动态请求的 WEB CLIENT

陆 晟、 龚 俭
东南大学计算机系　南京 210096

摘要：本文分析了动态请求的 Web Client 在 Web 技术中重要性；介绍了三种不同的方法以实现动态请求的 Web Client；并从不同的角度比较了这三种方法。

关键词：计算机网络、 WEB技术。

一、引言

在 Intranet 大发展的今天，人们开发了越来越多的基于 Web 的程序。这些程序常常采用的方式如图1：

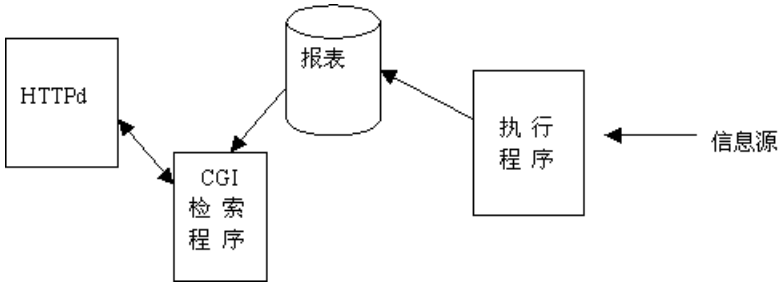


图 1. Web管理系统的一般结构

执行程序在 WWW Server 端自动执行，生成各种报表，例如：网络监视程序把网络的流量情况记录下来。客户通过 WWW 浏览器(如 Netscape)请求 Server 端，在 Server 端的 CGI 程序提供动态执行的能力，将客户的请求结果动态生成并翻译成 HTML 格式返回给浏览器。

但是，客户可能会提出新的要求，一种常见的情况是：“用户要求能够在一段时间内连续跟踪 Server 端的执行程序的运行结果。” 例如：连续监视目前的网络流量。

因此，需要有某种基于 Web 的技术，能够在 Client 端定期向 Server 端发请求，能够解释 Server 的响应，并且能够把响应动态地显示给用户。

本文中将介绍三种不同的方法来实现这个要求，它们分别是：基于 Netscape 附件的技术、基于 X Windows 的技术和基于 Java 的技术。

二、用基于 Netscape 附件的技术实现动态请求的 Web Client

这个技术的基本思想是：把 Server 端回送的数据定义为某一种特殊类型，于是， Netscape 会根据定义调用某一特殊的附件来解释该数据，而这个附件是根据需要自己编写的，这样就可以在 Client 端动态的请求 Server 端了。

它的原理如图2：

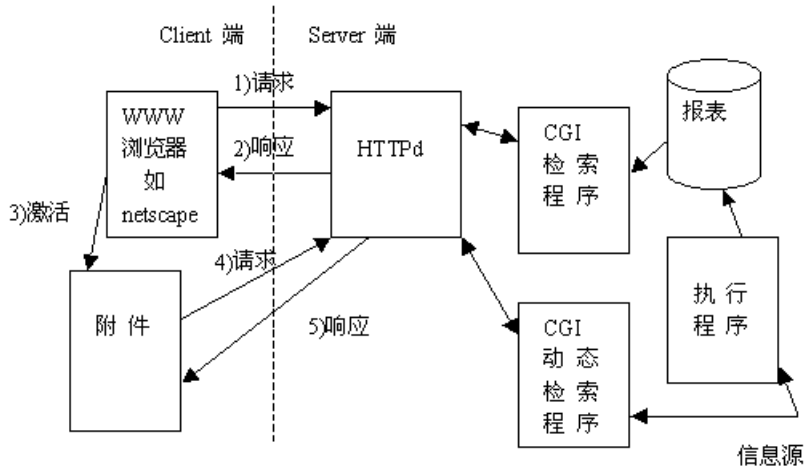


图 2. Netscape附件方式工作原理图

比较图中的 Server 端和前面基于 Web 系统的一般结构，可以看到仅仅多了 CGI 的动态检索程序。在指定 Netscape 访问该程序时，该程序回送的类型是某种特殊的类型，例如：

```
Content-type: video/dy
```

这种类型是自定义的(不再是“Content-type: text/html”)，相应地，在 Netscape 的“Option”菜单中的“General Referances…”对话框中的“Helpers”中定义对这种类型数据的解释器为自己编写的附件(正如使用“媒体播放器”解释“audio/basic”类型的数据一样)。

因此，执行的过程是：

1) Netscape 向 Server 发出对 CGI 动态检索程序的 URI 的请求；
2) Server 回送一种特别的数据；
3) Netscape 根据 Helpers 的定义激活指定的附件；
4) 该附件定期向 Server 端发送 HTTP 格式的请求；
5) 该附件接受 Server 端的响应，并解释数据内容绘制在屏幕上。

至于附件的编写方法，则倚赖于 Client 端的窗口系统，比如，在使用 X Windows 的环境下，该附件的编写需要使用 X Windows 技术和 Unix 系统的 Socket；在 Windows 环境下，则使用 Windows 编程和 Winsock。

三、用基于 X Windows 的技术实现动态请求的 Web Client

这个技术实际上已经不是一个 Client 方面的技术了，用一句简单的话说就是将 Server 端的运行程序直接甩到 Client 端的屏幕上。

它的工作原理如图3：

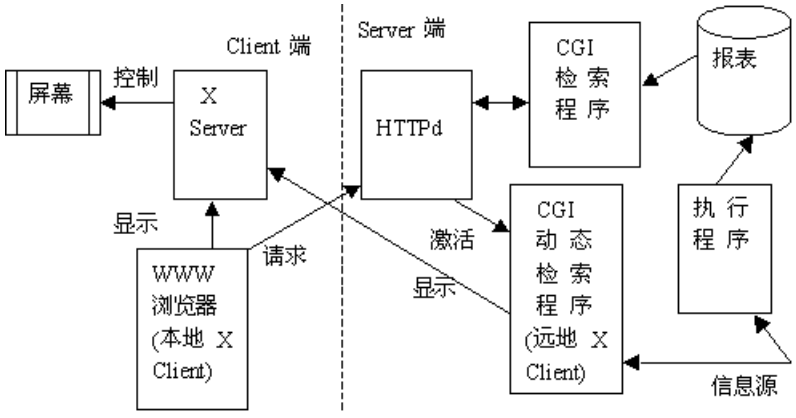


图 3. X Windows方式工作原理图

但是，为了将 X Windows 的甩屏功能集成在 Web 技术中实现，需要在 Server 端编写一个 CGI 程序。在这个 CGI 程序中需要设置甩屏幕所要的各种环境变量，生成出用户请求的内容，例如，动态跟踪的时间等等。最后，把用户请求的内容变成程序所能识别的形式，例如，程序的参数等，并把程序激活。

下面用一个例子说明。例如：“traffic”是一个能够连续画出当前网络流量情况的 X 程序，该程序放在“/usr/local/bin”目录下，该程序不需要参数，于是可以用 shell 编写一个名为“calltraff”的 CGI 程序用于调用“traffic”程序：

```
#!/bin/sh
echo "Content-type: text/html"
echo
echo "<html><body><center><h1>Traffic</h1></center>"
echo "Please wait!"
echo "</body></html>"
PROGRAM="/usr/local/bin/traffic"
LD_LIBRARY_PATH="/usr/openwin/bin"
DISPLAY="$REMOTE_HOST:0"
export LD_LIBRARY_PATH
export DISPLAY
$PROGRAM &
exit 0
```

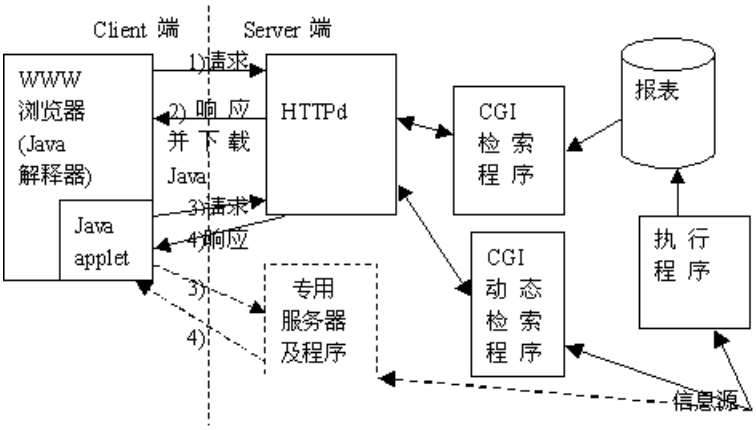
由于“traffic”是 X 程序，所以要定义动态连接库的环境变量，在这个程序中的环境变量“LD_LIBRARY_PATH”用于 Solaris 2.X。而宏“REMOTE_HOST”是 HTTP 的标准宏。

此外值得注意的还有：1) 该 CGI 程序必须显示些什么，否则 Netscape 会报错； 2) 本机必须定义“xhost”以允许 Web Server 所在的主机甩屏幕。

在“calltraff”编写完后，只要在某个 HTML 文档中定义“ ”就可以在 Web 中调用“traffic”来监视流量。

四、用基于 Java 的技术实现动态请求的 Web Client

图 4. Java方式工作原理图



事实上， Java 技术是最直观和最容易被想到的方法，因为如果说 CGI 提

供了在 Server 端动态执行程序的能力的话， Java 提供了在 Client 端动态执行程序的能力。

Java 方式的工作原理类似与 Netscape 附件的方法，只不过使用的语言是 Java，显示的结果在浏览器中，而不需要另外弹出一个窗口；而且程序本身是通过 Web 下载到浏览器上的。 Java 方式的工作原理如图4：

Java 中提供了各种 Internet 网络服务的包，其中也包括 Socket 等，只需要在 Java 程序中定义“import java.net.*”即可。

这种 Java 程序属于 applet，必需包括两个部分：网络传输部分和图形绘制部分。在本文中假定读者对 Java applet 的编写和图形绘制比较熟悉，所以仅简介网络传输部分。

在这种 Java applet 实现的类定义中，需要定义一个类型为 Socket 的变量、一个类型为 PrintStream 的变量和一个类型为 DataInputStream 的变量。例如：

```
Socket kkSocket = null;
PrintStream os = null;
DataInputStream is = null;
```

当利用 new 操作时就可以和给定的目标建立联系，例如：

```
kkSocket = new Socket("hostname", 80);
os = new PrintStream(kkSocket.getOutputStream());
is = new DataInputStream(kkSocket.getInputStream());
```

这样就建立了这个 Java applet 和名为“hostname”的主机上的 HTTPd 的连接。以后就可以通过 os 写请求，通过 is 得到 HTTPd 的响应。

与此同时， applet 中的绘图部分把从 Server 端得到的信息绘制在浏览器中。为了能够实现这一点，这个 Java applet 可以使用多线程的方式。

至于在 Server 端接受响应的程序可以是 CGI 程序(此时利用 HTTP 协议)，也可以是专用程序(可以 bind 在非标准的 port 上，此时交互的可以是自定义的信息)。编写程序所使用的语言可以多种多样，比如： C、 shell、 perl、 Java 等等均可。

此外，必须说明的是，并不是所有支持 Java applet 的浏览器都能够对这样的 applet 作出正确的解释，这取决于浏览器对 Java 解释的实现程度。

五、三种技术的比较

这三种技术实际上都是变通方法，也就是通过其他的技术和 WEB 技术的结合来实现的。下面通过几个方面来比较它们的不同。

1. 对环境的要求

Java 的方法对环境的要求最少，因为 Netscape 等浏览器可以直接支持 Java 的解释。 X Windows 的方法对系统环境的要求最多，它要求 Web 的 Server 端和 Client 端同时支持 X 协议，而且 Client 端还要能够显示 X 窗口。 Netscape 附件的方法对环境的要求介于两者之间，它仅要求 Client 端是某种窗口环境即可，比如， X Windows 或 Windows 3.X 或 Windows 95 等等。

2. 实现的难易程度

在这三种方法中，最容易实现的是基于 X Windows 的技术，因为，各种 Server 端的执行程序往往是 Unix 程序，而且常常有自带的 X 界面，这种方法只要写一个几行的 CGI 程序就可以将其连入 Web。

最复杂的是基于 Netscape 附件的方式。它要求同时在 Server 端和 Client 端编程，而且，自制的外挂附件必须支持 HTTP 协议。此外，如果要求跨平台特性，必须为不同的窗口界面开发外挂附件。

Java 的方法仅需要在 Client 端开发程序，当然， Server 端也要有能理解 Client 端请求的程序(常为 CGI 程序)，不过这个程序可以用各种方法实现，比如： Unix 的 shell，并不一定要使用 Java。

3. 配置的难易程度

Java 的方法配置最为简单，只需要浏览器支持 Java 即可。

X Windows 的方法，需要在 Client 端设置 xhost。

Netscape 的方法最为复杂，需要为每一个 Client 配置一个附件，而且需要在 Netscape 中作相应的设置。

4. 加载速度和执行时对 Netscape 的影响

Netscape 附件的方式加载最快，因为程序在本地， Java 要从 Server 下载程序，而 X Windows 要从 Server 甩屏幕，所以速度较慢。

在用 X Windows 方式显示动态信息时， Netscape 一直忙碌，只有关闭窗口后才会空闲下来。在用其它两种方式时，加载完毕以后 Netscape 就空闲下来，可以从事其它工作。

六、结束语

动态请求的 Web Client 技术是 WEB 技术的重要组成部分，在开发各种基于 WEB 的程序时，有很大的应用价值。本文中所提到的三种方法是我们在开发基于 WEB 的网络管理系统时总结出来的。有一定的借鉴意义。

Dynamical-Request Web Client

LU Sheng, GONG Jian
Southeast University 210096

Abstract: The importance of the web client which can request dynamically is discussed in the paper. Three different methods of dynamical-request web client are introduced and are compared from different aspects.

Key Words: computer network, Web-based technique.

作者介绍：
龚俭 东南大学教授， CERNET专家委员会成员，主要研究方向：开放分布式处理、网络管理、网络安全。
陆晟 东南大学硕士研究生，主要研究方向为网络管理和网络安全。