

# 局域网狼人杀对战系统

## 背景介绍：

狼人杀是风靡于大学生群体中的一项多人参与的智力游戏，游戏中需要一名法官主持并裁判游戏。为了使用程序自动实现法官的作用，让用户更方便地进行游戏，我们开发了本程序，这是一个局域网内的狼人杀对战系统，用户可在同一局域网内使用本程序与同伴进行完整的狼人杀游戏。本程序支持狼人、女巫、预言家、守卫、猎人、村民等角色，支持内置默认角色分配方案（5-15人）及用户自定义分配角色方案，支持屠边和屠城模式，支持狼人自爆、警长选举、警徽流转等功能。

## 需求分析：

本程序可分为通信模块和逻辑模块。

通信模块负责不同主机之间的通讯，包括连接的建立、命令的发送、接受和解析。

逻辑模块负责按游戏规则对玩家的输入进行反馈和胜负判定。进一步可细分为角色信息模块和、游戏进程模块和日志记录模块。

## 分工：

通信模块和 GUI：叶梓杰（2016011380）

### 游戏进程模块：

王琛（2016011360）：

process 基类的构建

process 子类：Guarding,Witching,Predicting,Po\_electing

刘应天（2016011358）：

process 子类：Killing,Hunting,Po\_passing

李映辉（2016011372）：

process 子类：Chat

曾军（2016011359）：

Process 子类：Calculating

ProcessManager 类

### 角色信息模块：

王琛（2016011360）：Character 类

曾军（2016011359）：CharacterFac 类

日志记录模块：刘应天（2016011358）

## 框架设计：

整体框架设计采用通信模块和逻辑模块解耦的方案，即通信模块向逻辑模块提供预先设定好的若干接口，逻辑模块通过这些接口与主机交互。在开发的过程中，我们为了更好的稳定性，曾将通信模块完全更换，逻辑模块仅仅修改了数行便成功复用，这足以证明我们的设计在通信和逻辑模块的解耦合方面是较为成功的。

通信模块主要分为连接的建立、命令的发送、接收和执行。

### 连接的建立：

连接的建立方式有两种方式：通过广播信息获取服务信息并连接或者通过 IP 地址直接连接。TCP 连接建立之后，生成一个对应的 Client 类，通信模块通过 Client 类向逻辑模块提供通信接口。

### 命令的发送、接受和执行：

命令的发送和执行通过 TCPSocket 即可方便实现。但在执行时，TCP 连接的粘包问题会带来命令解析的困难。为此，我们定义了指令之间的分割标示，并用一个类维护已经收到的命令，发现完整命令时执行。

### 游戏进程模块：

游戏进程模块采用了调用环的架构，这是一种变形的责任链模式，一个回合中所涉及的进程类对象在游戏开始时被创建并构建成一个链表环，在游戏进行时按照顺序依次调用。这样的设计便于拓展程序功能，如欲增加功能，只需在调用环的构建中增加新的进程类即可。

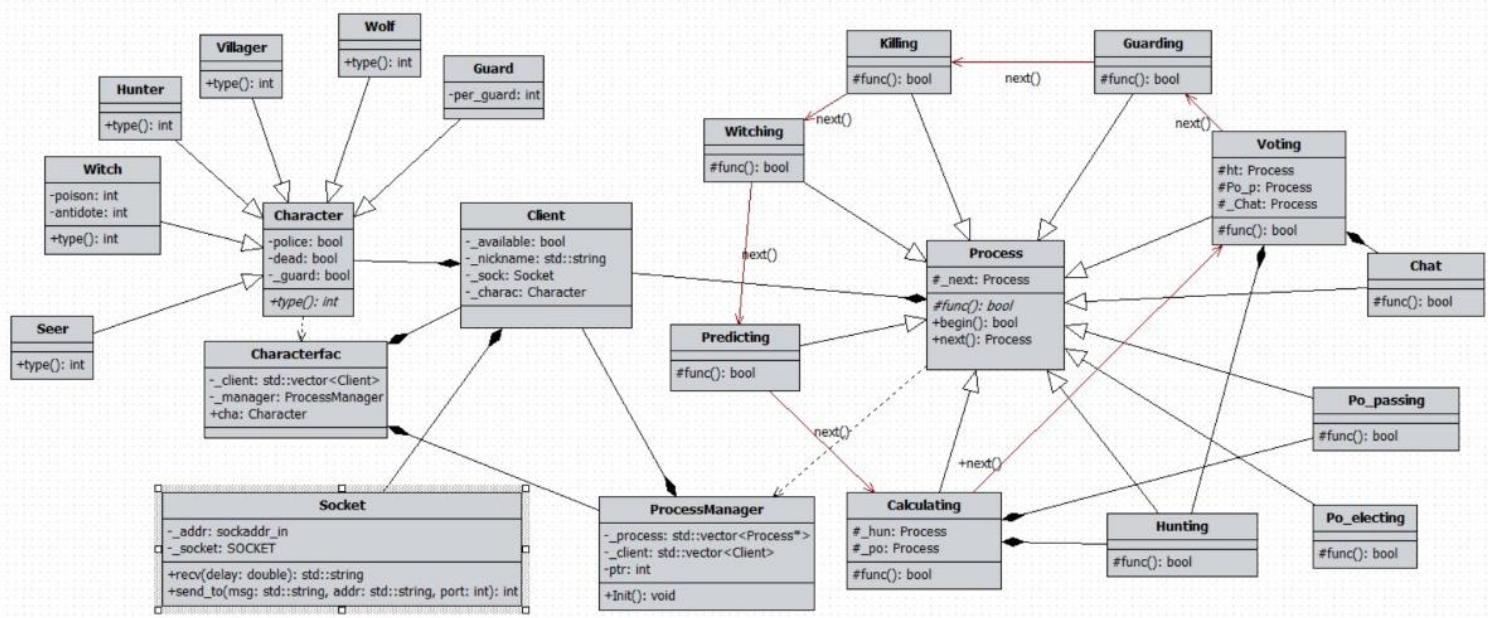
### 角色信息模块：

在游戏开始时根据内置分配方案或用户输入随机分配玩家角色，生成相应的 Character 类并与 Client 类绑定。在游戏进程中，角色信息类为进程类提供更改和查询玩家角色状态的接口。

### 日志记录模块：

游戏过程中的判断有时不仅需要知晓角色的状态，还需要知晓各角色曾经发出的动作和其时间顺序。为此，我们设计了日志记录模块，在玩家发出动作时进行记录，并向进程类提供查询接口。

UML 类图：



## 技术细节：

使用的开源资源：qt

在实现调用环的时候，考虑到我们的程序不同于传统的责任链模式，递归的深度将持续增加直至游戏结束，若采用逐层递归调用的方式，可能会造成栈溢出，于是我们设计了 ProcessManager 类作为 Process 类的调用者。每个 Process 对象的调用结束后，ProcessManager 将会通过函数的返回值决定游戏结束或者继续调用下一个 Process 对象。

## 总结：

由于队内成员均是编程新手，以前均无算法竞赛经历或工程经验，刚开始时我们几乎是一脸茫然。然而经过大家的商讨和共同努力，我们最终完成了这项任务。在这个过程中，我们也体会到了 OOP 思想的强大，体会到一个好的架构设计可以给开发带来很多的方便。同时，我们也感受到了利用开源资源的重要性。也许我们做得不是最好的，但我们的的确已经尽了我们全部的努力去尝试做好这件事，也感受到了亲自完成一个小工程的成就感和团队协作的强大，我们每个队员在这个过程中都有自己的收获。