DAWN

Software Architecture Document

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 Architecture Document | Zihan Xu, Yi Kuang, Chenyu Yang, Yuting Lan, Jianzhen Cao |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[1引言 Introduction 1](#_Toc6660234)

[1.1编写目的 Purpose For This Document. 1](#_Toc6660235)

[1.2适用范围 Domain 1](#_Toc6660236)

[1.3定义 Definition 1](#_Toc6660237)

[1.4参考资料 Reference 1](#_Toc6660238)

[1.5概述 Overview 2](#_Toc6660239)

[2 目前软件系统体系架构 Existing Software Architecture 2](#_Toc6660240)

[3 软件系统架构设计目标Software Architecture Design objectives 2](#_Toc6660241)

[4 建议的软件系统架构 Proposed Software Architecture 3](#_Toc6660242)

[4.1 概述 3](#_Toc6660243)

[4.2 用例视图 Use Case View 3](#_Toc6660244)

[4.3 系统逻辑视图 Logical View 4](#_Toc6660245)

[4.4系统运行视图 System Process View 16](#_Toc6660246)

[1. 4.5系统实现视图 System Implementation View 17](#_Toc6660247)

[4.6系统物理视图 Physical View 18](#_Toc6660248)

[4.7 Boundary Condition Design 19](#_Toc6660249)

[4.8数据管理设计 Data Management System 21](#_Toc6660250)

[4.9 其他设计 22](#_Toc6660251)

Software Architecture Document

# 1引言 Introduction

## 1.1编写目的 Purpose For This Document.

说明：编写这份软件架构设计的目的，并指出预期的读者。

This is a software architecture document. This document describes the system structure of the software. It is mainly expressed in various models in the design model. However, not all models in the design model are included in this document. This document generally covers use cases that have a significant impact on the architecture and introduces related models around them. No need to mention that this document should not only give the model diagram, but also provide a description of the reasons for making the appropriate design decisions.

## 1.2适用范围 Domain

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。

This document is written for our game DAWN.

To our best knowledge, this document will not affect any other products.

## 1.3定义 Definition

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。

All the glossaries are provided in the glossary document(词汇表\_en.docx).

## 1.4参考资料 Reference

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。

<<Object-Oriented Software Engineering Practice Guide-2>> Shanghai Jiao Tong University Press, 2016

<<Object-Oriented Software Engineering - Using UML, Patterns, and Java>> (3rd edition), Tsinghua University Press, 2011

## 1.5概述 Overview

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。

This document includes four parts: Introduction, Current System Architecture, System Architecture Design Objectives, and Recommended Software System Architecture. The current system section analyzes the current offline second-hand trading market and points out its shortcomings. The system architecture design goals are combined with software requirements to list the goals of the system design. The proposed software system architecture gives an explanation of the architecture and subsystems of the system, and displays the object design, hardware and software deployment, data management, software control, and boundary conditions of the system in a combination of textual representation and model diagram. The various parts of this document are closely related, complement each other and contrast, and present the software architecture of the system.

# 2 目前软件系统体系架构 Existing Software Architecture

对目前存在的系统进行描述，如果原来没有系统，则对相类似的系统进行描述。指出目前系统存在的问题，新系统希望得到的改进。

对原来系统架构的描述可以采用UML包图，或者部署图。

Unfortunately, we don’t have an existing software architecture to refer to.

# 3 软件系统架构设计目标Software Architecture Design objectives

本节说明对构架具有某种重要影响的软件需求和目标，例如：安全性、保密性、市售产品的使用、可移植性、分销和重复使用。还应记录可能适用的特殊约束：设计与实施策略、开发工具、团队结构、时间表、遗留代码等

The design goals of the system architecture are as follows:

1. Availability: If the system is unavailable when players enter the game, it will cause negative influence. Therefore, the system needs to ensure high availability.
2. Security: Important personal information is retained in the database, so the security of the system must be guaranteed.
3. High performance: The system responds in real time during operation and has a large flow rate, so it requires high performance.
4. Scalability: The system does not require relatively high-standard hardware when the initial scale is small. However, consideration should be given to the expansion of the system as the number of users increases.

# 4 建议的软件系统架构 Proposed Software Architecture

## 4.1 概述

对整个软件的架构进行概述，给出软件架构采用的架构模式及其选择理由。

描述软件架构中重用的框架、中间件和类库。

简略描述包含哪些子系统，每个子系统的功能是什么。

1）客户机／服务器模式（ ｃ ｌ ｉ ｅｎ ｔ ／ ｓ ｅ ｒ ｖ ｅ ｒ，Ｃ／Ｓ）

２）模型视图控制器模式（ ｍｏｄ ｅ ｌ，ｖ ｉ ｅｗａｎｄｃ ｏｎ ｔ ｒ ｏ ｌ ｌ ｅ ｒ，ＭＶＣ）

1. Client/Server

The client is App on users’ phone, including interaction with users and store of a little information in the game process. The server controls the entire game process and other use cases, such as registration, Login, joining room and so on.

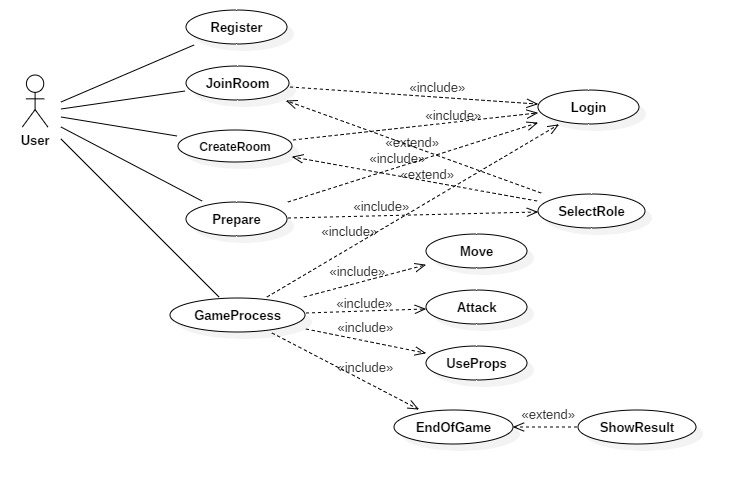
The reason for choosing this mode is that this mode is mature, and the method of separating the client and the server conforms to the environment of App development, and also facilitates the concentration and management of data.

1. Model, View and Controller, MVC

The system organizes subsystems in three levels. The model includes a dataset to store users’ account information. The view provides an interface (just UI). And the controller is to control and implement system functions, including UserManagement, RoomManagement, and RoleManagement & MapManagement during a game.

Most of this system is developed independently. A third-party library is used to support the implementation of interactive communication functions and animation display.

## 4.2 用例视图 Use Case View

本节选择列出所有用例中的某一些用例或场景，这些用例或场景应体现最终系统中重要的、核心的功能；或是在构架方面涉及范围很广（使用了许多构架元素）；或与构架中某一特别的设计有关系。

## 4.3 系统逻辑视图 Logical View

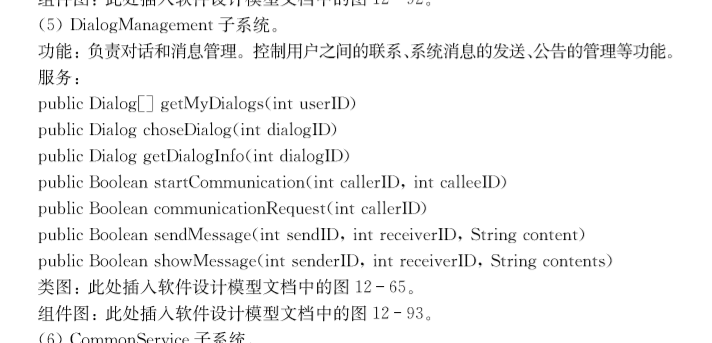
1. 系统架构

首先利用UML的包图，给出一个系统架构图，简要介绍该系统架构的特点，各个子系统的功能；

1. 子系统 Subsystems

针对每一个子系统描述其：

* 子系统的功能
* 子系统向外界提供的服务的详细描述（定义其接口具体形式）
* 给出每一个子系统的组件图、类图



1. 用例实现

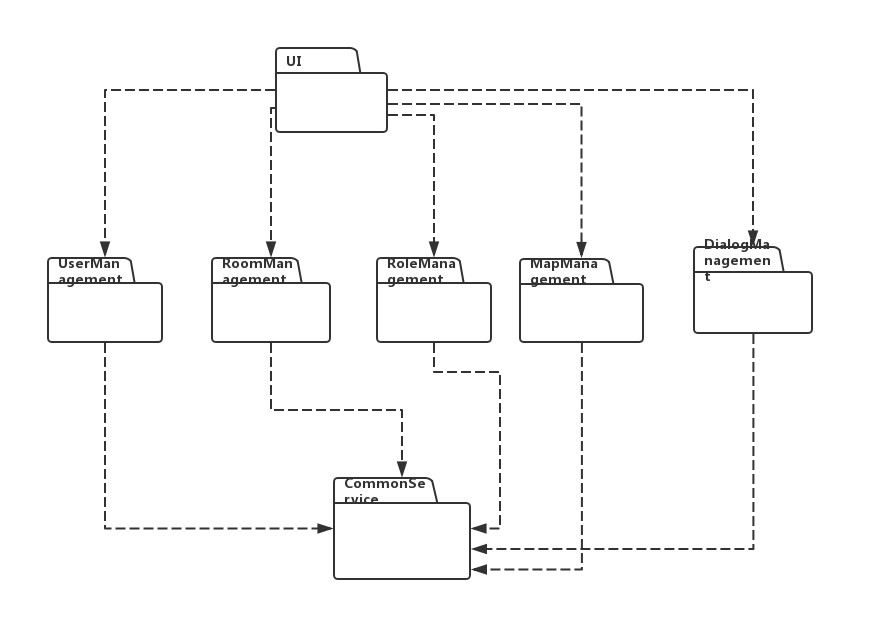
针对选择的每一个核心用例，用顺序图刻画其具体实现(copy)

1. 子系统协作

以交互图的形式围绕核心用例，刻画子系统之间的协作过程。此时，每一个子系统是一个黑盒子，在交互图中作为一个对象。

1. System Architecture

We have six subsystems: User Interface Subsystem, User Management Subsystem, Room Management Subsystem, Role Management Subsystem, Map Management Subsystem, Common Service Subsystem



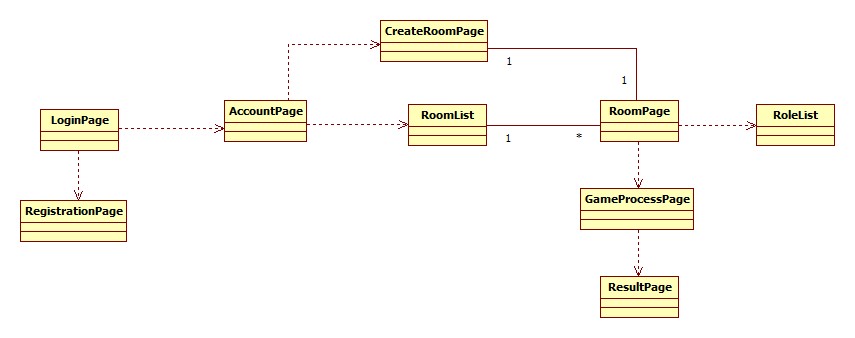
1. Subsystem

1) User Interface Subsystem

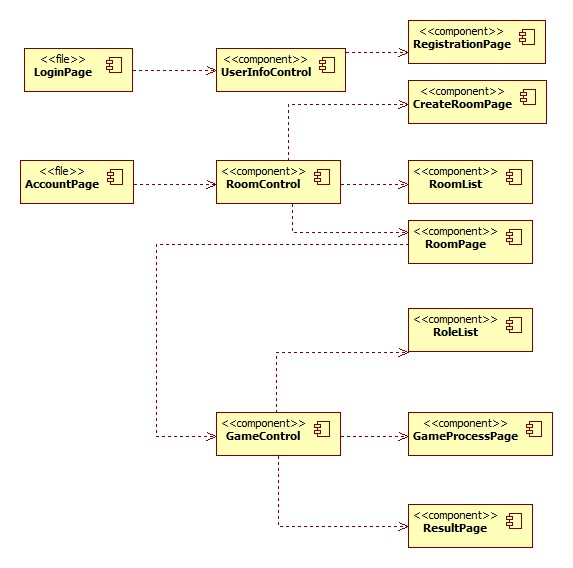
**Function**: Interact with user.

**Service**: The UI layer provides kinds of input boxes, buttons and text boxes to show information to user and interact with user.

**Class Diagram:**



**Component Diagram:**



2) User Management Subsystem

**Function: Provide management service for user management, including registering, login and so on.**

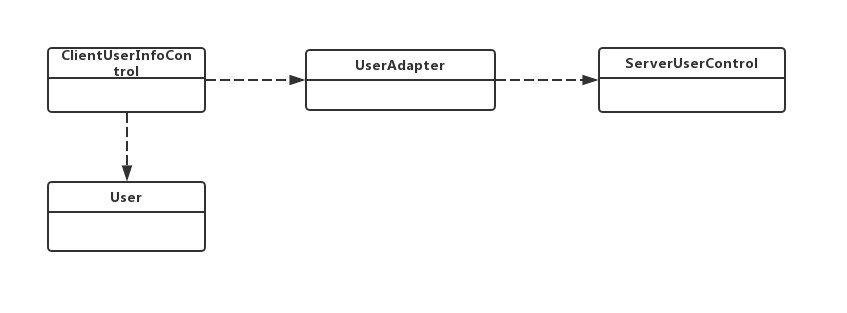
**Service:**

public Boolean summitRegister(list personalInfomation)

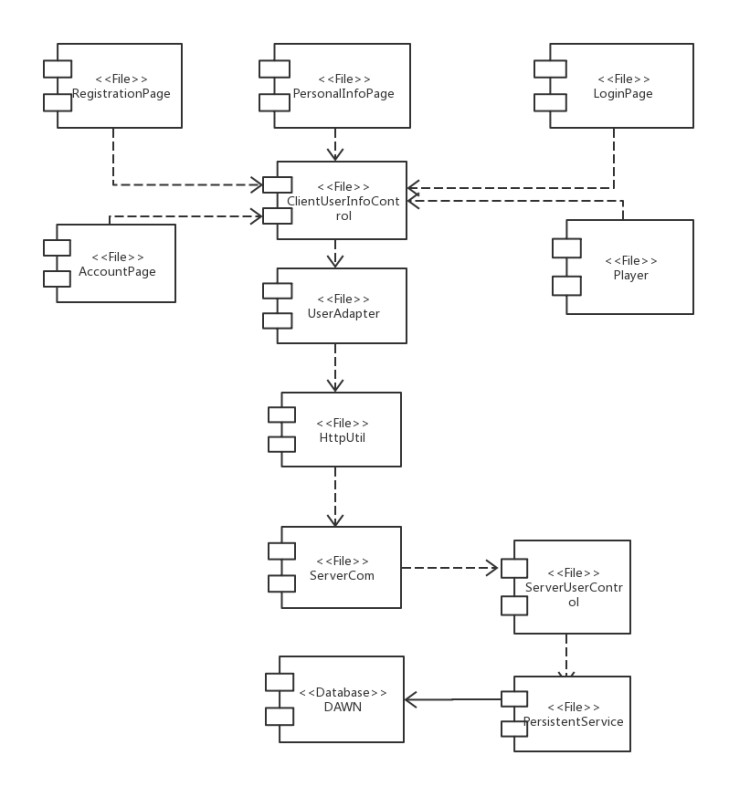
public Boolean submitLogin(String userID, String password)

public list showUserInformation(String UserID)

**Class Diagram:**

****

**Component Diagram:**



3) Room Management Subsystem

**Function**: Manage the creation of a room and choose the room to enter.

**Service**:

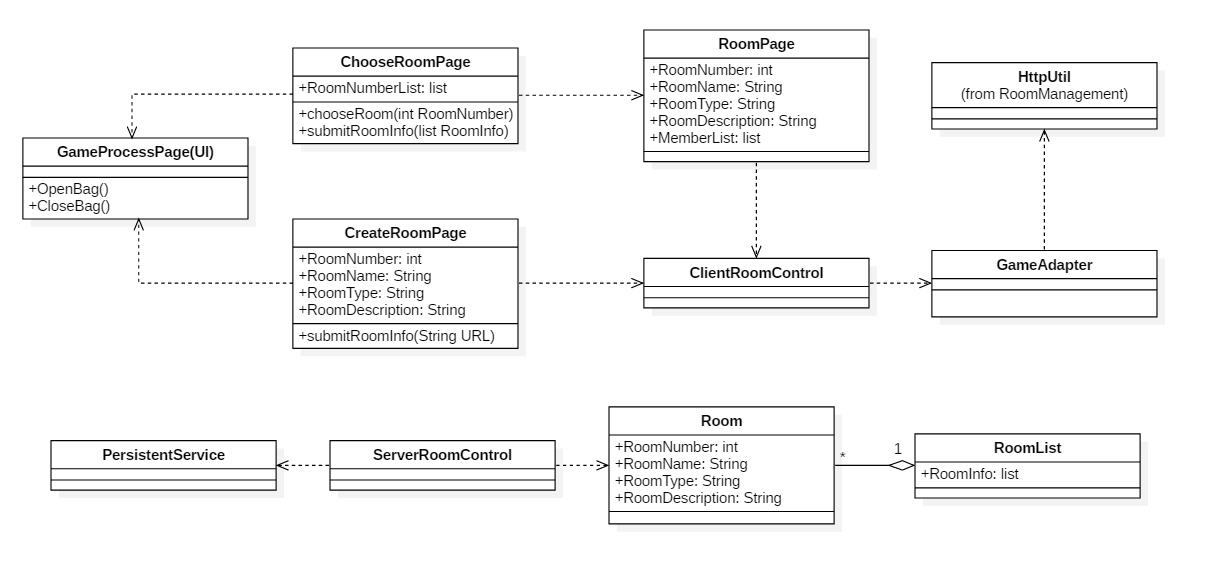
public Boolean chooseRoom(Room chosenRoom)

public Room createRoom(list roomInfomation)

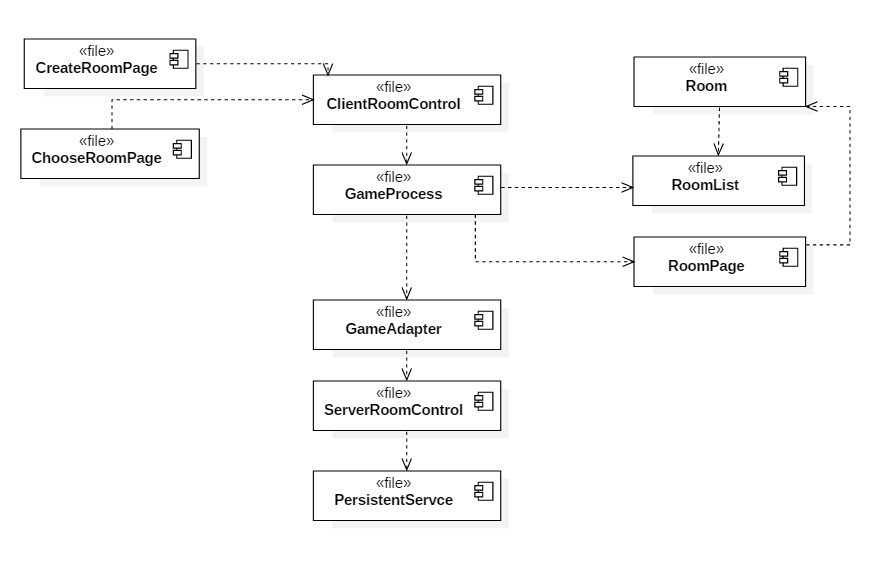
public list requestRoomList(Boolean showAllRoom)

public Boolean joinRoom(Room chosenRoom)

**Class Diagram:**

****

**Component Diagram:**

****

4) Role Management Subsystem

**Function:** Provide management service for role management, including creating and changing a role.

**Service:**

public Role RoleCreate(Player p)

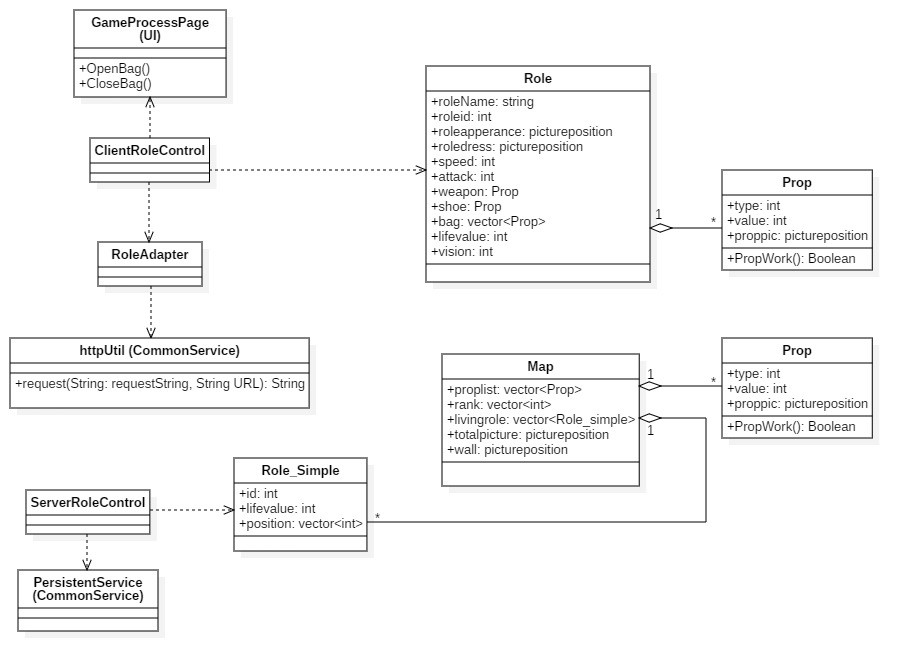
public Boolean makeattack(int direction, int[] position)

public Boolean addprop(Prop p)

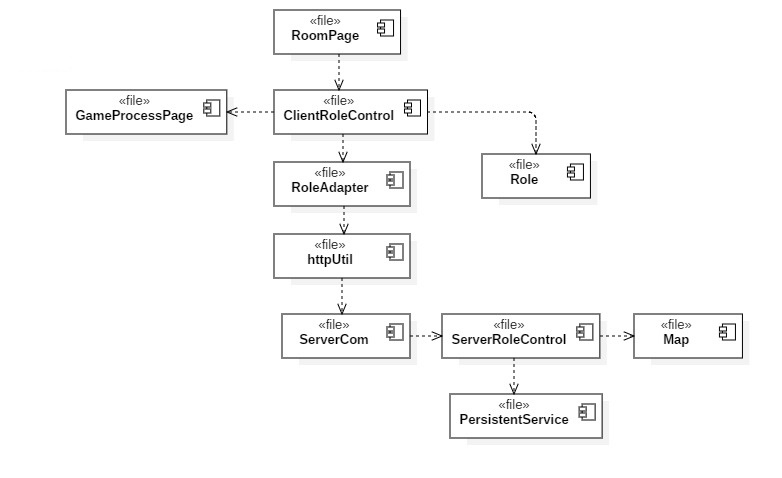
public Boolean useprop(Prop p)

public Boolean SubmitLifevalue(int ID, int value)

**Class Diagram:**

****

**Component Diagram:**



5) Map Management Subsystem

**Function:** Providemanagement service for maps, including creating and movements in the map.

**Service:**

public Map MapCreate(Room r)

public void move(int direction, int[] position, int speed)

public Boolean pickup(Prop[] plist, int[] position)

public Boolean submitdamage(int targetID, int value)

public Map getmapdata()

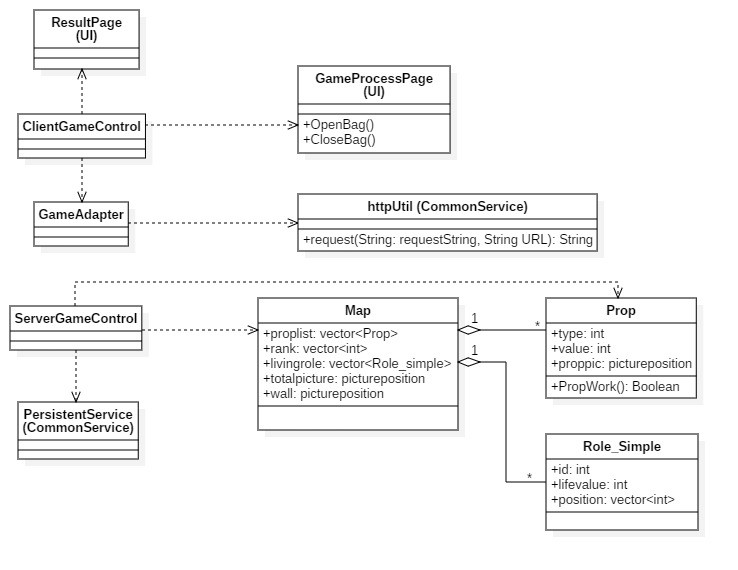
public void InfoUpdate(Map m)

public void RankRecord()

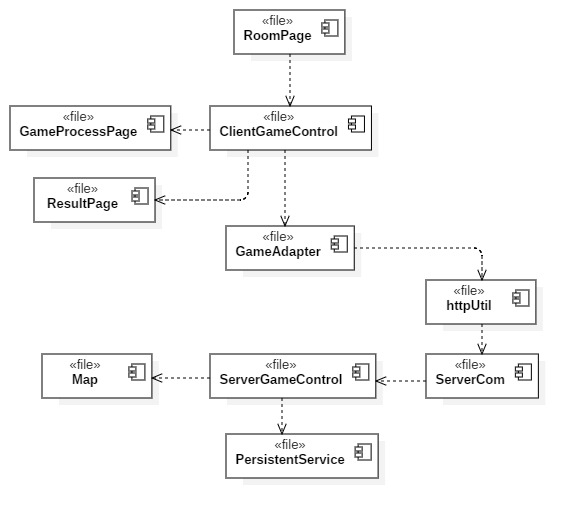
public Boolean SendResult()

public void ShowResult()

**Class Diagram:**

****

**Component Diagram:**



6) Common Service Subsystem

**Function:** Provide data access management, client and server communication functions, mainly room, game, user information.

**Service:**

public String request（String requestString，String url）

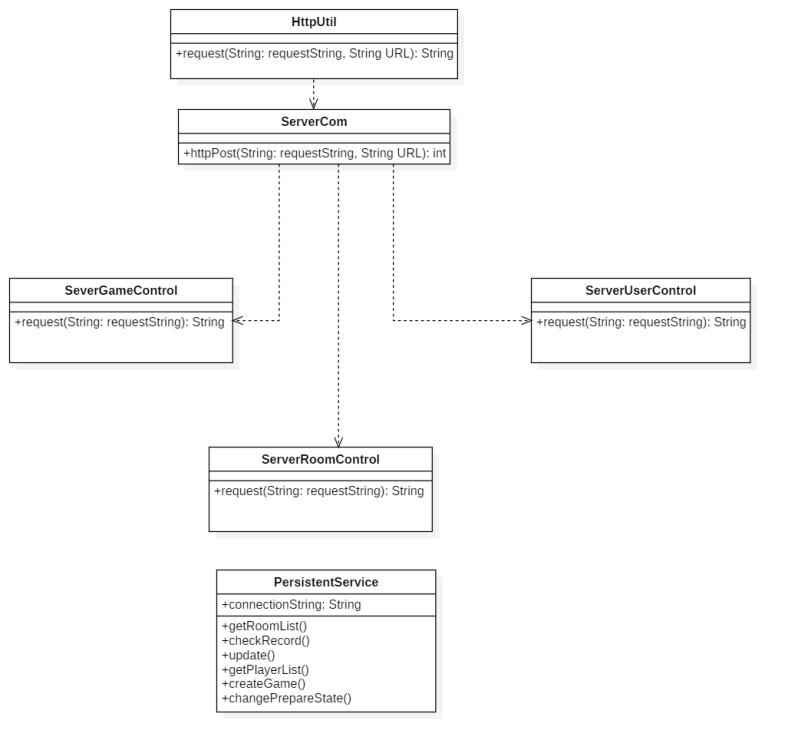
public Boolean ChangeRoleState(int roomID, int roleID)

public Boolean ChangePrepareState(int roomID)

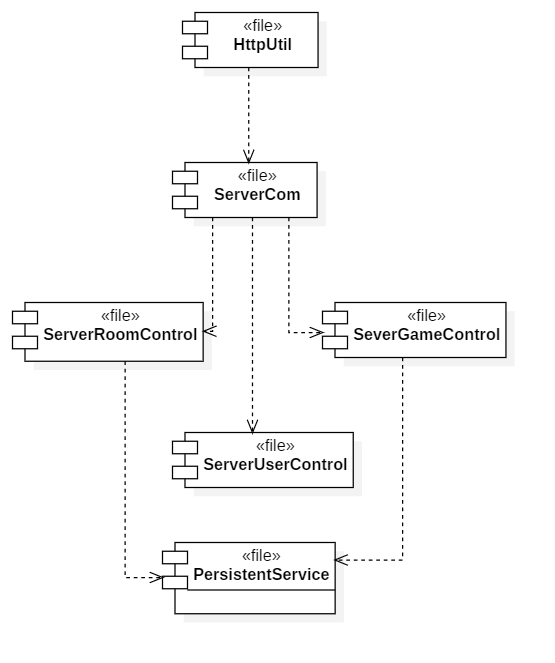
public Boolean UpdateUserInfo(int AccountID,String contents)

public Boolean ChangeGameState(int roomID,String contents)

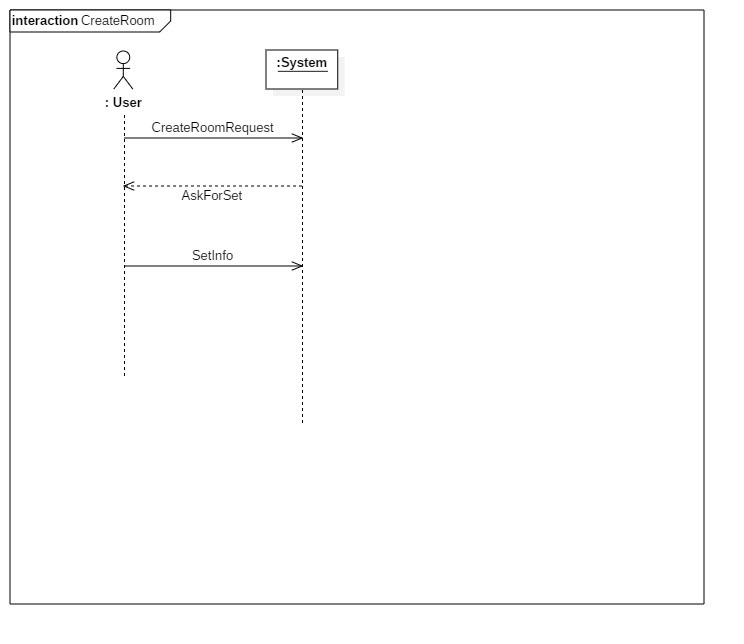
**Class Diagram:**

****

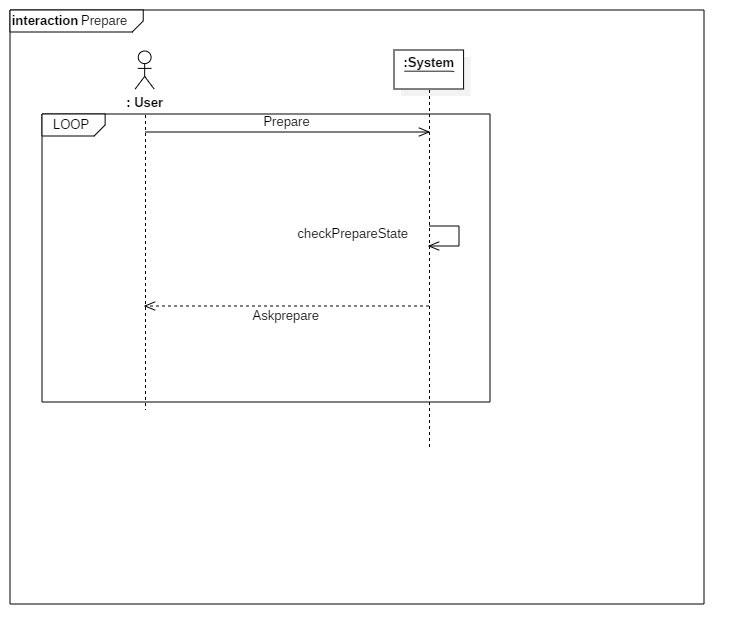
**Component Diagram:**

****

1. Use Case Implementation
2. CreateRoom



1. Prepare

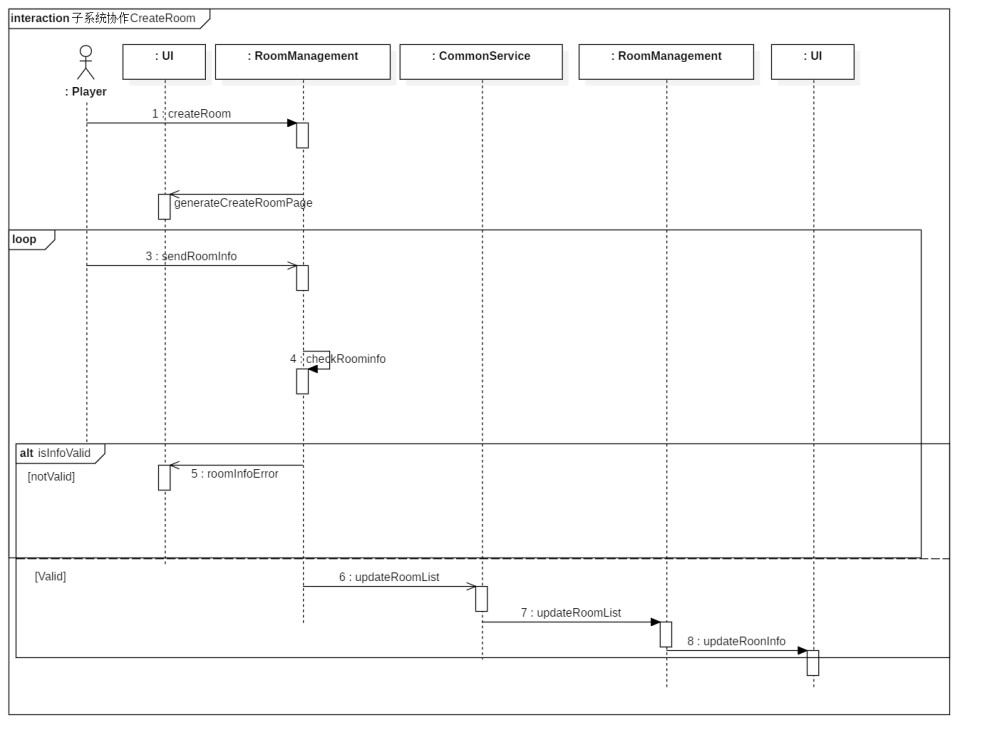


3) GameProcess

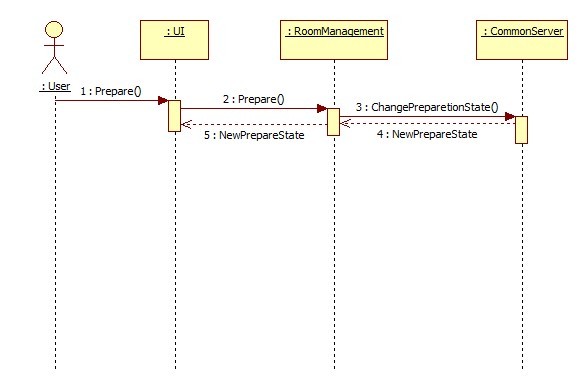


4) ShowResults

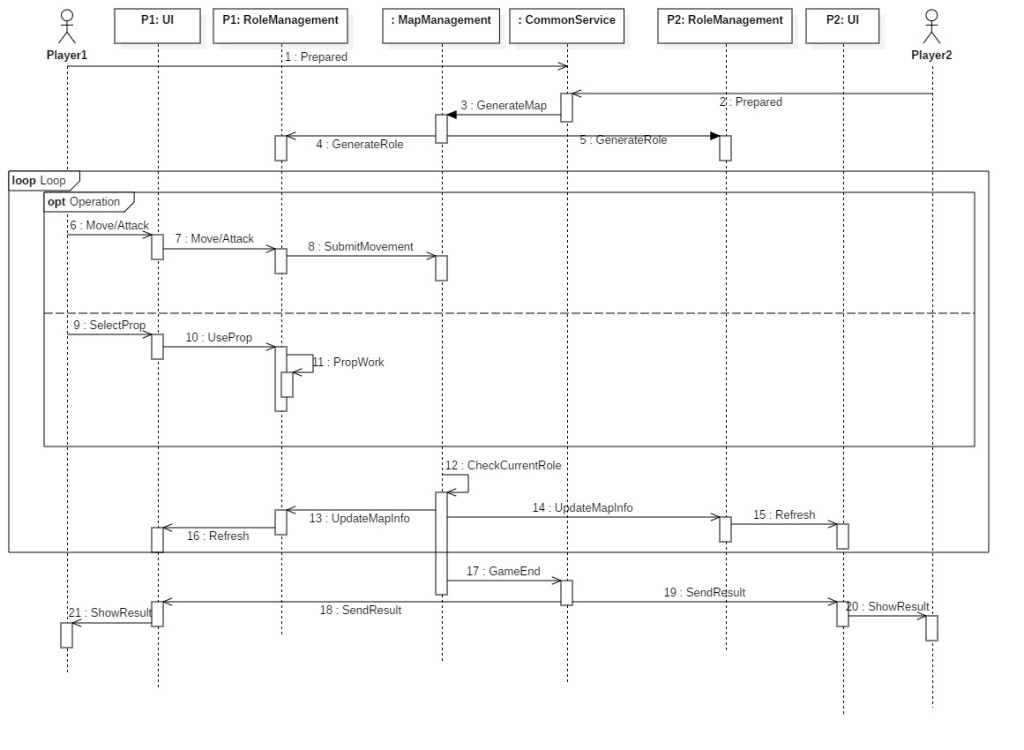
1. Subsystem Collaboration  
   1) CreateRoom



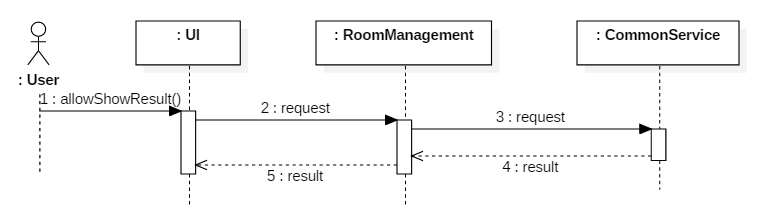
2) Prepare



3) GameProcess



4) ShowResults



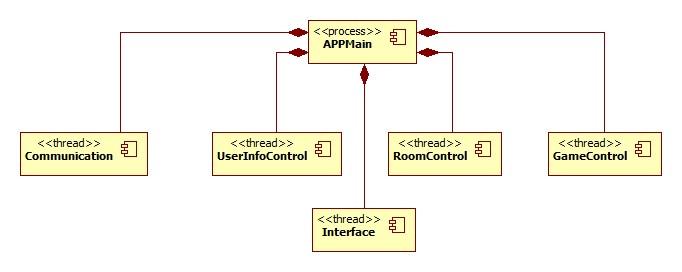
## 4.4系统运行视图 System Process View

该部分首先描述系统的控制流结构的选择，例如是采用数据流驱动、事件驱动，还是采用多线程。

如果软件中涉及到并发的情形，提供相应的进程视图给以相应的解释。

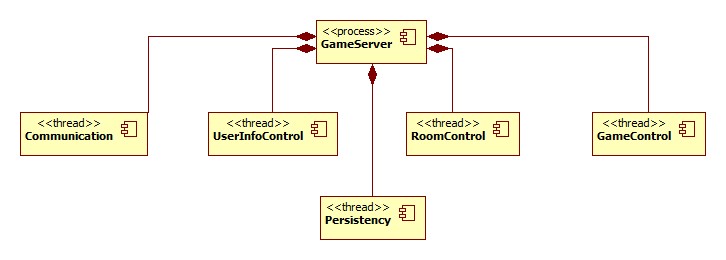
1. Client process diagram

On the client side, to provide the user with the best experience, UI thread only deal with the interaction between boundary objects and user. Each controller owns an exclusive thread, the same with communication module. Such design can ensure that the front end, function implementation and communication will not be block.



1. Server process diagram

In the design of server process, for the communication part, in response to the request of each customer, multithread management will be performed according to the concurrency of request. At the same time, each controller has their own thread. For persistent service, we will multi-threading mechanism to ensure its responsiveness.



## 4.5系统实现视图 System Implementation View

该部分分两部分。

1. 系统开发模型：首先说明开发环境、开发语言、版本控制工具的选择，在开发环境中一个项目包含哪些目录，包含哪些源文件，用组件图来说明源文件包含的类，以及源文件之间的依赖关系。
2. 系统实现模型：定义编译后的可执行文件及其依赖关系。通过组件图来表示可执行文件及其依赖关系。

1) System development environment

Development environment: Eclipse, AndroidStudio.

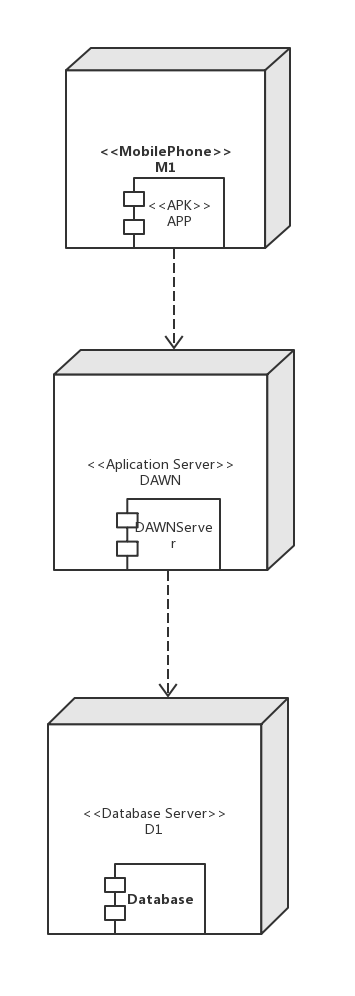
Development language: Java.

2) System development model

In this system, the corresponding software file definition is performed according to the component diagram. Therefore, the system development model is consistent with the component diagram in the software design model.

## 4.6系统物理视图 Physical View

利用UML的部署图描述系统的软硬件部署方式。列出硬件配置的规格要求以及选择理由。



Hardware configuration requirements

(1) user machine: a phone with at least 1.5GHz dual core processor and 2GB memory. To have good user experience, this is the minimum configuration to run most of the android game smoothly.

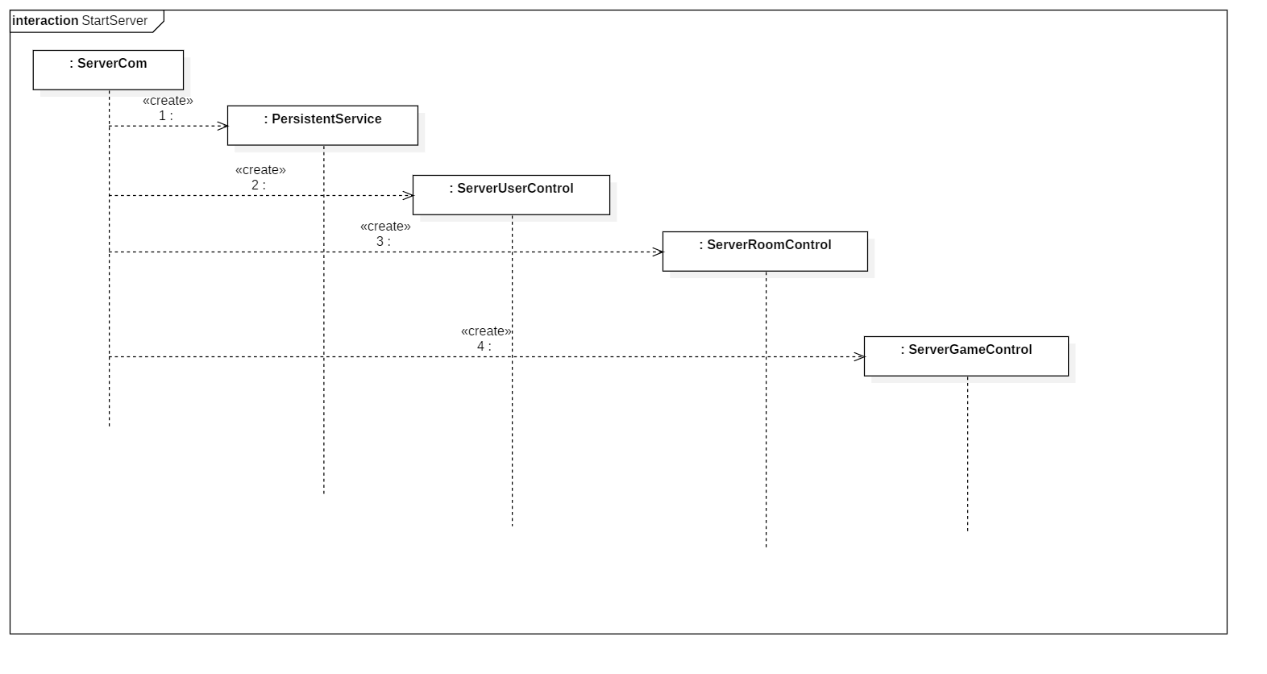
(2) Application Server: One or more Server with dual core i series processor, 4GB memory and 20GB storage. Considering the limited number of user, a laptop with similar or better configuration is enough.

(3) Network: 50Mbps internetwork. There are plenty of data exchange during a game, which means larger bandwidth is needed.

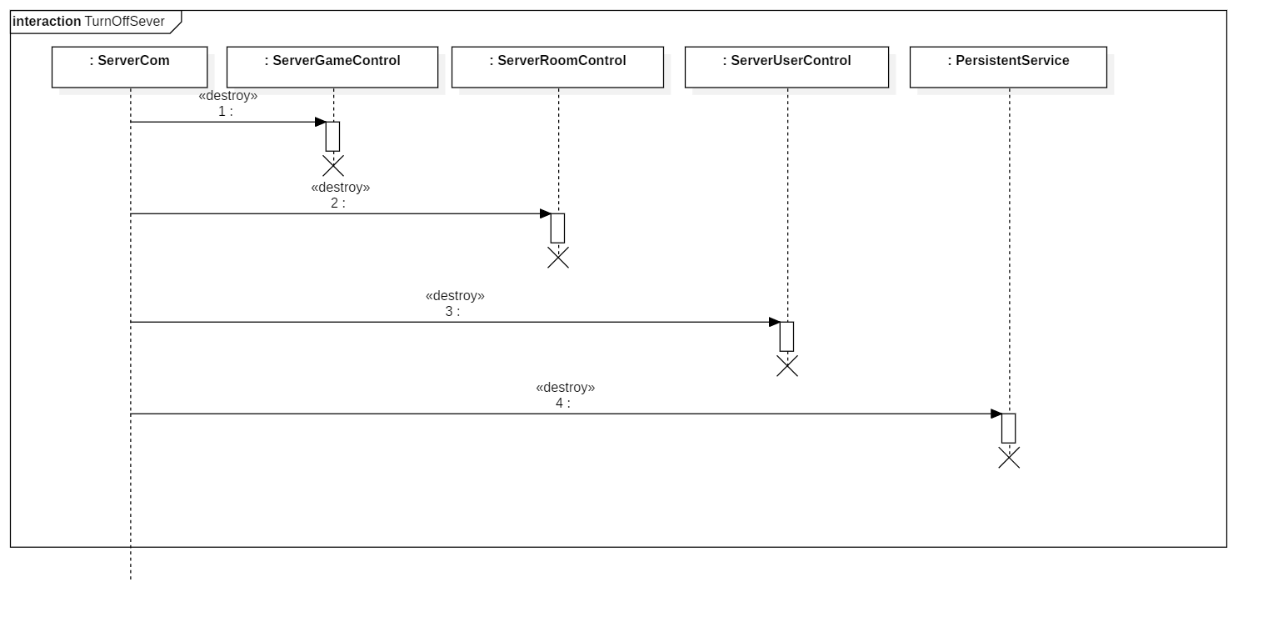
## 4.7 Boundary Condition Design

描述了系统中如何启动、关闭以及进行错误处理。描述的方式是采用用例的方式来进行说明，即提供启动用例、关闭用例、错误处理用例的分别说明，并通过交互图来说明这些用例是如何实现的。

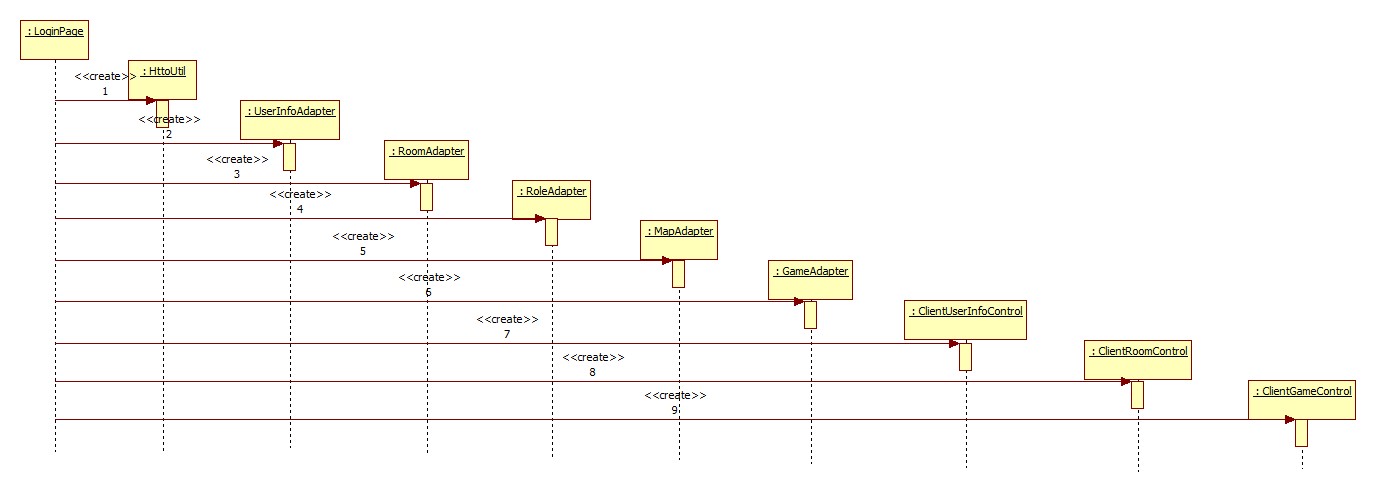
1. Start Up Server



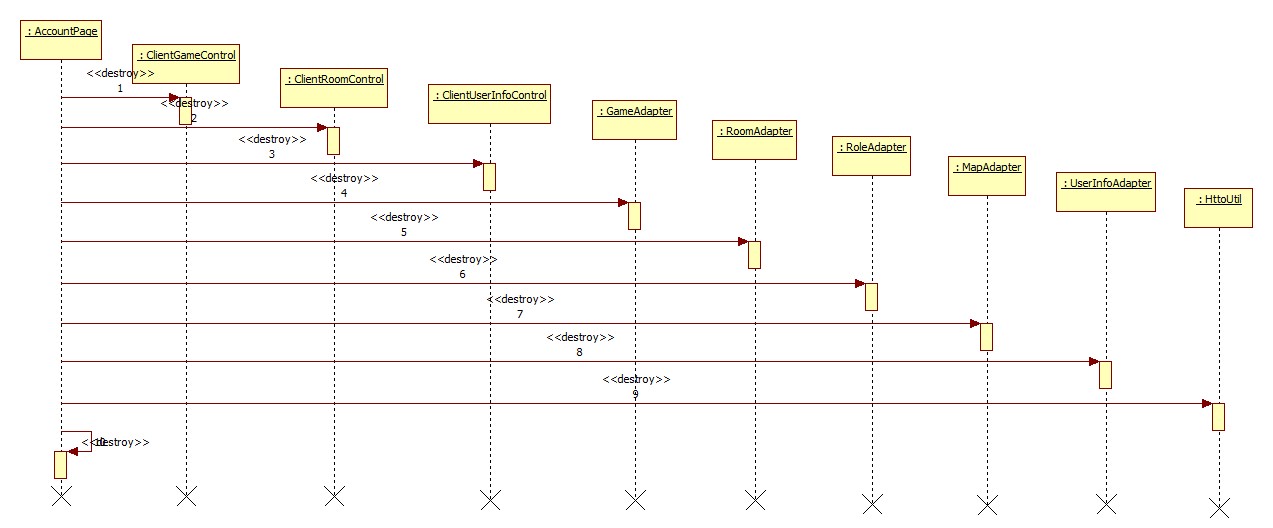
1. The user clicks on the app icon on the server
2. Start the ServerCom object.
3. Start the PersistentService PersistentService object
4. Start the SeverUserControl, SeverRoomControl , ServerGameControl object in turn, and pass the reference to the ServerCom object and the PersistentService object to these objects.
5. Shut Down Server



1. User clicks the app's close button
2. The system delete SeverGameControl, SeverRoomControl , ServerUserControl object in turn
3. The system disconnects from the database and deletes the PersistentService object.
4. The system deletes the ServerCom object.
5. Start Up App



1. The user clicks the APP icon.
2. The system creates LoginPage object.
3. LoginPage object create HttpUtil object.
4. LoginPage object create UserInfoAdapter, MapAdapter, RoleAdapter, RoomAdapter and GameAdapter object and post HiipUtil object to them.
5. LoginPage object create ClientUserInfoControl, ClientRoomControl and ClientGameControl object and post the reference of UserInfoAdapter, MapAdapter, RoleAdapter, RoomAdapter and GameAdapter object to them individually.
6. Shut Down App



1. Shut down APP on phone.
2. Shut down all the ClientControl objects in turn.
3. Shut down all the Adapter objects in turn.
4. Shut down HttpUtil object.
5. Shut down AccountPage.
6. Abnormality
   1. When system goes wrong, we first need to determine where the abnormality comes from.
   2. If the problem came from server, we will have to logout all users and restart the server; if the problem came from app, we will only need to restart the app.
   3. The database itself maintains its correctness by transactional features.

## 4.8数据管理设计 Data Management System

采用UML的类图表示那些需要持久化的数据。

说明选择的数据持久化保存的方式及理由。

如果是数据库保存，确定所选的数据库管理系统，并给出数据库的表设计，如果是文件系统，说明文件的目录结构，文件的格式。

Considering that the data in the game is temporary, when a game comes to an end, data about this game in no longer needed. only the user information needs to be stored persistently.

This system has a large amount of user information and needs to be updated frequently. So we use a database to store user information and room lists.

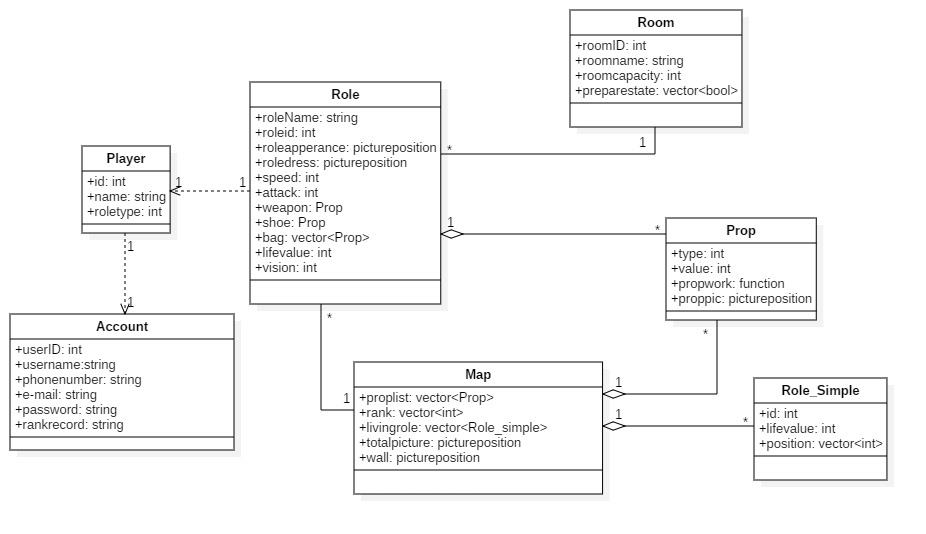
1. Account List

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No. | Field | Description | Type | Empty | Key | Unit | Remark |
| 1 | userID | user’s ID | int | N | Y |  |  |
| 2 | password | user’s password | String | N | N |  |  |
| 3 | name | user’s name | String | N | N |  |  |
| 4 | PhoneNumber | user’s phone | String | N | N |  |  |
| 5 | e-mail | user’s e-mail | String | Y | N |  |  |
| 6 | RankRecord | user’s rankrecord | String | N | N |  |  |

1. Room List

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No. | Field | Description | Type | Empty | Key | Unit | Remark |
| 1 | roomID | room’s ID | int | N | Y |  |  |
| 2 | roomName | room’s name | String | N | N |  |  |
| 3 | roomCapacity | the maximum of players in the room | int | N | N |  |  |

The entity class diagram:



## 4.9 其他设计

列出针对设计目标提出的特别的设计考虑，比如说访问控制和安全，可靠性设计等。

（1）访问控制和安全设计

采用表格列出不同的操作者对不同对象的权限；

描述用户认证的方式；

如果需要，给出数据的加/解密方式、给出接口调用的安全认证；

以及其他安全问题。

（2）可靠性设计

如果有特定的可靠性要求，给出在可靠性方面采取的特定设计方案。

(1) Access Control and Security Design

Use tables to list the permissions of different operators on different objects;

Describe the way user authentication is performed;

If necessary, give the data encryption/decryption method and give the security authentication of the interface call;

And other security issues.

(2) Anti-plug Design

Check each player’s relevant data especially overly outstanding players.

(3) Reliability Design

If there are specific reliability requirements, give a specific design approach to reliability.