项目五 即时通信系统（聊天室）

版本号v1.0

## 1.任务需求描述

1. 服务器实现同时接入多个客户端
2. 结合项目三，每个客户通过认证，并各自使用不同的帐号
3. 客户端可以看到当前接入用户的列表
4. 结合项目二，客户端可以选择与某个指定用户进行聊天，或者选择将数据发送给所有客户
5. 结合项目三，客户可以上传或下载文件
6. 两个客户之间可以通过服务器的中转在线相互传递文件，即不是一边先上传，完毕后，另一边再下载，而是通过服务器代理中转的“端到端”传输（类似QQ的在线传输）
7. 客户机能够接收服务器的广播命令

## 2.需求分析

项目五是项目二、项目三的整合，同时也是一个吸取之前经验教训，优化设计的方案。具体来说，为更好满足任务需求，需完成以下3个主要任务：

1. 界面整合修改
2. 聊天消息PDU重新设计
3. 重要代码重构

## 3.关键技术分析

1. 质询报文的构成格式分析：应包含整数的长度，整数数值等要素。由于整数长度可变，且存在网络字节序的问题，实现上可考虑应用位域的方法或按位运算的方法组织整数。
2. 认证算法的实现

分析：项目需求中制定的认证算法模拟CHAP协议认证过程。CHAP协议认证过程的加密算法在需求分析中已经说明。简而言之，在成功获得服务器传来的整数后，进行加法，异或的简单运算，并准确传回给服务器比较，即可完成认证。

1. 协议设计要点

分析：除了满足功能外，为保持效率，注重字节对齐等问题。

1. 客户端在线文件直传

分析：初步方案为，利用项目三的传输文件的基础上，当一个用户发起一次文件直传请求，服务器会通过CtlPDU向对方客户端确认，同意后，分别向双方发起建立一个数据连接，成功后向发送方发送一个传输文件开始的FilePDU，之后在进行转发，收发双方使用停等协议确保文件发送成功，并进行必要的差错控制。

## 4.数据结构设计

服务器，客户端PDU通用。PDU包括质询报文PDU，文件列表传输报文PDU，文件传输报文PDU，控制命令报文PDU，以及聊天消息报文PDU。以下列出各PDU并注释。

1. **质询报文PDU**

#define INQUIRING\_PDU\_TYPE 1 //质询报文类型代号

typedef struct InquiryPDUS {

char type; //PDU类型，值为INQUIRING\_PDU\_TYPE = 1

char num; //存储质询数个数（范围1-16）

char each\_len[4]; //表达每个质询数的长度。每个表达占2位，值1、2、3、0分别表示质询数长度为1、2、3、4。4个字节32bit，最多能表达16个数。

char inquiry\_buf[INQUIRY\_BUF\_SIZE]; //每个质询数，高位存储在前，低位存储在后，符合网络字节序

} InquiryPDU;

1. **控制命令报文PDU**

#define CTL\_PDU\_TYPE 2

#define CTL\_BUF\_SIZE 255 //一个报文载荷所能存储的最大字节数

//以下为命令位cmd取值，改为2开头的两位数，与CTL\_PDU\_TYPE看齐

#define REQ\_ACCOUNT 20 //服务器向客户机请求用户帐号

#define REQ\_FILELIST 21 //客户机向服务器请求获取文件列表

#define REQ\_DOWNFILE 22 //客户机向服务器请求文件下载

#define REQ\_UPFILE 23 //客户机向服务器请求文件上传

#define REQ\_TRANSLATE 24 //文件中转确认信息

#define RE\_ACCESS 25 //服务器告知客户机登陆成功

#define RE\_TIMEOUT 26 //服务器告知客户机超时

#define RE\_ERR\_ACCOUNT 27 //服务器告知客户机用户名错误

#define RE\_ERR\_INQUIRY 28 //服务器告知客户机质询结果不通过

#define RE\_ERR\_FILENAME 29 //服务器告知客户机文件名错误

typedef struct CtlPDUS{

unsigned char type; //PDU类型，值为CTL\_PDU\_TYPE = 2

unsigned char cmd; //控制命令，可选参数见于上文宏定义

unsigned char len; //报文中载荷的长度，0~255

unsigned char id; //在文件上传时使用，标识文件传输对象为服务器或者其他用户

char buf[CTL\_BUF\_SIZE]; //载荷，根据不同的命令cmd值

//可填用户名、上下载的文件名等

} CtlPDU;

说明：该类型PDU主要用于客户机的控制套接字与服务器交互必要的控制信息。例如登录时，服务器向客户机请求帐号；得到客户机返回后，经验证后再发送质询PDU；得到质询返回值且验证正确后，客户机可以根据需求向服务器发送请求，包括获取文件列表、下载文件、上传文件等。

1. **文件/文件列表传输报文PDU**

#define FILE\_PDU\_TYPE 4 //文件传输类型

#define FILE\_BUF\_SIZE 255 //一个报文载荷所能存储的最大字节数

//数据连接建立后的第一帧皆由服务器发送至客户机

//通过不同No值，告知该连接的功能

#define GET\_FILELIST\_START 2

#define DOWN\_FILE\_START 3

#define UP\_FILE\_START 4

typedef struct FilePDUS{

unsigned char type; //PDU类型，值为FILE\_PDU\_TYPE = 4

unsigned char No; //传输时，PDU编号，可选参数0 或1第一份文件/文件列表PDU，No值可选2、3、4

unsigned char len; //默认为报文长度总长度。当值为0时，表示传输结束

char buf[FILE\_BUF\_SIZE]; //载荷，具体包括文件列表各文件名或文件数据流

} FilePDU;

说明：当服务器在控制套接字上收到客户机的不同的请求之后，向客户机新建数据连接，发送不同类型的文件/文件列表PDU第一帧，具体由PDU中的No值来区分，客户机收到第一帧后，做对应的准备处理，之后开始正常的文件/文件列表传输。

1. **聊天消息报文PDU**

#define CHAT\_PDU\_TYPE 6 //聊天报文类型代号

#define CHAT\_BUF\_SIZE 255 //一个报文载荷所能存储的最大字节数

#define ADD\_FRD 60 //服务器通知客户机，添加聊天用户指令

#define DEL\_FRD 61 //服务器通知客户机，删除聊天用户指令

#define FRD\_MSG 62 //单发聊天信息

#define BROCAST\_MSG 63 //群发聊天信息

typedef struct ChatPDUS{

unsigned char type; // PDU类型，值为CHAT\_PDU\_TYPE = 6

unsigned char cmd; //命令，包括好友上线、下线、单播信息、广播信息

unsigned char frdno; //用户ID（Friend No）

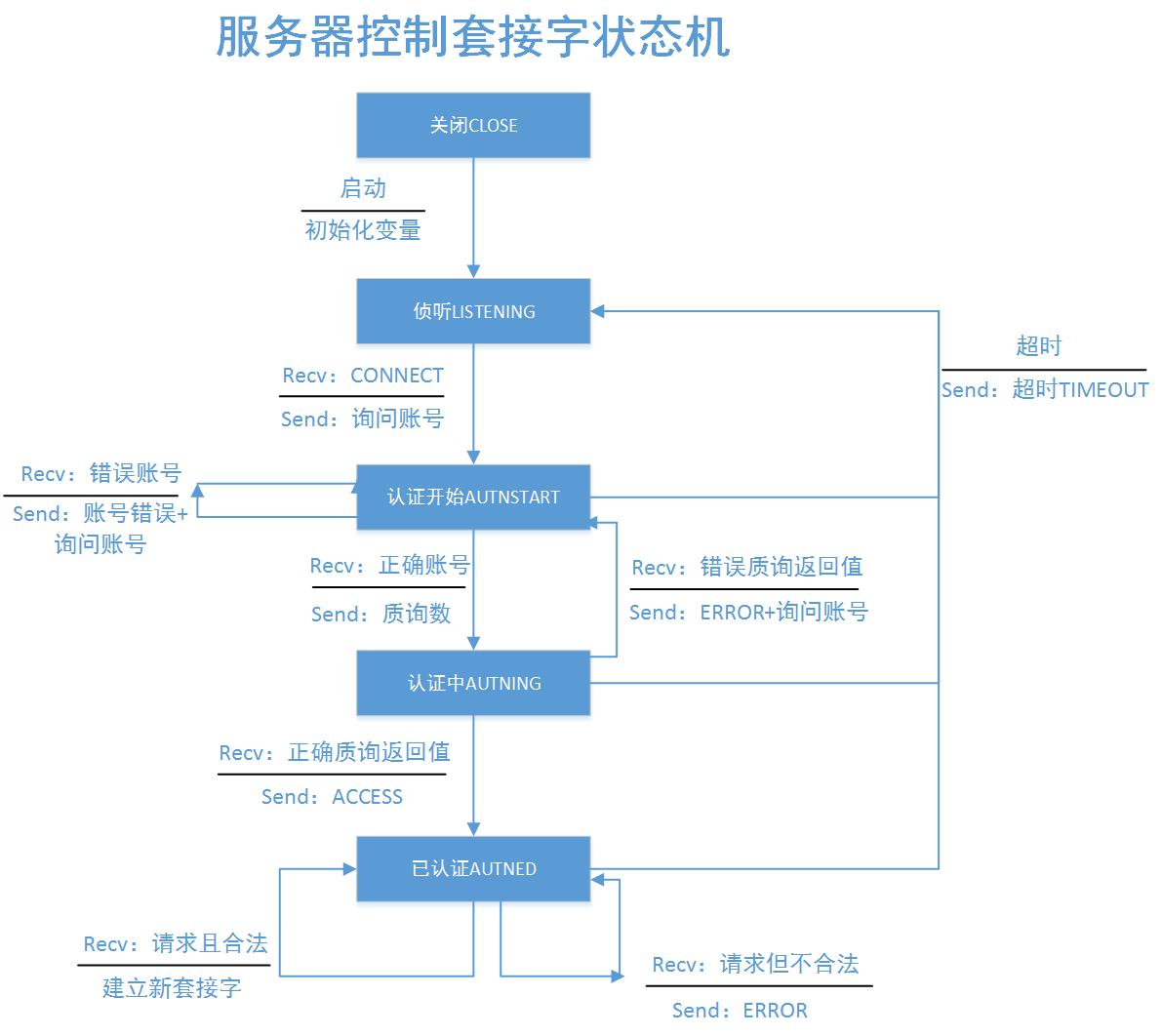
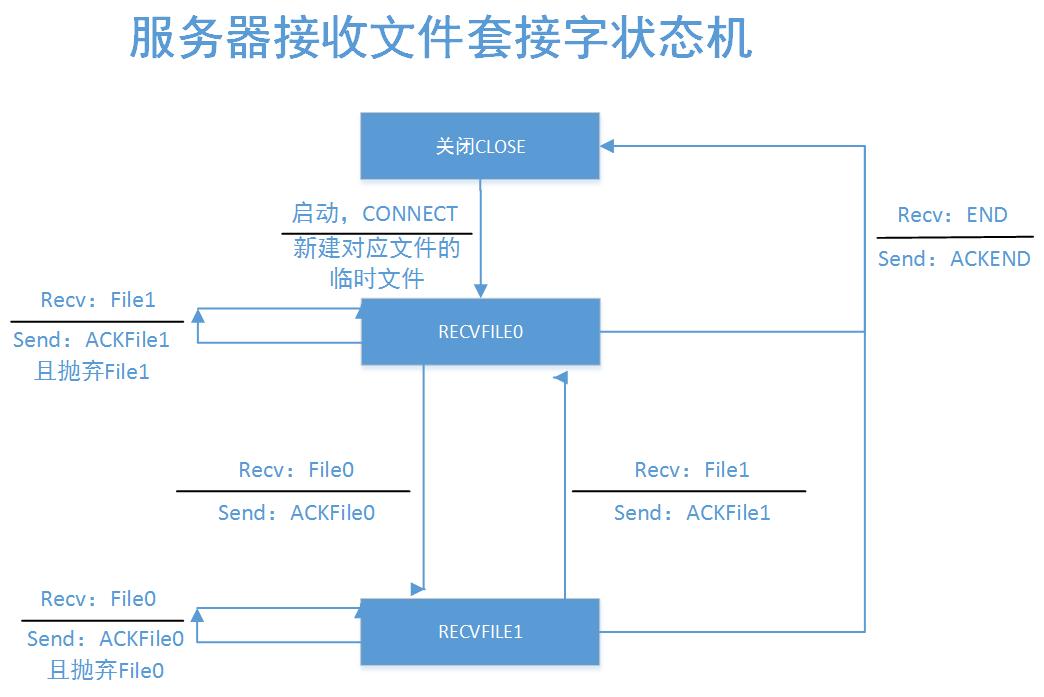
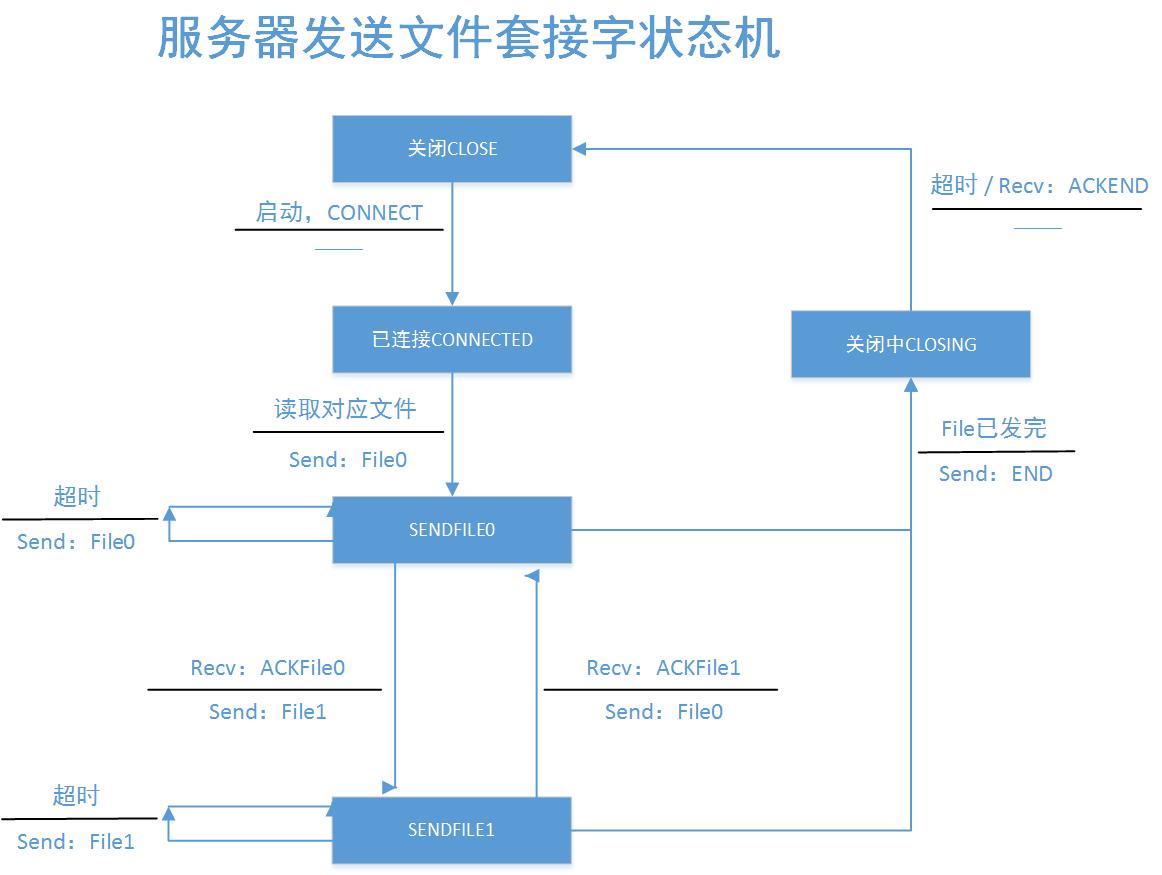
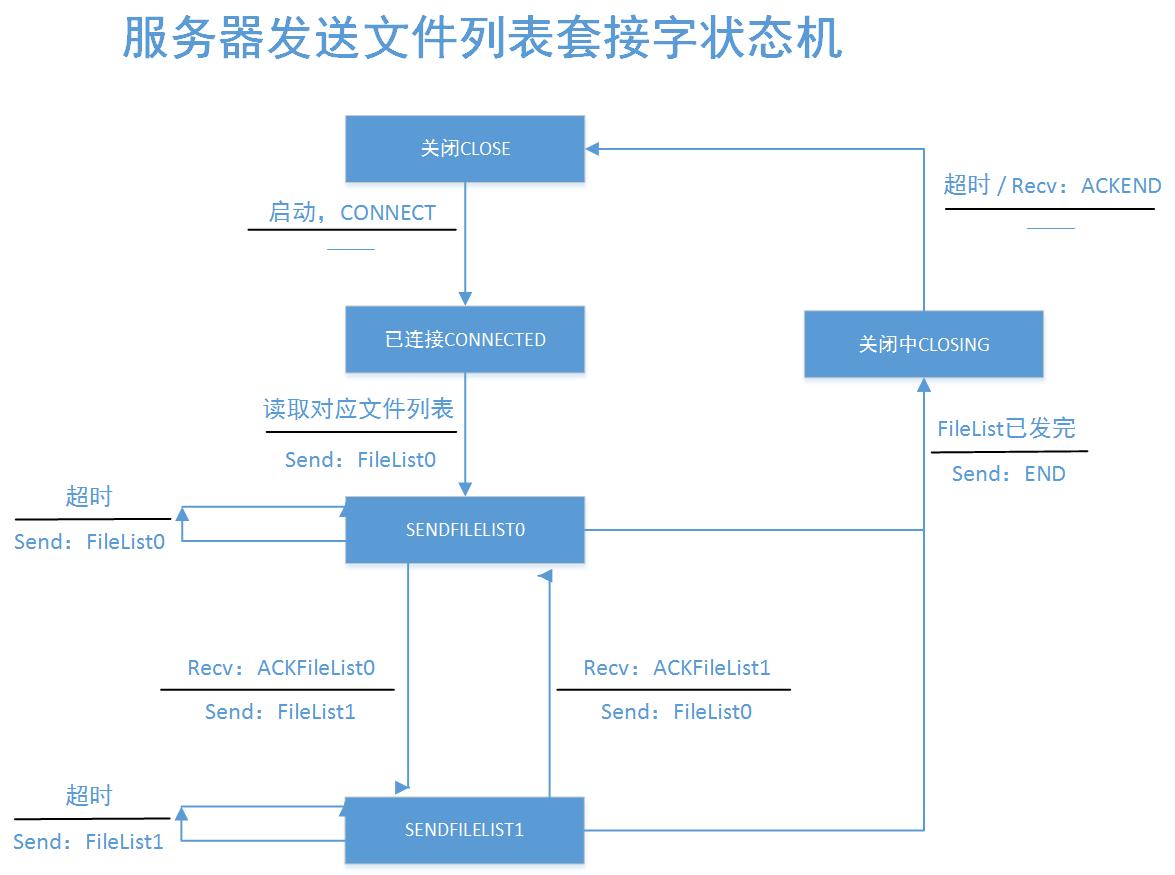
char msg[CHAT\_BUF\_SIZE]; //载荷，包括聊天消息或上下线用户名等

} ChatPDU;

说明：基本沿用了项目二的PDU设计，略作调整，命令cmd位标识不同的命令，好友上下线等都是有服务器发现有变动时主动发给客户机。例如有新用户上线，服务器告知所有已在线用户上线的新用户的情况，并将所有已在线用户的信息告知新用户，下线类同。

## 5.程序框架及流程设计

以下给出服务器在4种关键状态下的有限状态机。客户端的状态机可以对应推导出。



## 6.软件设计方案修订

6.1、修订原因

NULL

6.2、修订后的版本号

NULL