李昕 左明（江苏 徐州 中国矿业大学 计算机学院 221008）

基于Java与XML的P2P应用开发

Development of P2P Aplication Based on Java and XML

摘 要：目前P2P技术逐渐得到了广泛应用。给出了一个带有发现和查找服务器的P2P应用模型，介绍了该模型的三个重要组成部分：服务器、监听器、浏览器。重点探讨了Java与XML在P2P程序开发中的应用。

关键词：XML P2P 监听器

1 引言

随着时代的发展，Internet上的资源呈现爆炸式增长的态势。而与此同时，资源的流向却趋于集中化，大量公开的资源以Server形式在Internet上提供。Server集中式的服务方式有许多技术弊端。最主要的问题就是资源无法得到充分利用。有资料统计，全球Server提供的资源加在一起还不足Internet资源总量的1%。就是说最多最好的资源实际上是存在于我们每一个人的PC中。人们迫切希望能打破Server的垄断，在Internet上拥有属于自己的空间。P2P技术正是基于这个目标而诞生的。

P2P技术将Internet服务提供方式划分为3种：完全基于Server（Server-based），少量借助Server（with-Server），完全脱离Server（non-Server）。P2P主要面向后两种情况。“少量借助Server”这种方式是现在比较常见的P2P解决方案。像曾惹来广泛争议的Napster、现在欧美非常流行eDoney，以及我国P2P fans开发的Jelawat、Workslink等，都属于这类产品。目前这类产品多以File Sharing服务为主，并兼有简单即时通信功能。这种方式的一个主要特点是，Server的功能已经远远退化，一般只作为Index Server使用，提供所有Peer以及之上各种文件列表查找索引服务。本文即探讨了一种带有发现和查找服务器的P2P应用实现。

2 系统组成

系统主要由以下3部分组成：服务器、浏览器、监听器，下面分别予以分析。

2.1 服务器

服务器保留所有注册监听器的清单和这些监听器共享给其他端资源的详细描述。监听器从服务器上得到其他监听器和共享资源的信息后，余下的工作就在监听器间处理。

2.2 浏览器

浏览器充当用户与计算机之间的接口。通过浏览器，客户发送请求给对方监听器和接受来自对方监听器的响应。

2.3 监听器

监听器执行的第一项工作就是由浏览器调用，登录到服务器上，通知服务器它的存在并列出它共享资源的清单。这里要说明的是，监听器只是发送文件和文件夹的位置，而不是文件和文件夹的内容。监听器和浏览器一起构成了通信的一端。

3 工作过程

图1给出了在P2P程序中服务器、浏览器、监听器这三者之间的关系。

在这个应用程序中，被描述为监听器的端负责其他端产生的请求，而浏览器是产生所有请求的端。服务器维护保存所有注册的端的信息的数据库，如端的IP地址、登录名和端已声明为其他端共享资源的信息。程序中组件间的通信是基于XML的，用XML解析器来

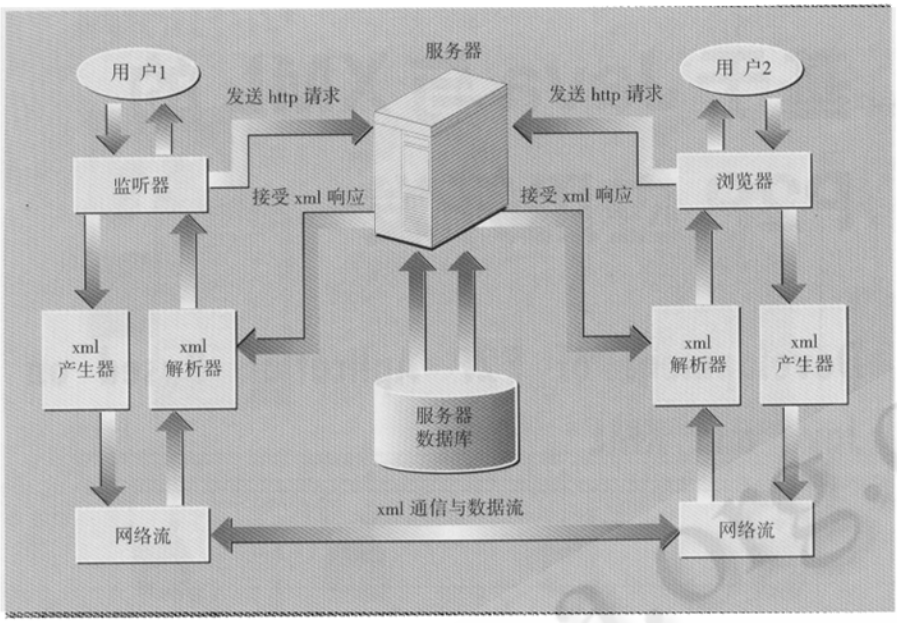

图 1　系统工作的大致流程

解析产生的响应和请求；XML产生器产生合理的请求和响应。下面是系统的工作过程：

3.1 监听器向服务器的注册

程序开始时，监听器会产生一个登录窗口，提示你键入自己端的名字，用来对你进行身份验证。在内部，使用监听器的IP地址而不是登录名来进行通信。键入登录名后，发送一个包含登录名、IP地址和共享资源等信息的HTTP请求，提交到服务器，服务器调用名称为login.asp的脚本，该脚本验证从监听器收到的信息，如果传送的信息是正确的，则服务器返回验证成功，并把监听器的传送的上述信息记录到数据库中；然后给监听器一个登录成功的XML响应。如服务器发现发送的信息有歧义，验证就会失败，接着给监听器一个登录失败的XML响应。当监听器注销时，便从服务器中删除相应的条目。

3.2 浏览器与服务器间的通信

一旦监听器成功登录到服务器上，监听器便可准备处理其他端的要求。而浏览器为了查找内容，发送一个HTTP请求到服务器，服务器再一次对端产生XML格式的响应。

之所以使用XML作为各组件间通信的媒介，是因为现在有适合各种语言调用的XML解析器，从而使程序有更好的可读性与扩展性。

浏览器根据自己的需求找到相应的监听器后，服务器的作用就不存在了，因为它已经完成了初始化端间的通信的任务。接下来，通信在两个端（浏览器和监听器）之间直接出现。

3.3 端与端之间的通信

端与端之间有两种最基本的通信：上载和下载。浏览器选定合适的监听器后，向它发出一个要求下载其主机上共享文件的XML请求，监听器得到这个请求后，便把相应的文件写到网络流中；同时，浏览器开始从同一个网络流中读数据。上载文件与之类似，浏览器产生XML上载请求并发送到监听器中去。接到请求后，监听器打开流，从而可以和用户通信。这里由于监听器可同时接受多个浏览器的访问，所以在监听器的设计中采用了多线程的技术。

4 系统设计中的关键点

系统采用Java语言来开发，Java语言一直都被认为是最适合进行XML编程的语言之一。与平台无关的语言Java加上与平台无关的数据XML，将使程序具有更好的移植性与扩展性。下面主要探讨在具体开发过程中遇到的三个主要问题。

4.1 XML文件解析器

在P2P系统中监听器与浏览器之间的请求与响应信息主要以XML格式发送，所以系统中必须实现对XML数据的解析。下面给出XML解析器的主要代码：

```
//XMLParser.java
.....
//导入XML解析器用到的包和类
import org.xml.sax..*
 import org.apache.xerces.parsers.SAXParser;
.....
 public class XMLParser
{
Vector value;//定义对象value,保存解析后的xml信息
public void perform(String uri)//参数uri中存有被解析文件的信息
{System.out.println("Parsing XML File: "+uri+"\n\n");
try
{XMLParser parser=new SAXParser();//产生XMLParser类对象
 MyContentHandler contHandler=new MyContentHandler();//产生MyContentHandler类对象
 Parser.SetContentHandler(contHandler);//回调函数，建立对象contHandler和parser的联系
 Parser.parse(uri);//解析xml文件
 value=contHandler.returnvector();解析后的xml信息存于对象value中
}
catch(SAXException e)//异常处理
 {System.out.println("Error in parsing:"+e.getMessage());
```

```
    }
  }
  ......
    }
```

把要解析的XML文件通过参数uri传给类XMLParser，在这个类的实现中，我们调用了解析XML文件的类SAXParser，程序首先定义了一个SAXParser类的对象，接着产生一个MyContentHandle类的对象，在这个类中，XML解析器产生回调，调用方法SetContentHandler，而由MyContentHandler类的对象contHandler 作为参数。XML文件经解析后，把数据存于Vector类的对象value中，这个对象返回到调用程序。最后如果程序出现异常，则显示出错信息。这里我们省略了MyContentHandler类中如何从Vector类的对象value中提取有效信息的代码。

4.2 XML 文件产生器

与XML解析器相对应，我们要把浏览器与监听器之间的请求与响应信息用XML文件来表示，这就要用到XML文件产生器。下面给出XML文件产生器中浏览器发给监听器请求转化为XML的主要代码：

```
    //XMLWriter.java
    public class XMLWriter
    {
  StringBuffer sbuf=new Buffer();
  String type="",mask="";
  void requestString(String filetype, String filename)
  { // 在字符串缓冲区加上XML和p2p_lng等所有请求的共有标记
    sbuf.append("<?xml version=\"1.0\" encodeing=\"utf-8\" ?>");
    sbuf.append("<p2p.lng>");
    //根据请求的参数作相应的处理
  sbuf.append("<request type=\""+filetype+"\">");
        if (filetype.equals("SHOWFILE") || filetype.equals("DOWNLOAD") || filetype.equals("UPLOAD") )
        {sbuf.append("<scope type=\""+filename+"\" mask=''>");
  sbuf.append("</scope>");
  }
      //把余下的标记补齐
  sbuf.append ("</request>");
    sbuf.append("</p2p_lng>");
    //把缓冲区中的内容以字符串的形式从程序中返回
  tt = sbuf.toString();
        return tt;
  }
  ......
  }
```

浏览器发给监听器请求包括上载、下载、显示文件等等，在XML文件产生器中，首先定义了sbuf，type，mask等变量，接着在字符串缓冲区加上所有请求的共有标记。如XML标记和p2p_lng标记等。然后，我们根据请求的参数，如SHOWFILE、DOWNLOAD、UPLOAD等加上适当的范围类型，把余下的标记补齐。最后我们把缓冲区中的内容以字符串的形式从程序中返回。一个浏览器发给监听器请求的XML文件就这样产生了。而监听器发送给浏览器的响应XML文件的产生过程与之类似。

4.3 监听器的实现

监听器是P2P应用程序的核心，它处理客户请求，包括显示文件、下载文件、上载文件等。为了读取请求并对客户响应，这个类必须可解析XML文件，调用XMLParser类；而产生XML文件时，调用XMLWriter类。监听器工作的流程及主要代码略。

监听器在固定的端口PORT监听，一旦有连接，产生一个新的套接字处理与浏览器的通信。根据浏览器的请求，分别进行不同的处理。

5 结束语

本文给出了一个典型的P2P应用的框架及相应的实现代码，但在此基础上，还可做很多有益的扩展。比如可增加流媒体的操作功能，而这只需要在监听器端增加一个访问类型为STREAM文件的判断，并增加相应的控制代码；还可增加浏览器与监听器间消息通信的功能，把消息转化为固定的XML格式并改写XML解析器来处理它即可实现。

参考文献

1 许斌等，JXTA-Java P2P 网络编程技术 [M]，清华大学出版社，2003。

2 吴文辉等，对等网络编程源代码解析[M]，电子工业出版社，2002。

3 P2P 应用程序框架[EB/OL].http://www.huihoo.com/p2p/p2p.html,2001。

4 P2P 技术与应用[EB/OL].http://www2.ccw.com.cn/01/0128/d/0128d06_1.asp,2001。