**<项目名称>**

**软件架构设计**

**版本 <1.0>**

**修订历史记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **修订说明** | **作者** |
| <2018-4-26> | <1.0> | <第一次版本更新> | <kiwi和她的5个小矮人全体组员> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[**1引言**... 1](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817922)

[1.1编写目的... 1](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817923)

[1.2适用范围... 1](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817924)

[1.3定义... 1](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817925)

[1.4参考资料... 1](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817926)

[1.5概述... 1](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817927)

[**2 目前软件系统体系架构**.. 1](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817928)

[**3 软件系统架构设计目标**.. 2](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817929)

[**4 建议的软件系统架构**... 2](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817930)

[4.1 概述... 2](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817931)

[4.2 用例视图... 2](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817932)

[4.3 系统逻辑视图... 2](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817933)

[4.4系统运行视图.. 3](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817934)

[4.5系统实现视图.. 38](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817935)

[4.6系统物理视图.. 3](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817936)

[4.7 边界条件设计... 3](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817937)

[4.8数据管理设计.. 3](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817938)

[4.9 其他设计... 3](https://docs.google.com/document/d/15C3GCA_qKrFPfNLFOTQsi12_t9W7xCMxjTqCwfreu1s/edit#_Toc448817939)

**软件需求规约**

# **1引言**

## **1.1编写目的**

本规约旨在通过与客户进行沟通交流，获取客户对于软件的功能需求与非功能需求，基于用例与文字陈述，使得双方对该软件的功能理解达成一致，从而初步构建出软件在应用域的基本框架和运行逻辑。本规约供预期读者，即软件的设计者、开发者以及测试者为实际的开发和测试工作做准备，使得开发过程更有序规范地进行。

## **1.2适用范围**

本联机狼人杀平台是基于Python和javascript开发，使用Python中的MySQL接口作为关联型数据库环境，可通过微信小程序进行游戏。

## **1.3定义**

具体内容参考词汇表

## **1.4参考资料**

a. 《可行性分析报告》（GB8567-88）

b. 《面向对象软件工程——使用UML、模式与Java》（第三版），清华大学出版, 2011

## **1.5概述**

本文件中简述了目前系统的功能和存在的关于运营模式、软硬件限制、游戏体验较差等问题，并由此设计了一个游戏规则相对灵活，游戏方式更加多样的建议系统。在确定大体设计目标后，细化了建议系统的功能需求，将系统划分为实现不同功能的子系统。同时明确了系统的可用性、可靠性、可支持性、性能和接口等非功能需求。再根据这些需求，设计了六个场景，藉由场景规划了用例模型。同时根据界面友好度和接口等非功能需求确定了界面的demo。

# **2 目前软件系统体系架构**

目前能够在微信中使用的系统多采用客户机/服务器体系结构风格。用户使用该系统作为线下狼人杀游戏的辅助，通过该系统分配角色给用户并引导用户进行游戏。

该系统目前存在的问题在于角色组合类型不够多样，开始游戏的条件比较苛刻。新系统希望在这两点上进行改进。新系统能够增加游戏内角色组合类型的多样性，并且将线下的狼人杀变为线上，即用户能够在网上匹配其他用户，而不必线下凑齐足够的用户。

# **3 软件系统架构设计目标**

保密性：本软件需要对用户的隐私和个人信息作一定的保护，避免信息的泄露。

可修改性：本软件应能较快地更新版本、增加新的版子。

性能：应能支持用户流畅地进行游戏，语音传输的延迟应不超过1s

高可用性：软件应支持高并发的用户游戏进程

开发工具：微信小程序

时间表：2018年5月22日前完成初版。

# **4 建议的软件系统架构**

## **4.1 概述**

本系统采用了三层体系架构风格，实现接口层和应用程序的逻辑层的分离，并且将接口层与应用程序，以及应用程序与存储系统的交互分离，使系统具有较高的鲁棒性。

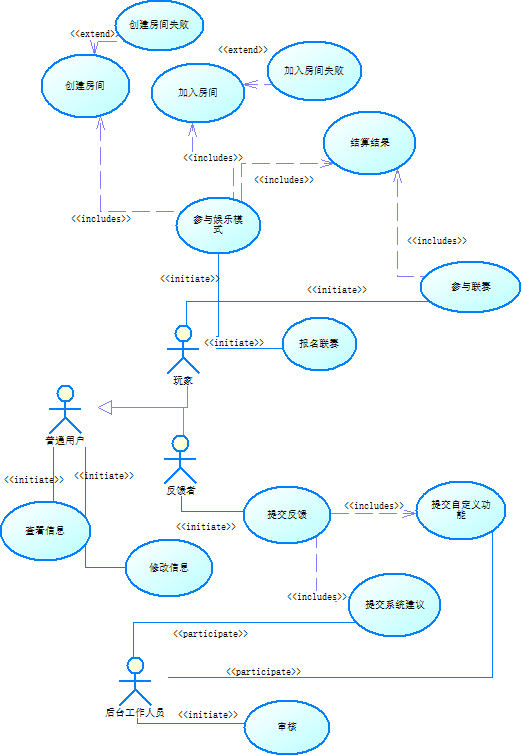
软件将使用腾讯云提供的视频直播API以实现语音交互。

框架：koa+knex+wafer2

接口层包含OnlineKillClient子系统，负责与用户的交互，并将用户的输入传输进OnlineKillServer。

应用层包含多个子系统：OnlineKillServer将用户的输入信息传输给下层server，并根据server的实时反馈与控制，修改用户的界面。OnlineKillServer子系统接受client的信息，将信息分派给不同的下级子系统进行处理，并且将下级子系统的决策提交给client进行界面修改。UserManagement子系统进行用户的注册、登出、用户信息的修改操作。 TournamentManagement子功能使用户能够参与联赛。用户进入后将自动被匹配，进入一个房间并开始游戏，游戏结束由进入结算系统，并且给出反馈。AdviseManagement子系统使用户能够提交系统反馈与建议，使管理者审核建议并提交反馈，如果用户建议的版子通过审核，管理者可以考虑上线新的版子。

## **4.2 用例视图**



|  |  |
| --- | --- |
| 用例名称 | 创建房间 |
| 参与者 | 由玩家启动 |
| 事件流 | 1.玩家选择游戏版子；  2.RoomManagement确定服务器未超负荷，则创建房间，并通知玩家房间创建成功。如果创建失败，则发生创建房间失败用例；  3. 玩家进入自己创建的房间； |
| 入口条件 | 玩家点击创建房间按钮。 |
| 出口条件 | 玩家开始游戏。 |
| 质量需求 | * RoomManagement要在10秒内响应 |

|  |  |
| --- | --- |
| 用例名称 | 参与联赛 |
| 参与者 | 由玩家启动 |
| 事件流 | 1.LeaueManagement通过申请，玩家进入房间。  2.玩家点击确认，开始游戏，待其他玩家确认开始游戏后，游戏开始。  3.游戏结束，LeagueManagement公布各玩家身份及战果，并将结果提交给EvaluationManagement  4.EvaluationManagement对数据库更新玩家段位，并将此更新通知给LeagueManagement |
| 结算结果（使用包含关系）  1.在场上仅剩一个阵营的玩家时，EvaluationManagement开始结算结果。  2.EvaluationManagement宣布游戏结束，判断剩余玩家所属阵营获得胜利，并公布各玩家所属角色。 |
| 入口条件 | 玩家在联赛时间内点击参与联赛按钮，向LeagueManagement发出申请。 |
| 出口条件 | LeagueManagement将段位更新提示显示给玩家 |
| 质量需求 | LeagueManagement能够正确分配场次。 |

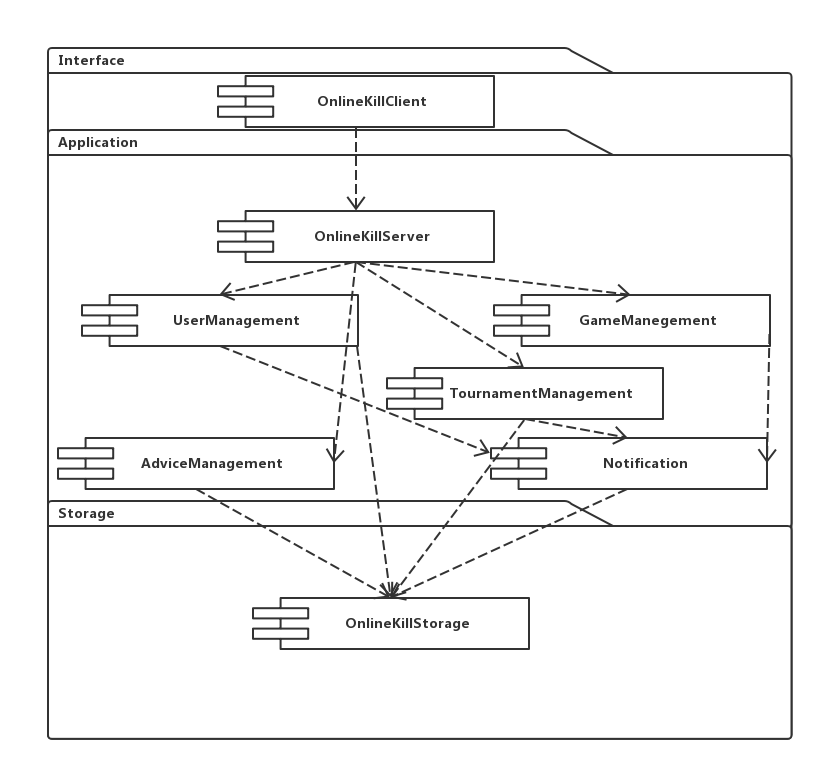
|  |  |
| --- | --- |
| 用例名称 | 提交自定义功能 |
| 参与者 | 由玩家启动 |
| 事件流 | 1.反馈者选择提交自定义功能。  2.反馈者得到一张表单，反馈者在表单上填写建议实现的功能。  3.反馈者将表单提交，并启用审核用例。 |
| 入口条件 | 反馈者进入反馈界面。 |
| 出口条件 | 反馈者提交表单。 |
| 质量需求 |  |

|  |  |
| --- | --- |
| 用例名称 | 审核 |
| 参与者 | 由后台工作人员启动  反馈者参与 |
| 事件流 | 1. 后台工作人员选择一条反馈进行处理。  2. 如果该反馈是玩家对系统的建议或BUG反馈，后台工作人员调查问题并给出解决方案，尽可能解决问题，并及时回复反馈者。  3.如果该反馈是玩家提出的版子建议，则由后台工作人员与公司讨论商议是否上线该版子，并给予反馈，如若该版子被采纳，反馈者将通过RewardManagement获得奖励。 |
| 入口条件 | 后台工作人员收到反馈者的反馈 |
| 出口条件 | 反馈者收到后台工作人员的回复。 |
| 质量需求 | 玩家个人信息更新在2s内完成。 |

## **4.3 系统逻辑视图**

（1） 系统架构

在游戏组织部分，我们选择了三层体系架构风格，一个狼人用户户子系统OnlineKillClient为用户提供一个前端来启动所有与组织有关的用例（例如，选手注册RegisterPlayer、申请ApplyFor）。狼人杀子系统OnlineKillServer负责访问控制、并发控制和为应用逻辑来代理嵌入的子系统。最底层是系统存储子系统OnlineKillStorage来实现的，负责存储持久性数据。

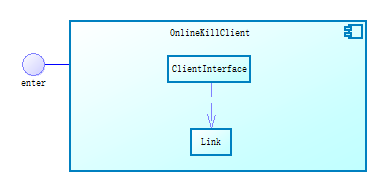


（2） 子系统

1. OnlineKillClient子系统

功能：将用户的输入信息传输给下层server，并根据server的实时反馈与控制 修改用户的界面。

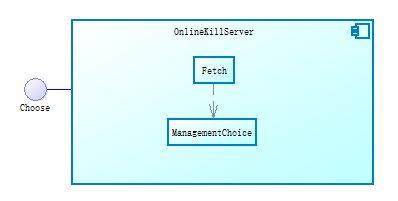
服务：该子系统提供了与用户交互的服务。



2. OnlineKillServer子系统

功能：该系统接受client的信息，将信息分派给不同的下级子系统进行处理，并且将下级子系统的决策提交给client进行界面修改。

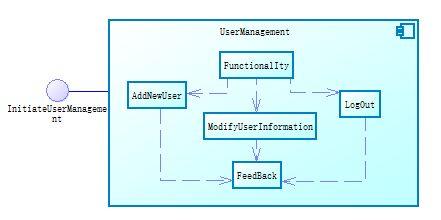
服务：该子系统分发任务给下级子系统。



3. UserManagement子系统

功能：进行用户的注册、登出、用户信息的修改操作。

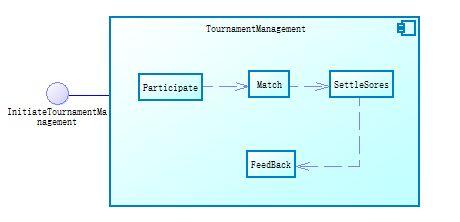
服务：使用户能查看其个人信息，使管理者能够管理用户信息。



4. TournamentManagement

功能：使用户能够参与联赛。用户进入后将自动被匹配，进入一个房间并开始游戏，游戏结束由进入结算系统，并且给出反馈。

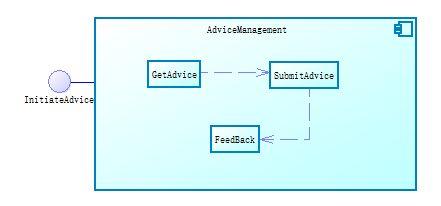
服务： 给server提交Tournament相关信息和结果。



5. AdviseManagement子系统

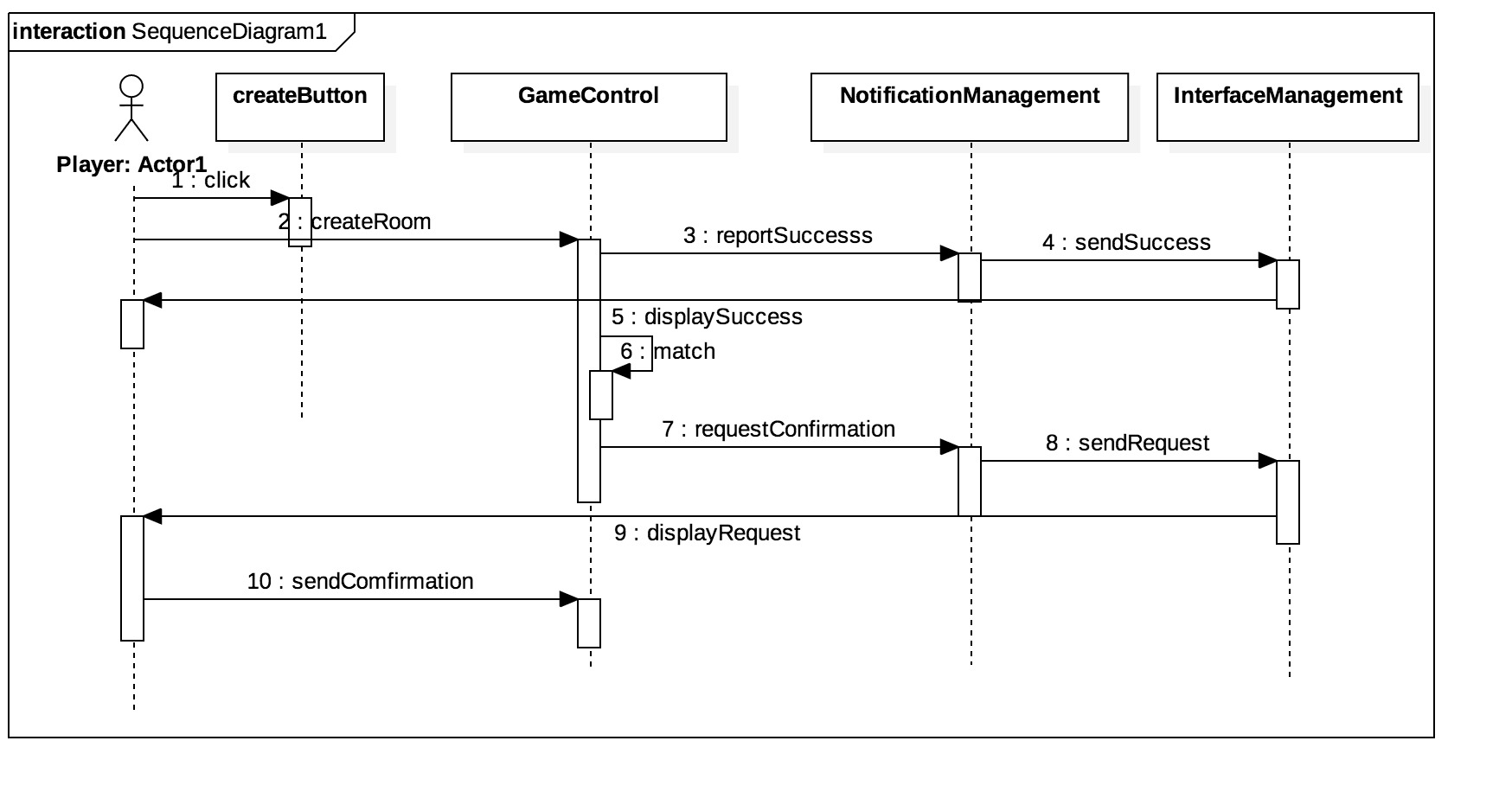
功能：使用户能够提交系统反馈与建议，使管理者审核建议并提交反馈，如果用户建议的版子通过审核，管理者可以考虑上线新的版子。

服务：提交用户的建议给server，使管理者可以管理建议。

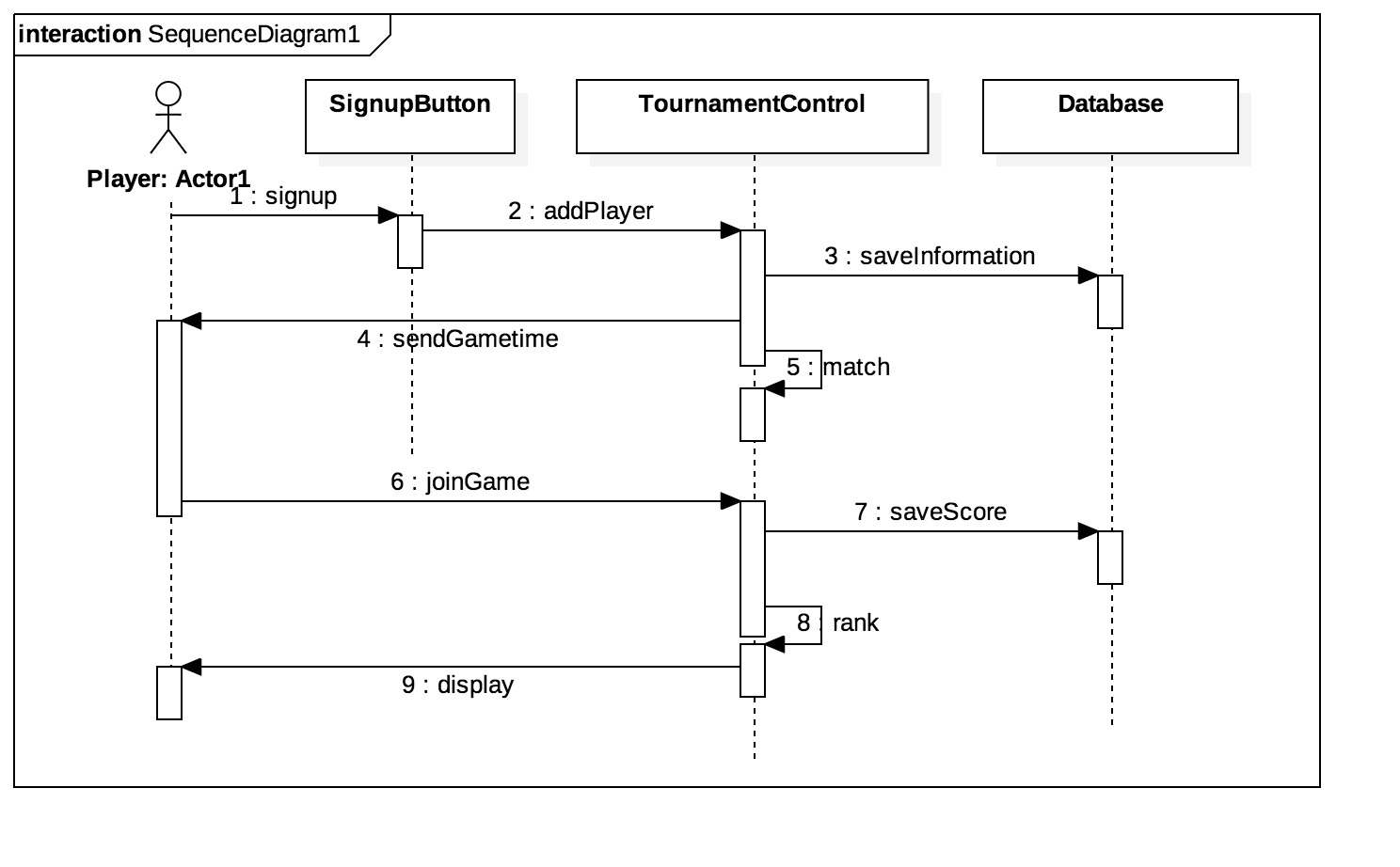


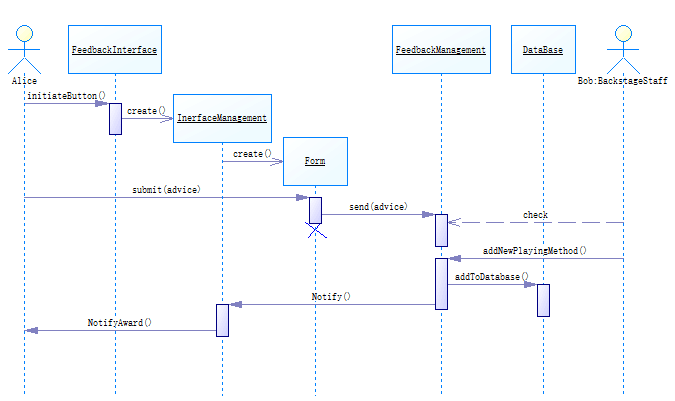
（3）用例实现

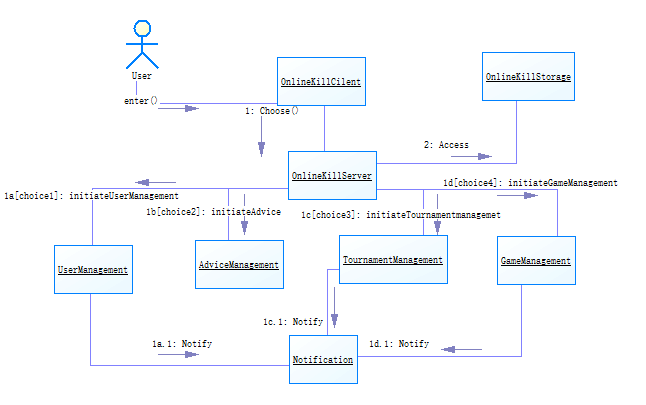
1. 创建房间



2.参与联赛



3. 提交反馈 



*交互图*

## **4.4系统运行视图**

本系统在房间内采用事件驱动，在房间外采用多线程。

## **4.5系统实现视图**

系统开发模型

开发环境：微信web开发者工具, koa, knex, wafer2

开发语言： Python, javascript

版本控制工具：git

目录：

├─client # 小程序端代码

│ ├─imgs # 图片存放处

│ ├─pages # 各个页面代码的存放处

│ │─utils # 工具包

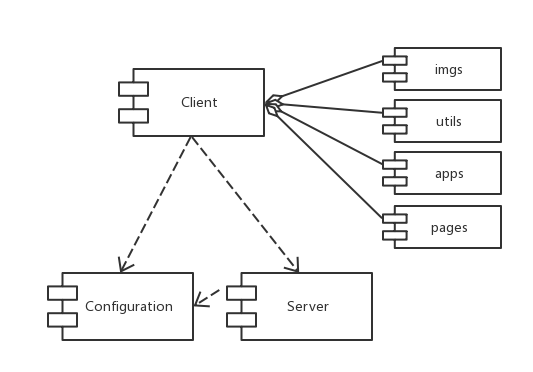
│ ├─app.js # 全局JS

│ ├─app.json # 全局配置

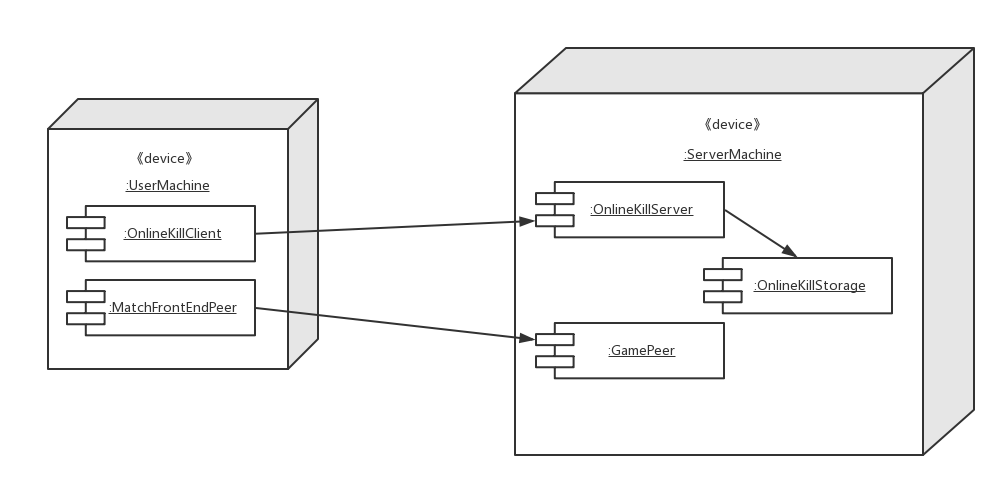
│ ├─app.wxss # 全局样式

├─server # 服务器端代码

└─project.config.json # 项目配置



## **4.6系统物理视图**



## **4.7 边界条件设计**

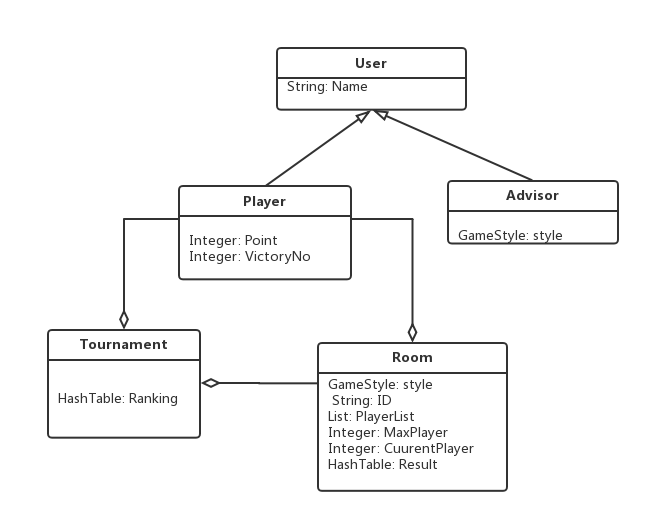
|  |  |
| --- | --- |
| InitiateOnlineKill | OnlineKillOperator 创建OnlineKill, 给出其名称，选择持久性存储子系统（要么是平面文件，要么是数据库文件），配置资源参数。 |
| ManageGames | OnlineKillOperator 创建或删除一个Game，包括为GamePeer 和 MatchFrontEndPeer 定制代码。 |
| Convert Persistent Storage | 当OnlineKillServer关闭时，OnlineKill能够进行持久性存储的转换，将其从平面文件存储为数据库存储或从数据库存储变为平面文件存储。 |
| StartOnlineKillServer | OnlineKillOperator启动OnlineKillServer。一旦服务器的启动完成， Players和Advisors就可以启动它们的任何用例了。 |
| ShutDownOnlineKillServer | OnlineKillOperator停止OnlineKillServer。服务器终止任何正在进行的Matches并存储任何高速缓冲的数据。MatchFrontEndPeers和GamePeers被关闭。一旦这个用例结束， Players，Advisors就不能或修改OnlineKill了。 |

描述了系统中如何启动、关闭以及进行错误处理。描述的方式是采用用例的方式来进行说明，即提供启动用例、关闭用例、错误处理用例的分别说明，并通过交互图来说明这些用例是如何实现的。

## **4.8数据管理设计**

4.8.1 持久化数据

1. 用户User
2. 玩家Player
3. 建议者Advisor
4. 联赛Tournament
5. 房间Room



4.8.2 保存方式

所有的数据都选用关联数据库的方式进行保存，因为更加方便查询和保存每次数据的变动和查询记录，方便例如连接中断的异常发生时，将系统恢复至异常发生前的状态。

4.8.3数据库的表设计

4.8.3.1 用户

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | 类型 | 范围 | 允许空值 | 主键 | 说明 |
| UserName | String | 20 | Not NULL | 是 | 账号名 |

4.8.3.2 玩家

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | 类型 | 范围 | 默认 | 允许空值 | 主键 | 说明 |
| UserName | String | 20 |  | Not NULL | 是 | 账号名 |
| Point | Integer | [0,2^32] | 0 | Not NULL |  | 账号积分 |
| VictoryNo | Integer | [-2^32,2^32] | 0 | Not NULL |  | 胜负数 |

4.8.3.3 建议者

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | 类型 | 范围 | 默认 | 允许空值 | 主键 | 说明 |
| UserName | String | 20 |  | Not NULL | 是 | 账号名 |
| Rule | GameRule | DEFAULT | DEFAULT | NULL |  | 游戏规则 |

4.8.3.4 房间

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | 类型 | 范围 | 默认 | 允许空值 | 主键 | 说明 |
| RoomID | String | 20 |  | Not NULL | 是 | 房间号 |
| PlayerL | List |  | [] | NULL |  | 玩家列表 |
| MaxPlayer | Integer | 定值 | 15 | Not NULL |  | 最大人数 |
| CurrentPlayer | Integer | [0,15] | 0 | Not NULL |  | 当前人数 |
| Result | HashTable |  | 0 | Not NULL |  | 游戏结果 |

## **4.9 其他设计**

访问控制

1. 访问矩阵

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Object Actors | OnlineKill | User | Tournament | Room |
| OnlineKillOperator | <<create>>  创建用户 | <<create>>  撤销用户 | <<create>>  存档 | <<create>> |
| Player |  | 设置信息 | 查看排行 | 进入房间 |
| Advisor | 提交组合 |  |  |  |
| Room |  | 更新信息 | 更新信息 |  |

1. 用户认证方式

我们采用组合认证的形式来确保账号的安全性，首先由于是微信小程序，所以我们能够通过用户授权来获取用户的个人信息，同时向用户的手机发送验证码来确认用户登陆。