**RemoteControl 开发文档**

1. **介绍**

本文档描述了一个远程控制软件的功能，使用细节，使用技术和软件模型。本远程控制软件是通过局域网的ip地址实现对对方电脑的鼠标，键盘等事件的控制。可以使用本软件方便的实现对他人电脑的控制，帮助他人解决无法解决的问题。致力于开发更流畅的远程控制，将不断改进和优化。

1. **使用细节**
2. 输入ip地址和端口 ( 默认为10086 ) 号找到对方电脑。
3. 对方电脑要先点击共享屏幕然后才能让对方电脑连接。
4. 连接方电脑点击连接按钮便可连接对方电脑，如果在指定时间（8s）内没有连接成 功，则认为连接失败。
5. 连接成功后连接方关闭窗口即可断开连接。
6. 被控制方也可以主动断开连接。
7. **成员分工**

余泓钰2022051615287：负责界面和获取图片数据，数据的前后端交互部分。

夏瑞涛2022051615270：负责网络层和多线程处理，前后端事件处理的交互部分。

陈锐凝2022051615288：负责鼠标键盘等各种事件的处理，和前后端事件传递部分。

1. **技术实现**

对于网络层，完全使用Boost::asio的网络库通过异步网络io的API来实现全双工的客户端与服务端的交互。通过采用Boost::thread，QThread相结合的多线程编程更好的实现了网络层和Qt事件循环线程的分离，网络层完全使用Std与Boost的结构，而Qt事件循环线程则采用QThread的机制更好的实现对象树机制。网络层则通过使用闭包的方式来实现对Socket生命周期的管理。Qt事件循环及创建的继承QThread的子类通过使用互斥锁和阻塞等待的机制的有关的Qt类来实现了与网络层之间的通信。

对于Qml，通过编写图片提供类的方案实现了网络端获得的图片向上传递的过程，再通过编写qml rootcontext 的 contextproperty 的类来实现了前端Qml层事件的向下传递。

对于传递的各种数据事件，通过创建事件节点的类EventNode 来储存各种事件的信息，然后通过eventToDo函数来对各种事件做分发。在通过外部可执行程序xdotool使用Qt类来执行该可执行程序来实现对各类事件的实际操作。

1. **技术难点**

一.后端

（一）网络层方面：

如何正确实现网络层和多线程操作，如何确保各数据在多个线程都可能有引用的条件下，正确处理生命周期的问题.比如说

1.正确的创建，删除和复用数据。在比如说正确协调相同资源，共享对象的互斥操作，

来避免死锁和饥饿问题的出现。

1. 正确释放已经弃用的线程，避免强制停止和为终止线程导致对象无法删除的问题。
2. 正确处理网络层粘包和闭包问题。

（二）键鼠方面

1. 使用什么API或者什么程序来实现对鼠标键盘等实现的操作。
2. 前后端通信

如何实现前端和后端之间的通信，将后端的图片正确的传递到前端，然后在前端正确的打开图片。

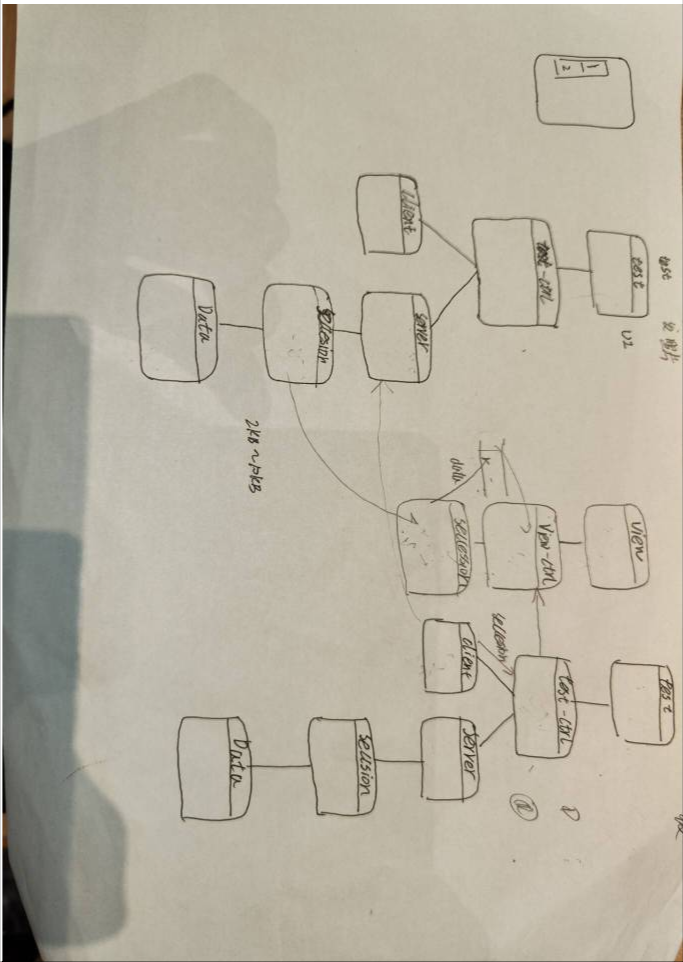
三.前端

前端图片具有缓冲区，导致从相同路径获取的已经改变的图片资源无法被前端重新加载。使用路径加随机数的方式避免路径不变的问题

1. **开发缺陷**

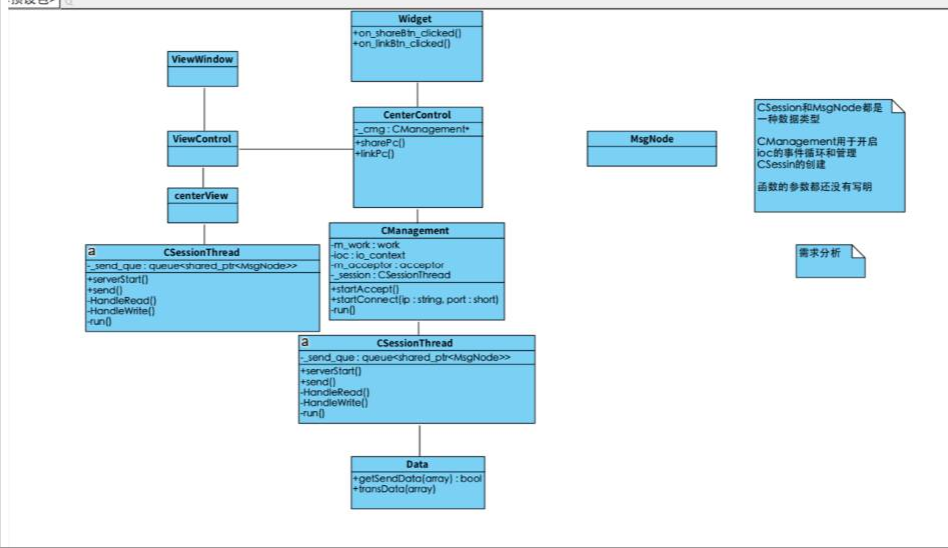
未使用流媒体的机制来传送数据，而是使用图片的方式，导致帧率十分的低对网络环境的要求十分的高。前端界面比较呆板不够fashion。按键事件偶尔会出现按键释放后，对端电脑任然在长按的问题。

1. **开发过程**
2. 初版类图，认为网络层就只需要一个client类和一个server类即可，而且没有包含对按键鼠标事件的描述。

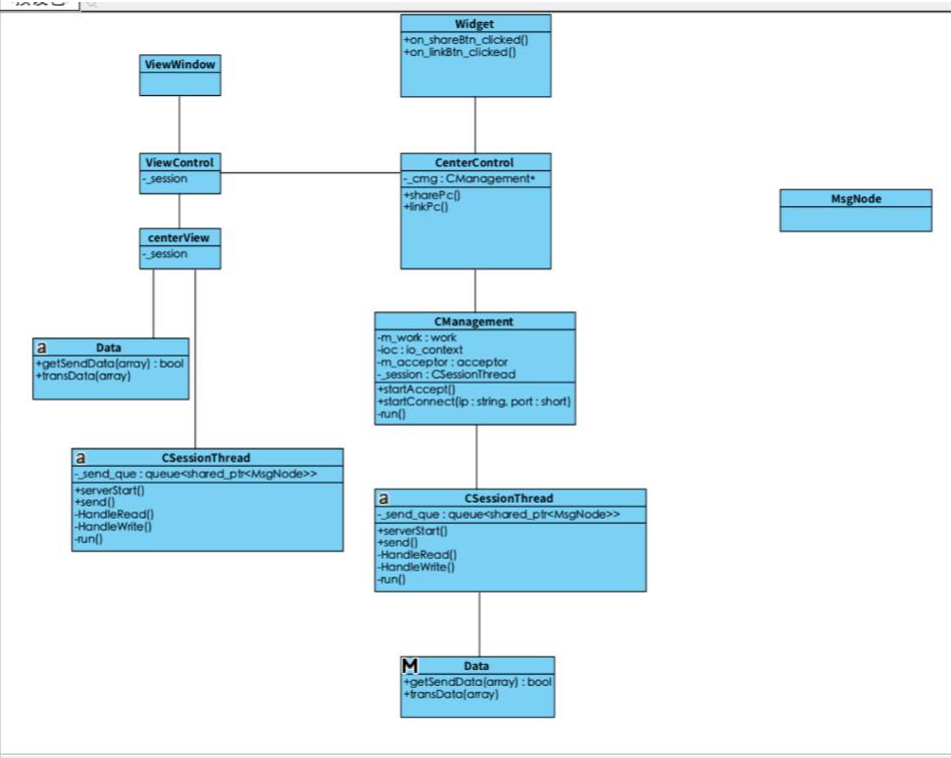


1. 第二版类图，完善了网络层，使用cmanagement来管理用CSession

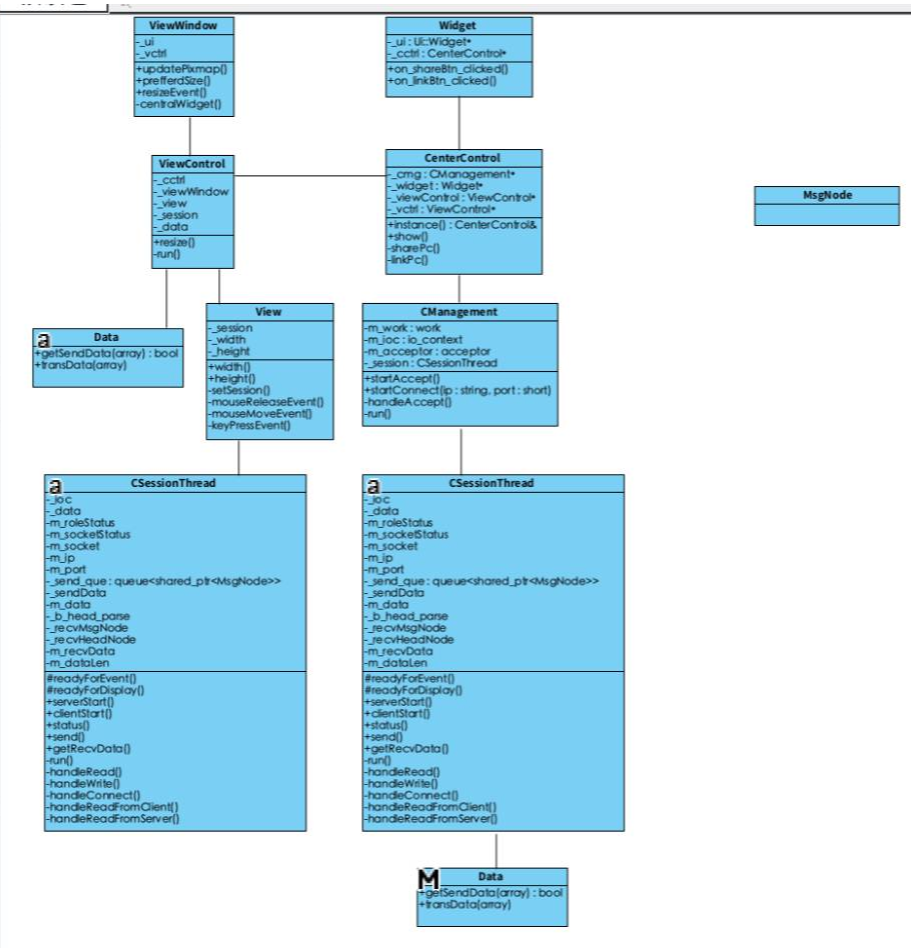
Thread封装的socket 的创建，将server和client整合到一起，因为一个交互的一端即是服务端又是客户端。



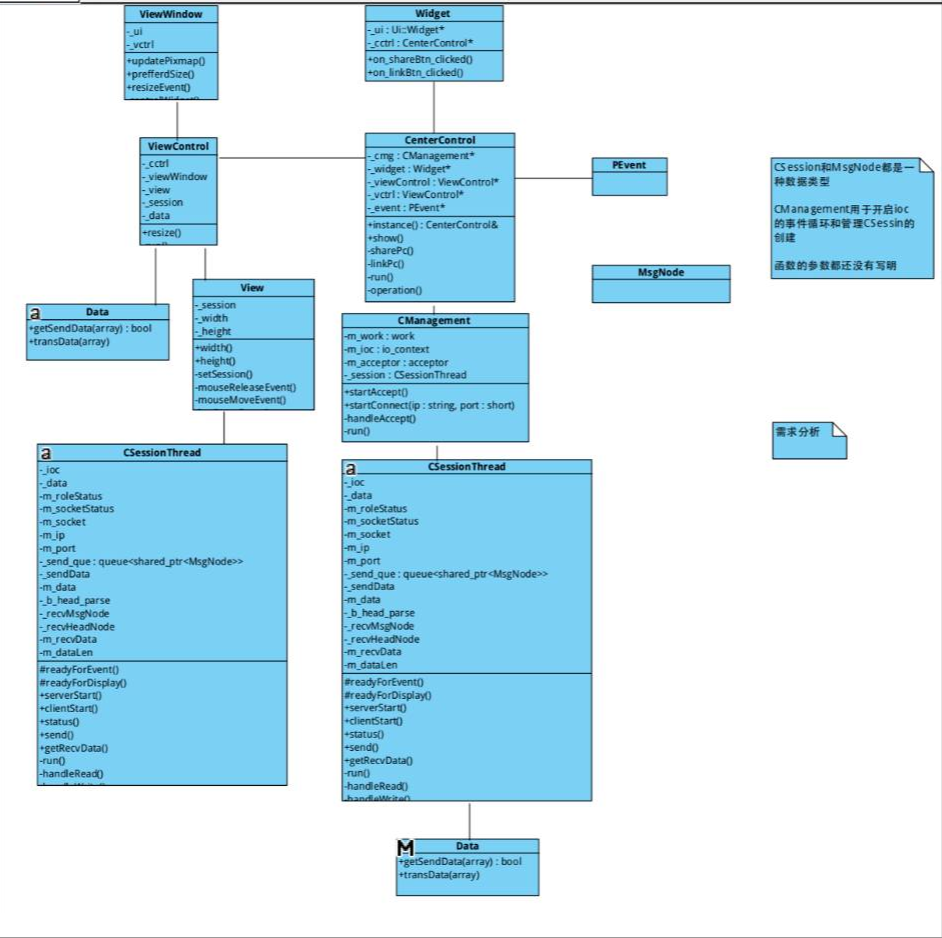
1. 第三版类图，显示连接画面的类centerView来直接控制他的两个工具类，data和CSessionThread。



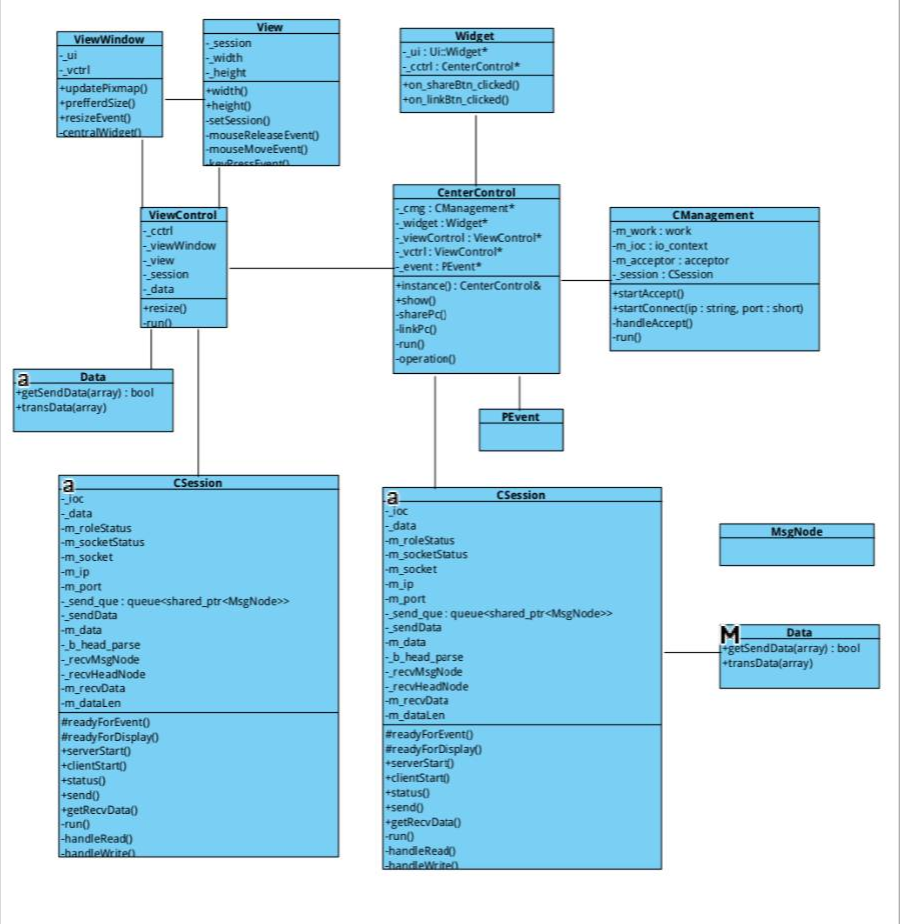
1. 第四版类图，将工具类data直接由viewcontrol来管理，将图片数据解出来。



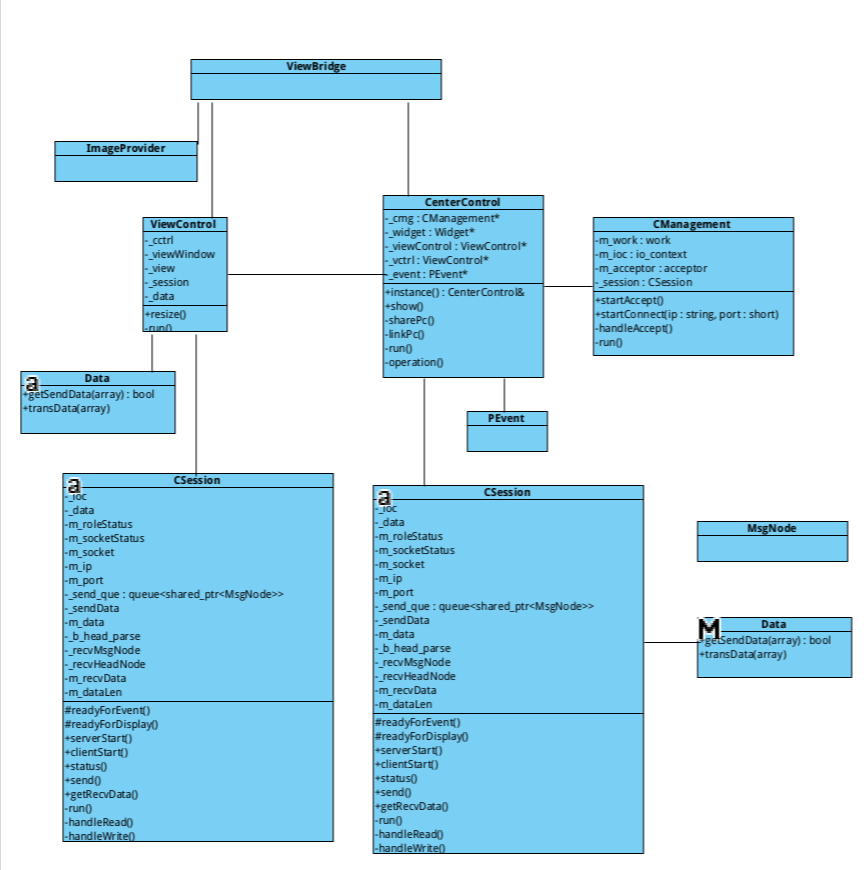
1. 第五版明确了事件处理类PEvent的所处位置



1. 第六版类图，明确了数据的获取和解数据都应该由viewcontrol来完成，然后将数据提供给view来显示，然后将事件传递给viewcontrol，由他来传递给对端电脑。



1. 第七版类图，通过viewBridge来实现Centercontrol和ViewControl与前端QML的交互。



1. **类及功能**
2. ViewBridge: 提供前后端交互的服务。
3. ImageProvider: 为前端提供图片。
4. CenterControl: 程序的总控制类，能控制ViewControl的创建，和接受的按键鼠标事件交给PEvent。
5. ViewControl: 通过控制CSession和Data实现对图片的接收，处理然后传递给ViewBridge。
6. CManagement：控制CSesion的创建，分发和网络层io\_context的管理。
7. PEvent：控制xdotool来实现鼠标按键事件的分发和处理。
8. Data：对图片进行转字节流，和字节流转图片。
9. CSession：同时作为客户端和服务端，进行两端全双工的通信。
10. MsgNode：存储网络传输过程的数据的容器。