网络SDK设计方案

###### 接口

实现登录、获取设备列表、实时视频、录像查询和回放、告警回调等接口。见文件kenetsdk.h。



###### 实现

程序静态结构如下图所示：



消息控制核心实现采用command模式（参考大华sdk对DVR设备的处理），将不同消息的处理分派给不同的Channel类，进行发送和解析。Device是Channel的组合，负责消息的分发与链接。Channel类中消息的创建和解析委托给MsgProcess类进行处理，对于不同的协议，派生不同的MsgProcess。

Controller类用来封装接口，这里有3个接口，链接接口，录像接口，实时视频接口。每个单独的接口可以用来独立执行功能。使用device的派生类来实现接口。

对于通讯的3层处理，数据收发层采用QAbstractScoket进行通讯，协议处理层为Channel类，接口层是controller类。Device类将协议处理层和接口层连接起来，接口层的实现。

程序调用流程：

登录流程：用户在登录前首先要创建一个主设备，主设备用于保存网络参数、错误码等全局参数。然后用户需要建立一个Connection设备对象，调用ConnectDevice接口，连接到消息服务器，然后调用LoginServer接口，Connection设备对象会将登录行为委托给ChLogin对象进行处理，等待ChLogin的响应，然后返回结果给用户。如下图所示：



监听流程：用户首先创建一个主设备，调用ListenDeviceConnet开启监听。当设备有链接到来时，创建一个ConnectionDevice对象。然后可以使用此对象进行其他操作。



实时视频流程：用户在进行实时视频前需要先创建ConnectionDevice对象（见登录流程），然后创建RealPlayController对象，RealPlayController通过ChRealStart发送媒体请求消息，然后通过ChRealData请求视频数据，ChRealData会将视频数据用signal的方式发送给RealPlayDevice的Slot，RealPlayDevice调用回调函数通知用户视频数据。

****

录像查询：用户在进行录像查询时首先要创建RecordQueryDevice。使用接口函数FindFile进行查询时，每次查询一小时的录像文件，循环查询，直到找到录像文件，到达本次查询的结束时间时返回。使用ChRecordQuery对象发送查询消息，在发送第一条查询消息时等待消息返回，ChRecordQuery通过signal将收到的录像信息发送给RecordQueryDevice。在使用FindNextFile查询录像时，若是查询到最后一条并且最后一条录像信息的结束时间小于查询的结束时间，则使用最后一条录像信息的结束时间为开始时间，查询一小时内的录像文件，直到查到文件或者到达本次查询的结束时间为止，若没有录像文件，则没有下一条信息。

对于08协议设备，一个ConnectionDevice只能拥有一个RecordQueryDevice。



录像下载：

对于一个录像文件，用户可以使用回调函数获得数据，也可以告知sdk下载到相应文件，或同时进行。

Sdk对于正在下载的文件应该进行保护，避免写冲突。

###### 可扩展性

网络SDK程序的主要扩展方向：

1. 是现有协议增加新的消息，即设备增加新的功能。要定义新的controller接口，使用新的Device派生类进行实现，增加新的channel，在MsgProcess中增加新的函数用于协议的解析。
2. 需要使用新的协议实现现有功能。需要派生MsgProcess和MsgDispatch，并实现其中的方法。
3. 通讯方式的改变，如从Tcp改为Udp。将新的通讯方式封装成QAbstractSocket的派生类，添加新的通讯类型，在工厂类中构造此类。