|  |  |  |
| --- | --- | --- |
|  | 第10组 大保天天健 | |
| 组员： | 陈俭军  陈鹏  蒋秉良  焦紫顺  李淳雨  李豪俊 | 121250010  121250013  121250058  121250063  121250066  121250067 |

|  |
| --- |
|  |
| 龙珠消消乐 软件详细设计文档 |
|  |

**目录**

[1. 引言 3](#_Toc390018665)

[1.1 编制目的 3](#_Toc390018666)

[1.2 对象与范围 3](#_Toc390018667)

[1.3 参考文献 3](#_Toc390018668)

[1.4 文档更新记录 3](#_Toc390018669)

[2. 产品概述 3](#_Toc390018670)

[3. 体系结构模型 4](#_Toc390018671)

[3.1 整体架构描述 4](#_Toc390018672)

[3.1.1 逻辑视角 4](#_Toc390018673)

[3.1.2 架构设计 4](#_Toc390018674)

[3.1.3 物理部署 6](#_Toc390018675)

[3.2 客户端详细设计 7](#_Toc390018676)

[3.2.1 View模块的分解 7](#_Toc390018677)

[3.2.1.1 view模块的职责 7](#_Toc390018678)

[3.2.1.2 view模块的接口规范 8](#_Toc390018679)

[3.2.1.3 view模块的设计原理 9](#_Toc390018680)

[3.2.2 controller模块的分解 10](#_Toc390018681)

[3.2.2.1controller模块的职责 10](#_Toc390018682)

[3.2.2.2 controller模块的接口规范 10](#_Toc390018683)

[3.2.2.3启动模块的设计原理 12](#_Toc390018684)

[3.2.3 model模块的分解 13](#_Toc390018685)

[3.2.3.1 model模块的职责 13](#_Toc390018686)

[3.2.3.2 model模块的接口规范 13](#_Toc390018687)

[3.2.3.3 启动模块的设计原理 19](#_Toc390018688)

[3.3 界面模块的设计 19](#_Toc390018689)

[4. 系统体系结构设计思路 19](#_Toc390018690)

[5. 依赖视角 19](#_Toc390018691)

1. 引言

## 编制目的

本文档给出龙珠消消乐游戏的详细设计描述和模型图，在软件架构的基础上进行了进一步分析，为系统的维护、扩展及后续开发提供参考依据。

## 对象与范围

本报告详细完成对龙珠消消乐游戏的详细设计，达到指导后续软件构造的目的。

本文档的读者是大保天天健团队内部的开发工程师，面向开发人员和最终用户编写，是了解系统的导航。

## 参考文献

1）IEEE标准

2）《龙珠消消乐需求规格说明文档》

3）《龙珠消消乐软件架构设计文档》

## 文档更新记录

**表1.1 文档更新记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **作者** | **版本描述** | **日期** |
| 1.0 | 焦紫顺 | 详细设计初稿 | 2014/5/1 |
| 1.1 | 李豪俊 | 界面设计添加 | 2014/5/15 |
| 1.2 | 李淳雨 | 包图修改 | 2014/6/1 |

# 产品概述

游戏架设在著名漫画《七龙珠》故事背景下，为一个少年通过操控消除游戏实现修炼、收集龙珠以及天下第一武道会比武的过程（分别对应单人、协作、对战）。区别于一般消除类游戏缺乏故事情节性的特点，本游戏拥有较为完整的故事背景，游戏过程具有连贯性。用户在游戏过程中，可以充分享受游戏的情节趣味。

# 体系结构模型

## 整体架构描述

### 逻辑视角

龙珠消消乐游戏中，选择了MVC体系结构风格，将系统分为3个模块（视图、模型、控制器）能够更好地示意整个高层抽象。

（1）视图：用于前台界面展示和配置

（2）控制器：处理视图的数据，并向模型发送数据

（3）模型：包含业务控制和逻辑，定义和存储系统中相关数据

MVC体系结构的逻辑视角和逻辑设计方案如图×所示。

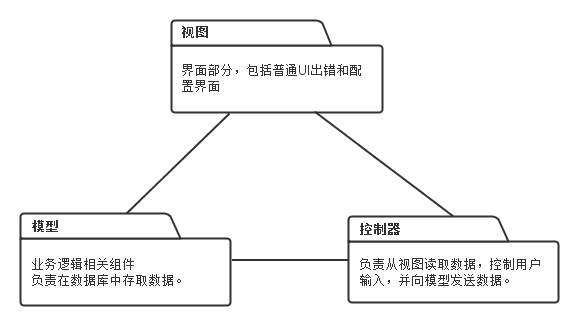
****

图3.1.1 逻辑视角设计

### 架构设计

系统架构中的对象分为7类

1. UI对象，负责处理系统数据的实现和用户的交互
2. UIService,负责提供视图的抽象接口，改变视图状态
3. IController对象，控制器负责获取用户输入，并调用IService模块的服务
4. IService对象，负责提供服务的抽象接口，获取从数据端组装后的数据
5. ServiceImp对象，负责对于抽象接口的实现模块
6. IDTO对象，负责封装从IDAO获取的批量数据的接口
7. IDAO对象，负责与数据库实体交互，获取数据
8. Entity对象，该模块用来将从数据库中获取的数据封装成数据实体

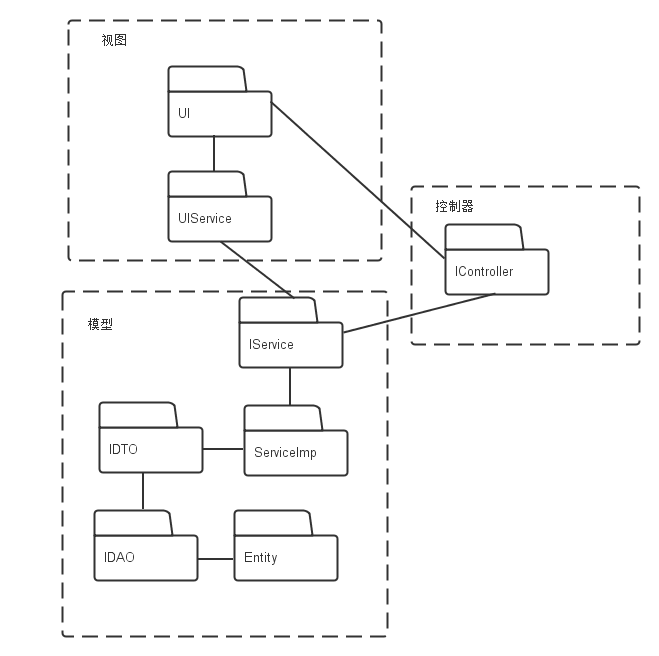


图3.1.2 架构设计包图

### 物理部署

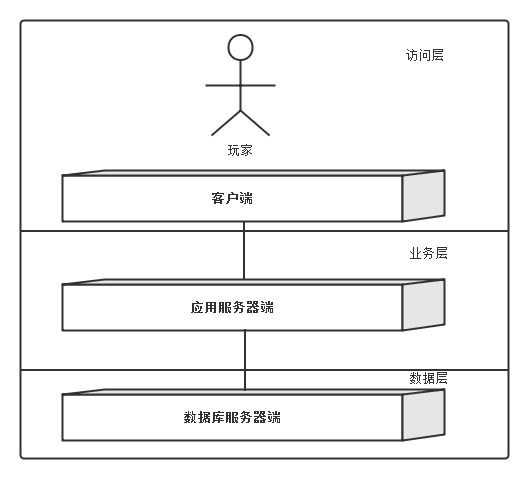
****

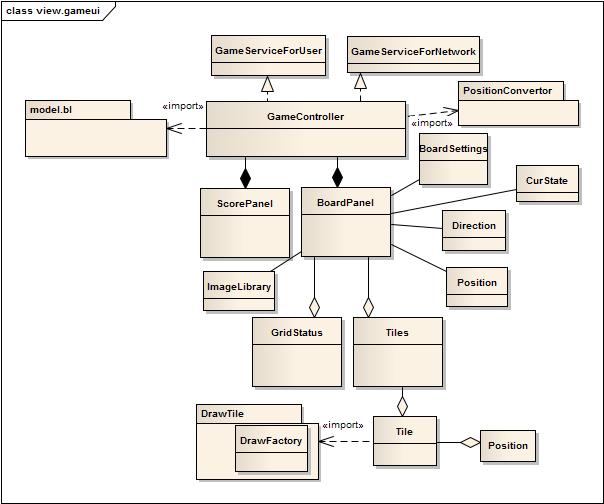
图3.1.3 物理部署示意图

游戏系统可以部署在以下3个物理层次。

1. 访问层：用于用户访问系统的层次
2. 业务层：部署业务控制和逻辑的层次
3. 数据层：部署和存储系统中相关数据的层次

## 客户端详细设计

### View模块的分解



**图3.2.1 View.gameui 详细类图**

#### 3.2.1.1 view模块的职责

##### 3.2.1.1.1 ui.gameui模块的职责

gameui模块负责用户进行游戏时的信息展示以及用户的操作响应

**表3.2.1 各个类功能描述**

|  |  |
| --- | --- |
| **类名称** | **功能描述** |
| GameController | 作为一个游戏的控制器，控制一盘游戏的必要的界面逻辑并与服务器交换信息。 |
| BoardPanel | 游戏棋盘显示界面 |
| ScorePanel | 游戏分数显示界面 |
| PositionConvertor | 坐标转换器，实现下落两种方式 |
| GridStatus | 记录棋子的的动画状态 |
| Tiles | 作为棋子类的容器 |
| Tile | 棋子实体类，储存棋子信息 |
| ImageLibratry | 图片库，负责图片文件读取 |
| BoardSettings | 储存棋盘设定参数 |

##### 3.2.1.1.2 ui.login模块的职责

login模块负责用户登录界面的显示以及对用户在登录界面的操作响应

##### 3.2.1.1.3 ui.solo模块的职责

solo模块负责用户单人游戏的选择和游戏设置，以及用户在该界面的操作响应

##### 3.2.1.1.4 ui.cooperation模块的职责

cooperation模块负责用户协作游戏的选择和游戏设置，以及用户在该界面的操作响应

##### 3.2.1.1.5 ui.fight模块的职责

fight模块负责用户对战游戏的选择和游戏设置，以及用户在该界面的操作响应

#### 3.2.1.2 view模块的接口规范

##### 3.2.1.2.1 gameui模块的接口规范

**表3.2.2 gameui模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| gameui.GameController.  init(String message) | 语法 | public void init(String message) |
| 前置条件 | 已创建一个GameController对象 |
| 后置条件 | 调用GameController中init方法，初始化棋盘数据 |
| gameui.GameController. swapSuccess(String message) | 语法 | public void swapSuccess(String message) |
| 前置条件 | 已创建一个GameController对象 |
| 后置条件 | 调用GameController中swapSuccess方法，对棋盘的两只棋子调用交换 |
| gameui.GameController. swapFail(String message) | 语法 | public void swapFail(String message) |
| 前置条件 | 已创建一个GameController对象 |
| 后置条件 | 调用GameController中swapFail方法，  通知交换失败，对棋盘进行交换失败操作 |
| gameui.GameController. ItemSuccess (String message) | 语法 | public void ItemSuccess (String message) |
| 前置条件 | 已创建一个GameController对象，触发道具 |
| 后置条件 | 调用GameController中ItemSuccess方法 |
| gameui.GameController. SecondEliminate(String message) | 语法 | public void SecondEliminate (String message) |
| 前置条件 | 已创建一个GameController对象 |
| 后置条件 | 调用GameController中SecondEliminate方法，进行二次消除操作。 |
| gameui.GameController.  hint(String message) | 语法 | public void hint (String message) |
| 前置条件 | 已创建一个GameController对象 |
| 后置条件 | 调用GameController中hint方法，界面进行提醒操作 |
| gameui.GameController.  StartSuper() | 语法 | public void StartSuper() |
| 前置条件 | 已创建一个GameController对象，界面进行进入超级模式操作。 |
| 后置条件 | 调用GameController中StartSuper方法 |
| gameui.GameController.  EndSuper() | 语法 | public void EndSuper () |
| 前置条件 | 已创建一个GameController对象 |
| 后置条件 | 调用GameController中EndSuper方法，界面进行结束超级模式操作。 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| NetGameService.swap(Position p1,Position p2) | 处理用户交换操作，做出相应响应 | |
| NetGameService.triggerItem(Positon p) | 处理用户触发道具信息，做出相应响应 | |
| NetGameService.animateFinish() | 用户界面完成一次交换爆炸展示，处理用户界面二次消除 | |

#### 3.2.1.3 view模块的设计原理

将BoardController作为控制器，一方面根据服务请求对BaordPanel，ScorePanel等界面面板进行操作，另一方面，从监听器监听的用户操作请求此Controller发送信息并等待应答。一个Controller对应一次游戏。对界面的变化均从Controller调用。

### controller模块的分解

#### 3.2.2.1controller模块的职责

##### 3.2.2.1.1 netlistenerbl.netUserListenerBL模块的职责

netUserListenerBL模块负责用户的登录以及状态查询的反馈

##### 3.2.2.1.2 netlistenerbl.netCoopGameHallListenerBL模块的职责

netCoopGameHallListenerBL模块负责玩家在协作游戏大厅的行为的反馈

##### 3.2.2.1.3 netlistenerbl.netCoopRoomListenerBL模块的职责

netCoopRoomHallListenerBL模块负责玩家在协作游戏房间的行为的反馈

##### 3.2.2.1.4 netlistenerbl.netBattleListenerGameHallBL模块的职责

netBattleGameListenerHallBL模块负责玩家在对战游戏大厅的行为的反馈

##### 3.2.2.1.5 netlistenerbl.netBattleRoomListenerBL模块的职责

netCoopRoomHallListenerBL模块负责玩家在对战游戏房间的行为的反馈

#### 3.2.2.2 controller模块的接口规范

##### 3.2.2.2.1 netlistenerbl.netUserListenerBL模块的接口规范

**表3.2.3 netUserListenerBL模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| NetUserBLListenerService.selectService(String message) | 语法 | Public void selectServcie(String message ); |
| 前置条件 | 已创建一个netUserListenerBL领域对象 |
| 后置条件 | 调用netUserListenerBL领域对象的selectServcie方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| netUserListenerBL.selectService() | 对返回信息进行处理 | |
| LoginService.receiveMessage() | 调用界面接受信息的接口 | |

##### 3.2.2.2.2 netlistenerbl. netCoopGameHallListenerBL模块的接口规范

**表3.2.4 netCoopGameHallListenerBL模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| netCoopGameHallListenerServcie.selectService(String message) | 语法 | Public void selectServcie(String message ); |
| 前置条件 | 已创建一个netCoopGameHallListenerBL领域对象 |
| 后置条件 | 调用netCoopGameHallListenerBLL领域对象的selectServcie方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| netCoopGameHallListenerBL.selectService() | 对返回信息进行处理 | |
| CoopGameHallService.receiveMessage() | 调用界面接受信息的接口 | |
| CoopGameHallService.setUserList(ArrayList<ActiveUser> userList) | 调用界面设置用户列表 | |
| CoopGameHallService.setRoomList(CoopRoom[] roomList) | 调用界面设置房间列表 | |
| CoopGameHallService.getCoopUserList() | 获取用户列表 | |
| CoopGameHallService.getUser | 获取玩家信息 | |
| CoopGameHallService.getRoomList | 获取房间列表 | |

##### 3.2.2.2.3 netlistenerbl. netCoopRoomListenerBL的接口规范

**表3.2.4 netCoopRoomListenerBL模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| netCoopRoomListenerServcie.selectService(String message) | 语法 | Public void selectServcie(String message ); |
| 前置条件 | 已创建一个netCoopRoomListenerBL领域对象 |
| 后置条件 | 调用netCoopRoomListenerBLL领域对象的selectServcie方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| netCoopRoomListenerBL.selectService() | 对返回信息进行处理 | |
| CoopRoomService.receiveMessage() | 调用界面接受信息的接口 | |
| CoopRoomService.setUserList(ArrayList<ActiveUser> userList) | 调用界面设置用户列表 | |
| CoopRoomService.getPlayerUserList() | 获取玩家列表 | |
| CoopRoomService.getUser() | 获取玩家信息 | |

##### 3.2.2.2.3 netlistenerbl. netBattleGameHallListenerBL的接口规范

**表3.2.5 netBattleGameHallListenerBL接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| netBattleGameHallListenerServcie.selectService(String message) | 语法 | Public void selectServcie(String message ); |
| 前置条件 | 已创建一个netBattleGameHallListenerBL领域对象 |
| 后置条件 | 调用netBattleGameHallListenerBLL领域对象的selectServcie方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| netBattleGameHallListenerBL.selectService() | 对返回信息进行处理 | |
| BattleGameHallService.receiveMessage() | 调用界面接受信息的接口 | |
| BattleGameHallService.setUserList(ArrayList<ActiveUser> userList) | 调用界面设置用户列表 | |
| BattleGameHallService.setRoomList(BattleRoom[] roomList) | 调用界面设置房间列表 | |
| BattleGameHallService.getBattleUserList() | 获取用户列表 | |
| BattleGameHallService.getUser | 获取玩家信息 | |

##### 3.2.2.2.4 netlistenerbl.netBattleRoomListenerBL的接口规范

**表3.2.6 netBattleRoomListenerBL接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| netBattleRoomListenerServcie.selectService(String message) | 语法 | Public void selectServcie(String message ); |
| 前置条件 | 已创建一个netBattleRoomListenerBL领域对象 |
| 后置条件 | 调用netBattleRoomListenerBLL领域对象的selectServcie方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| netBattleRoomListenerBL.selectService() | 对返回信息进行处理 | |
| BattleRoomService.receiveMessage() | 调用界面接受信息的接口 | |
| BattleRoomService.setUserList(ArrayList<ActiveUser> userList) | 调用界面设置用户列表 | |
| BattleRoomService.getUser | 获取玩家信息 | |

#### 3.2.2.3启动模块的设计原理

本模块为MVC框架中的view模块，在接受用户触发的同时，也要接收底层model模块的调用，因此view层必须向model提供接口。

### model模块的分解

#### 3.2.3.1 model模块的职责

##### 3.2.3.1.1 netbl.netUserBL模块的职责

netUserBL模块负责用户的登录以及状态查询

##### 3.2.3.1.2 netbl.netCoopGameHallBL模块的职责

netCoopGameHallBL模块负责玩家在协作游戏大厅的行为的处理

##### 3.2.3.1.3 netbl.netCoopRoomBL模块的职责

netCoopRoomHallBL模块负责玩家在协作游戏房间的行为的处理

##### 3.2.3.1.4netbl.netBattleGameHallBL模块的职责

netBattleGameHallBL模块负责玩家在对战游戏大厅的行为的处理

##### 3.2.3.1.5 netbl.netBattleRoomBL模块的职责

netBattleRoomHallBL模块负责玩家在对战游戏房间的行为的处理

##### 3.2.3.1.6 gameModel模块的职责

gameModel模块负责对游戏进行逻辑控制，存储游戏状态信息。例如判断消除的条件、返回消除的结果、控制游戏的时间。

#### 3.2.3.2 model模块的接口规范

##### 3.2.3.2.1 netbl.netUserBL模块的接口规范

**表3.2.7 netUserBL模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| NetUserBLService.Login() | 语法 | Public void Login( ); |
| 前置条件 | 已创建一个netUserBL领域对象 |
| 后置条件 | 调用netUserBL领域对象的login方法 |
| NetUserBLService.Register() | 语法 | public void Register(String userID,  String password) |
| 前置条件 | 已创建一个netUserBL领域对象 |
| 后置条件 | 调用netUserBL领域对象的ChangePassword方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| netUserBL.Login() | 对用户进行身份认证 | |
| netUserBL.Register(String userID,String password) | 对用户进行注册 | |

##### 3.2.3.2.2 netbl.netCoopGameHallBL模块的接口规范

**表3.2.8netCoopGameHallBL**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| NetCoopGameHallBLService.Enter() | | 语法 | Public void Enter(String userID ); |
| 前置条件 | 已创建一个netCoopGameHallBL领域对象 |
| 后置条件 | 调用netCoopGameHallBL领域对象的Enter方法 |
| NetCoopGameHallBLService.Quit() | | 语法 | public void Quit(String userID) |
| 前置条件 | 已创建一个netCoopGameHallBL领域对象 |
| 后置条件 | 调用netCoopGameHallBL领域对象的Quit方法 |
| NetCoopGameHallBLService.requestUserList() | | 语法 | Public void requestUserList() |
| 前置条件 | 已创建一个netCoopGameHallBL领域对象 |
| 后置条件 | 调用netCoopGameHallBL领域对象的requestUserList方法 |
| NetCoopGameHallBLService.requestCoopRoomList() | 语法 | | Public void requestCoopRoomList() |
| 前置条件 | | 已创建一个netCoopGameHallBL领域对象 |
| 后置条件 | | 调用netCoopGameHallBL领域对象的requestCoopRoomList方法 |
| NetCoopGameHallBLService.enterCoopRoom(String userID,int roomNO) | 语法 | | Public void enterCoopRoom(String userID, int roomNO) |
| 前置条件 | | 已创建一个netCoopGameHallBL领域对象 |
| 后置条件 | | 调用netCoopGameHallBL领域对象的enterCoopRoom方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| netCoopGameHallBL.Enter(String userID) | | 用户登录协作游戏大厅 | |
| neCoopGameHallBL.Quit(String userID) | | 用户退出协作游戏大厅 | |
| netCoopGameHallBL.requestUserList() | | 用户请求游戏大厅当前在线用户列表 | |
| netCoopGameHallBL.requestCoopRoomList() | | 用户请求游戏大厅当前房间列表 | |
| netCoopGameHallBL.enterCoopRoom | | 用户进入游戏大厅 | |

##### 3.2.3.2.3 netbl.netCoopRoomBL模块的接口规范

**表3.2.9 netCoopRoomBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| NetCoopRoomBLService.requestUserList(int roomNO) | | 语法 | Public void requestUserList(int roomNO ); |
| 前置条件 | 已创建一个netCoopRoomBL领域对象 |
| 后置条件 | 调用netCoopRoomBL领域对象的requestUserList方法 |
| NetCoopRoomBLService.setReady() | | 语法 | public void setReady(String userID,int roomNO) |
| 前置条件 | 已创建一个netCoopRoomBL领域对象 |
| 后置条件 | 调用netCoopRoomBL领域对象的setReady方法 |
| NetCoopRoomBLService.chatInRoom(String message,String userID,int roomNO) | | 语法 | Public void charInRoom(String message,String userID,int roomNO) |
| 前置条件 | 已创建一个netCoopRoomBL领域对象 |
| 后置条件 | 调用netCoopRoomBL领域对象的chatInRoom 方法 |
| NetCoopRoomBLService.quitCoopRoom(String userID, int roomNO) | 语法 | | Public void quitCoopRoom(String userID, int roomNO) |
| 前置条件 | | 已创建一个netCoopRoomBL领域对象 |
| 后置条件 | | 调用netCoopRoomBL领域对象的quitCoopRoom方法 |
| NetCoopRoomBLService.kickPlayer(String roommasterID,String userID,int roomNO) | 语法 | | Public void kickPlayer(String roommasterID,String userID, int roomNO) |
| 前置条件 | | 已创建一个netCoopRoomBL领域对象 |
| 后置条件 | | 调用netCoopRoomBL领域对象的kickPlayer方法 |
| NetCoopRoomBLService.initGame(boolean c, boolean d, boolean e, String userID,  int roomNO) | 语法 | | Public void initGame(boolean c, boolean d, boolean e, String userID,  int roomNO) |
| 前置条件 | | 已创建一个netGameBL领域对象 |
| 后置条件 | | 调用netGameBL领域对象的initGame方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| netCoopRoomBL. requestUserList(int roomNO) | | 用户请求房间用户列表 | |
| neCoopRoomBL. setReady (String userID,int roomNO) | | 用户设置准备状态 | |
| netCoopRoomBL. chatInRoom(String message,String userID,int roomNO) | | 用户在房间内发送聊天信息 | |
| netCoopRoomBL. quitCoopRoom(String userID, int roomNO) | | 用户请离开房间 | |
| netCoopGameHallBL. kickPlayer(String roommasterID,String userID,int roomNO) | | 房主踢人 | |
| netGameBL.initGame(boolean c, boolean d, boolean e, String userID,  int roomNO) | | 初始化游戏 | |

##### 3.2.3.2.4 netbl.netBattleGameHallBL模块的接口规范

**表3.2.9 netBattleGameHallBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| NetBattleGameHallBLService.Enter() | | 语法 | Public void Enter(String userID ); |
| 前置条件 | 已创建一个netBattleGameHallBL领域对象 |
| 后置条件 | 调用netBattleGameHallBL领域对象的Enter方法 |
| NetBattleGameHallBLService.Quit() | | 语法 | public void Quit(String userID) |
| 前置条件 | 已创建一个netBattleGameHallBL领域对象 |
| 后置条件 | 调用netBattleGameHallBL领域对象的Quit方法 |
| NetBattleGameHallBLService.requestUserList() | | 语法 | Public void requestUserList() |
| 前置条件 | 已创建一个netBattleGameHallBL领域对象 |
| 后置条件 | 调用netBattleGameHallBL领域对象的requestUserList方法 |
| NetBattleGameHallBLService.requestBattleRoomList() | 语法 | | Public void requestBattleRoomList() |
| 前置条件 | | 已创建一个netBattleGameHallBL领域对象 |
| 后置条件 | | 调用netBattleGameHallBL领域对象的requestBattleRoomList方法 |
| NetBattleGameHallBLService.enterCoopRoom(String userID,int roomNO) | 语法 | | Public void enterBattleRoom(String userID, int roomNO) |
| 前置条件 | | 已创建一个netBattleGameHallBL领域对象 |
| 后置条件 | | 调用netBattleGameHallBL领域对象的enterBattleRoom方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| netBattleGameHallBL.Enter(String userID) | | 用户登录对战游戏大厅 | |
| netBattleGameHallBL.Quit(String userID) | | 用户退出对战游戏大厅 | |
| netBattleGameHallBL.requestUserList() | | 用户请求游戏大厅当前在线用户列表 | |
| netBattleGameHallBL.requestCoopRoomList() | | 用户请求游戏大厅当前房间列表 | |
| netBattleGameHallBL.enterCoopRoom | | 用户进入游戏大厅 | |

##### 3.2.3.2.5 netbl.netCoopRoomBL模块的接口规范

**表3.2.10 netCoopRoomBL模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| NetBattleRoomBLService.requestUserList(int roomNO) | | 语法 | Public void requestUserList(int roomNO ); |
| 前置条件 | 已创建一个netBattleRoomBL领域对象 |
| 后置条件 | 调用netBattleRoomBL领域对象的requestUserList方法 |
| NetBattleRoomBLService.setReady() | | 语法 | public void setReady(String userID,int roomNO) |
| 前置条件 | 已创建一个netBattleRoomBL领域对象 |
| 后置条件 | 调用netBattleRoomBL领域对象的setReady方法 |
| NetBattleRoomBLService.chatInRoom(String message,String userID,int roomNO) | | 语法 | Public void charInRoom(String message,String userID,int roomNO) |
| 前置条件 | 已创建一个netBattleRoomBL领域对象 |
| 后置条件 | 调用netBattleRoomBL领域对象的chatInRoom 方法 |
| NetBattleRoomBLService.quitCoopRoom(String userID, int roomNO) | 语法 | | Public void quitBattleRoom(String userID, int roomNO) |
| 前置条件 | | 已创建一个netBattleRoomBL领域对象 |
| 后置条件 | | 调用netBattleRoomBL领域对象的quitBattleRoom方法 |
| NetBattleRoomBLService.kickPlayer(String roommasterID,String userID,int roomNO) | 语法 | | Public void kickPlayer(String roommasterID,String userID, int roomNO) |
| 前置条件 | | 已创建一个netBattleRoomBL领域对象 |
| 后置条件 | | 调用netBattleRoomBL领域对象的kickPlayer方法 |
| NetBattleRoomBLService.initGame(boolean c, boolean d, boolean e, String userID,  int roomNO) | 语法 | | Public void initGame(boolean c, boolean d, boolean e, String userID,  int roomNO) |
| 前置条件 | | 已创建一个netGameBL领域对象 |
| 后置条件 | | 调用netGameBL领域对象的initGame方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| netBattleRoomBL. requestUserList(int roomNO) | | 用户请求房间用户列表 | |
| neBattleRoomBL. setReady (String userID,int roomNO) | | 用户设置准备状态 | |
| netBattleRoomBL. chatInRoom(String message,String userID,int roomNO) | | 用户在房间内发送聊天信息 | |
| netBattleRoomBL. quitCoopRoom(String userID, int roomNO) | | 用户请离开房间 | |
| netBattleGameHallBL. kickPlayer(String roommasterID,String userID,int roomNO) | | 房主剔除玩家 | |
| netGameBL.initGame(boolean c, boolean d, boolean e, String userID, int roomNO) | | 初始化游戏 | |

##### 3.2.3.2.6 gameModel模块的接口规范

**表3.2.10** gameModel**模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| gameModel. initGame(String item) | | 语法 | public String initGame(String item); |
| 前置条件 | 已创建一个gameModel领域对象 |
| 后置条件 | 向上一层发送结果信息 |
| gameModel. swap(String twoBlock) | | 语法 | public String swap(String twoBlock); |
| 前置条件 | 已创建一个gameModel领域对象 |
| 后置条件 | 调用sendMessage()方法 |
| gameModel. triggerItem(String message) | | 语法 | public String triggerItem(String message); |
| 前置条件 | 已创建一个gameModel领域对象 |
| 后置条件 | 调用sendMessage()方法 |
| gameModel. animateFinish(); | 语法 | | public String animateFinish(); |
| 前置条件 | | 已创建一个gameModel领域对象 |
| 后置条件 | | sendMessage()方法 |
| gameModel. getTip() | 语法 | | public String getTip(); |
| 前置条件 | | 已创建一个gameModel领域对象 |
| 后置条件 | | sendMessage()方法 |
| gameModel. startSuper() | 语法 | | public String startSuper(); |
| 前置条件 | | 已创建一个gameModel领域对象 |
| 后置条件 | | sendMessage()方法 |
| gameModel. endSuper() | 语法 | | public String endSuper(); |
| 前置条件 | | 已创建一个gameModel领域对象 |
| 后置条件 | | sendMessage()方法 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| gameui.GameController.init(String message) | | 开始游戏，初始化游戏界面 | |
| gameui.GameController. swapSuccess(String message) | | 交换成功调用界面动画 | |
| gameui.GameController. swapFail(String message) | | 交换不成功调用界面动画 | |
| gameui.GameController. ItemSuccess(String message) | | 触发道具成功调用界面动画 | |
| gameui.GameController. SecondEliminate(String message) | | 二次消除，即砖块下落后又可以进行消除 | |
| gameui.GameController. hint(String message) | | 界面显示提示 | |
| gameui.GameController. StartSuper() | | 开始超级模式，界面动画 | |
| gameui.GameController. EndSuper() | | 结束超级模式，界面动画 | |

#### 3.2.3.3 启动模块的设计原理

## 服务器模块设计

### Gamemodel模块分解

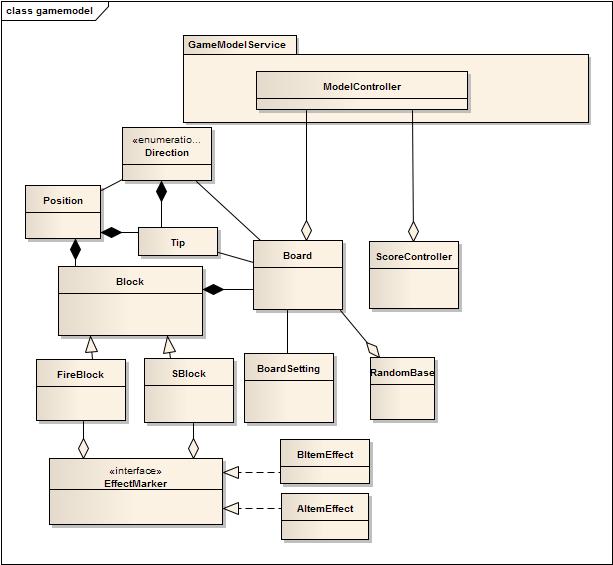


图 Gamemodel 模块包图

#### 3.3.1.1 Gamemodel模块

Gamemodel是游戏的逻辑处理模块，负责游戏规则细节的全部算法实现，作为模型层向外提供消除操作逻辑接口和返回消除结果。

**表 Gamemodel 模块各个类功能描述**

|  |  |
| --- | --- |
| 类名称 | 功能描述 |
| ModelController | 作为一个游戏的控制器，控制一盘游戏的底层操作逻辑并与服务器交换信息。 |
| Board | 棋盘逻辑实现类，封装棋盘的数据及逻辑算法 |
| ScoreController | 封装游戏分数的逻辑 |
| Tip | 游戏提示信息封装类 |
| Position | 位置封装类 |
| Direction | 方向枚举类 |
| Block | 棋子逻辑实体类，封装棋子信息 |
| Sblock | A类棋子，继承棋子类 |
| FireBlock | B类棋子，继承棋子类 |
| EffectMarker | 标记消除位置封装类 |
| AitemEffect | A道具消除效果，实现EffectMarker |
| BitemEffect | B道具消除效果，实现EffectMarker |
| RandomBase | 随机数方法库封装类 |
| BoardSettings | 储存棋盘底层设定常量参数 |

#### 3.1.1.2 Gamemodel模块接口规范

**表 Gamemodel接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| Board.ArrayList<Block> checkInstable() | | 语法 | Public ArrayList<Block> checkInstable() |
| 前置条件 | Board已初始化，交换结束 |
| 后置条件 | 检测是否有消除的棋子，返回消除棋子列表 |
| checkMatchAfterItem(int row, int col) | | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
|  | | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
|  | 语法 | |  |
| 前置条件 | |  |
| 后置条件 | |  |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |

## 界面模块的设计

# 系统体系结构设计思路

# 依赖视角

龙珠消消乐游戏客户端开发包图如图x所示，服务器端开发包图如图x所示。

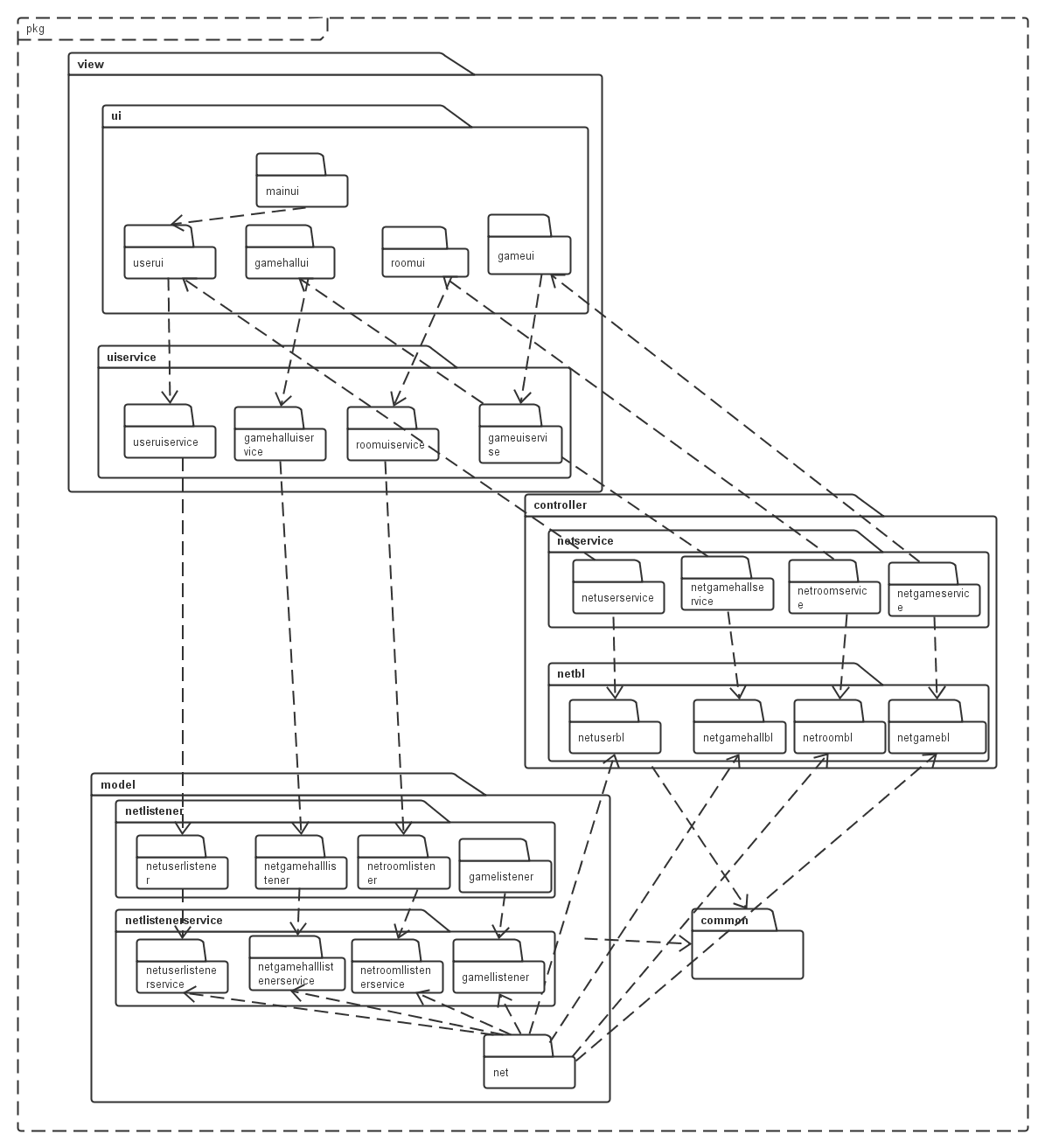


图4 龙珠消消乐游戏客户端包图

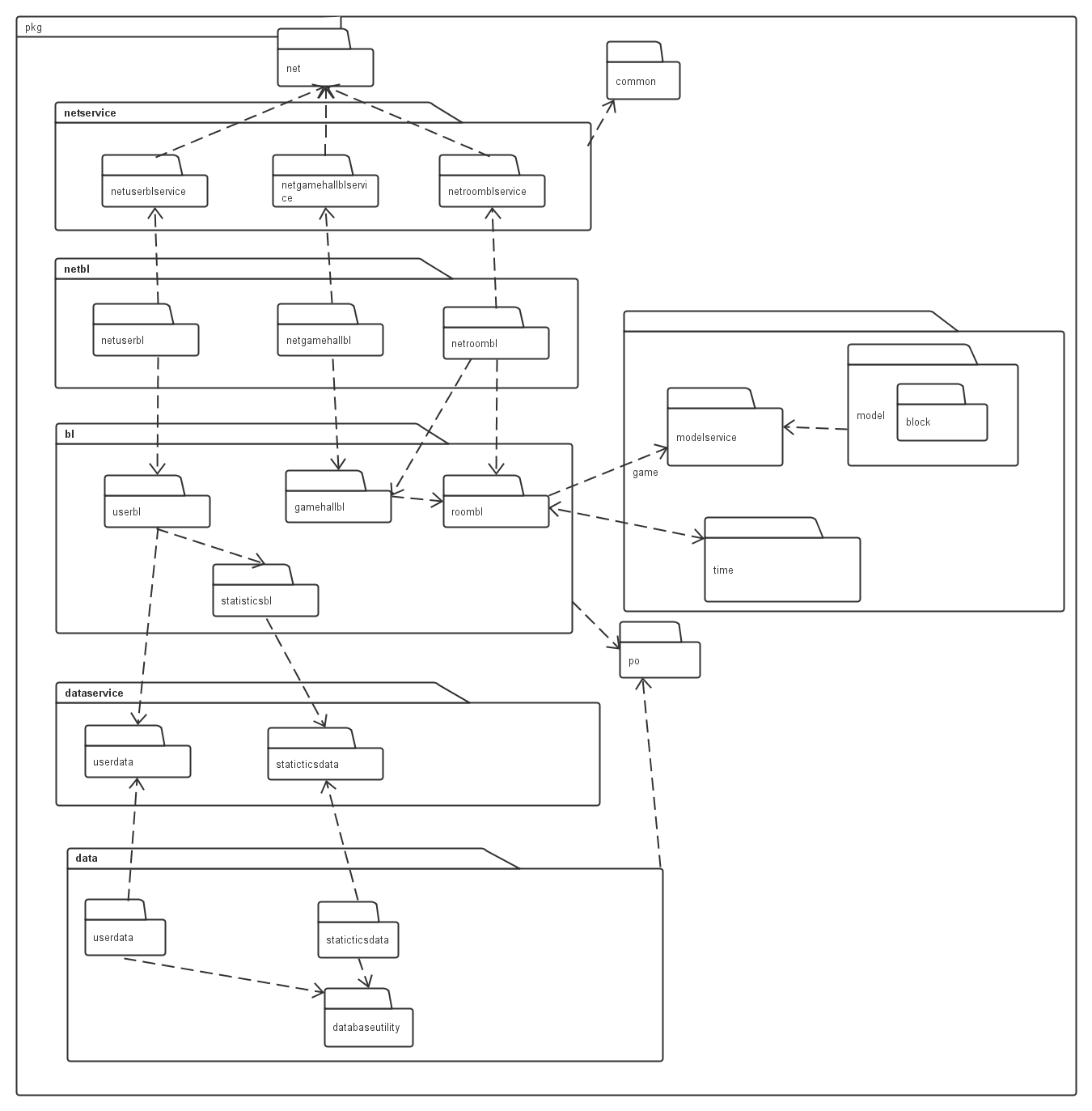


图5龙珠消消乐游戏服务器端包图