# ftpserver关键模块详细设计

## 登录模块

登录模块中维护一个User类，类图如下：

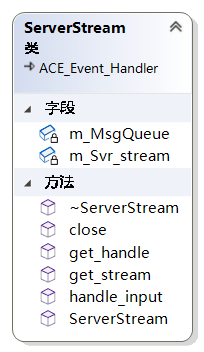


其中该类维护了当前用户所在路径，当前用户ip、用户名、密码、状态等。当有新的连接时候则实例化一个User对象，若登录成功则维护在handlerToUser中。

handlerToUser的数据结构是一个哈希表，key为新连接的通道id，value为当前连接上的用户User。

## 连接通道模块

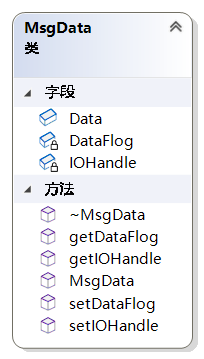
当客户端与服务器建立连接时，会新建一个通道Stream，该类图：



该类继承ACE\_Event\_Handler，成员变量有ACE\_SOCK\_STREAM类型的m\_Svr\_stream用于控制信号的传输，同时组合了自定义数据TashThread（第四任务模块介绍到）类型的m\_MsgQueue用于处理特定业务逻辑。当通道有信号发生时，调用handle\_input,该方法主要当前收到的信息封装成MsgData（第三消息类型模块介绍到）并且放入到m\_MsgQueue消息队列中。

## 消息类型模块

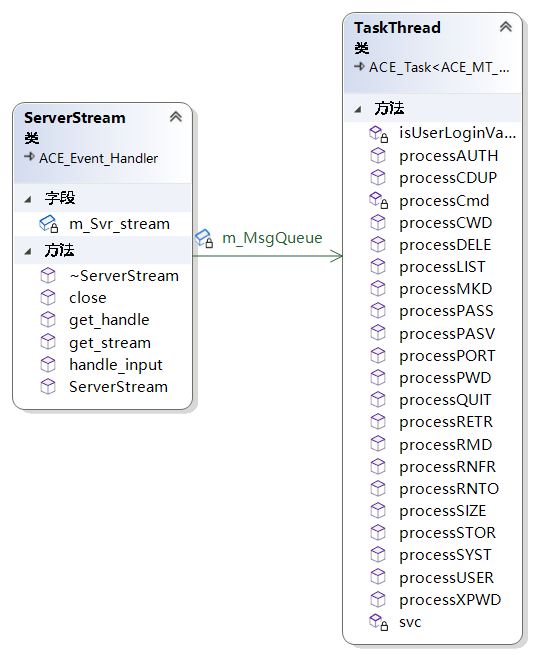
服务器将收到的消息封装成MsgData，类图如下：



其中Data保存客户端传输命令与参数，成员变量IOHandle的类型为ACE\_Handle，用于保存该传输信号的通道。TashThread处理该数据对象的时候需要获取该通道返回结果给客户端。

## 任务处理模块

该模块为业务处理核心模块，封装在TashThread中，类图如下：



该类为ServerStream的组合类，同时继承自ACE\_Task,该类在Stream构造函数中被维护的单例线程池所实例化。当该类中的消息队列不为空的时候，也就是父类的getq（）方法不阻塞的时候，说明有任务需要处理。此时，在消息队列中取出消息MsgData进行业务处理。

## 系统的关键数据结构

维护用户信息的handlerToUser变量使用数据结构为unordered\_map。由于处理业务逻辑经常需要获取用户状态，因此维护成unordered\_map读取用户状态效率较好。

维护通道的数据结构为双向链表list，用户建立连接后将该通道信息维护在list类型的通道池中。

任务队列线程单例池用ACE\_Singlton维护，本系统使用10个线程初始化单例池。其模板实例化为：ACE\_Singleton<TaskThread,ACE\_Thread\_Mutex>