

手动网络拓扑的 B/S 模式实现

张磊 熊齐邦

(同济大学 计算机科学与技术系 , 上海 200331)

摘 要 比较了手动网络拓扑和自动网络拓扑的优缺点 , 指出手动网络拓扑仍是不可缺少的管理方式。讨论了 B/S 模式实现网络拓扑的两种方案 , 并选择网络和服务器的负担较小的方案来实现。介绍了 SVG , JavaScript , XML - RPC , JSP 和 Java 的 B/S 方式的实现方案 , 并解决了以下关键性问题 : 结点和线路如何与 SVG 图形元素对应问题 ; 远程数据如何传输 ; 客户端根据远程数据如何将状态实时显示在 SVG 图形上。给出了采用这一方法的手动网络拓扑的设计和实现 , 以及应用效果。

关键词 手动网络拓扑 ; B/S 方式 ; 可伸缩矢量图形 ; XML - RPC ; 简单网络管理协议

中图分类号 : TP311.52

文献标识码 : A

文章编号 : 1673 - 629X(2007)02 - 0149 - 04

Implementing Manual Network Topologies with B/S Mode

ZHANG Lei , XIONG Qi-bang

(Dept. of Computer Science and Technology , Tongji University , Shanghai 200331 , China)

Abstract Compare advantages and disadvantages between manual network topologies and auto network topologies , and point to that manual network topologies is indispensable for network management. Then discuss two solutions of implementing manual network topologies with B/S mode and choose the least weight solution of network and server to realize. Introduce the B/S - mode solution using SVG , JavaScript , XML - RPC , JSP and Java. The solution mainly resolves those problems : corresponding nodes and lines between SVG elements , transferring remote data , the client displaying real - time state in SVG refer to remote data. The implementation and the design are described , demonstrating a successful application to manual network topologies.

Key words manual network topologies ; B/S mode ; scalable vector graphics ; XML - RPC ; SNMP

0 引 言

网络拓扑信息是网络管理的基础 , 目前主要有两种方式来获得网络拓扑 : 手动绘制方式和自动发现方式。自动发现方式有一定的优点 : 减轻了管理员的工作 ; 当网络状态改变时 , 可以自动更新拓扑。但是相对于手动绘制方式来说 , 又有以下问题 :

(1) 结点发现缓慢 , 自动发现时发送的大量轮询包给网络带来负担 ;

(2) 无法屏蔽掉不需要的结点 , 如机房机器 ;

(3) 当轮询时结点发生故障 , 无法发现该结点 , 会得出错误的网络拓扑 ;

(4) 因为有些特殊的设置 , 导致获得错误的网络拓扑 , 比如有的网段设置了 VLAN (虚拟局域网) , 这时用自动发现方式很难获得正确的网络拓扑 ;

(5) 自动发现方式固定 , 如果获得了一个错误的网络拓扑 , 管理员很难手动进行修正 ;

(6) 要获得网络拓扑 , 必须遍历整个网络的状态 , 如果管理员只是关心一个大型网络的主干网状态 , 这种遍历不但浪费时间 , 而且效果很差。

手动绘制方式不存在以上问题 , 虽然它增加了管理员的工作量 , 但是对于网络管理来说 , 又是必不可少的。它使管理员能够关注重要的结点和链路信息 , 能够以和物理状态对应的方式来获得网络拓扑状态 , 能够很好地修正自动发现算法。在实际应用中 , 手动绘制方式和自动发现方式往往结合起来 , 这样能取得较好的管理效果。

网络管理系统^[1]一般采用 B/S 模式实现 , 有以下两种方案来实现网络状态显示。

1) 服务器生成模式 : 用户通过客户端浏览器向服务器发出服务请求后 , 服务器处理这一请求 , 并将网络拓扑图返回给客户端。

2) 客户端生成模式 : 用户通过客户端浏览器向服务器发出请求后 , 服务器只将所需的关键数据发送给

客户端,由客户端通过本地数据处理生成体现网络状态的网络拓扑图。

服务器生成模式的方案显示简单、交互性差,而且数据传输量大、客户等待时间长。采用客户端生成模式,可以让服务器只把关键的拓扑数据发给客户端,避免了数据冗余;对于一些无须经过服务端的操作(如放大缩小、拖动画面),也可以立即在本地执行,提高了系统的交互性。下面先简单介绍采用的相关技术,然后对这一方案的具体设计和实现作出描述。

1 相关技术简介

(1)SVG 技术^[2],全称为 Scalable Vector Graphics (可伸缩矢量图形)。它是 W3C 制定的、用矢量描述图形的 XML 应用标准。它有着许多的优点,比如可扩展性(scalable),动态的,交互性强。SVG 定义了丰富的事件,包括鼠标事件和键盘事件,只要对 SVG 进行相关的脚本编程就可以实现 SVG 文件的交互操作。SVG 还方便和 Javascript 结合在一起,使用 Javascript 对 SVG 属性的操作可以形成强大的显示效果。

(2)XML-RPC 技术^[3]是一种规范和一组实现方案,允许运行于不同操作系统、不同环境下的软件通过 Internet 进行远程过程调用(RPC)。RPC 是一种通过网络来发送基于消息的请求的方式,而 XML-RPC 是在 RPC 的基础上发展起来的。XML-RPC 采用 HTTP 为底层通信协议、XML^[4]为数据编码的格式。

(3)SNMP^[5]最初是 IETF 研究小组为了解决 Internet 上的路由器管理问题而提出的,随着技术和协议的发展,SNMP 成为网络管理方面事实上的标准。SNMP 定义了网元信息交换的规范^[7],它的目标是保证管理信息在任意两点中传送,便于网络管理员在网络上的任何结点检索信息、进行修改、寻找故障,并完成故障诊断、容量规划和报告生成。

(4)JSP/Servlet^[6]是 Sun 公司推出的基于 Java 的生成动态 Web 页面的技术。JSP 允许将脚本语言嵌入到 HTML 中,在 JSP 文件传输到客户端前,其语法在服务器端被解析,脚本程序被转换成 Servlet 进行处

理。Servlet 则是一种小型的 Java 程序,它扩展了 Web 服务器的功能,当接受客户端请求时,Servlet 开始执行,执行后生成动态的 HTML 数据发送到客户端显示。JSP/Servlet 结合后台的 Java 程序就可以实现强大的 B/S 模式程序。

2 设计方案

设计上整个系统可以分为 3 个部分:手动预处理、服务器端和客户端。主要解决了以下问题:结点和线路如何与 SVG 图形元素对应问题;远程数据如何传输;客户端根据远程数据如何将状态实时显示在 SVG 图形上。系统框架如图 1 所示。

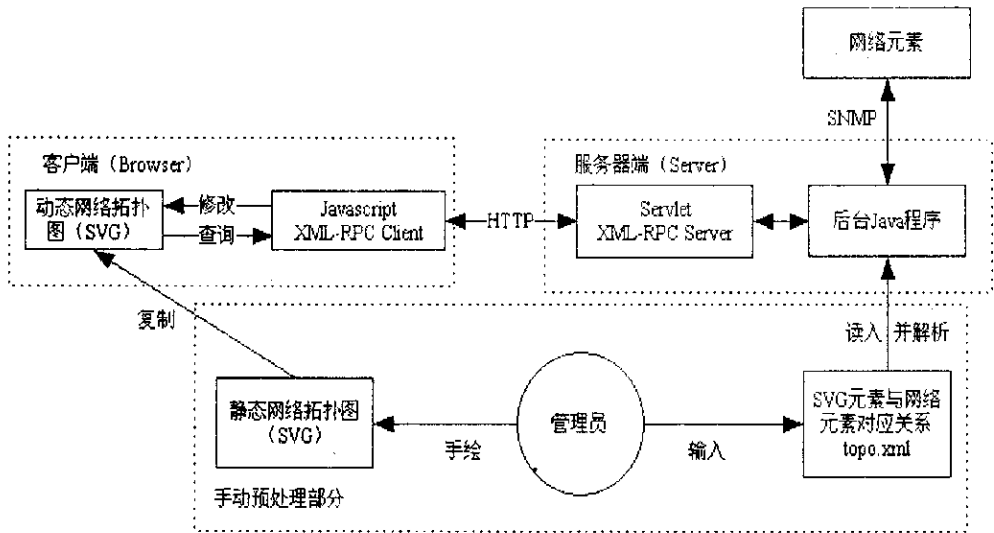


图 1 网络管理系统手动拓扑模块的框架

2.1 手动预处理

在这个过程中,管理员需要完成两个工作(1)静态网络拓扑图的绘制(2)SVG 元素与网络元素对应关系的输入。前者的主要流程是:浏览器中显示一个带地图的 SVG 画板,管理员可以将所绘静态网络拓扑图保存为 SVG 文件。后者的主要流程是:在地图上,点击相应网络元素和网络链路,弹出编辑窗口,管理员可以输入网络元素的 IP 地址和网络链路两端的连接信息。

在静态网络拓扑图的绘制过程中,SVG 画板的制作是一个难点。为了方便管理员使用,决定采用 B/S 模式实现,这需要使用 Javascript 技术来操作 SVG 结点。画板页面调用了—个 svg 文件,代码片断如下:

```
<svg>
<g id="bgControl">
    <image id="background" xlink:href="map.jpg" width=
"521" height="772" onclick="deSelector()" />
</g>
<g id="drawBoard" onmouseover="onSel(evt)" onmouseout =
```

```
" offSe( )"onclick=" selector( evt )">
< line id=" line_ 894503 "x1 =" 623 "x2 =" 673 " y1 =" 705 " y2 =
" 606 " stroke=" green " stroke - width=" 1 ">
</line>
</g>
</svg>
```

位于底层的 id 名为 bgControl 的<g>(组)元素导入了名为 map. jpg 的背景地图 ,位于上层的 id 名为 drawBoard 的<g>元素则定义了响应各种鼠标动作的 Javascript 函数。 Javascript 函数里 ,可以通过调用 getSVGDocument()来获得 SVG 对象的引用 ,并调用 getElementById()函数来具体操作一个 SVG 元素。这样管理员的操作就可以在 Javascript 函数的帮助下直接对 SVG 文本进行修改 ,达到了所见即所得的效果。

在 SVG 元素与网络元素对应关系的输入过程中 ,点击 SVG 元素后 Javascript 函数可以获得一个代表鼠标动作的 mouseEvt 对象 ,并通过 mouseEvt. getTarget()来获得鼠标点击 SVG 元素的引用。如果鼠标点击对象是结点 ,则弹出一个 JSP 页面要求输入 IP 地址 ;如果点击对象是链路 ,则弹出 JSP 页面要求输入链路一端的 IP 地址和连接端口索引(因为链接的状态可以由一端结点状态来反应)。 JSP 页面则把这种对应关系加工保存为名为 topo. xml 的 XML 文件 ,以便服务器端调用。数据库中保存了 IP 的相关信息及相应的 novelid ,为管理方便 ,在文件中只记录 novelid ,文件类似如下 :

```
< ? xml version=" 1.0 "encoding=" UTF - 8 "? >
< topo>
    < nodes>
        < node> < id> symbol_ 1127173620411 </id> < nodeid> 2
    </nodeid> </node>
    </nodes>
    < links>
        < link> < id> line_ 1126706818809 </id>
            < nodeid> 2 </nodeid>
            < ifindex> 1 </ifindex>
        </link>
    </links>
</topo>
```

2.2 客户端

有了静态网络拓扑图 ,还要将网络状态结合进去 ,才能形成动态网络拓扑图。为了尽可能地反映即时网络状态 ,这里采用了一个 Javascript 函数 setInterval(" updateSVG();" ,300000) ,每隔五分钟刷新一次客户端。刷新工作也就是分别调用函数 updateNode()和 updateLinks() ,向服务器端查询结点和链路的状态。

为了能与服务器端进行沟通 ,需要用 Javascript 在

客户端建立一个 XML - RPC Client ,主要部分的 Javascript 实现如下 :

```
...
xmlrpc=importModule( " xmlrpc " )//引入 Jsolait 的 XML - RPC
模块
nodeServer = new xmlrpc. ServerProxy( " /map/XmlRpcServlet " ,
[ " TopoHandler. GetNodeStatus " ] )//将服务器的 TopoHandler.
GetNodeStatus 通过 XmlRpcServlet 映射到客户端的 nodeServer
上
...
```

执行这段代码后 ,在 Javascript 里调用函数 nodeServer. TopoHandler. updateNodes() ,就相当于调用服务器端的函数 TopoHandler. updateNodes()。这样就可以取到 SVG 图形元素 id 对应的结点的网络状态。由此调用本地函数设置 SVG 元素的结点颜色 ,异常时改变为红色 ,起到报警效果。链路状态的处理和结点类似 ,从服务器端传来链路的使用率高低 ,据此标以不同的颜色 ,来反映网络链路的繁忙程度。

2.3 服务器端

服务器端主要有两个工作 :a. 建立 XML - RPC server ,将客户端函数和服务器端函数绑定在一起 ;b. 对网络设备进行查询 ,返回给调用函数。

建立连接的程序如下 :

```
xmlrpc= new XmlRpcServe( )//新建 XML - RPC 服务实例 ,将
TopoHandler 作为 XML - RPC 服务提供给客户端
xmlrpc. addHandler( " TopoHandler " ,new TopoHandler( ) )//执行
客户端的 XML - RPC 请求
byte[ ] result = xmlrpc. execute( request. getInputStream( ) )//将
执行结果返回给客户端
response. setContentType( " text/xml " );
response. setContentLength( result. length );
OutputStream out = response. getOutputStream( );
out. write( result );
```

对网络设备进行查询 ,需要解析 topo. xml。对于网络结点 ,取得其 IP 地址 ;对于链路 ,取得其链路一端的 IP 地址和连接端口索引。取得这些物理参数后 ,就可以使用 Java 程序发送 SNMP 包来获得网络结点和链路的状态。为了客户端能区分 ,这些状态还要与静态网络拓扑图中的 SVG 元素的 id 对应起来 ,因此调用 getSvgId(nodeid) 后返回 SVG 元素的 id 如 symbol_ 1127173620411 ,然后将取得的状态顺序连接成一个字符串发送给客户端。

链路的使用率的计算是个难点 ,因为 SNMP 中并没有直接提供这样一个参数。但是我们可以通过单位时间的丢包数来反映链路的使用率 ,在相同带宽条件下 ,单位时间的丢包数越高 ,说明链路越拥挤。在 SNMP 中 interfaces 组下面的 ifOutDiscards 参数记录了接

口当前的丢包数,ifSpeed 参数记录了接口当前带宽,隔一定时间取得 ifOutDiscards 参数的差值,再除以带宽值 ifSpeed,就相当于获得了链路的使用率。

3 实际应用

采用了以上手动网络拓扑的设计方案,所研发的网络管理系统取得了良好的效果。图 2 是网络管理系统在某大学校园网的应用(方框表示该链路正在绘制中),它有以下特点:

(1)与校园地图紧密结合在一起,直观地反映了网络结点和链路的物理位置,极大地方便了管理员的使用;

(2)网络状态可以实时反映,当网络结点异常时或链路繁忙时,能以不同颜色予以显示和报警;

(3)由于 SVG 的优良特性,该状态图可以自由缩放、移动,使得管理员既可以对整个校园网有一个宏观的了解,也可以对校园网某个区域有细节上的把握。

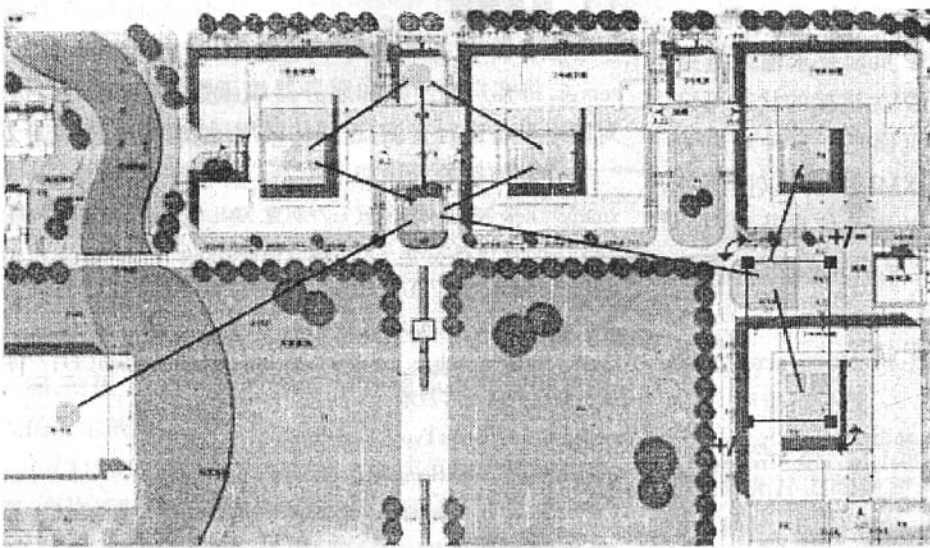


图 2 某大学校园网手动网络拓扑图

4 结 语

文中提出了一种 B/S 模式实现的手动网络拓扑的新方案,它能够让管理员结合实际情况手动编辑网络状态图,并能有效地与物理地图对应起来,B/S 的实

现方式,也让管理员在任何联网的计算机前,都可以对网络状态进行监控,提高了网络管理效率。

此方案在小型和中型网络的环境下,特别是在网络中集成了大量不同厂商的网络设备、网络链路数不多但是连接复杂的环境下,都有很好的应用。但是对于大型网络和网络链路连接简单的环境下,手动操作的工作量显得有些大,这时需要结合自动拓扑发现等技术,以取得更好的效果。

今后将在手动拓扑和自动拓扑发现结合等方面进行研究,以实现更完善的 Web 的网络管理系统。

参考文献:

- [1] Martin - Flatin J P. Web - Based Management of IP Networks and Systems[M]. Chichester: John Wiley & Sons, 2003.
- [2] W3C. Scalable Vector Graphics (SVG) 1.1 Specification[EB/OL]. 2005. <http://www.w3.org/TR/SVG/>.
- [3] Userland. XML - RPC Specification[EB/OL]. 2005. <http://www.xmlrpc.com/spec>.
- [4] Suresh R, Shukla P, Schwenke G. XML - based data systems for Earth science application[C]//Geoscience and Remote Sensing Symposium 2000. Proceedings. IGARSS 2000. IEEE 2000 International. [s.l.][s.n.]. 2000.
- [5] Stallings W. SNMP, SNMPv2, SNMPv3, and RMON 1 and 2[M]. [s.l.]: Addison - Wesley/Pearson, 2001.
- [6] The Java Servlet API, Sun Microsystems' White Paper [M]. US: Sun Microsystems, 1998.
- [7] Case J, Harrington D, Presuhn R, et al. Message Processing and Dispatching for the Simple Network Management Protocol (SNMP) [S]. RFC 2572, 1999.

(上接第 65 页)

- 性研究[J]. 计算机技术与发展, 2006, 16(5): 16-17.
- [2] 兰 草. 废钢铁企业生产系统优化模型[EB/OL]. 2003-06-07. <http://lw.xabest.com/2003/6-7/200367921110-1.html>.
- [3] 王 珊. 数据库技术与联机分析处理[M]. 北京: 科学出版社, 1998.

- 版社, 1998.
- [4] Vaserstein L N, Byrne C C. Introduction to Linear Programming[M]. [s.l.]: Prentice Hall/Pearson, 2005.
- [5] 周三元. 面向 Agent 的企业信息系统建模方法[D]. 北京: 北京理工大学, 2003.