第 31 卷 第 19 期 计 算 机 工 程 2005 年 10 月
Vol.31 № 19 Computer Engineering October 2005
·工程应用技术与实现· 文章编号：1000—3428(2005)19—0195—03 文献标识码：A 中图分类号：TP311.12

基于 XML 的异构数据交换系统的研究与实现

杨 剑[1]，唐慧佳[1]，孙林夫[2]，王胜银[1]
（1．西南交通大学计算机与通信工程学院，成都 610031；2. 西南交通大学 CAD 工程中心，成都 610031）

摘 要：XML 的可扩展性和自描述性以及其它特性为异构数据交换提供了新的思路。基于 XML 该文提出了一个异构数据交换的系统模型并探讨了该系统中各模块的功能和工作流程。

关键词：XML；电子商务；数据交换

Research and Realization of Heterogeneous Data Exchange System Based on XML

YANG Jian[1], TANG Huijia[1], SUN Linfu[2], WANG Shengyin[1]
(1. School of Computer & Communication Engineering, Southwest Jiaotong University, Chengdu 610031;
2. CAD Engineering Center, Southwest Jiaotong University, Chengdu 610031)

【Abstract】 Due to its extensibility and ability of self-description and other characteristics, XML provides a new way for heterogeneous data exchange. Based XML, this paper introduces a mode for heterogeneous data exchange system, discusses the function of every module in system and its workflow.

【Key words】 XML; E-business; Data exchange

1 XML 优点及相关技术

XML 是 World Wide Web 联盟（W3C）的一个开放标准，它是一组规则和准则的集合，由于以无格式文本（而不是二进制格式）来描述结构化数据，因此它具有良好的数据存储格式、开放性、可扩展性、自描述性、高度的结构化、便于网络传输等特性。XML 所关心的主要是数据，而其它的因素像数据结构和数据类型、表现和操作都由其它的以 XML 为核心的相关技术来完成。其中 XSLT(Extensible Stylesheet Language Transformations)用于将 XML 文档转换成其它类型的文档或其它格式的 XML 文档。Xpath(Xml Path Language)用于定义如何在 XML 文档中查找和定位数据。XML Schema 用 XML 描述了一类 XML 文档的数据结构和数据类型。通过使用上述几种技术，我们便可方便地把需要交换的数据转换成 XML 文档，在互联网和企业内部的不同应用程序间方便地提交数据、处理数据，完成数据的交换。

2 基于 XML 的数据交换模型

2.1 数据交换的实现模式

（1）面向数据的交换模式

面向数据的数据交换发生在不同系统的数据库之间，通过将数据从一个数据源移植到另一个数据源来完成数据的交换。其优点是它直接在数据访问层作应用开发，无须对应用逻辑与数据结构作任何改变。缺点是因为要直接对数据库进行操作，编程人员需要对原有数据访问层作大量修改同时又必须保持数据的完整性；在跨越互联网进行数据交换时，因为系统需要了解双方数据库的实现方式，需要承担很高的安全风险；而且，对于不同的应用，这种数据交换方式需要作不同的设计，系统的可重用性很低。

（2）面向应用接口的交换模式

面向应用接口的数据交换是指按各个应用接口所需数据格式在接口间转换数据、传输数据。这样，开发者就能够将现有应用捆绑在一起，允许它们共享商业逻辑和信息，这种策略的局限性一般是由接口的特征和功能所决定的。

（3）面向方法的交换模式

面向方法的数据交换是指在网络环境中的跨平台应用程序之间，建立一个可供多方共享的方法，实现数据共享和交换，采用面向方法的数据交换成可以实现 Internet 环境下的企业应用的松散耦合和集成，使企业可以方便地集成现有的应用并开发新的应用。

2.2 应用范围

本文设计的基于 XML 的异构数据交换系统是一个独立的第三方软件平台，可以为制造业、商业等各行业提供统一的数据交换服务，在具体实施时，根据需求的不同，可采用以上所有 3 种模式来实现。其主要应用范围为：

（1）企业内部的信息系统集成：用以实现企业内不同应用软件、不同部门之间的电子数据交换。把企业内以部门、业务为核心的闭环信息系统联系起来，形成更大更有效率的有机整体。提高数据的时效性、真实性、广泛性和使用效率。应用软件开发商和系统集成商将不必过多考虑软件之间的数据交换问题，可以专注于商业逻辑的开发。

（2）建立企业电子数据港(数据交换中心)： 采用通用数据交换技术构建企业的电子数据港，就如同建立了企业自己的数据交换中心。直接对外发布和接收电子数据，使企业大大提高与商业伙伴之间的数据沟通能力和内部运作效率。

（3）远程数据备份和数据同步：使用通用数据交换技术可以实现异地数据备份或数据同步，而不必顾忌数据来自什么地方，使用什么计算机系统，来源于何种数据库。

（4）企业 B2B 应用集成： 使用通用数据交换技术可以使企业迅速实现 B2B 应用集成。加快商务流程的速度，拓宽商业覆盖面，

基金项目：国家 863/CIMS 主题基金资助项目（2002A413620）
作者简介：杨 剑（1975—），男，硕士生，主研方向：电子商务，网络信息技术，异构数据交换技术；唐慧佳，副教授；孙林夫，教授、博导；王胜银，硕士生
收稿日期：2004-08-24 **E-mail：**yangjian2001@sina.com

加强和上下游供应商和经销商的合作和交流，更好地进行客户关系管理。

数据交换平台将数据交换功能作为一种Web服务提供给用户，用户通过一定的配置，在数据源地址和目标地址间建立相应的数据映射后，便可自动完成数据的交换。利用数据交换平台进行数据交换的系统，如果能够兼容XML，可以根据SOAP协议进行扩展，将数据交换的功能直接在系统中集成；如果不能兼容XML，则需要通过平台提供的XML生成服务将数据转换为相应的XML文档，再实现数据交换功能。而对于有能力进行二次开发的用户，数据交换平台也提供可扩展的.NET组件，可以根据需要进行进一步开发从而得到最终的应用程序。XML数据交换的整体示意如图1。

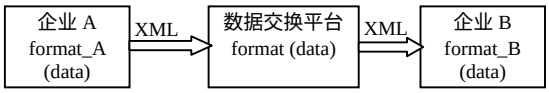

图1 XML数据交换整体示意

假设企业A有数据data，采用格式*format_A*封装，构成数据集*format_A (data)*，而企业B只能识别采用格式*format_B*封装的数据，同时，为保持最大的兼容性，平台将采用一个有最大兼容性的数据结构*format*，我们的数据交换平台就是要找出一种转换方式*f(X)*，使

$$\begin{cases} f_{A\to O}(x): format\_A(data)\to format(data) \\ f_{O\to B}(x): format(data)\to format\_B(data) \end{cases}$$

成立，其中$f_{A\to O}(X)$表示从企业A的格式转换为平台标准格式映射函数，而$f_{O\to B}(x)$表示从平台标准格式转换为企业B的格式映射函数。

整个系统的设计主要包括客户端和服务器端，客户端实现数据的封装和发送，服务器端则实现数据格式的注册、数据转换以及数据的处理。

2.3 数据交换客户端

（1）数据格式的注册

在数据交换前，发送方和接收方都需要在服务器端注册需要交换的数据类型、所采用的数据格式，即所采用的XML Schema。

（2）XML生成器

XML生成器把企业需要传送的数据按注册的XML Schema转换为相应的XML文档。

（3）XML封装/发送器

生成的XML文档采用SOAP格式封装，并用HTTP协议发送到数据交换服务器上。其实服务器端和客户端都将采用SOAP和HTTP相结合来传递消息，所以XML封装/发送器在服务器端和客户端都是必不可少的。

客户端的结构如图2所示。

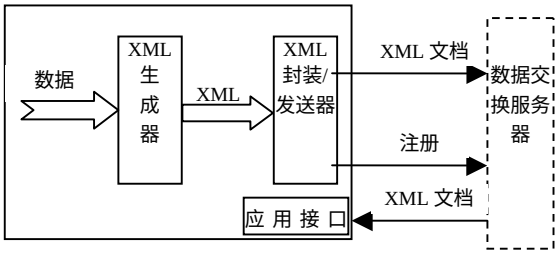

图2 XML数据交换客户端

2.4 数据交换服务器

（1）XML检查器

XML检查器的功能主要是对客户端的身份进行验证，同时检查客户端发送的XML文档格式是否良好，内容是否完整。

（2）格式转换器

格式转换器接收客户端注册的XML Schemas声明，并推导出把按该XML Schemas生成的XML文档转换到以平台采用的XML Schemas生成的XML文档所需的XSLT文件。

（3）XML映射器

XML映射器用上面生成的XSLT文档把客户端发送的XML文档转换为符合平台所采用格式的XML文档，以及从该XML文档到符合客户端格式的XML文档。

（4）XML处理器

XML处理器是数据交换的核心，它把标准格式的XML文档转换为符合需求的数据，如果接收方是平台自身，则按需求提交到数据库或相应的应用程序接口；如是另一个客户端，则需要再次调用映射器按相应的XSLT把采用平台格式的XML文档转换到符合客户端格式的XML文档。

服务器端的结构如图3所示。

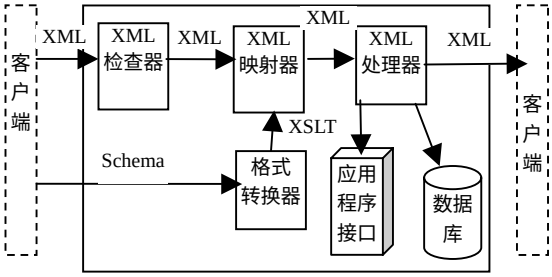

图3 XML数据交换服务器

如果客户端的数据直接来自于数据库，生成器和处理器直接面向数据库，则可用面向数据的交换模式实现数据交换；对于面向应用接口的数据交换，可按各应用接口的规范格式化数据，在生成器或处理器中接收、处理和传递这些数据，实现面向接口的交换；将数据交换的核心部分封装成Web服务，使之可由各应用程序互相访问，则可实现面向方向的数据交换模式。

3 数据交换流程

假设企业A和企业B之间有数据交互，完整的数据交换流程如下：

（1）企业A和企业B分别向数据交换平台注册并分别提交*format_A*和*format_B*，即他们用于生成XML文档的XML Schema；

（2）根据行业标准，对注册的数据类型建立一个统一的数据结构*format*，即该数据的标准XML Schema；

（3）根据双方的XML Schemas，生成转换规则$f_{A\to O}(X)$和$f_{O\to A}(X)$，即对应的XSLT文档，设为$XSLT_{A\to O}$和$XSLT_{O\to B}$；

（4）接收企业A向交换中心提交的XML数据—*format_A (data)*；

（5）验证企业A的身份和接收方企业B的身份；

（6）验证*format_A (data)*的格式是否正确，内容是否完整；

（7）调用$XSLT_{A\to O}$将*format_A (data)*转换为标准的XML文档，即$f_{A\to O}(x): format_A(data)\to format(data)$；

（8）调用$XSLT_{O\to B}$将采用平台格式的XML文档转换为企业B能识别的格式，即$f_{O\to B}(x): format(data)\to format_B(data)$；

（9）把生成的XML数据—*format_B (data)*发送到企业B。

如果数据交换的目的地为平台本身，则在第（8）步，XML处理器根据需求把XML数据转换为所需的各种数据，然后按交换的目的发送到数据库或提交到各种应用接口。

4 设计与实现

基于以上的分析，我们设计了一套完整的数据交换系统，用于不同的企业之间交互数据，首先定义了一套完整的XML封装规则，以便于在交换时方便地获取数据相关信息，其中主要的方法原型如下(由于篇幅有限，有所节略)。

(1) 生成 XML 文档

表 1 为本文定义的一系列属性。

表 1　XML 封装规则

ID	此 XML 文档的唯一标识，系统通过 ID 来标记和识别 XML 文档
Source	数据发送方的名称
SourceIP	数据发送方的 IP 地址
Destination	数据接收方的名称
DestinationIP	数据接收方的 IP 地址
Action(=“submit”| “backInfo”| “transform”| “operate”)	该 XML 的动作标识：submit—提交数据；backInfo—返回信息到客户端；transform—将此 XML 文件转换为其它文件：operate—直接进行数据的选择、查入、更新和删除操作
DataType	传入数据的数据类型（如订单、产品目录等）

下面是按以上规则生成的一个 XML 文档：

```
<?xml version="1.0" encoding="UTF-8"?>
<OrderTable id="orderId" Source="SenderName" SourceIP= "SomeIP" Destination="accepterName" DestinationIP="SomeIP" Action=" submit " DataType="order">
  <Order>… </ Order ></ OrderTable >
```

（2）将 XML 文档发送到远程服务器或客户端

用 SOAP 协议封装该文档，通过 HTTP 协议传到数据交换中心平台。用于发送的 Send 方法采用了.NET 类库提供的 HttpWebRebquest 和 HttpWebResponse 类，这两个类用于处理程序同 Web 服务器之间的 HTTP 通信。首先实例化一个 HttpWebRequest 对象，用它的 GetRequestStream()方法来发送数据。

```
HttpWebRequest sendRequest= (HttpWebRequest) WebRequest. Create(HttpSite)
System.IO.Stream request_stream= sendRequest. GetRequest Stream()；
```

再实例化一个 HttpWebResponse 对象，用它的 GetResponseStream ()方法来接收服务器返回的消息。

```
HttpWebResponse sendResponse= (System.Net.Http WebResponse) sendRequest.GetResponse();
System.IO.Stream response_stream= sendResponse. Get Response Stream();
```

（3）接收传入的 XML 文档

对 XML 文档的接收在设计中采用了.NET 的异步编程机制，交换中心首先异步读取传入的 XML 文档：先实例化一个 AsyncCallback 类，以启动异步的功能并在异步的功能结束后提供一个方法调用：myCallBack=new AsyncCallback (this.OnCompleted Read)，再用 OnBeginRead()开始异步读取传入的 XML 文件，接下来调用 private void OnCompletedRead(IAsyncResult asyncResult)方法读入 XML 文档。

读取完毕后，解析该 XML 文档，获得表 1 所定义的各个属性，根据相关值作进一步处理。

（4）验证用户是否注册

```
bool Isregister(string sender,string dataType))
```

（5）获取从发送者的 XML 到符合平台标准的 XML 转换或从符合平台标准的 XML 到接收者转换时所需的 XSLT

```
string GetXslt(string sender,string acceper,string dataType)
```

如：企业 A 的订单类 Schema 如下:

```
<?xml version="1.0" standalone="yes" ?>
<xsd:schemaxmlns:xsd= "http://www.w3.org/2001/XMLSchema">
<xsd:element name="Order">
<xsd:complexType><xsd:sequence>
…
<xsd:element name="Item" type="xsd:string" minOccurs="0" />
<xsd:element name="Date" minOccurs="0" maxOccurs=“ 0 ” maxOccurs="unbounded">
<xsd:complexType><xsd:sequence>
<xsd:element name="Month" type="xsd:string" minOccurs="0" />
<xsd:element name="Day" type="xsd:string" minOccurs="0" />
<xsd:element name="Year" type="xsd:string" minOccurs="0" />
…
```

而企业 B 的订单类 Schema 如下：

```
…
<xsd:element name="Order">
<xsd:complexType><xsd:sequence>
<xsd:element name="Date" type="xsd:string" minOccurs="0" />
<xsd:element name="Item" minOccurs="0" maxOccurs= "unbounded">
…
```

通过该方法生成的 XSLT 文档为：

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl= "http://www.w3.org/1999/XSL/ Transform">
<xsl:output method="xml" indent="yes" />
<xsl:template match="/">
<Order Id=…>
<Date>
<xsl:value-of select="/Order/Date/Year"/>/
<xsl:value-of select="/Order/Date/Month"/>/
<xsl:value-of select="/Order/Date/Day"/>
</Date>
<Item>
<xsl:apply-templates select="/Order/Item"/>
 </Item>
…
```

（6）XML 映射

用 XSLT 文档把客户端发送的 XML 文档转换为符合平台标准格式的 XML 文档，或从该标准格式的 XML 文档到符合客户端格式的 XML 文档，这里采用了.NET 的 XslTransform 类中的 Transform 方法来进行转换。

```
xslTransform.Transform(new XPathDocument (xmlReader), null, xmlWriter, null);
```

通过以上方法，本文实现了数据交换功能，并将其应用于区域协同电子商务项目中，很好地解决了系统间以及系统内部的数据交换问题。

5 结束语

基于 XML 的数据交换平台的建立使不同企业的异构系统之间实现最大限度的协同，并能扩展现有数据交换应用，从而使企业间的应用集成成为可能，使企业的供应链各环节有机地结合起来，建立基于 Internet 的供应链协作模式，从根本上促进电子商务的发展。但应该注意到 XML 文档一般都比采用二进制格式的文件要大,当有大量数据需传输时，网络的负荷是需要着重考虑的问题，另外，如何保证数据交换的事务性、如何保证数据安全性等是需要进一步考虑的问题。

参考文献

1 Simple Object Access Protocol(SOAP)1.1 [EB/OL]. http://www3.org/TR/ SOAP, 2002-03-20

2 Pendyala V S, Shim S S Y, Gao Z. An Xml Based Framework for Enterprise Application Integration[J]. IEEE International Conference on E-commerce, 2003-06:128-135