

**课程设计报告**

**课程名称：计算机网络技术**

**设计题目：TCP演示软件设计**

**专业名称：软件工程**

**开发者 ：yocichen**

**2018年12月 14 日**

**一、概述**

**1.1 设计目的**

使学生能对TCP协议的过程更加熟悉，能深入理解协议的有关概念、掌握TCP协议的工作原理，能在实现动态演示协议的工作过程中体会所学知识。

**1.2 设计内容**

1.2.1 TCP协议建立连接（三报文握手）过程的动态演示；

1.2.2 TCP协议释放连接（四次挥手）过程的动态演示；

1.2.3 TCP协议拥塞控制 （慢开始、拥塞避免、快重传和快恢复）过程的动态演示。

**1.3 设计要求**

1.3.1能够动态演示协议工作过程；

1.3.2能够给出协议原理的文字和语音介绍；

1.3.3能够通过协议相关参数设定，完成协议某一工作过程的演示；

**二、设计任务分析**

**1.需求说明**

（1）用户能演示TCP协议主要过程（包括建立连接、释放连接、拥塞控制）的工作过程；

（2）用户可以通过点击相关按钮开始可暂停动态演示过程。

**2.用例图**



系统用例图

**3.用例图说明**

用户可以驱动演示三报文握手、拥塞控制、释放连接三个用例，进行TCP协议主要过程的演示。在三个用例中，可以驱动暂停和继续演示用例，来达到暂停和继续动画播放的目的。

**三、总体设计**

**1.框架：MFC**

**2.程序结构图**



程序框架图

**3.函数说明**

|  |  |  |
| --- | --- | --- |
| 函数名称 | 完成功能 | 参数 |
| Onlink() | 连接建立响应 | NULL |
| Onbreak() | 释放连接响应 | NULL |
| OnSlowBegin() | 拥塞控制、慢开始、拥塞避免、快恢复演示 | NULL |
| TCP\_link() | 创建连接：三次握手演示 | NULL |
| TCP\_break() | 四次挥手演示 | NULL |
| DrawLine(CPoint ,CPoint , COLORREF ) | OnSlowBegin()函数调用，完成图形绘制 | 两个CPoint对象引用、一个COLORREF对象引用 |
| OnContinue() | 断掉计时器 | NULL |
| Speak(CString str) | 语音合成 | CString字符串 |
| OnStopTimer() | 继续计时器 | NULL |

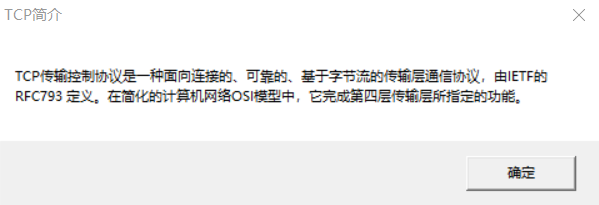
**四、程序实现**

**1.结果**

**1.1登陆及协议简介**

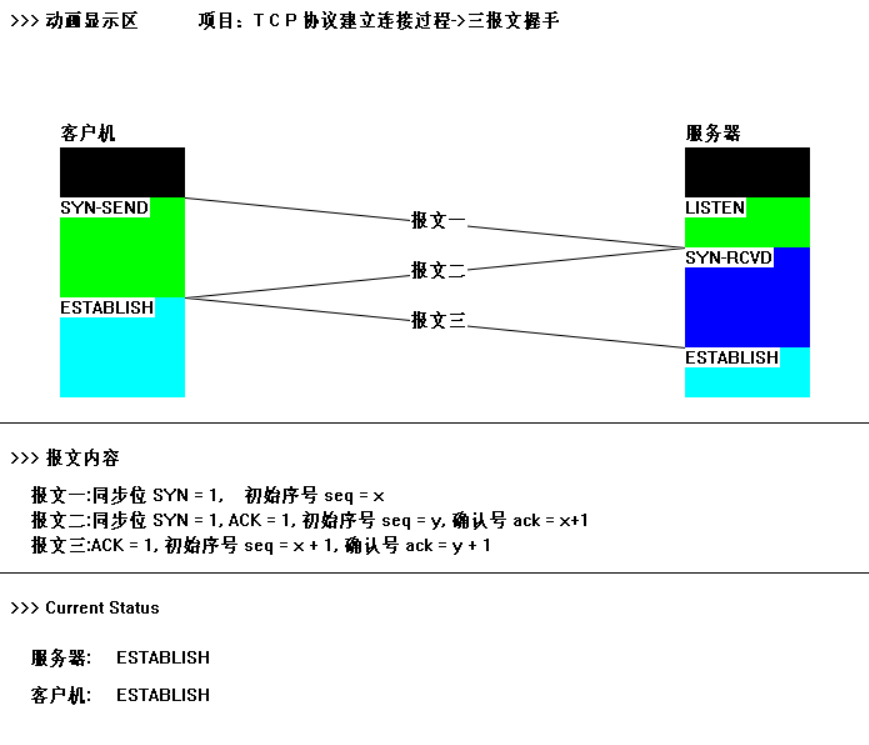


登陆界面



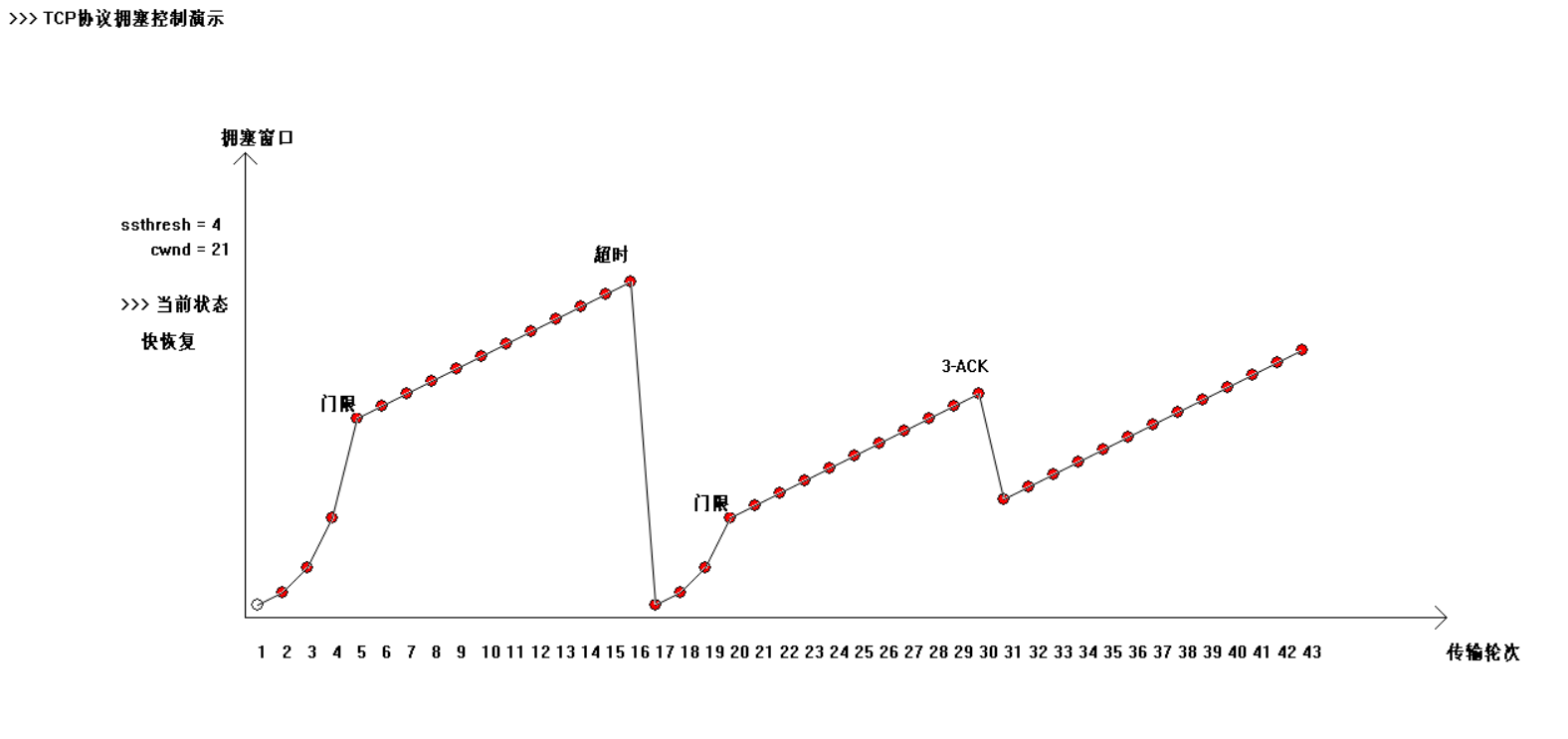
简介

**1.2三次握手**



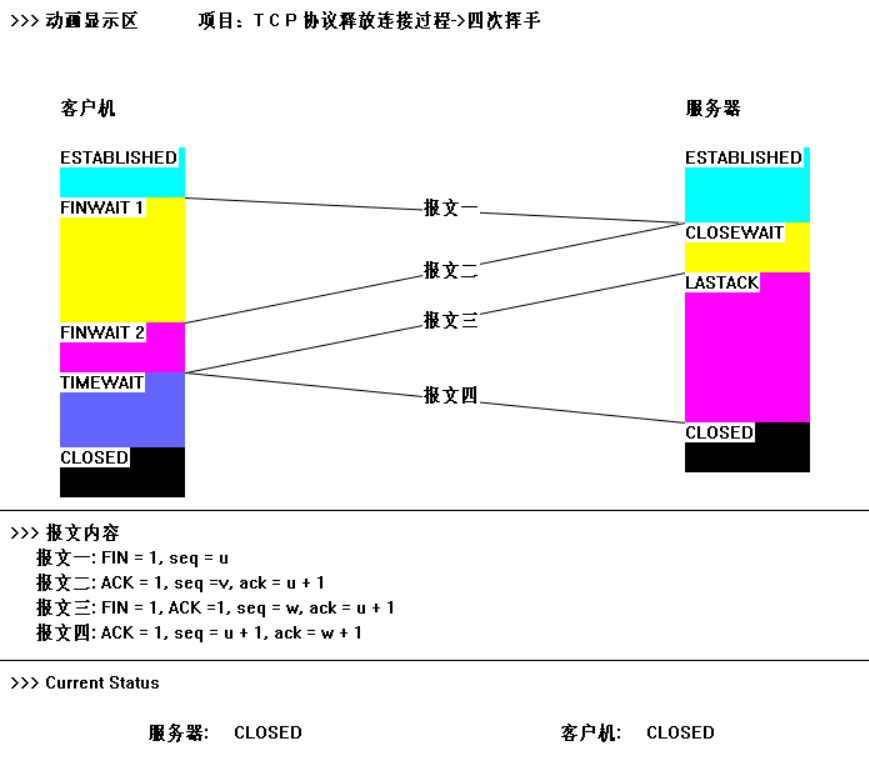
三次握手演示结果

**1.3拥塞控制**

****

拥塞控制演示结果

**1.4 四次挥手**



四次挥手演示结果

**2.主要源代码**

**//封装SAPI**

void CTCPDemoView::Speak(CString str)

{

//----------------------------

ISpVoice \* pVoice = NULL;

//获取ISpVoice接口：

HRESULT hr = CoCreateInstance(CLSID\_SpVoice, NULL, CLSCTX\_ALL, IID\_ISpVoice, (void \*\*)&pVoice);

if(SUCCEEDED(hr))

{

pVoice->SetVolume(100);//设置音量

pVoice->SetRate(0);//[-10, 10]

hr = pVoice->Speak(CStringToBSTR(str), 0, NULL);

pVoice->Release(); //释放token

pVoice = NULL;

}

::CoUninitialize();

//-----------------------------

}

**//创建链接：三次握手**

void CTCPDemoView::TCP\_link()

{

RedrawWindow();

//自定义坐标系

CDC \*pDC = GetDC();

CRect rect;

GetClientRect(&rect);

pDC->SetMapMode(MM\_ANISOTROPIC);

pDC->SetWindowExt(rect.Width(), rect.Height());

pDC->SetViewportExt(rect.Width(), -rect.Height());

pDC->SetViewportOrg(rect.Width()/2, rect.Height()/2);

rect.OffsetRect(-rect.Width()/2, -rect.Height()/2);

CPoint p0, p1;

COLORREF clr = RGB(255, 0, 0);

pDC->Rectangle(-x-150, y+100, x+150, -y-100);

pDC->TextOut(-x-140, y+90, ">>> 动画显示区");

pDC->TextOut(-x+10, y+90, "项目：T C P 协议建立连接过程->三报文握手");

pDC->MoveTo(-x-150, y-240);

pDC->LineTo(x+150, y-240);

pDC->TextOut(-x-140, y-260, ">>> 报文内容");

//pDC->MoveTo(-x-150, y-280);

//pDC->LineTo(x+150, y-280);

pDC->MoveTo(-x-150, y-360);

pDC->LineTo(x+150, y-360);

pDC->TextOut(-x-140, y-380, ">>> Current Status");

pDC->TextOut(-x-140, y-420, " 服务器: CLOSE");

pDC->TextOut(-x-140, y-450, " 客户机: CLOSE");

//---------------------------------------------------

Speak("T C P 协议建立连接过程:三报文握手");

pDC->TextOut(x, y, "服务器");

CBrush brush;

brush.CreateSolidBrush(RGB(0, 0, 0));

//服务器 CLOSED

CRect s1(x, y-20, x+100, y-60);

pDC->FillRect(&s1, &brush);

brush.DeleteObject();

Speak("服务器启动");

brush.CreateSolidBrush(RGB(0, 255, 0));

CRect s2(x, y-60, x+100, y-100);//server LISTEN

Speak("服务器进入LISTEN状态");

pDC->FillRect(&s2, &brush);

pDC->TextOut(x, y-60, "LISTEN");

pDC->TextOut(-x-140, y-420, " 服务器: LISTEN");

brush.DeleteObject();

Speak("客户机启动");

pDC->TextOut(-x-100, y, "客户机");

CRect c1(-x, y-20, -(x+100), y-60);//srevr CLOSED

brush.CreateSolidBrush(RGB(0, 0, 0));

pDC->FillRect(&c1, &brush);

brush.DeleteObject();

//握手1

Speak("客户机发送连接请求");

//pDC->MoveTo(-x, y-60);

//pDC->LineTo(x, y-100);

p0.x = -x;p0.y = y-60;

p1.x = x; p1.y = y-100;

wuLine(p0, p1, clr);

//显示报文1内容

pDC->TextOut(-20, y-70, "报文一");

pDC->TextOut(-x-140, y-290, " 报文一:同步位 SYN = 1, 初始序号 seq = x");

//客户机SYN-SEND状态

Speak("客户机进入SYN-SEND状态");

CRect c2(-x, y-60, -(x+100), y-140);

brush.CreateSolidBrush(RGB(0, 255, 0));

pDC->FillRect(&c2, &brush);

pDC->TextOut(-x-100, y-60, "SYN-SEND");

pDC->TextOut(-x-140, y-450, " 客户机: SYN-SEND");

brush.DeleteObject();

Speak("服务器收到连接请求，进入SYN-RECEIVED状态");

//服务器SYN-RCVD状态

CRect s3(x, y-100, x+100, y-180);

brush.CreateSolidBrush(RGB(0, 0, 255));

pDC->FillRect(&s3, &brush);

pDC->TextOut(x, y-100, "SYN-RCVD");

pDC->TextOut(-x-140, y-420, " 服务器: SYN-RCVD");

brush.DeleteObject();

//握手2

Speak("服务器发送ACK");

//pDC->MoveTo(x, y-100);

//pDC->LineTo(-x, y-140);

p0.x = x;p0.y = y-100;

p1.x = -x; p1.y = y-140;

wuLine(p0, p1, clr);

//显示报文2内容

pDC->TextOut(-20, y-110, "报文二");

pDC->TextOut(-x-140, y-310, " 报文二:同步位 SYN = 1, ACK = 1, 初始序号 seq = y, 确认号 ack = x+1");

//客户机ESTABLISHED状态

Speak("客户机收到ACK，进入ESTABLISHED状态");

CRect c3(-x, y-140, -(x+100), y-220);

brush.CreateSolidBrush(RGB(0, 255, 255));

pDC->FillRect(&c3, &brush);

pDC->TextOut(-x-100, y-140, "ESTABLISH");

pDC->TextOut(-x-140, y-450, " 客户机: ESTABLISH");

brush.DeleteObject();

//握手3

Speak("客户机发送已连接");

//pDC->MoveTo(-x, y-140);

//pDC->LineTo(x, y-180);

p0.x = -x;p0.y = y-140;

p1.x = x; p1.y = y-180;

wuLine(p0, p1, clr);

//显示报文3内容

pDC->TextOut(-20, y-150, "报文三");

pDC->TextOut(-x-140, y-330, " 报文三:ACK = 1, 初始序号 seq = x + 1, 确认号 ack = y + 1");

//服务器ESTABLISHED状态

Speak("服务器收到连接信息，进入ESTABLISHED状态");

CRect s4(x, y-180, x+100, y-220);

brush.CreateSolidBrush(RGB(0, 255, 255));

pDC->FillRect(&s4, &brush);

pDC->TextOut(x, y-180, "ESTABLISH");

pDC->TextOut(-x-140, y-420, " 服务器: ESTABLISH");

brush.DeleteObject();

Speak("三次握手完成，连接已建立，可进行数据传输");

//------------------------------------------------------

}

**//四次挥手**

void CTCPDemoView::TCP\_break()

{

RedrawWindow();

//自定义坐标系

CDC \*pDC = GetDC();

CRect rect;

GetClientRect(&rect);

pDC->SetMapMode(MM\_ANISOTROPIC);

pDC->SetWindowExt(rect.Width(), rect.Height());

pDC->SetViewportExt(rect.Width(), -rect.Height());

pDC->SetViewportOrg(rect.Width()/2, rect.Height()/2);

rect.OffsetRect(-rect.Width()/2, -rect.Height()/2);

CPoint p0, p1;

COLORREF clr = RGB(255, 0, 0);

pDC->Rectangle(-x-150, y+100, x+150, -y-120);//全局

pDC->TextOut(-x-140, y+90, ">>> 动画显示区");

pDC->TextOut(-x+10, y+90, "项目：T C P 协议释放连接过程->四次挥手");

pDC->MoveTo(-x-150, y-310);

pDC->LineTo(x+150, y-310);

pDC->TextOut(-x-140, y-320, ">>> 报文内容");

//pDC->MoveTo(-x-150, y-280);

//pDC->LineTo(x+150, y-280);

pDC->MoveTo(-x-150, y-430);

pDC->LineTo(x+150, y-430);

pDC->TextOut(-x-140, y-440, ">>> Current Status");

pDC->TextOut(-x-50, y-480, " 服务器: CLOSE");

pDC->TextOut(x-120, y-480, " 客户机: CLOSE");

//--------------------------------------------

Speak("T C P 协议释放连接过程:四次挥手");

CBrush brush;

brush.CreateSolidBrush(RGB(0, 255, 255));

//服务器

pDC->TextOut(x, y+20, "服务器");

CRect s1(x, y-20, x+100, y-80);

pDC->FillRect(&s1, &brush);

brush.DeleteObject();

pDC->TextOut(x, y-20, "ESTABLISHED");

pDC->TextOut(-x-50, y-480, " 服务器: ESTABLISHED ");

//客户机

pDC->TextOut(-x-100, y+20, "客户机");

brush.CreateSolidBrush(RGB(0, 255, 255));

CRect c1(-x, y-20, -x-100, y-60);

pDC->FillRect(&c1, &brush);

brush.DeleteObject();

pDC->TextOut(-x-100, y-20, "ESTABLISHED");

pDC->TextOut(x-120, y-480, " 客户机: ESTABLISHED ");

//挥手-1

Speak("客户机发送断开报文段1");

//pDC->MoveTo(-x, y-60);

//pDC->LineTo(x, y-80);

p0.x = -x;p0.y = y-60;

p1.x = x; p1.y = y-80;

wuLine(p0, p1, clr);

pDC->TextOut(-10, y-60, "报文一");

pDC->TextOut(-x-140, y-340, " 报文一: FIN = 1, seq = u");

Speak("客户机进入FINWAIT1状态");

brush.CreateSolidBrush(RGB(255, 255, 0));

CRect c2(-x, y-60, -x-100, y-160);

pDC->FillRect(&c2, &brush);

brush.DeleteObject();

pDC->TextOut(-x-100, y-60, "FINWAIT 1");

pDC->TextOut(x-120, y-480, " 客户机: FINWAIT 1 ");

//服务器收到数据报1，进入CLOSEWAIT状态

Speak("服务器收到报文1，进入CLOSEWAIT状态");

brush.CreateSolidBrush(RGB(255, 255, 0));

CRect s2(x, y-80, x+100, y-120);

pDC->FillRect(&s2, &brush);

brush.DeleteObject();

pDC->TextOut(x, y-80, "CLOSEWAIT");

pDC->TextOut(-x-50, y-480, " 服务器: CLOSEWAIT ");

//挥手-2

Speak("服务器回复ACK1");

//pDC->MoveTo(x, y-80);

//pDC->LineTo(-x, y-160);

p0.x = x;p0.y = y-80;

p1.x = -x; p1.y = y-160;

wuLine(p0, p1, clr);

pDC->TextOut(-10, y-110, "报文二");

pDC->TextOut(-x-140, y-360, " 报文二: ACK = 1, seq =v, ack = u + 1");

//客户机收到ACK1

Speak("客户机收到ACK1，进入FIWAIT 2状态");

brush.CreateSolidBrush(RGB(255, 0, 255));

CRect c3(-x, y-160, -x-100, y-200);

pDC->FillRect(&c3, &brush);

brush.DeleteObject();

pDC->TextOut(-x-100, y-160, "FINWAIT 2");

pDC->TextOut(x-120, y-480, " 客户机: FINWAIT 2 ");

//服务器进行必要数据传输，传输完成后发送ACK2

Speak("服务器进行必要数据传输");

//挥手-3

Speak("数据传输完成，发送确认释放连接报文");

//pDC->MoveTo(x, y-120);

//pDC->LineTo(-x, y-200);

p0.x = x;p0.y = y-120;

p1.x = -x; p1.y = y-200;

wuLine(p0, p1, clr);

pDC->TextOut(-10, y-150, "报文三");

pDC->TextOut(-x-140, y-380, " 报文三: FIN = 1, ACK =1, seq = w, ack = u + 1");

Speak("进入LAST ACK");

brush.CreateSolidBrush(RGB(255, 0, 255));

CRect s4(x, y-120, x+100, y-240);

pDC->FillRect(&s4, &brush);

brush.DeleteObject();

pDC->TextOut(x, y-120, "LASTACK");

pDC->TextOut(-x-50, y-480, " 服务器: LASTACK ");

//客户机收到ACK3

Speak("客户机收到确认释放报文，进入TIMEWAIT状态");

brush.CreateSolidBrush(RGB(100, 100, 255));

CRect c4(-x, y-200, -x-100, y-260);

pDC->FillRect(&c4, &brush);

brush.DeleteObject();

pDC->TextOut(-x-100, y-200, "TIMEWAIT");

pDC->TextOut(x-120, y-480, " 客户机: TIMEWAIT ");

//挥手-4

Speak("客户机发送ACK");

//pDC->MoveTo(-x, y-200);

//pDC->LineTo(x, y-240);

p0.x = -x;p0.y = y-200;

p1.x = x; p1.y = y-240;

wuLine(p0, p1, clr);

pDC->TextOut(-10, y-210, "报文四");

pDC->TextOut(-x-140, y-400, " 报文四: ACK = 1, seq = u + 1, ack = w + 1");

//服务器收到ACK4

Speak("服务器收到ACK，进入CLOSED状态");

brush.CreateSolidBrush(RGB(0, 0, 0));

CRect s5(x, y-240, x+100, y-280);

pDC->FillRect(&s5, &brush);

brush.DeleteObject();

pDC->TextOut(x, y-240, "CLOSED");

pDC->TextOut(-x-50, y-480, " 服务器: CLOSED ");

//客户机closed

Speak("客户机等待2MSL，进入CLOSED状态");

brush.CreateSolidBrush(RGB(0, 0, 0));

CRect c5(-x, y-260, -x-100, y-300);

pDC->FillRect(&c5, &brush);

brush.DeleteObject();

pDC->TextOut(-x-100, y-260, "CLOSED");

pDC->TextOut(x-120, y-480, " 客户机: CLOSED ");

//--------------------------------------------

Speak("四次挥手完成，连接已释放");

}

**//简介显示（多线程实现话音同步）**

UINT CTCPDemoView::TCP\_summary(LPVOID lparam)

{

CString str1 = "TCP传输控制协议是一种面向连接的、可靠的、基于字节流的传输层通信协议，由IETF的 RFC793 定义。在简化的计算机网络OSI模型中，它完成第四层传输层所指定的功能。";

//CString str2 = "功能：TCP协议用于应用程序之间的通信。当应用程序希望通过 TCP协议 与另一个应用程序通信时，它会发送一个通信请求。这个请求必须被送到一个确切的地址。在双方“握手”之后，TCP 将在两个应用程序之间建立一个全双工 (full-duplex) 的通信。这个全双工的通信将占用两个计算机之间的通信线路，直到它被一方或双方关闭为止。";

CTCPDemoView \*pThis = (CTCPDemoView \*)lparam;

pThis->MessageBox(str1, "TCP简介", MB\_OK);

//MessageBox(str, "TCP简介", MB\_OK);

//Speak(str);

return 0;

}

//summary语音

UINT CTCPDemoView::TCP\_summary\_speak(LPVOID lparam)

{

CString str1 = "TCP传输控制协议是一种面向连接的、可靠的、基于字节流的传输层通信协议，由IETF的 RFC793 定义。在简化的计算机网络OSI模型中，它完成第四层传输层所指定的功能。";

//CString str2 = "功能：TCP协议用于应用程序之间的通信。当应用程序希望通过 TCP协议 与另一个应用程序通信时，它会发送一个通信请求。这个请求必须被送到一个确切的地址。在双方“握手”之后，TCP 将在两个应用程序之间建立一个全双工 (full-duplex) 的通信。这个全双工的通信将占用两个计算机之间的通信线路，直到它被一方或双方关闭为止。";

CTCPDemoView \*pThis1 = (CTCPDemoView \*)lparam;

pThis1->Speak(str1);

//pThis1->Speak(str2);

return 0;

}

//多线程 显示、语音

void CTCPDemoView::Onsummary()

{

if(land\_status == false)

{

land();

}

RedrawWindow();

// TODO: Add your command handler code here

m\_pthread=AfxBeginThread((AFX\_THREADPROC)TCP\_summary, (LPVOID)this);

m\_pthread=AfxBeginThread((AFX\_THREADPROC)TCP\_summary\_speak, (LPVOID)this);

//pThread = AfxBeginThread(TCP\_summary\_Speak, NULL, THREAD\_PRIORITY\_NORMAL, 0, 0, NULL);

}

**五、心得与体会**

在本次课程设计中，通过TCP主要过程演示软件的设计与实现，我加深了对TCP协议的理解，特别是对三次握手、四次挥手以及拥塞控制过程和有关报文有了深入的探究。另外，对MFC框架也更为熟悉了。

**参考文献**

[1] 谢希仁《计算机网络》（第七版）。

[2] 百度、CSDN、MSDN、cnblogs等网络资料。