Visual C#.NET 中文版

Web 服务开发基础

孙永强　杨丽坤　编著

清华大学出版社

(京) 新登字 158 号

内 容 简 介

随着 Internet 的蓬勃发展和软件开发技术的不断进步，XML Web 服务应运而生，它是一种全新的软件开发模式，也必将成为今后主流的软件开发技术。

本书主要介绍如何使用 Visual C#.NET 中文版开发 Web 服务，并详细而系统地介绍了与 Web 服务相关的各种标准规范和协议，主要包括 XML 标准、HTTP 协议、SOAP 协议、WSDL 规范、UDDI 规范，以及全局 XML Web 体系结构等。这些标准规范和协议是开发 Web 服务的基础，因此，本书侧重于这些基础知识的介绍，并通过具体的实例来演示如何使用这些标准规范和协议。

本书适合于已具有 C#语言的基础知识，又想开发 Web 服务的开发人员阅读。

图书在版编目(CIP)数据

Visual C#. NET 中文版 Web 服务开发基础/孙永强，杨丽坤编著. —北京：清华大学出版社，2002
ISBN 7-302-05714 1

Ⅰ. V… Ⅱ. ①孙…②杨… Ⅲ. C 语言—程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2002)第 056284 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：张秋香
封式设计：王伟
封面设计：康博
印 刷 者：北京大中印刷厂
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16 **印张**：23.25 **字数**：551 千字
版　　次：2002 年 8 月第 1 版 2002 年 8 月第 1 次印刷
书　　号：ISBN 7-302-05714-1/TP・3371
印　　数：0001～5000
定　　价：35.00 元

目　　录

第 1 章　Web 服务概述 ····· 1

1.1　Web 服务的概念 ····· 1

1.2　Web 服务的基础结构 ····· 2

1.3　Web 服务的目录 ····· 3

1.4　Web 服务的描述 ····· 4

1.5　Web 服务的消息格式 ····· 4

1.6　全局 XML Web 服务的体系结构 ····· 5

1.7　小结 ····· 7

第 2 章　创建和使用 Web 服务 ····· 8

2.1　创建 Web 服务 ····· 8

2.1.1　定义 Web 服务类 ····· 11

2.1.2　定义 Web 服务的方法 ····· 13

2.2　测试 Web 服务 ····· 16

2.3　创建 Web 服务客户程序 ····· 19

2.3.1　创建 ASP.NET Web 应用程序 ····· 19

2.3.2　创建 Windows 应用程序 ····· 26

2.4　小结 ····· 27

第 3 章　XML 基础知识 ····· 28

3.1　概述 ····· 28

3.2　XML 文档的结构 ····· 31

3.2.1　前言 ····· 32

3.2.2　元素和属性 ····· 33

3.3　命名空间 ····· 35

3.4　示例程序 ····· 38

3.5　小结 ····· 44

第 4 章　XML 架构 ····· 45

4.1　XML 架构的结构 ····· 49

4.2　元素和属性声明 ····· 50

4.2.1　全局元素和属性 ····· 51

4.2.2　出现约束 ····· 52

4.3 类型定义……54
4.3.1 内建简单类型和方面……55
4.3.2 简单类型定义……57
4.3.3 复杂类型定义……60
4.3.4 匿名类型定义……61
4.3.5 定义派生类型……62
4.3.6 抽象元素和类型……64
4.3.7 控制派生类型的创建和使用……66
4.4 注释……67
4.5 元素的内容模型……69
4.6 属性组……73
4.7 命名空间与名称限定……74
4.8 使用其他架构中的类型……77
4.8.1 使用 include 直接引入其他架构中定义的类型……77
4.8.2 使用 redefine 重新定义外部架构中的组件……79
4.8.3 使用 import 导入其他架构中的类型……81
4.9 小结……82

第 5 章 HTTP 协议……83
5.1 概述……83
5.2 HTTP 消息……86
5.2.1 HTTP 请求消息……88
5.2.2 HTTP 响应消息……89
5.3 头字段……90
5.3.1 通用头字段……90
5.3.2 请求头字段……92
5.3.3 响应头字段……96
5.3.4 实体头字段……97
5.4 请求方法……98
5.4.1 OPTIONS 方法……98
5.4.2 GET 方法……99
5.4.3 HEAD 方法……99
5.4.4 POST 方法……100
5.4.5 PUT 方法……100
5.4.6 DELETE 方法……101
5.4.7 TRACE 方法……101
5.5 状态码……101

5.6 示例程序……106
5.7 小结……111

第 6 章 SOAP 协议……112
6.1 概述……112
6.2 SOAP 消息……114
6.2.1 Envelope 元素……117
6.2.2 Header 元素……118
6.2.3 Body 元素……120
6.2.4 Fault 元素……121
6.3 SOAP 消息交换模型……123
6.4 SOAP 协议绑定框架……126
6.5 SOAP 编码……127
6.5.1 简单类型……127
6.5.2 字符串……128
6.5.3 枚举……129
6.5.4 字节数组……130
6.5.5 多态访问器……130
6.5.6 复合类型……130
6.6 把 SOAP 用于 RPC……141
6.6.1 RPC 和 SOAP Body 元素……141
6.6.2 RPC 和 SOAP Header 元素……142
6.6.3 RPC 错误……142
6.6.4 示例……142
6.7 传输层消息交换模式……144
6.7.1 单一请求响应 TMEP……145
6.7.2 错误处理……145
6.8 默认的 HTTP 绑定……146
6.8.1 消息交换操作……148
6.8.2 SOAPAction 特性……150
6.9 示例程序……151
6.10 小结……167

第 7 章 WSDL 规范……168
7.1 概述……168
7.2 WSDL 文档的结构……176
7.2.1 definitions 元素……177

7.2.2 types 元素 ····· 180
7.2.3 message 元素 ····· 181
7.2.4 portType 元素 ····· 182
7.2.5 binding 元素 ····· 184
7.2.6 service 元素 ····· 186
7.3 扩充性元素 ····· 188
7.4 SOAP 绑定 ····· 190
7.4.1 soap:binding 元素 ····· 190
7.4.2 soap:operation 元素 ····· 190
7.4.3 soap:body 元素 ····· 191
7.4.4 soap:fault 元素 ····· 192
7.4.5 soap:header 和 soap:headerfault 元素 ····· 193
7.4.6 soap:address 元素 ····· 193
7.5 HTTP GET 和 POST 绑定 ····· 194
7.6 MIME 绑定 ····· 197
7.7 示例程序 ····· 199
7.7.1 C#语言对 WSDL 的支持 ····· 199
7.7.2 创建示例程序 ····· 201
7.8 小结 ····· 211

第 8 章 UDDI 规范 ····· 212
8.1 概述 ····· 212
8.2 UDDI 数据结构 ····· 214
8.2.1 businessEntity 数据结构 ····· 215
8.2.2 businessService 数据结构 ····· 217
8.2.3 bindingTemplate 数据结构 ····· 218
8.2.4 tModel 数据结构 ····· 220
8.2.5 publisherAssertion 数据结构 ····· 221
8.3 UDDI API ····· 222
8.3.1 查询 API ····· 223
8.3.2 发布 API ····· 230
8.3.3 SDK 中的错误处理 ····· 237
8.4 SOAP 的使用细节 ····· 238
8.5 小结 ····· 240

第 9 章 DISCO 规范 ····· 241
9.1 DISCO 文档的结构 ····· 241

9.2 DISCO 客户端工具 ……242
9.2.1 命令行工具 disco.exe ……243
9.2.2 直接在浏览器中查看 DISCO 文档 ……244
9.2.3 在集成开发环境中发现服务并查看 DISCO 文档 ……244
9.3 DISCO 重定向 ……246
9.4 DISCO 和动态发现 ……247
9.5 在代码中处理 DISCO 文档 ……248
9.5.1 DiscoveryClientProtocol 类 ……249
9.5.2 DiscoveryDocument 类 ……249
9.6 示例程序 ……250
9.7 小结 ……252

第 10 章 使用 ASP.NET 创建 Web 服务 ……253
10.1 在 ASP.NET Web 服务中管理状态 ……253
10.1.1 使用 Application 对象保存数据 ……253
10.1.2 使用 Session 对象保存数据 ……254
10.1.3 示例程序 ……254
10.2 在 Web 服务中访问 ASP.NET 内建对象 ……256
10.2.1 访问 Server 对象 ……256
10.2.2 访问 Request 对象 ……257
10.2.3 访问 Response 对象 ……259
10.3 在 ASP.NET Web 服务中处理异常 ……261
10.3.1 SoapException 异常 ……261
10.3.2 SoapHeaderException 异常 ……265
10.4 小结 ……265

第 11 章 使用 SOAP 报头 ……266
11.1 SoapHeader 类和 SoapUnknownHeader 类 ……266
11.2 SoapHeaderAttribute 特性类 ……267
11.3 创建使用 SOAP 报头的 Web 服务 ……267
11.3.1 定义表示报头条目的类 ……268
11.3.2 向 Web 服务类中添加表示报头条目的字段 ……268
11.3.3 使用 SoapHeader 特性为方法指定要处理的报头条目 ……269
11.4 创建使用 SOAP 报头的客户程序 ……270
11.5 小结 ……273

第 12 章　定制与扩展 SOAP 消息 ····· **274**
12.1　定制 SOAP 消息 ····· 274
12.1.1　SOAP 消息的格式 ····· 274
12.1.2　SoapDocumentMethodAttribute 特性 ····· 278
12.1.3　SoapRpcMethodAttribute 特性 ····· 282
12.1.4　使用 XML 序列化定制 SOAP 消息 ····· 283
12.2　扩展 SOAP 消息 ····· 284
12.2.1　从 SoapExtension 类派生扩展类 ····· 285
12.2.2　从 SoapExtensionAttribute 类派生特性类 ····· 289
12.2.3　应用 SOAP 扩展 ····· 290
12.3　测试 SOAP 消息定制与扩展 ····· 291
12.4　小结 ····· 293

第 13 章　全局 XML Web 服务体系结构 ····· **294**
13.1　WS-Inspection 规范 ····· 295
13.1.1　WS-Inspection 文档的结构 ····· 296
13.1.2　WSDL 绑定 ····· 299
13.1.3　UDDI 绑定 ····· 302
13.1.4　发布 WS-Inspection 文档 ····· 304
13.2　WS-Security 规范 ····· 305
13.2.1　证书交换 ····· 306
13.2.2　消息完整性 ····· 307
13.2.3　消息机密性 ····· 309
13.3　WS-License 规范 ····· 311
13.4　WS-Routing 规范 ····· 314
13.4.1　WS-Routing 前向消息路径的处理机制 ····· 315
13.4.2　使用反向消息路径 ····· 318
13.4.3　WS-Routing 报头条目元素 ····· 321
13.4.4　WS-Routing 错误消息 ····· 323
13.5　WS-Referral 规范 ····· 327
13.5.1　WS-Referral 语句 ····· 328
13.5.2　插入和删除 SOAP 路由器 ····· 330
13.5.3　WS-Referral 查询消息的交换 ····· 332
13.5.4　WS-Referral 注册消息的交换 ····· 333
13.5.5　WS-Referral 报头 ····· 335
13.6　小结 ····· 336

第 14 章　完整的示例程序 ——图书管理 Web 服务 ……337
14.1　数据库设计……337
14.2　创建 Web 服务……342
14.2.1　向 Web 服务中添加数据模块……343
14.2.2　向 BookManager 中添加自定义 SOAP 报头条目……344
14.2.3　向 Web 服务中添加 SOAP 扩展支持……345
14.2.4　定义 Web 服务的方法……348
14.3　创建 Windows 客户程序……353
14.4　小结……357

第1章　Web服务概述

近年来，软件的发展经历了两个重要的阶段：面向对象和组件化开发。其中，面向对象技术可以实现源代码级的重用，而组件开发技术则可以实现二进制代码级的重用。通过使用组件，用户可以像组装零件一样把多个组件组装成一个完整的应用程序。当前，应用最为广泛的两个组件开发标准是 CORBA 和 DCOM，这两个标准都属于专有标准，并且只能在特定的环境中使用。

如何把使用不同编程语言、不同对象模型在不同的操作系统上建立的应用程序集成在一起，并把它们转换成容易使用的 Web 应用程序是当前软件业面临的挑战之一。Web 服务正是针对这种挑战应运而生的，它实际上就是在 Internet 环境中使用的组件开发技术。

1.1　Web 服务的概念

广义上讲，Web 服务就是一种可以通过 Internet 标准协议或规范来访问的应用程序。换句话说，Web 服务就是可以通过 URL 访问的资源，这些资源可以通过编程方式向客户返回信息。

Web 服务是 Microsoft 公司.NET 策略的核心，也是.NET Framework 的关键组件。Web 服务建立在 HTTP、XML 和 SOAP 等开放标准之上，它是 Web 分布式应用程序的基本构建块，同时，也构成了 Microsoft 的可编程 Web 理念的基础。

Web 服务像组件一样，也表示一个封装了一定功能的黑盒子，用户可以重用它而不用关心它是如何实现的。Web 服务提供了定义良好的接口，这些接口描述了它所提供的服务，用户可以通过这些接口来调用 Web 服务提供的功能。开发者可以通过把远程服务、本地服务和用户代码结合在一起来创建应用程序，图 1-1 显示了使用 Web 服务的应用程序的一般模型。

从图 1-1 中可以看出，用户可以在多种不同类型的应用程序(比如桌面应用程序和浏览器等)中通过 Internet 标准协议来访问 Web 服务。这些 Web 服务可以位于不同的物理位置，并且可以在不同的系统中实现，它们的功能通过接口来调用。

与当前传统的组件技术不同，Web 服务使用标准通用的协议，而不是特定于对象模型的协议(比如 DCOM、RIM 或 IIOP，这些协议要求在客户机和服务器上同时具有特定的基础结构)。在 Web 环境中，把组件的实现紧密地绑定在特定的组件技术上是不切实际的，

这将导致应用程序很难维护。而 Web 服务是使用 HTTP 和 XML 等标准协议和数据格式进行通信，所有支持这些协议和数据格式的系统都将能支持 Web 服务。

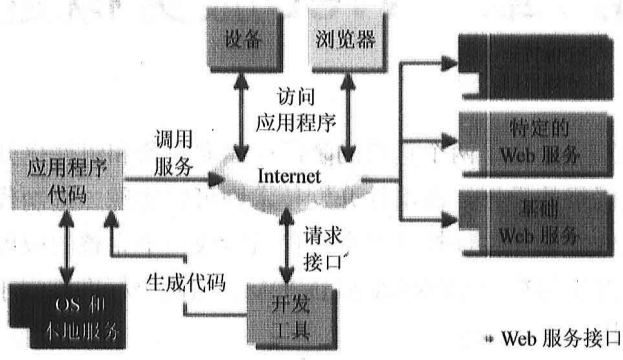

图 1-1　使用 Web 服务的应用程序的模型

更重要的是，Web 服务以消息的形式来提供其服务，它使用基于 XML 标准的消息来作为数据通信的基本方法，这使得 Web 服务完全与开发语言、平台和对象模型无关。Web 服务可以使用任何语言、对象模型在任何的平台上实现，并且任意的应用程序都可以使用 Web 服务。只要描述 Web 服务功能的接口和消息序列以及协议保持不变，Web 服务和客户应用程序就可以任意改变而不会相互影响。

提示：

在 Web 服务中大量地使用 XML 来表示命令和数据，比如 SOAP、WSDL 等都是基于 XML 的。正因为 XML 在 Web 服务中起着至关重要的作用，所以 Web 服务也称为 XML Web 服务。

1.2　Web 服务的基础结构

因为 Web 服务要在异构的 Web 环境中使用，而在这个环境中又存在着多种不同的操作系统、对象模型和编程语言，所以 Web 服务必须具有以下的特点：

- 松散耦合：客户程序仅使用自描述的、基于文本的消息与 Web 服务进行通信。
- 方便的通信：连接到 Internet 上的任意系统或设备都可以与 Web 服务进行通信。
- 通用的数据格式：Web 服务使用被广泛支持的、标准的 XML 来描述其数据格式，所有支持这种标准的系统都可以理解 Web 服务的消息。

Web 服务的基础结构提供以下的功能：定位 Web 服务的发现机制；定义服务用法的服务描述以及进行通信的标准消息格式，图 1-2 显示了 Web 服务的基础结构。

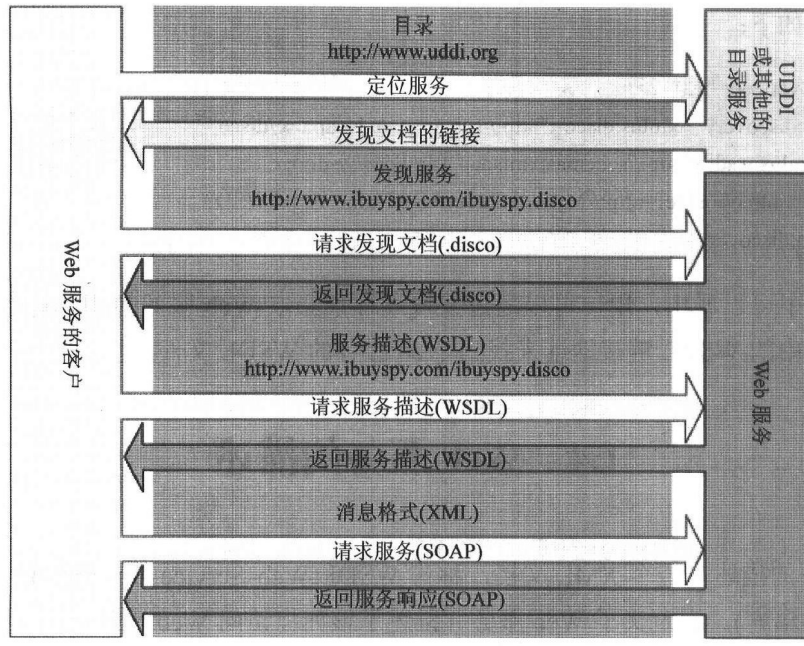

图 1-2　Web 服务的基础结构

上图中的 UDDI、WSDL 和 SOAP 等标准规范都是基于 XML 标准的，下面依次介绍这些标准的规范。

1.3　Web 服务的目录

Web 服务也是 Internet 上的资源，因此必须要有某些方法来查找特定的 Web 服务。Web 服务的目录提供了一个中心位置，这样，Web 服务提供者就可以在这里发布有关他们的服务的信息，同时 Web 服务的客户也可以到这里来查找他们所需要的服务。Web 服务目录本身也可以是一个 Web 服务，它可以响应客户的请求并返回查询结果。用户可以使用 Web 服务目录查找提供特定服务的组织，也可以查找特定组织所提供的服务。

UDDI(Universal Description，Discovery and Integration，通用描述、发现和集成)规范定义了一个发布和发现 Web 服务的标准方法。UDDI 是一个基于 SOAP 协议的 Web 服务，它为企业或开发人员提供了一个在企业环境或 Internet 环境中共享 Web 服务的基础。

从图 1-2 中可以看出，当用户向 UDDI 服务发送请求时，将会获得一个指向 Web 服务发现文档(.disco 文件)的链接，然后就可以使用该链接来访问具体的.disco 文件。

每一个 Web 服务都可以有一个.disco 文件，它是一个 XML 文档，其中包含了描述 Web 服务的 WSDL 文档的链接，用户可以通过它来获取 Web 服务的 WSDL 文档的位置。例如，在下面的.disco 文件中指出 WSDL 文档的地址为：“http://MyWebServer/UserName.asmx?WSDL”。

文件代码如下：

```
<?xml version="1.0" ?>
<disco:discovery  xmlns:disco="http://schemas.xmlsoap.org/disco"
    xmlns:wsdl="http://schemas.xmlsoap.org/disco/wsdl">
<wsdl:contractRef ref=http://MyWebServer/UserName.asmx?WSDL />
</disco:discovery>
```

从图 1-2 中可以看出，用户可以通过发现过程来获取 Web 服务的.disco 文件，然后再通过这个文件中的 WSDL 链接地址来找到 Web 服务的 WSDL 文件。

1.4 Web 服务的描述

Web 服务的描述是一个 XML 文档，称作 WSDL(Web Service Description Language，Web 服务描述语言)，它定义了 Web 服务并描述了该如何访问 Web 服务。

WSDL 文档把 Web 服务定义为一组处理消息的网络端点或端口(port)。在 WSDL 中，端点和消息的定义都是抽象的，必须把它们绑定到具体的网络协议和消息格式上才能定义一个具体的端点。

在 WSDL 中包含了数据类型的定义、消息格式的定义、服务所支持的操作的定义等多种元素。

WSDL 是可扩展的，用户可以使用它来描述端点和消息而不用考虑实际通信所使用的网络协议是什么。WSDL 可以和 SOAP、HTTP 以及 MIME 等协议一起使用。

当用户获取 Web 服务的 WSDL 文档之后，就能从中得知 Web 服务所处的位置、它所包含的方法和每个方法的参数以及返回值的类型等信息，然后就可以使用这些信息来执行方法调用了。

1.5 Web 服务的消息格式

用户可以使用一个或多个开放协议(比如结合使用 HTTP 和 SOAP)来构造 Web 服务。当然，使用不同的协议时，相应使用的基础结构也将有所不同。

Web 服务不仅仅局限于提供远程过程调用(RPC，Remote Procedure Call)方式的访问，它们也可以用来交换结构化的信息，比如订单和发票数据，并且也可以用来自动化以及连接内部和外部的商业过程。

HTTP 协议使用标准的 HTTP 方式进行参数编码，并使用名称/值对的方式来传送参数。每一个 HTTP 请求都包含一组请求头和具体信息，而每一个 HTTP 响应则包含一组响应头和响应数据。HTTP-GET 方法把参数直接附在 URL 的后面，这种方式也称作查询字符串；

HTTP-POST 方法通过把参数的名称/值对放到实际的 HTTP 请求消息中来传送参数，而不是放到 URL 中传送。

SOAP 是一种用来在 Web 上交换结构化类型信息的基于 XML 的简单的轻型协议。这个协议定义了一个消息传送框架，它具有高度的模块化特点和高度的可扩展性。SOAP 使用标准的传输协议来传送消息，它可以充分利用现有的 Internet 开放体系结构，并且，可以很容易地被任意系统所支持。

1.6　全局 XML Web 服务的体系结构

随着 XML Web 服务解决方案被越来越多的人所接受并且变得越来越复杂，为其提供可用性、可靠性和安全性也就变得日益重要。为此，Microsoft 发布了一系列新的规范这些规范构成了全局 XML Web 服务体系结构。这些规范建立在当前 XML Web 服务的标准规范之上，其目的就是为基本的 XML Web 服务规范提供额外的功能。图 1-3 显示了全局 XML Web 服务的体系结构。

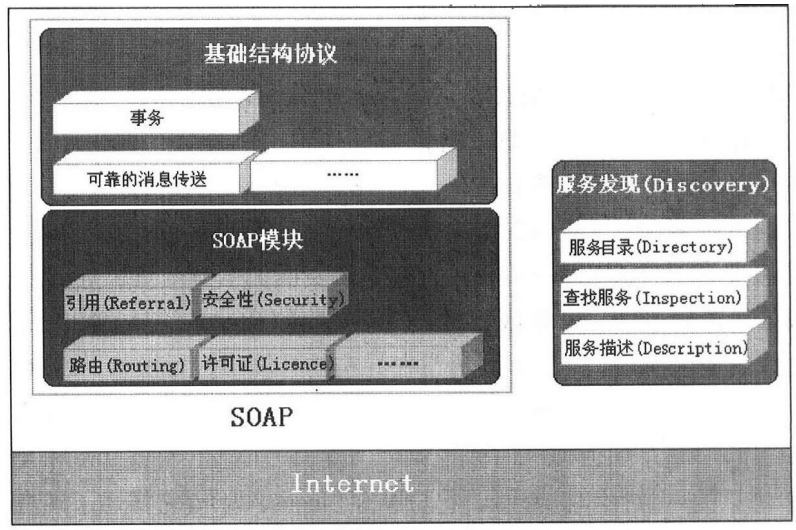

图 1-3　全局 XML Web 服务体系结构

从图 1-3 中可以看出，XML Web 服务体系结构建立在 Internet 和 XML 技术之上，它是一个组件化的体系结构，这表示体系结构中的规范要和其他规范一起使用以实现特定的需求。这个体系结构分为 3 层：SOAP 层、SOAP 模块层和基础结构协议层。

SOAP 是一个基于 XML 的，用来在分散的和分布的环境中传送信息的、轻型的和可扩展的协议。SOAP 主要定义了一个用来描述消息结构和消息处理模型的框架，同时也定义了一组序列化数据的编码规则以及一个执行远程过程调用的约定。

SOAP 模块利用 SOAP 的可扩展能力来构造较高级的功能。SOAP 模块比较简单并且

功能集中，它表示可以单独使用或者组合使用的功能元素。SOAP 模块所提供的功能相对稳定，通过它可以充分利用 XML Web 服务的基础结构。

基础结构协议建立在 SOAP 模块之上，它用来提供端对端的功能。它们可以跨越消息序列维护状态并且可以把多个消息的结果聚集成一个较高级别的结果。

全局 XML Web 服务体系结构具有 4 个设计原则。

- 模块化：这个体系结构建立在模块化的组件之上，而不是定义一个庞大的提供端对端功能的规范。这些模块可以组合在一起来解决现实问题。这种构建模块方式的一个明显好处就是可以创建一个灵活的体系结构，当需要添加一个新特性或者扩展已有特性时，只需创建或修改一个协议就可以了。
- 通用目标：全局 XML Web 服务体系结构主要用于广泛的 XML Web 服务环境，包括 B2B、EAI 以及 B2C 等应用环境。它所包含的模块相互独立，既可以单独使用也可以组合使用。
- 协同作用：全局 XML Web 服务体系结构不需要中心服务器或集中管理。这个体系结构允许跨越可信任边界，在自治实体间进行通信。体系结构中的所有模块和协议都不关心消息端点的实现技术。在消息端点可以使用任何技术，并且消息端点的技术可以任意地升级，而且消息端点所提供的服务可以任意地委派给其他端点或者任意地从其他端点引用服务。
- 完全建立在标准之上：全局 XML Web 服务体系结构完全建立在标准的 XML Web 服务规范(比如 SOAP、WADL 和 UDDI 等)之上。

Microsoft 推出了 4 个新的规范—— WS-Security、WS-License、WS-Routing 和 WS-Referral，这些规范均建立在 SOAP 之上。

建立和管理安全的 XML Web 服务的组织需要确保只有通过验证的人才被允许使用 XML Web 服务，并且 XML Web 服务所发送和接收的 SOAP 消息只能被适当的人修改和查看。因为这些安全的 XML Web 服务通常在异构的环境中进行操作，所以，底层的 XML Web 服务安全性协议必须非常灵活。

- WS-Security 描述了如何使用现有的 W3C 安全规范(XML Signature 和 XML Encryption)来确保 SOAP 消息的一致性和机密性。它与 WS-License 规范(它描述了现有的数字证书和相关的信任语义如何安全地关联到 SOAP 上)一起构成了 XML Web 服务安全体系的底层。将来的安全规范将建立在这个基础之上，以提供证书交换、信任管理、证书撤消以及其他高级功能的机制。
- WS-Security 是一个简单的、无状态的 SOAP 扩展，它描述了数字证书应该如何被放到 SOAP 消息中，以及这些证书应该如何被关联到一个消息上以确保消息的完整性和机密性。通过使用 WS-Security，XML Web 服务可以检查收到的 SOAP 消息，并且根据它所包含的证书来确定是否处理这个请求。WS-Security 支持多种数字证书和技术，包括公共密钥和对称密钥加密技术。

WS-License 描述了如何在 WS-Security 证书中使用多种通用的许可证格式(包括 X.509

证书和 Kerberos 证书)。

因为在全局 XML Web 服务体系结构中要大量地使用 SOAP 消息，所以，必须要有某种方法来在多种不同的系统中寻址和传送 SOAP 消息。

- WS-Routing 是一个简单的、无状态的 SOAP 扩展，它以异步的方式在多种通信传输层(比如 TCP、UDP 和 HTTP)上发送 SOAP 消息。WS-Routing 提供了相应的寻址机制，允许为消息指定完全的消息路径(包括消息的返回路径)，所以能实现单向传送消息、双向传送消息(比如请求/响应方式)、对等会话以及长时间的会话。
- WS-Referral 是一个简单的 SOAP 扩展，它允许动态配置消息路径中 SOAP 节点之间的路由。这个配置协议允许 SOAP 节点有效地把它的处理任务部分或全部委派给其他的节点。

上面 4 个规范都是模块化的，它们可以组合在一起使用。例如，WS-Security 可以对一个使用 WS-Routing 头的 SOAP 消息进行数字签名。其中每一个规范都提供了可扩充的和可组合的机制以便允许将来的全局 XML Web 服务体系结构规范能够合并成一个完整的解决方案。

把多个组织的应用集成在一起可能会导致包括从传送错误到不兼容错误在内的多种错误。可靠的消息传送和事务允许 XML Web 服务的建立者控制错误的程度和最终的影响。

1.7 小　结

本章简单地介绍了一些 Web 服务的基本概念及其他相关的概念，包括 UDDI、WSDL、SOAP 等。通过上面的介绍，用户可以看出，Web 服务是一种在 Web 环境中使用的组件开发技术，它建立在标准的 Internet 协议(HTTP 和 XML 等)之上，用户可以通过 SOAP、HTTP GET/POST 等方式来访问它。另外，Web 服务还是一个不断完善的技术，与它相关的规范还在不断地推出。

接下来的几章将依次详细介绍与 Web 服务相关的一些基本概念，包括 XML 规范、HTTP 协议、SOAP 协议、UDDI 规范和 WSDL 规范等。

第2章　创建和使用Web服务

Visual Studio.NET 对创建和使用 Web 服务提供了充分的支持。在这一章里，我们主要介绍如何使用 Visual Studio.NET 来创建 Web 服务及如何使用 Web 服务的客户程序。

为了成功地创建本章中的 Web 服务项目，需要用户在开发机器上安装 IIS(Internet Information Server)5.0 以上版本和.NET Framework SDK，或者能够连接到安装有相应软件的服务器上。

2.1　创建 Web 服务

Visual Studio.NET 为创建 Web 服务提供了现成的模板，因此，Web 服务的创建过程非常简单。在 Visual Studio.NET 中创建 Web 服务主要使用 ASP.NET 应用程序框架，请使用以下的步骤创建 Web 服务：

(1) 选择【文件】|【新建】|【项目】命令，打开【新建项目】对话框，如图 2-1 所示。

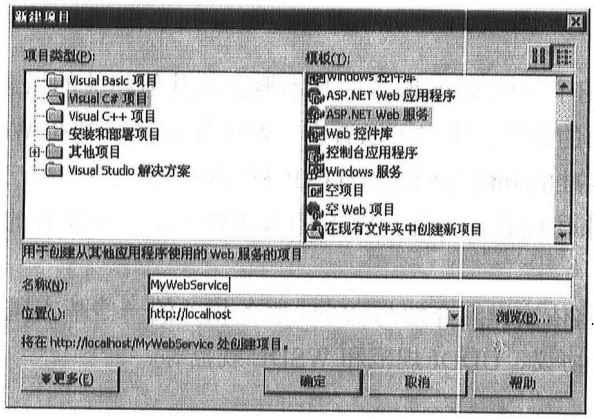

图 2-1 【新建项目】对话框

(2) 在【新建项目】对话框左边的【项目类型】列表框中选择【Visual C#项目】，然后在对话框右边的【模板】列表框中选择【ASP.NET Web 服务】模板。

(3) 在【新建项目】对话框下边的【名称】文本框中输入新项目的名称，例如 MyWebService，并在【位置】下拉列表框中输入保存 Web 服务项目的 IIS 服务器的地址(本例中使用的是 http://localhost，即开发机器和 IIS 服务器是同一台机器)，或者通过【浏览】按钮选择一个地址。