**一、远程服务器：**

**登录线程**

服务器启动时只有一个线程在工作，那就是登录线程

此线程监听各客户端的登录指令，一旦有客户发出登录请求，即刻为此客户端启动新的线程并创建新的Socket用于数据转发。

**消息转发线程**

新的socket创建后绑定一个可用的端口号，并将此端口号通过登录socket发送给客户端（因为新的socket端口号为随机分配，客户端在未获得端口号的情况下无法与之通信），客户端收到服务器分配的新的端口号后，向新端口号再次发送请求注册信息，此时客户端与新的socket建立连接，新socket向客户端回应注册成功指令，客户端注册成功，然后保存当前客户端信息（客户端ID，与客户端地址、与此客户端信息交互的socket套接字）

注：将与客户端通信的套接字保存在客户信息中极其重要，因为客户端的地址和与之通信的服务端socket向对应，当其他客户端需要向此客户端发送消息时，服务器使用保存的套接字进行通信，其他套接字无法与注册时保存的地址进行通信！

新线程中的新socket开始监听客户端数据，音频、视频、其他数据、退出登陆等指令

**心跳包线程**

客户端定时向服务器的新端口号发送心跳数据包，服务器将收到客户端心跳包的时间进行记录，同时，心跳包线程按照固定频率遍历已登录的客户端，将客户端上次心跳时间超出阈值的客户从客户列表中删除。

**二、语音视频传输客户端**

**Ui界面：**

点击小窗可进行我方图像和对方图像位置切换，小窗可自由拖动

保存用户ID以及下次是否自动登录

**语音视频传输客户端程序**

一个UDP接收器（UdpReceiver）、一个UDP发送器（UdpSender）

UdpSender只负责向服务器发送需要传输的视频语音消息

UdpRceiver 一方面负责接收服务器发送的视频语音消息、心跳包、其他指令，另一方面需要向服务器发送登录指令、呼叫其他客户端指令，退出指令等等

**客户端登录线程（位于UdpReceiver）**

客户端向服务器的登录端口号发送请求登录指令，随即启动数据接收线程，时刻准备接收服务器发来的消息。收到服务器发来的新端口号newPort后，向newPort再次发送请求注册消息。

**客户端数据接收线程**

QUdpSocket数据是在槽函数中进行接收，当收到消息后触发信号，然后在槽中读取，测试发现，槽函数所在线程与QUdpSocket实例化所在线程一致，目前UdpReceiver在主线程中被实例化，暂未发现卡顿问题。

UdpReceiver的接收槽函数，接收服务器两个不同端口发送的数据，第一个是注册端口，第二个是数据端口。(此socket分别与两端口建立了连接）客户端注册成功后便记录了服务端的数据端口！

**通话建立**

客户端A呼叫客户端B:

A启动摄像头、音频并定时向服务器发送消息（消息头中包含有B的ID）；

服务器查看B是否在客户列表，

B不在客户列表，服务器向A发送B离线指令，A停止呼叫，关闭

B在列表，服务器向B发送连接请求，B同意请求后启动消息收发，一方面显示A发送的消息，另一方面向A发送消息。

服务器更新两客户端状态，在A的信息中添加B，在B的信息中添加A，表示双方正在通话

**通话终止**

通话中的任意一方均可以终止通话

1. A向服务器发送通话终止指令并关闭通信外设，
2. 服务器收到指令后将A中B的信息删除，并向B发送断开连接指令
3. B收到断开连接指令后，关闭通信外设并向服务器发送断开指令，
4. 服务器收到指令后将B中A的信息删除，并向A发送断开连接指令

至此，双方完全断开连接。