华东师范大学软件学院实验报告

实验课程：OOAD	**年级：**2017 级	**实验成绩：**
实验名称:OOAD 分析设计（BlackJack）	**姓名：**彭钧涛、郭源杰、李尚真	
实验编号：No.2	**学号：**10164601140、10164800305、10161900218	**实验日期：**2019 年 10 月 20 日
指导教师：姜宁康	**组号：**4	**实验时间：**4 学时

一、Lab **Objectives 实验目的**

1/ Exercise the design process for a Domain Model 练习概念模型的设计过程

2/ Exercise the design process for a Design Model 练习设计模型的设计过程

2/ Experience OOAD thoughts by implementing own design solution 初步掌握 OOAD 的方法，通过系统的实现，体会 OOAD 的思想

二、**Contents and steps 实验内容与实验步骤**

Steps: Please use tools (such as EA, Rose) to draw diagrams. Each group hand in only one result, including Lab report and source code

1．3 人一个小组，理解系统需求，进行系统分析设计：

1） 给出设计模型，主要是设计类图，对类的主要操作，要给出功能描述

2） 针对 2 个重要的功能，画出顺序图

2．系统实现、测试、演示

3．检查

提交实验报告（电子版 ）、项目源代码、运行时录屏

评分要点：

1/ 在完成基本要求的基础上可以美化 UI（可选）

2/ 网上有现成的源代码：只可参考，绝对不能照抄（检查时会问某段代码、某个变量的功能、用处）

3/ 代码需要有一些注释（as more and clear as possible）

4/ 截至时间：2019.10.31（W09）

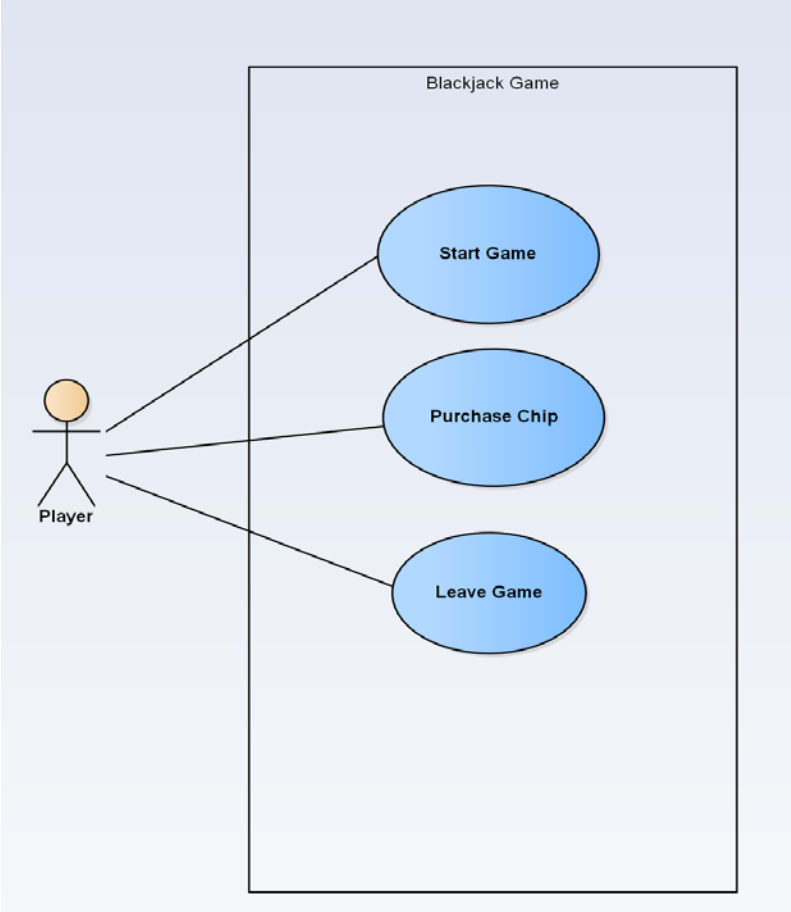
三、Lab**Environment 实验环境**

1/ Computer 计算机一台， 2/ UML tool 工具 3/ Java/C++ development environment 开发环境

四、**Lab processes and analysis 实验过程与分析**

3 人一个小组，理解系统需求，进行系统分析设计：

a) 用例图



b) 用例描述

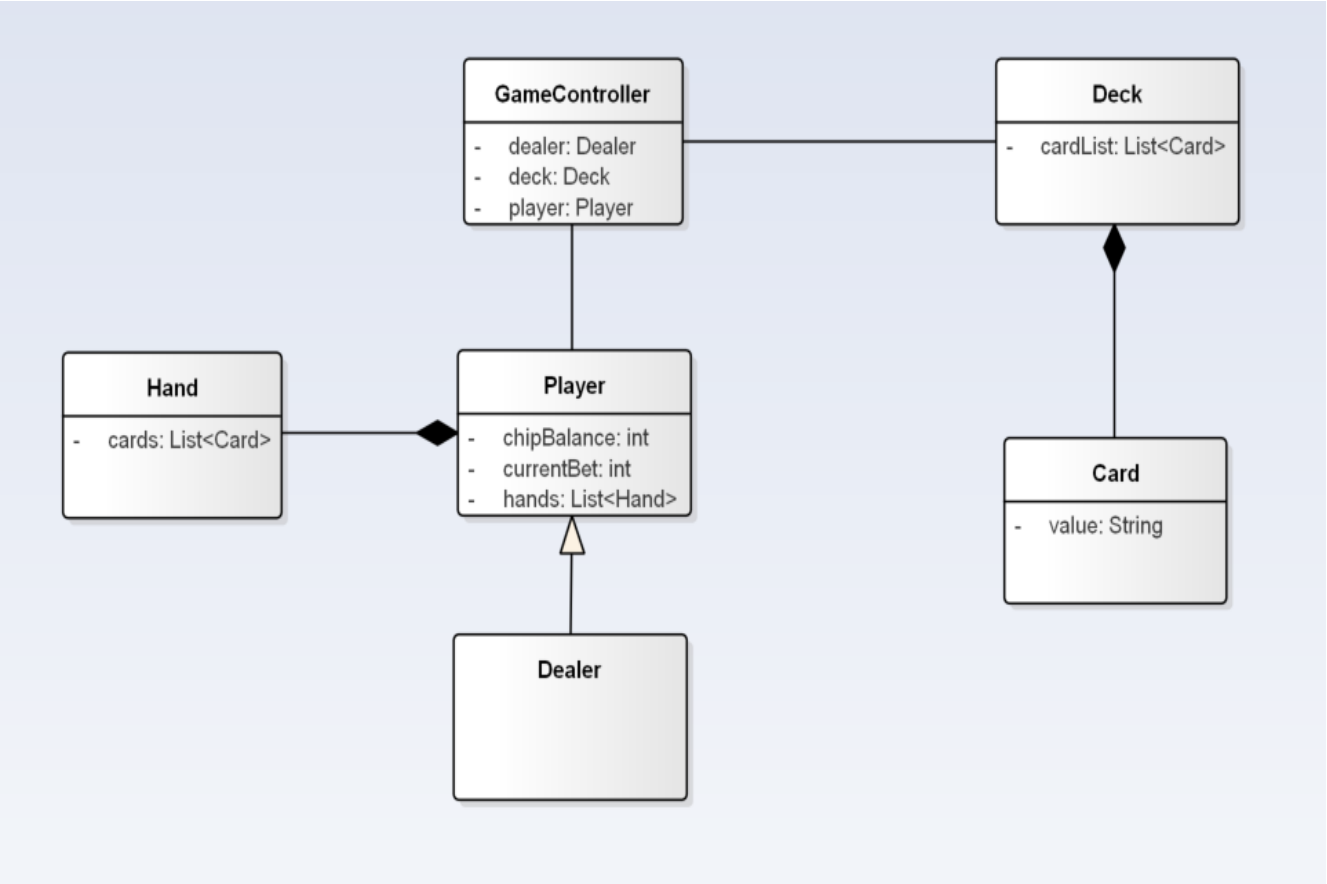
Use Case Name	**Start Game**
Use Case Description	Starts a blackjack game
Primary Actor	Player
Precondition	N/A
Main Incident Flow	1. The use case starts when player indicates the wish to start game. 2. System verifies the player's status 3. Player receives cards 4. System indicates player and dealer's shown card 5. System calculates the card value Repeats step 3-5 until player doesn't want new card. 6. System indicates the round result. 7. Use case ends in success.
Exception Flow	2.A Player doesn't have enough chip balance 2.A.1 System informs player to purchase chip 2.A.2 Use case ends in failure 5.A Player or dealer's card value is

	greater than 21 5.A.1 Use case continues to step 7
Postcondition	N/A

Use Case Name	**Purchase Chip**
Use Case Description	Add chips to player
Primary Actor	Player
Precondition	N/A
Main Incident Flow	1. Use case starts when player indicates the wish to purchase chip 2. System asks the player the amount of chip 3. Player inputs the system the amount of chip 4. System informs the player the result of purchase 5. Use case ends in success
Exception Flow	3.A Illegal input 3.A.1 System informs the input error 3.A.2 Use case ends in failure
Postcondition	Player's chip balance increases

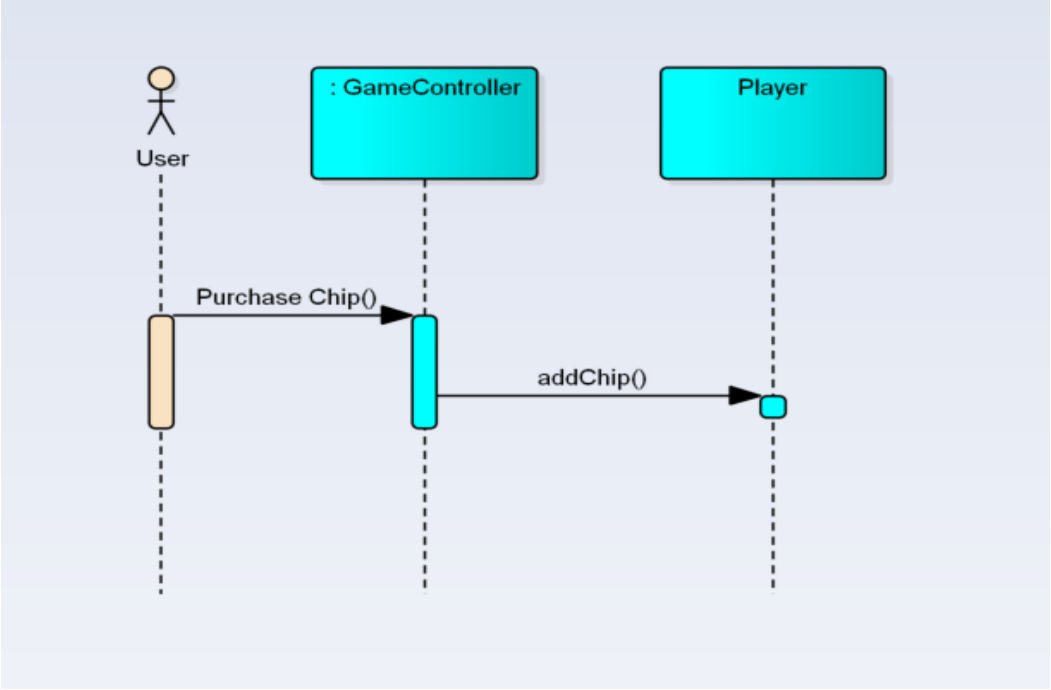
Use Case Name	**Leave Game**
Use Case Description	Leaves and terminates the game
Primary Actor	Player
Precondition	N/A
Main Incident Flow	1. Player indicates the wish to leave. 2. System terminates.
Postcondition	N/A

c) 领域模型



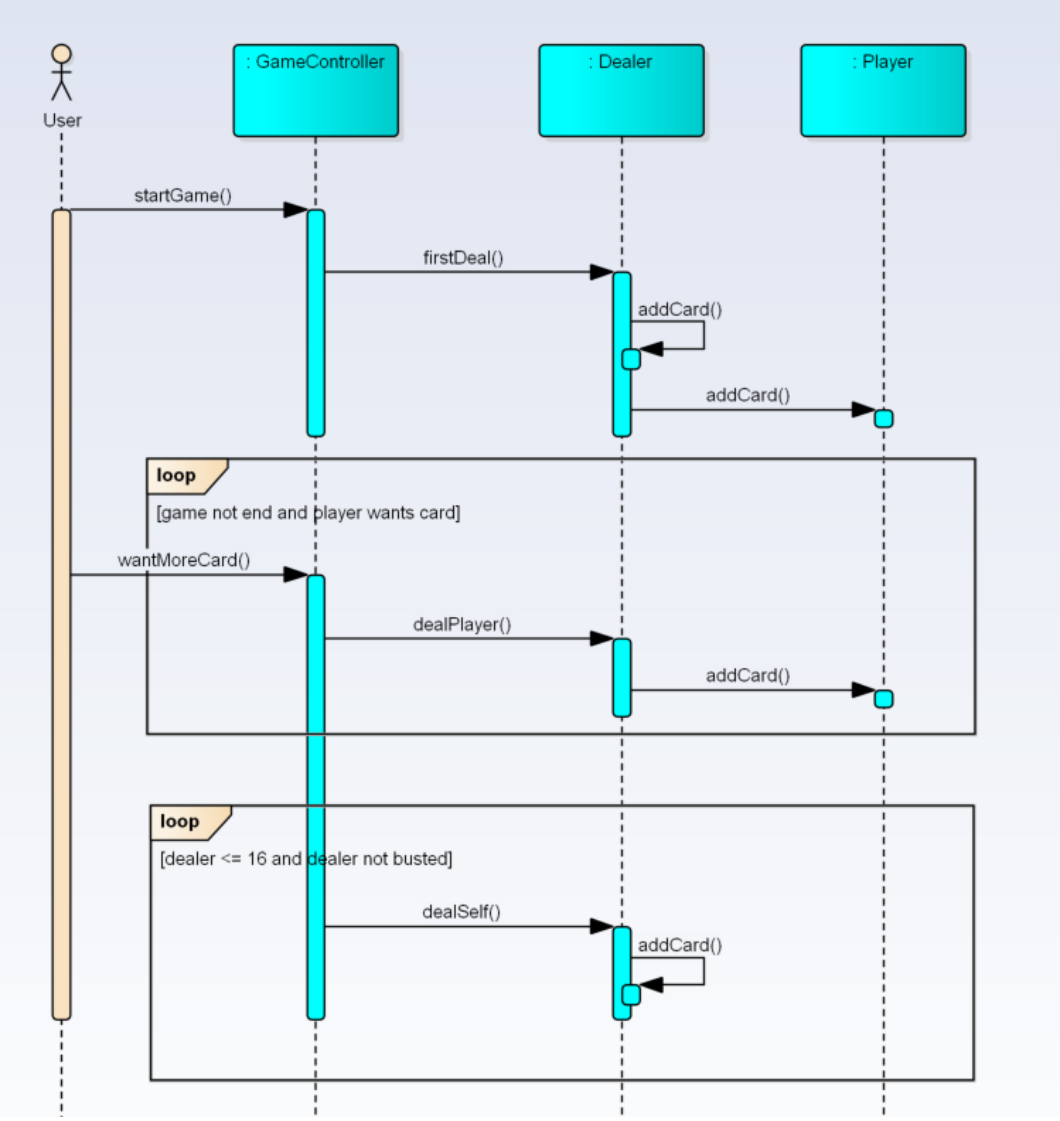
我们将手牌 Hand 抽象为类的主要目的是 Hand 是能够判断扑克面值的最小单位。如果以 Card 为单位判断牌值，则单个的 Card 对象不具备足够的信息来判断自己的面值，例如单张 A 可以判断为 1 或 11，会给高层的游戏逻辑设计增添麻烦。因此将计算手牌总面值的 Responsibility 赋给了 Hand 类。

d) 顺序图 1 Purchase Chip



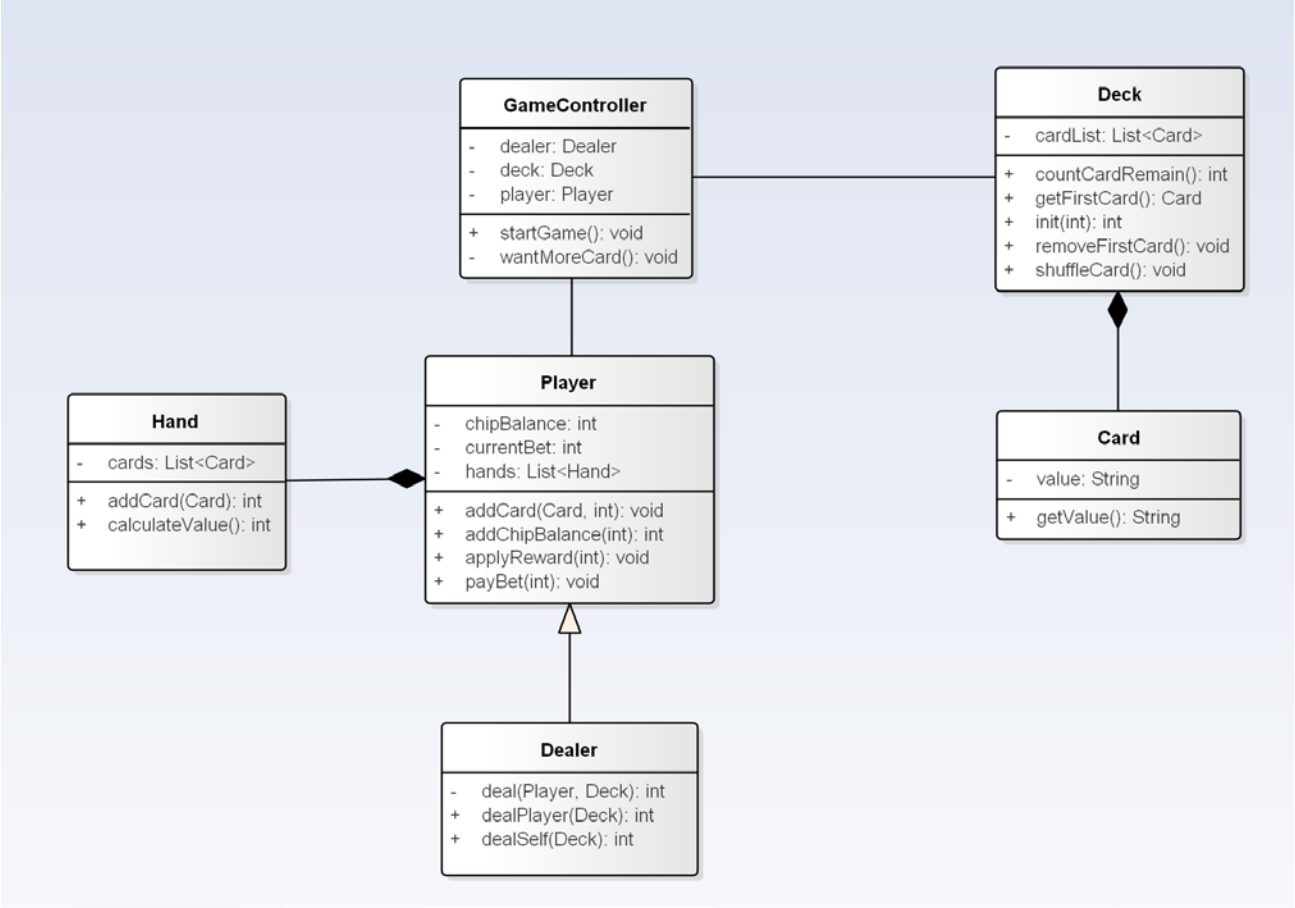
没有筹码则无法参与赌博，所以我们认为充值筹码的用例是必要的，由参与者发送充值消息，然后由游戏控制器给 Player 对象添加一定数量的筹码。

e) 顺序图 2 Start Game



这是主要用例 Start Game 的顺序图，参与者发起开始游戏的消息后分三个片段：第一是初始发牌，一人两张；第二是在游戏没有结束（爆牌）的前提下循环轮询玩家是否要牌，直到玩家不要牌；第三是在玩家不要牌后，庄家给自己发牌，直到大于等于 17 点。

f) 类图



i. GameController 作为总控类，有着开始游戏主循环 startGame()、询问玩家是否要牌 wantMoreCard()两个方法。

ii. Player 玩家类，有筹码余额 chipBalance、当前下注 currentBet、手中手牌列表 hands 的属性。它的方法 addCard()用于添加手牌，addChipBalance()用于充值筹码、payBet()用于下注、applyReward()用于应用每局比赛的输赢情况。

iii. Dealer 庄家类，庄家是一种特殊的玩家，所以继承了 Player 类。庄家又有发牌的职责，所以多了 dealPlayer()给玩家发牌、dealSelf()给自己发牌的方法。

iv. Hand 类主要用于抽象玩家手牌和计算手牌点数，因此有 cards 列表属性，添加卡牌 addCard()、计算手牌值 calculateValue()方法。

v. Deck 作为牌堆，用以适应多副牌的情况，cardList 是整个牌堆的所有手牌的列表，init()用于使用副数初始化牌堆，shuffleCard()用于洗牌，get/removeFirstCard()用于获取牌堆顶部卡牌便于发牌。

vi. Card 是本游戏的基础，有一个字符串属性对应牌面值（不是具体手牌值）及其对应的 getter 方法 getValue()。

五、Summary 实验结果总结

设计中遇到的问题：

1/ 第一次设计好类图后，在编写代码过程中发现实现困难

2/ 对于类应该如何设计，以及应该将各个类抽象到什么程度的问题，小组成员之间在讨论中存在分歧，如果抽象程度太低，将导致开发时重复代码偏多，以及后期难以扩展，如果抽象程度太高，将耗费大量时间和系统计算资源在各个函数之间不停跳转，并且加大开发难

度

3/ 对于卡牌 Card 这个类是否要具有获得其卡价值 value 的功能，最初的设计 Card 具有 key 和 value 两个属性，但是对于特殊卡牌 A，其价值可以为 1 或 11，并且单从功能设计角度考虑，卡牌不具有能动属性。

解决的办法：

1/ 针对开发过程中遇到的问题，对各个类所拥有的方法和属性重新分析，然后再次尝试实现，经过多次重复这个过程，我们成功设计出一个结构比较合理并且方便实现的类图。

2/ 经过商讨，我们选择保持一部分功能的可扩展性，在其它一些功能上放弃可扩展性，例如对于与用户进行交互、输入输出的功能，我们决定直接由 GameController 类负责，不再另外写专门的 UI 类；而对于将来可能产生的一个用户有多副手牌的需求，我们将 Card 数组再抽象一层，成为 Hand 类，让每个 Player 实例持有一个 Hand 列表，负责存储多组手牌（当前程序中固定为 1 副）。

3/ 经过商讨，我们选择将计算卡牌价值的功能交给更高层的类(Dealer,Player)进行，一方面这种功能更符合实际设计，另一方面在一定程度上解决了对于特殊卡牌 A 有两种牌值的情况

六、附录

GitHub 入口: https://github.com/UncooleBen/OOAD-Labs/tree/master/lab02-blackjack/

录屏 screen record.mp4

代码 见压缩包