DAWN

Software Design Model

Version 1.0

**By:**

Dawn Group

2019-04

**Group Member:**

Zihan Xu

Yi Kuang

Chenyu Yang

Yuting Lan

Jianzhen Cao

**Document Language:**

English

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 2019-4-16 | 1.0 | Finish the 1st edition of Software Design Model | Zihan Xu, Yi Kuang, Chenyu Yang, Yuting Lan, Jianzhen Cao |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[1引言 1](#_Toc5983162)

[1.1编写目的 1](#_Toc5983163)

[1.2适用范围 1](#_Toc5983164)

[1.3定义 1](#_Toc5983165)

[1.4参考资料 1](#_Toc5983166)

[1.5概述 1](#_Toc5983167)

[2．用例视图 1](#_Toc5983168)

[3. 逻辑视图 2](#_Toc5983169)

[3.1系统结构 2](#_Toc5983170)

[3.2 Use-Case实现 2](#_Toc5983171)

[3.3 设计类图 2](#_Toc5983172)

[3.4 其他图 2](#_Toc5983173)

[4. 实现视图 2](#_Toc5983174)

[5. 进程视图 3](#_Toc5983175)

[6. 部署视图 3](#_Toc5983176)

Software design patten

# 1 Introduction

## Purpose xzh

说明：编写这份软件设计模型的目的，并指出预期的读者。

## 1.2适用范围 ky

列出此文档适用的软件应用程序，以及受到此文档影响的任何其他产品。

## 1.3定义 ky

列出本文件中用到的专门术语的定义和外文的首字母缩写词。可以引用项目词汇表来提供。

## 1.4参考资料 lyt

列出本文中各处引用的文档资料，包括每个文档的标题、文档编号、发表日期和出版单位并列出能够得到这些文件资料的来源。

## 1.5概述 cjz

说明本文件中其他各部分包含的内容，与本文件的内容组织方式。

# 2． 用例视图Use-case Diagram

尽管Use-Case主要是分析阶段的产物，但是将Use-Case图放在模型中，便于理解后续的设计。在此部分中，需要提供一张用例图。

可以补充活动图来说明业务逻辑。

# 3. 逻辑视图Logic Diagram

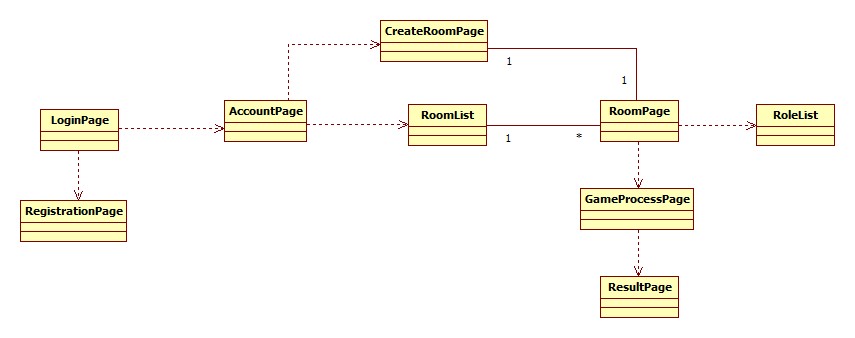
## 3.1 系统结构System structure

首先利用UML的包图，画出一个系统架构的表示图。

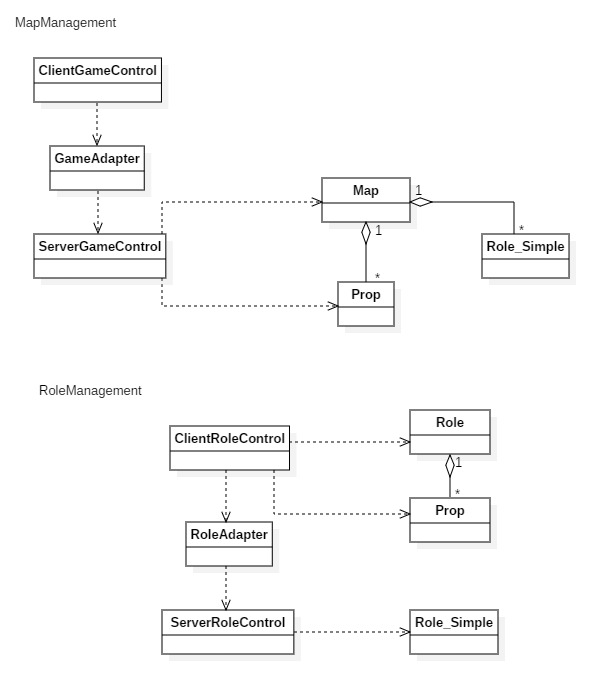
针对每一个包，画一张它所包含的类的类图。

xxxxxxxxxxxxxxxxxxx

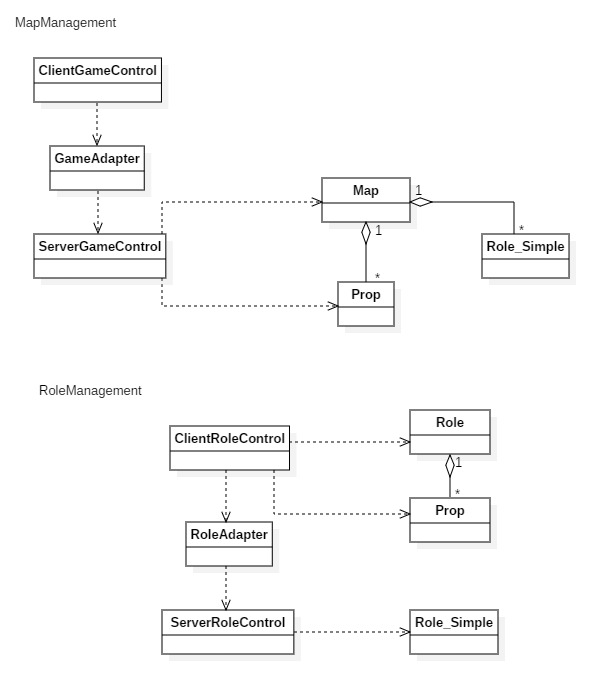
1. User-Interface Subsystem



1. User-management Subsystem
2. Role-management Subsystem



1. Map-management Subsystem



1. Common-service Subsystem

## 3.2 用例实现Use-Case Implementation

在该节中需要针对每一个Use-Case，通过交互图的方式表达相应的设计。因此，其格式可以表示为：

1. <Use-Case 1>实现

采用交互图的形式完整定义该Use-Case的实现过程。一个Use-Case中所有系统操作都必须得到相应的设计。如果一个用例比较复杂，可以分几个交互图进行表达。

1. <Use-Case 2>实现

……

1. <Register> Implementation

(2) <Login> Implementation

(3) <CreateRoom> Implementation

(4) <JoinRoom> Implementation

(5) <SelectRole> Implementation

(6) <Prepare> Implementation

(7) <GameProcess> Implementation

(8) <Move> Implementation

(9) <Attack> Implementation

(10) <Use> Implementation

t

(11) <EndOfGame> Implementation

(12) <ShowResults> Implementation

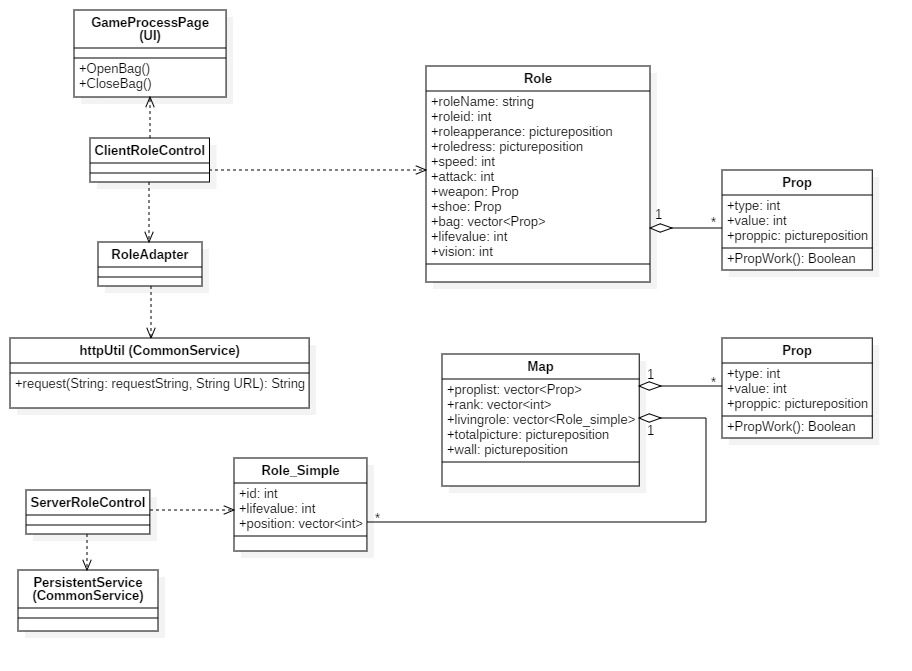
## 3.3 设计视图Design Class Diagrams

将Use-Case实现中涉及到的所有设计类以及它们的关联关系，画在一张设计类图中。如果类比较多，我们可以先画一个包图，然后画出各个包中包含的类图。

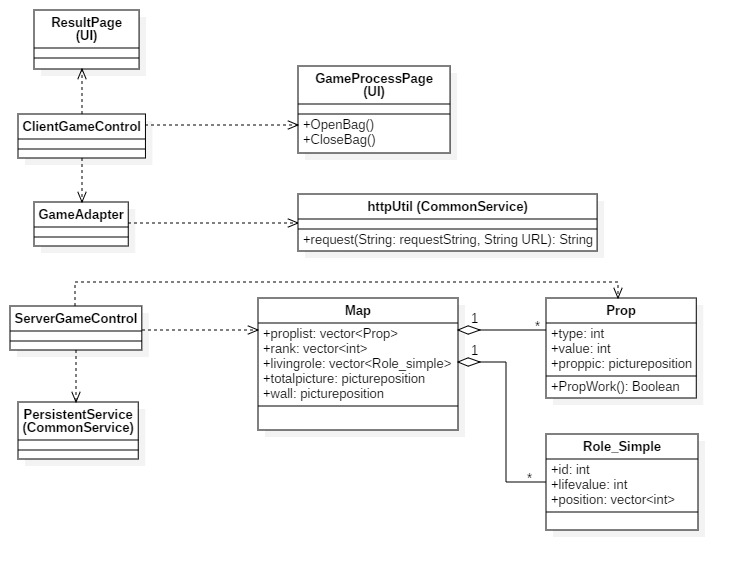
Since there too many classes, we decided to create a Package Diagram and then draw Class Diagrams on the basis of subsystems.

xxxxxxxxxxxxxxx

1. User-Interface Subsystem
2. User-management Subsystem
3. Role-management Subsystem



1. Map-management Subsystem



1. Common-service Subsystem

## 3.4 其他图

如果需要，针对某一个类给出其状态图。

如果需要，针对某一流程构造活动图。

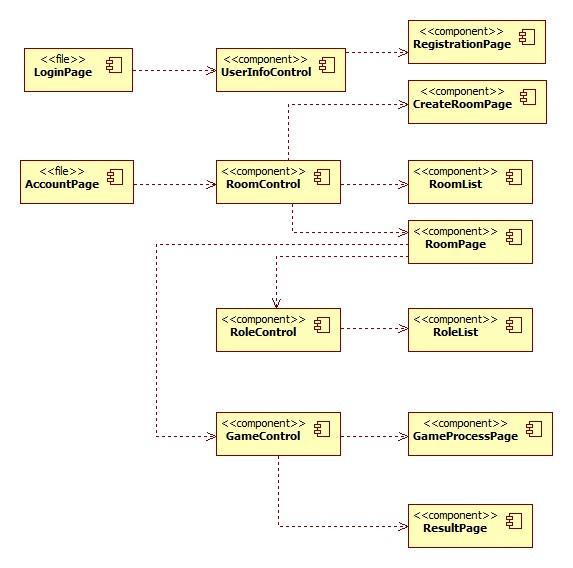
# 4. 实现视图Implementation Diagram

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：

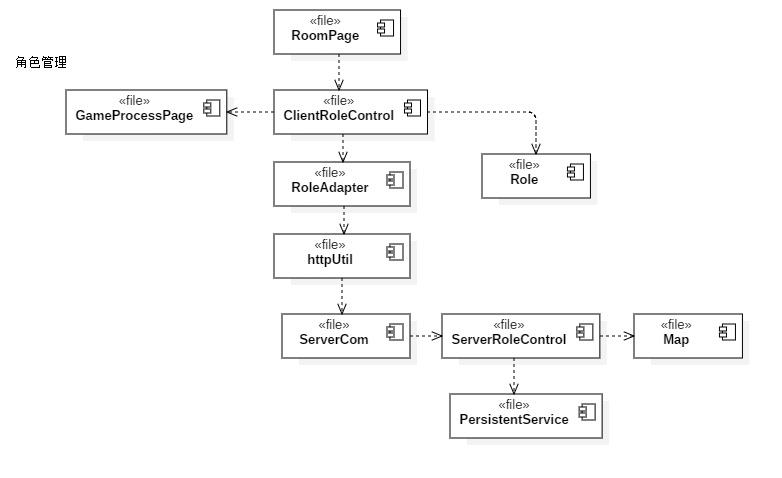
1. 一种为开发环境中的组件构成及依赖关系。开发环境中的组件指的是程序源文件以及它所依赖的其他文件。
2. 另一种为编译后生成的组件及其依赖关系。

组件内部可以画出它包含的类以及类的关系。

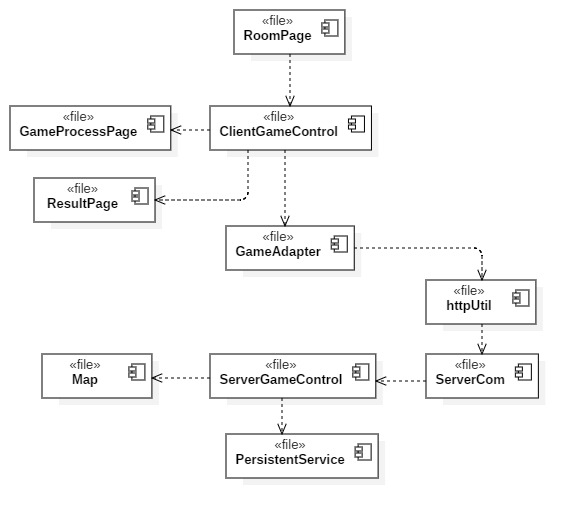
1. Development Model Composition
   1. User-Interface Subsystem



* 1. User-management Subsystem
  2. Role-management Subsystem



* 1. Map-management Subsystem



* 1. Common-service Subsystem

1. Post-compilation

# 5. 进程视图 Process View

通过类图或者组件图的形式表示进程类、线程类及其相互关系，通过类图表示进程、线程中包含的类。

Xxxxxxxxxxxx

Xxxxxxxxxxxx

# 6. 部署视图 Deployment View

画出系统的物理部署图。

Xxxxxxxxxxxx