开发专家 EXPERT

飞思科技 FECIT Sci-Tech www.fecit.net

Web Services

Web Services Web Services Web Services Web Services Web Services

原理与开发实务

Web Services Web Services Web Services Web Services

林弘之 编著
飞思科技产品研发中心 改编

随书附赠的光盘
内容为书中范例
源文件

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

开发专家
EXPERT

Web Services

Web Services Web Services Web Services Web Services Web Services

原理与开发实务

Web Services Web Services Web Services Web Services

林弘之 编著
飞思科技产品研发中心 改编

本书配有光盘，需要的读者请到多媒体阅览室（新馆 301 室）联系。

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内容简介

本书以中国台湾地区畅销书《Web Services 实作——使用 SOAP ToolKit 与 Visual Studio.NET》为蓝本改编。Web 服务主要使用 SOAP 协议进行客户与服务器之间的通信，因而得到了广泛的发展。本书针对这一越来越广泛的应用，深入浅出地介绍了 Web 服务的原理及其在各种开发工具下的实现。使读者在了解了 Web 服务后，无论使用的开发工具是什么，都可以容易地在本书中找到类似实例，更快地掌握 Web 服务开发技巧。随书附赠的光盘内容为书中范例源文件。

本书适合中高级程序开发人员阅读使用。

版权贸易合同登记号：01-2003-4812

图书在版编目（CIP）数据

Web Services 原理与开发实务 / 林弘之编著；飞思科技产品研发中心改编. —北京：电子工业出版社，2003.11

ISBN 7-5053-9158-5

Ⅰ.W... Ⅱ.①林...②飞... Ⅲ.互联网络－网络服务器 Ⅳ.TP368.5

中国版本图书馆 CIP 数据核字（2003）第 082938 号

责任编辑：王树伟　杨　鸽

印　　刷：北京大中印刷厂

出版发行：电子工业出版社　　http://www.phei.com.cn

北京海淀区万寿路 173 信箱　邮编：100036

经　　销：各地新华书店

开　　本：787×1092　1/16　印张：21.75　字数：556.8 千字　附光盘 1 张

版　　次：2003 年 11 月第 1 版　　2003 年 11 月第 1 次印刷

印　　数：5 000 册　　定价：35.00 元（含光盘）

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077

出版说明

Web Services

“开发专家”是电子工业出版社计算机研发部长期以来精心培育的计算机科学技术类本版品牌。这个品牌是由多个专题系列组成的横向大系列，涵盖了计算机技术的各个方面，特别是一直受到极大关注的程序开发类系列，例如《开发专家之数据库》、《开发专家之网络编程》、《开发专家之 Delphi》、《开发专家之 Sun ONE》、《开发专家之 Oracle》和《嵌入式开发专家》等。这些专题系列基于各自的角度，从纵向上包含了该专题的所有内容。因此，整个“开发专家”的品牌架构纵横交错，囊括了所有的计算机技术和所有的技术层面，海纳百川而又极具可扩展性。

“开发专家”的作者队伍主要依托于“飞思科技产品研发中心”。“飞思科技产品研发中心”由专业的策划人员、权威的技术专家和资深的作者队伍共同组成。在图书的出版上，形成了以研发为基础、以出版为中心、以服务为支持的专业化出版框架和流程。通过深入的市场调查和技术跟踪，在综合了技术需求和读者焦点等因素的基础上，形成各系列丛书的写作重点和大纲，然后聘请业界的最前沿学者进行写作。同时，策划工作全程介入写作进程，严格控制写作质量，用最专业的技术背景、最深刻的理论基础、最具代表性的案例、最能为专业读者接受的形式，为读者提供品质最佳的图书产品，体现了出版者和著作者的完美结合。

多年来，我们始终把创造社会效益摆在首位，秉承一切为国内计算机技术专业读者服务的精神，为推动国内信息技术的发展、为体现国内技术的原创水平，穷尽所有的创意与努力，将出版者的命运与读者的支持紧紧地连在了一起。

在此，我们临出版之残酷竞争而不惧，旌旗猎猎而异军突起，这与广大读者的支持是分不开的。为使我们的脚步更坚实，使我们的队伍永远保持活力和创造力，我们期待着您能为我们的前进贡献出您的意见和建议。同时，我们也在等待着您的加入。

我们的联系方式：

咨询电话：（010）68134545　　68131648

答疑邮件：support@fecit.com.cn

网　　址：http://www.fecit.com.cn　　http://www.fecit.net

答　　疑：http://www.fecit.com.cn 的“问题解答”专区

通用网址：计算机图书、FECIT、飞思教育、飞思科技、飞思

电子工业出版社计算机研发部

前言

Web Services

随着计算机网络的发展，分布式应用（分布式计算与分布式数据访问）也越来越受到广泛青睐。通常我们说网络资源博大精深，只是简单地指网络上的信息量大，但一个明显的事实是，网络不仅仅给我们提供信息，更可以为我们提供服务。

分布式应用的结构已经从两层结构发展到多层结构，所使用的协议有 CORBA（通用对象请求代理机制）、DCOM（分布式组件对象模型）、RMI（远程方法接口）。虽然这些协议都与操作系统无关，但是还是有一个缺陷，就是使用 CORBA 协议的服务器，访问的客户端也必须是支持 CORBA 协议的访问式，而不能是仅支持 DCOM 协议的客户端。这样对用户来说，要想取得远程的服务，还得先清楚对方使用的协议，因而产生了很大的不便。这时候 SOAP（简单对象访问协议）出现了，它直接被 IE 支持，因而解决了客户端与服务器匹配的问题。

Web 服务主要使用 SOAP 协议进行客户与服务器之间的通信，因而得到了广泛的发展。本书针对这一越来越广泛的应用，深入浅出地介绍了 Web 服务的原理及在各种开发工具下的实现。使得读者在了解了 Web 服务后，无论使用什么开发工具，都可以容易地在本书中找到类似实例，更快地掌握 Web 服务开发技巧。

本书由电子工业出版社计算机研发部暨飞思科技产品研发中心策划并组织改编，原著作者是中国台湾林弘之先生。在改编的过程中，我们并没有一味地照搬原书的内容，而是根据客观现实的发展，对部分开发工具软件的版本进行了调整。原书成书于 2002 年 7 月，到现在，部分开发工具软件的版本已有了更新，如出现了 Delphi 7，开发 Web 服务的工具有了 MS SOAP Toolkit 3.0 等。本书根据最新的软件版本对原书的内容进行了补充与修改，使内容适合于现今流行的软件版本，比较容易为广大读者所接受。

本书主要由陆正武改编，陆正中负责审稿。参与改编及程序调试工作的还有张勇、安冀苗、潘明、陈敏、杨枭。另外，梅成刚、石正贵、马进德、李净等人也参与了讨论，并给出了不少有益的意见。由于时间仓促，作者水平有限，书中难免有不妥之处，欢迎广大读者提出宝贵意见。

我们的联系方式：

咨询电话：（010）68134545　　68131648

答疑邮件：support@fecit.com.cn

网　　址：http://www.fecit.com.cn　　http://www.fecit.net

答　　疑：http://www.fecit.com.cn 的“问题解答”专区

通用网址：计算机图书、FECIT、飞思教育、飞思科技、飞思

飞思科技产品研发中心

目 录

Web Services

基 础 篇

软件服务开发篇

案例研究篇

Web Services

基础篇

第 1 章　软件服务的实现：Web Services

本章主要内容如下：

- 软件服务的构想
- 以 XML 为基础的 SOAP 消息交换
- 用 WSDL 做服务描述
- 用 UDDI 做服务索引
- 软件服务的未来

1.1　软件服务的构想

硬件的发展速度一日千里，我们因此享受到了很多便利，比如，通信迅捷、便宜但运算速度很快的计算机，轻薄短小的手持式设备（如 PDA），具有运算与图形处理能力的无线通信设备等。

虽然设备的体积越来越小，功能越来越强大，但是在使用这些设备时，总是会感到缺少了点什么。这些缺少的东西，就是内容（Content）。

用手机可以听 MP3，可以上网取得实时信息，这些已经不是新闻。虽然这些信息已经可以称为 Content，但是这些 Content 之间却缺少集成，以至于想要汇总许多信息时，必须分别进入每一个信息服务的站点。

Web Services 的开发有一个很重要的概念：软件组件的制作流程也能像计算机零件组装一样，分工成 IC 设计、IC 制造、IC 封装测试、主机板系统集成，每个阶段的产品各自由不同厂商去研发与创新，最后将系统集成后再提供给用户一个整体的服务。

在组件组装的概念之下，若将每一个软件服务组件视为一个软件 IC，软件组件的设计与开发就是软件产业的上游，而组合许多软件组件成为某一个特定领域服务的系统集成厂商，就是软件产业的中下游。软件服务趋势将有可能渐渐地发展成一个兴盛的新产业。

Web Services 中 UDDI 规格的最高目标就是集成可能分别由不同厂商所开发的软件组件，根据 End Users 的个人需求，包装相关服务给用户。而在这个人人几乎都会上网的时代，网络环境的成熟足以为 Web Services 提供一个开放的服务平台。也许正如微软某人所预言的，未来取得软件服务，将会有如打开水龙头就有自来水那么简单。

为了实现软件服务的架构，Web Services 相关的标准与协议在近几年来陆续被制定与推广，众多厂商也提供了开发 Web Services 的工具与快速开发环境。这些标准与协议，大部分都是以 XML（eXtensible Markup Language）为发展基础。这其中最主要的标准与协议包括：

- SOAP（Simple Object Access Protocol）：定义点对点远程服务访问协议。
- WSDL（Web Services Description Language）：定义互联网络服务的描述。

- UDDI（Universal Description, Discovery and Integration）：定义互联网络服务的注册与查找机制。

在 UDDI（www.uddi.org）的技术白皮书中，将这几个协议与标准的关系界定成以下几个网络服务层，如表 1-1 所示。

表 1-1

软件服务互通协议（尚未定义）
服务描述，检索与整合（UDDI）
简单对象访问协议（SOAP）
可扩展置标语言（XML）
互联网络协议（HTTP、TCP/IP）

在表 1-1 中，UDDI 这一层负责提供 Web Services 的注册与检索，甚至提供互相串联集成的环境。不过 UDDI 的服务必须建构在底层的标准协议如 SOAP、XML、HTTP 之上。在接下来的几节中，将会对各层的组合要件一一进行介绍。

1.2 以 XML 为基础的 SOAP 消息交换

Web Services 的服务方式是由用户前端（客户端可能是任何应用程序包括浏览器），将客户端的服务请求（Request）封装成 SOAP（Simple Object Access Protocol）的消息格式，通过 HTTP、SMTP 等通信协议传送到服务器端。

若以采用 HTTP 通信协议的网络服务为例，网络服务的用户只要通过 HTTP 就可以访问建构在互联网络服务器（Web Server）上的网络服务（Web Services）。如此简单且一致的访问方式，简化了以往 Client-Server 访问服务的烦琐步骤与设置，并以通用的 XML 格式作为信息传递的包的基础，增添了建构在异构系统的服务透明性。

如果软件服务都可以建构在互联网络上，并采用一致的访问协议，将可促成各种服务间可以串联集成的能力。而这个用来访问网络服务的一致性协议就是 SOAP（Simple Object Access Protocol），所有的 SOAP 消息包都以 XML 的格式为基础，由此可见 XML 在 Web Services 中的重要性。

SOAP，英文全称“Simple Object Access Protocol”，顾名思义就是一种可以用比较简便的访问方式来取得服务的通信协议。通过 SOAP 可以在分布式的环境中互换消息，譬如互联网络中的每一个参与者互换消息，也就是可以从他处取得服务，或是选择自己就是这个环境中的一个软件服务的提供者。

虽然 SOAP 一开始是由 IBM 与 Microsoft 草拟制定的，不过目前 SOAP 已经成为 W3C（World Wide Web Consortium）的一个 Note，目前的版本为 1.1 版，可以通过 http://www.w3.org/TR/SOAP/这个网址找到 SOAP 的规格书。

SOAP 可以与其他通信协议例如 HTTP、SMTP 等结合。SOAP 也提供了一个简单且较为便利的机制，使得互联网络中各个端点间可以互相交换结构型与非数据类型的信息流，而这一切都是基于 XML 的消息交换。

SOAP 本身并没有直接定义或包含任何跟应用层面相关的语法，而是通过定义一个简单的机制将应用程序的服务意义模块化，并且将取得服务的方式与传输的数据都一并包装在 SOAP 的包之中。如此可让 SOAP 拥有更大的可伸缩性空间，这样可以在更多消息交换平台与异构系统之间顺利运行。

1.2.1　SOAP 的组成部分

SOAP 是一个建构在 XML 之上的协议，包含了以下三个主要部分：

- Envelope 消息包：通过建立消息包的架构，描述 SOAP 消息包的组成，以及决定由谁来处理消息。
- Encoding 编码规则：定义数据串行化（Serialization）的机制、消息包处理方式，以及对于应用程序所定义的数据类型与消息包的编码与交换规则。
- RPC 机制：处理远程过程调用（Remote Procedure Call，RPC）的相关机制等。

这三个部分在功能上彼此独立，所以可以分别定义。因此处理消息包与定义编码规则时，各自有相对应的名字空间（Name Space）来定义这些方法，为的就是通过名字空间做到各自功能模块化。

以 SOAP 为通信协议的 Web Services，在其传递的 SOAP 包，也就是以 XML 为基础的消息内容中的每一个元素（Element）与属性（Attribute）都必须包含适当的 SOAP 的名字空间（name space）的信息，而 SOAP 的应用程序，无论是客户端或是服务器端的程序，都要有能力处理所接收到的含有 SOAP 名字空间的 XML 消息包。这两个名字空间分别是：

- SOAP 包的名字空间：

"http://schemas.xmlsoap.org/soap/envelope/"

- SOAP 消息编码与串行化的名字空间（Name Space）：

"http://schemas.xmlsoap.org/soap/encoding/"

虽然 SOAP 消息本身即是 XML，但是在定义上 SOAP 消息却不能包含 XML 的文件类型声明（DTD，Document Type Declaration），也不能包含任何处理脚本（Processing Instructions）。SOAP 对消息内部每一个编码的元素（Element）都设置了一个惟一的“id”属性值。SOAP 也可以使用“href”属性值，类型为“uri-reference”，用来指定引用到资源所需的地址信息。

1.2.2　SOAP 消息包

现在我们来看看 SOAP 消息包的格式。在下面的例子中，一个 SOAP 的 HelloWorld 服务请求（Request）传送到网络服务的服务器端。这个服务请求只有一个输入参数，这个参数是一个字符串。服务器端会将此字符串包装在 SOAP 的响应消息中返回调用端。在整个 XML 文件的最上层是 SOAP 的包元素 Envelope，表示在 <soap:Envelope...>...</soap:Envelope...>中所包含的即是所要传送的 SOAP 消息。

这里的 XML 的名字空间采用 SOAP 标识符命名而不用与应用程序相关的标识符，是为了达到与应用程序独立的目的。值得注意的是，在这个消息包中，并没有特别指明此 SOAP 消息文件必须以 HTTP 来传送。

SOAP 的消息包含在 HTTP 的服务请求（Request）中，在此消息包中，可以看到所调用的函数名称（HelloWorld）、输入参数（字符串："Hello World!"）等。代码如下：

```
POST /HelloWorld HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://test.com/HelloWorld/HelloWorld"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:
soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <m:HelloWorld xmlns="http://test.com/HelloWorld/"/>
<msg>Hello World!</msg>
</m:HelloWorld>
  </soap:Body>
</soap:Envelope>
```

下面是服务器端响应的消息，包括 Web Services 处理的结果（返回字符串“Hello World!”），也都包含在 HTTP 的响应（Response）中。代码如下：

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:
soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <HelloWorldResponse xmlns="http://test.com/HelloWorld/">
      <HelloWorldResult>Hello World!</HelloWorldResult>
    </HelloWorldResponse>
  </soap:Body>
</soap:Envelope>
```

1.3　用 WSDL 做服务描述

WSDL 文件（Web Services Description Language），也是符合 XML 格式的文件。WSDL 的主要目的是描述互联网络服务器端所提供的软件服务。使用 WSDL 标准可以为服务器端提供的服务建立一份服务描述文件。

在服务描述文件中除了可用于识别所提供的服务，还描述了每个服务可提供的操作方法或称服务函数（Operation）。对于每一个操作方法而言，WSDL 还描述了客户端要求这些服务时，必须遵循的事项及传输数据的格式等。

一个服务的描述（WSDL 文件）就好像是客户端和服务器端之间的协议，因为客户端和服务器端在初始化服务时都需要 WSDL 文件。当客户端根据 WSDL 送出合约格式的 SOAP 消息服务请求时，服务器端必须提供 WSDL 中要求的服务项目给客户端。

在我们的 HelloWorld 例子中，假设有一个 HelloWorld.wsdl 定义了一个 Web 服务为：

```
HelloWorld:
...
<service name='HelloWorld' >
    <port name='HelloWorldServerSoapPort' binding='wsdlns:
HelloWorldServerSoapBinding' >
      <soap:address location='http://localhost:8080/
WebServices/HelloWorld/Service/Rpc/VbSrv/HelloWorld.WSDL' />
    </port>
</service>
...
```

这个服务中有一个服务函数（operation）也叫做 HelloWorld。

```
<portType name='HelloWorldServerSoapPort'>
  <operation name='HelloWorld' parameterOrder='msg'>
    <input message='wsdlns:HelloWorldServer.HelloWorld' />
    <output message='wsdlns:HelloWorldServer.
HelloWorldResponse' />
  </operation>
</portType>
```

这个服务描述文件位于服务器端，当有客户端想要向 Server 取得服务时，必须先从 Server 取得一份服务的 WSDL，客户端用 WSDL 中关于服务的描述来建立 SOAP 的服务请求（Request），然后客户端将服务请求传送到服务器端请求服务，服务器端执行相对的服务函数 HelloWorld，并将结果也就是“Hello World!”字符串以 SOAP 的消息包的方式返回客户端，如图 1-1 所示。

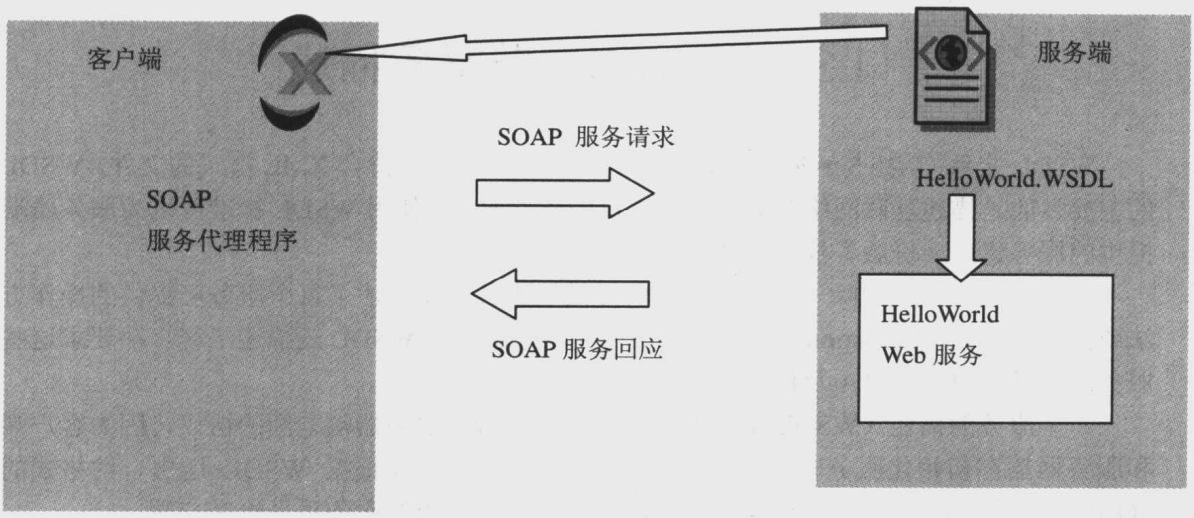

图 1-1

HelloWorld.wsdl 程序代码如下：

```
<?xml version='1.0' encoding='UTF-8' ?>
 <!-- Generated 03/19/02 by Microsoft SOAP Toolkit WSDL File
 Generator, Version 1.00.623.1 -->
<definitions    name ='HelloWorld'     targetNamespace = 'http://
tempuri.org/wsdl/'
             xmlns:wsdlns='http://tempuri.org/wsdl/'
             xmlns:typens='http://tempuri.org/type'
             xmlns:soap='http://schemas.xmlsoap.org/wsdl/soap/'
             xmlns:xsd='http://www.w3.org/2001/XMLSchema'
             xmlns:stk='http://schemas.microsoft.com/soap-
toolkit/wsdl-extension'
             xmlns='http://schemas.xmlsoap.org/wsdl/'>
  <types>
    <schema targetNamespace='http://tempuri.org/type'
      xmlns='http://www.w3.org/2001/XMLSchema'
      xmlns:SOAP-ENC='http://schemas.xmlsoap.org/soap/
encoding/'
      xmlns:wsdl='http://schemas.xmlsoap.org/wsdl/'
      elementFormDefault='qualified'>
    </schema>
  </types>
  <message name='HelloWorldServer.HelloWorld'>
        <part name='msg' type='xsd:string'/>
  </message>
  <message name='HelloWorldServer.HelloWorldResponse'>
    <part name='Result' type='xsd:string'/>
  </message>
```

```
  <portType name='HelloWorldServerSoapPort'>
    <operation name='HelloWorld' parameterOrder='msg'>
      <input message='wsdlns:HelloWorldServer.HelloWorld' />
      <output message='wsdlns:HelloWorldServer.
HelloWorldResponse' />
    </operation>
  </portType>
  <binding name='HelloWorldServerSoapBinding' type='wsdlns:
HelloWorldServerSoapPort' >
    <stk:binding preferredEncoding='UTF-8'/>
    <soap:binding style='rpc' transport='http://schemas.
xmlsoap.org/soap/http' />
    <operation name='HelloWorld' >
      <soap:operation soapAction='http://tempuri.org/action/
HelloWorldServer.HelloWorld' />
      <input>
        <soap:body use='encoded' namespace='http://tempuri.
org/message/'
                   encodingStyle='http://schemas.xmlsoap.org/
soap/encoding/' />
      </input>
      <output>
        <soap:body use='encoded' namespace='http://tempuri.
org/message/'
                   encodingStyle='http://schemas.xmlsoap.org/
soap/encoding/' />
      </output>
    </operation>
  </binding>
  <service name='HelloWorld' >
    <port name='HelloWorldServerSoapPort' binding='wsdlns:
HelloWorldServerSoapBinding' >
      <soap:address location='http://localhost:8080/
WebServices/HelloWorld/Service/Rpc/VbSrv/HelloWorld.WSDL' />
    </port>
  </service>
</definitions>
```

请注意，除非指定其他的名字空间，否则“wsdl”是 WSDL 文件中所有元素默认使用的名字空间（Name Space）。