一个管理业务接入的模型与实现 [1]

詹志强 邱雪松 孟洛明

(北京邮电大学程控交换技术与通信网国家重点实验室 北京 100876)

摘 要: 分析了电信管理网中管理业务接入的特点，基于这些特点，提出了一个采用可扩展标记语言(XML)技术和模型－视图－控制器(MVC)结构的管理业务接入模型，并给出实现的情况。

关键词: 电信管理网，网络管理业务接入模型，可扩展标记语言，模型－视图－控制器结构

中图分类号: TN913.2 **文献标识码**: A **文章编号**: 1009-5896(2004)01-0014-06

The Network Managenent Service Access Model and Its Implementation

Zhan Zhi-qiang Qiu Xue-song Meng Luo-ming

(*National Lab of Switching Tech. and Telecom. Networks, BUPT, Beijing* 100876, *China*)

Abstract The characteristics of the network management service access in the Telecommunication Management Network (TMN) are analyzed. According to these characteristics, the Network Management Service Access Model(NMSAM) applying eXtensible Markup Language(XML) and Model-View-Controller (MVC) architecture is presented. The implementation of the NMSAM is also given.

Key words Telecommunication Management Network(TMN), Network Management Service Access Model(NMSAM), eXtensible Markup Language (XML), Model-View-Controller (MVC)

1 引言

在基于电信管理网 TMN[1] 的网络管理系统中，网络管理业务接入就是指用户通过工作站(WS, Work Station)使用 TMN 所提供的管理业务的能力和方法。电信网络管理业务接入在整个网络管理系统中占有很重要的位置，它的设计与实现常常影响整个网络管理系统的成败。因此，在人们不断对网络管理系统的功能、质量、能力以及网络管理业务的接入手段等提出新要求的同时，网络管理业务接入的模型及其实现对建成一个可伸缩、可扩展、鲁棒性好的网络管理系统起到了至关重要的作用。文献 [2] 和文献 [3] 分别研究了基于应用服务器、基于 Web/Java 的管理业务接入方法。文献 [2] 采用了基于 TCP/IP 的前后端通信机制，这种机制容易造成前后端过分紧密的耦合；文献 [3] 是采用了基于 CORBA 的前后端通信机制，但由于 CORBA 产品的普遍低效和对单一厂家的严重依赖性，不同厂商 CORBA ORB 脆弱的互操作性，造成这种方式不太适合满足从互联网中灵活接入网管系统的需求，又由于程序员们必须处理数据排列和数据类型所需的协议唯一的消息格式规则，使开发和升级工作量非常大 [4] 。

本文第 2 节首先分析了网络管理业务接入的特点，基于这些特点，在第 3 节提出了一个采用可扩展标记语言 (XML, eXtensible Markup Language)[5] 技术和模型－视图－控制器 (MVC, Model-View-Controller)[6,7] 结构的网络管理业务接入模型 (NMSAM, Network Management Service Access Model)，第 4 节给出了实现的情况，文章最后指出所需的进一步工作。

[1] 2002-08-14 收到，2002-11-18 改回

国家杰出青年科学基金 (No.60025104)；国家自然科学基金重大研究计划项目 (No.90204002)；青年科学基金项目 (No.60202003) 资助课题

2 网络管理业务接入的特点

网络管理业务接入涉及到 3 个实体和两个接口：用户；实现 WSF(Work Station Function) 的工作站 WS；实现 OSF(Operating System Function) 的运行系统 OS；用户和 WS 之间的 G 接口；WS 和 OS 之间的 F 接口。其中 G 接口是提供给网络管理系统使用者使用的网络管理系统的图形用户界面部分，F 接口要实现的功能是将在 TMN 内部使用的数据格式转换为适合人机界面显示的数据格式。对于网络管理系统的使用者来说，管理业务接入就是网管系统的全部，它的设计和实现的好坏直接影响到用户对网管系统的理解和应用，一个质量低劣的用户界面会直接影响运营商的运营效率。对于网管系统开发人员来说，他们本应当关注网管的各种核心业务，可是很多时候他们花费大量时间和精力去重复开发低效的用户界面。

一个低效的管理业务接入方式通常存在的问题表现在：缺乏个性化的图像，不直观的界面效果，不开放且笨重的前后端连接，困难而低效的维护和升级工作等等。产生上述问题的原因在于：界面程序中存在不合理的耦合，包括前端和后端的耦合 (表现在前端掺杂了后端的逻辑，后端死绑在前端)、管理对象和图形对象之间以及功能模块之间过分紧密的耦合；缺乏一个清晰的面向对象的程序结构等等方面。

TMN 没有定义网络管理业务接入的方式，在设计和实现网络管理业务接入系统时，要尽力避免上述的各种问题。通常，一个高质量的网络管理业务接入系统应该具有以下特点：

(1) 直观的界面效果，通常可以将各种管理对象用以下 4 种方式显示出来：包含节点和连接的网络拓扑图，包括板卡、端口等的设备面板图，资源树形图，二维表格；

(2) 支持远程接入：因为电信网的用户可能分布在不同的地方，所以接入方式要支持远程接入，用户通常希望可以通过互联网接入网管系统；

(3) 支持多用户接入：接入管理系统的用户可能同时会有多个；

(4) 支持不同操作平台的接入：原因是接入用户所使用的接入设备可能不同；

(5) 支持同步操作：支持同步操作，以保证数据完整和一致；

(6) 确保系统稳定和易于维护和升级：随着用户对新的管理需求的提出，可能引起网管系统的增加、修改，所以系统要易于维护和升级。

3 网管业务接入模型

基于以上网管业务接入的特点，本文提出了一个采用 XML 和 MVC[6,7] 结构的网络管理业务接入模型。该模型采用面向对象的技术，遵循 MVC GUI 设计模式，采用基于 XML 的 F 接口。

3.1 NMSAM 的逻辑结构

3.1.1 NMSAM 的结构和工作流程　图 1 给出了 NMSAM 的逻辑结构和工作流程，实现 NMSAM 的功能实体是后台 OSF 和前端 WSF，WSF 和后台 OSF 之间采用基于 XML 的接口，图中的箭头表示了信息传递的方向，菱形和椭圆形表示 NMSAM 中的组件，方块表示各种对象。

为了说明 NMSAM，首先需要介绍几个概念：(1) 管理对象，就是指各种网络管理对象实例，它们是由后台 OSF 发送数据到 WSF，并由 XML 解析器生成对象实例；(2) 图形组件，在图 1 所示的 NMSAM 的逻辑结构和工作流程中，NMSAM 的图形组件 (如图 1 中大椭圆所示)，都采用典型的 MVC[6,7] 结构，图形组件可以将管理对象表示为各种不同格式，如拓扑图、设备面板图、树形图、表格，图形组件 MVC 结构包括了 Model(模型)，View(视图)，Controller(控制器)3 个部分，按照图形组件表示对象的不同格式，将图形组件分为拓扑图组件、设备面板图组件、树形图组件、表格组件等 4 种类型；(3)Model，MVC 结构中一种对象，负责维护数据并提供数据访问方法；(4)View，负责绘制 Model 的部分数据或所有数据的可视图；(5)Controller，负责处理事件；(6) 代理对象，即 Model 中的数据，是管理对象经过适配器适配而来的，代理对象包含了将管理对象在不同图形组件中描绘出来所需的足够信息；(7) 图形对象，即 View 中的可视对象，如标签、图形、文字或者它们的组合等等，图形对象是各种代理对象的图形化表现。

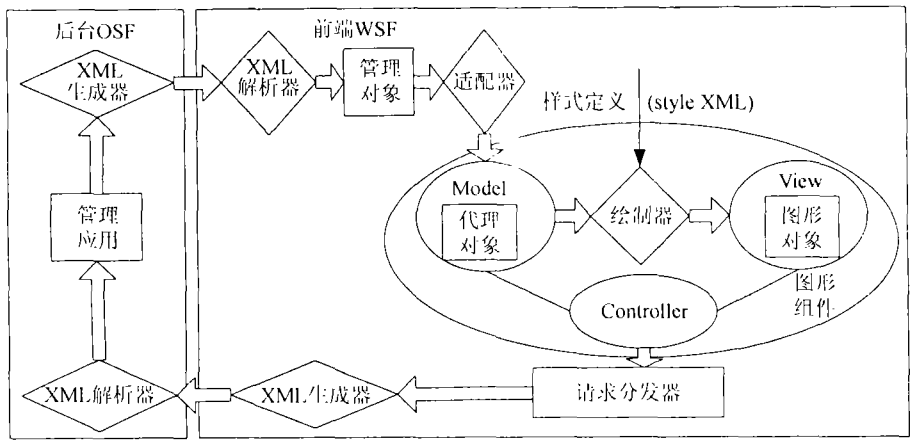

图 1 NMASM 的逻辑结构和工作流程

NMSAM 中图形组件中的 Model 中容纳了各种代理对象，不同的图形组件需要不同的代理对象。例如拓扑图组件需要表示拓扑节点和连接的代理对象，树形图组件需要表示树结点的代理对象；同样地，一个将要显示在拓扑图中的管理对象，将会被适配成拓扑节点或者连接，作为代理对象存放在拓扑图组件的 Model 中。当图形组件 Model 中添加、删除、修改代理对象时，会触发绘制器在 View 中添加、删除、修改相应的图形对象，例如在拓扑视图中添加一个节点或者连接。如果采用 Java 实现，NMSAM 图形组件中的 Model 可以继承 Java Swing[5,6] 模型，当 Model 中的代理对象发生任何改变，Swing 使用监听器来通知相应的 View，后者将及时更新显示。当一个代理对象有多个 View 时，这种方式保证了多个 View 的同步更新和一致性。

图形组件中的绘制器根据 XML 格式的样式定义来创建、删除、修改 View 中的各种图形对象，这些样式定义种类繁多，例如拓扑图中节点标签的字体大小、颜色、图标、告警显示方式等等。图形组件中的 View 是图形对象的容器，并提供了对图形对象的访问方法。

图形组件中的 Controller 负责处理用户的输入，创建或修改代理对象，并且将修改在 View 中体现出来，每个 View 有一个 Controller 。Controller 一直决定哪些 View 和 Model 应该在某个给定的时刻是活动的，它一直负责接收和处理用户的输入，来自用户输入的任何变化都被从 Controller 送到 Model, View 或者请求分发器，例如拓扑图中节点的暂时移动，Controller 可以将其送给 Model 处理，而如果要在 View 中创建新的节点，则应该将这个请求发送给请求分配器，由后者送到后台 OSF 。

作为一种面向对象的 GUI 设计模式，MVC 结构可以清晰地将管理对象，Model 数据 (代理对象) 和它们的图形表现 (图形对象) 以及用户与它们的交互区分开来。引入代理对象，是为了达到可以将管理对象同时显示在多个不同图形组件中的目的，这样就避免将管理对象直接作为图形组件中的代理对象，而必须将管理对象通过适配器转换成不同图形组件中的代理对象，实现相同管理对象同时有多个不同格式的显示，如某个交换机可以同时显示在拓扑图和表格中，拓扑图和表格分别对应一个图形组件实例。除此之外，适配器还可以有过滤功能，这样一来，适配器将只把符合条件的管理对象转换为代理对象。

NMASM 的显示流程如下：(1) 后台 OSF 向前端 WSF 发送以 XML 格式描述的各种管理数据；(2)XML 解析器完成数据解析，生成各种管理对象，如告警、网元、链路、机架等等，这些对象是 WSF 中其它组件可以操作或者使用的对象；(3) 适配器将管理对象转换成各种图形组件所需的代理对象；(4) 图形组件中的绘制器将代理对象转换成 View 中的图形对象；(5)View 将各种图形对象显示出来，呈现在用户面前。

当用户在界面上做各种操作时，都会被图形组件的 Controller 接受下来，并将其发送到 Model, View 或者后台 OSF, 从而更新 View, 流程如下：(1) 用户在 View 中创建、删除、修改一个对象；(2) 该 View 关联的 Controller 接受这个动作，并判断这个动作应该转发给谁；(3)

如果需要发给后台 OSF 处理，则将调用请求分发器提供的 API； (4) 请求分发器确定接受请求的后台模块，完成请求的包装； (5)XML 生成器将请求转换为 XML 格式的请求，并发送给相应的后台 OSF； (6) 后台 OSF 中的 XML 解析器进行解析，交管理应用完成请求的处理，然后由 XML 生成器将处理结果以 XML 格式发给 WSF； (7)WSF 按照前面所说的 NMASM 的显示流程完成界面的更新。可以看到，采用这种流程，使得前后端的逻辑得以清晰分开，降低了前后端耦合度。

3.1.2 基于 XML 的 F 接口　NMASM 中 WS 和 OS 之间的 F 接口是基于 XML 的。根据界面操作需求，该接口设计主要包括了对管理对象的增加、删除、修改、查询等操作。

电信网络管理中涉及的管理对象类是各种各样的，这么多的对象类如果全部用代码来定义将会是一个巨大的工作量，更不用说维护和修改了。采用 XML 技术可以大大降低这个工作量。NMASM 中，管理对象类和实例都是用 XML 语言来描述，程序的工作仅仅是解析这个描述文件，这样就实现了管理数据与程序的完全分离，使程序结构清晰，减少了工作量。在 XML 文件中定义了管理对象类的类名、各属性名及其类型。对于管理对象实例，描述其具体的属性值，使用关键字 "addObject", "removeObject", "update" 等定义对管理对象实例的增、删、改。下面是描述增加一个管理对象类的例子，这是 OSF 收到 WSF 的请求后，发送给 WSF 的 XML 文件。有关 WSF 发送给 OSF 的各种请求的 XML 定义，与此类似。

```
<classes>
 <!- Node ->
 <class>                                          定义对象类
    <name>Node</name>                             --类名
   <attribute>                                    --属性描述
    <name>UserName</name>                         --属性名
    <javaClass>java.lang.String</javaClass>       --属性类型
   </attribute>
   <attribute>
    <name>position</name>
    <javaClass>Metarnet.Point</javaClass>
   </attribute>
   <attribute>
    <name>AlarmLevel</name>
    <javaClass>java.lang.Integer</javaClass>
   </attribute>
   </class>
</classes>
<addObject id="demoNode">                         --添加对象实例
 <class>Node</class>
    <attribute name="UserName">node1</attribute>
    <attribute name="position" javaClass="Metarnet.Point">
        < x > 300 < /x >
        < y > 156 < /y >
    </attribute>
    <attribute name="NewAlarm"> 1 </attribute>
 </addObject>
```

3.2 NMSAM 的物理结构

为了满足网络管理远程、多用户、不同操作平台接入的需求，并确保系统稳定和易于维护升级，NMSAM 采用了如图 2 所示的基于 WEB 和总线结构的物理结构。在实际网络管理系统开发中，通常将 WSF 和 OSF 作为独立分布的应用实体进行开发，而 OSF 通常也被划分为多个管理功能模块进行开发。为了方便各模块实体之间的通信，NMSAM 采用通信总线进行通信，如 CORBA, DCOM 等。WSF 功能采用基于 WEB 方式实现，用户界面采用 Java Applet, 存在于 WWW 服务器中，用户使用 WWW 浏览器将其下载到本地运行。浏览器在解释运行 Applet 时，Applet 会通过通信总线接口 (即 F 接口) 与各管理功能模块通信。

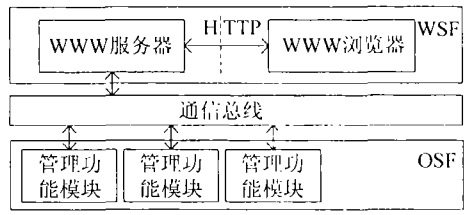

图 2 基于 WEB 和总线结构的 NMASM 的物理结构

3.3 NMASM 的特点

NMASM 具有以下特点：

(1) 采用了 MVC 结构，遵循使界面中各组件分离的独立性、自解释、弱耦合的原则，很好地表达了用户的交互和系统模式，方便用多个不同格式的视图来显示多套数据，使代码重复达到最低， MVC 结构封装了很多其它设计模式的优秀特性，满足了网管界面对灵活性、可扩展性、可维护性、高交互性的要求。同时由于分离了模式中的控制和数据表现，降低了系统复杂度，可以缩短产品开发时间。

(2) 采用了基于 XML 的 F 接口，相对于基于 API 的接口和基于 CORBA 的接口，显著降低了前后端的耦合程度。随着开放的分布式操作技术广泛应用于 TMN 中， TMN 必须继续保持其优点并逐渐吸收新的技术来保证网络互操作性的高效、可靠、安全运行。XML 由于其使用的灵活性、简单性、易扩展性，易于传输、安全性等特点正被广泛应用于 TMN 中。TMN 的管理和代理之间允许使用 XML 进行信息传输和信息描述，同样在 F 接口之间也可以引入 XML 技术，从而获得上述各种好处。

(3) 采用了基于 XML 的样式定义，这种方式简便灵活，方便随时修改样式，大大减少了编程和维护工作量，对 XML 样式的定义者来说，不需要涉及具体的编程，提高了效率。

(4) 大量采用 XML 技术，节约了开发时间和成本。

(5) 采用了基于 WEB 和总线结构，显著降低了开发的难度和费用，简化了客户端的安装配置，无需不断升级客户机的软硬件。

4 NMSAM 的实现

NMSAM 实现的主要关键技术有：Java 图形界面开发技术[8,9]、XML 处理、软件总线等。目前处理分析 XML 文件的几个主要应用程序接口规范有 DOM(Document Object Model)[10], SAX(Simple API for XML)[11] 和 XSLT(XSL Transformations)[12], 它们提供了创建、分析、处理 XML 文件的基本方式。

我们在多专业网综合网管原型系统中采用了 NMSAM 。原型系统运行在 UNIX 工作站上，WSF 和 OSF 分别采用 Java 和 C++ 开发，使用 CORBA 作为通信总线，采用 JDOM[13](一种 DOM 的 Java 实现) 和 Xerces C++ Parser[14](一种 DOM, SAX 的 C++ 实现) 进行 XML 文件处理。原型系统的界面 WSF 与后台各模块采用 CORBA 总线进行连接，主要方式是：界面使用 CORBA 请求向后台各模块发送请求，后台模块通过 Event Service 向界面发送各种界面更新通知，其中请求和通知的内容是前面提出的 XML 文件。原型系统的 WSF 中定义了本地网运行商 (分公司) 、台站、网元、链路、路径、终端点、端口、告警、性能通知等管理对象，WSF 向后台的各种请求操作，如管理对象的创建、删除、修改、查询、拓扑查询等操作，都被封装成 XML 格式的字符串，并作为 CORBA 调用的参数传递给后台模块，后台模块再将各种应答封装成 XML 格式的字符串，通过 Event Service 发送给 WSF 。原型系统将所有参数都封装成 XML 格式的字符串进行传递，这样做的最大好处是：当 WSF 和后台接口之间的参数传递发生变换和升级时，只需要修改 XML 字符串，不需修改 IDL 接口，避免了重新编译 IDL 接口文件和重新编写整个前后端程序，减少了修改和升级工作量。

5 结论和进一步的工作

本文分析了网络管理业务接入的特点，基于这些特点，提出了一个采用 XML 技术和 MVC 结构的网管业务接入模型 NMSAM，经实践证明，该模型确保了网管业务接入部分的可伸缩、可扩展和好的鲁棒性，降低了前后端的耦合，降低了开发费用和难度，满足了管理业务接入的各种要求，对网管业务接入的开发具有指导作用。

基于 XML 的 F 接口方式，为采用基于 HTTP 的 XML 传输技术 SOAP[15] 和 Web Services[16] 技术提供了可能，从而获得大量好处。Web Services 技术提供了一个全新的编程模型来利用开放因特网标准建立分布式应用程序。这一全新的分布式计算解决方案，利用因特网技术的开放性解决了许多 CORBA 和 DCOM 的面临的互操作性问题，由于大量使用了成熟的 Web 技术，可大大降低开发成本。同时，Web Servics 使用 HTTP 来实现防火墙友好和不确定有效负载，可以保证网管业务接入的高效和安全性。随着 Web Servics 的不断完善和成熟，基于 Web Servics 的网管接入技术是需要继续研究的问题。

参　考　文　献

[1] ITU-T Rec. M.3010, Principles for a telecommunication management network, 2000.

[2] 张积友，雷湘侣. 基于应用服务器的 TMN 管理业务接入方法. 数据通信，2001, (1): 50–51.

[3] 郭涞炜，刘铁钢，孟洛明. 基于 WEB 的 TMN 管理业务接入模型. 电子科技导报，1999, (6): 19–22.

[4] Schmidt D C, Vinoski S. Object Interconnections, CORBA and XML. http://www.lmihoo.com/xml/index.html, 2001.

[5] Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation, 2000.

[6] Burbeck S. Applications programming in smalltalk-80(TM), How to use Model-View-Controller (MVC), http://st-www.cs.uiuc.edu/users/smarch/ st-docs/mvc.html, 1992.

[7] Krasner G E, Pope S T. A description of the Model-View-Controller user interface paradigm in the smalltalk-80 system. *Journal of Object Oriented Programming*, 1988, 1(3): 26–49.

[8] Geary D M. Graphic Java Mastering the JFC Volume II: Swing, 3rd Edition, Prentice Hall PTR, 1999.

[9] Trail: Creating a GUI with JFC/Swing, http://java.sun.com/docs/ books/tutorial/uiswing /index.html, 2001.

[10] Document Object Model (DOM) Level 2 Core Specification, W3C Recommendation, 2000.

[11] SAX 2.0: The simple API for XML, http://java.sun.com/webservices/docs/1.0/tutorial/doc/JAXPSAX.html, 2000.

[12] XSL Transformations (XSLT), W3C Recommendation, 1999.

[13] JDOM API, http://www.jdom.org/docs/apidocs/, 2001.

[14] Xerces C++ Parser, http://xml.apache.org/xerces-c/, 2001.

[15] Simple Object Access Protocol (SOAP) 1.1W3C Recommendation, 2000.

[16] Web Services Website, http://www.webservices.org.

詹志强：　男，1976 年生，博士生，研究方向：多专业网综合管理与通信软件.

邱雪松：　男，1973 年生，副教授，研究方向：通信软件与网络管理体系结构.

孟洛明：　男，1955 年生，教授，博士生导师，研究方向：TMN、通信软件及网络管理.