

网络环境下几种流行的编程方法

孙牧 汪齐贤 (北方交通大学通信控制系 100044)

摘要:本文介绍了四种在不同网络环境下不同层次上的编程方法的原理和特点。

关键词:网络编程 NetBIOS IPX SPX SOCKET 面向对象 Java

本文通过介绍四种实用、流行的网络应用编程方法的原理,以期使读者对它们的特点,应用环境进行比较,以便合理选用。

一、NetBIOS 程序设计

NetBIOS(Network Basic Input/Output System)是由IBM公司开发的网络高层编程接口(居于OSI参考模型的第5层)。由于它并没有规定特定的下层网络通信协议,因此其提供的服务可以在不同的网络技术上实现。目前,NetBIOS已成为事实上的局域网标准,很多著名的局域网厂家(如3COM, Novell等)的产品都提供了它的仿真软件,因此NetBIOS在异种机联网方面有很好的兼容性。

NetBIOS提供四种类型的服务:

- 名字服务:用于产生和检测组名和个体名字以及删除个体名字,这些名字既能是硬件名,也能是系统名,是

协议实体的标识。

- 数据报服务:无连接传输,数据报中信息字段不超过512字节。

- 会话服务:面向连接,保证可靠,顺序地提交报文,最大长度为64K字节。

- 普遍服务:用于重新设置适配器状态,可能时中止应用命令等。

NetBIOS的每一种服务对应若干条命令,应用程序调用这些命令时,首先要建立一个网络控制段(NCB),它被用于应用程序和NetBIOS之间传递命令,参数以及返回结果。

由于大部分NetBIOS命令有等待(WA)和非等待(NW)两种工作方式,相应地,应用程序有以下三种结构模式:

- 等待方式:应用程序发出WA命令后,挂起,直至NetBIOS命令完成后才继续执行后边的指令。

·循环检测方式:发出 NW 命令后返回,然后循环检测 NCB 中的最终返回代码字段,以判断命令执行结果。

·异步处理方式:发出 NW 命令后立即返回,执行后继指令。当 NetBIOS 命令在后台完成后,会根据一个特殊字段(4 个字节,仅在此模式下有效)找到 Post 程序的入口地址,自动进行后继处理。

无论采用哪一种结构模式,调用 NetBIOS 的应用程序都应分为以下三个步骤:

(1)构造网络控制块(NCB);

(2)把 NCB 的块址写入 ES:BX 寄存器中,作为指向 NCB 的远程指针。

(3)调用 DOS 中断 5CH。

二、IPX/SPX 程序设计

IPX/SPX(Internetwork/Sequenced Packet Exchange)是 Novell 公司开发的网络通信协议,只能用在 Netware 网络操作系统中。它们特别适合开发两站之间的对等通信(Peer to Peer),由于减少了文件服务器这一中间环节,从而提高了数据通信的实时性和灵活性。IPX/SPX 是 Netware 为用户提供的 4 个网络访问接口的一部分,见图 1。

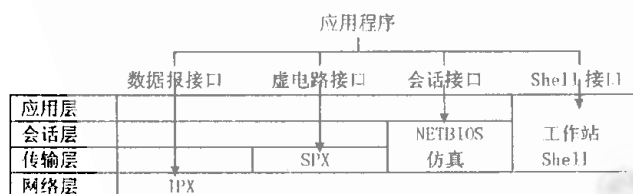


图 1

IPX 负责在网络之间进行路由选择和转发分组,而 SPX 则在 IPX 服务的基础上提供了分组排序和重传的功能,从而保证顺序可靠地提交报文。

与 NetBIOS 类似,IPX/SPX 也是用一组命令来提供服务的。在调用命令之前,也都要先建立一个事件控制块(ECB),这种数据结构提供了 IPX/SPX 操作所需的各种信息。同样 ECB 也可以提供一个事件处理程序(ESR)地址供 IPX/SPX 命令完成后执行。

IPX/SPX 安装后成为内存驻留进程,所用命令均在

后台执行,命令执行后会在 ECB 的完成代码字段给出执行结果。

三、Socket 程序设计

Socket(套接字)最初是由加利福尼亚大学 Berkeley 分校为 UNIX 操作系统开发的网络通信接口,随着 UNIX 的广泛使用,它已成为当前最流行的网络通信应用程序接口之一。目前也出现了将 Socket 移植到 DOS 和 Windows 下的标准,如:Windows Socket 规范。

概念上讲,Socket 是可以被命名和寻址的通信端点,使用中的每一个 Socket 都有其类型和一个与相连的进程。Socket 采用客户机/服务器的通信机制,提供了一系列的系统调用,使用它们可以实现 TCP/UDP、ICMP 和 IP 等多种网络协议之间的通信。

Socket 有三种主要类型:

·Stream Sockets 接口:提供面向连接的服务,可通过内置的流控机制解决数据的拥塞,可发送任意长度的数据。

·Datagram Sockets 接口:无连接,数据通过相互独立的包传送,包无序,可靠性无保证,长度有限。

·Raw Sockets 接口:允许对低层协议如 IP 和 ICMP 直接访问,主要用于新的网络协议实现的测试,下面通过一个面向连接的传输发生的典型情况说明 Socket 网络通信的实现。

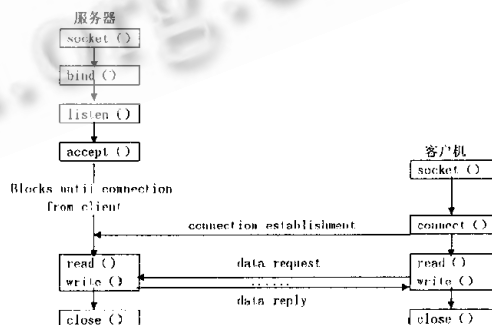


图 2

由图 2 可看出,客户机和服务器的关系是不对称的。服务器首先启动,然后在某一个时间启动客户机与服务器建立连接。服务器和客户机开始都必须调用 socket() 建立一个套接字(Socket),然后服务器调用 bind() 将套接

字与一个本地网络地址捆扎在一起,再调用 Listen()使套接字处于一种被动的准备接收状态,同时规定它们请求队列长度,之后服务器就可以调用 accept()来接收连接了。客户机在建立套接字之后,便可以通过调用 connect()和服务器建立连接。连接建立后,客户机和服务器之间就可以通过 read()和 write()发送,接收数据。数据传送结束后,双方调用 close()关闭套接字。

四、面向对象技术在网络编程中的应用

网络通信是一个复杂过程,通常把整个过程划分为数个功能层次,每一层的功能具体,固定,且只有一定的服务访问点(SAP),上层访问 SAP 时,无须知道功能是如何实现的。层与层之间有许多类似的功能操作(如组帧、包或报文,握手等)。正是这种功能的抽象化、封装化、多态性和继承性使得面向对象技术与网络编程有着天然的联系!

请看下面这个 OSI 层的一个最简单的抽象——Layer 类。

```
class layer{
public:
    layer(int aType); //构造函数
    layer(); //析构函数
    virtual void Command (int cmd , int &errorCode, int
reason);
    virtual void Command (int cmd, int &errorCode, data-
Packet * data);
```

/cmd - 执行某原语的标识, data... 指向要发送/接收的数据包的指针。

//reason - 执行该原语的原因标识。

private: int type; //层类型

layer (nPlus; //指向上 一层的指针

layer * nMinus; //指向下一层的指针

Boolean BlockXmit; //阻塞发送标识

BOolean blockRecv; //阻塞接收标识

}

通过定义这样一个抽象类,可以实现层中通用的和

不变部分的接口。此外,我们可以定义更多的公共成员函数,来隐藏对数据成员的使用,像阻塞/疏通、设置和获得(N+1)层和(N-1)层指针的函数,都可以在抽象类中实现,其他所有子类都将自动继承这些函数的功能。还可以通过在子类中重载 Command()函数实现不同层次的原语。

可以看出,如何定义符合实际协议层次结构的抽象类库,以及如何对类中不同数据类型进行操作是面向对象程序设计中重点要研究的问题,也是一个难点,它令一般的网络用户望而却步。令人兴奋的 Java 出现后,形势开始有所转变。

JAVA 语言是适合于分布式计算环境的完全的面向对象的编程语言,它提供了多个可用于访问各种标准 Internet 协议(如: Telnet、FTP、NNTP 和 WWW)的类库,利用它们,编程人员可以很方便,快捷地编写出功能完善的 Internet 应用程序。

五、结束语

综上所述,NetBIOS 在局域网环境下具有很好的兼容性,IPX/SPX 在 Netware 网中很适于开发站到站的实时通信程序,Socket 则可用来实现基于 TCP/IP 协议的可靠的、双向的、点对点的、基于流的连接。而基于面向对象技术的 JAVA 语言能开发出高效、简洁、跨平台的 Internet 应用程序。当然,要真正开发网络应用程序时,我们还需对这些技术的细节进行反复地揣摩和实践!

参考文献

- [1] 《网络环境下的 C 编程技巧及实例》, 方敏 等编著 西安交通大学出版社
- [2] 《Windows sockets 网络程序设计指南》 蒋东兴 等编著 清华大学出版社
- [3] 《Data Communication Using Object - Oriented Design and C + +》 Anil. Ananthaswamy McGraw - Hill Inc.
- [4] 《Java 程序设计实用指南》 廖卫东 等编著 机械工业出版社

(来稿时间:1998 年 6 月)