**C:\Users\sse\Desktop\logo_01.png**

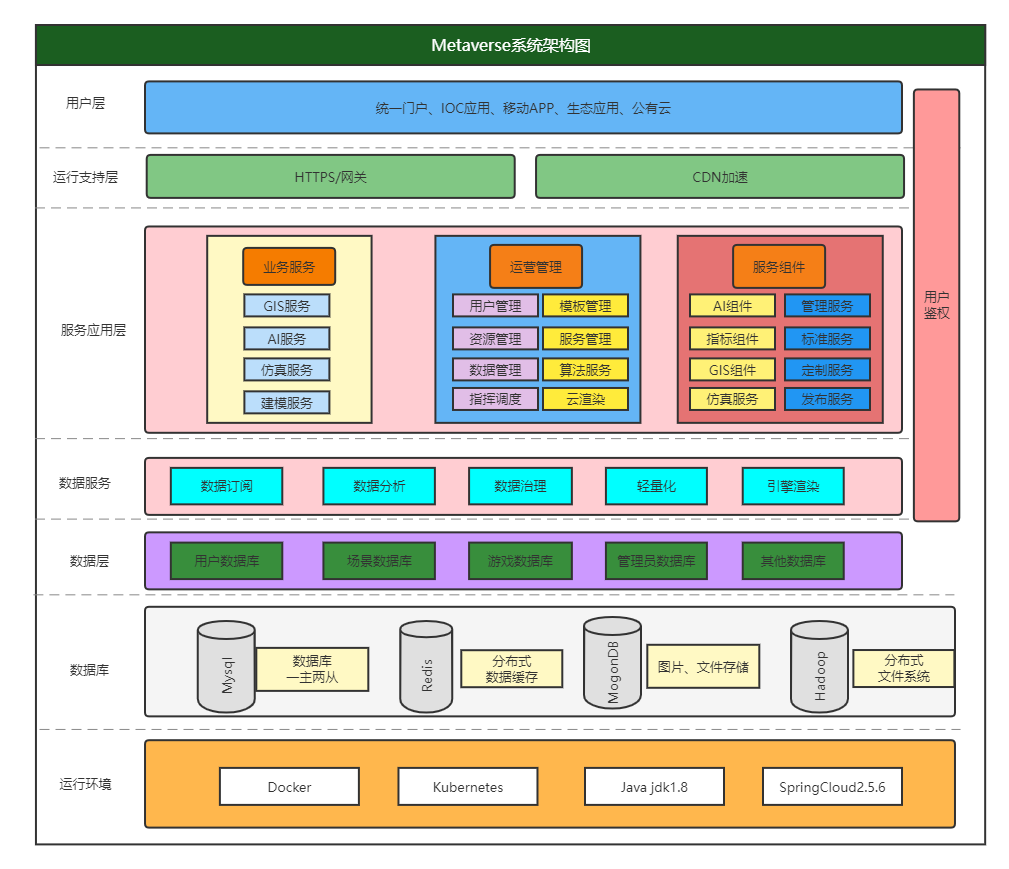
**软件体系结构**

**大作业报告**

|  |  |
| --- | --- |
| **姓名：** | **陈俊哲** |
| **学号：** | **19301141** |

A company is going to develop a Metaverse system. The system has the following major function: a user can live in the Metaverse, and a system manager can manage the Metaverse and users. The first function is achieved by a Metaverse living subsystem, and the second function is by a Metaverse management subsystem.

# 质量属性

软件架构图

# 系统体系风格

## Client-Server

1、将元宇宙用户生活构件和服务器构件分别运行在不同的计算机上，有利于分布式数据的组织和处理。

2、用户端侧重数据的显示和分析，服务器则注重数据的管理，包括收据订阅、数据分析、数据治理、引擎渲染等等。

3、构件之间是彼此独立和充分隔离的。

4、将大规模的业务逻辑分布到多个通过网络连接的低成本的计算机，降低了系统的整体开销。

## Model-View-Controller (MVC)

1、多个视图与一个模型相对应。

2、具有良好的移植性。

3、元宇宙系统被分割为三个独立的部分，当功能发生变化时，改变其中的一个部分就能够满足要求。

## Event Driven

1、事件声明者不需要知道哪些构件会响应事，如通过对元宇宙生活用户实时积分、游戏的监控，来自动响应后续的事件。

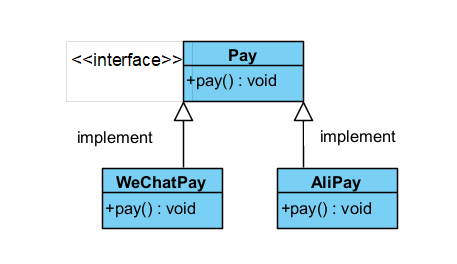
2、提高了元宇宙系统的复用能力。

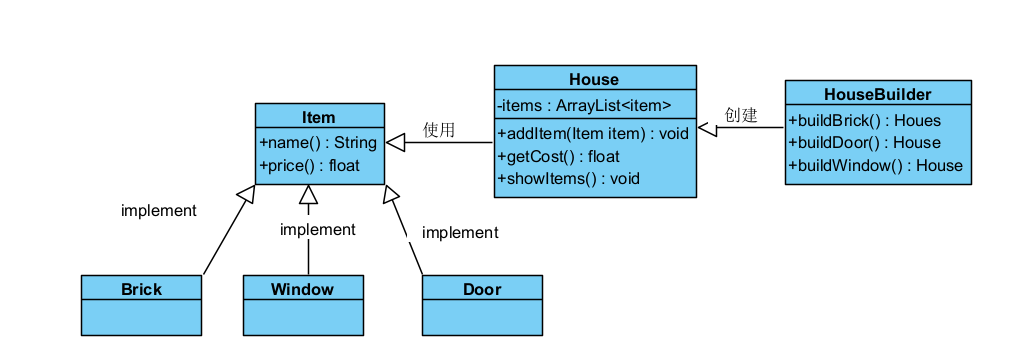
3、便于元宇宙用户生活系统升级。

# 设计模式

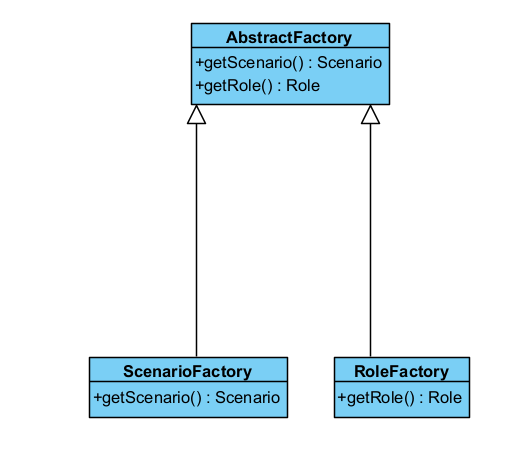
## 用户社交功能

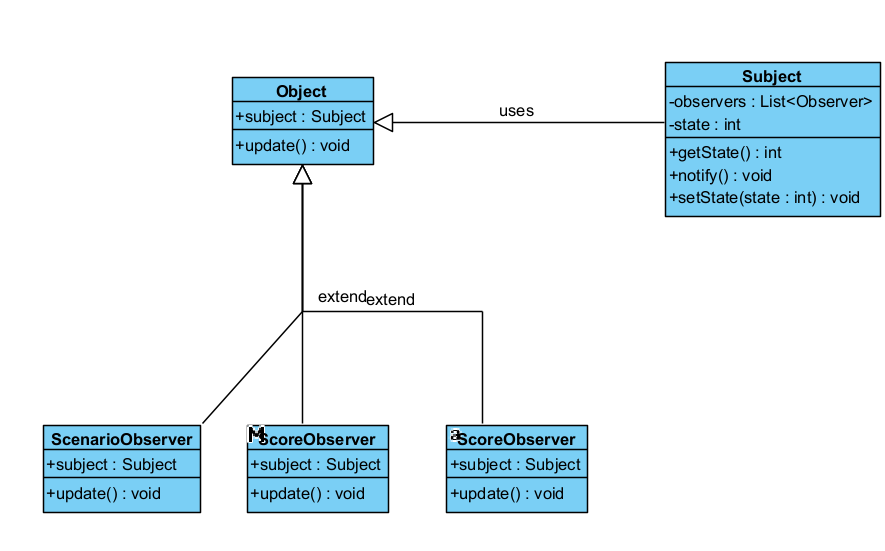
## 用户建造功能

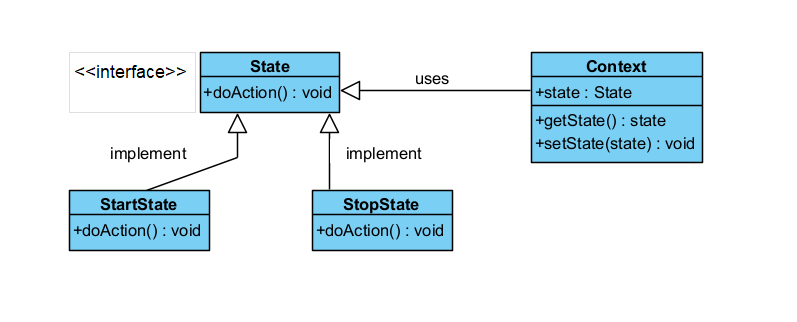
工厂方法模式：创建一个支付的接口，有不同的支付方式。

建造者模式：然后我们创建一个 *House*类，带有 *Item* 的 *ArrayList* 和一个通过结合 *Item* 来创建不同类型的 *House*对象的 *HouseBuilder*。

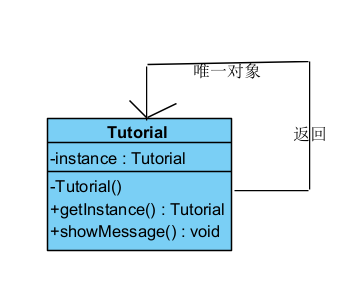
## 用户游戏功能

抽象工厂模式：游戏创建及运行有一个基类，游戏需要一个类去创建对应游戏的场景及角色。

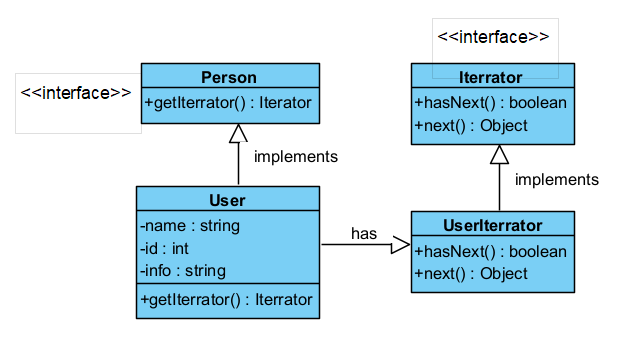
观察者模式：它将游戏对象与对象之间创建一种依赖关系，当其中一个对象发生变化时，它会将这个变化通知给与其创建关系的对象中，实现自动化的通知更新。

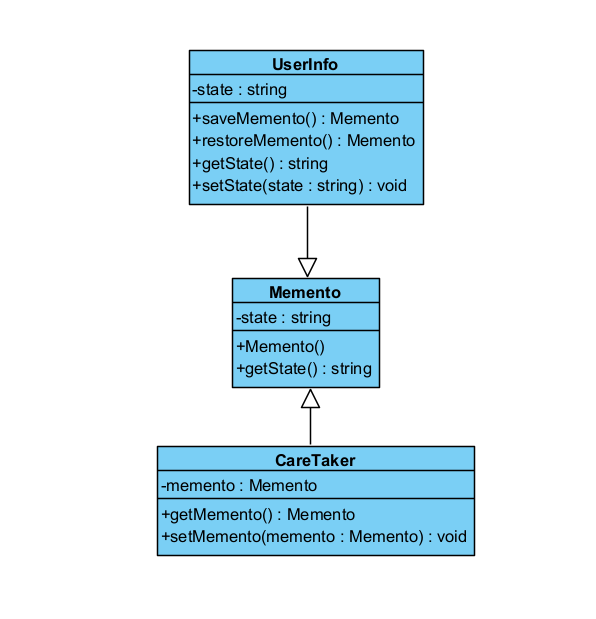
状态模式：在进行游戏的时候各个用户可以有不同的分数状态，对应着不同的游戏。

## 用户教程功能

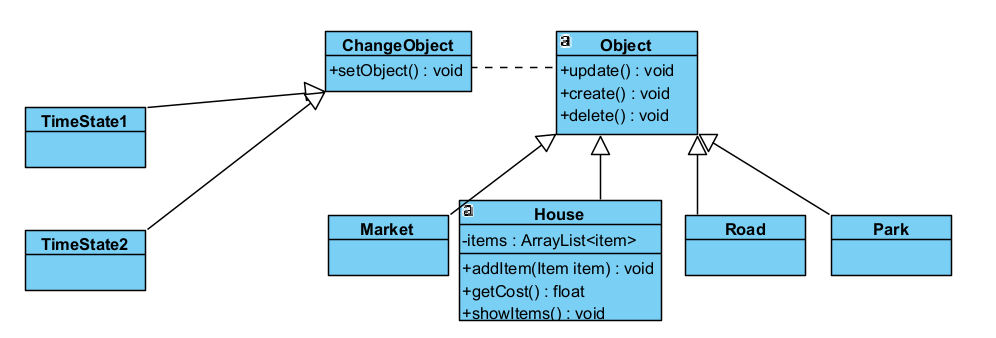
单例模式：针对每一个用户只有一份教程，防止创建的一个对象需要消耗的资源过多。

## 用户管理功能

迭代器模式：迭代器模式分离了用户对象的遍历行为，抽象出一个迭代器类来负责，这样既可以做到不暴露集合的内部结构，又可让外部代码透明地访问用户集合内部的数据。

备忘录模式：在用户信息管理模块中，用户可以修改自己的各项信息，在进行了错误操作之后，用户可以进行撤销，恢复到之前的状态，使得系统更加的人性化。

## 对象管理功能

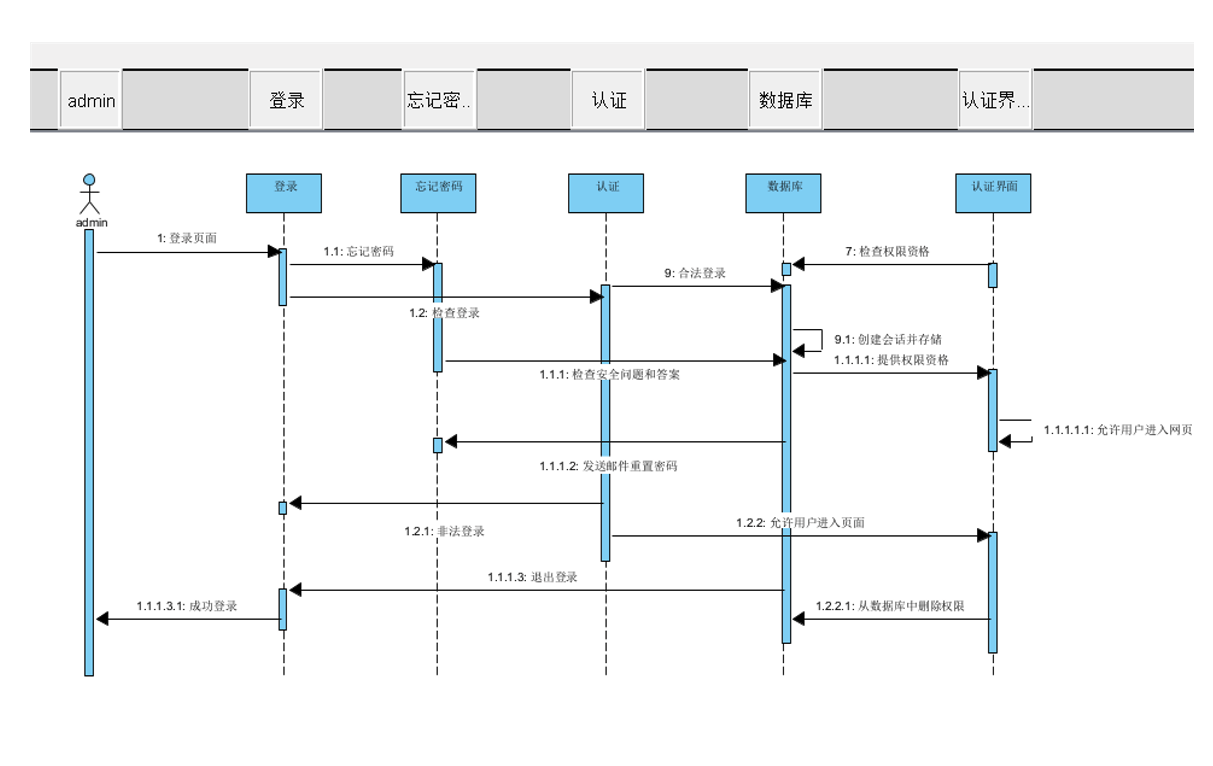
装饰器模式：在处于不同的节假日及特定时间时，对象的功能要求可以动态地添加，也可以再动态地撤销时。

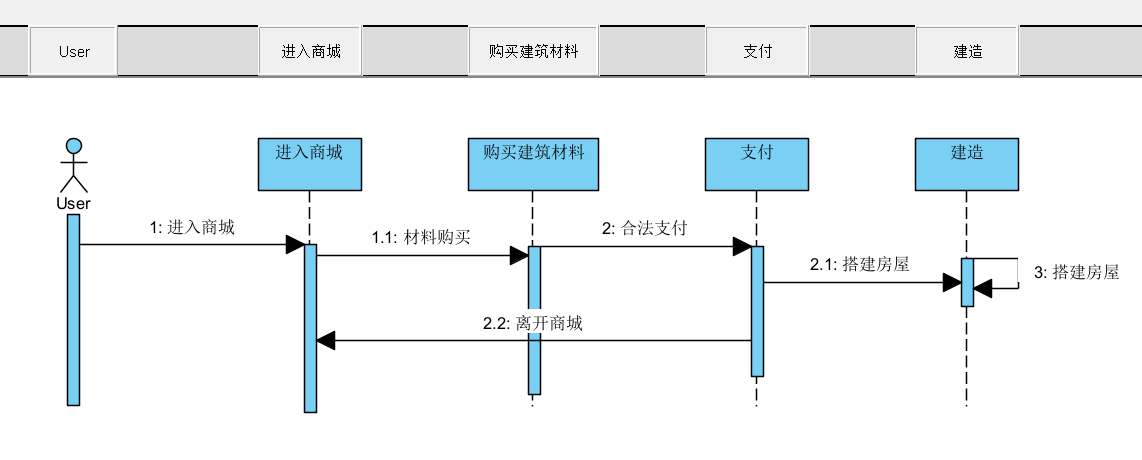
# 视图模型

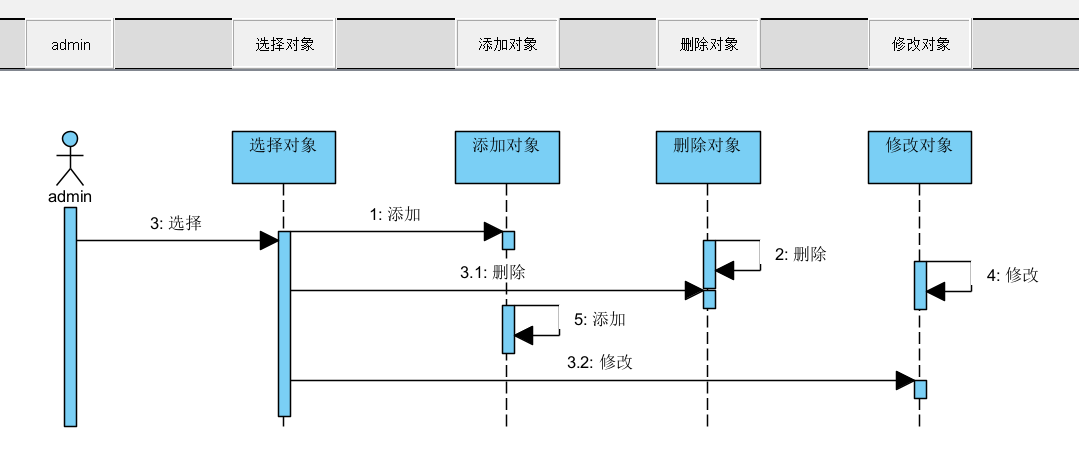
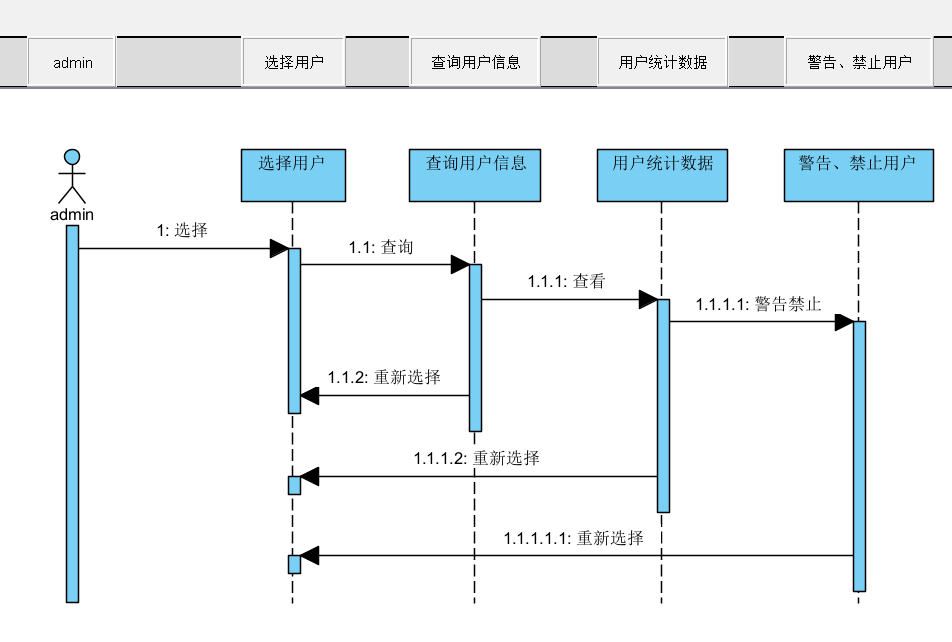
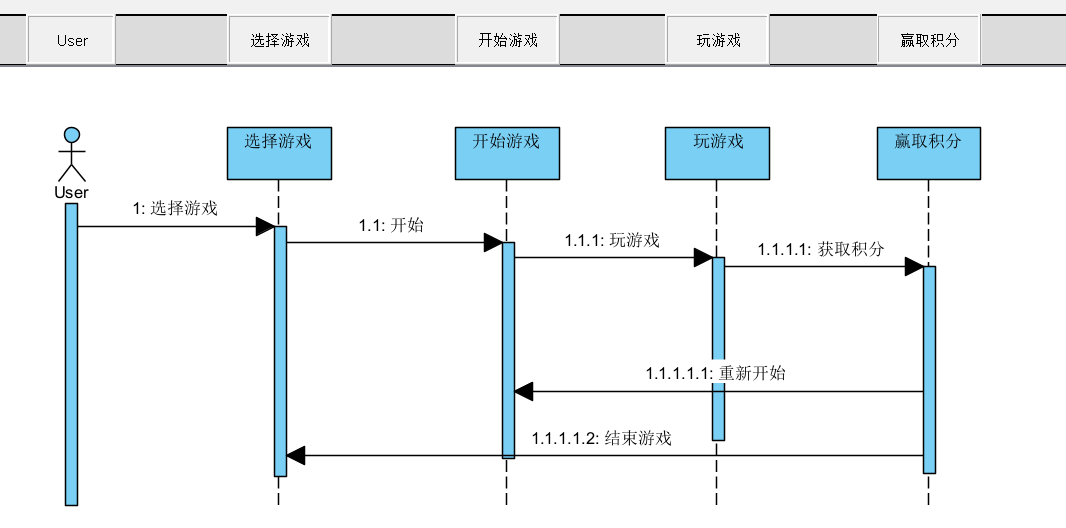
## 逻辑视图

参照设计模式

## 过程视图

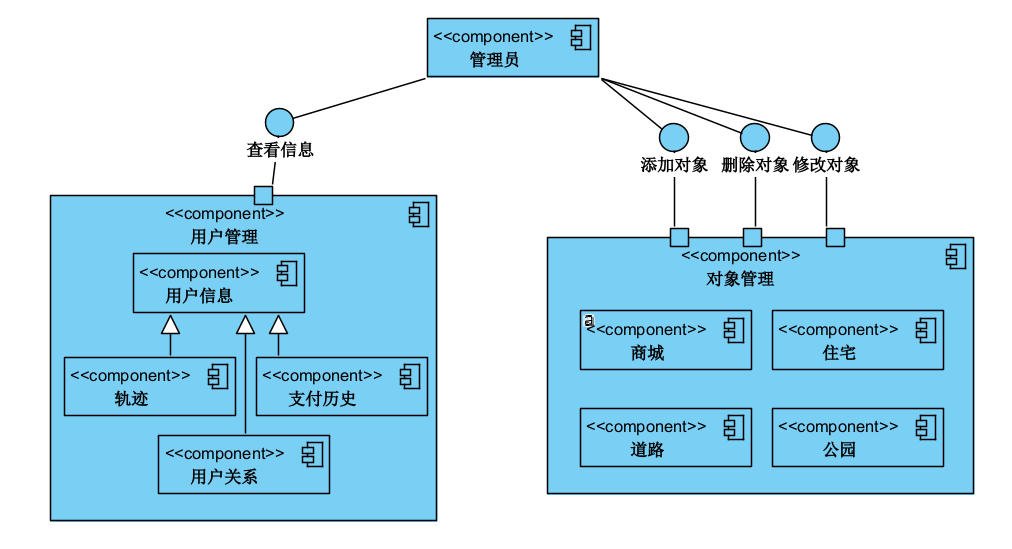
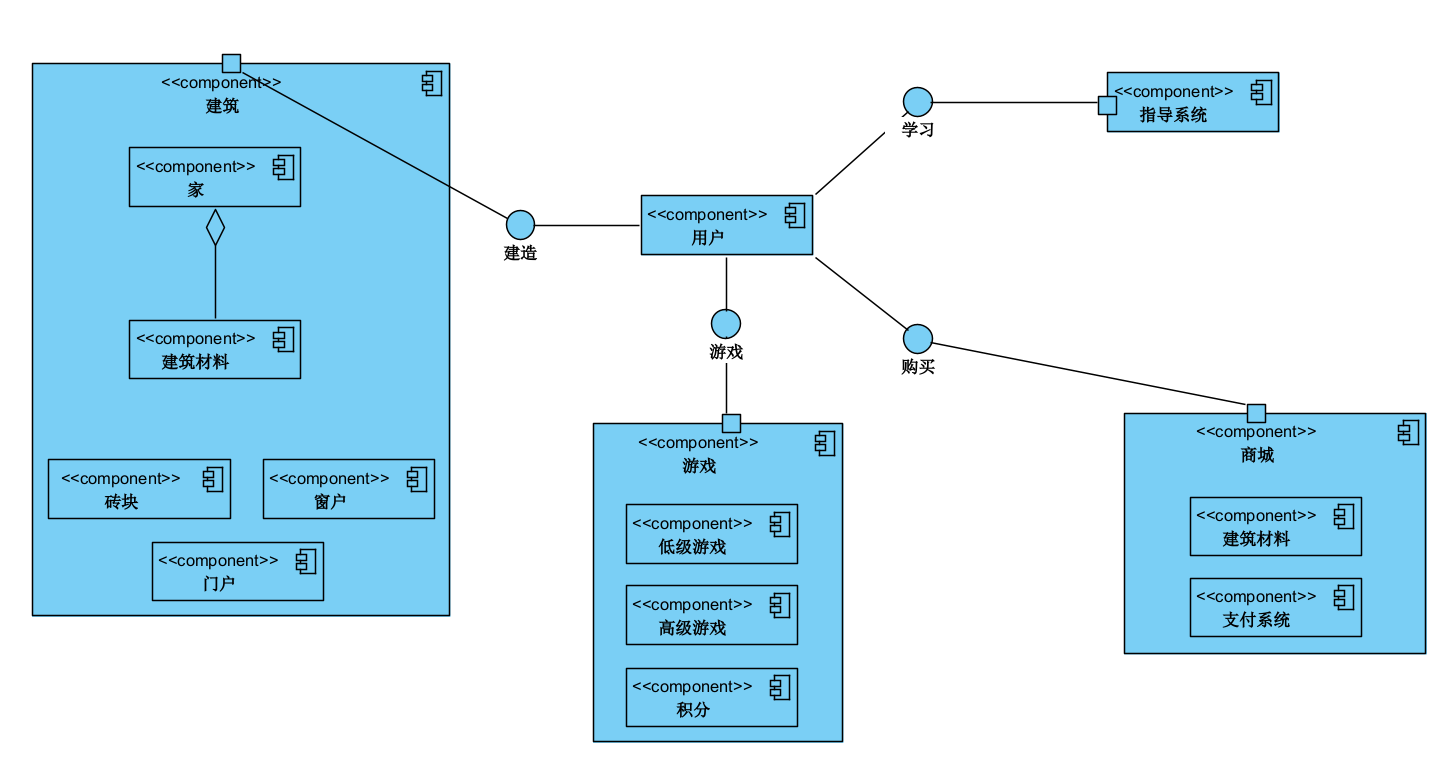
时序图



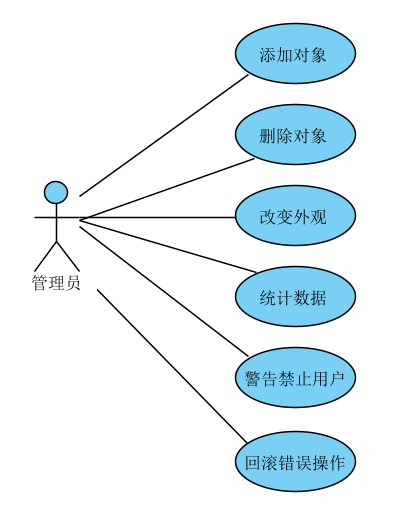
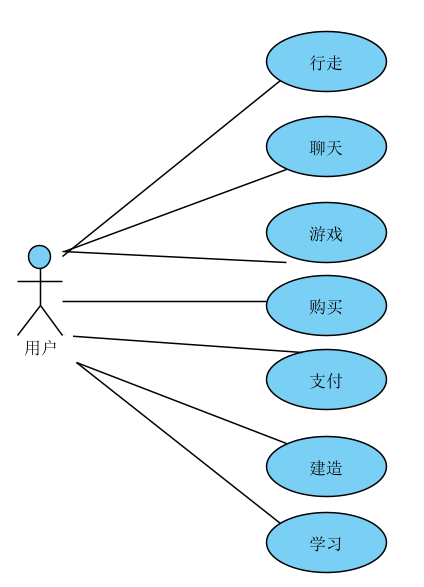
 

## 开发视图

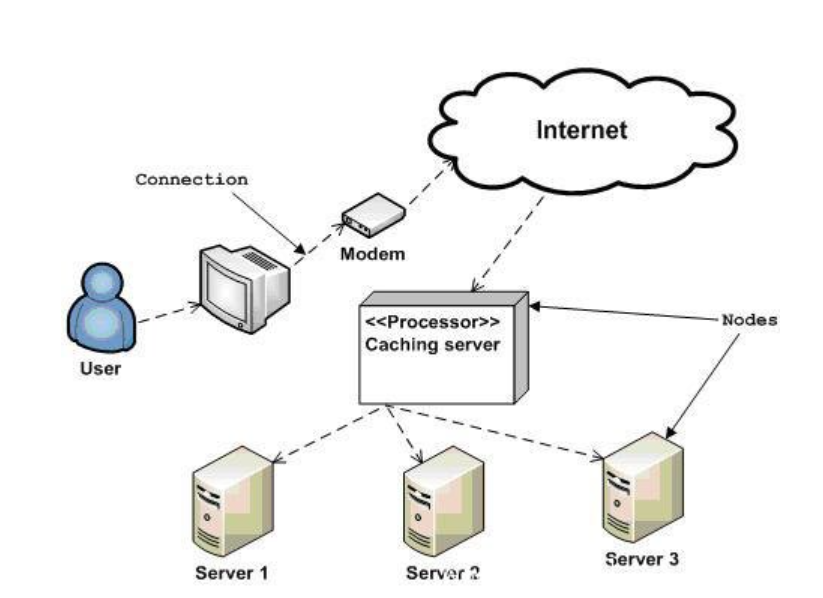
组件图、类图

## 场景视图

用例图

## 物理视图

部署图

# 设计原则

|  |  |  |  |
| --- | --- | --- | --- |
| 系统 | 功能 | 设计模式 | 设计原则 |
| 用户 | 用户建造功能 | 工厂方法模式 | 依赖倒置原则、里氏替换原则 |
| 建造者模式 | 开闭原则 |
| 用户游戏功能 | 抽象工厂模式 | 依赖倒置原则、里氏替换原则 |
| 观察者模式 | 单一责任原则、开闭原则 |
| 状态模式 | 单一责任原则 |
| 用户教程功能 | 单例模式 |  |
| 管理员 | 用户管理功能 | 迭代器模式 | 单一责任原则、开闭原则 |
| 备忘录模式 | 单一责任原则 |
| 对象管理功能 | 装饰器模式 | 单一责任原则、开闭原则 |