东南大学自动化学院

实 验 报 告

课程名称：信息通信网络概论

第 2 次实验

实验名称：FTP 客户端通信程序设计

院 （系）：自动化 专 业：自动化

姓 名：陈鲲龙 学 号：08022311

实 验 室：金智楼 实验组别：

同组人员： 实验时间： 2024 年 5 月 15 日

评定成绩： 审阅教师：

目　　录

实验报告内容：

一． 实验目的和要求

1. 实验目的和要求
1. 实验目的：
1) 学习使用 MFCWinlnet 函数编写 FTP 客户端程序
2) 理解相关类和函数的使用
3) 熟悉 FTP 的编程过程
2. 实验要求：
1) 理解掌握 CInternetSession、CFtpFileFind 类及 GetFtpConnection()、FindFile()、FindNextFile() 、GetFile()、PutFile()函数的使用
2) 使用 MFC WinInet 函数编写一个简单的 FTP 客户端程序，实现 FTP 客户端软件的基本功能—检索 FTP 文件服务器，上传和下载

二． 实验原理

在使用 MFC 编程时，要连接到 FTP 服务器必须建立一个 CinternetSession 对象，用类 CinternetSession 创建并初始化一个或几个同时存在的 Internet 会话,并描述与代理服务器的连接，如果在程序运行期间需要保持与 Internet 的连接，可以创建一个 CinternetSession 对象作为类 CwinApp 的成员。MFC 中的类 CftpConnection 与 Internet 服务器的连接，并直接操作服务器上的目录和文件，FTP 时 MFC 的 WinInet 支持的三个 Internet 功能之一，我们需要先创建一个 CinternetSession 实例和一个 CftpConnection 对象就可以实现和一个 FTP 服务器通信，不需要直接创建 CftpConnection 对象，而是通过调用 CinternetSession::GetFtpConnection 来创建 CFtpConnection 对象并返回一个指向该对象的指针。
在 FTP 连接的程序中，主要用下列的一些类和函数：
1.CInternetSession 类:
用类 CinternetSession 创建并初始化一个或几个同时存在的 Internet 会话
其中：
CInternetSession::CInternetSession 该函数用于建立会话；
CInternetSession(LPCTSTR pstrAgent = NULL, DWORD dwContext = 1, DWORD dwAccessType = INTERNET_OPEN_TYPE_PRECONFIG, LPCTSTR pstrProxyName = NULL, LPCTSTR pstrProxyBypass = NULL, DWORD dwFlags = 0)；在本实验中可以直接使用 pSession=new CInternetSession(AfxGetAppName(),1,PRE_CONFIG_INTERNET_ACCESS)；其他参数缺省；

CInternetSession::GetFtpConnection 该函数用于建立连接；
CFtpConnection*GetFtpConnection(LPCTSTR pstrServer, LPCTSTR pstrUserName = NULL, LPCTSTR pstrPassword = NULL, INTERNET_PORT nPort = INTERNET_INVALID_PORT_NUMBER, BOOL bPassive = FALSE)；本实验中可以只要前三个参数，另外在使用该函数时要进行异常处理；

CException::Delete 用于删除已经确认的异常 void CException::Delete()。

2.CFtpConnection 类:

MFC 中的类 CftpConnection 管理用户与 Internet 服务器的连接，并直接操作服务器上的目录和文件
其中：
CFtpConnection::GetFile: BOOL GetFile(LPCTSTR pstrRemoteFile, LPCTSTR pstrLocalFile, BOOL bFailIfExists = TRUE,DWORD dwAttributes =FILE_ATTRIBUTE_NORMAL, DWORD dwFlags = FTP_TRANSFER_TYPE_BINARY, DWORD dwContext = 1)；下载存储路径为 pstrRemoteFile，文件名为 pstrLocalFile 的文件；下载成功返回非 0 值，否则返回 0；
pConnection->GetFile(strSourceName,strDestName)

CFtpConnection::PutFile 用于上传文件：BOOL PutFile(LPCTSTR pstrLocalFile, LPCTSTR pstrRemoteFile, DWORD dwFlags = FTP_TRANSFER_TYPE_BINARY, DWORD dwContext = 1)；可以只用前两个参数，后面的用缺省的即可。

3.try-catch:捕获用户输入的信息错误
try{用户输入的信息的代码}
catch(Exception ex){处理异常的代码}

4. CFileFind 包含的成员函数有：
CFileFind::GetFileName 用于取得文件名：virtual CString GetFileName( ) const；
CFileFind::IsDirectory：BOOL IsDirectory( ) const 该函数用于判断查找到的文件是否是文件夹，若是返回非 0 值，否则返回 0。

5.CFtpFileFind 包含的成员函数有：
CFtpFileFind::FindNextFile：virtual BOOL FindNextFile()该函数用于查找下一个文件，与 CFtpFileFind::FindFile（）配合使用；
CFtpFileFind::FindFile：virtual BOOL FindFile(LPCTSTR pstrName = NULL, DWORD dwFlags = INTERNET_FLAG_RELOAD)该函数用于文件查找：查找 pstrName 名的文件，若 pstrName 为空，默认为(*)，bContinue=pFileFind->FindFile("*")；查找成功返回非 0 值，否则返回 0。
CFtpFileFind::CFtpFileFind：CFtpFileFind(CFtpConnection* pConnection, DWORD dwContext = 1)；该函数用于建立文件查找类；

三． 实验方案与实验步骤

实验方案：
1) 建立 MFC 工程
2) 设置 MFC 界面，建立控件和函数、编辑框和变量之间的联系
3) 填充相应的函数命令，实现与 FTP 服务器的连接，获取当前文件目录，完成下载和上传文件等功能

流程图：

四． 实验设备与器材配置

1、VC6.0

2、基于 TCP/IP 协议的 Windows 网络硬软件环境

五． 实验记录

创建 Internet 会话：

```
if(strConnect=="连接")
{
    //创建Internet会话
    pSession=new CInternetSession(AfxGetAppName(),1,PRE_CONFIG_INTERNET_ACCESS);
```

建立 FTP 连接并进行错误处理：

```
try
{
    //试图建立FTP连接
    pConnection=pSession->GetFtpConnection(m_strFtpSite,m_strName,m_strPwd);
}
catch (CInternetException* e)
{
    //错误处理
    e->Delete();
    pConnection=NULL;
    AfxMessageBox("连接错误，请检查地址和用户名密码");
    return;
}
```

控件：

```
if (pFileFind)
[
    pFileFind->Close();
    delete pFileFind;
]
if (pSession)
[
pSession->Close();
delete pSession;
]
m_BtnConnect.SetWindowText("连接");
m_cancel.EnableWindow(TRUE);
m_BtnUpLoad.EnableWindow(FALSE);
   //激活用来输入的文本和编辑框控件
m_EditFtp.EnableWindow(TRUE);
m_EditName.EnableWindow(TRUE);
m_EditPwd.EnableWindow(TRUE);
m_StaFtp.EnableWindow(TRUE);
m_StaName.EnableWindow(TRUE);
m_StaPwd.EnableWindow(TRUE);
```

下载：

先获得当前输入：

```
void CFtpDlg::OnDownload()
{
    //获得当前输入
    UpdateData(TRUE);
    int nSel = m_ListFile.GetCurSel();
    CString strSourceName;
```

获得下载文件的路径和名称：

```
if(strSourceName.GetAt(0)!='[')
{
    //选择的是文件
    CString strDestName;
    CFileDialog dlg(FALSE,"",strSourceName);
    if(dlg.DoModal()==IDOK)
    {
        //获得下载文件在本地机上存储的路径和名称
        strDestName = dlg.GetPathName();
```

```
//下载文件
if(pConnection->GetFile(strSourceName,strDestName))
{
    AfxMessageBox("下载成功！",MB_OK|MB_ICONINFORMATION);
    //禁用下载按钮
    m_BtnDownLoad.EnableWindow(FALSE);
else
    AfxMessageBox("下载失败！",MB_OK|MB_ICONSTOP);
```

上传：

获得当前输入，并禁用一些控件：

```
void CFtpDlg::OnUpload()
{
    //获得当前输入
    UpdateData(TRUE);
    //禁止用来输入的文本和编辑框控件
    m_EditFtp.EnableWindow(FALSE);
    m_EditName.EnableWindow(FALSE);
    m_EditPwd.EnableWindow(FALSE);
    m_StaFtp.EnableWindow(FALSE);
    m_StaName.EnableWindow(FALSE);
    m_StaPwd.EnableWindow(FALSE);
    //禁止查询按钮
    m_BtnConnect.EnableWindow(FALSE);
//调用函数上传文件
if(pConnection->PutFile(strSourceName,strDestName))
{

    AfxMessageBox("上传成功！",MB_OK|MB_ICONINFORMATION);
    while(m_ListFile.GetCount()!=0)
        m_ListFile.DeleteString(0);
    Find();
}
else
    AfxMessageBox("上传失败！",MB_OK|MB_ICONSTOP);
```

六. 实验总结

通过此次实验，我了解了用 MFC WinInet 函数编写一个 FTP 客户端程序的基本步骤，使用 MFC WinInet 函数编写一个简单的 FTP 客户端程序，实现 FTP 客户端软件的基本功能—检索 FTP 文件服务器，上传和下载。在实验中，我初步了解了 CInternetSession、CFtpFileFind 类及 GetFtpConnection()、FindFile()、FindNextFile() 、GetFile()、PutFile()函数的使用，并学习了 FTP 的编程的过程，这个实验让我对 MFC 的搭建也更加熟悉，更清楚地认识了 MFC。

七. 思考题或讨论题

1. 简述 FTP 的作用原理？

FTP（File Transfer Protocol，文件传输协议）是一种用于在网络上进行文件传输的标准协议。它允许用户从一个计算机向另一个计算机发送和接收文件，通常用于网站维护、软件发布、文件备份等场景。FTP 通过客户端-服务器模型工作。客户端是用户操作的一端，负责发起文件传输请求；服务器端则是存储文件和处理传输请求的一端；在开始文件传输之前，客户端需要与 FTP 服务器建立连

接。通常，FTP 服务器监听在标准端口 21 上等待客户端连接请求。连接建立后，客户端可以通过认证来获取对文件系统的访问权限；客户端通过控制连接（通常在端口 21）向服务器发送命令，如上传文件、下载文件、列出目录内容、创建目录等。FTP 定义了一套命令集，每个命令对应一个特定的操作；当需要传输文件内容时，FTP 使用数据连接来实际传输文件数据。数据连接可以是两种类型：主动模式：客户端向服务器发起数据连接，客户端指定一个端口用于数据传输；被动模式：服务器向客户端发起数据连接，服务器指定一个端口用于数据传输。数据传输可以是二进制或 ASCII 模式，具体取决于传输的文件类型（如文本文件或二进制文件）；FTP 使用两个不同的连接：控制连接和数据连接。控制连接用于传输命令和响应（如用户验证、文件操作请求等），而数据连接用于传输实际的文件内容；原始的 FTP 协议不提供加密功能，所有数据（包括用户名和密码）都是以明文形式传输的。为了增加安全性，通常可以使用安全的 FTP 协议（如 FTPS）或在 FTP 上使用 SSH（如 SFTP）来加密数据传输。总体来说，FTP 通过简单而直接的方式允许用户在计算机之间传输文件，它的设计考虑了可靠性、灵活性和广泛的应用场景，使其成为一个重要的网络文件传输协议。

2. 调研说明 WinSock 和 Berkeley Socket 在用法的相同点和不同点。

WinSock 和 Berkeley Socket 都是用于网络编程的接口

相同点：

1.网络编程接口：

Berkeley Socket：最初由伯克利大学开发，定义了一组 API，用于在应用程序中创建网络通信套接字，实现数据的发送和接收；WinSock（Windows Sockets）：是微软在 Windows 操作系统上实现的一套网络编程接口，它的设计借鉴了 Berkeley Socket，提供了类似的功能和接口。

2.功能性：

两者都允许应用程序创建网络连接、传输数据、关闭连接等基本网络通信操作；提供了 TCP 和 UDP 等传输协议的支持，允许应用程序根据需要选择合适的协议进行通信。

3.跨平台性：

Berkeley Socket 最初设计为跨平台，因此可以在各种操作系统（如 Unix/Linux、Windows 等）上使用；WinSock 虽然首先是为 Windows 开发的，但它的接口和部分功能也有对应的跨平台实现（例如在一些类 Unix 系统中），使得部分代码可移植性较高。

不同点：

1.平台依赖性：

Berkeley Socket：原始的 BSD 套接字 API 是跨平台的，但不同操作系统实现可能有些许差异，尤其是在 Windows 上的实现需要通过兼容层（如 Cygwin）或使用第三方库（如 Boost.Asio）来支持；WinSock：是微软专门为 Windows 操作系统开发的网络编程接口，因此在 Windows 平台上使用较为方便和直接。

2.API 调用方式：

Berkeley Socket：API 调用和参数传递方式更加接近传统的 Unix 系统调用风格，通常使用简单的函数调用来完成网络操作；WinSock：虽然功能类似，但在 API 的调用方式、函数名称和部分参数上有所不同，与 Windows 编程模

型更加一致。

3.扩展功能：

WinSock：在 Windows 平台上，WinSock 提供了一些额外的功能和扩展，如与 Windows 消息处理的集成、事件通知机制等，使得在 Windows 环境下编写网络应用更为方便；Berkeley Socket：原始的 BSD 套接字 API 在功能上相对较为简单，更专注于基本的网络通信操作，不像 WinSock 那样直接与操作系统的其他特性集成。

4.历史和发展：

Berkeley Socket：是由伯克利大学开发，作为 Unix 系统的一部分首次推出，后来被各种操作系统广泛采纳和实现；WinSock：是在 Windows 3.1 时代由 Microsoft 引入，随着 Windows 平台的发展不断进行了更新和扩展，逐渐成为 Windows 网络编程的标准接口。

总结，Berkeley Socket 和 WinSock 在功能上有很多相似之处，都是重要的网络编程接口，但在实现细节、平台适用性和扩展功能等方面存在明显的差异。选择使用哪一种取决于开发者的操作系统平台和具体的需求。