级版本和新版本的有关信息。
- 第三个选项是完全绕过注册过程，直接下载代码。

对于本书而言，不注册也能下载代码，但是，如果您愿意注册后下载代码，您的注册信息不会泄漏给第三方。关于这方面的详细的条款和条件，可以通过点击下载页面上的相关链接来查看。

从我们的站点上下载的文件都使用 WinZip 压缩过的文档。保存文件到本地磁盘上的文件夹中后，需要使用一个解压缩程序(例如 WinZip 或 PKUnzip)来解压缩文件。在解压缩文件时，通常将代码解压缩到每一章所在的文件夹中。在解压缩的过程中，应确保解压缩程序(WinZip、PKUnzip 等)被设置为使用原有文件夹名。

勘误表

我们已经尽最大努力确保本书中的文本和代码没有错误，但是错误仍然在所难免。如果您发现本书存在错误，例如拼写错误或不正确的代码段，请反馈信息给我们，我们将不胜感激。勘误表的发送可以节约其他读者学习本书的时间，而且能够帮助我们提供更高质量的信息。您的反馈信息将被检查，如果正确，将被粘贴到本书的勘误页面上，或者在本书的后续版本中使用。

要在我们的站点上找到勘误表，请访问 <http://www.wrox.com/>，并通过 Advanced Search 或者书名列表轻松定位本书页面。然后，单击 Book Errata 超链接即可，该链接位于本书的详细页面中的封面图解下面。

E-mail 支持

如果您希望直接向详细了解本书的专家咨询本书中的问题，可以发送电子邮件到 support@wrox.com，要求在邮件的主题栏中带上本书的书名和 ISBN(国际标准图书编号)的后 4 位数字。一个典型的电子邮件应包括下面的内容：

- 在主题栏中必须有本书的书名、ISBN 的后 4 位数字和问题的页码。
- 邮件正文中应包括读者的名字、联系信息和问题。

我们将不给您返回无用邮件，因为我们仅仅需要有用的详细资料，以便节约您和我们的时间。当您发送一个电子邮件信息时，它将经过下面一系列的支持：

- 客户支持：首先，您的信息将被递送到我们的客户支持人员手中，并由他们阅读。对于一些被频繁提到的问题将被归档，并将立即回答有关本书或者 Web



站点的任何常见问题。

- 编辑支持：接着，一些深层次的问题将被送到对本书负责的技术编辑手中，他们在程序设计语言或者特定的产品上有着丰富的经验，能够回答相关主题的详细技术问题。
- 作者支持：最后，如果编辑不能回答您的问题(这种情况很少发生)，他们将请求本书的作者。我们将尽量保护作者免受干扰，以便不影响其写作。然而，我们也非常高兴转寄给他们一些特殊的问题。所有 Wrox 公司的作者都为他们的书提供技术支持。作为回应，他们将发送电子邮件给用户和编辑，进而使所有的读者受益。

Wrox 公司的支持过程仅仅对那些与我们出版的书目内容直接相关的问题提供支持，对于超出常规书目支持的问题，您可以从 <http://p2p.wrox.com/> 论坛的公共列表中获得支持信息。

p2p.wrox.com 站点

为了便于作者和其他人讨论，特将讨论内容加入到 P2P 站点的邮件列表中，而且我们独特的系统将 programmer to programmer™(由程序员为程序员而著)的编程理念与邮件列表、论坛、新闻组以及所有其他服务内容(一对一的邮件支持系统除外)相联系。如果您向 P2P 发送一个问题，应该相信它一定会被登录邮件列表的 Wrox 公司作者和其他相关专家所检查到。无论您是在阅读本书，还是在开发自己的应用程序，都可以在 p2p.wrox.com 站点中找到许多对自己有所帮助的邮件列表。

按照下面的步骤可以预订一个邮件列表：

- (1) 登录 <http://p2p.wrox.com/> 站点。
- (2) 从左边的主菜单栏选择一个适当的种类。
- (3) 单击希望加入的邮件列表。
- (4) 按照说明订阅并填写自己的邮件地址和密码。
- (5) 回复您收到的确认邮件。
- (6) 使用预定管理程序加入更多的邮件列表并设置自己的邮件首选参数。

本系统提供最佳支持的原因

您可以加入整个邮件列表，也可以只接收每周的邮件摘要。如果您没有时间和工具来接收邮件列表，可以直接查找我们的在线文档。独特的 Lyris 系统可以将一些没有用的垃圾邮件删除，并保护您的电子邮件地址不被侵扰。当存在加入和离开列表以及任何有关列表的其他常见问题时，请发送邮件到 listsupport@p2p.wrox.com。

目 录

第 1 章 Web 服务基础知识	1
1.1 Web 服务概述	1
1.2 Web 服务的必要性	2
1.3 何时使用 Web 服务	3
1.4 Web 服务如何工作	5
1.5 .NET 中的 Web 服务	9
1.6 小结	11
第 2 章 Web 服务数据	12
2.1 XML 基础知识	12
2.2 XML 中的数据表示	20
2.3 XML 架构	22
2.3.1 XML 架构基础知识	23
2.3.2 架构类型定义	33
2.4 命名空间	39
2.5 处理 XML 数据	44
2.5.1 文档对象模型	45
2.5.2 SAX	46
2.6 .NET 中的 XML	46
2.6.1 命名空间 System.Xml	47
2.6.2 处理 XML	56
2.6.3 加载、保存 XML 文档	59
2.7 小结	69
第 3 章 描述 Web 服务	70
3.1 Web 服务描述	71
3.1.1 Web 服务描述语言	71
3.1.2 WSDL 扩展	84
3.1.3 WSDL 示例	92
3.1.4 使用 WSDL	96
3.2 Web 服务发现	103
3.2.1 UDDI	103



3.2.2	VSDISCO	103
3.3	小结	104
第 4 章	Web 服务活动与通信	105
4.1	简单对象访问协议(SOAP)	105
4.1.1	SOAP 结构	106
4.1.2	在 HTTP 上使用 SOAP	113
4.1.3	多部分消息	114
4.1.4	SOAP 错误处理	116
4.1.5	定义 SOAP 消息结构	118
4.2	Web 服务通信示例	124
4.3	小结	132
第 5 章	.NET Web 服务	134
5.1	ASP.NET Web 服务	134
5.2	Visual Studio .NET Web 服务	151
5.2.1	在 VS 中创建 Web 服务	151
5.2.2	在 VS 中使用 Web 服务	156
5.3	Web 服务技术	164
5.3.1	使用头	164
5.3.2	SOAP 扩展	178
5.3.3	控制 SOAP 文档格式	194
5.4	在 .NET 中实现 XML 串行化	200
5.4.1	XmlSerializer 类	201
5.4.2	XML 串行化属性	205
5.4.3	生成架构	209
5.4.4	Web 服务 XML 串行化	210
5.4.5	非 XML 串行化	211
5.5	小结	215
第 6 章	数据用作 Web 服务(I)	216
6.1	ADO.NET 概述	216
6.1.1	ADO.NET 对象模型	217
6.1.2	使用 ADO.NET DataReader 类	218
6.2	用于数据检索的 Web 服务	226
6.2.1	WroxEvents 数据库	226
6.2.2	维护静态数据的本地副本	239

6.3 小结	241
第 7 章 数据用作 Web 服务(II)	242
7.1 简单数据添加	242
7.2 DataSet 类	249
7.2.1 DataAdapter 对象	250
7.2.2 更新数据	256
7.2.3 非连接的 DataSet	260
7.3 并发	267
7.4 小结	272
第 8 章 Web 服务实际应用(I)	273
8.1 将电子商务构建成 Web 服务的原因	273
8.2 设计 Web 服务	274
8.3 构建 Web 服务	276
8.3.1 Product Web 服务	278
8.3.2 Customer Web 服务	289
8.3.3 Cart Web 服务	301
8.3.4 Order Web 服务	309
8.4 小结	318
第 9 章 Web 服务实际应用(II)	319
9.1 安全性	319
9.1.1 保证连接安全	319
9.1.2 身份验证与授权	320
9.2 如何部署 Web 服务	323
9.3 使用 Web 服务	328
9.4 小结	333

第1章 Web服务基础知识

欢迎学习本书。本书将介绍如何利用 Visual Basic .NET 和 .NET Framework 构建 Web 服务，以及如何利用这些服务完成各种任务。为实现此目的，首先介绍作为 Web 服务基础的其他一些技术。很好地掌握这些基础技术，可以使自己创建的 Web 服务更加合理，从而不仅能改善程序代码的性能，而且还可以将已有的程序代码(不论是自己的代码还是合作伙伴的代码)很好地集成在一起来提供 Web 服务。本书最后还要介绍一些有关 Web 服务实现方式的“现实世界”应用实例。通过学习这些实例，可以使读者能够扩充已有的程序代码，以创建功能强大的 Web 服务。

本章并不涉及 Web 服务的具体细节，但要用一定的篇幅来介绍其基础知识。首先要介绍的自然是“Web 服务”这一术语的确切含义。此外，还需要了解 Web 服务为什么前景如此辉煌以及何时应该使用 Web 服务。

本章还要介绍与 Web 服务有关的一些标准。这些标准均不依赖 .NET(本书介绍内容的相当大的部分适用于利用其他技术开发的 Web 服务)。在后面的章节中还要详细介绍这些标准。最后，本章还要介绍 .NET 方面的细节：介绍 Web 服务如何适用于 .NET Framework；如何容易地利用 Visual Basic .NET 创建基本的 Web 服务并运行它。

在介绍具体内容之前，首先需要说明的是：最初构建 Web 服务时，会感到似乎处处存在困难，它好像是一个难以入门的技术。由于创建 Web 服务时所涉及的标准似乎非常复杂，为了有效地使用 Web 服务，需要更新设计观念。因此，很容易使人在构建 Web 服务时气馁。但是，一旦创建并运行了 Web 服务，就会忘记上述问题的存在。确实有许多标准，且这些标准没有对应的文档资料。但对于有能力的编程人员来说，是完全能够掌握它们的。事实上，如果这些标准不是以非常枯燥的形式提供的话，掌握这些标准就是一件轻而易举的事。

因此，不要放弃本书的学习。虽然当优化应用程序时花时间全面掌握与 Web 服务有关的标准是重要的，但需要说明的是，在许多情况下，这些标准的细节会变得并不重要。一旦掌握了基本的标准，大部分编程工作将涉及标准的 .NET 编程事宜。

1.1 Web 服务概述

本书将“Web 服务”看成是一项技术，该技术为一些功能提供了一个接口，从而可以让其他程序通过 Web 访问这些功能。这里提到的功能可能是从数据库得到数据、



完成一个数学计算,或者是完成更复杂的任务(如对哈勃太空望远镜提供远程控制等)。在本书中介绍的不同种类的 Web 服务之间通信时需要使用专门的标准与协议。这些标准和协议可以使人们创建允许任何人通过 Internet 使用的 Web 服务,并允许 Web 服务使用任意数量的第三方应用程序。这些标准要规定以下内容:

- 用于数据交换的与平台无关的格式。
- 描述数据结构的方式。
- 包装数据的标准方法,可能需要通过 Internet 来传输这些数据。
- 使 Web 服务描述其自身的描述方式、Web 服务能完成的操作以及客户需要的数据结构。
- 框架,该框架能通过 Web 服务的功能或描述、利用程序确定 Web 服务的位置。

对于非技术性介绍而言,知道有这些标准以及知道它们的作用就已足够了。有关这些标准本身更详细的介绍,见本章“Web 服务标准”一节。

1.2 Web 服务的必要性

随着 Internet(以及网络)的成熟,电子商务对商务(B2B)集成正在快速发展。这里的集成指在用于多种业务活动的计算机之间,通过 Web 来直接通信,且通信时通常很少有人工干预或没有人工干预。此集成只是最近才对大多数机构可行,且随着技术的不断改进以及成本的不断下降,已越来越受到欢迎。这种集成的优势是显而易见的:它可以提高生产力、降低由于人为错误而引起的滞后。这两方面均直接影响着公司的生存。

此外,社团应用程序复杂性的增加更加需要分布式应用程序体系结构。当今的应用程序很少与它们需要的所有资源放在一台服务器上,甚至很少放在一个平台上。实际上,应用程序和资源均可能分布于许多计算机:有些通过本地网络连接,有些则不然。将来,应用程序的两个模块之间的惟一连接将通过 Internet 实现。这意味着即使是单个的公司也能够从 B2B 事务处理需要的电子通信中受益。

因此,对于属于单独机构或多个团体的计算机而言,它们均希望彼此通过 Internet 交流。利用适用于 Internet 通信的最基本的协议,可以有无数种方法来实现这一点。但创建实现此目的的程序时会轻而易举地用掉数月的开发时间以及大量的资金。如果读者尝试此工作的话,可能会以极高的效率以及创新的结果而告终。然而,当希望自己的系统与不同机构的系统交流时应该怎么办呢?在两个不同机构之间进行翻译也许不是一件容易的事,但却是可能的。

过去,人们做了许多努力来利用分布式体系结构系统解决上述问题。但大多数系统甚至在企业范围就有部署上的缺陷,更不用说在全球范围了。其中涉及的问题包括将

部署只限制在指定的平台以及诸如在社团防火墙中打开请求端口这样的安全性事宜等。造成这些问题的主要原因是较大的供应商们不能就采用统一的方式来完成工作方面达成协议。然而, 这些供应商们现在面临着一个不可更改的事实, 即彼此必须共存, 以跻身于全球市场中。这也是实现标准化的驱动力所在。

上述原因使业界产生了开发新标准的要求, 从而也就产生了 Web 服务。这些标准使得快速、低费用地创建、使用 Web 服务成为可能。它既适用于自己的 Web 服务, 也适合由其他机构提供的服务。利用这一事实, 就可以考虑利用 Web 服务实现业务活动的可能性, 且可以获得的众多收益。

首先讨论通过一系列扩展了的功能来增加利润的能力, 即利用 Web 服务, 可以潜在地在业务与外部消费之间建立连接、有效地生成一个销售新渠道。例如, 假设有一个机构, 它可以进行开发投资, 以编写程序代码来将内部 HTML 文档转换成 PDF 文件格式。Web 服务则可以容易地提供这样的代码、允许外部机构在支付费用的前提下访问这一功能。此外, 对于内部用户来说, 它还具有简化对此代码访问的优点, 特别是对在家中工作的人来说更是如此。另外, 可以很容易地访问 Web 服务代码意味着不需要在不同的位置复制代码, 因为每个人均访问一个集中化的版本。

Web 服务的另一个主要优点是可以使用户容易地适应更改与升级。有一个标准可以使 Web 服务描述它们自己, 因此对 Web 服务的更改可以容易地传送到 Web 服务客户。例如, 假设有一个 Web 服务用于提供股票值, 以响应股票市场标识符。一旦确定了该服务的位置, 就可以方便地使用它, 因为它为使用服务和表示结果需要的数据格式提供了完整的描述。对内容格式化方式的更改很容易实现, 此时除了要更改将数据串行化到 Web 服务格式的方式外, 在许多情况下只需对客户代码做很小的改动。事实上, 通常可以将对客户代码的这种更改让一个工具来完成, 该工具用于向应用程序提供从 Web 服务中获得的数据, 通常称此工具为代理程序。这样的代理程序能够更新它们自己, 以响应数据格式的变化, 从而可以让用户考虑更重要的事情。

也许 Web 服务的最重要的特性是利用 XML 实现的交叉平台兼容性。即使一个单独的机构使用的是相同的操作系统、相同的软件等(对此也不能过分乐观), 但不能期待其他机构也使用相同的工具。由于 Web 服务各项标准对这种一致性没有要求, 因此可以从连接 Internet 的任何类型的设备中使用给定的 Web 服务, 这样的设备可以是移动电话, 也可以是大型机。

1.3 何时使用 Web 服务

可以将分布式体系结构分成两个主要部分, 即内部分布式体系结构和外部分布式体系结构。内部分布式体系结构涉及机构如何通过其内部网络来部署应用程序这样的



事宜；而外部分布式体系结构则涉及企业如何将其应用程序和业务过程与外部机构相集成。

内部分布式体系结构带来的挑战之一是要与新加入的组织相集成，而该组织可能采用的是完全不同的技术解决架构。Web 服务能够简化此集成，因为它不需要强迫任何人来使用任何特殊的技术；开发人员可以在不中断所需要的正常操作过程的前提下提供和使用相关的信息。另外一个挑战是要扩充当前的应用程序，如也许要添加像新硬件这样的外部资源。当 Web 服务进入平衡状态时，用户在一定程度上可以不考虑如何使新添加的内容适应已有的基础结构，因为无论添加什么，Web 服务均能够简化在全异(disparate)设备之间的通信。

与此类似，在分布式处理的外部体系结构中，业务关系得到了简化。例如，假设您是某一运输公司的老板，您的公司愿意与名为 Sports Warehouse 的公司做生意。Sports Warehouse 公司可通过 Web 服务提供全部的订货系统和产品数据，而您的运输公司则能够提供服务：允许 Sports Warehouse 公司查询您的系统，以得到运输费用方面的信息；提交运输要求；或得到当前所运货物的状态。由这些 Web 服务提供的功能组合能够保证您与 Sports Warehouse 公司之间的无缝集成。当然，这并不是新构思，但在几分钟内就可以集成、而不需要用于实际实现方式的技术的这种能力正是 Web 服务所具有的功能。只要两个机构在使用 Web 服务语言，就可以大大缩短集成时间。

读者也许会问：什么时候不应该使用 Web 服务呢？内部分布式体系结构的主要问题是性能问题。通过 HTTP(甚至是在本地网络)通信可能要比采用更直接的通信方式慢，且由于需要将数据转换成用于 Web 服务的与平台无关的数据或要从与平台无关的数据进行转换，因此会进一步减慢其通信速度。虽然利用 Web 服务能够迅速进行通信，但不应该放弃考虑其他方式。利用 Microsoft 的 DCOM 或 Java 中的对应技术(取决于已存在的技术)，可以在更加紧凑、合理的方案中得到更好的结果。由于本地通信是在完全已知的端点之间进行，因此通常不需要利用 Web 服务中与平台无关这一优点，且为了性能而牺牲可用性是偶尔需要做出的选择。

对于外部分布式体系结构也要考虑同样的问题(以及解决方案)，虽然它主要考虑的问题可能是安全性问题。例如，一些 Web 服务竞争者能够提供更加安全的框架，甚至将用户限制在指定的平台上。这些竞争者的另一个优势是在大多数情况下他们具有良好的素质，具有许多可节省劳动力的有效工具，以及如果用户遇到困难时，他们能提供大量的文档资料。

由于有这么多的难题，因此最后的决策通常要由从事分布式应用程序开发人员的经验来定。如果公司雇用了世界上最好的 DCOM 程序员，则没有必要强迫他们改变他们的工作方式。当然，公司可能要面临该程序员提出的问题：更改 Web 服务技术，但这是完全不同的事情。

1.4 Web 服务如何工作

前面概述了 Web 服务的目的以及业务用途,现在则应该详细了解使服务成为可能的各个标准。底层的所有东西是用于 Internet 通信的主协议,即超文本传输协议(Hypertext Transfer Protocol, 简称为 HTTP)。在大多数情况下,该协议用于 Web 服务通信的传输协议(虽然 Web 服务规范不需要它)。Web 服务通信的基本原理如图 1-1 所示。

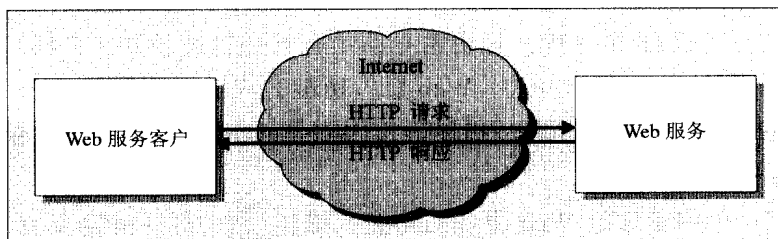


图 1-1

图 1-1 说明了如何通过 HTTP,以请求、响应会话的方式从客户向 Web 服务进行通信。事实上,实际的限制要比这里所示的少。Web 服务通信也可以是单向的,即客户发出请求,但不等待响应。也可以使用 Web 服务启动对话框进行通信,该对话框可以实现单向通信或双向通信。

与访问 Internet 上的其他对象一样,访问 Web 服务也需要 URL。当进行 Web 冲浪时,利用 URL 提供了一个浏览器应用程序,该程序将 HTTP 请求传送到与 URL 相关联的 Web 服务器,并检索包含要显示的 HTML 的 HTTP 响应。返回的 HTML 也许是一些处理结果,如输入到 Web 窗体然后提交给 Web 服务器的信息,此时传送到 Web 服务器(该服务器驻留 Web 页)的 HTTP 请求将包含附加的信息。当查询 Web 服务时,访问 URL,且实际上是以相同的方式访问 Web 服务器。Web 服务和相关联的协议规范会告诉用户在 HTTP 请求中应该放入什么,以及能够检索什么样的内容。

前面介绍过,在 Web 服务中使用 HTTP 并不是强制性的。然而,使用 HTTP 确实可提供许多帮助。对于初学者而言,HTTP Web 服务易于掌握,且使 Web 服务的部署更加容易。此外,HTTP 有一个关系密切的伙伴,即 HTTPS。HTTPS 可以使数据安全地传送。用 HTTP 作为主传输机制的系统可以切换到 HTTPS,这是因为这两个协议的非安全方面是等同的。对于 Web 服务通信而言也是如此。因此,用户不需要做额外的工作,就可以对其 Web 服务的安全性进行选择。



Web 服务标准

本章开始部分对 Web 服务下的定义中,有 5 个方面涉及到用于实现 Web 服务功能的标准与协议。这 5 个方面与下面的实际技术相关:

- 用于数据交换的与平台无关的数据格式:用可扩展标记语言(即 XML)可得到此数据格式。
- 描述数据结构的方式:XML 数据结构是利用 XML 架构确定的。
- 封装数据的标准方法。可能需要通过 Internet 来传输这些数据:XML 数据可以直接封装到 HTTP 消息中,但更常见的是将它打包到 SOAP 文档中。
- Web 服务描述其自身的方式、Web 服务所能完成的操作以及客户需要的数据结构:Web 服务通过 Web 服务描述语言(Web Service Description Language, 简称 WSDL)来描述。
- 框架,该框架用于通过 Web 服务的功能或描述,利用程序确定 Web 服务的位置。可以利用通用描述、发现和集成(Universal Description, Discovery, and Integration, 简称 UDDI)来发现 Web 服务。

下面简要介绍这些技术以及它们在 Web 服务中的作用。

1. XML

XML 是一个标准,它允许将数据编码成纯文本文件。这样做有显著的优点,即任何人都可以读它,因为在所有平台上,文本文件是标准文件。当然,XML 要遵循一些标准,如使用什么语言编码、每个符号要使用多少数据等。但在所有可以使用 Web 服务的设备上,这些问题都是可以解决的。

XML 与 HTML 有很大的相似之处,其中包括它们均为标记(markup)语言这一事实。标记语言利用纯文本来提供结构和含义。该结构和含义通过将纯文本分成若干段、使每一段通过标记文本来定义确切的意思来实现。例如,用 HTML 可以将文本段标记为粗体或斜体,或用标记来指定要插入位图图像。XML 的工作原理与此类似,不同之处是它使用的标记不适合 Web 页。用标记可以表示所希望的任何事情。例如,可以用对应的属性将文本区标记为具体的数据块。

当阅读一个 XML 文档时,要解释标记,以将存储的数据转换成所希望的平台专用的数据格式。在 .NET Framework 中,可以将存储在对象中的数据直接串行化到 XML,并可以容易地将数据反串行化回对象。.NET 对象的串行化版本不需要以这种方式反串行化,因为能够读 XML 的任何东西(如前面所述,这意味着许多事情能够阅读)能够解释数据,将它转换成不同的格式(如果希望的话)。

XML 的灵活性是它能用于所有的 Web 服务通信,甚至可以在 XML 中存储二进制数据(如果希望存储的话)。XML 的惟一缺陷是 XML 文档存储数据的效率不如存储纯二

进制数据那样高，而且需要被解释。这样的缺陷可能会导致在交换和使用数据时使性能受到影响。

本书第 2 章将介绍 XML。

2. XML 架构

XML 架构提供了在 XML 文档中执行规则的方式。这些规则既可以用于文档的结构(即每一位位于文档的什么位置)，也可以用于在特定段中允许的数据类型(例如，“此段必须包含整数”或“此段必须是由三个字母组成的词”)。可以交换与平台无关的 XML 文档，但如果没有像 XML 架构这样的约束，就不能保证 XML 文档中包含有正确的数据，甚至不能保证数据以正确的方式格式化。

XML 架构中最为有趣的一面是 XML 架构本身是在 XML 中编写的，而且完全符合 XML 规范。这就使得 XML 架构易于学习，而且意味着它们可以由能识别 XML 文档的设备识别。

本书第 2 章还将介绍 XML 架构。

3. SOAP

SOAP 曾经是简单对象访问协议(Simple Object Access Protocol)的简称。但随着 SOAP 技术的发展，SOAP 规范中已不再具有此含义。SOAP 的最初设计用于远程处理对象。现在，它的用途更加专业化，几乎是专门用于 Web 服务的实现方式。

前面介绍的 XML 和 XML 架构这两个技术已足以与 Web 服务交换数据。事实上，当直接利用 HTTP 访问 Web 服务时，均需要 XML 和 XML 架构。然而，也可以将 XML 数据放入称为是 SOAP 的消息中。

SOAP 是使用 XML 的另一项技术。将 XML 数据有效地放到 SOAP 消息中意味着要用附加的 XML 包围该消息。这样做有两方面的原因。首先，它意味着与 Web 服务的所有通信符合同样的标准结构；第二，SOAP 协议包括一些对 Web 服务特别有用的功能。

SOAP 消息由包含头段和主体段的信封组成。与 Web 服务交换的 XML 数据位于主体，SOAP 头则用于交换其他功能以及应用程序专用的信息。此 SOAP 头信息可以用这样的方式标记，即 Web 服务必须对它起作用，且必须在它的响应中说明它有作用。在更高级的 Web 服务设计中要使用此功能。SOAP 头也可以包含有关在主体段中的数据编码方面的信息，以及所允许使用的应用程序专用架构方面的信息，后者是 Web 服务的另一高级功能。

主体段的另一主要作用是包含错误信息。在 SOAP 消息处理过程中，如果出现问题，可通过 SOAP 错误信息传递错误。SOAP 错误信息包含了比 HTTP 错误消息更多的信息。.NET Framework 处理 SOAP 错误的方式是一个非常重要的内容，本书后面将介绍此方面的内容。



第 4 章将更详细地介绍 SOAP。

4. WSDL

前面介绍的三种技术均是 Web 服务通信时所需要的技术，但这些技术不能解决这样一个问题：知道在第一个位置要传输什么。为解决此问题，Web 服务有一个相关联的 WSDL 文档，该文档用于描述 Web 服务能够完成的全部操作。

Web 服务操作的描述的组成部分有：所使用的数据结构、在请求和响应服务时存在的这些数据结构的组合、包含这些请求与响应的消息格式以及由服务支持的不同操作需要的特定访问方法。为实现这些功能，WSDL 文档(由 XML 编写)包含有架构信息、将架构定义与消息结构相关联的信息以及一些 HTTP 和 SOAP 专用的信息。

给出一个 WSDL 文档，就可以编写与它对应的 Web 服务的代码，且不需要其他信息。

当然，知道 Web 服务操作的参数以及返回值并不需要告诉您要由 Web 服务做什么样的处理。例如，任何函数适用于以两个数为参数并返回第三个数的操作。

在 .NET 中，很少会发现要直接处理 WSDL，这是因为 .NET Web 服务会自动根据用于服务的代码自动生成该信息。然而，对于一些特殊的方案，却要由用户处理 WSDL。

本书第 3 章将介绍 WSDL。

5. UDDI

WSDL 准确地定义了 Web 服务，但它不足以允许人们找到 Web 服务。如果您不知道 Web 服务的存在，就不能找到描述它的 WSDL。获得 WSDL 时需要一些中心存储库。用户可以利用该库进行查询，以找到 Web 服务。这也是 UDDI 要做的工作。

最初的设想是建立一个非常大的公共存储库，该库包含 UDDI 信息，即可由要确定 Web 服务位置的任何人使用的信息。然而，这样做有它自身的问题(在第 3 章可看到此问题)。现在的趋势似乎是各机构使用其自己的 UDDI 存储库。这样做是可行的，因为能够通过程序来利用 UDDI 确定 Web 服务的位置并使用 Web 服务。因此，如果由于某种原因导致 Web 服务崩溃，UDDI 知识库可用于立即确定服务的位置并备份服务。

6. 协议小结

为说明上述各种技术之间的关系，可以将 Web 服务通信与传送信件联系起来。其中，XML 是写信时使用的语言(就像英语、法语、西班牙语等一样)。由于已经同意使用一种语言，因此能够以可理解的方式读、写信件。XML 架构(在服务的 WSDL 中使用)定义了所使用语言的语法，即如何构成句子、所使用标点的确切位置、要使用什么样的标点符号等。

然后就是信本身了。通常要将信的日期放在信的右上角，将地址放在左边，开头是 Dear John 等。这些均与 SOAP 的作用相同。SOAP 包装了消息内容。最后，如果希望