《Greed 游戏系统的设计与实现》

小组成员： 刘含宇（2016211390）

张绍磊（2016211392）

关磊 （2016211400）

张明睿（2016211364）

班 级： 2016211310

课程名称： 面向对象程序设计实践Java

指导老师： 房鸣

# 一、Greed游戏概述

本游戏使用6个立方体骰子，可供两个以上游戏者集体游玩。骰子的六面分别标有阿拉伯数字1~6的点数。游戏时若干个骰子同时集中掷下，依据所得到点数组合计算得分。

**游戏计分规则如下：**

·获得一个1点得100分；

· 获得一个5点得50分；

· 获得三个1点得1000分；

· 获得六个1点得3000分；

· 获得三个相同点（三个一点除外）将此单独点数乘以100后计算得分；（例：三个2点得分为2×100=200分）。

**游戏控制规则如下：**

· 游戏者依次获得一轮游戏机会，得分由各轮游戏的得分累计获得。

· 每一轮开始时有六个骰子可以投掷。每掷一把后计算得分并将得分的骰子取出，再汇集其余的骰子掷下一把。逐次掷下去直至掷完骰子或主动放弃再掷。

· 游戏开始时，掷一把得到300分或以上分既可入局（已得积分带入）。否则将失去本轮游戏资格须待下一轮机会。

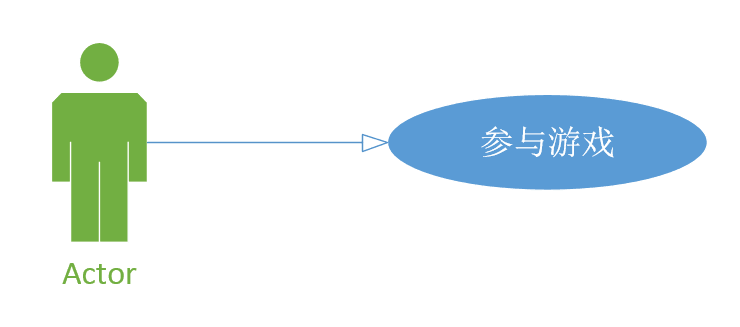
· 在入局后的每轮投掷中，若有一把未能得分，则本轮已得的全部积分作废且失去本轮游戏资格须待下一轮机会。

· 首先获得3000分者为赢家。

# 二、用例模型

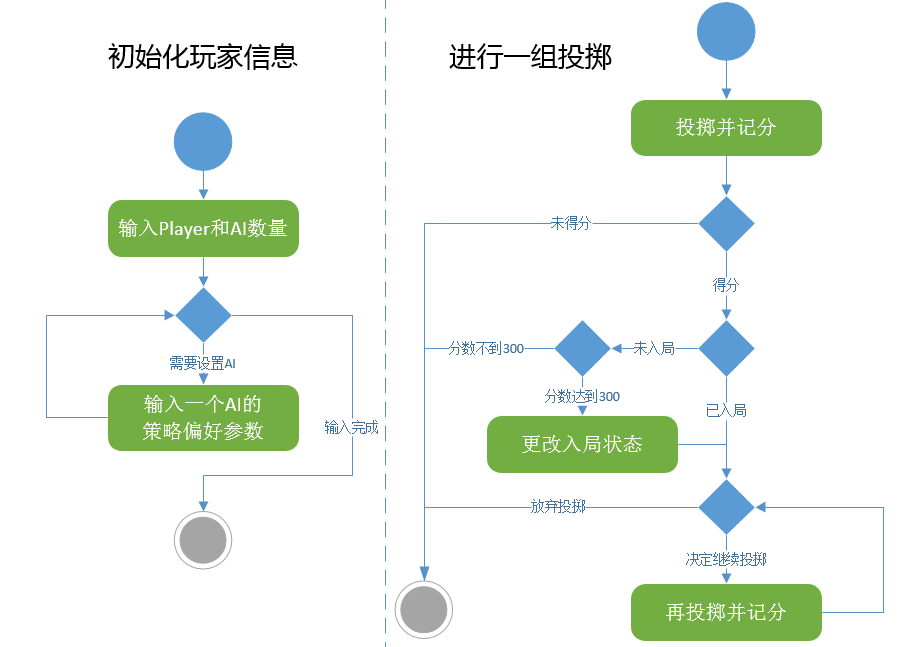
## 1. 用例图

由于这个游戏规则非常简单，只有一个参与游戏的玩家角色，并不能像上课时教务选课的例子分成多个角色；同时因为类之间耦合关系非常紧密，无法像选课过程一样拆成多个用例，所以我们最后设计的用例模型为一个Actor和一个用例。



## 2. 活动图

活动图体现了系统的总体流程，根据之前的系统分析，系统的活动图如下图所示：



## 3. 问题陈述

设计并实现一个Greed游戏。本游戏使用六个立方体骰子，可供两名及以上玩家和一名裁判共同游玩。骰子的六面分别标有1~6的点数。Greed游戏的计分方法已在游戏概述中写明。

游戏开始前，用户向裁判发出申请，添加玩家角色或者电脑控制角色并为电脑控制角色选择策略偏好。策略偏好分为保守型、激进型、智慧型。裁判为所有角色按添加顺序从1开始编号。游戏过程中，所有角色按照编号由小到大的顺序依次获得一轮游戏机会，得分由各轮游戏的得分累计获得。

所有角色最开始得分均为0分，且处于未入局状态。每一轮开始时，每个角色有六个骰子可以投掷。角色掷骰子，裁判根据投掷结果计算本轮本把得分。

若角色处于未入局状态，并且本轮第一把投掷所得的分数小于300分，则当前角色的这一轮投掷结束，并且当轮得分清零，仍处于局外。若本轮第一把投掷所得的分数大于等于300分，则该角色携带第一把得分入局。

若角色处于已入局状态，裁判询问角色是否继续投掷，将得分的骰子取出，再汇集其余的骰子掷下一把。逐次掷下去直至掷完骰子或主动放弃再掷。投掷过程中，角色若有一把未能得分，则本轮已得的全部积分作废且失去本轮游戏资格。

角色主动放弃或者失去本轮游戏资格后，裁判计算出该角色本轮总得分，加入该角色总分。并将盛放有六个骰子的杯子按顺序转交给下一位角色。下一角色开始投掷。

游戏过程中任意时刻，若一位角色总得分大于等于3000分，则该角色获得胜利。

## 4. 术语定义

**玩家角色：**

参与游戏的实体，自然人。可以在一台机器上分别控制角色，进行游戏。

**电脑控制角色：**

参与游戏的AI，由电脑控制。可以在一台机器上根据策略偏好控制抉择，进行游戏。策略偏好分为激进型、保守型、智慧型。激进型勇于尝试丢失分数的风险，相比基准更大概率选择继续投掷；保守型不敢冒着丢失分数的风险继续投掷，相比基准更小概率选择继续投掷；智慧型以期望为基准，根据当前得分判断是否继续投掷。

**骰子：**

一种立方体玩具，其 6 个面上分别刻有 1－6 数值不等的点数，投掷后其朝上面对应的点数即为所得点数，本游戏中骰子数目为 6 个。

**轮：**

即在没有玩家达到获胜条件的情况下，所有玩家都进行一次游戏叫做一轮。本游戏进行的轮数不定，如果在某轮中，一旦有某个玩家达到获胜条件，则整个游戏结束。

**入局：**

游戏开始时，所有玩家处于未入局状态。游戏中，未入局玩家掷一把得到300分或以上分既可入局。否则将失去本轮游戏资格须待下一轮机会。

**裁判：**

即指该游戏的整个控制中心，在现实中，可以为游戏进程控制者，他控制着整个游戏的流程。

**计分板：**

记录着每个玩家的游戏得分。

# 三、类筛选过程

问题陈述反映了该游戏所要求达到的功能，从问题陈述中获得的名词经过筛选可以作为类的参考，从以上的问题陈述中可以获得的名词如下图所示：



优化筛选对象类：

（1）增加系统外的类：

UI类：用于获得用户输入信息，如添加角色、选择是否继续投掷骰子等用户操作，并将游戏中各角色分数、轮数等信息显示到屏幕。

（2）删除类：

Greed游戏：“Greed游戏”是要设计并实现的游戏系统名称，不属于一个单独的类，故删除。

申请：申请属于用户向裁判类发送的信息，不属于类，故删除。

轮类、把类：“轮”和“把”是游戏过程中出现的抽象名词，不能构造相应的类，故删除。

角色：角色属于抽象类，故删除

策略偏好：策略偏好属于AI类的方法。本身是动作，不是单独的类，故删除。

局内、局外：局内、局外是角色的状态，属于角色类的属性。本身是属性，不是单独的类，故删除。

计分方法：计分方法属于骰子类的方法。本身是动作，不是单独的类，故删除。

胜利：胜利属于角色的状态，不属于类，故删除。

（3）合并类

"本轮得分"和"投掷结果"是"骰子"类的属性，都不能自成为一个实

体。所以合并成骰子类。

­经优化筛选对象类如下图所示：



综上分析，依据系统功能需求共创建6个类：骰子类（Dice）、裁判类（Referee）、玩家角色类（Player）、电脑控制角色类（AI）、计分板类（ScoreBoard）、UI界面类（UI）。

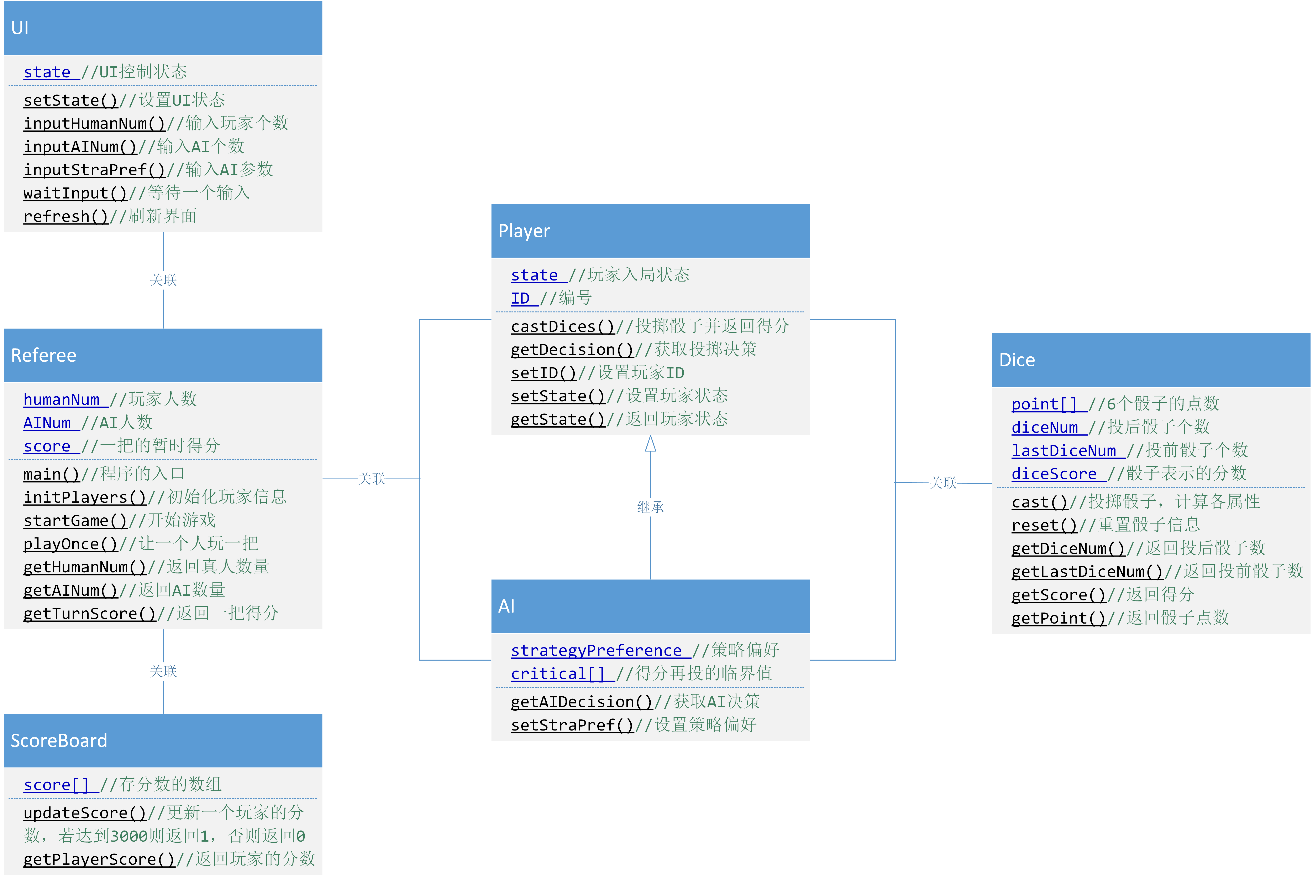
# 四、类的关系模型

类的模型的分析是基于我们的以下认识：所有游戏中真实出现的信息都应该作为各个实体类的属性；所有对于这些属性的操作，包括计算，修改和读取都应该作为相应类中的方法；所有使各个类协同工作的流程控制都应该在控制类Referee中；所有与外界Actor交互的输入输出显示都应该在边界类UI中。

比如投掷骰子这一动作，在真实游戏中是由人做出，但是在面向对象中骰子点数作为骰子类的属性，对于骰子点数的生成和分数计算等作用于这一属性的方法显然应该与属性在同一类中，而不是在玩家类或控制类中。这也是面向对象在思路上更加清晰的体现。

在我们的类关系模型中，玩家类Player与AI是继承派生关系，基类Player中包含真人玩家和AI玩家的基本信息，包括ID，入局状态等，而派生类AI中则包含其独有的策略偏好，策略数据等属性。

我们单独把玩家的总分提取出来作为ScoreBoard类是基于这样的想法：玩家分数是一个属性，对于这个属性的修改操作如果存在于玩家，相当于实际情况中玩家可以自己改变分数，那么直接改成3000就赢了，这是不合理的。所以分数属性和修改的方法应该提取出来，由Referee和ScoreBoard进行分数的控制。



# 五、事件脚本

1、 每次新游戏开始时，输入m、n（0<=m+n<=10）分别表示玩家角色和电脑控制角色的数量，接着输入n个整数，表示n个电脑的策略偏好，其中1表示智慧型、2表示保守型、3表示激进型。UI类将信息传入裁判类。

2、 裁判类根据接收到的信息，初始化玩家和电脑控制角色，并且从1开始依次对ID编号。

3、 用户向发送开始游戏信息。

4、 控制类判断当前角色是玩家角色还是电脑控制角色。若是玩家角色，则跳转至4；若是电脑控制角色，则跳转至8。

4、 控制类通知玩家类开始本轮游戏，玩家通知骰子类投掷骰子，并且返回这一把的得分。

5、 玩家把本吧得分传递给控制类。

6、 UI类获得当前骰子点数信息，并且刷新画布。

7、 若本把得分大于300分或者已入局，则该玩家入局并且控制类向玩家类传递获得决策信息，玩家类向UI类继续传递，并从UI依次返回用户的决策。若选择继续投掷，则开始下一轮投掷。否则裁判向下一个玩家发送开始游戏的消息，回到4。若一把得分为0，则清空本轮得分，裁判向下一个玩家发送开始游戏的消息，回到4。

8、 控制类通知AI类开始本轮游戏，AI通知骰子类投掷骰子，并且返回这一把的得分。

9、 AI把本把得分传递给控制类。

10、 UI类获得当前骰子点数信息，并且刷新画布。

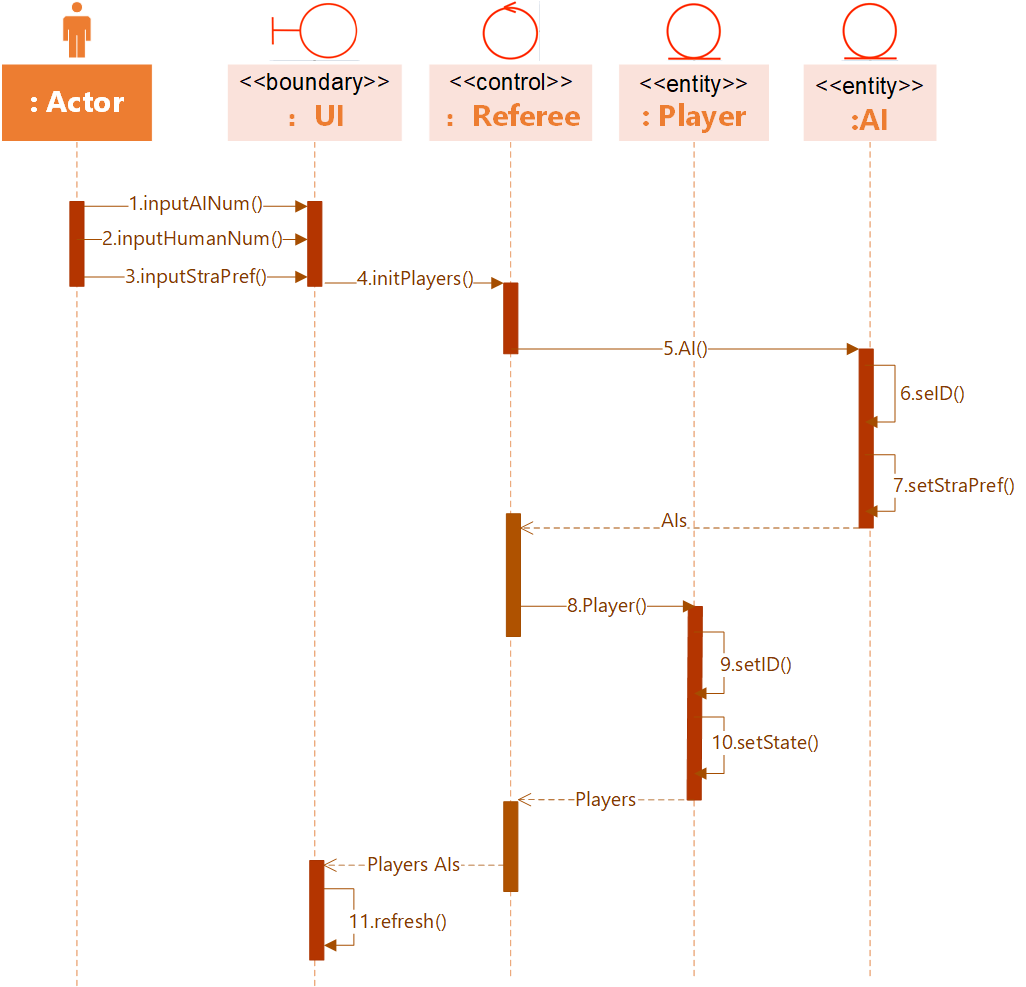
11、 若本把得分大于300分或者已入局，则该AI入局并且控制类向AI类传递获得决策信息，AI通过计算将决策信息返回给控制类。若选择继续投掷，则开始下一轮投掷。否则裁判向下一个玩家发送开始游戏的消息，回到4。若一把得分为0，则清空本轮得分，裁判向下一个玩家发送开始游戏的消息，回到4。

12、 若某玩家总分达到既定分数裁判宣布当前玩家赢，并宣布游戏结束。

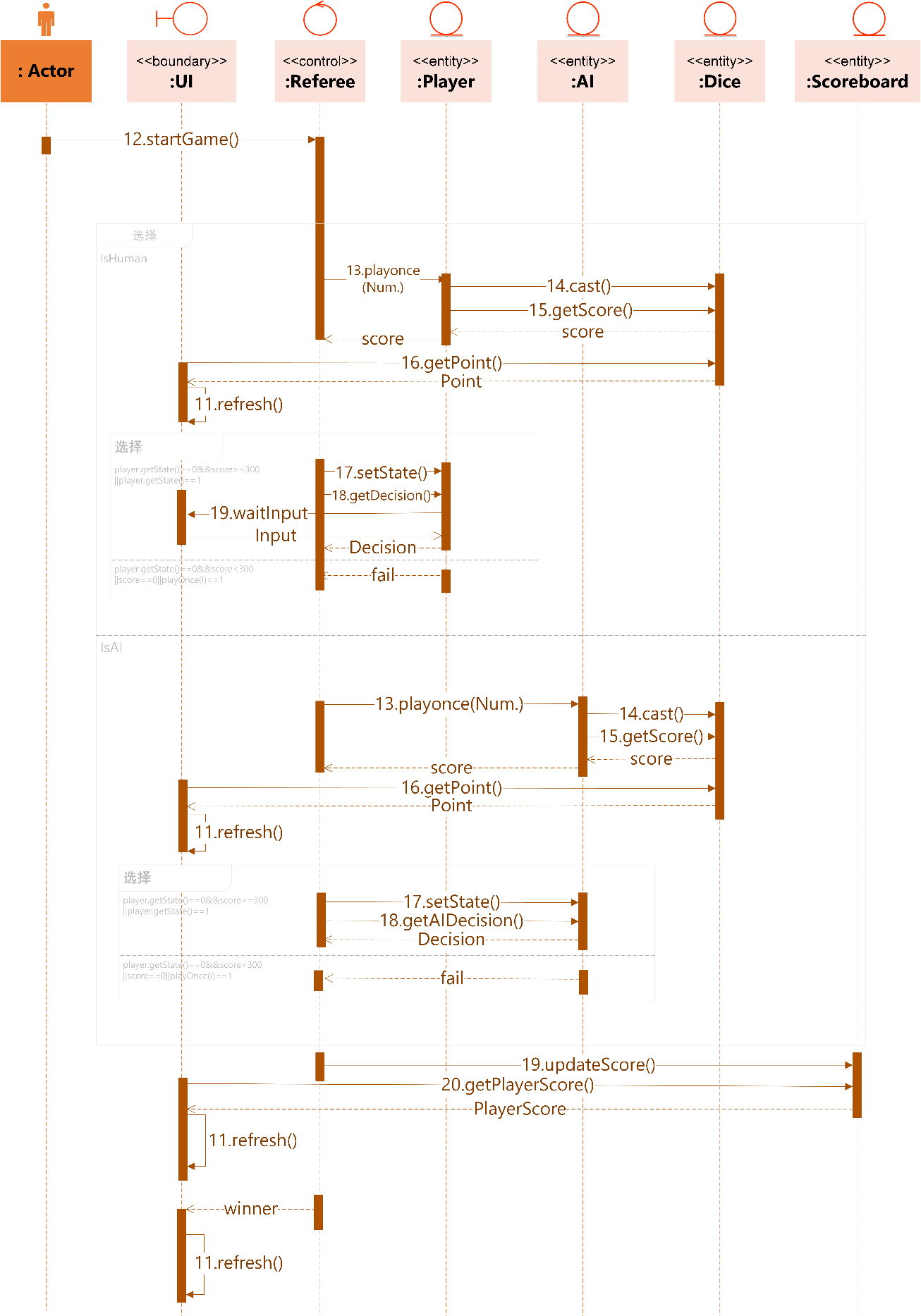
# 六、顺序图

依据事件脚本，设计系统序列图如下所示：

**游戏开始前：**

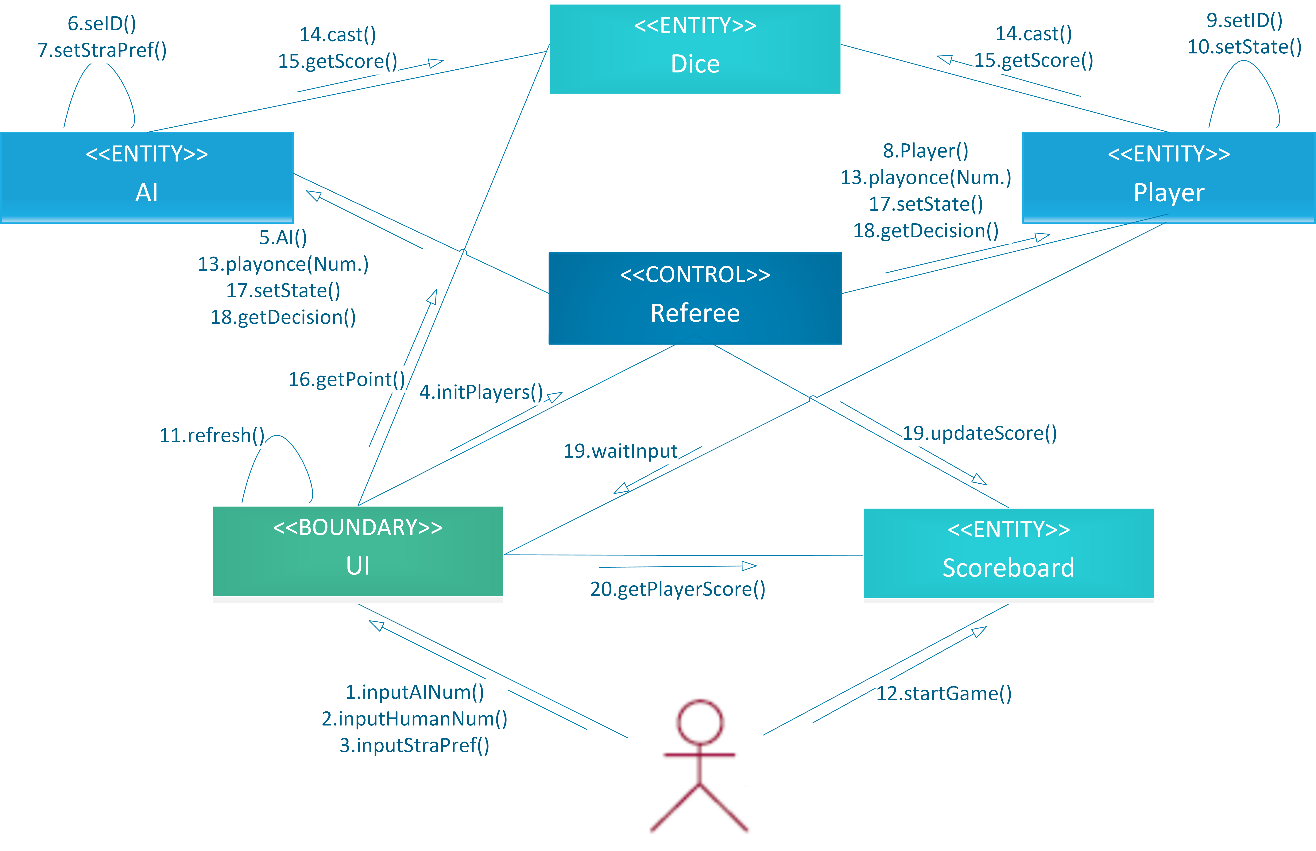


**游戏进行中：**

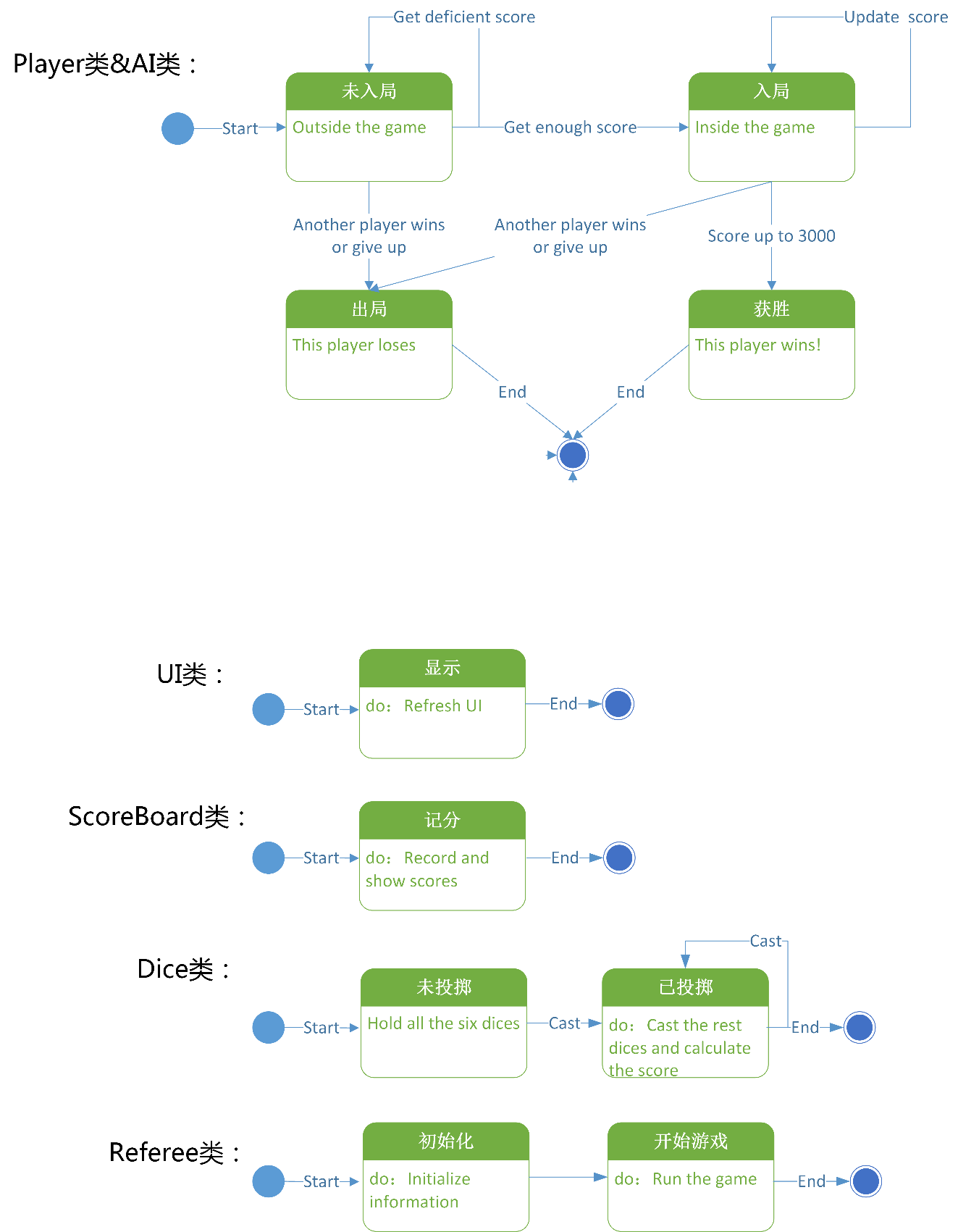


# 七、协作图

根据系统序列图可以得到相应的协作图，如下图所示：



# 八、状态图



# 九、设计亮点

1. 设计了一个最多可容纳10人一起参与的Greed游戏系统，该游戏系统角色由不同数量的真人玩家和由电脑控制的角色组成。

2. 对于电脑玩家，我们为电脑控制的角色赋予了三种策略偏好，当面临不同分数处境时，电脑角色能依靠策略偏好分别做出保守型、激进型和智慧型三类不同类型的决策。

3. 对于真人玩家，我们为玩家提供了随时退出游戏的选项，在玩家有较大分数差距退群之后，游戏可以跳过已经退出的玩家继续正常进行。

4. 在游戏进行中的环节，为了保证游戏的公平性，防止玩家随意篡改个人分数，我们增加了记分板类，将玩家的分数作为记分板属性的一部分。

5. 代码部分结构与设计完全吻合。

6. 清晰的图形化交互界面，操作易用性好。

# 十、代码反馈设计

原始类：



代码反馈之后的类关系：

