

基于 XML 的三层 C/S 模型

邓东华 杨宗凯 乐春晖 (华中科技大学电信系电子商务中心 430074)

摘要:可扩展标志语言(简称XML)在实现信息标准化、信息统一、信息的交流与共享上有着独特的技术优势,因此它受到了广泛的重视。本文介绍基于XML的三层Client/Server模型的应用,并具体介绍其实现过程。

关键词:XML Client/Server 模型

1 问题的提出

可扩展的标记语言(Extensible Markup Language,缩写为XML)是SGML(Standard Generalized Markup Language,标准通用标记语言)的一个子集,是简化的SGML。它是互联网联合组织(W3C)创建一组规范,以便于软件开发人员和内容创作者在网页上组织信息,其目的不仅在于满足不断增长的网络应用需求,而且也在于确保在通过网络进行交互合作时,具有良好的可靠性与互操作性。

随着Web技术、分布式对象技术两者的有机结合,传统的Client/Server主从结构逐渐向灵活的多级分布式Web计算模型进行着演变。多级分布式Web体系结构可划分为三层:表示层(客户机层)、功能层(应用服务器层)、数据层(数据库服务器层)。

而由于XML可以自定义文件类型,有利于信息的表达和结构化组织,而且可以以一致的方式格式化和传送数据,若将其运用到三层Client/Server模型中,必将产生很大的网络应用优势。于是也就引出了本文讨论的问题——基于XML的三层C/S模型。

2 基于XML的三层C/S模型

从网络体系结构各层次上看,基于XML的三层C/S模型具体实现的功能概括如下:

数据层——实现数据的集成,XML数据产生于多种数据源,但都以统一的XML格式表达传输。

功能层——实现数据的发送与处理,应用服务器通过HTTP交换数据,通过DOM处理XML数据。

表示层——实现数据的显示,XML数据的可以有多种表现形式,而且可以为外部直接访问、编辑或转换,也

可被其他系统所用。

基于XML的三层Client/Server模型的应用结构如图1所示:

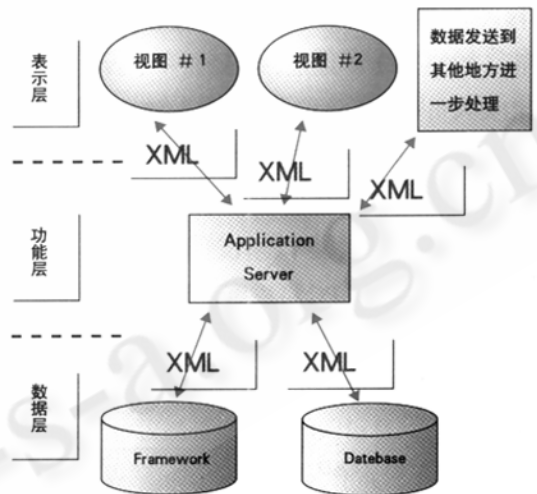


图1 基于XML的三层C/S模型的应用结构

在基于XML的三层C/S模型中把XML作为一种结构化信息交换的表示方法,负责所有与数据源的通信,根据最终用户的要求,聚集和组织从多个远程数据库服务器上获取数据,并用XML形式交互式的把数据源返回的消息传递给客户机。

以下是对基于XML的三层C/S模型应用过程的分析:

2.1 从数据层获取数据生成XML文档

由于XML能够使不同来源的结构化的数据很容易的结合在一起,所以通过XML,可以在中间层服务器上对从后端数据库和其他应用处来的数据进行集成。然后,数据就能被发送到客户或其他服务器做进一步的集合、处理

和分发。

虽然每个数据库描述数据都是不同的,但XML可以自己定义文件标签,例如:通过XML,图书管理资料就可以很容易以标准的方式按照书名、作者、图书简介或其他的标准进行分类。

2.2 XML 数据的发送与刷新

利用XML开放的、基于文本的格式,可以将它通过HTTP像HTML一样传送。同时,中间层应用服务器将支持XML更新功能,将数据服务器上数据的变化及时地传递给客户,反之亦然。因此,中间层能够从客户端得到更新的数据,并把数据传送到数据储存服务器上。

2.3 XML 数据在表示层的显示

XML描述的数据发表示层后,能够用多种方式显示。XML定义的数据,其显示与内容是分开的,因此允许对同一数据指定不同的显示方式,使数据更合理地表现出来。CSS和XSL为数据的显示提供了公布的机制,本地的数据能够以客户配置、使用者选择或其他标准决定的方式动态地表现出来。如图2所示。

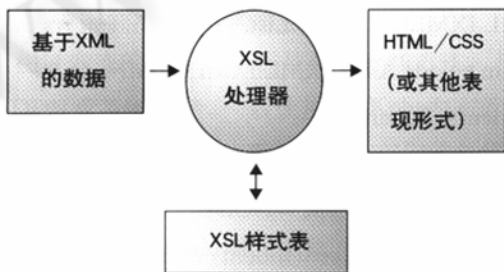


图2 XML数据通过XSL不同形式的表

HTML描述数据的外观,而XML描述数据本身。由于XML注重的是内容,其显示要通过XSL(eXtensible Stylesheet Language,可扩展样式语言)或CSS来实现。XSL是为XML文件定义的一种标识语言,它提供类似但却远远超过CSS的强大功能。

XSL以包含了一套元素集的XML语法规范而定义,该语法规范将被用来把XML文件转换成HTML文件或XML文档。一个XSL样式表集合了一系列设计规则以用于将信息从XML文件中汲取出,并将其转换成HTML等其他格式。XSL凭借其可扩展性能够控制无穷无尽的标签,而控制每个标签的方式也是无穷尽的。这就给Web提供了高级的布局特性。

2.4 处理和编辑XML数据

在基于XML的C/S模型中,不但能够通过XML集合多个数据源的数据,并把数据源数据及时传输给表示层,而且,只要能理解XML数据,就能从客户层处理和编辑中间层的数据,然后通过中间层与数据层之间的数据刷新机制,对数据层数据进行刷新。

运用XML解析器能够读入一串XML数据,经过处理,产生一棵结构树,并且使用DOM把所有数据元素作为对象。解析器或者用脚本把数据进行进一步的处理,或者把数据移交给另外的应用软件或对象进行进一步的处理。

DOM实际上是一个应用编程接口(API),用来定义一种标准方法。通过这种方法,能够处理XML结构树的元素。对象模式控制着使用者如何同结构树交流,并且把所有树的元素作为对象暴露出来。DOM用扩展方式支持名域、数据类型、查询和XSL转化。

3 基于XML的三层C/S模型的应用实例

下面具体介绍如何实现基于XML的三层C/S模型,以我们做过的三层C/S模式的图书资料管理为例,它是基于XML表达、传输与处理的,其详细实现如下:

3.1 获取数据生成XML文档

中间层通过XML获取数据源数据,生成XML文档,其中每一的书目信息包括:书名、作者、图书简介、出版日期。下面所列的就是获取数据源的一段典型例程。

```

<%@ LANGUAGE=VBScript %>
<? XML VERSION="1.0" encoding="GB2312" ?>
<图书资料管理>
<%Set Connection=Server.CreateObject("ADODB.
Connection")
Connection.Open
Set ItemRecordset =Connection.Execute("select * from
item ")
Do while Not ItemRecordset.EOF%>
<书目>
<书名><%= ItemRecordset("书名") %></书名>
<作者><%= ItemRecordset("作者") %></作者>
<图书简介><%= ItemRecordset("图书简介") %></
图书简介>
<出版日期><%= ItemRecordset("出版日期") %></
出版日期>
</书目>
  
```

```
<%ItemRecordset.MoveNext
```

```
Loop%>
```

```
</图书资料管理 >
```

3.2 XML 文档

XML 通过 HTTP 从应用服务器传送到客户机层, 并且支持 XML 更新功能, 使得中间层或数据服务器上数据的变化可以传递给客户。

3.3 XML 数据的在表示层 (客户机层) 的表达

发送到客户端的数据 (图书资料管理数据), 可以根据用户对象及其不同的应用要求有不同的显示形式。如下所示的 XSL 样式表, 就是一个显示示例, 将生成一个 HTML 文档, 包括一个表格, 其中的一行就为一个“书目”元素。

```
<xsl:template match="/">
<HTML>
<HEAD>
<TITLE> 图书资料管理 </TITLE>
</HEAD>
<BODY>
<TABLE>
<xsl:for-each select=" 图书资料管理 ">
<TR>
<xsl:for-each select=" 书目 ">
<TD>
<xsl:process-children/>
</TD>
</xsl:for-each>
</TR>
</xsl:for-each>
</TABLE>
</BODY>
</HTML>
</xsl:template>
```

3.4 编辑、处理数据的实现

只要理解了中间层 XML 文档的数据, 就能从任何地方处理和编辑数据 (图书资料管理数据)。

如下例所示, LoadDocument 程序打开 XML 文档, 然后调用另一个程序 DisplayNode, 它用来显示文档的结构。LoadDocument 传递给正在打开的 XML 文档的 ChildNodes 属性一个引用作为它的参数, 同时传递一个整数用来标示开始显示的层次级别。代码利用参数来格式化在 Visual

Basic 文档结构显示窗口中的文本。

DisplayNode 属性的函数会遍历文档寻找需要的 NODE_TEXT 节点类型串, 一旦代码找到一个 NODE_TEXT 的节点, 它会利用 NodeValue 属性来获得相应的文本串。另外, 当前节点的 ParentNode 属性指向一个元素类型的节点。元素节点实现了一个 NodeName 属性,

```
Public Sub LoadDocument()
Dim xDoc As MSXML.DOMDocument
Set xDoc = New MSXML.DOMDocument
xDoc.validateOnParse = False
If xDoc.Load("C:\ My Documents \ sample.xml")
Then
DisplayNode xDoc.childNodes, 0
Else
' The document failed to load.
End If
End Sub

Public Sub DisplayNode(ByRef Nodes As
MSXML.IXMLDOMNodeList, _ByVal Indent As Integer)
Dim xNode As MSXML.IXMLDOMNode
Indent = Indent + 2
For Each xNode In Nodes
If xNode.nodeType = NODE_TEXT Then
Debug.Print Space$(Indent) & xNode.parentNode.
nodeName & _
":" & xNode.nodeValue
End If
If xNode.hasChildNodes Then
DisplayNode xNode.childNodes, Indent
End If
Next xNode
End Sub ■
```

参考文献

- 1 Hiroshi Maruyama, Kent Tamura, Naohiko Uramoto. XML and Java Developing Web Applications [M]. Addison-Wesley, August 1999.
- 2 Ed Tittel, Norbert Mikula, Ramesh Chandak. XML For DUMMIES [M]. IDG BOOKS Worldwide, Inc, 1999.
- 3 宋晓梁, 刘东生, 许满武. 中间件及其在三层客户机/服务器模型中的应用 [N]. 计算机应用 990713