# 基于 Socket 的网络编程技术及其实现

刘 骏， 颜钢锋
(浙江大学 电气工程学院,浙江 杭州 310027)

**摘 要**: 在介绍 TCP 协议客户端和服务器端进程通信流程和具体实现的基础上,以 Delphi 为环境编程语言,说明了在 Windows 下利用 Socket 进行网络编程的方法和特点.给出了一个用局域网进行文件传输和系统远程控制的实例.

**关键词**: Socket;网络编程;远程控制;TCP 协议
**中图分类号**: TP 311.52 **文献标识码**: A

## Network Programming Technique and Its Realization Based on Socket

LIU Jun, YAN Gang-feng
(College of Electrical Engineering, Zhejiang University, Hangzhou 310027, China)

**Abstract:** The paper introduces the process communication procedure and its specific realization between the client and server based on TCP protocol. Based on the introduction, the paper propose the methods and features of network socket programming using Delphi. An instance of file transfer and distant system control is also presented.

**Key words:** Socket; network programming; remote control; TCP protocol

Socket 是建立在传输层协议(主要是 TCP 和 UDP)上的一种套接字规范,最初由美国加州 Berkley 大学提出,为 UNIX 操作系统开发的网络通信接口,它定义了两台计算机间的通信规范(也是一种编程规范). 如果两台计算机是利用一个"通道"进行通信,那么这个"通道"的两端就是套接字. Socket 屏蔽了底层通信软件和具体操作系统的差异,使得任何两台安装了 TCP 协议软件和实现了 Socket 规范的计算机之间的通信成为可能. Socket 接口是 TCP/IP 网络最为通用的 API,也是在 Internet 上进行应用开发最通用的 API[1].

文中介绍了 Socket 程序设计的基本原理,并结合实例介绍了 Socket 程序设计的基本方法,给出了局域网文件传输和计算机系统远程控制的实例.

## 1 Socket 基本原理

在 Windows 网络编程中,套接字接口主要有 3 种类型:流式套接字,数据报套接字以及原始套接字.流式套接字定义了一种面向连接的服务,实现了无差错无重复的顺序数据传输,无长度限制. 数据报套接字接口定义了一种无连接的服务,数据通过相互独立的报文进行传输,是无序的,并且不保证可靠. 原始套接字允许对低层协议 IP 或 ICMP 直接访问,主要应用网络协议的测试,例如 Windows 带的 Ping 程序,就是通过 ICMP 实现的[2].

在现在的网络应用中,通信双方最常见的交互模式便是 Client/ Server 模式. 客户/服务器模式通常采用监听/连接的方式实现. 服务器端应用程序在一个端口监听对服务的请求,也就是说,服务进程一直处于休眠状态,直到有一个客户对这个服务提出了连接请求,此时服务线程被"唤醒"并为客户提供服务,即对客户的请求做出适当的反应[3].

采用面向连接的协议(如 TCP)时,服务器处理的请求比较复杂,并不是简单的请求应答所能够解决的,而且大多数 TCP 服务器是并发服务器,因此需要经过反复的交互[2]. 使用面向连接的协议时,典型的套接字接口调用流程如图 1 所示.

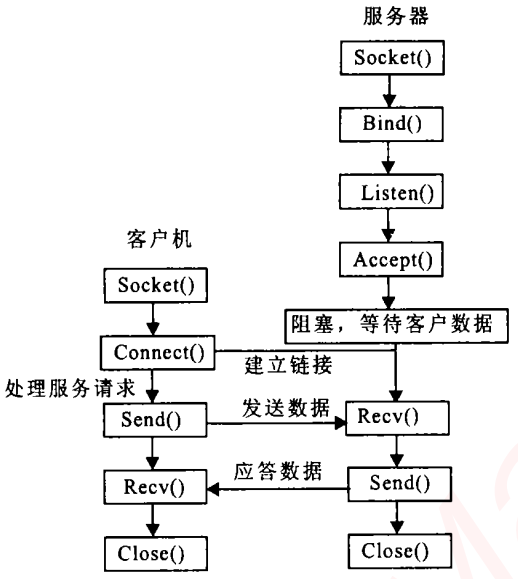


**图 1 Socket 通信过程示意**

**Fig. 1 The diagram of Socket communication procedure**

具体实现时,服务器进程首先在约定的端口处开启一个监听 Socket,负责监听客户端的请求,用 Accept() 循环从接受队列依次取出每一个客户进程,连接后生成新的 Socket,此时服务器 Fork() 出一个子进程专门处理该客户,父进程则关闭新的 Socket,继续处理下一个客户进程. 而服务器监听用的 Socket 对 Fork 出的子进程无用,所以子进程将其关闭,用产生的新 Socket 与客户交换信息,直到对方关闭此连接,子进程终止.

## 2 Delphi 环境下的网络编程

虽然现在有很多工具如 FTP 程序可以在网络上传输数据和文件,但是通过 WinSock 编程有更大的灵活性,它不需要关心网络连接的细节,作者也可以用它来实现自己定义的一些协议. 如果一开始就编写分布式系统进程通信恶程序,则必须对相关的网络协议、系统的低层知识以及网络软硬件技术有较深的了解与掌握. 而在 Windows NT 下,可以使用很多编程语言,如 VC + + , Java , Delphi 等. 其中,Delphi 对原来的 Windows Sockets 库函数进行了很好的包装,封装了各种功能,使编程更加简单.

Delphi 6 中网络组件分为 ClientSocket 和 ServerSocket,分别用于客户端和服务器端. 对于 ClientSocket,它是请求方,即它是主动地建立联接;而 Serversocket 用于响应方,其动作是侦听以及被动接受联接.

组件 Serversocket 属性是动态的,伴随着一个新的 ClientSocket 与之链接的同时,就会产生一个新的 Socket 与该 ClientSocket 对应,进行单独的通信. 因此,在同一个 ServerSocket 中,可以与多个 ClientSocket 保持同时链接和各自的通信. ServerSocket 的属性 Socket. ActiveConnections 用于表示客户端联接的数量,属性 Socket. Connections [index] 用于访问单个与 ClientSocket 联接的 Socket. 下面以 3 个实例介绍 Socket 编程的一些关键技术.

### 2.1 局域网文件传输的实现

指定 TClientSocket 服务器有两种方式:如指定 http://www.inprise.com 或者机器名. 这种方式较直观,但要进行域名解析,速度会慢一些;另一种方法是指定主机的 IP 地址.

指定连接服务器的端口号,也有两种方式,一是设置 Service 使用默认端口号,一是直接设置 Port 端口号. 在 1024 的端口号中,很多已经分配,如 FTP 端口为 20 和 21,SMTP 的端口为 25,Web 服务器端口为 80,建议最好将端口设置为 1 024 以上.

建立起客户和服务器的联接后,就可以进行通信. Delphi 为 TServerSocket 和 TClientSocket 提供了几种通信用的方法,用 SendText 发送文本信息,用 SendStream 发送流,用 SendBuf 发送指定数据的长度.

需要注意的是,由于 Windows 默认缓冲区大小为 8K,所以当发送大于 8K 的信息时,必须采用分组循环发送的方式. 服务器程序主要负责监听客户端的联接,开始先通过 SendText 向客户机发送服务器名、IP 地址,文件大小和文件名. 客户机发送"send",服务器把文件保存到内存流中,开始按照 BufferSize(此处为 2K) 大小来分段发送数据,直到发送完为止,此时再清空内存流,等待下次发送[2].

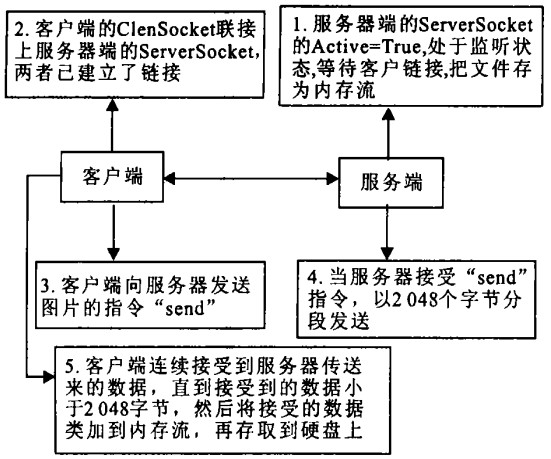


**图 2　分组循环发送文件示意**

**Fig. 2　Diagram of file transmitting by groups**

发送文件的核心程序如下:

```
Const BufSize = 2048;
Procedure TMyPicture.SendFilePart;
Var
    SendSize:longint; //待发送的字节数
    Buf:array[0..BufSize-1] of char //建立一块2K的字符缓冲区
begin
    If MyStream.Size = 0 then GetFileReady(); //内存流大小为0,则读取文件到内存流
    if LeftSize > BufSize then SendSize:= BufSize;// 余下的文件>2K,则发送BufSize(2K)
    else SendSize:= LeftSize //余下的文件不足2K,则全部发送完
    MyStream.ReadBuffer(Buf, SendSize);//把内存流写入缓冲区
    LeftSize:= LeftSize-SendSize; //LeftSize 标志剩余字节数
    If LeftSize = 0 then MyStream.Clear; //全部发送,则清空内存流
    Try
        ClientSocket1.Socket.SendBuf(buf, SendSize);// 发送缓冲区
    Except
        MyStream.Clear;
    End;
End;
```

同样,在客户机接受的时候,要判断接受到的流的大小,如果流的大小等于2K,则发送继续,服务器继续发送BufSize大小,客户机循环接受,直到介绍到的大小小于2K,最后再从流中读取文件[4].

### 2.2　通过 Socket 编程实现计算机的远程控制

有一些共享软件(如Netmeeting,PcAnyWhere)可以实现远程控制,通过共享桌面的形式,多远程主机作任何操作,就像控制本机一样.这种控制技术在远程设备(软件)的维护,监控与故障诊断等方面有很大的应用前景.在一些网络型的监控系统中,监控软件由设备厂家提供,不可能自己修改.但使用者通过Socket通信技术结合计算机系统控制的一些API,从而实现对计算机的远程控制.比如屏幕捕捉、开关机、对对方的鼠标及键盘和运行的进程、线程实行监控、视屏传输等.控制端和被控端连接成功后,在控制端ButtonClick事件中ControlSocket.Socket.SendText(‘reboot’);在被控端UnderControlSocket的OnClientRead事件中加入:if Socket.ReceiveText = ’reboot’ then ExitWindowsEx(EWX-REBOOT,2);//重启的API函数当然可以把上面的重启的API函数换成其他的API函数,就可以实现其他的远程控制.

## 3　结　语

本文以Delphi语言为例,重点介绍了利用Socket编程技术实现局域网的文件传输,计算机的远程监控等技术.目前计算机网络持续而高速地发展,其中基于TCP/IP协议网络已经成为计算机之间组网的常见形式.基于TCP/IP的网络编程,也得到了广泛的应用[5].目前,大多数远程进程间通信代码是用Socket编写的,实际应用中用Socket传输信息并不是独立的,它在多线程的处理环境中应用更为广泛.

# 嵌入式资源免费下载

## 总线协议：

1. 基于 PCIe 驱动程序的数据传输卡 DMA 传输
2. 基于 PCIe 总线协议的设备驱动开发
3. CANopen 协议介绍
4. 基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计
5. FPGA 实现 PCIe 总线 DMA 设计
6. PCI Express 协议实现与验证
7. VPX 总线技术及其实现
8. 基于 Xilinx FPGA 的 PCIE 接口实现
9. 基于 PCI 总线的 GPS 授时卡设计
10. 基于 CPCI 标准的 6U 信号处理平台的设计
11. USB30 电路保护
12. USB30 协议分析与框架设计
13. USB 30 中的 CRC 校验原理及实现
14. 基于 CPLD 的 UART 设计
15. IPMI 在 VPX 系统中的应用与设计
16. 基于 CPCI 总线的 PMC 载板设计
17. 基于 VPX 总线的工件台运动控制系统研究与开发
18. PCI Express 流控机制的研究与实现
19. UART16C554 的设计
20. 基于 VPX 的高性能计算机设计
21. 基于 CAN 总线技术的嵌入式网关设计
22. Visual C 串行通讯控件使用方法与技巧的研究
23. IEEE1588 精密时钟同步关键技术研究
24. GPS 信号发生器射频模块的一种实现方案
25. 基于 CPCI 接口的视频采集卡的设计
26. 基于 VPX 的 3U 信号处理平台的设计
27. 基于 PCI Express 总线 1394b 网络传输系统 WDM 驱动设计
28. AT89C52 单片机与 ARINC429 航空总线接口设计
29. 基于 CPCI 总线多 DSP 系统的高速主机接口设计
30. 总线协议中的 CRC 及其在 SATA 通信技术中的应用
31. 基于 FPGA 的 SATA 硬盘加解密控制器设计
32. Modbus 协议在串口通讯中的研究及应用
33. 高可用的磁盘阵列 Cache 的设计和实现
34. RAID 阵列中高速 Cache 管理的优化

35. 一种新的基于 RAID 的 CACHE 技术研究与实现
36. 基于 PCIE-104 总线的高速数据接口设计
37. 基于 VPX 标准的 RapidIO 交换和 Flash 存储模块设计
38. 北斗卫星系统在海洋工程中的应用
39. 北斗卫星系统在远洋船舶上应用的研究
40. 基于 CPCI 总线的红外实时信号处理系统
41. 硬件实现 RAID 与软件实现 RAID 的比较
42. 基于 PCI Express 总线系统的热插拔设计
43. 基于 RAID5 的磁盘阵列 Cache 的研究与实现
44. 基于 PCI 总线的 MPEG2 码流播放卡驱动程序开发
45. 基于磁盘异或引擎的 RAID5 小写性能优化
46. 基于 IEEE1588 的时钟同步技术研究
47. 基于 Davinci 平台的 SD 卡读写优化

# VxWorks：

1. 基于 VxWorks 的多任务程序设计
2. 基于 VxWorks 的数据采集存储装置设计
3. Flash 文件系统分析及其在 VxWorks 中的实现
4. VxWorks 多任务编程中的异常研究
5. VxWorks 应用技巧两例
6. 一种基于 VxWorks 的飞行仿真实时管理系统
7. 在 VxWorks 系统中使用 TrueType 字库
8. 基于 FreeType 的 VxWorks 中文显示方案
9. 基于 Tilcon 的 VxWorks 简单动画开发
10. 基于 Tilcon 的某武器显控系统界面设计
11. 基于 Tilcon 的综合导航信息处理装置界面设计
12. VxWorks 的内存配置和管理
13. 基于 VxWorks 系统的 PCI 配置与应用
14. 基于 MPC8270 的 VxWorks BSP 的移植
15. Bootrom 功能改进经验谈
16. 基于 VxWorks 嵌入式系统的中文平台研究与实现
17. VxBus 的 A429 接口驱动
18. 基于 VxBus 和 MPC8569E 千兆网驱动开发和实现
19. 一种基于 vxBus 的 PPC 与 FPGA 高速互联的驱动设计方法
20. 基于 VxBus 的设备驱动开发
21. 基于 VxBus 的驱动程序架构分析
22. 基于 VxBus 的高速数据采集卡驱动程序开发
23. Vxworks 下的冗余 CAN 通讯模块设计

24. WindML 工业平台下开发 S1d13506 驱动及显示功能的实现
25. WindML 中 Mesa 的应用
26. VxWorks 下图形用户界面开发中双缓冲技术应用
27. VxWorks 上的一种 GUI 系统的设计与实现
28. VxWorks 环境下 socket 的实现
29. VxWorks 的 WindML 图形界面程序的框架分析

# Linux：

1. Linux 程序设计第三版及源代码
2. NAND FLASH 文件系统的设计与实现
3. 多通道串行通信设备的 Linux 驱动程序实现
4. Zsh 开发指南-数组
5. 常用 GDB 命令中文速览
6. 嵌入式 C 进阶之道
7. Linux 串口编程实例
8. 基于 Yocto Project 的嵌入式应用设计
9. Android 应用的反编译
10. 基于 Android 行为的加密应用系统研究
11. 嵌入式 Linux 系统移植步步通
12. 嵌入式 CC++语言精华文章集锦
13. 基于 Linux 的高性能服务器端的设计与研究
14. S3C6410 移植 Android 内核
15. Android 开发指南中文版
16. 图解 Linux 操作系统架构设计与实现原理（第二版）
17. 如何在 Ubuntu 和 Linux Mint 下轻松升级 Linux 内核
18. Android 简单 mp3 播放器源码
19. 嵌入式 Linux 系统实时性的研究
20. Android 嵌入式系统架构及内核浅析
21. 基于嵌入式 Linux 操作系统内核实时性的改进方法研究
22. Linux TCP IP 协议详解
23. Linux 桌面环境下内存去重技术的研究与实现
24. 掌握 Android 7.0 新增特性 Quick Settings
25. Android 应用逆向分析方法研究
26. Android 操作系统的课程教学
27. Android 智能手机操作系统的研究
28. Android 英文朗读功能的实现
29. 基于 Yocto 订制嵌入式 Linux 发行版
30. 基于嵌入式 Linux 的网络设备驱动设计与实现

31. 如何高效学习嵌入式
32. 基于 Android 平台的 GPS 定位系统的设计与实现
33. LINUX ARM 下的 USB 驱动开发
34. Linux 下基于 I2C 协议的 RTC 驱动开发

# Windows CE:

1. Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计
2. Windows CE 的 CAN 总线驱动程序设计
3. 基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究
4. 基于 Windows CE.NET 平台的串行通信实现
5. 基于 Windows CE.NET 下的 GPRS 模块的研究与开发
6. win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码
7. Windows 下的 USB 设备驱动程序开发
8. WinCE 的大容量程控数据传输解决方案设计
9. WinCE6.0 安装开发详解
10. DOS 下仿 Windows 的自带计算器程序 C 源码
11. G726 局域网语音通话程序和源代码
12. WinCE 主板加载第三方驱动程序的方法
13. WinCE 下的注册表编辑程序和源代码
14. WinCE 串口通信源代码
15. WINCE 的 SD 卡程序[可实现读写的源码]
16. 基于 WinCE 的 BootLoader 研究
17. Windows CE 环境下无线网卡的自动安装
18. 基于 Windows CE 的可视电话的研究与实现
19. 基于 WinCE 的嵌入式图像采集系统设计
20. 基于 ARM 与 WinCE 的掌纹鉴别系统

# PowerPC:

1. Freescale MPC8536 开发板原理图
2. 基于 MPC8548E 的固件设计
3. 基于 MPC8548E 的嵌入式数据处理系统设计
4. 基于 PowerPC 嵌入式网络通信平台的实现
5. PowerPC 在车辆显控系统中的应用
6. 基于 PowerPC 的单板计算机的设计

7. 用 PowerPC860 实现 FPGA 配置
8. 基于 MPC8247 嵌入式电力交换系统的设计与实现
9. 基于设备树的 MPC8247 嵌入式 Linux 系统开发
10. 基于 MPC8313E 嵌入式系统 UBoot 的移植
11. 基于 PowerPC 处理器 SMP 系统的 UBoot 移植
12. 基于 PowerPC 双核处理器嵌入式系统 UBoot 移植
13. 基于 PowerPC 的雷达通用处理机设计
14. PowerPC 平台引导加载程序的移植
15. 基于 PowerPC 嵌入式内核的多串口通信扩展设计
16. 基于 PowerPC 的多网口系统抗干扰设计

## ARM：

1. 基于 DiskOnChip 2000 的驱动程序设计及应用
2. 基于 ARM 体系的 PC-104 总线设计
3. 基于 ARM 的嵌入式系统中断处理机制研究
4. 设计 ARM 的中断处理
5. 基于 ARM 的数据采集系统并行总线的驱动设计
6. S3C2410 下的 TFT LCD 驱动源码
7. STM32 SD 卡移植 FATFS 文件系统源码
8. STM32 ADC 多通道源码
9. ARM Linux 在 EP7312 上的移植
10. ARM 经典 300 问
11. 基于 S5PV210 的频谱监测设备嵌入式系统设计与实现
12. Uboot 中 start.S 源码的指令级的详尽解析
13. 基于 ARM9 的嵌入式 Zigbee 网关设计与实现
14. 基于 S3C6410 处理器的嵌入式 Linux 系统移植
15. CortexA8 平台的μC-OSⅡ及 LwIP 协议栈的移植与实现
16. 基于 ARM 的嵌入式 Linux 无线网卡设备驱动设计
17. ARM S3C2440 Linux ADC 驱动
18. ARM S3C2440 Linux 触摸屏驱动
19. Linux 和 Cortex-A8 的视频处理及数字微波传输系统设计
20. Nand Flash 启动模式下的 Uboot 移植
21. 基于 ARM 处理器的 UART 设计
22. ARM CortexM3 处理器故障的分析与处理
23. ARM 微处理器启动和调试浅析
24. 基于 ARM 系统下映像文件的执行与中断运行机制的实现
25. 中断调用方式的 ARM 二次开发接口设计

# Hardware:

1. DSP 电源的典型设计
2. 高频脉冲电源设计
3. 电源的综合保护设计
4. 任意波形电源的设计
5. 高速 PCB 信号完整性分析及应用
6. DM642 高速图像采集系统的电磁干扰设计
7. 使用 COMExpress Nano 工控板实现 IP 调度设备
8. 基于 COM Express 架构的数据记录仪的设计与实现
9. 基于 COM Express 的信号系统逻辑运算单元设计
10. 基于 COM Express 的回波预处理模块设计
11. 基于 X86 平台的简单多任务内核的分析与实现
12. 基于 UEFI Shell 的 PreOS Application 的开发与研究
13. 基于 UEFI 固件的恶意代码防范技术研究
14. MIPS 架构计算机平台的支持固件研究
15. 基于 UEFI 固件的攻击验证技术研究
16. 基于 UEFI 的 Application 和 Driver 的分析与开发
17. 基于 UEFI 的可信 BIOS 研究与实现
18. 基于 UEFI 的国产计算机平台 BIOS 研究

# Programming:

1. 计算机软件基础数据结构 - 算法
2. 高级数据结构对算法的优化
3. 零基础学算法
4. Linux 环境下基于 TCP 的 Socket 编程浅析
5. Linux 环境下基于 UDP 的 socket 编程浅析