酒店预订管理系统

System of Booking Hotel

体系结构设计文档

V 1.1

小组37 陈统 刚昭 高岳 贺云青

更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 更新者 | 更新内容 | 版本号 |
| 2016.10.9 | 全体组员 | 第一版本初稿 | V 0.1 |
| 2016.10.11 | 全体组员 | 拆分酒店模块，增加评价模块，修改依赖关系，增加部分图 | V 0.2 |
| 2016.10.14 | 全体组员 | 增加了接口规范 | V0.3 |
| 2016.10.14 | 高岳 | 增加了界面跳转图，业务逻辑层的职责说明，信息视角 | V0.4 |
| 2016.10.15 | 贺云青 | 增加了数据层的职责说明 | V1.0 |
| 2016.10.15 | 高岳 | 增加了PO的类定义 | V1.1 |

目录

[1.引言 4](#_Toc464291817)

[1.1编制目的 4](#_Toc464291818)

[1.2词汇表 4](#_Toc464291819)

[1.3参考资料 4](#_Toc464291820)

[2.产品描述 4](#_Toc464291821)

[3.逻辑视角 4](#_Toc464291822)

[4.组合视角 6](#_Toc464291823)

[4.1开发包图 6](#_Toc464291824)

[4.2运行时进程 8](#_Toc464291825)

[4.3物理部署 9](#_Toc464291826)

[5.接口视角 9](#_Toc464291827)

[5.1模块的职责 9](#_Toc464291828)

[5.2用户界面层的分解 11](#_Toc464291829)

[5.2.1用户界面层模块的职责 12](#_Toc464291830)

[5.2.2用户界面模块的接口规范 12](#_Toc464291831)

[5.2.3用户界面模块设计原理 12](#_Toc464291832)

[5.3业务逻辑层的分解 12](#_Toc464291833)

[5.3.1业务逻辑层模块的职责 13](#_Toc464291834)

[5.3.2 业务逻辑模块的接口模范 13](#_Toc464291835)

[5.4数据层的分解 21](#_Toc464291836)

[5.4.1数据层模块的职责 21](#_Toc464291837)

[5.4.2 数据层模块的接口规范 21](#_Toc464291838)

[6.信息视角 27](#_Toc464291839)

[6.1数据持久化对象 27](#_Toc464291840)

[6.2数据库表 27](#_Toc464291841)

# 1.引言

## 1.1编制目的

本报告详细完成对酒店预订管理系统的概要设计，达到指导详细设计和开发的目的，同时实现测试人员及用户的沟通。

本报告面向开发人员、测试人员及最终用户编写，是了解系统的导航。

## 1.2词汇表

|  |  |
| --- | --- |
| **缩写或单词** | **解释** |
| SBH | 酒店预订管理系统 |
| user | 用户 |
| hotel | 酒店 |
| order | 订单 |
| promotion | 促销策略 |
| comment | 评价 |

## 1.3参考资料

《酒店预订管理系统用例文档》

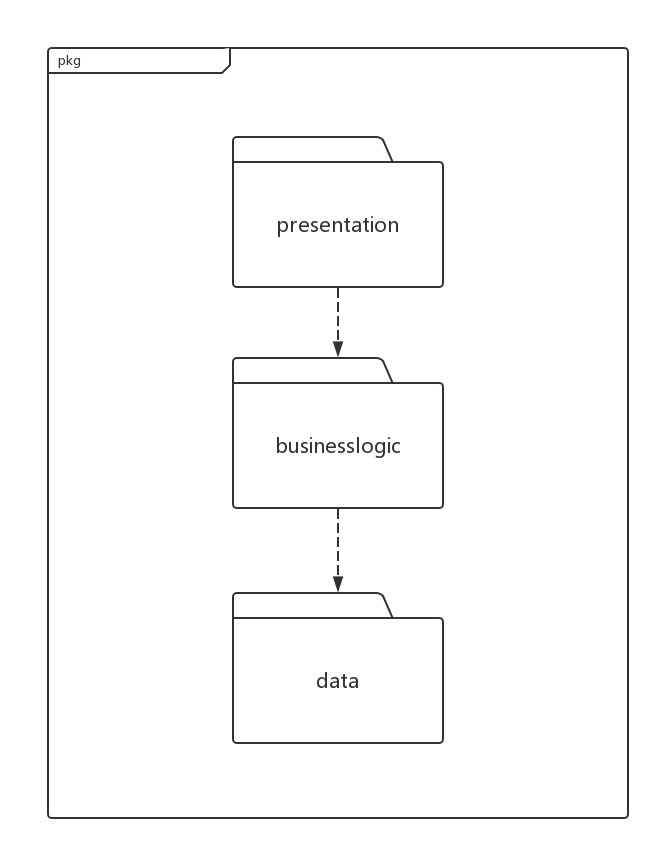
《酒店预订管理系统需求规格说明文档》

# 2.产品描述

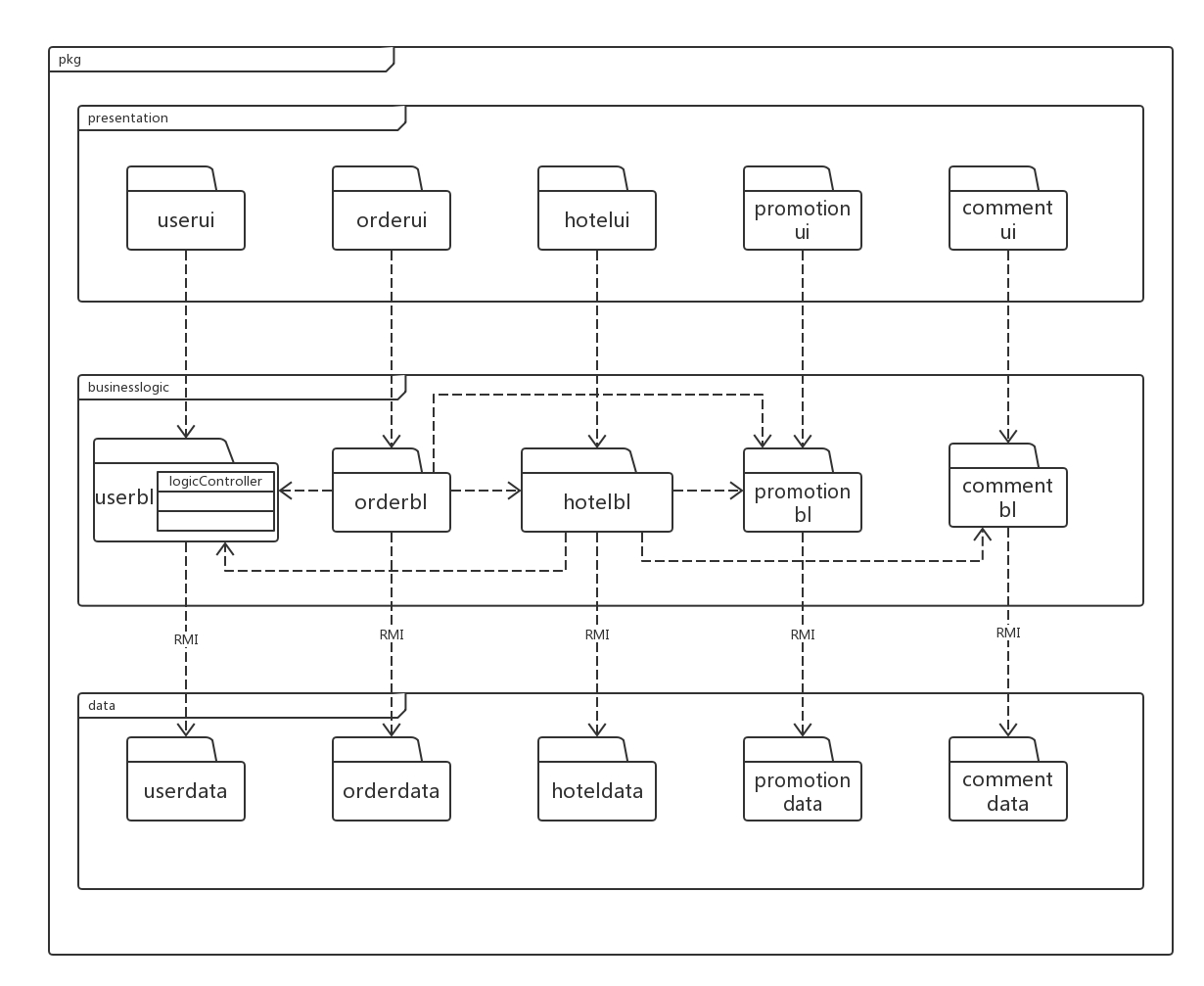
参考《酒店预订管理系统用例文档》和《酒店预订管理系统需求规格说明文档》中对产品的概括描述。

# 3.逻辑视角

酒店预订管理系统中，选择了分层体系结构的风格，将系统分为3层（展示层、业务逻辑层、数据层）能够很好的示意整个高层抽象。展示层包括GUI页面的实现，业务逻辑层包含业务逻辑处理的实现，数据层负责数据的持久化和访问。分层体系结构的逻辑视角和逻辑设计方案如图1和图2所示。



**图1：分层结构**



**图2：逻辑设计方案**

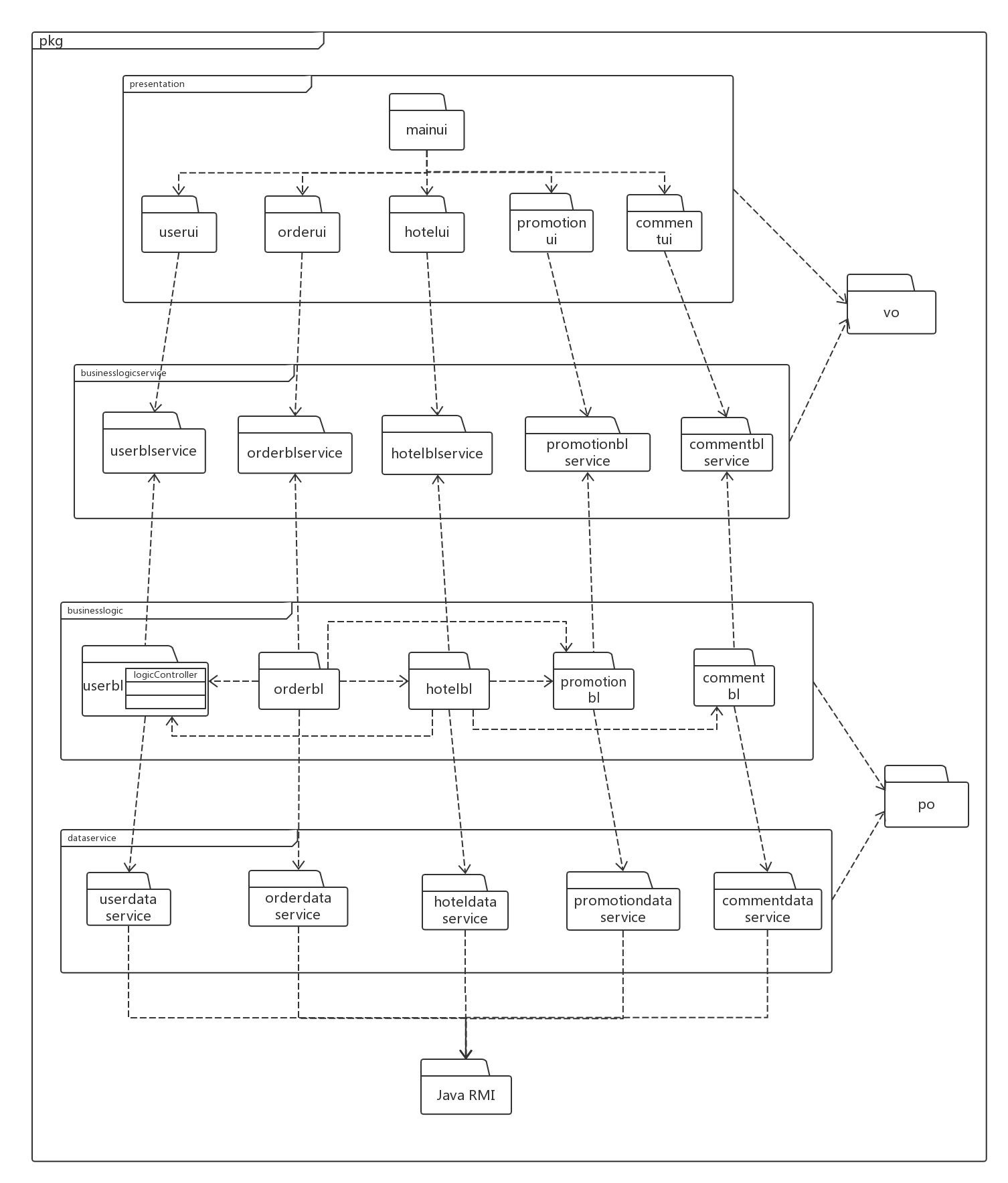
# 4.组合视角

## 4.1开发包图

酒店预订管理系统的最终开发包设计如表1所示：

**表1酒店预订管理系统的最终开发包设计**

|  |  |
| --- | --- |
| **开发（物理）包** | **依赖的其他开发包** |
| mainui | userui, orderui, hotelui, promotionui, vo |
| userui | userblservice, 界面类库包, vo |
| userblservice |  |
| userbl | userblservice, userdataservice, po |
| userdataservice | Java RMI,po |
| userdata | userdataservice , databaseutility, po |
| orderui | orderblservice, 界面类库包, vo |
| orderblservice |  |
| orderbl | orderdataservice, orderblservice, userbl, hotelbl, promotionbl, po |
| orderdataservice | Java RMI, po |
| orderdata | orderdataservice , databaseutility, po |
| hotelui | hotelblservice, 界面类库包, vo |
| hotelblservice |  |
| hotelbl | hoteldataservice, hotelblservice, commentbl, orderbl, userbl, promotionbl, po |
| hoteldataservice | Java RMI, po |
| hoteldata | hoteldataservice , databaseutility, po |
| commentui | commentblservice, 界面类库包, vo |
| commentblservice |  |
| commentbl | commentdataservice, commentblservice, po |
| commentdataservice | Java RMI, po |
| commentdata | commentdataservice, databaseutility, po |
| promotioniui | promotionblservice, 界面类库包, vo |
| promotionblservice |  |
| promotionbl | promotiondataservice, promotionblservice, po |
| promotiondataservice | Java RMI, po |
| promotiondata | promotiondataservice , databaseutility, po |
| vo |  |
| po |  |
| 界面类库包 |  |
| databaseseutility | JDBC |
| Java RMI |  |



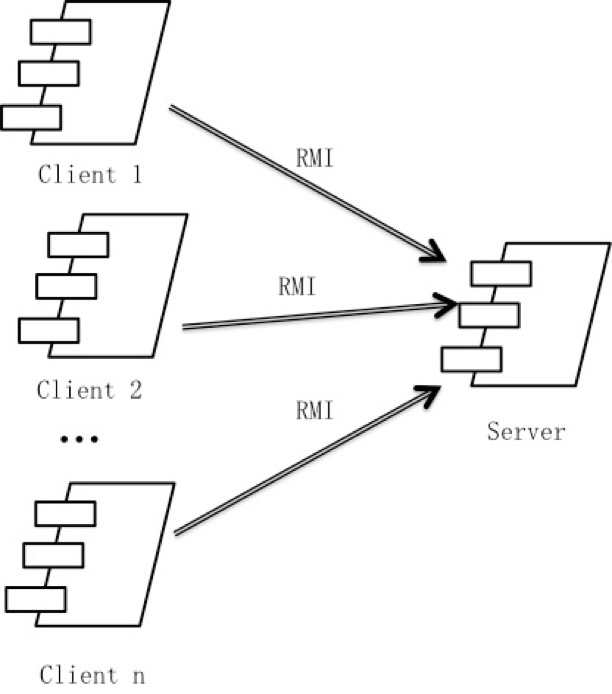
**图3 客户端开发包图**



**图4 服务器端开发包图**

## 4.2运行时进程

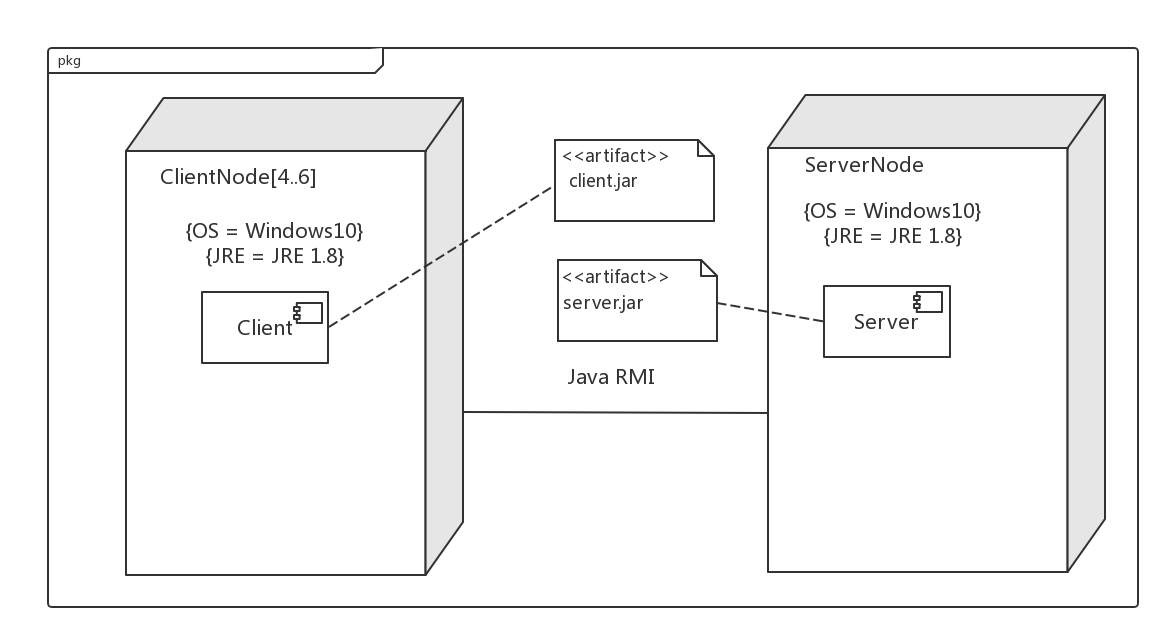
在酒店预订管理系统中，会有多个客户端进程和一个服务器端进程，其进程图如图5所示。结合部署图，客户端进程是在客户端机器上运行，服务器端进程是在服务器端机器上运行。



**图5 进程图**

## 4.3物理部署

酒店预订管理系统系统中客户端构件是放在客户端机上，服务器端构件是放在服务器端机器上。部署图如图6所示。

****

**图 6 部署图**

# 5.接口视角

## 5.1模块的职责

客户端模块和服务器模块视图分别如图7和图8所示。客户端各层和服务器端各层的职责分别如表2和表3所示。

**网络模块**

**展示层**

**业务逻辑层**

**启动模块**

**图7 客户端模块视图**

**业务逻辑层**

**网络模块**

**启动模块**

**图8 服务器端模块视图**

**表2 客户端各层的职责**

|  |  |
| --- | --- |
| 层 | 职责 |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 用户界面层 | 基于窗口的进销存管理系统客户端用户界面 |
| 业务逻辑层 | 对于用户界面的输入进行响应并进行业务处理逻辑 |
| 客户端网络模块 | 利用java RMI机制查找RMI服务 |

**表3 服务器端各层的职责**

|  |  |
| --- | --- |
| 层 | 职责 |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 数据层 | 负责数据的持久化及数据访问接口 |
| 服务器端网络模块 | 利用Java RMI机制开启RMI服务，注册RMI服务 |

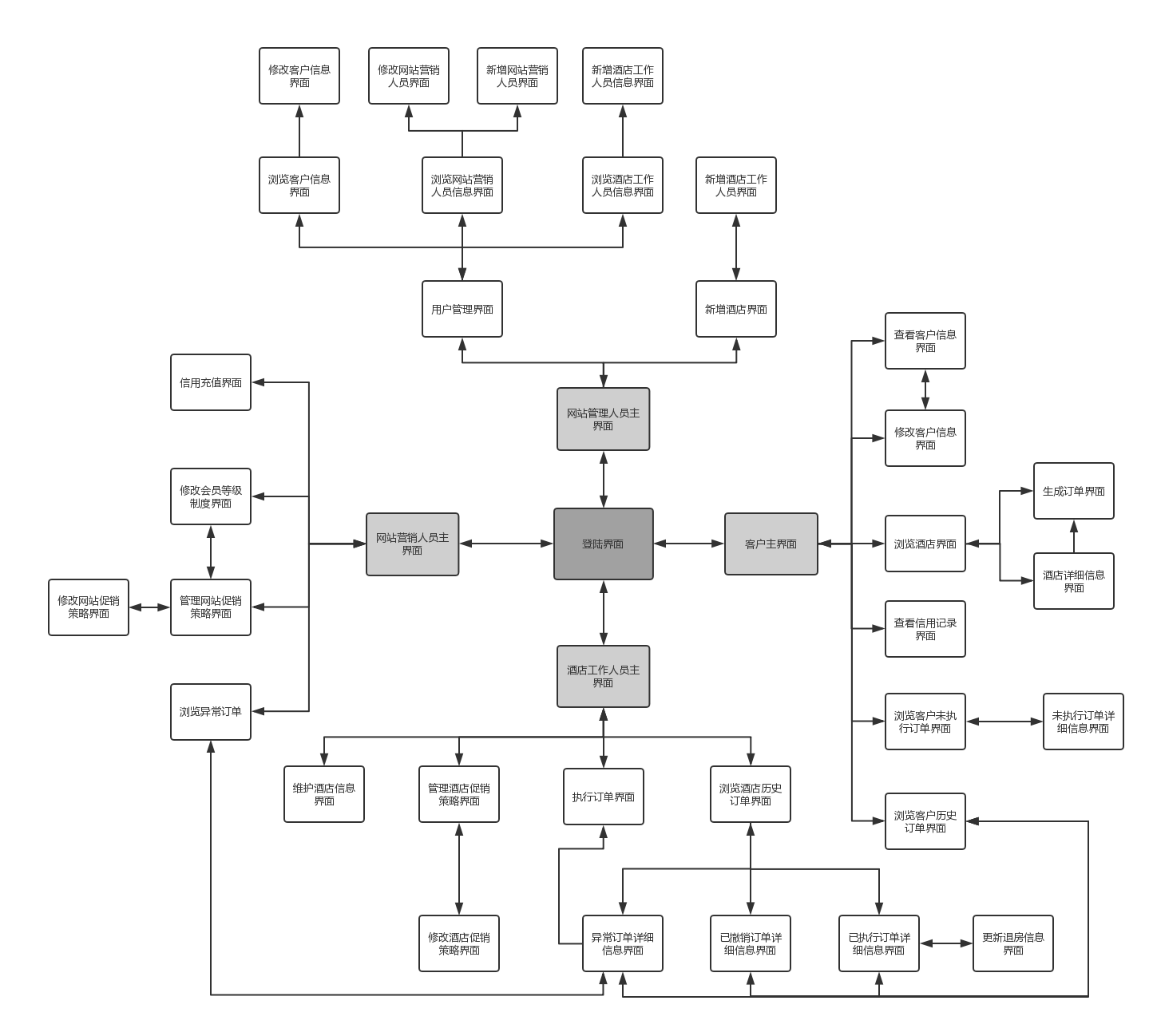
每一层只是使用下方直接接触的层。层与层之间仅仅是通过接口的调用来完成的。层之间调用的接口如表4所示。

**表4 层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| userblservice  orderblservice  hotelblservice  promotionblservice  commentblservice | 客户端展示层 | 客户端业务逻辑层 |
| userdataservice  orderdataservice  hoteldataservice  promotiondataservice  commentdataservice | 客户端业务逻辑层 | 服务器端数据层 |

## 5.2用户界面层的分解

根据需求，系统存在37个用户界面（加粗为主界面）：**登陆界面**、**客户主界面**、查看客户信息界面、修改客户信息界面、浏览酒店界面、查看信息用记录界面、浏览客户未执行订单界面、浏览客户历史订单界面、生成订单界面、酒店详细信息界面、未执行订单详细信息界面、**酒店工作人员主界面**、维护酒店信息界面、管理酒店促销策略界面、执行订单界面、浏览酒店历史订单界面、修改酒店促销策略界面、异常订单详细信息界面、已撤销订单详细信息界面、已执行订单详细信息界面、更新退房信息界面、**网站营销人员主界面**、信用充值界面、修改会员等级制度界面、网站促销策略界面、浏览异常订单界面、修改网站促销策略界面、**网站管理人员主界面**、用户管理界面、新增酒店界面、浏览客户信息界面、浏览网站营销人员信息界面、浏览酒店工作人员信息界面、新增酒店工作人员界面、修改客户信息界面、修改网站营销人员界面、新增网站营销人员界面、新增酒店工作人员界面



**图9 酒店预订管理系统界面跳转图**

### 5.2.1用户界面层模块的职责

**表5 用户界面层模块的职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| MainFrame | 界面frame，负责界面的显示和界面的跳转 |

### 5.2.2用户界面模块的接口规范

**表6 用户界面层模块的接口规范**

|  |  |  |
| --- | --- | --- |
| MainFrame | 语法 | init(args:String[]) |
| 前置条件 | 用户触发相应需要显示界面的条件 |
| 后置条件 | 显示Frame并且加载LoginPanel |

**表7 用户界面层模块需要的服务接口**

|  |  |
| --- | --- |
| 服务名 | 服务 |
| bl.accountservice | 登录界面的业务逻辑接口 |
| bl.\*blservice | 每个ui模块会有对应的数据逻辑接口 |

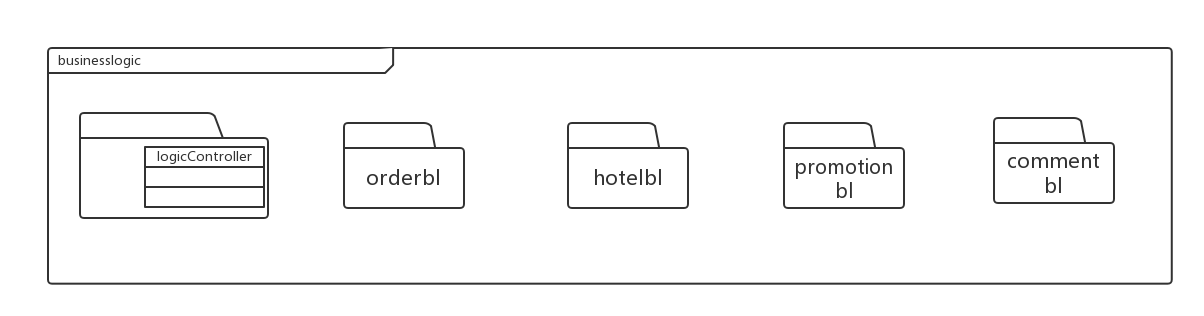
### 5.2.3用户界面模块设计原理

用户界面拟用javafx来实现。

## 5.3业务逻辑层的分解

业务逻辑层包括多个针对界面及业务逻辑处理对象。例如User对象负责处理登录界面

的业务逻辑。业务逻辑层的设计如下图所示：



**图10 业务逻辑层的设计**

### 5.3.1业务逻辑层模块的职责

**表8 业务逻辑层模块的职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| userbl | 实现登录与对用户账户信息增删改查的服务 |
| orderbl | 实现对订单进行增删改查操作的服务 |
| hotelbl | 实现对酒店信息进行增改查和对酒店可用客房进行修改的服务 |
| promotionbl | 实现对网站促销策略、酒店促销策略和会员登记制度进行制定、修改、增加的服务 |
| commentbl | 实现客户对订单进行评价的服务 |

### 5.3.2 业务逻辑模块的接口模范

* + - 1. **userbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| User.login | 语法 | public ResultMessage login(String userName, String password) |
| 前置条件 | 账号、密码符合语法规则 |
| 后置条件 | 系统确认账号密码是否正确，若正确，进入主界面并记录登录信息，否则提示错误信息 |
| User.logout | 语法 | public ResultMessage logout(String ID) |
| 前置条件 | 系统已登录 |
| 后置条件 | 系统退出登录，记录登出信息 |
| User.add | 语法 | public ResultMessage add(UserVO vo) |
| 前置条件 | 账户信息符合规则且齐全 |
| 后置条件 | 系统创建该账户并持久化增加该账户数据 |
| User.findbyID | 语法 | public UserVO findbyID(String ID) |
| 前置条件 | 账户ID符合输入语法要求 |
| 后置条件 | 系统返回查找到的搜索结果 |
| User.findbyUserName | 语法 | public UserVO findbyUserName(String userName) |
| 前置条件 | 账户名符合输入语法要求 |
| 后置条件 | 系统返回查找到的搜索结果 |
| User.modifyPassword | 语法 | public ResultMessage modifyPassword(String userName, String oldPassword, String newPassword) |
| 前置条件 | 新密码符合输入语法要求，旧密码匹配正确 |
| 后置条件 | 系统修改该账户的密码数据 |
| User.modify | 语法 | public ResultMessage modify(UserVO vo) |
| 前置条件 | 账户信息符合规则且齐全 |
| 后置条件 | 系统修改该账户并持久化保存该账户数据 |
| User.addCreditRecord | 语法 | public ResultMessage addCreditRecord (CreditRecordVO vo) |
| 前置条件 | 该账户ID为客户 |
| 后置条件 | 系统增加该客户信用记录并持久化保存数据 |
| User.findCreditRecord | 语法 | public ArrayList<CreditRecordVO> findCreditRecord(String ID) |
| 前置条件 | 该账户ID为客户且输入符合语法要求 |
| 后置条件 | 返回该账户所有信用 |
| 需要的接口（需接口） | | |
| 服务名 | 服务内容 | |
| UserDataService.  add(UserPO po) | 根据提供的用户po增加持久化对象 | |
| UserDataService.  findbyID(String ID) | 根据用户ID查找持久化对象 | |
| UserDataService.  findbyUserName(String userName) | 根据UserName查找持久化对象 | |
| UserDataService.  modify (UserPO po) | 修改单一持久化对象 | |
| UserDataService.  Check(String userName, String password) | 检查账户密码是否正确 | |
| UserDataService.  addCreditRecord  (CreditRecordPO po) | 根据提供的信用记录po增加持久化对象 | |
| UserDataService.  findCreditRecord  (String ID) | 根据用户ID查找所有信用记录的持久化对象 | |
| UserDataService.init() | 初始化持久化数据库 | |
| UserDataService.finish() | 结束持久化数据库的使用 | |

* + - 1. **orderbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Order.createOrder | 语法 | public ResultMessage createOrder(OrderVO vo) |
| 前置条件 | orderIfo包含完整且正确的信息 |
| 后置条件 | 创建和保存新的未执行订单 |
| Order.cancelOrder | 语法 | public ResultMessage cancelOrder (String orderID) |
| 前置条件 | 存在未执行订单 |
| 后置条件 | 系统成功/失败撤销一条未执行订单 |
| Order.getOrder | 语法 | public OrderVO getOrder (String ID) |
| 前置条件 | ID输入符合规范 |
| 后置条件 | 返回一个订单的vo |
| Order.getAbnormalOrder | 语法 | public ArrayList<OrderVO> getAbnormalOrder () |
| 前置条件 | 无 |
| 后置条件 | 返回当日所有异常订单的vo |
| Order.getUserOrderList | 语法 | public ArrayList<OrderVO> getUserOrderList (String userID, OrderType type) |
| 前置条件 | 无 |
| 后置条件 | 系统返回某客户某种订单的订单列表 |
| Order.getHotelOrderList | 语法 | public ArrayList<OrderVO> getHotelOrderList(String hotelID, OrderType type) |
| 前置条件 | 无 |
| 后置条件 | 系统返回某酒店某种订单的订单列表 |
| Order.executeOrder | 语法 | public ResultMessage executeOrder (String orderID) |
| 前置条件 | 存在未执行订单 |
| 后置条件 | 未执行订单被置为已执行订单，信用记录被更新 |
| Order.cancelAbnormalOrder | 语法 | public ResultMessage cancelAbnormalOrder (String orderID, Boolean isHalf) |
| 前置条件 | 存在异常订单 |
| 后置条件 | 系统将异常订单更新为已撤销订单，记录撤销时间，并更新信用值和信用记录 |
| 需要的接口（需接口） | | |
| 服务名 | 服务内容 | |
| User.addCreditRecord (CreditRecordVO vo) | 增加该客户信用记录并持久化保存数据 | |
| User.findbyID (String ID) | 返回一个UserVO | |
| Promotion.getPromotionBy  Hotel (String hotelAddress) | 返回符合该酒店订单的所有促销策略（包括酒店促销策略与网站促销策略） | |
| Hotel.getHotelVO(String hotelAddress) | 返回一个酒店的vo | |
| Hotel.changeRoom(  RoomTypetype, int num) | 系统更新酒店的可用客房信息并保存酒店的po | |
| OrderDataService.  add (OrderPO po) | 系统添加新的订单记录 | |
| OrderDataService.  getOrderPO(String orderID) | 系统返回一条订单信息 | |
| OrderDataService.  getUserOrderPO (String userID, OrderType type) | 返回某客户某种类全部订单po | |
| OrderDataService.  getHotelOrderPO (String hotelID, OrderType type) | 返回某酒店某种类全部订单po | |
| OrderDataService.  getAbnormalOrderPO (Date date) | 返回当日全部异常订单po | |
| OrderDataService.  modify (OrderPO po) | 更新一个po | |

* + - 1. **hotelbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotel.findHotels | 语法 | public ArrayList<HotelVO>  findHotels(HotelFilter filter ) |
| 前置条件 | 无 |
| 后置条件 | 系统返回该筛选条件下的所有酒店vo |
| Hotel.getHotel | 语法 | public HotelVO getHotel(String hotelAddress) |
| 前置条件 | 无 |
| 后置条件 | 系统返回一个该酒店的vo |
| Hotel.addHotel | 语法 | public ResultMessage addHotel(HotelVO hotelVO, UserVO userVO) |
| 前置条件 | 酒店信息、酒店工作人员信息输入合法 |
| 后置条件 | 系统增加酒店和酒店工作人员的持久化对象 |
| Hotel.modifyHotel | 语法 | public ResultMessage modifyHotel(HotelVO vo) |
| 前置条件 | 酒店信息输入合法 |
| 后置条件 | 系统更新酒店po |
| Hotel.changeRoom | 语法 | public ResultMessage changeRoom(RoomType type, int num) |
| 前置条件 | 房间类型数量输入合法 |
| 后置条件 | 系统更新酒店po |
| Hotel.getHotelVO | 语法 | public HotelVO getHotelVO(String hotelAddress) |
| 前置条件 | 存在对应酒店 |
| 后置条件 | 返回一个酒店的vo |
| Hotel.getCitys | 语法 | public ArrayList<String> getCitys() |
| 前置条件 | 无 |
| 后置条件 | 返回系统中所有城市的名称 |
| Hotel.getAreas | 语法 | public ArrayList<String> getAreas(String city) |
| 前置条件 | 城市名称存在 |
| 后置条件 | 返回系统中该城市的所有商圈名称 |
| 需要的接口（需接口） | | |
| 服务名 | 服务内容 | |
| Promotion.  getPromotionByHotel  (String hotelAddress) | 根据酒店地址返回酒店的促销策略 | |
| HotelDataService.  addHotelInfo(HotelPO po) | 增加酒店的持久化对象 | |
| HotelDataService.  modifyHotelInfo (HotelPO po) | 修改酒店的持久化对象 | |
| HotelDataService.  getHotelInfo(String hotelAddress) | 根据酒店地址返回单一的酒店持久化对象 | |
| HotelDataService.  findHotels(HotelFilter filter) | 系统返回该筛选条件下的所有酒店vo | |
| User.add (UserVO vo) | 增加酒店工作人员的持久化对象 | |
| Comment.getComments(String hotelAddress) | 根据酒店地址返回该酒店的所有评论的持久化对象 | |

* + - 1. **promotionbl模块的接口规范**

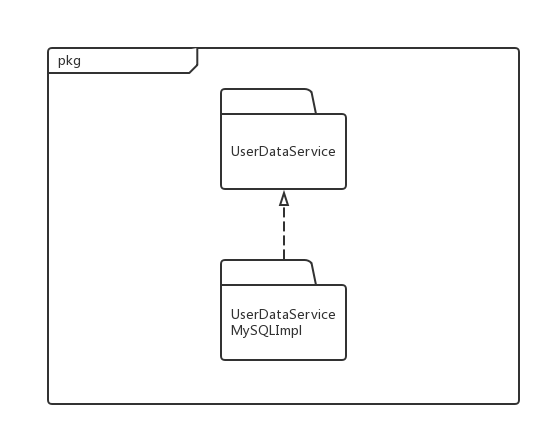
|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Promotion.add | 语法 | public ResultMessage add(PromotionVO vo) |
| 前置条件 | 系统登录账户为酒店工作人员或网站营销人员 |
| 后置条件 | 系统持久化增加该促销策略数据 |
| Promotion.getPromotionBy Hotel | 语法 | public ArrayList<PromotionVO> getPromotionByHotel (String hotelAddress) |
| 前置条件 | 输入的地址名称符合文本规则 |
| 后置条件 | 系统根据酒店地址返回符合该酒店订单的所有促销策略（包括酒店促销策略与网站促销策略） |
| Promotion.  showHotelPromotion | 语法 | public ArrayList<PromotionVO> showHotelPromotion (String hotelAddress) |
| 前置条件 | 输入的地址名称符合文本规则 |
| 后置条件 | 系统根据酒店地址返回该酒店的所有促销策略 |
| Promotion.  showWebsitePromotion | 语法 | public ArrayList<PromotionVO> showWebsitePromotion() |
| 前置条件 | 无 |
| 后置条件 | 系统返回所有网站促销策略 |
| Promotion.modify | 语法 | public ResultMessage modify (PromotionVO vo) |
| 前置条件 | 填写的内容符合规范且齐全 |
| 后置条件 | 系统修改该策略并持久化存储该策略数据 |
| Promotion.delete | 语法 | public ResultMessage delete (PromotionVO vo) |
| 前置条件 | 无（删除表明该数据必然是存在的） |
| 后置条件 | 系统永久性删除该策略的数据 |
| Promotion.showLevel | 语法 | public LevelVO showLevel () |
| 前置条件 | 无 |
| 后置条件 | 系统返回用户等级信息 |
| Promotion.modifyLevel | 语法 | public ResultMessage modifyLevel (LevelVO vo) |
| 前置条件 | 输入的内容符合规范且齐全 |
| 后置条件 | 系统修改用户等级策略并持久化存储该策略数据 |
| 需要的接口（需接口） | | |
| 服务名 | 服务内容 | |
| PromotionDataService.  add (PromotionPO po) | 根据提供的促销策略po增加持久化对象 | |
| PromotionDataService.  modify (PromotionPO po) | 修改单一促销策略持久化对象 | |
| PromotionDataService.  showHotelPromotion (String hotelAddress) | 根据hotelAddress查找该酒店的所有持久化对象 | |
| PromotionDataService.  showWebsitePromotion () | 查找所有网站营销策略 | |
| PromotionDataService.  delete (PromotionPO po) | 删除单一持久化对象 | |
| PromotionDataService.  showLevel () | 返回用户等级策略的持久化对象 | |
| PromotionDataService.  modifyLevel (LevelPO po) | 修改单一用户等级策略的持久化对象 | |
| PromotionDataService.init() | 初始化持久化数据库 | |
| PromotionDataService.  finish() | 结束持久化数据库的使用 | |

* + - 1. **commentbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Comment.getComments | 语法 | public ArrayList<CommentVO>  getComment(String hotelAddress) |
| 前置条件 | 存在该酒店的评价 |
| 后置条件 | 返回该酒店的所有评价 |
| Comment.addComment | 语法 | public ResultMessage addComment(CommentVO vo) |
| 前置条件 | 评价输入合法 |
| 后置条件 | 系统增加该评价的持久化对象 |
| 需要的接口（需接口） | | |
| 服务名 | 服务内容 | |
| CommentDataService.find(String hotelAddress) | 根据酒店地址返回多个评价的持久化对象 | |
| CommentDataService.insert(Com  mentPO po) | 在数据库中插入单一的评价的持久化对象 | |

## 5.4数据层的分解

数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增、删、改、查。例如，Userbl的服务主要由UserDataService提供。持久化数据将以数据库的形式被保存。以UserDataService为例，数据层模块的具体描述下图所示：



**图11 数据层模块的描述**

### 5.4.1数据层模块的职责

数据层模块职责如下表所示

**表9 数据层模块的职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| Userdataservice | 负责保存用户账户的信息，进行增删改查的操作 |
| OrderDataService | 负责保存各类订单，供读取修改写入 |
| HotelDataservice | 负责保存酒店的信息，供读取修改写入 |
| PromotionDataService | 负责保存促销策略，供读取修改写入 |
| CommentDataService | 负责保存评价信息，供读取和写入 |

### 5.4.2 数据层模块的接口规范

* + - 1. **UserDataService模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserDataService.add | 语法 | public ResultMessage add (UserPO po) throws RemoteException |
| 前置条件 | 同样的账户在数据库中不存在 |
| 后置条件 | 在数据库中增加一个UserPO记录 |
| UserDataService.findbyID | 语法 | public UserPO findbyID(String ID) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的UserPO记录 |
| UserDataService.  findbyUserName | 语法 | public UserPO findbyUserName(String userName) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按UserName进行查找返回相应的UserPO记录 |
| UserDataService.modify | 语法 | public ResultMessage modify (UserPO po) throws RemoteException |
| 前置条件 | 在数据库中存在同样的PO |
| 后置条件 | 更新一个PO |
| UserDataService.Check | 语法 | public ResultMessage Check(String userName, String password) throws RemoteException |
| 前置条件 | 账号、密码合法 |
| 后置条件 | 系统检查是否存在账户账户密码是否对应 |
| UserDataService.  addCreditRecord | 语法 | public ResultMessage addCreditRecord(CreditRecordPO po) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 在数据库中增加一个CreditRecordPO记录 |
| UserDataService.  findCreditRecord | 语法 | public CreditRecordPO findCreditRecord (String ID) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的CreditRecordPO记录 |
| UserDataService.init | 语法 | public void init() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| UserDataService.finish | 语法 | public void finish() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用 |

* + - 1. **orderDataService模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderDataService.  getOrderPO | 语法 | public OrderPO getOrderPO(String orderID) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 系统返回一条订单信息 |
| OrderDataService.  getUserOrderPO | 语法 | public ArrayList<OrderPO> getUserOrderPO (String userID, OrderType type) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回某客户某种类全部订单po |
| OrderDataService.  getHotelOrderPO | 语法 | public ArrayList<OrderPO> getHotelOrderPO (String hotelID, OrderType type) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回某酒店某种类全部订单po |
| OrderDataService.  getAbnormalOrderPO | 语法 | public ArrayList<OrderPO> getAbnormalOrderPO (Date date) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回当日全部异常订单po |
| OrderDataService.  add | 语法 | public ResultMessage add (OrderPO po) throws RemoteException |
| 前置条件 | 该订单在数据库中不存在 |
| 后置条件 | 系统返回添加结果记录 |
| OrderDataService.  modify | 语法 | public ResultMessage modify (OrderPO po) throws RemoteException |
| 前置条件 | 在数据库中存在同样订单号的po |
| 后置条件 | 更新一个po |
| OrderDataService.init | 语法 | public void init() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| OrderDataService.finish | 语法 | public void finish() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用 |

* + - 1. **HotelDataService**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelDataService.init | 语法 | public void initial() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| HotelDataService.addHotelInfo | 语法 | public ResultMessage  addHotelInfo(HotelPO po) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 在数据库中插入一个po |
| HotelDataService.modifyHotelInfo | 语法 | public ResultMessage  modifyHotelInfo(HotelPO po) throws RemoteException |
| 前置条件 | 数据库中存在一个相同地址信息的po |
| 后置条件 | 在数据库中更新一个po |
| HotelDataService.getHotelInfo | 语法 | public ResultMessage getHotelInfo(String hotelAddress) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 在数据库中增加订单的流转信息 |
| HotelDataService.findHotels | 语法 | public ResultMessage findHotels(HotelFilter filter) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 在数据库中增加订单的流转信息 |
| HotelDataService.finish | 语法 | public void finish() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用 |

* + - 1. **PromotionDataService模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionDataService.  add | 语法 | Public ResultMessage add (PromotionPO po) throws RemoteException |
| 前置条件 | 同样的PromotionPO在数据库中不存在 |
| 后置条件 | 在数据库中增加一个PromotionPO记录 |
| PromotionDataService.  modify | 语法 | Public ResultMessage modify (PromotionPO po) throws RemoteException |
| 前置条件 | 在数据库中存在同样的PO |
| 后置条件 | 更新一个PO |
| PromotionDataService.  showHotelPromotion | 语法 | Public ArrayList<PromotionPO> showHotelPromotion (String hotelAddress) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按hotelAddress进行查找返回该酒店的PromotionPO记录 |
| PromotionDataService.  showWebsitePromotion | 语法 | Public ArrayList<PromotionPO> showWebsitePromotion () throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回网站策略的PromotionPO记录 |
| PromotionDataService.  delete | 语法 | Public ResultMessage delete (PromotionPO po) throws RemoteException |
| 前置条件 | 在数据库中存在同样的PO |
| 后置条件 | 删除一个PO |
| PromotionDataService.  showLevel | 语法 | Public LevelPO showLevel () throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回LevelPO记录 |
| PromotionDataService.  modifyLevel | 语法 | Public ResultMessage modifyLevel (LevelPO po) throws RemoteException |
| 前置条件 | 在数据库中存在同样的PO |
| 后置条件 | 更新一个PO |
| PromotionDataService.  init | 语法 | Public void init() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| PromotionDataService.  finish | 语法 | Public void finish() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用 |

* + - 1. **CommentDataService**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CommentDataService.init | 语法 | Public void initial() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| CommentDataService.find | 语法 | Public CommentPO  find(String hotelAddress) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按地址进行查找返回相应的CommentPO结果 |
| CommentDataService.insert | 语法 | public ResultMessage  insert(CommentPO po) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 在数据库中插入一个po |
| CommentDataService.finish | 语法 | public void finish() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库的使用 |

# 6.信息视角

## 6.1数据持久化对象

系统的PO类就是对应的相关的实体类，在此只做简单介绍：

* UserPO类为用户信息，包含用户名、用户ID、密码、用户类型、其他个人信息属性
* OrderPO类为订单信息，包含开始时间、退房时间、最晚订单执行时间、房间类型及数量、预计入住人数、有无儿童、订单状态、撤销时间（已撤销状态订单）。入住酒店名称、酒店地址、订单号属性
* HotelPO类为酒店信息，包含地址、所属商圈、简介、设施服务、价格、星级、评分、可用客房信息（可用客房的类型、数量、原始价格）、房间入住信息（房间号、入住时间、预计离开时间）属性
* PromotionPO类为促销策略，包含折扣额度、适用日期、适用人员、适用条件属性
* LevelPO类为会员等级制度，包含每一等级所需信用值属性
* CommentPO类为酒店评价，包含评价人，评价时间，用户打分，具体评价内容
* CreditRecordPO为信用记录类，包含信用记录的时间，订单号，动作（订单执行、订单异常、订单撤销、充值），信用度变化、信用度结果属性

持久化用户对象LevelPO类的定义如图所示：



## 6.2数据库表

本系统使用数据库来存储文件，数据库表包含User表、Order表、Hotel表、Promotion表、Level表、Comment表、City表、Area表、Credit表