程序流程：

1. 客户端向服务器发出连接，两者建立好TCP连接；
2. 客户端向服务器发出自身ID，服务器建立好clientId与ClientAgentId的映射表；
3. 客户端向服务器发送数据（头部+负载，其中头部某字段规定了需要进行信息交互的对端客户端的clientId）;
4. 服务器根据映射表，找到需要的客户端，从而进行中继。

补充：

1）所谓中继，就是通过服务器实现客户端A与客户端B之间的消息交互。基于基类，主要是服务器readBack后调用对端的ClientAgent->sendPackage；

2）无论是客户端还是服务器，都是发送/接收完整的一个数据包后，才会进行下一个数据包的接收/发送。但对于非阻塞的Epoll这并不是说注册到它里面的某个fd，会一个读操作后紧接着一个写操作，而是读写操作不定次数的交互。对于第一句话的理解，应该是保证数据接收/发送的有序正确性。

离线消息存储机制：

客户端一旦与服务器断开连接，服务器向该客户端所对应的套接字里面发送数据时，就会出现错误；因此对于服务器而言，一旦发生写错误，则认为对应的客户端发生连接中断。

连接中断时触发的操作：

1. 发生中断时，离线消息即为该客户端对应的ClientAgent中的接收到的数据buffer（虽然每次都是由对端客户端的ClientAgent的readBack函数中的该ClientAgent->sendPackage触发）,若它不为空，则说明存在离线消息，并将该buffer中的数据存储到磁盘文件中，文件以该clientId为字眼命名；
2. 回收该客户端对应的ClientAgent（基类有其回收机制，在Epoll回收前，因进行步骤1的操作），此时对端客户端对应的ClientAgent中的readBack函数中因找不到该ClientAgent而无法进行数据的发送（因此要进行该判断操作，并且判断该ClientAgent是否存在尤其重要）。

与客户端连接好后，根据客户端发来的clientId先判断是否存在相应字眼的文件名，若存在，则存在离线消息，需先进行离线消息的发送操作，然后进行相应的中继流程；若不存在，则不存在离线消息，直接进行相应的中继流程。

补充：1）由于进行磁盘I/O为低速操作，为避免阻塞主线程，该类操作交于子线程进行。

2）客户端中断连接时，服务器第二次wirte会出错，并且发送sigpipe信号，该信号将中断进程，所以应该捕捉该信号以改变其默认操作。