基于SOAP的应用集成方案

杨阳　杨宗凯　(武汉 华中科技大学电信系 430074)

摘要：简单对象访问协议 (简称SOAP) 提供了在无中心分布式的环境中对等实体间使用XML交换结构化有类型数据的简单、轻量的机制，简单、扩展方便的特点使其可以适用于多种系统，在应用集成方面有着独特的优势。本文介绍基于SOAP的应用集成方案，并具体介绍其实现过程。

关键词：SOAP　应用集成　HTTP XML　(Application Integration)

1　SOAP概述

SOAP是简单对象访问协议 (Simple Object Access Protocol) 的简写，它是一个在分散的，分布式环境中交换结构化、有类型数据的简单、轻量级的协议。它把成熟的基于HTTP的Web技术和XML的灵活性、可扩展性结合在一起，有助于实现多种异构平台之间的相互访问，从而使存在的应用能够被更广泛的用户访问。

SOAP是第一个没有发明任何新技术的技术，它的两个基础HTTP和XML都是已经广泛应用的协议和标准，HTTP用于实现SOAP的RPC风格的传输，而XML是SOAP的编码模式。在一个普通的HTTP服务器 (如Apache、IIS等) 上加载一个XML解析包和一个SOAP模块，就可以方便的构造一个SOAP服务器。传统的HTTP服务器的广泛应用对于SOAP的推广十分有益，并且，HTTP服务器只是处理SOAP请求的一种方式，对于建立SOAP应用是充分的，但不是必要的。

SOAP协议1.1版于2000年5月被Ariba、Compag、IBM、Microsoft等多家著名IT企业共同推荐给国际组织W3C，之后迅速的得到了业界的普遍支持，成为许多上层应用协议的基础，应用前景十分看好。目前，SOAP协议发展到了1.2版 (W3C于2001年7月发布)。

2　问题的提出

应用集成是大型计算机系统开发中非常关键的技术，随着计算机系统越来越广泛的应用，集成技术的重要性日益显得突出。过去，开发人员一直通过集成本地系统服务来构建应用系统。在新的互联网时代，开发人员逐渐致力于结构复杂的N层系统，通过网络的分布式集成技术成为人们研究的热点。

分布式的应用系统建立需要使用分布式的组件对象模型，如目前应用比较广泛的Microsoft公司的DCOM和对象管理组织 (OMG) 提出的CORBA，DCOM和CORBA作为比较成熟的体系，已各自占据了相当大的市场。但是，这两种系统使用了不同的数据表达方式，互不兼容。并且，它们有一个共同的缺陷，那就是无法扩展到互联网上。它们要求通信两端有同类基本结构，紧密耦合，一旦一方的执行机制发生变化，那么另一方就会崩溃；它们穿透互联网上各种各样的防火墙的能力不能令人满意。

而基于网络的集成方案通常需要是松耦合、跨平台、语言无关、与特定服务无关的，而SOAP恰恰能做到这一点。SOAP所使用的HTTP协议在Internet应用广泛，绝大多数防火墙都不限制HTTP协议，并且，它还是一个相当有用的RPC协议，它提供了组帧、连接管理、序列化对象等功能的支持，仅缺少一种标准格式来表达一个RPC调用，而XML正好补充了HTTP在这方面的不足。XML是一种中立的数据表达方式——XDR (XML Data Representation)，它格式灵活，易于扩展，基于文本，可以在任何平台

上被很容易的处理。

SOAP的主要制定者之一Microsoft公司已在DCOM中对SOAP提供了较好的支持，对象管理组织也已提出基于SOAP的CORBA的建议，这对SOAP的推广使用提供了有力的支持。

3 基于SOAP的应用集成方案

基于SOAP的应用集成方案，是将企业已有的各个分散的、独立的系统用SOAP连接起来，组成一个有机的整体，完成更多的功能，以实现企业自动化、商务流程的整合或B2B电子商务，达到使原有系统增值的目的，而又无需对原有系统做大的改动。在这个方案里面，SOAP是底层的通信方式，它的作用是作为消息的载体，实现分布式对象访问和应用系统之间的调用。

企业原有的各个系统之间相互独立，接口互不统一，为达到集成的目的，首先要将它们按照SOAP的标准进行包装，发布成可供其他系统、用户访问的服务。一个SOAP服务可以由一个类来实现，也可以是一段封装的脚本代码，一个服务可以包含一个或多个方法。根据SOAP服务的入口地址、服务名、方法名，提交方法所需的参数，即可进行一次SOAP通信，并得到调用返回的结果。在用户看来，访问远程对象就像使用本地对象一样简单、方便。

服务发布之后，还需要以XML文档的格式对服务进行描述，并将服务的描述存放在注册库(Registry & Repository)中。这样做的好处是，其他系统可以通过读取、分析服务描述文档进行对服务的访问，而无需在程序中预先写定，使得服务接口的更新、扩展非常方便。

在原有系统之外，需要开发一个核心模块来实现对原有系统的调度，我们称这个核心模块为应用集成系统(Application Integration System，简称AIS)。在集成的时候，利用开发工具或手工制定出各系统之间的业务关系、调用规范，即工作流文档，然后将此文档存放在注册库中，以便AIS运行时动态加载该文档。这样，开发出来的应用系统就可以通过SOAP来有次序的调用指定的网络服务，将各个分散的、独立的子系统有机的结合起来，构成一个完整的、可完成复杂功能的系统。

企业内部集成完成之后，即可发布对外的服务，使外部企业的系统能够访问本企业的服务，进行B2B交易。

总的说来，基于SOAP的应用集成方案由以下步骤构成：

(1) 对原有系统的接口进行包装，以服务的形式发布，使其他系统可以通过SOAP进行调用；

(2) 将各系统发布的服务进行描述和注册，以便集成系统能够发现和访问它们的服务；

(3) 制定工作流描述文档，程序对此文档进行动态加载、解析、执行，实现商务流程自动化；

(4) 企业发布对外服务，实现B2B交易。

基于SOAP的应用集成方案有以下特点：

① 松耦合。XML是一种基于文本的编码方式，任何平台都可以方便的编码和解码；参与集成的各方可以使用不同的平台和编程语言；任何一方可更改其执行机制，而不影响整个系统的正常运行。

② 易扩展。可以比较容易的做到增加或减少参与集成的子系统。

③ 基于Internet。大多数防火墙都不限制HTTP协议，这使系统可方便的应用于Internet上。

基于SOAP的集成方案中，参与集成的各方的接口以及企业对外发布的接口都是以"服务"的形式提供的，这就需要服务的描述和服务的发现机制。在高级的应用中，可以使用WSDL (Web服务描述语言) 来实现对Web服务的描述，使用UDDI机制 (统一描述、发现、集成) 来实现服务的发现与集成。在简单的应用中，为了缩短开发时间，并使系统避免过于复杂，用户可以不使用WSDL和UDDI，

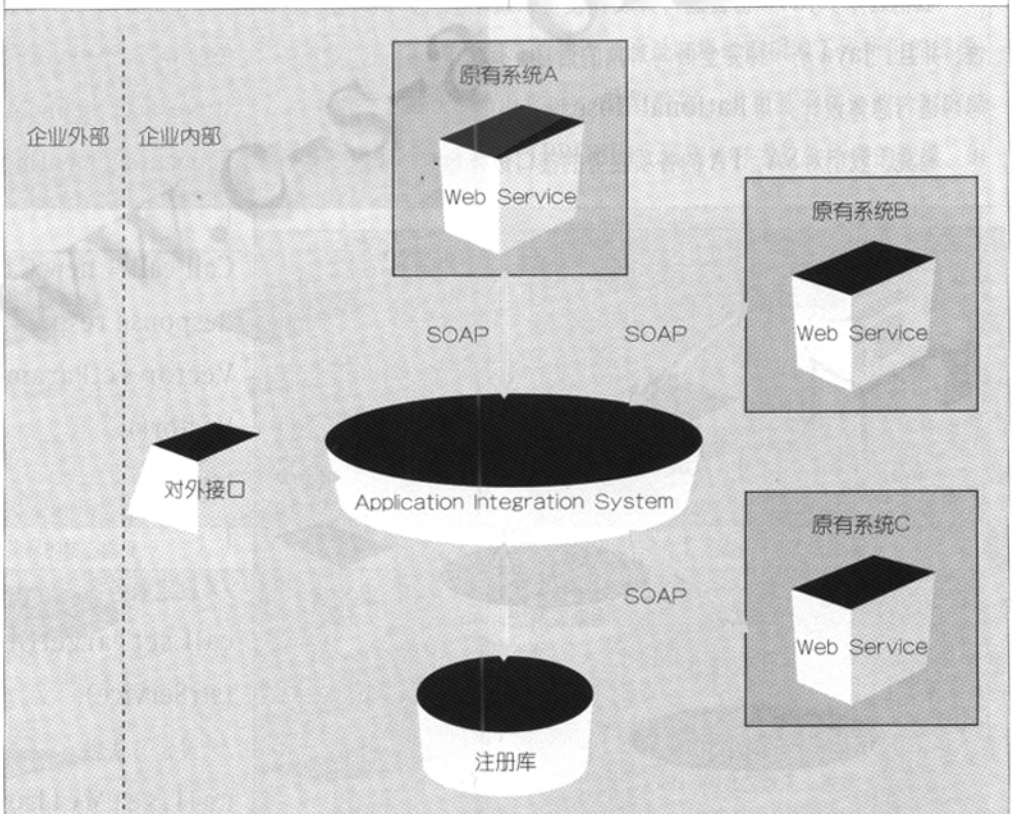

图1 基于SOAP的应用集成系统

但可以参考它们，自己定义一套服务描述和发现的机制，使得各子系统之间可以准确的描述和访问对方的服务。

部署一个基于SOAP的集成系统，需要进行几个方面的设置。首先，需要建立一个Web服务器，能够进行HTTP协议数据的处理，目前使用最为广泛的两个Web服务器分别是IIS和Apache Web Server。接着，需要在Web服务器上加载一个负责处理SOAP消息的通信模块，对HTTP数据包中携带的SOAP消息进行处理。为了处理XML格式的数据，还需要有一个XML解析器。目前，Microsoft公司和Apache组织分别提供了基于COM和Java的SOAP开发包、XML解析器，用户可以根据自己的实际情况来选择。

4 基于SOAP的应用集成方案的应用实例

下面具体介绍如何实现一个基于SOAP的应用集成系统，以我们做过的基于SOAP的模拟企业销售系统为例，逐步说明如何实现这一系统。

为简单起见，我们假定模拟的企业只有两个部门：销售部门(SA)和财务部门(FA)，销售部门负责接受用户定单、生成发货单、接受用户付款等业务，财务部门负责生成售货单、根据付款更新单据状态等业务。我们的任务是用SOAP将它们的业务集成起来，使外部用户系统能够访问企业的销售服务，以实现企业内部商务流程自动化和企业间的B2B电子商务。

我们选择了Java语言进行开发，使得系统具有良好的平台无关性，并且，Java是一种完全面向对象的语言，有利于运用面向对象的思想进行总体设计。用Rational Rose设计出的系统协作图如下：

首先，我们对SA、FA的各项业务的接口进行包装，使其他系统可通过SOAP对其进行访问；其次，需要对业务流程用XML进行描述，服务的描述文档和工作流的描述文档都存放在注册库中，应用集成系统AIS读取注册库得到描述文档，进而解析执行。即业务流程与AIS的程序相互独立，从而可以方便的更改业务流程而不改动程序，使业务容易更新、扩展。注册库可以使用新型的XML数据库(如Tamino数据库)，也可以使用传统的关系数据库，为简单起见，我们在实践中使用了关系数据库。

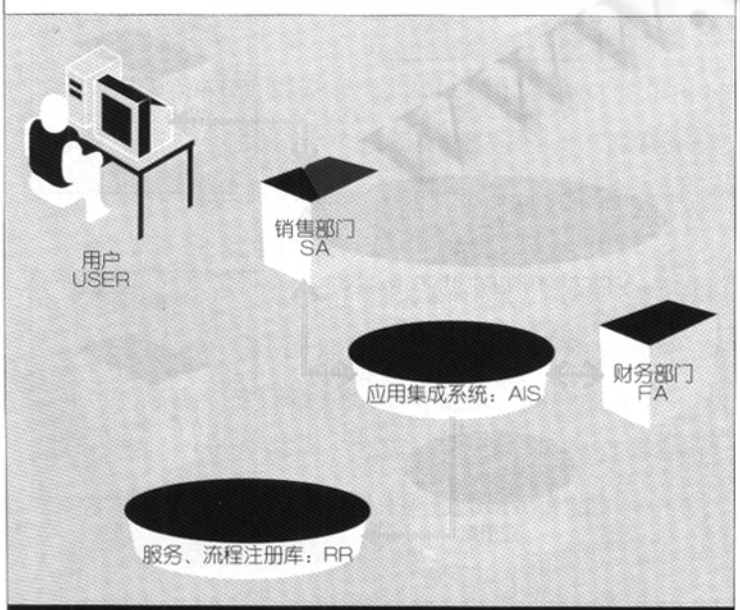

图2 系统协作图

上述步骤完成之后，AIS即可通过SOAP来调用SA、FA的服务，实现企业内部集成，接着发布对外服务，即可实现B2B业务。

一段典型的SOAP通信的代码如下：

```
public Object invoke
(String strServerURL,String
strService, String StrMethod,
Element elementInput)
{
Call call = new Call();
Response resp;
Vector vctParams = new
Vector();
try
{
//设定服务名
call.setTargetObjectURI
(strService);
//设定方法名
call.setMethodName
(strMethod);
call.setEncodingStyleURI
(Constants.NS-URI-LIT-
ERAL-XML);
vctParams.addElement
(new Parameter
("inputxml", Element.class,
elementInput, Constants.
NS-URI-LITERAL-XML ) );
//设定需提交的参数
call.setParams(vctParams);
//调用服务，得到执行结果
resp = call.invoke (new
URL (strServerURL ), "" );
}//try
catch (Exception e)
{ return null;
}//catch
if ( !resp.generatedFault())
return resp.getReturn
Value().getValue();
else
return null;
}//invoke()
```

参考文献

1 http://www.w3.org/TR/2001/WD-soap12-20010709/。

2 沈苗箭、范剑波、张森，基于分布式网络的SOAP协议，计算机应用研究，2001年4期。

3 http://d23xapp2.cn.ibm.com/developerWorks/xml/index.shtml。

4 http://www.w3.org/TR/2001/NOTE-wsdl-20010315。