**网络技术实践**

**实验报告**

**学号：1120141945**

**姓名：高子恺**

**班级：07111406**

**北京理工大学**

**计算机学院**

**2017年10月**

**实验一 Socket编程实验**

**一、设计要求**

**1. 实验目的**

WinSock 是 Windows 操作系统下的 Socket 编程接口，通过 WinSock 函数库 可以实现基于 TCP/IP 协议的进程之间通信。

* 理解基于 WinSock 的客户/服务器概念
* 掌握使用 WinSock 进行编程的方法
* 了解常见 WinSock 开发模式的使用

**2. 实验内容**

基于 WinSock 开发一个简单的客户/服务器文本传输程序，客户端能够发送由标准输入得到的文本，服务器能够接收并将其显示在标准输出上。学生将编写一个TCP或UDP客户端和一个TCP或UDP服务器。实现从远程服务器检索温度、湿度和光数据的客户机。客户机应该连接到服务器，保持连接，并检索一些读数。客户机可以像您希望的那样多次从服务器请求读数。

服务器负责接受客户端连接，并根据客户发送的命令发送正确的数据。

服务器返回的数据如下：

温度= 60 湿度= 0 光= 1

**3. 实验步骤**

**步骤 1** 需求分析

程序功能为：

（1） 服务器可以接受任何客户的连接

（2） 服务器在同一时刻只能与一个客户通信，直到该客户退出才可以接收下一个客户

（3） 客户程序使用命令行参数指定服务器地址

（4） 客户端输入的文本都发送给服务器

（5） 客户使用 Ctrl+C 键停止发送，关闭连接

**步骤 2** 服务器程序开发

以 Visual C++ Console 控制台程序的形式编写服务器程序，服务器程序采用 循环服务器的模式，不支持多线程操作。

**步骤 3** 客户端程序开发

以 Visual C++ Console 控制台程序的形式编写客户程序，与服务器建立连接 后开始接收标准输入的文本，并发送到服务器。

**步骤 4** 编译和执行程序

在 Visual C++开发环境中包含必要的头文件和连接的 WinSock 库，分别进行客户程序和服务器程序的编译和连接，生成可执行文件。首先运行服务器程序，然后运行客户程序，测试程序运行结果是否符合要求。

**二、系统总体设计说明**

分别编写服务端和客户端程序，其中服务端指定自己的IP地址，并且设置可接受任意IP的连接；客户端指定所要连接的服务端的IP地址进行连接。使用WinSock库中函数进行Socket进行连接通信。

**三、编程/开发环境**

操作系统：Windows 10专业版

处理器：Intel(R) Xeon(R) CPU E5-2660 v2 @2.20GHz

内存：32GB

系统类型：64位操作系统，基于x64的处理器

IDE：Microsoft Visual Studio Community 2017 版本 15.2 (26430.15) Release

**四、系统设计详细说明**

**代码说明：**

**Server.cpp：**

初始化WSA

|  |
| --- |
| WORD sockVersion = MAKEWORD(2, 2);  WSADATA wsaData;  if (WSAStartup(sockVersion, &wsaData) != 0)  {  return 0;  } |

创建套接字

|  |
| --- |
| SOCKET server = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);  if (server == INVALID\_SOCKET)  {  printf("socket error !");  return 0;  } |

将IP设为127.0.0.1，端口设置为8888，绑定socket

|  |
| --- |
| sockaddr\_in serv\_addr;  serv\_addr.sin\_family = AF\_INET;  serv\_addr.sin\_port = htons(8888);  //serv\_addr.sin\_addr.S\_un.S\_addr = INADDR\_ANY;  serv\_addr.sin\_addr.S\_un.S\_addr = inet\_addr("127.0.0.1");  if (bind(server, (LPSOCKADDR)&serv\_addr, sizeof(serv\_addr)) == SOCKET\_ERROR)  {  printf("bind error !");  } |

读服务器数据

|  |
| --- |
| FILE \* fp\_hum = fopen("humidity.dat", "r");//打开dat格式的输入文件  FILE \* fp\_light = fopen("light.dat", "r");//打开dat格式的输入文件  FILE \* fp\_tem = fopen("temperature.dat", "r");//打开dat格式的输入文件  int i = 0;  int hum[30], light[30], tem[30];  for (i = 0; i < 28; i++) {  fscanf(fp\_hum, "%d", &hum[i]);//从输入文件读取一个整数  fscanf(fp\_light, "%d", &light[i]);//从输入文件读取一个整数  fscanf(fp\_tem, "%d", &tem[i]);//从输入文件读取一个整数  }  fclose(fp\_hum);  fclose(fp\_light);  fclose(fp\_tem); |

开始监听是否有客户端接入

|  |
| --- |
| if (listen(server, 5) == SOCKET\_ERROR)  {  printf("listen error !");  return 0;  } |

等待客户端的连接

|  |
| --- |
| while (true)  {  //连接  printf("等待连接...\n");  client = accept(server, (SOCKADDR \*)&ip\_addr, &nAddrlen);  if (client == INVALID\_SOCKET)  {  printf("accept error !");  continue;  }  //连接成功  printf("接受到一个连接\n");  char \* sendData = "连接成功！\n";  send(client, sendData, strlen(sendData), 0);  break;  } |

与客户端进行通信，若收到消息，则发送成功信息给客户端

|  |
| --- |
| while (true)  {  //接收数据  int ret = recv(client, revData, 255, 0);  if (ret > 0)  {  revData[ret] = 0x00;  printf("查询时间：");  printf(revData);  printf("\n");  }  else if (ret == SOCKET\_ERROR)  {  closesocket(client);  break;  }  int time = atoi(revData);  char str1[25] ;  char str2[25] ;  char str3[25] ;  itoa(hum[time], str1, 10);  itoa(light[time], str2, 10);  itoa(tem[time], str3, 10);  char str[100];  strcpy(str, "当前传感器状态：湿度=");  strcat(str, str1);  strcat(str, "\t亮度=");  strcat(str, str2);  strcat(str, "\t温度=");  strcat(str, str3);  strcat(str, "\n");  //发送数据  char \* sendData = str;  send(client, sendData, strlen(sendData), 0);  } |

关闭socket，清理WSA，程序结束

|  |
| --- |
| closesocket(server);  WSACleanup(); |

**客户端：**

初始化WSA和建立套接字过程与服务端相似。

指定服务器地址，并通过socket进行连接

|  |
| --- |
| sockaddr\_in serAddr;  char server\_addr[25];  serAddr.sin\_family = AF\_INET;  serAddr.sin\_port = htons(8888);  //serAddr.sin\_addr.S\_un.S\_addr = inet\_addr("127.0.0.2");  printf("Please give a Server IP Adderss:");  scanf("%s", server\_addr);  serAddr.sin\_addr.S\_un.S\_addr = inet\_addr(server\_addr);  if (connect(sclient, (sockaddr \*)&serAddr, sizeof(serAddr)) == SOCKET\_ERROR)  {  printf("connect error !");  closesocket(sclient);  return 0;  } |

接收与发送数据部分

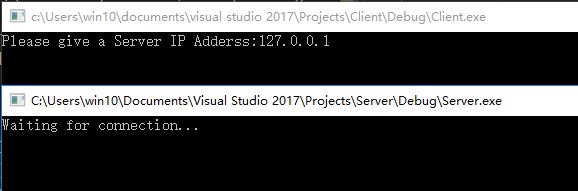
|  |
| --- |
| char sendData[100];  char recData[255];  int ret = recv(sclient, recData, 255, 0);  if (ret > 0)  {  recData[ret] = 0;  printf(recData); //接受成功的提示  }  printf("请输入要查询的时间：\n");  while (true)  {  scanf("%s", sendData);  send(sclient, sendData, strlen(sendData), 0);  char recData[255];  int ret = recv(sclient, recData, 255, 0);  if (ret > 0)  {  recData[ret] = 0x00;  printf(recData);  }  } |

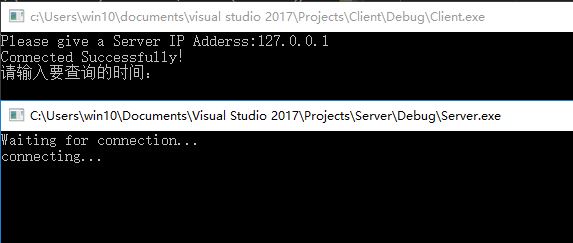
**五、程序测试与验证**

说明可执行程序的运行环境要求及运行方法。

主要运行界面和功能的截图。

服务端等待连接时：



当客户端指定服务器地址进行连接，连接成功后

客户端向服务端发送查询请求，服务端接受到消息，同时客户端收到来自服务端的反馈信息：

