WebService知识点

# WebService是什么

WebService是一种跨编程语言和跨操作系统平台的远程调用技术。

从表面上看，WebService就是一个应用程序向外界暴露出一个能通过Web进行调用的API，也就是说能用编程的方法通过Web来调用这个应用程序。把调用这个WebService的程序叫做客户端，把提供这个WebService的程序叫做服务端。

从深层次看，WebService是建立可互操作的分布式应用程序的新平台，是一个新标准。它定义了应用程序如何在Web上实现互操作性。可以使用任何语言，在任何平台上写WebService，可以通过WebService标准对这些服务进行调用和访问。

# WebService平台技术

XML+XSD、SOAP和WSDL就是构成WebService平台的三大技术。

## XML+XSD

WebService采用HTTP协议传输数据，采用XML格式封装数据（即XML中说明远程调用服务对象的哪个方法，传递的参数是什么，以及服务对象的返回结果是什么）。

XML解决了数据表示的问题，但它没有定义一套标准的数据类型，更没有说怎么去扩展这套数据类型。例如，整形数代表了什么？16位、32位还是64位？这些细节对实现互操作性很重要。

XML Schema(XSD)是专门解决这个问题的一个标准。它定义了一套标准的数据类型，并给出了一种语言来扩展这套数据类型。WebService平台就是用XSD来作为其数据类型系统。当你用某种语言来构造WebService时，为了符合WebService标准，所有使用的数据类型都必须转换为XSD类型。你用的工具可能已经帮你完成了这个转换，但很可能会根据需要修改一下转换过程。

## SOAP

WebService通过HTTP协议发送请求和接收结果时，发送的请求内容和结果内容都采用XML格式封装，并增加了一些特定的HTTP消息头，以说明HTTP消息的内容格式。这些特定的HTTP消息格式和XML内容格式就是SOAP协议。SOAP提供了标准的RPC方法来调用WebService。

SOAP协议 = HTTP协议 + XML数据格式。

SOAP协议定义了SOAP消息的格式，SOAP协议是基于HTTP协议的，SOAP也是基于XML和XSD的，XML是SOAP的数据编码方式。

## WSDL

WebService客户端要调用一个WebService服务，首先要知道这个服务的地址在哪，以及这个服务有什么方法可以调用。所以，WebService服务器端首先要通过一个WSDL文件来说明有什么服务可以对外调用，服务是什么（服务中有哪些方法，方法接受的参数是什么，返回值是什么），服务的网络地址用哪个url地址表示，服务通过什么方式来调用。

WSDL（WebServices Description Language）是一个基于XML的语言，用于描述WebService及其函数、参数和返回值。它是WebService客户端和服务端都能理解的标准格式。

WSDL文件保存在Web服务器上，通过一个url地址就可以访问到它。客户端在调用一个WebService服务端之前，要知道该服务的WSDL文件的地址。WebService服务提供上可以通过两种方式来暴露它的WSDL文件的地址：

* 注册到UDDI服务器，以便被人查找；
* 直接告诉客户端调用者。

# WebService开发

WebService开发可以分为服务器端开发和客户端开发两个方面

## 服务端开发

把公司内部系统的业务方法发布成WebService服务，供远程合作单位和个人调用。(借助一些WebService框架可以很轻松地把自己的业务对象发布成WebService服务，Java方面的典型WebService框架包括：axis，xfire，cxf 等，java ee服务器通常也支持发布WebService服务，例如JBoss。)

## 客户端开发

调用别人发布的WebService服务，大多数人从事的开发都属于这个方面，例如，调用天气预报WebService服务。（使用厂 商的WSDL2Java之类的工具生成静态调用的代理类代码；使用厂商提供的客户端编程API类；使用SUN公司早期标准的jax-rpc开发包；使用 SUN公司最新标准的jax-ws开发包。当然SUN已被ORACLE收购)

## WebService的工作调用原理

对客户端而言，我们给这各类WebService客户端API传递WSDL文件的url地址，这些API就会创建出底层的代理类。调用这些代理就可以访问WebService服务。代理类把客户端的方法调用编程soap格式的请求数据再通过HTTP协议发出去，并把接收到的SOAP数据变成返回值返回。

对服务端而言，各类WebService框架的本质就是一个大的Servlet，当远程调用客户端给它通过HTTP协议发送过来的SOAP格式的请求数据时，它分析这个数据，就知道调用哪个类的哪个方法。于是就查找和创建这个对象，并调用其方法，再把方法返回的结果包装成SOAP格式的数据，再通过HTTP响应信息回到客户端。

# 适用场合

## 跨防火墙通信

如果应用程序有成千上万的用户，而且分布在世界各地，那么客户端和服务器之间的通信将是一个棘手的问题。因为客户端和服务器之间通常会有防火墙或者代理服 务器。在这种情况下，使用DCOM就不是那么简单，通常也不便于把客户端程序发布到数量如此庞大的每一个用户手中。传统的做法是，选择用浏览器作为客户 端，写下一大堆ASP页面，把应用程序的中间层暴露给最终用户。这样做的结果是开发难度大，程序很难维护。如果中间层组件换成WebService的话， 就可以从用户界面直接调用中间层组件。从大多数人的经验来看，在一个用户界面和中间层有较多交互的应用程序中，使用WebService这种结构，可以节 省花在用户界面编程上20%的开发时间。

## 应用程序集成

企业级的应用程序开发者都知道，企业里经常都要把用不同语言写成的、在不同平台上运行的各种程序集成起来，而这种集成将花费很大的开发力量。应用程序经常 需要从运行在IBM主机上的程序中获取数据；或者把数据发送到主机或UNIX应用程序中去。即使在同一个平台上，不同软件厂商生产的各种软件也常常需要集 成起来。通过WebService，可以很容易的集成不同结构的应用程序。

## B2B集成

用WebService集成应用程序，可以使公司内部的商务处理更加自动化。但当交易跨越供应商和客户、突破公司的界限时会怎么样呢？跨公司的商务交易集成通常叫做B2B集成。WebService是B2B集成成功的关键。通过WebService，公司可以把关键的商务应用“暴露”给指定的供应商和客户。例如，把电子下单系统和电子发票系统“暴露”出来，客户就可以以电子的方式发送订单，供应商则可以以电子的方式发送原料采购发票。当然，这并不是一个 新的概念，EDI(电子文档交换)早就是这样了。但是，WebService的实现要比EDI简单得多，而且WebService运行在Internet 上，在世界任何地方都可轻易实现，其运行成本就相对较低。不过，WebService并不像EDI那样，是文档交换或B2B集成的完整解决方案。 WebService只是B2B集成的一个关键部分，还需要许多其它的部分才能实现集成。

用WebService来实现B2B集成的最大好处在于可以轻易实现互操作性。只要把商务逻辑“暴露”出来，成为WebService，就可以让任何指定 的合作伙伴调用这些商务逻辑，而不管他们的系统在什么平台上运行，使用什么开发语言。这样就大大减少了花在B2B集成上的时间和成本，让许多原本无法承受 EDI的中小企业也能实现B2B集成。

## 软件和数据重用

软件重用是一个很大的主题，重用的形式很多，重用的程度有大有小。最基本的形式是源代码模块或者类一级的重用，一种形式是二进制形式的组件重用。采用 WebService应用程序可以用标准的方法把功能和数据“暴露”出来，供其它应用程序使用，达到业务级重用。

# 不适用场合

## 单机应用程序

目前，企业和个人还使用着很多桌面应用程序。其中一些只需要与本机上的其它程序通信。在这种情况下，最好就不要用WebService，只要用本地的 API就可以了。COM非常适合于在这种情况下工作，因为它既小又快。运行在同一台服务器上的服务器软件也是这样。最好直接用COM或其它本地的API来 进行应用程序间的调用。当然WebService也能用在这些场合，但那样不仅消耗太大，而且不会带来任何好处。

## 局域网的同构应用程序

在许多应用中，所有的程序都是用VB或VC开发的，都在Windows平台下使用COM，都运行在同一个局域网上。例如，有两个服务器应用程序需要相互通 信，或者有一个Win32或WinForm的客户程序要连接局域网上另一个服务器的程序。在这些程序里，使用DCOM会比SOAP/HTTP有效得多。与 此相类似，如果一个.NET程序要连接到局域网上的另一个.NET程序，应该使用.NETremoting。有趣的是，在.NETremoting 中，也可以指定使用SOAP/HTTP来进行WebService调用。不过最好还是直接通过TCP进行RPC调用，那样会有效得多。