<移动迷宫>

软件设计模型

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **修订说明** | **作者** |
| <2017-04-26> | <1.0> | <初稿修订> | <组员> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[1引言 1](#_Toc448818441)

[1.1编写目的 1](#_Toc448818442)

[1.2适用范围 1](#_Toc448818443)

[1.3定义 1](#_Toc448818444)

[1.4参考资料 1](#_Toc448818445)

[1.5概述 1](#_Toc448818446)

[2．用例视图 1](#_Toc448818447)

[3. 逻辑视图 2](#_Toc448818448)

[3.1系统结构 2](#_Toc448818449)

[3.2 Use-Case实现 2](#_Toc448818450)

[3.3 设计类图 2](#_Toc448818451)

[3.4 其他图 2](#_Toc448818452)

[4. 实现视图 2](#_Toc448818453)

[4. 进程视图 3](#_Toc448818454)

[5. 部署视图 3](#_Toc448818455)

软件需求规约

# 1引言

## 1.1编写目的

本软件设计模型文档的编写目的是将激动个迷宫的各种设计模型进行绘制和整理，在之前的需求规约、系统分析的基础上，详细展示系统的结构，为后续的软件实现工作奠定基础。文档将围绕系统用UML语言，构造逻辑视图、实现视图、进程视图、部署视图等。本文档用于开发团队明确系统的设计模型，病以之为依据进行开发工作。

## 1.2适用范围

本文档适用的软件：移动迷宫。

与该软件相关的特性、子系统、模型等均符合本文档中的内容

## 1.3定义

本文件中涉及的术语定义在项目词汇表中给出。

## 1.4参考资料

《面向对象软件工程——使用UML、模式与JAVA》（第3版），清华大学出版社，2011。

## 1.5概述

本文档包括用例视图、逻辑视图、实现视图、进程视图和部署视图5个部分的模型。用例视图展现系统包含的用例。逻辑视图主要包括系统架构图、设计类图和用例的实现。实现视图对每个子系统构建组件图。进程视图用类图和组件图表示系统的进程和线程。部署视图展现系统的软硬件部署。本文件的各部分内容联系紧密，互为补充和对照，共同呈现本软件的设计模型

# 2．用例视图

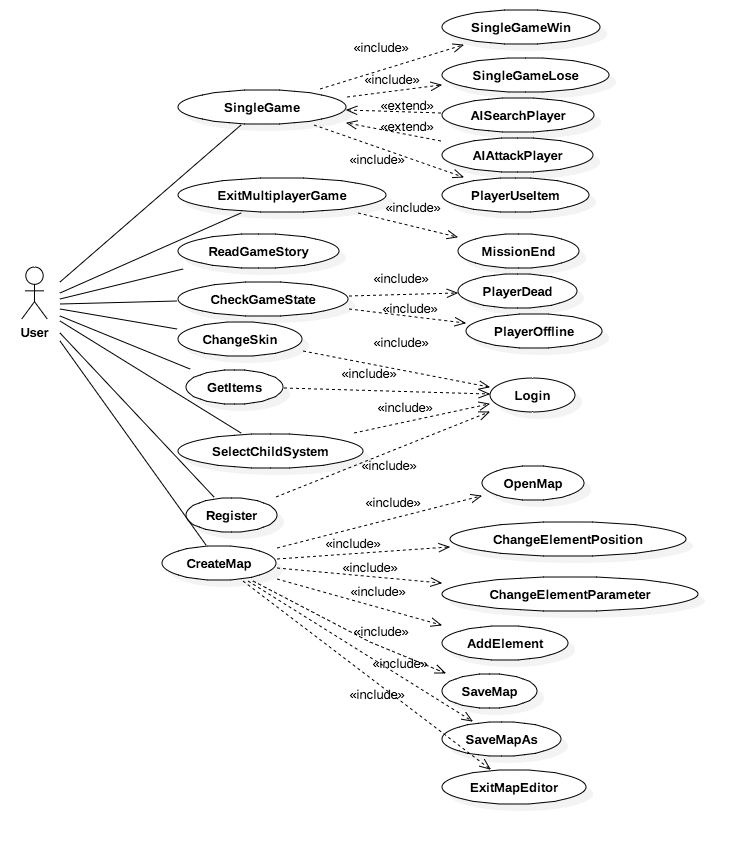


图2-1用例视图

# 3. 逻辑视图

## 3.1系统结构

系统架构包图如图3-1所示。整个系统分为7个子系统，分别是MainUI，ChangeSkin， SinglePlayerGame，MultiPlayerGame，MapEditor，StoryBackGround，CommonComponent。

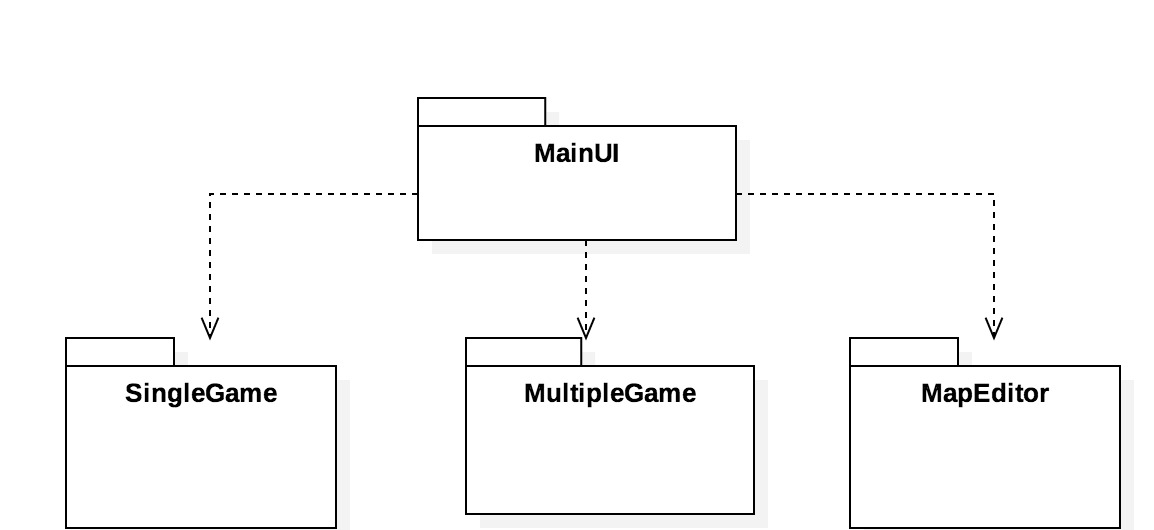


图3-1系统构架包图

每个子系统的包图如下：

1. MainUI的类图：

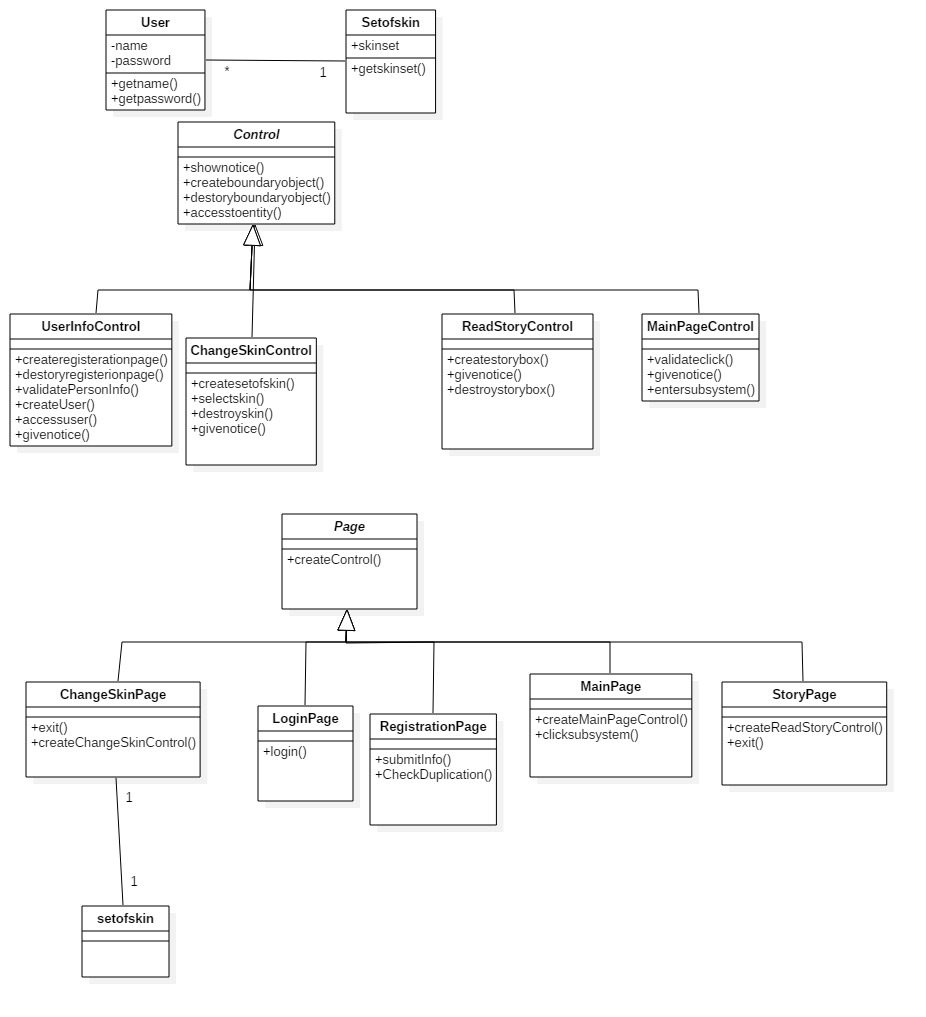


图3-2 MainUI类图

1. SingleGame的类图：

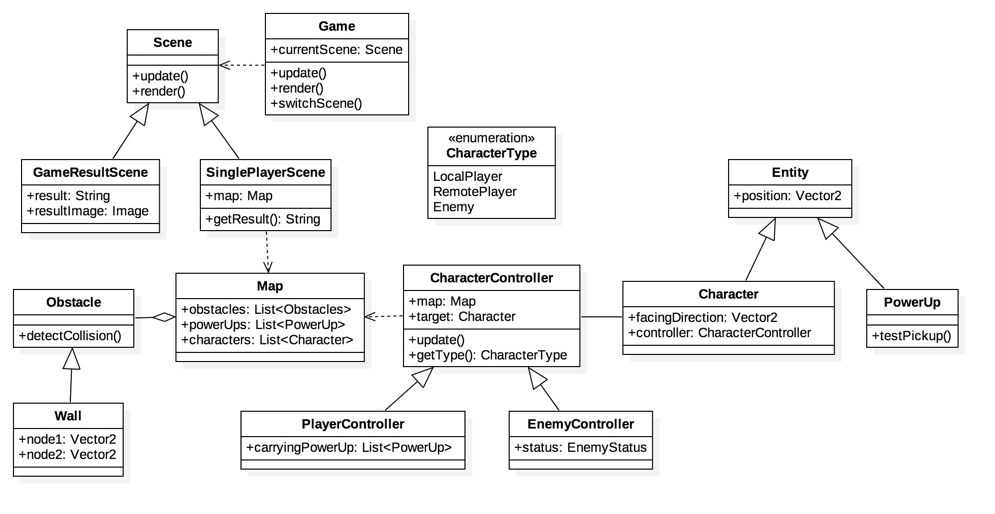


图3-3 SingleGame类图

1. MultipleGame的类图

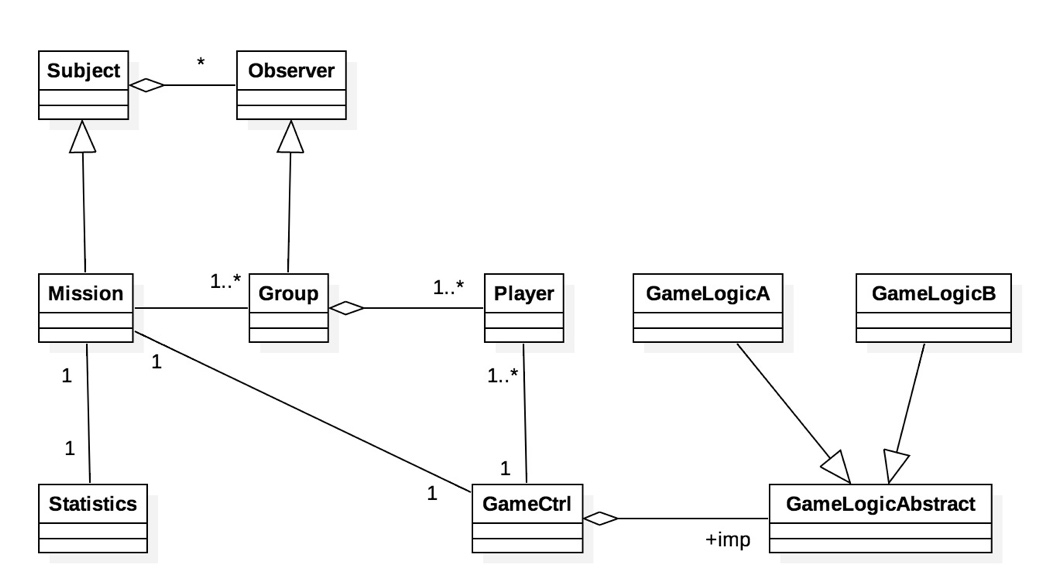


图3-4 MultipleGame类图

1. MapEditor的类图

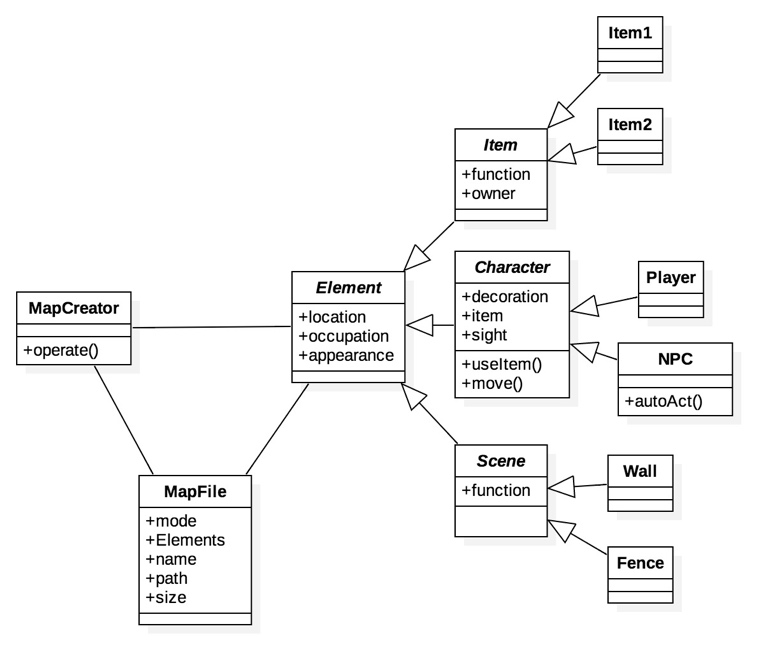
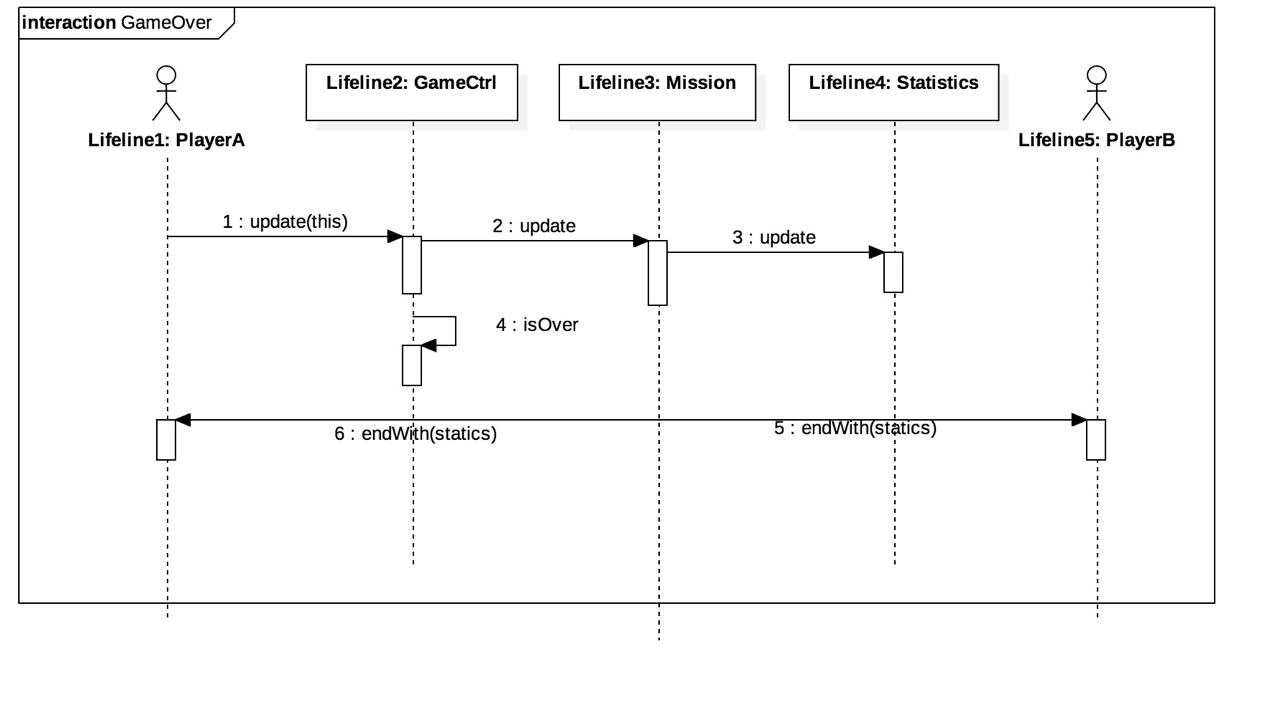
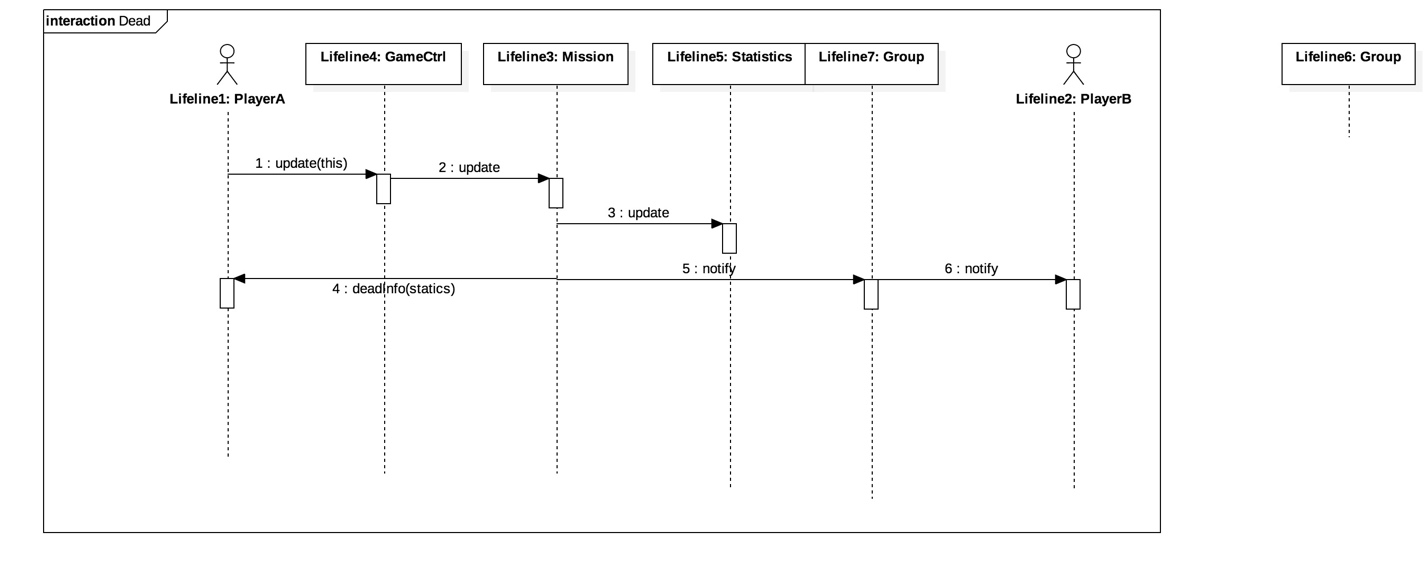


图3-5 MapEditor类图

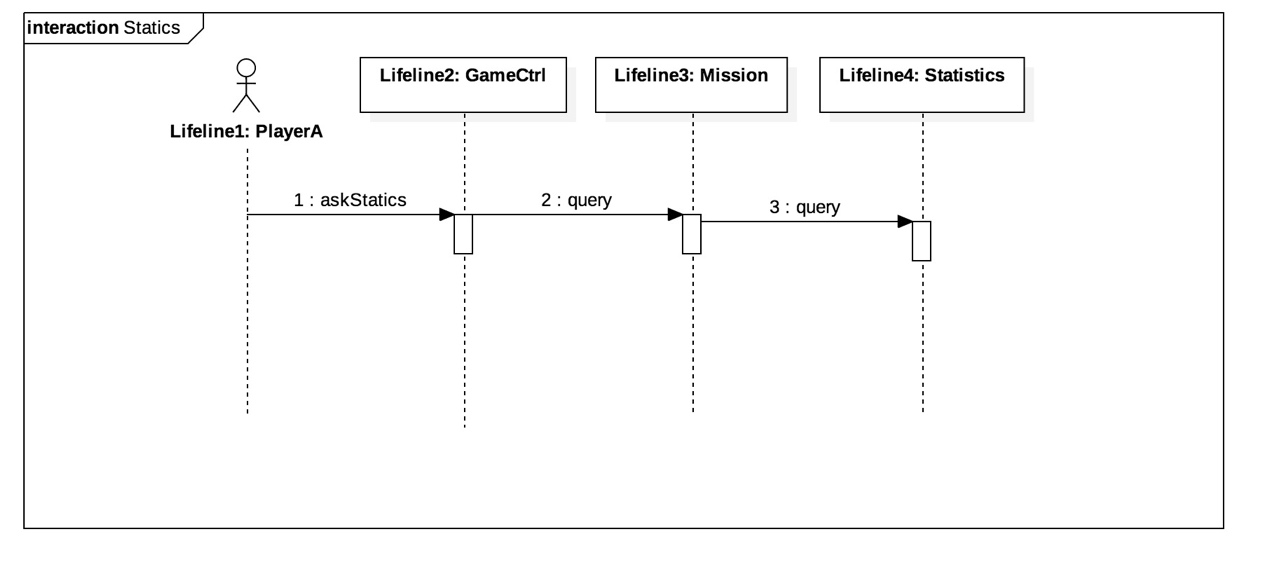
## 3.2 Use-Case实现

<GameOver>实现

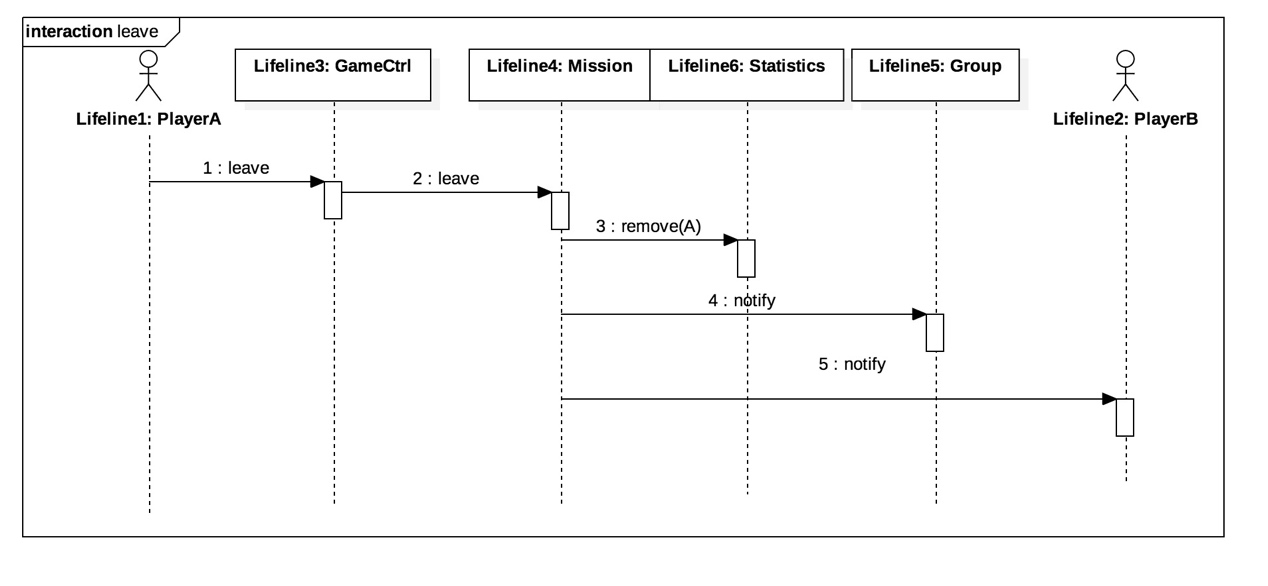
<Dead>实现



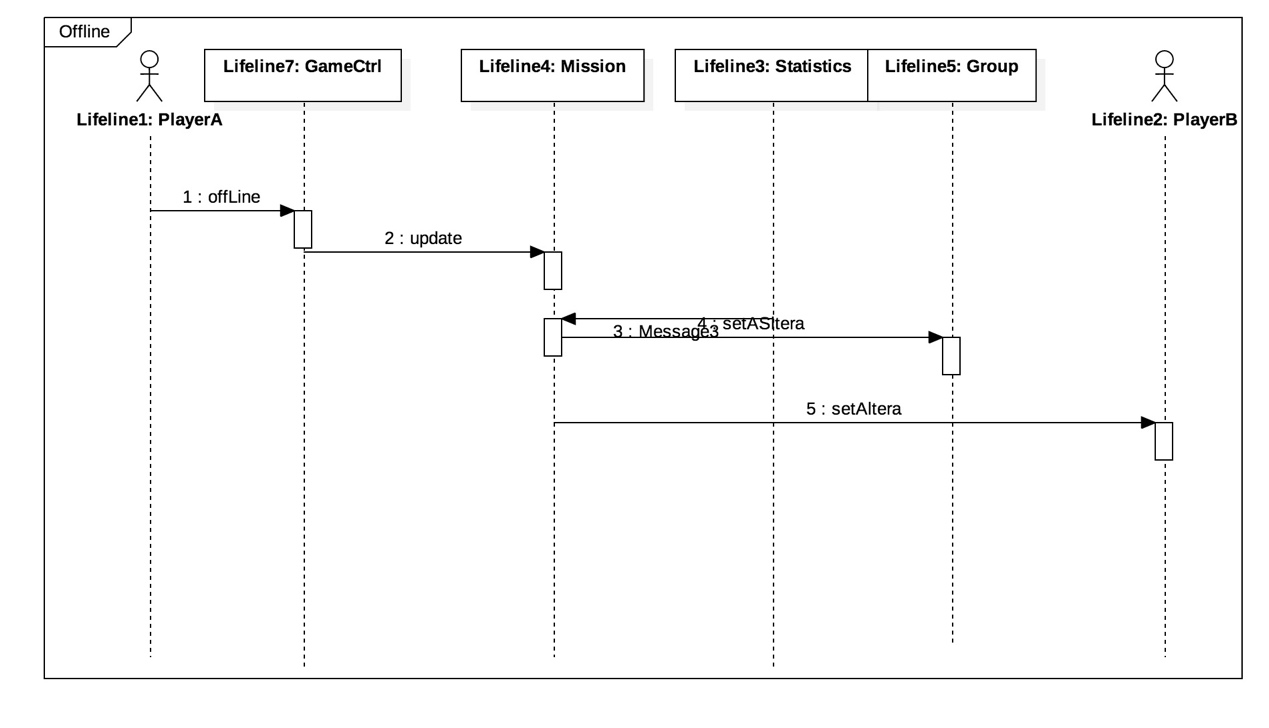
<Statics>实现



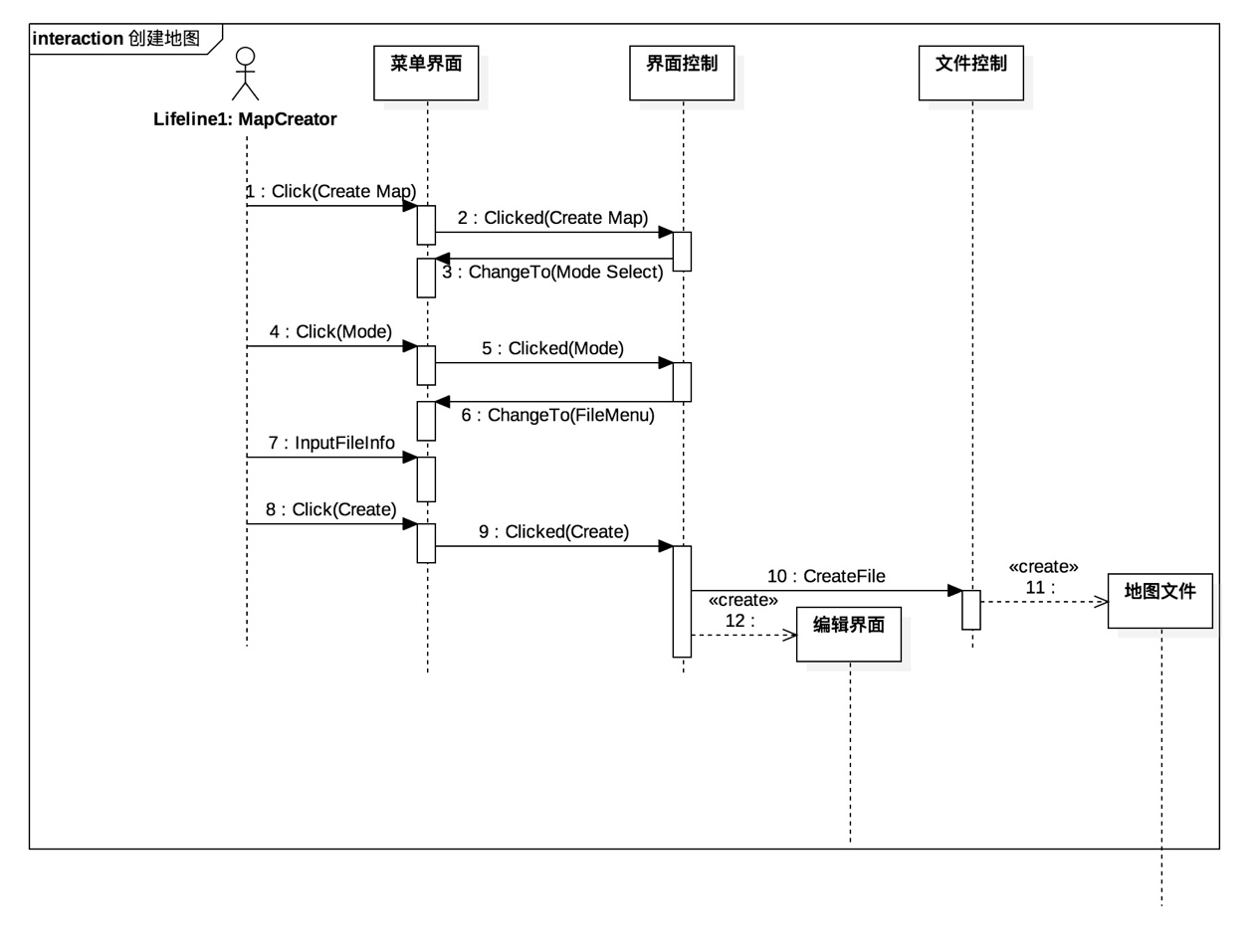
<Leave>实现



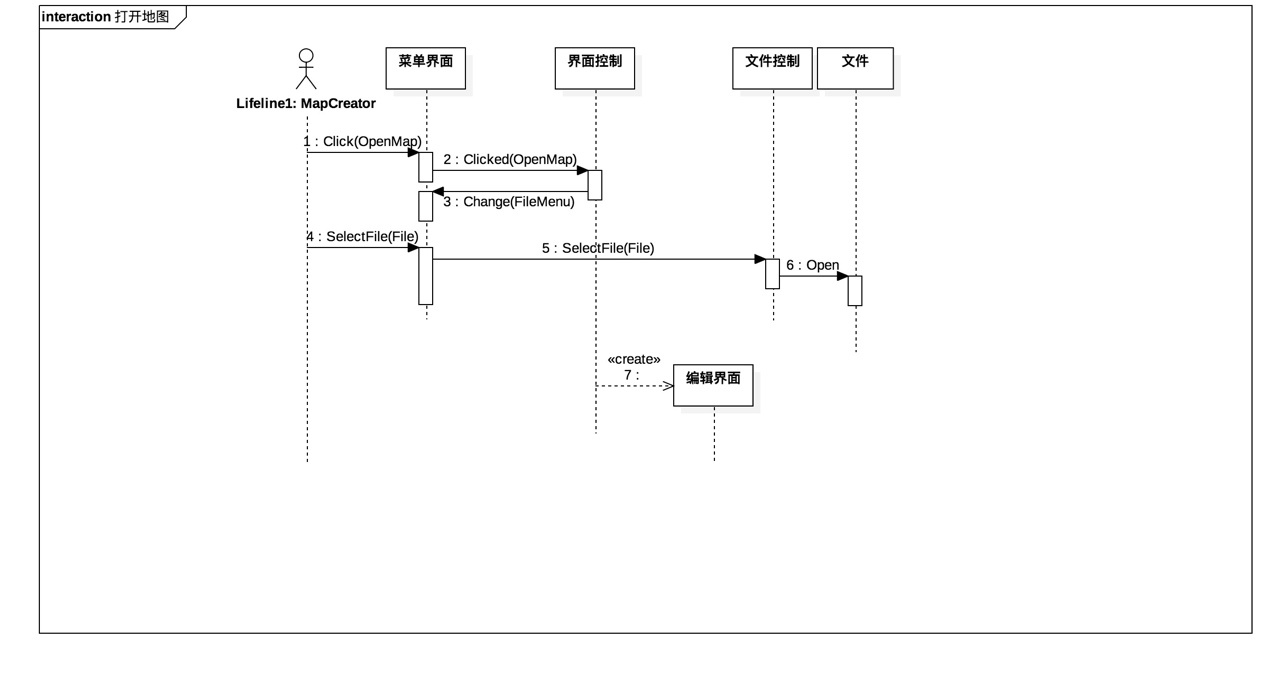
<Offline>实现



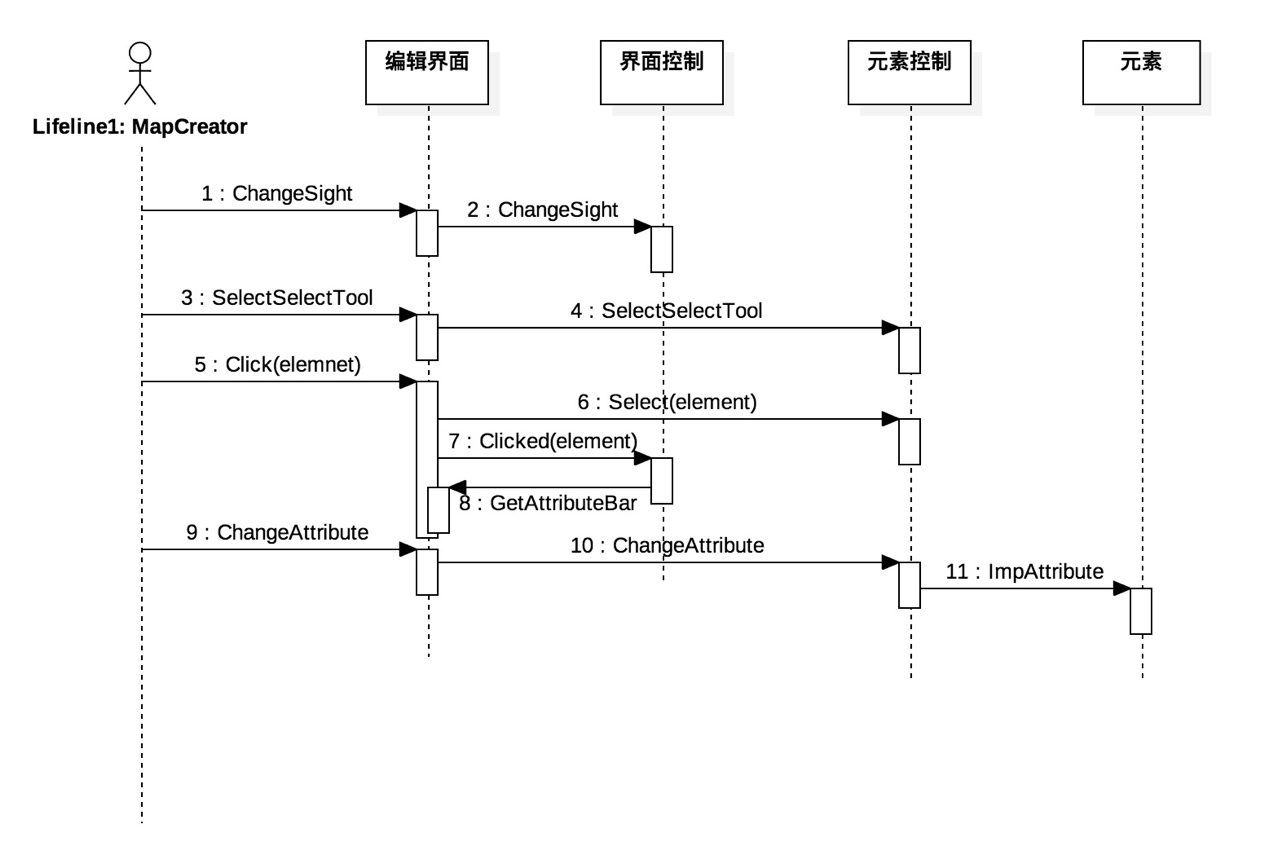
<CreateMap>实现



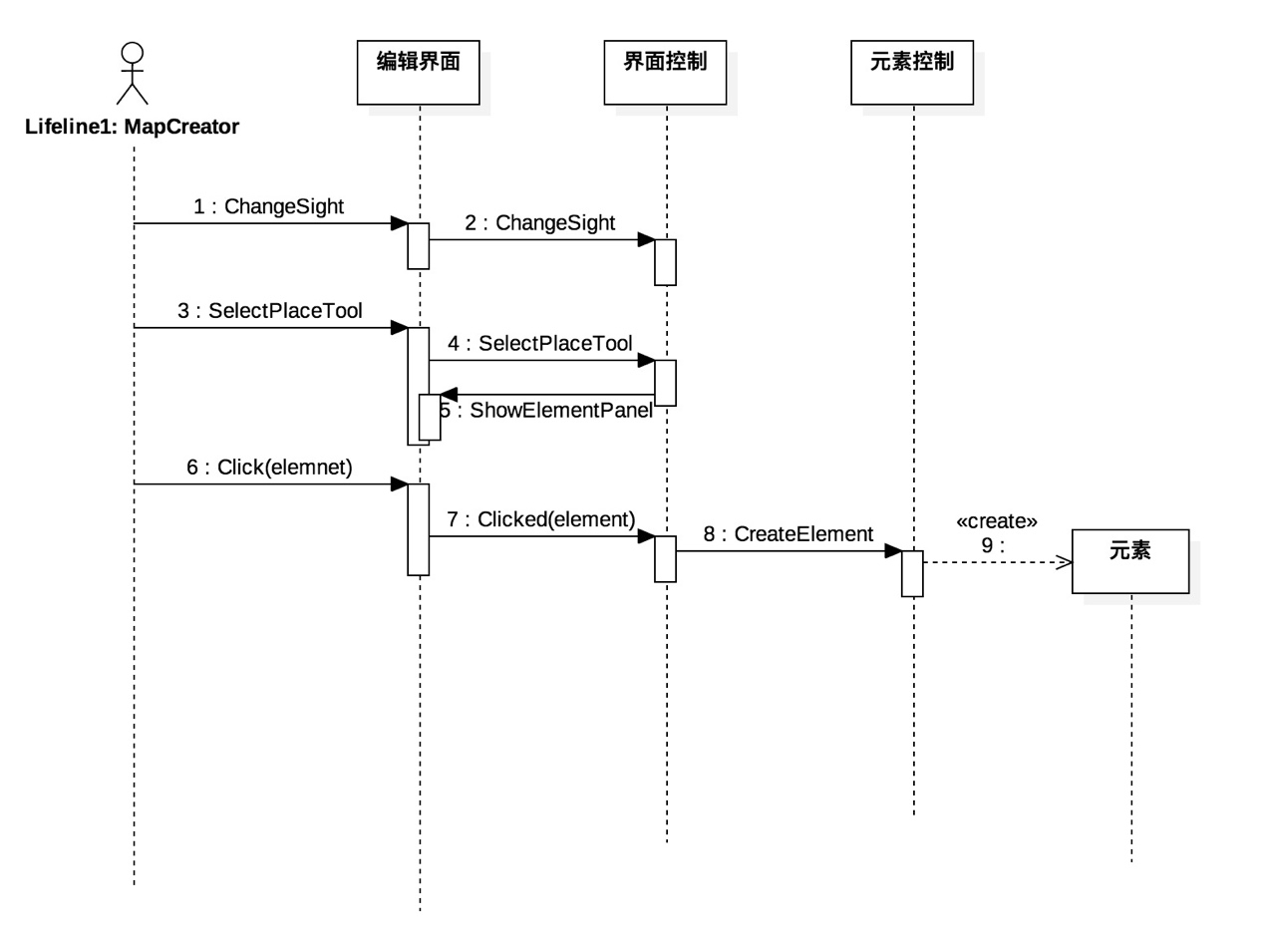
<OpenMap>实现



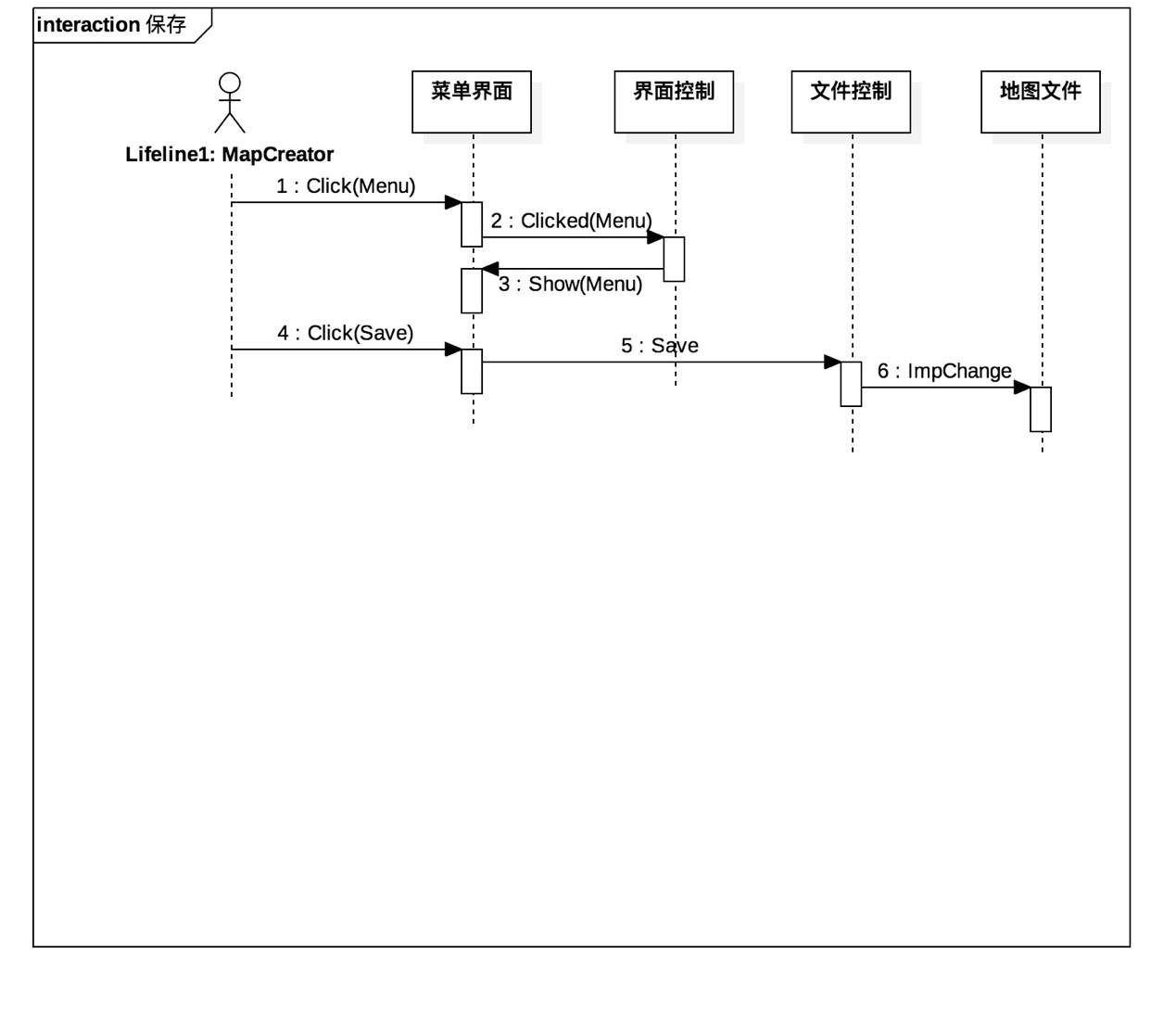
<Edit>实现



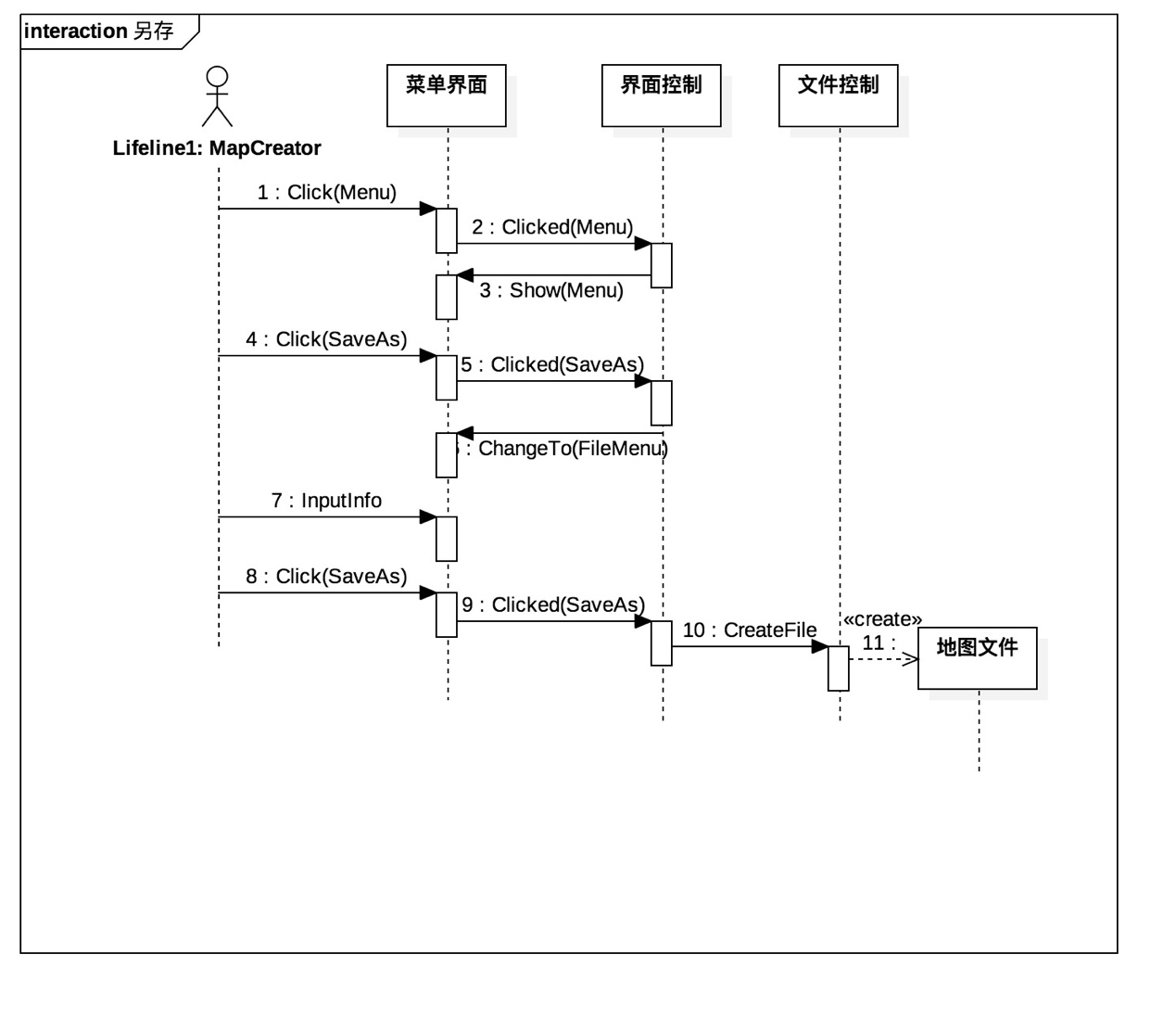
<Add>实现



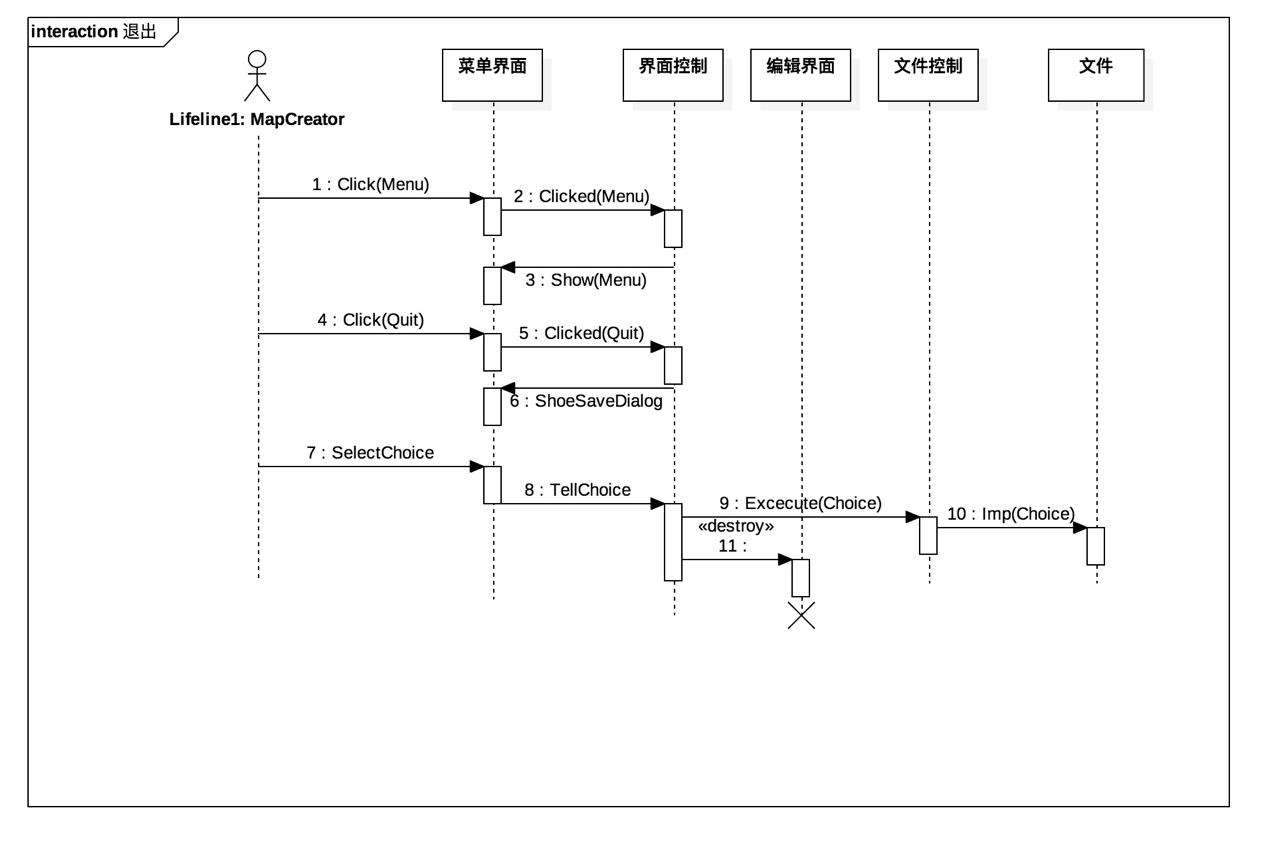
<Save>实现



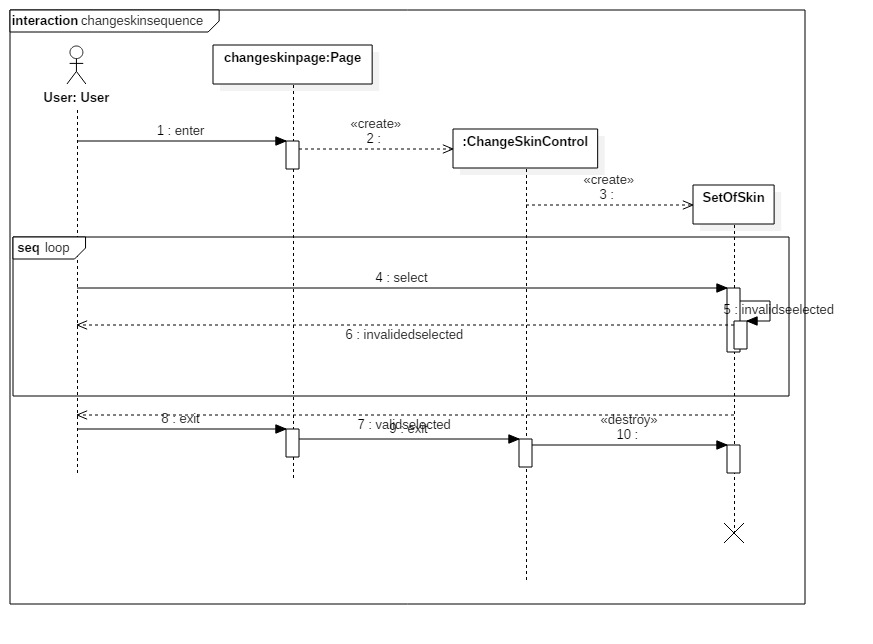
<SaveAs>实现



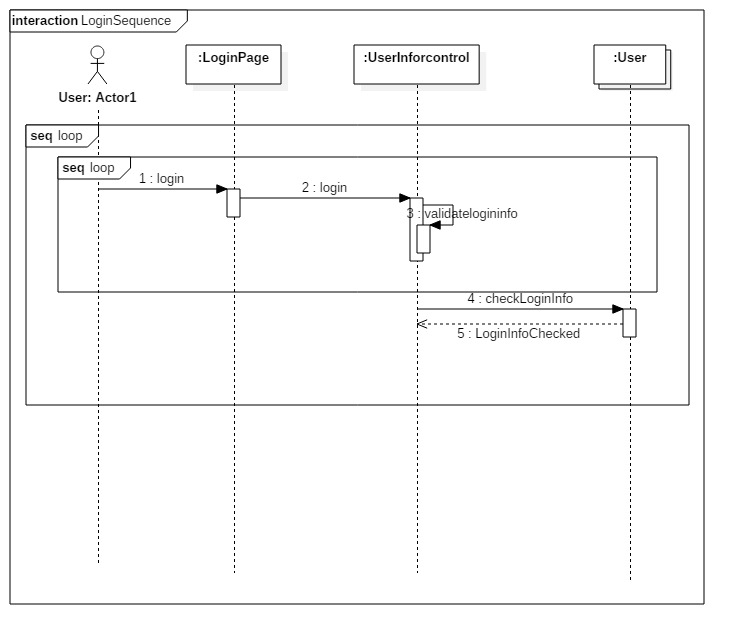
<Quit>实现



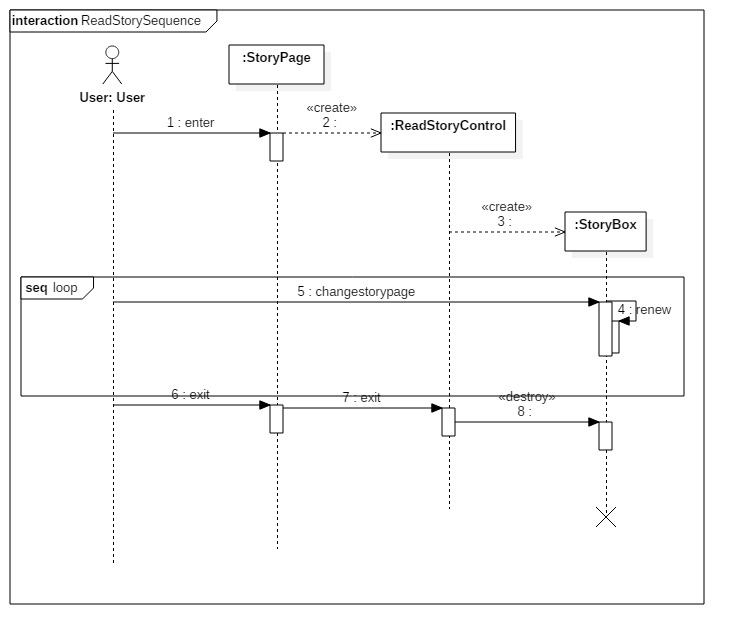
<ChangeSkin>实现



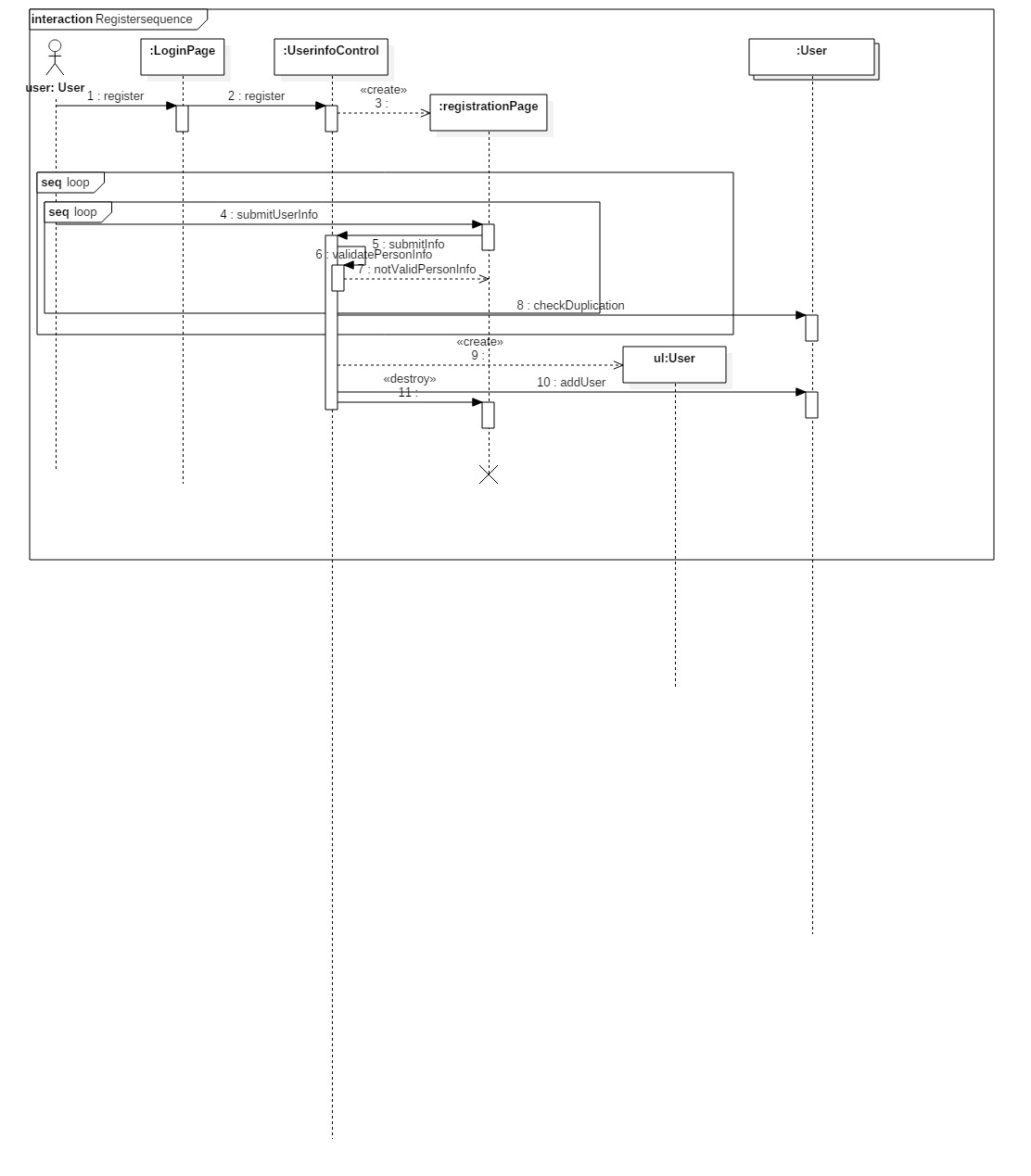
<Login>实现



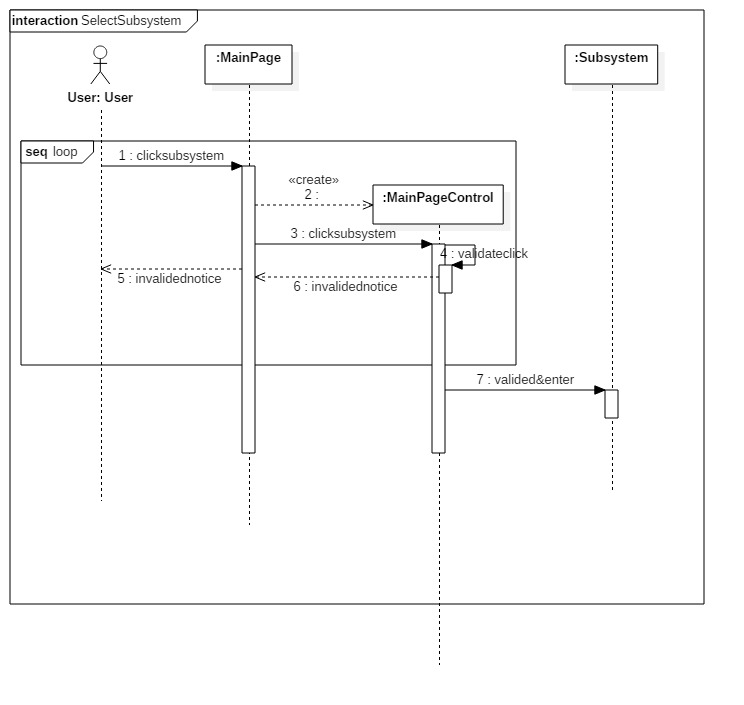
<ReadStory>实现



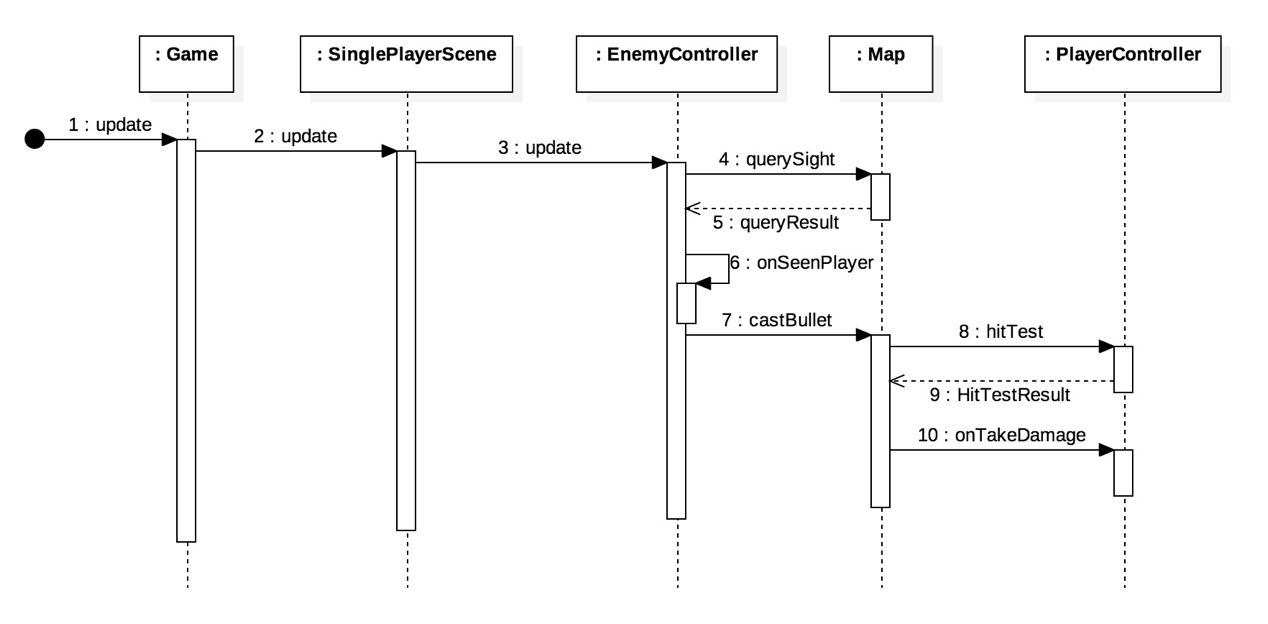
<Register>实现



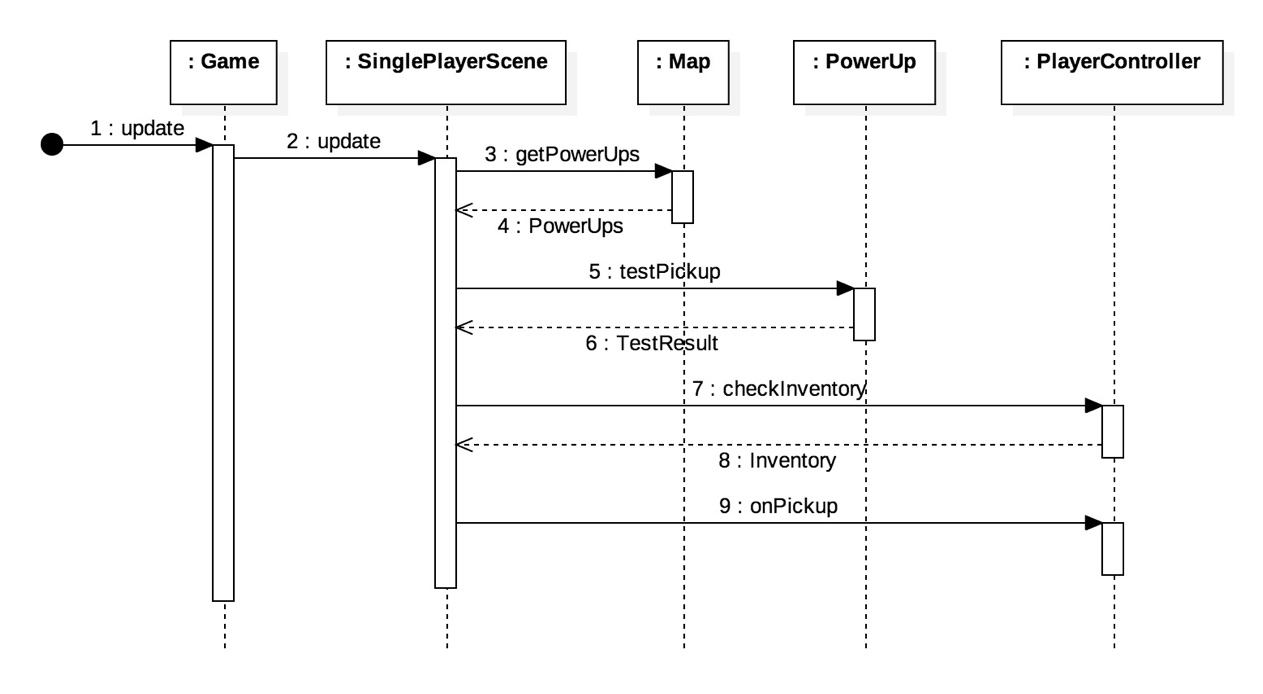
<Select>实现



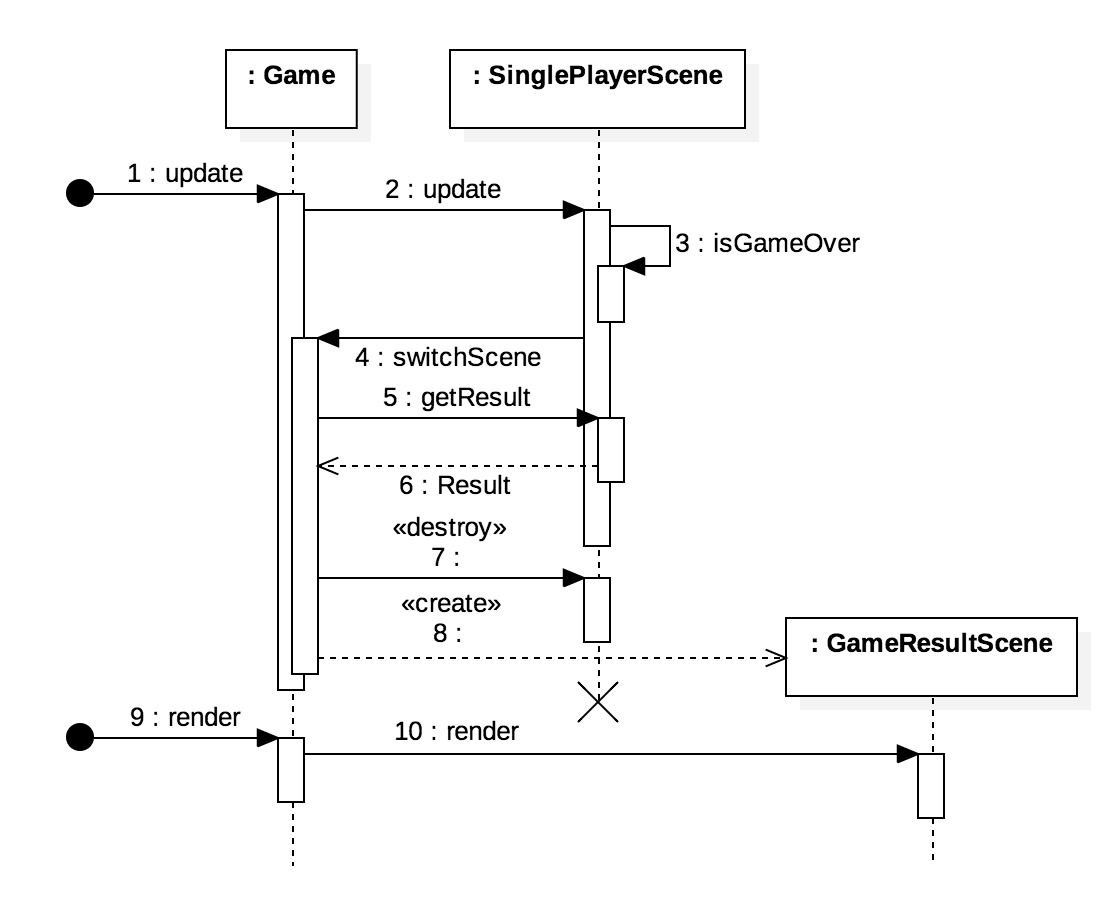
<AIAction>实现



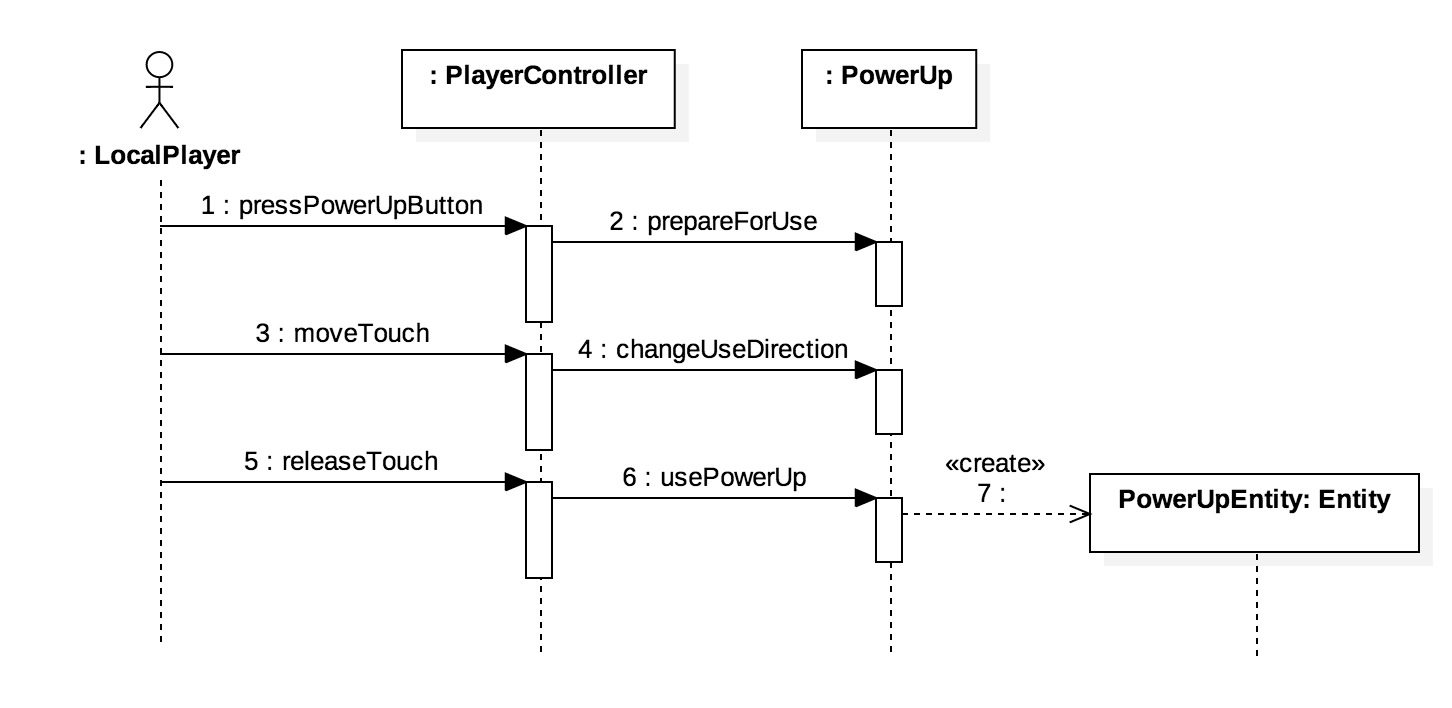
<SinglePlayerAcquirePowerUp>实现



<SinglePlayerGameOver>实现



<UserPowerUp>实现



## 3.3 设计类图

（1） 用户接口子系统

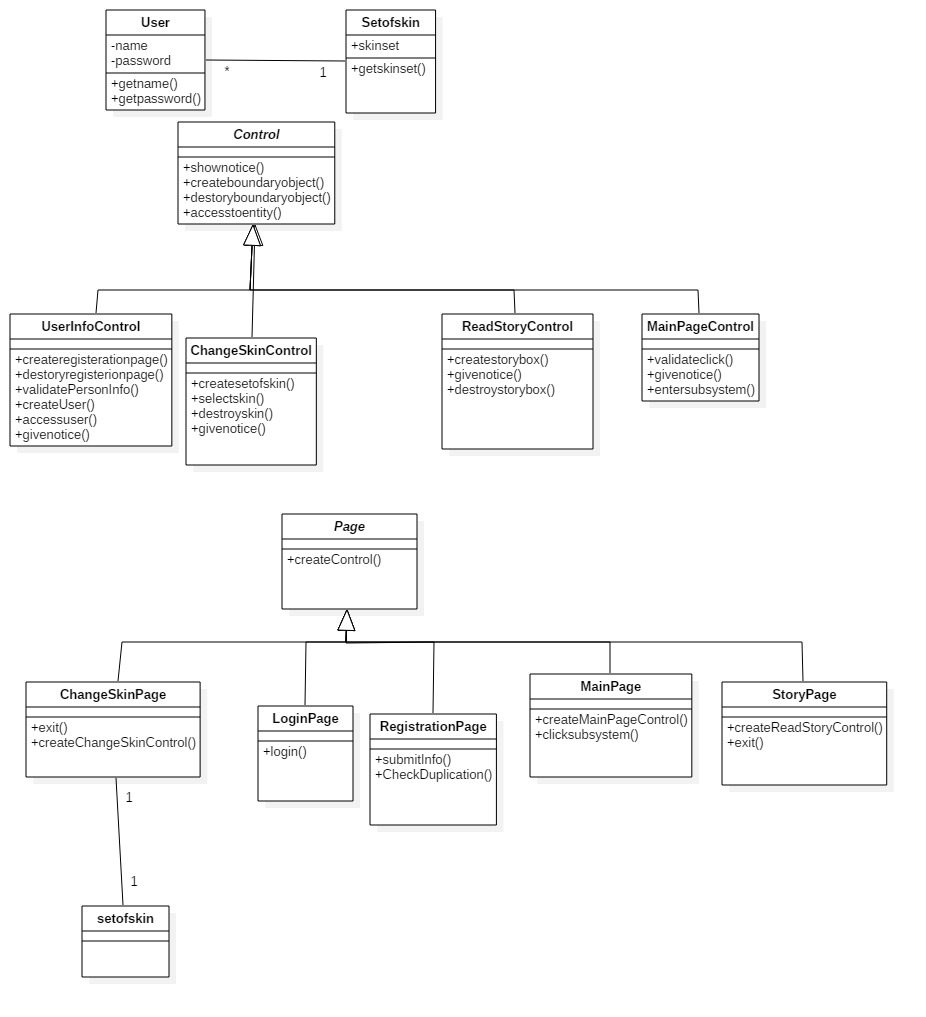


图3-6用户接口子系统类图

（2） 地图编辑器子系统

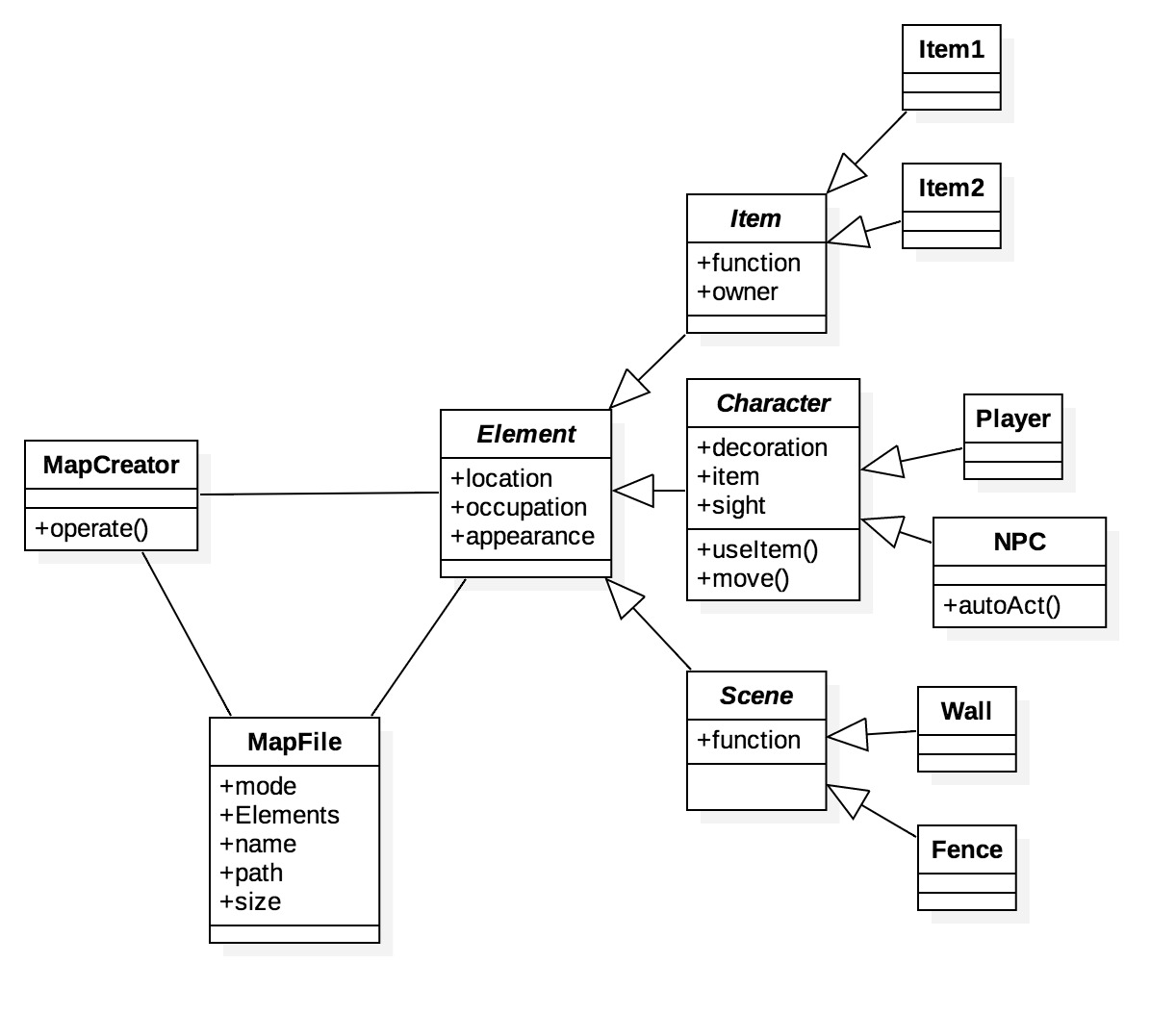


图3-7地图编辑器子系统类图

（3）多人游戏子系统

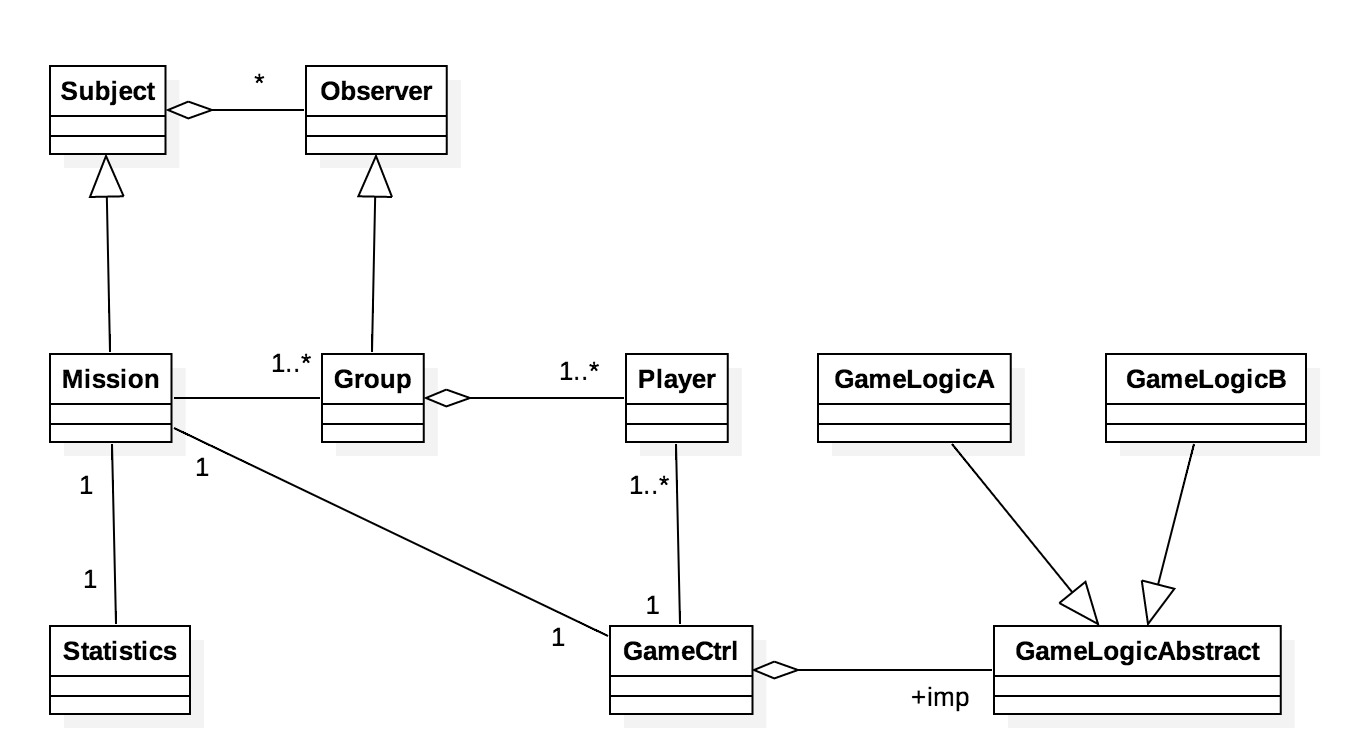


图3-8多人游戏子系统类图

（4）单人游戏子系统

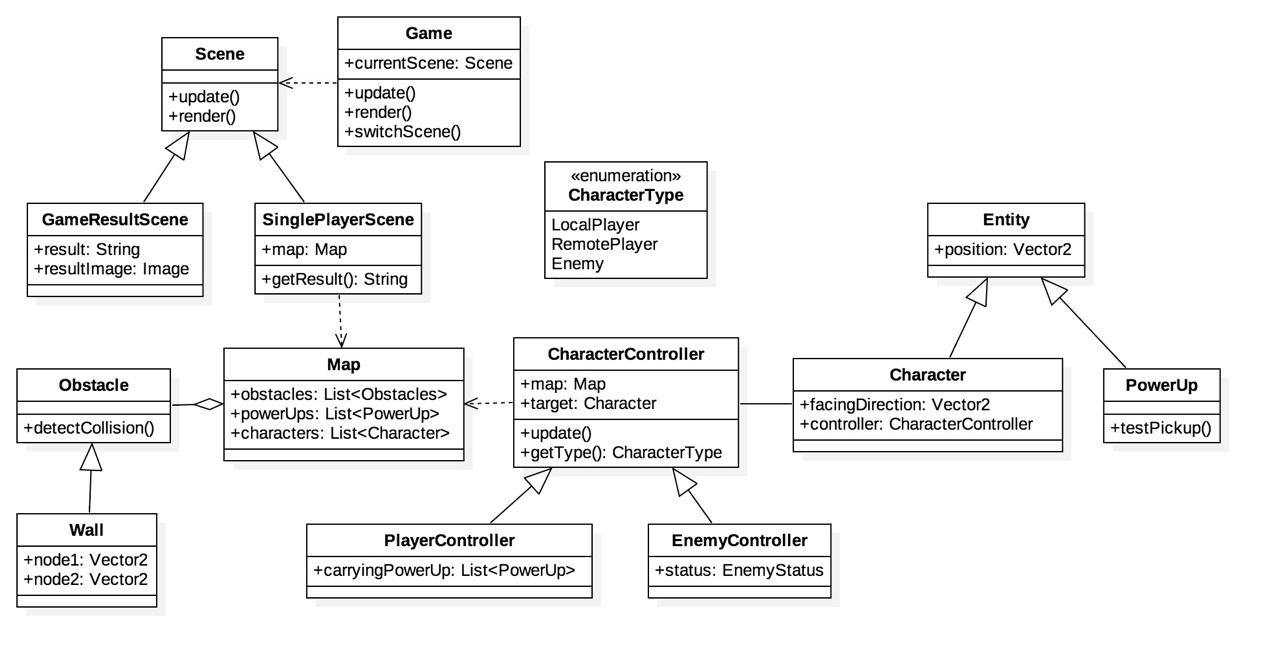
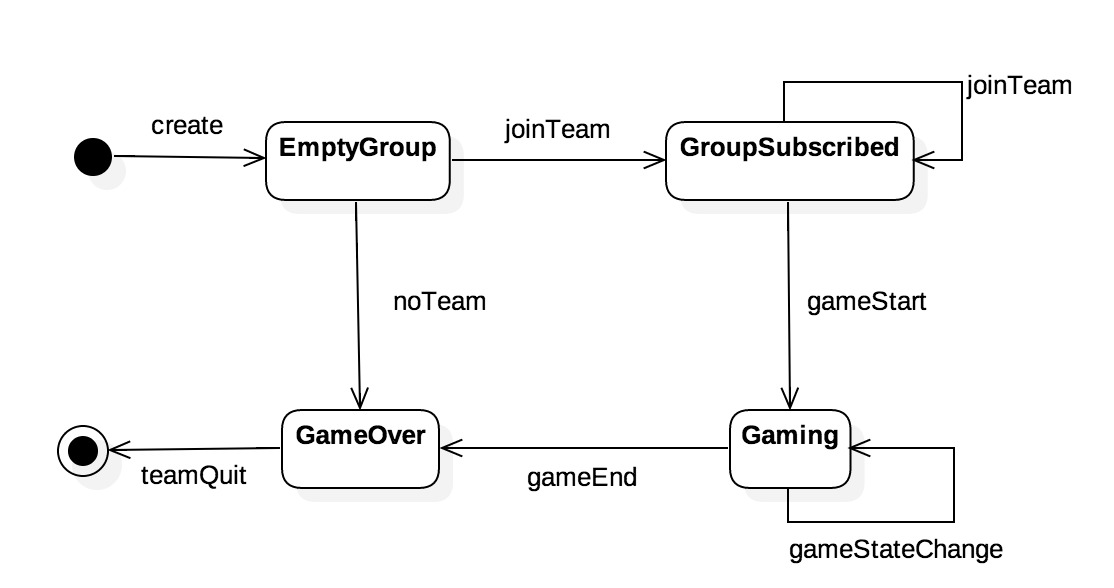


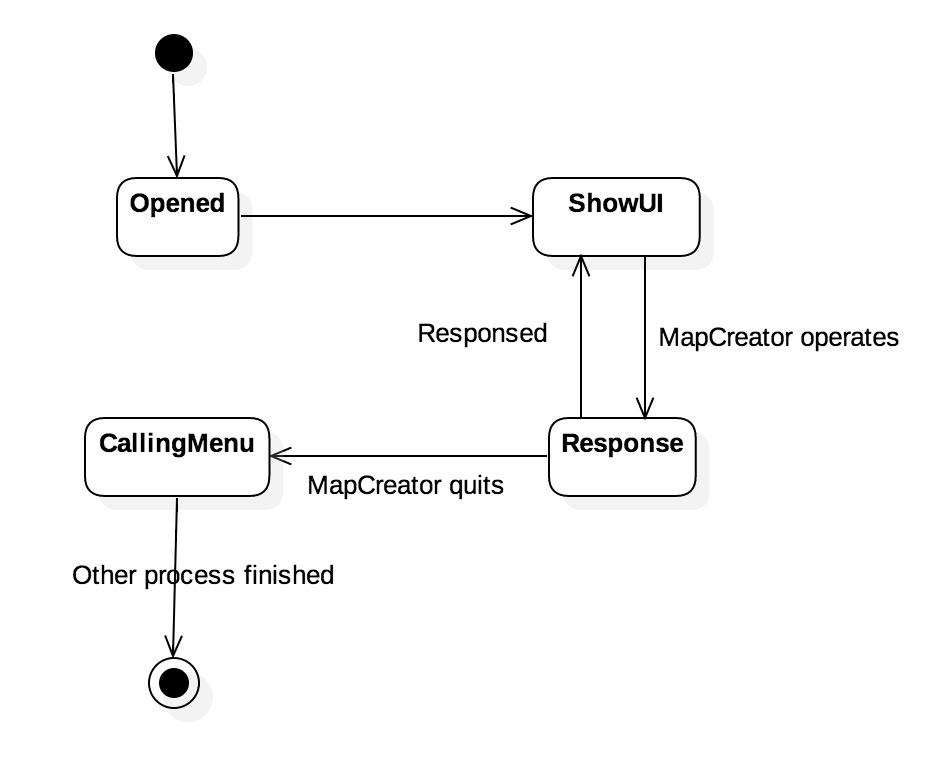
图3-9单人游戏子系统类图

## 3.4 其他图

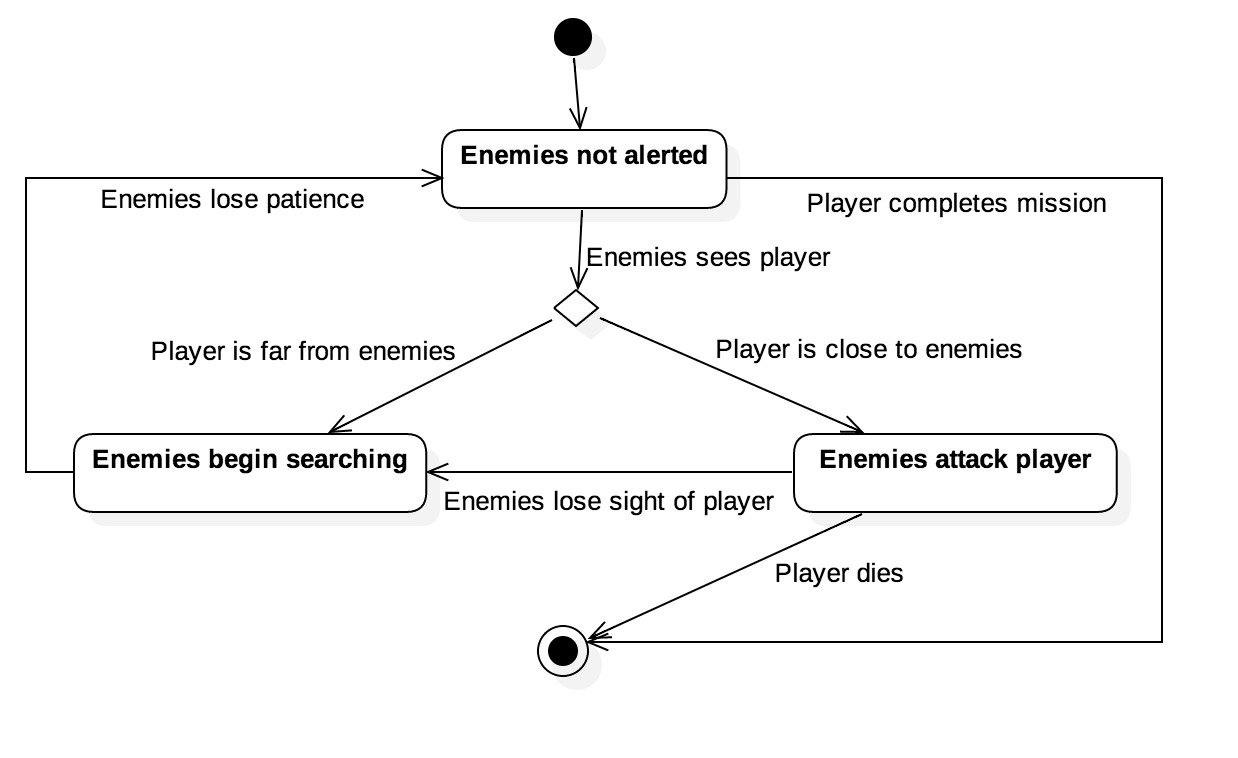
状态图1:Multiplayer



状态图2:EditUI



状态图3:SinglePlayer



# 4. 实现视图

针对每一个子系统，画出其对应的组件图。需要画两种组件图：

1. 用户接口子系统。

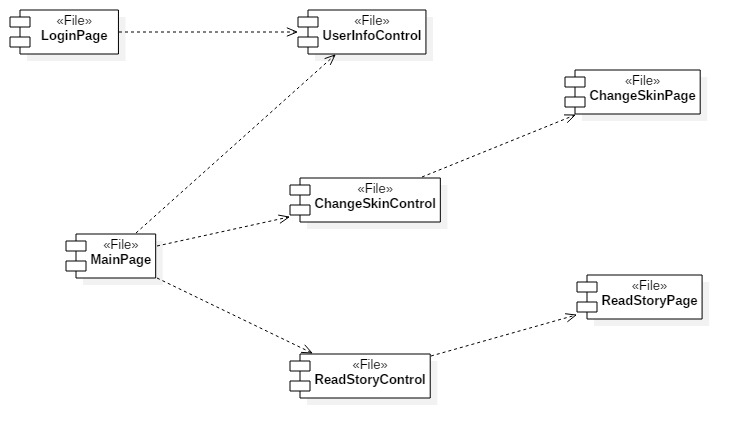


图3-10 用户接口子系统实现视图

1. 单人游戏子系统

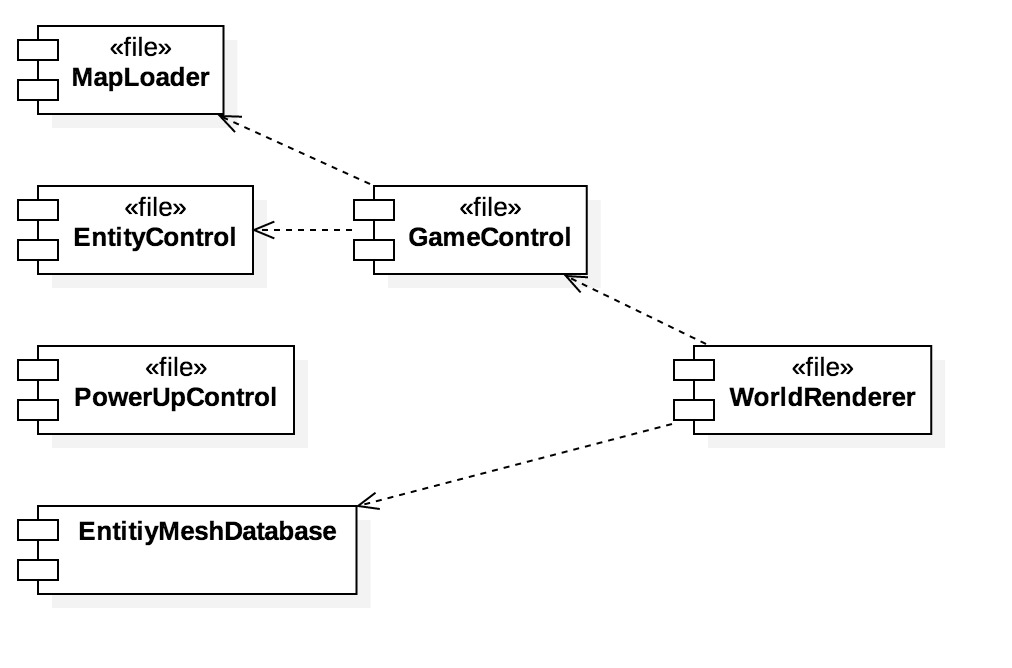


图3-11 单人游戏子系统实现视图

1. 多人游戏子系统

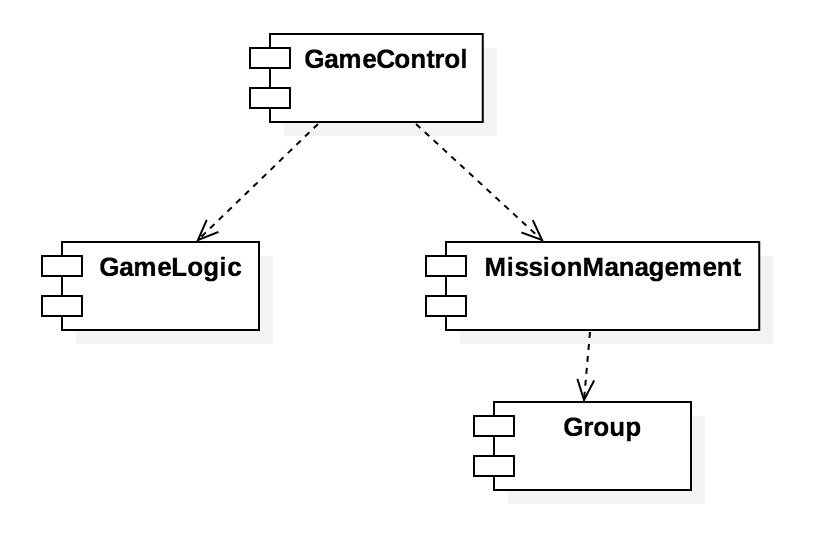


图3-12多人游戏子系统实现视图

1. 地图编辑器子系统

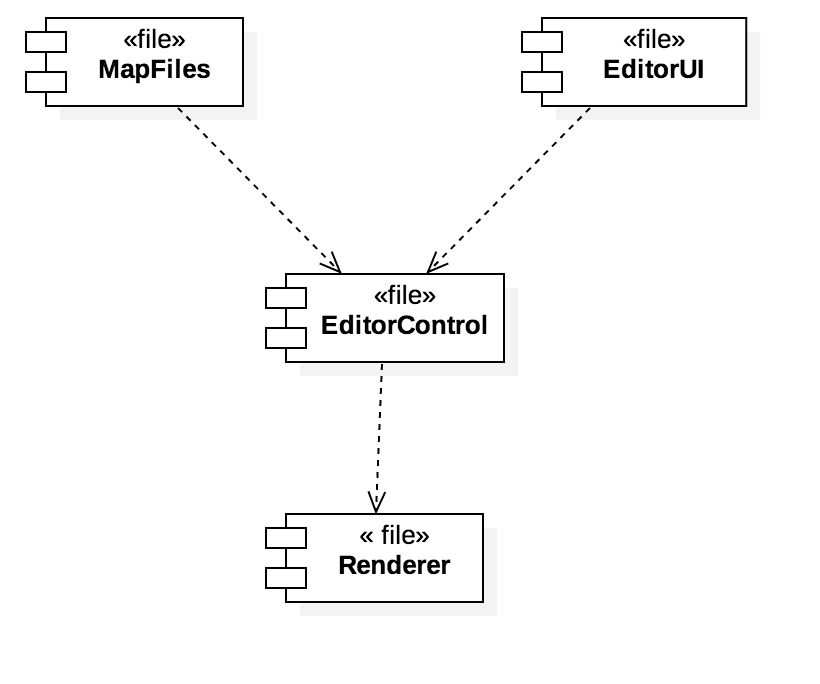


图3-13地图编辑器子系统实现视图

# 4. 进程视图

在客户端，为了保证用户体验，UI线程址负责边界对象与用户的交互。同时对每一个控制器单独启动一个线程。同时对于通信模块，也启动一个线程。这样的设计能够保证前台、功能实现、通信不会阻塞应用的运行。

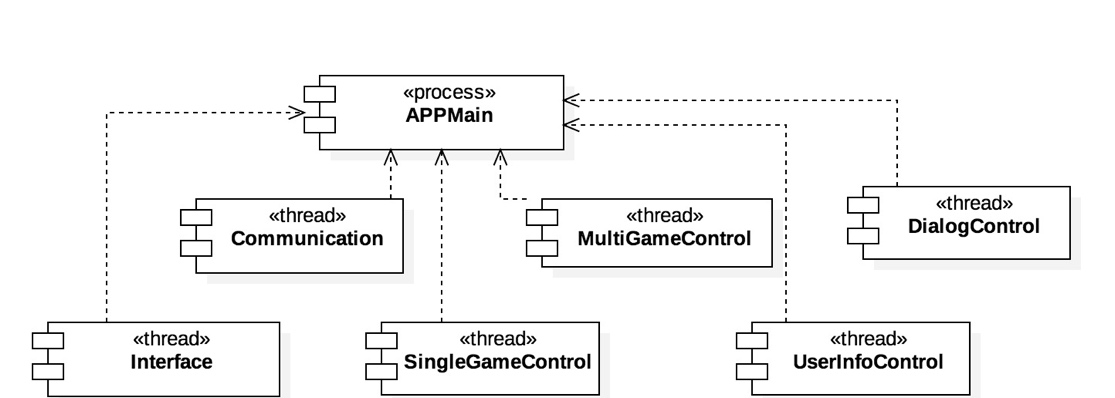


图3-14 进程视图

# 5. 部署视图

本系统是一个移动应用系统。因此客户端是手机上运行的App，后台通过数据库服务器进行数据存取。

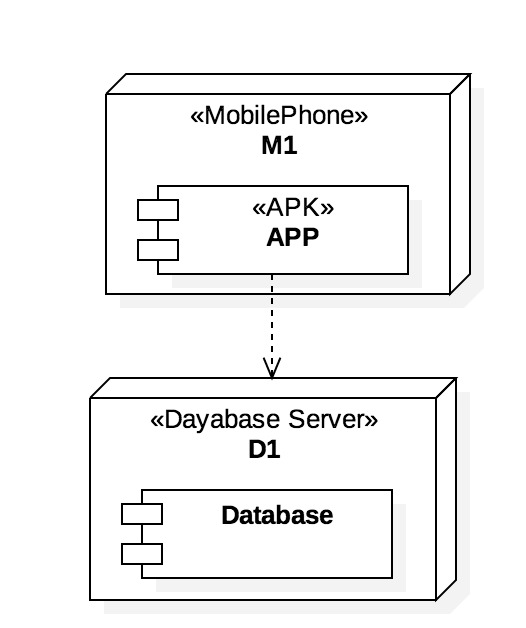


图3-15 部署视图