Java Applet 与 Web Server 对象间的调用技术

钱晓红　马沂夫　（东北大学计算中心　110006）

摘要：本文介绍了 JAVA Applet 与 Web Server 对象间的调用技术，并结合实例强调了 Java Applet 在 Web 信息发布中的重要作用。

关键词：Java Applet　Internet　Web

一、引言

随着 Internet 的迅速发展，许多传统的 Client/Server 应用系统正逐步转向 Browser/Web Server 模式。建立动态 Web Server 的方案层出不穷，这些方案大多在 HTML 中加入脚本语言（VBScript、JavaScript 等）并通过调用服务器方的对象实现对数据库的访问、文件的读写及其他一些功能，但它还无法与传统的 Client/Server 应用程序相比。Java 的出现正是为 Browser Web Server 模式提供了强有力的工具，弥补了某些不足。Java 与平台无关、安全、功能强大的网络应用等使得 JAVA 成为一种 WEB 上的理想开发工具，为解决动态 WEB 信息发布提供了较好的可能。

Java Applet 通常称为小程序，可以嵌入到 HTML 文档中，当有用户访问网页时，它可以从服务器端下载并由客户端的 Web 浏览器激活执行，Java Applet 可以建立自己的网络连接并使用各种需要的协议来获取服务器上的信息。它与其他对象之间的操作一般有如下几种方式：Java Applet 对宿主服务器的文件访问、Java Applet 对 RMI 对象的调用、Java Applet 与 Java Servlet 的通信、Java Applet 之间的通信方式等。

二、Java Applet 与 Web Server 对象间的操作

1. Java Applet 对宿主服务器的文件访问

只要得到在 Web Server 上的授权，Applet 就可以直接读写 Web Server 上的文件，下面这段程序打开一个 IP 地址为 202.118.0.1 的服务器宿主目录下的名为“data.txt”的文本文件并读它：

```
InputStream is;
Byte buffer[] = new byte[24];
String datafile = new String("http://202.118.0.1/data.txt");
try{
  is = (new URL (getDocumentBase( ), datafile)).openStream();
} catch(Exception e) {}
try{
  is.read(buffer,0,buffer.length);
} catch(IOException e) {}
```

应当注意的是：指定的 URL 地址一定是宿主服务器的地址，否则访问将被拒绝，openStream 是远程输入流。此外也可以采用套接字的方法实现对服务器文件的访问。

2. Java Applet 对 RMI 对象的调用

为通过网络执行其他机器上的代码，传统的方法不仅难以学习和掌握，也极易出错。JAVA 1.1 的远程方法调用采用了一种解决这一问题的最佳方法：某些对象正好位于另一台机器，我们可向它发送一条消息，并获得返回结果，就象那些对象位于本地机器一样。Java 远程方法调用（RMI）特性使客户机上运行的程序可以调用远程服务器上的对象。Applet 作为 RMI 的客户，能访问 RMI 服务器上的资源，可以通过向 RMI 发出查询数据库的请求，最后得到结果。它为分布式对象之间相互引用提供了机制。与 Servlet 不同的是 RMI 服务程序作为 Java Application 运行在服务器上而不需要 Web Server 软件的支持。下面程序实现了客户端从服务器得到的精确时间。

```
import java.rmi.*;
interface Time2 extends Remote{//定义接口，客户机和服务器共享
long getPerfectTime( ) throws RemoteException;
}
```

服务器端实施过程：

```
import java.rmi.*;
import java.rmi.server.*;
import java.rmi.registry.*;
import java.net.*;
public class Time1 extends UnicastRemoteObject
                   implements Time2
{//服务器必须扩展 UnicastRemoteObject 类并实现
```

```
Time2 远程接口
    public long getPerfectTime( ) throws RemoteException
    {
      return System.currentTimeMillis();
    }
  //构造器必须抛出 RemoteException 异常
  public Time1( ) throws RemoteException
  {super();}
  public static void main(String args[]){
//创建一个安全管理器,令其支持 RMI
  System. setSecurityManager ( new RMISecurityManager());
  Try{
    Time1 p=new Time1();
        //设置注册表,IP 地址为 202.118.10.60 端口为 2000
          Naming. bind ( "//202. 118. 10. 60: 2000/Time1",p);
      {catch (Exception e) {}
    }
  }
  客户端实施过程:
  public class DispTime{
  public static void main(String args[]){
    System. setSecurityManager ( new RMISecurityManager());
    Try{
      Time2 t = (Time2) Naming. lookup ( "rmi://202.118.10.60:2000/" + "Time1");
      System.out.println(t.getPerfectTime());
      {catch(Exception e){}
    }
  }
```

程序运行前服务器端必须创建根和干,并让服务端程序以一个独立的进程在后台运行。

3. Java Applet 与 Java Servlet 的通信

Java Servlet 是运行在服务器端的 Java"小程序",一种在服务器端执行请求/应答模式的服务程序。Applet 运行时间可以与 Servlet 进行通信,由 Servlet 来完成诸如数据库查询(通过 JDBC)等须在服务器端完成的功能,将结果传给 Applet。Servlet 给开发人员在服务器端的编程提供了很大的方便。由于 Java 的字节代码比其他语言更易被反汇编,所以具有重要价值的专用算法通常不会交给 Applet 去做,而可以采用 Servlet 与 Applet 协同工作的方案。此外,应用 Servlet 还可以建立 RMI 或 CORBA 的服务器程序,由客户端的 Applet 调用。目前 SUN 公司的 Servlet 支持十几种 Web Server,特别是 LiveSoftware 公司的 JRUN,支持包括 MS IIS 在内的多种平台的 Web Server,是功能强大的服务器方开发环境。

4. Java Applet 之间的通信

若嵌在网页中有多个 Applet,他们之间可以相互通信,例如下面的程序段假设有两个 Applet,分别为 App1 和 App2,如果符合通信的权限要求,App2 可以在 App1 上写一串文字:

```
Applet otherApplet = getAppletContext( ). getApplet();
If (otherApplet = null){
  Int x = otherApplet. size(0. width;
  Int y = otherApplet. size( ). height;
  Graphics g = otherApplet. getGraphics( );
  g. drawString("hello",10,20);
}
```

从以上的介绍可以看出,有了数据访问的渠道、权限和能力,再加上 Java 语言本身的强大功能,Applet 在 Web 浏览器上可以完成一些传统的 Client/Server 应用系统所能完成的任务。

三、应用举例

某一大型实时监控系统原是在小型机 UNIX 系统下采用面向对象技术、使用 C++ 语言开发的,其操作界面基于 X 窗口,并且显示的大量状态图形均是带有复杂属性的对象。由于网络通信负载过大,X 终端的数量受到限制,满足不了各部门对系统状态进行监视的要求,需要在原有系统上增加使用 Web 浏览器来监视系统状态的功能。我们采用了动态 Web Server 与 Java Applet 应用相结合方案,原有系统作为 Web 发布系统的数据源负责数据的抽取整理并存文件,Web Server 负责基于 Web 的信息发布,主要包括动态页面的生成、访问用户和权限的管理等功能。客户机通过 Web 浏览器访问 Web Server,数据文件放在 Web Server 上,并采用如上介绍的技术来实现 Client 与 Web Server 之间的协同工作,因此该系统很好地实现了系统所要求的功能。

参考文献

[1] [美] Bruce ckel 著,《Thinking in JAVA》,机械工业出版社

[2] 周世雄编著,《NT 动态网站设计指南》,大连理工大学出版社,1997

[3] SUN Educational Services,《Java Programming》,SUN Microsystems Inc. June 1997

(来稿时间:1999 年 6 月)