酒店预订系统

RSH(Reservation System of Hotel)

体系结构设计文档

**V2.1**最终版

南京大学CMBT工作组

2017-01-01

更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 吕丹  沈丽婷  王馨雨  蔚滢璐 | 2016-10-8 | 最初草稿 | V1.0 草稿 |
| 吕丹  沈丽婷  王馨雨  蔚滢璐 | 2016-10-10 | 审核汇总 | V1.1阶段报告 |
| 吕丹  沈丽婷  王馨雨  蔚滢璐 | 2016-10-12 | 增添用户界面层的分解 | V1.2.0  阶段成果 |
| 吕丹  沈丽婷  王馨雨  蔚滢璐 | 2016-10-14 | 增添信息视角 | v1.2.2.1  阶段成果 |
| 吕丹  沈丽婷  王馨雨  蔚滢璐 | 2016-10-16 | 增添业务逻辑层和数据层的分解 | V1.2.2.2  阶段成果 |
| 吕丹  沈丽婷  王馨雨  蔚滢璐 | 2016-10-16 | 完善接口视角，排版，校对 | V2.0  正式版 |
| 王馨雨 | 2016-12-25 | 根据实际情况更新接口 | v2.1 |
| 王馨雨 | 2017-01-01 | 修改层之间调用的接口 | v2.2最终版 |

目录

[1.引言 5](#_Toc471113646)

[1.1 编制目的 6](#_Toc471113647)

[1.2词汇表 6](#_Toc471113648)

[1.3 参考资料 6](#_Toc471113649)

[2.产品描述 6](#_Toc471113650)

[3.逻辑视角 7](#_Toc471113651)

[4.组合视角 8](#_Toc471113652)

[4.1开发包图 8](#_Toc471113653)

[4.2运行时进程 13](#_Toc471113654)

[4.3物理部署 14](#_Toc471113655)

[5.接口视角 15](#_Toc471113656)

[5.1 模块的职责 15](#_Toc471113657)

[5.2 用户界面层的分解 17](#_Toc471113658)

[5.2.1 用户界面层模块的职责 18](#_Toc471113659)

[5.2.2 用户界面模块的接口规范 18](#_Toc471113660)

[5.2.3 用户界面模块设计原理 19](#_Toc471113661)

[5.3业务逻辑层的分解 19](#_Toc471113662)

[5.3.1 业务逻辑层模块的职责 20](#_Toc471113663)

[5.3.2 业务逻辑模块的接口模范 20](#_Toc471113664)

[5.4 数据层的分解 39](#_Toc471113665)

[5.4.1 数据层模块的职责 40](#_Toc471113666)

[5.4.2 数据层模块的接口规范 41](#_Toc471113667)

[6.信息视角 51](#_Toc471113668)

[6.1 数据持久化对象 51](#_Toc471113669)

[6.2 Txt持久化格式 52](#_Toc471113670)

[6.3 数据库表 53](#_Toc471113671)

# 1.引言

## 1.1 编制目的

本报告详细完成对酒店预订系统的概要设计，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 |
| Order | 订单 |
| Search | 搜索 |
| Hotel | 酒店 |
| webstaff | 网站工作人员 |
| Bl | 业务逻辑 |
| Promotion | 促销 |

## 1.3 参考资料

《软件工程与计算（卷二）软件开发的技术基础》

《酒店预订系统用例文档V1.0》

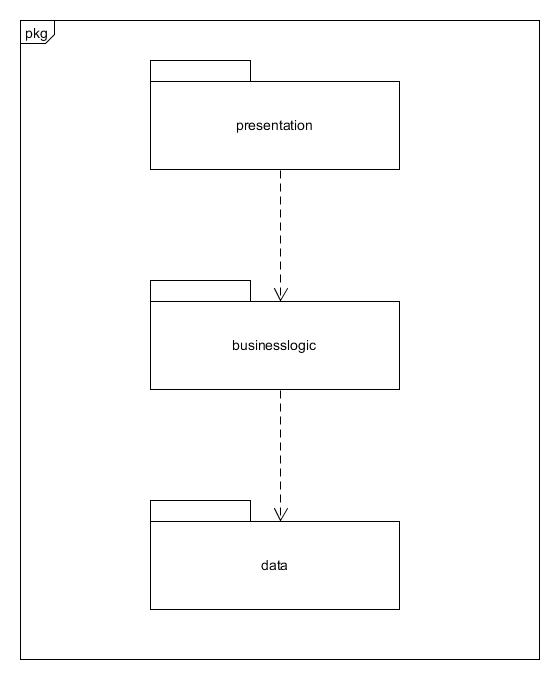
《酒店预订系统需求规格说明文档V1.0》

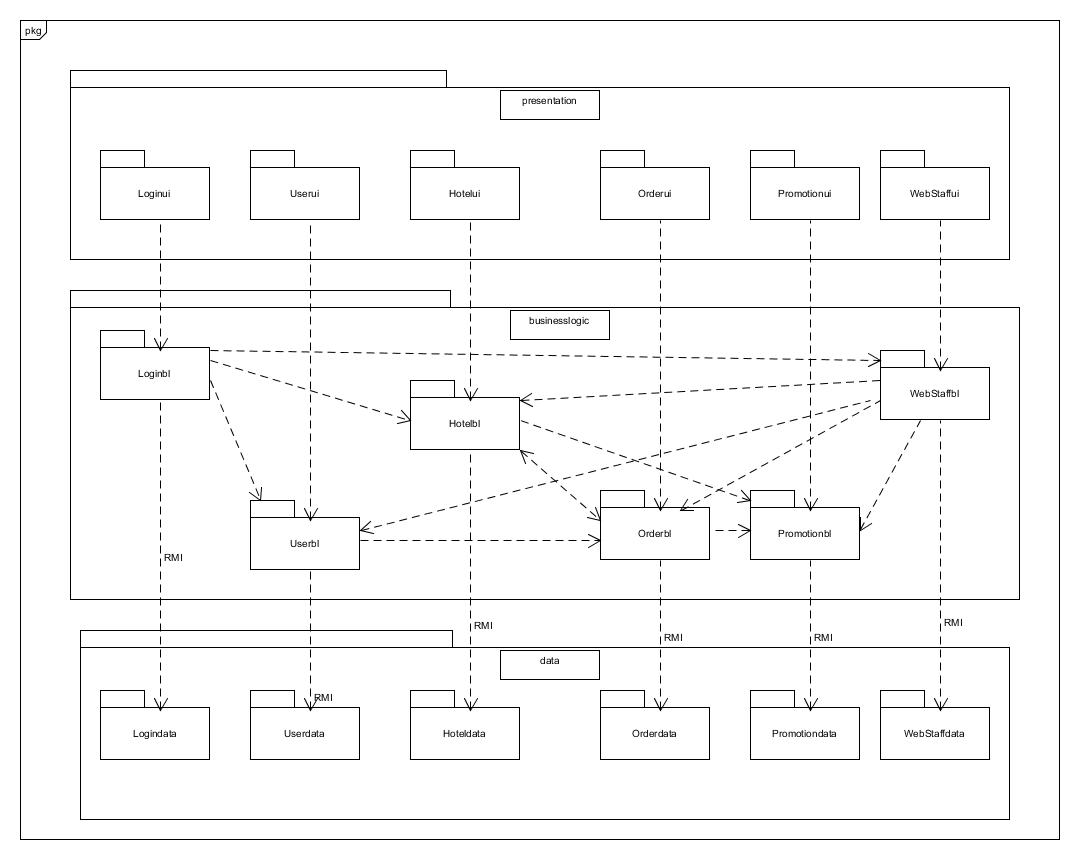
# 2.产品描述

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**：软件体系结构逻辑设计方案

# 4.组合视角

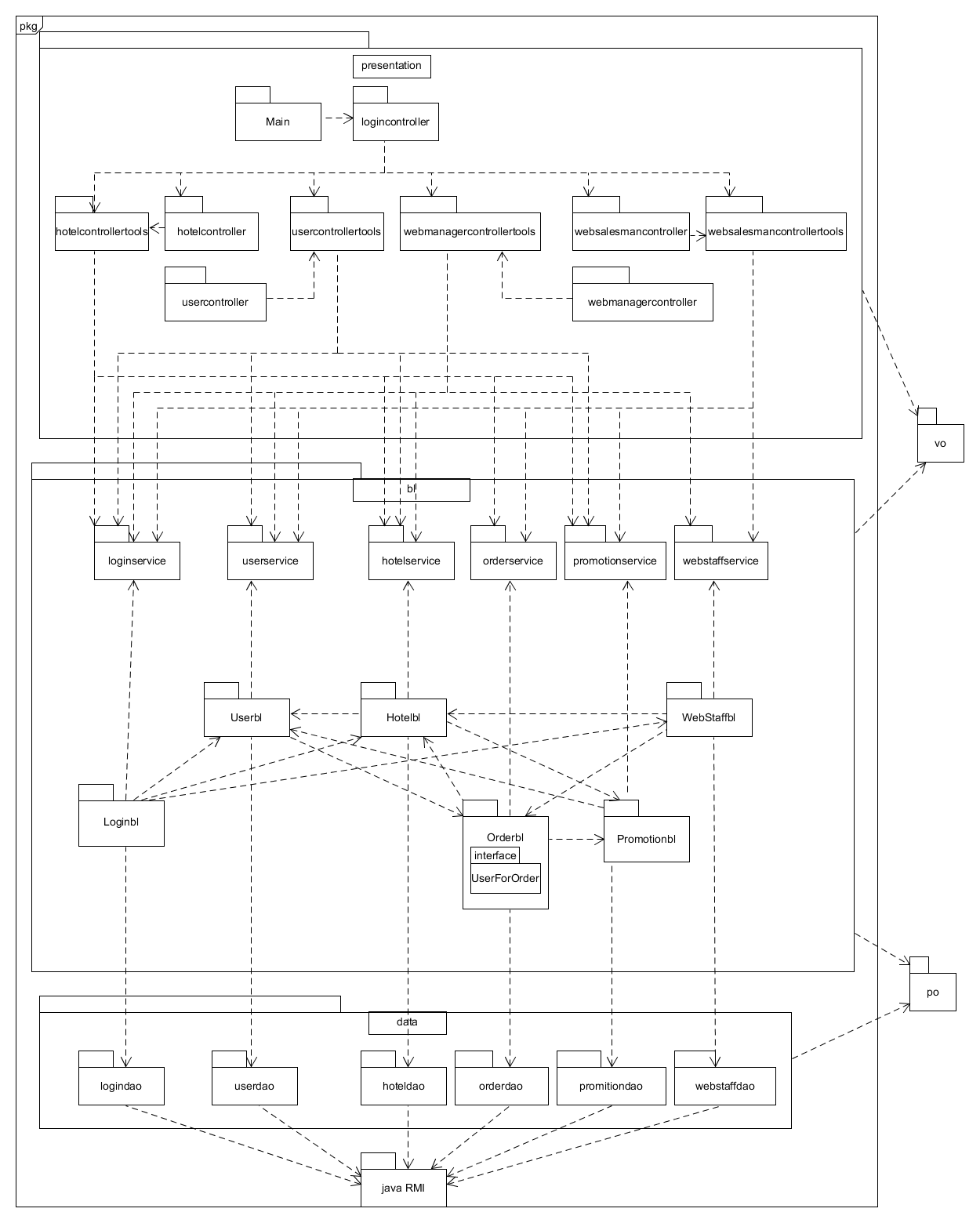
## 4.1开发包图

酒店预订管理系统的最终开发包设计如表1所示：

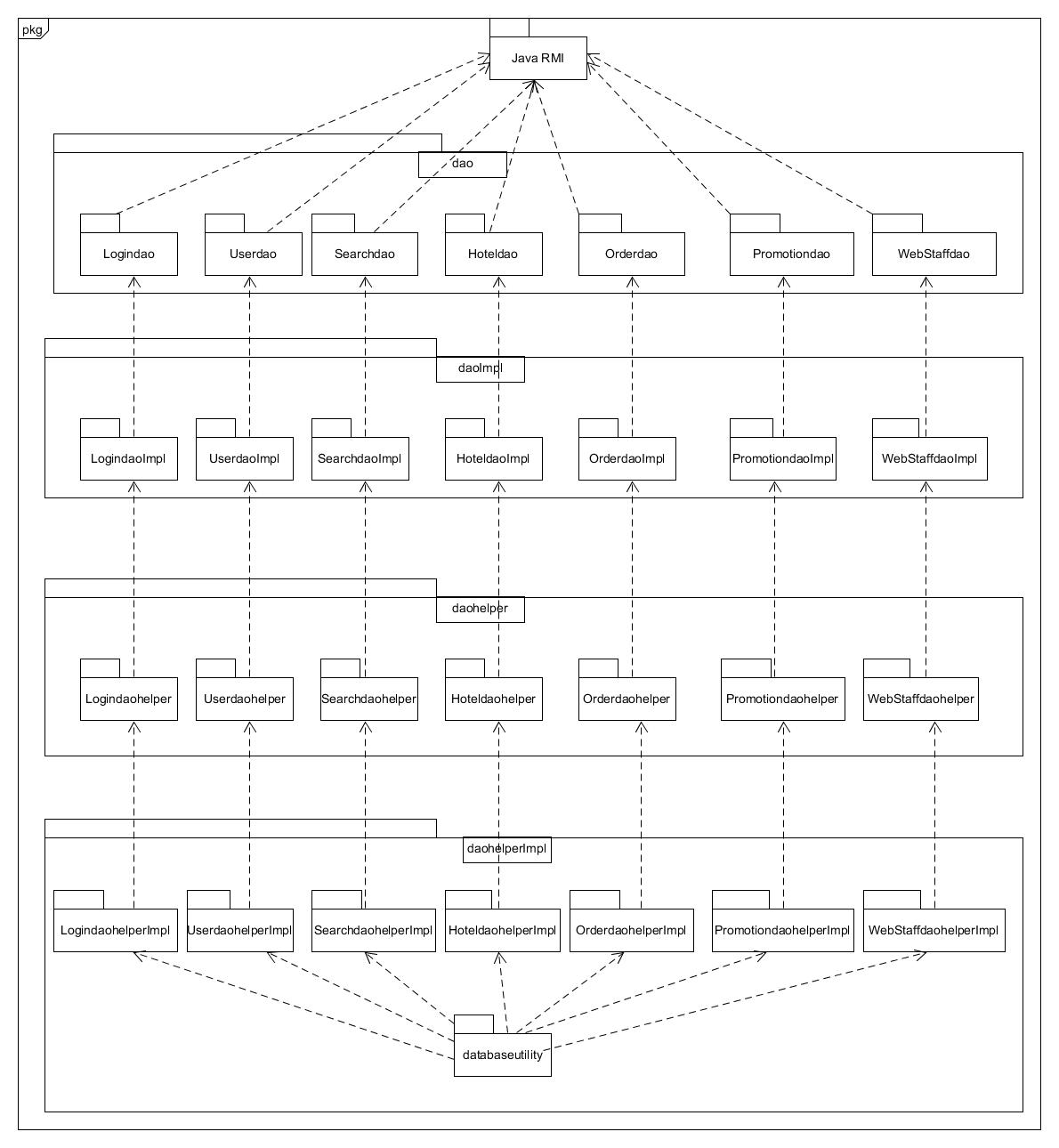
表**1**：酒店预订系统的最终开发包设计

|  |  |
| --- | --- |
| 开发（物理）包 | 依赖的其他开发包 |
| main | logincontroller |
| logincontroller | hotelcontrollertools,hotelcontroller,usercontrollertools,webmanagercntrollertools,websalesmancontrollertools,websalesmancontroller,vo |
| usercontroller | usercontrollertools,界面类库包,vo |
| hotelcontroller | hotelcontrollertools,界面类库包,vo |
| websalesmancontroller | websalesmancontrollertools,界面类库包,vo |
| webmanagercontroller | webmanagercontrollertools,界面类库包,vo |
| usercontrollertools | loginservice,userservice,hotelservice,promotionservice， vo |
| hotelcontrollertools | loginservice,orderservice,hotelservice,promotionservice， vo |
| webmanagercontrollertools | loginservice,userservice,hotelservice,webstaffservice， vo |
| websalesmancontrollertools | webstaffservice,userservice,hotelservice,promotionservice，orderservice, vo |
| loginservice |  |
| loginbl | loginservice,logindao,po,userbl,hotelbl,webstaffbl |
| logindao | JavaRMI,po |
| userservice |  |
| userbl | userblservice,userdao,po,orderbl |
| userdao | JavaRMI,po |
| hotelservice |  |
| hotelbl | hotelservice,hoteldao,po,userbl, promotionbl |
| hoteldao | JavaRMI,po |
| orderservice |  |
| orderbl | orderservice,orderdao,po,promotionbl,hotelbl |
| orderdao | JavaRMI,po |
| promotionservice |  |
| promotionbl | promotionservice,promotiondao,po,userbl |
| promotiondao | JavaRMI,po |
| webstaffservice |  |
| webstaffbl | webstaffservice,webstaffdao,po, orderbl,hotelbl,  userbl |
| webstaffdao | JavaRMI,po |
| logindaoimpl | logindao |
| logindaohepler | logindaoimpl |
| logindaoheplerimpl | logindaohepler,databaseutility |
| userdaoimpl | userdao |
| userdaohepler | userdaoimpl |
| userdaoheplerimpl | userdaohepler,databaseutility |
| hoteldaoimpl | hoteldao |
| hoteldaohepler | hoteldaoimpl |
| hoteldaoheplerimpl | hoteldaohepler,databaseutility |
| orderdaoimpl | orderdao |
| orderdaohepler | orderdaoimpl |
| orderdaoheplerimpl | orderdaohepler,databaseutility |
| promotiondaoimpl | promotiondao |
| promotiondaohepler | promotiondaoimpl |
| promotiondaoheplerimpl | promotiondaohepler,databaseutility |
| webstaffdaoimpl | webstaffdao |
| webstaffdaohepler | webstaffdaoimpl |
| webstaffdaoheplerimpl | webstaffdaohepler,databaseutility |
| Vo |  |
| po |  |
| utilitybl |  |
| 界面类库包 |  |
| Java RMI |  |
| databaseutility | JDBC |

酒店预订系统客户端开发包图如图3所示，服务器端开发包图如图4所示

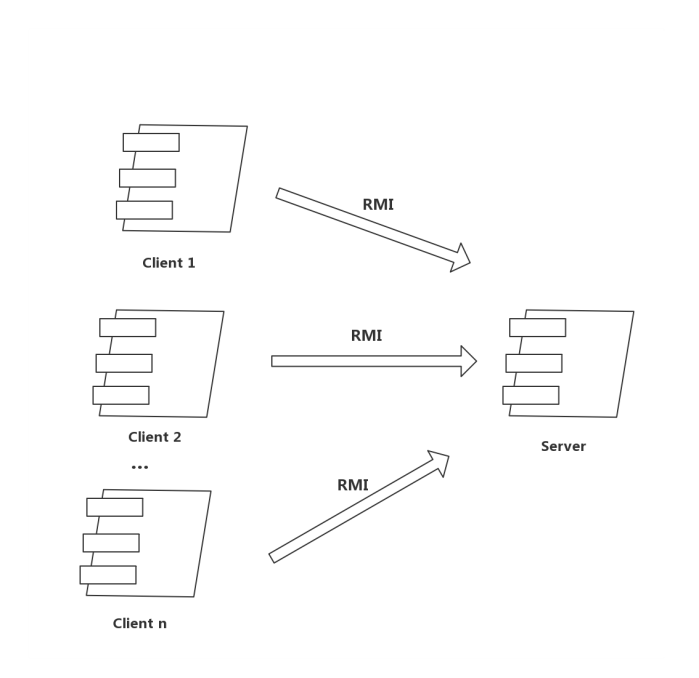


图**3**：客户端开发包图

****

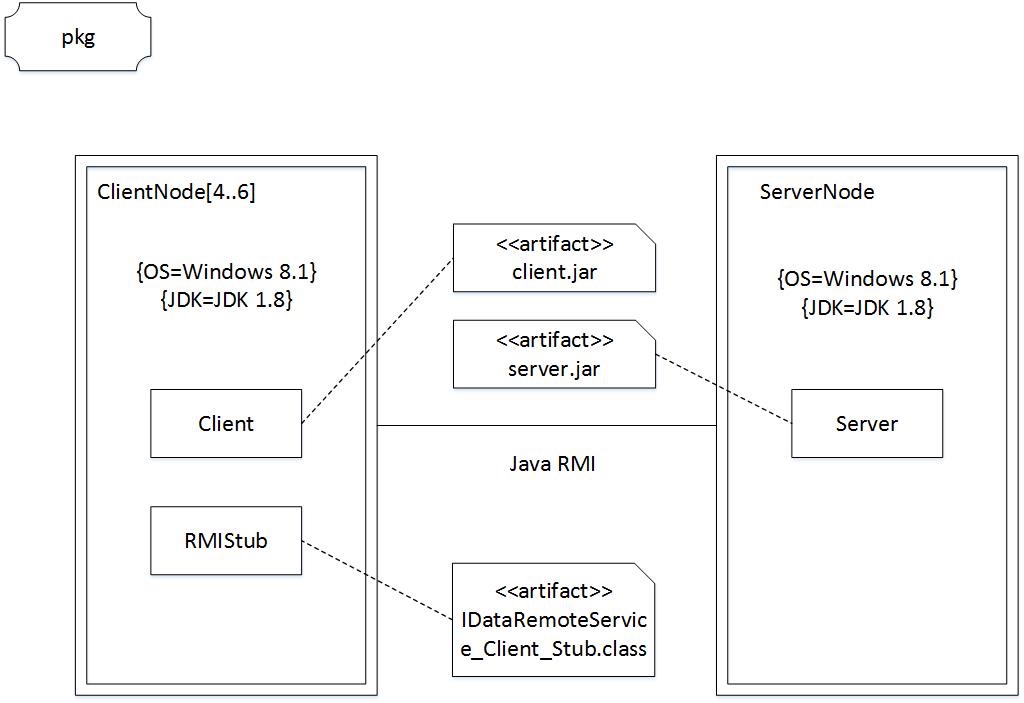
图**4**：服务器端开发包图

## 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。在客户端节点上，还要部署RMIStub构件。由于Java RMI 构件属于JDK 1.8的一部分。所以，在系统JDK环境已经设置好的情况下，不需要独立部署。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

客户端模块视图和服务器端模块视图分别如图7和图8所示。客户端和服务器端各层的职责

分别如表2和表3所示。

启动模块

展示层

业务逻辑层

网络模块

图**7:** 客户端模块式图

启动模块

业务逻辑层

网络模块

图**8:**服务器模块式图

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 | 基于窗口的酒店预订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 |

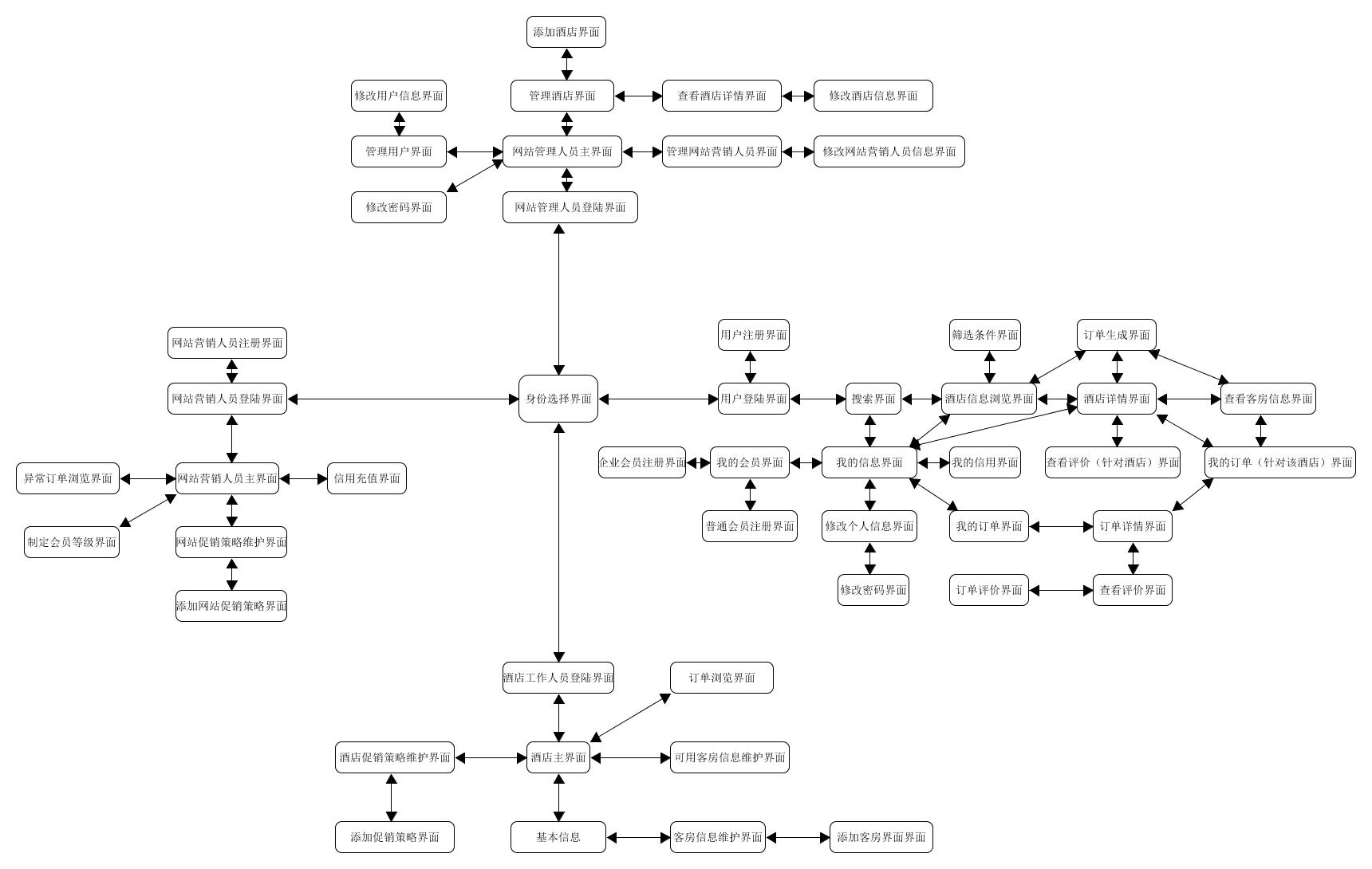
每一层只是使用下方直接接触的层。层与层之间仅仅是通过接口的调用来完成的。层之间调用的接口如表4所示。

表**4** 层之间调用的接口

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| HotelInfoService  HotelService  ManagerHotelService  SearchHotelService  LoginService  OrderForHotel  OrderForUser  OrderForWebsite  PromotionService  UserService  WebStaffService | 客户端展示层 | 客户端业务逻辑层 |
| HotelDao  LoginDao  OrderDao  PromotionDao  CreditRecordListDao  UserDao  WebManagerDao  WebSalesmanDao | 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35个用户界面：身份选择界面、登录界面、用户注册界面、用户登录界面、用户主界面、用户信息维护、会员注册界面、信用记录浏览界面、搜索界面、酒店浏览界面、酒店详情界面、订单生成界面、订单浏览界面、订单详情界面、订单评价界面、酒店主界面、酒店信息维护界面、客房信息维护界面、可用客房信息维护界面、订单搜索并浏览界面、酒店促销策略维护界面、网站营销人员主界面、网站促销策略维护界面、异常订单浏览界面、信用充值界面、网站管理人员主界面、用户信息维护界面、酒店信息浏览界面、酒店详情界面、网站营销人员信息维护界面

****

图**9:** 酒店预订系统界面跳转图

### 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

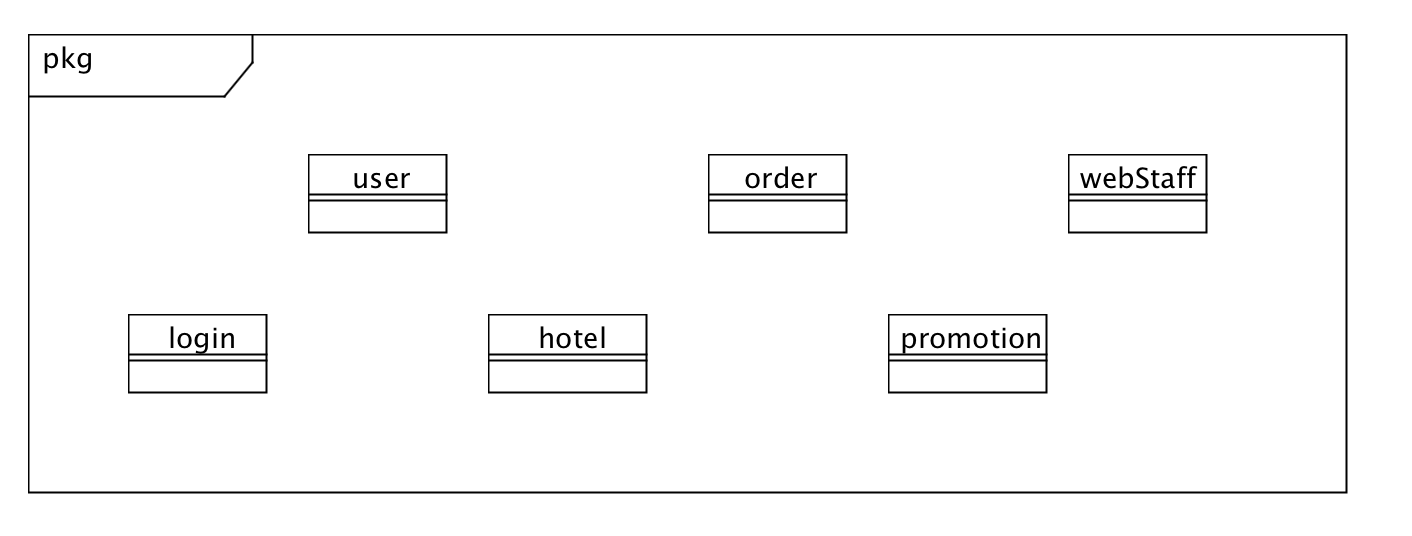
|  |  |
| --- | --- |
| 服务名 | 服务 |
| businesslogicservice.\*blservice | 每个ui模块都有一个相应的业务逻辑接口 |

### 5.2.3 用户界面模块设计原理

用户界面利用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。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

|  |  |
| --- | --- |
| 模块 | 职责 |
| login | 实现注册账号与登录的服务，为其他包提供注册账号与登陆的接口 |
| user | 实现对用户基本信息的增删改查,为其他包提供操作用户信息的接口 |
| hotel | 实现对酒店自身信息和酒店工作人员信息增删改查，为其他包提供操作酒店信息的接口 |
| order | 实现对订单信息的增删改查，为其他包提供操作订单信息的接口 |
| promotion | 实现对促销策略的增删改查，为其他包提供操作促销策略信息的接口 |
| webstaff | 实现对网站营销人员信息的增删改查 |

### 5.3.2 业务逻辑模块的接口模范

#### 5.3.2.1 login模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| login.register | 语法 | | public PersonVOregister(Role role) |
| 前置条件 | | 人员选择身份 |
| 后置条件 | | 系统增加该账号的持久化对象 |
| login.checkOnline | 语法 | | public ResultMessage checkOnline(Role role, String id, String password) |
| 前置条件 | | 身份确认后，输入的账号、密码符合语法规则 |
| 后置条件 | | 系统显示登录验证结果。若账号与密码相匹配，系统更新人员登录记录。 |
| login.logout | 语法 | | public ResultMessage logout(Role role, String id) |
| 前置条件 | | 人员已登陆并选择退出登录 |
| 后置条件 | | 系统更新人员登录记录。 |
| 需要的接口（需接口） | | | |
| 服务名 | | 服务内容 | |
| user.add(UserVO uservo) | | 增加单一持久化对象 | |
| user.checkPassword(String id, String password) | | 检查密码是否正确 | |
| hotel.checkPassword(String id, String password) | | 检查密码是否正确 | |
| webStaff.checkPassword(String id, String password) | | 检查密码是否正确 | |
| logindao.addOnline(OnlinePersonPO po) | | 增加在线人员持久化对象 | |
| logindao.deleteOnline(Role role,String id) | | 删除在线人员持久化对象 | |

#### 5.3.2.2 user模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| user.add | | 语法 | public ResultMessage add(UserVO userVO) |
| 前置条件 | 用户完成注册 |
| 后置条件 | 系统在数据库中为该用户填入已有信息，并初始化 |
| user.checkPassword | | 语法 | public ResultMessage checkPassword(String id, String password) |
| 前置条件 | 用户输入的账号、密码符合语法规则 |
| 后置条件 | 系统返回验证结果 |
| user.getInfo | | 语法 | public UserVO getInfo(String userid) |
| 前置条件 | 该账号存在 |
| 后置条件 | 返回用户值对象 |
| user.update | | 语法 | public ResultMessage update(UserVO userVO) |
| 前置条件 | 无 |
| 后置条件 | 系统根据值对象更新该账号的持久化对象 |
| user.addCreditRecord | | 语法 | public ResultMessage addCreditRecord(CreditRecordVO vo) |
| 前置条件 | 订单状态发生变化导致信用记录变化 |
| 后置条件 | 在数据库中增加一条信用记录 |
| user. getCreditRecordList | | 语法 | public Iterator<CreditRecordVO> getCreditRecordList(String userid) |
| 前置条件 | 该用户id存在 |
| 后置条件 | 返回信用记录列表 |
| user.registerMember | | 语法 | public ResultMessage registerMember(String userID) |
| 前置条件 | 该用户id存在 |
| 后置条件 | 将该用户的属性改为普通会员 |
| user.RegisterMember | | 语法 | public ResultMessage registerMember(String userID, String commerceName) |
| 前置条件 | 该用户id存在 |
| 后置条件 | 将该用户的属性改为企业会员 |
| user.setMemberStandard | | 语法 | public ResultMessage setMemberStandard(int boundaryForLevel) |
| 前置条件 | 无 |
| 后置条件 | 更新数据库中的会员等级标准 |
| user.getMemberStandard | | 语法 | public int getMemberStandard() |
| 前置条件 | 无 |
| 后置条件 | 返回会员等级标准 |
| user.getMemberLevel | | 语法 | public int getMemberLevel(int credit) |
| 前置条件 | 无 |
| 后置条件 | 返回会员等级 |
| user.changePassword | | 语法 | public ResultMessage changePassword(String userid, String oldPassword, String newPassword) |
| 前置条件 | 该用户已存在 |
| 后置条件 | 在数据库更该该用户的密码 |
| user.addCredit | | 语法 | public ResultMessage addCredit(int value, String userID) |
| 前置条件 | 该用户已存在 |
| 后置条件 | 为该用户充值信用 |
| user.hasReserved | | 语法 | public boolean hasReserved(String userID, String hotelID) |
| 前置条件 | 该用户和酒店存在 |
| 后置条件 | 返回该用户是否曾经预定过这个酒店 |
| user. canGenerateOrder | | 语法 | public boolean canGenerateOrder() |
| 前置条件 | 用户下订单 |
| 后置条件 | 返回该用户是否可以下订单 |
| 需要的接口（需接口） | | | |
| 服务名 | 服务内容 | | |
| userdao. getInfo(String id) | 返回用户单一持久化对象 | | |
| userdao. update(UserPO po) | 更新用户单一持久化对象 | | |
| userdao. addCreditRecord(CreditRecordPO po) | 增加信用记录持久化对象 | | |
| userdao. getCreditRecordList(String userid) | 返回用户信用记录列表 | | |
| userdao.setMemberLevel(int boundryForLevels) | 制定会员等级策略 | | |
| userdao. getMemberLevel() | 得到会员等级标准 | | |
| userdao.getAll() | 网站管理人员得到用户信息 | | |
| userdao.insert(UserPO userPO) | 增加用户持久化对象 | | |
| order.getOrderStateOfUser(String userID, String hotelID) | 返回该用户在酒店的最近订单的状态 | | |

#### 5.3.2.3 hotel模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| hotel.getHotelInfo | 语法 | public HotelVO getHotelInfo(String hotelID) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回该酒店的酒店基本信息 |
| hotel.updateHotel | 语法 | public ResultMessage updateHotel(HotelVO vo) |
| 前置条件 | 酒店工作人员输入的信息正确 |
| 后置条件 | 系统更新数据库中该酒店的酒店基本信息 |
| hotel.getRoomList | 语法 | public ArrayList<RoomVO> getRoomList(String hotelID) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回该酒店的客房信息列表 |
| hotel.addSpecialRoom | 语法 | public ResultMessage addSpecialRoom(RoomVO vo); |
| 前置条件 | 无 |
| 后置条件 | 在可用客房数据库中增加该特色可用客房数据 |
| hotel.getImageAddresses | 语法 | public ArrayList<String> getImageAddresses(String hotelID) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回该酒店的所有图片url列表 |
| hotel.getImageAddresses | 语法 | public String getImageAddress(String hotelID, String roomType) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回该酒店的该客房类型的图片url |
| hotel.getRoomAvailList | 语法 | public ArrayList<RoomAvailVO> getRoomAvailList(String hotelID, Date checkIn) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回该酒店的从checkin这天开始的可用客房信息列表 |
| hotel.plusRoomAvail | 语法 | public ResultMessage plusRoomAvail(String hotelID, String roomType, int num, Date checkIn, Date checkOut) |
| 前置条件 | 该房间存在 |
| 后置条件 | 增加该酒店该期间可用房间数量 |
| hotel.minusRoomAvail | 语法 | public ResultMessage minusRoomAvail(String hotelID, String roomType, int num, Date checkIn, Date checkOut) |
| 前置条件 | 该房间存在 |
| 后置条件 | 减少该酒店该期间可用房间数量 |
| hotel.numOfRoomAvail | 语法 | public int numOfRoomAvail(String hotelID, String roomType, Date checkIn, Date checkOut) |
| 前置条件 | 该客房类型存在 |
| 后置条件 | 返回这段时间内该客房类型的可用房间数目 |
| Hotel.updateAvailableRoomList | 语法 | Public ResultMessage updateAvailableRoomList(ArrayList<RoomAvailableVO> availableRoomList); |
| 前置条件 | 用户生成订单或退房 |
| 后置条件 | 更新该日期下的可用客房信息列表 |
| Hotel.getRoomType | 语法 | Public ArrayList<String> getRoomType() |
| 前置条件 | 已经从数据库中取得该酒店的可用客房信息列表 |
| 后置条件 | 返回该酒店的所有房间类型 |
| hotel.checkPassworld | 语法 | public static ResultMessage checkPassword(String hotelID, String password) |
| 前置条件 | id和密码格式正确 |
| 后置条件 | 查找是否存在相应的id，根据输入的password返回登陆验证的结果 |
| hotel.getRoomNorm | 语法 | public ArrayList<RoomNormVO> getRoomNorm(String hotelID) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回该酒店的房间规格列表 |
| hotel.getCheckInDDL | 语法 | public String getCheckInDDL(String hotelID) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回该酒店的最晚入住时间 |
| hotel.getCheckInDDL | 语法 | public String getCheckInDDL(String hotelID) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回该酒店的最晚入住时间 |
| hotel.updateGrade | 语法 | public ResultMessage updateGrade(String hotelID,int grade) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 更新数据库中该酒店的评分 |
| hotel. getHotelID | 语法 | public String getHotelID(String district) |
| 前置条件 | 地区转化成6位编码 |
| 后置条件 | 根据6位地区编码得到酒店id |
| hotel. addHotel | 语法 | public ResultMessage addHotel(HotelVO hotelVO) |
| 前置条件 | （初始化密码后，点击确认）注册成功，生成酒店 |
| 后置条件 | 增加酒店信息持久化对象 |
| hotel.getHotelList | 语法 | public ArrayList<HotelVO> getHotelList(String province, String city, String area) |
| 前置条件 | 该省市区存在 |
| 后置条件 | 返回符合该地址和商圈的经过综合排序的酒店列表 |
| hotel.Sort | 语法 | public ArrayList<HotelVO> sort(ArrayList<HotelVO> hotelVOs, SortBy sortBy, SortMethod sortM) |
| 前置条件 | 无 |
| 后置条件 | 返回排序后的酒店列表 |
| hotel.Select | 语法 | public ArrayList<HotelVO> select(ArrayList<HotelVO> hotelList,SelectConditionVO selectConditionVO) |
| 前置条件 | 已经从数据库中取得酒店列表 |
| 后置条件 | 返回筛选后的酒店列表 |
| hotel.Select | 语法 | public ArrayList<HotelVO> select(ArrayList<HotelVO> hotelList,String hotelName) |
| 前置条件 | 已经从数据库中取得酒店列表 |
| 后置条件 | 返回筛选后的酒店列表 |
| hotel. getHotelList | 语法 | public ArrayList<HotelVO> getHotelList() |
| 前置条件 | 无 |
| 后置条件 | 返回该所有酒店的信息列表 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| hoteldao.getHotel（String id） | 得到根据酒店id得到酒店单一持久化对象 | |
| hoteldao. getNewHotelID(String district) | 根据酒店传入地区得到初始化酒店账号 | |
| hoteldao. addHotel(HotelPO hotelPO) | 添加酒店持久化对象 | |
| hoteldao.updateGrade(String hotelID,int grade) | 更新酒店评分 | |
| hoteldao. updateHotel (HotelPO hotelPO) | 更新酒店持久化对象 | |
| hoteldao. addSpecialRoom(RoomPO roomPO) | 根据roompo里的酒店账号 添加酒店的客房信息 并将该房间类型可用客房数量置为总数 | |
| hoteldao. deleteSpecialRoom(RoomPO roomPO) | 根据roompo里的酒店账号 删除酒店的客房信息 | |
| hoteldao. getRoomList(String hotelID) | 根据酒店ID 得到酒店的所有房间的信息 | |
| hoteldao. updateRoom(RoomPO roomPO) | 根据roompo里的酒店账号 更新该房间类型的信息 | |
| hoteldao. getRoomAvailList(String hotelID, Date checkIn) | 根据酒店ID、日期 得到酒店在该日期下的所有客房的可用信息 | |
| hoteldao. changeRoomAvail(String id, String roomType,boolean isPlus, int num, Date checkIn, Date checkOut) | 更改酒店房间类型房间可用数量的信息 | |
| hoteldao. getHotelList(String district) | 得到一个地区酒店持久化对象列表 | |
| hoteldao. getAll() | 得到酒店持久化对象列表 | |
| hoteldao.getImageAddresses(String hotelID) | 得到该酒店的所有图片的url | |
| user. hasReserved(String userID, String hotelID) | 判断该用户是否在该酒店预定过 | |

#### 5.3.2.4 order模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| order. hotelClassify | 语法 | public ArrayList<OrderVO> hotelClassify(String hotelID, StateOfOrder state) |
| 前置条件 | 酒店工作人员选择了订单状态 |
| 后置条件 | 返回该酒店的该种订单状态的订单列表 |
| order. execute | 语法 | public ResultMessage execute(String orderID) |
| 前置条件 | 酒店执行订单 |
| 后置条件 | 根据订单id，在逻辑层记下该订单实际入住时间（日期＋时间）  更新数据库中该订单信息（状态及实际入住时间）  更新用户信用记录 |
| order.leaveUpdate | 语法 | public ResultMessage leaveUpdate(String orderID) |
| 前置条件 | 用户离开酒店 |
| 后置条件 | 根据订单id，在逻辑层记下该订单实际离开时间（日期＋时间）  更新数据库中该订单信息（实际离开时间） |
| order. hotelCancelAbnormal | 语法 | public ResultMessage hotelCancelAbnormal(String orderID) |
| 前置条件 | 在用户订单预计离开日期之前->用户交付订单实际价值 |
| 后置条件 | 更新订单实际入住时间（日期＋时间）  改变订单状态 ；（abnormal->executed）  恢复用户被扣除的信用值  置为已执行->增加信用值 |
| order.userClassify | 语法 | public ArrayList<OrderVO> userClassify(String userID, StateOfOrder state) |
| 前置条件 | 用户选择订单状态 |
| 后置条件 | 返回该用户的符合该订单状态的订单列表 |
| order.detail | 语法 | public OrderVO detail(String orderID) |
| 前置条件 | 用户查看订单详情 |
| 后置条件 | 返回该订单详情 |
| order.cancelMyOrder | 语法 | public int cancelMyOrder(String orderID) |
| 前置条件 | 用户取消未执行订单 |
| 后置条件 | 用户信用值扣除  酒店可用客房数量增加  订单状态改变 |
| order. getOrderStateOfUser | 语法 | public StateOfOrder getOrderStateOfUser(String userID, String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 返回该用户在酒店的最近订单的状态 |
| order. specificOrder | 语法 | public ArrayList<OrderVO> specificOrder(String userID,String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 返回在该酒店下的所有订单 |
| order. getTrueValue | 语法 | public String getTrueValue(OrderInfo orderInfo) |
| 前置条件 | 选择房间类型、房间数量完成 |
| 后置条件 | 返回得到优惠后的价格以及优惠策略  String#double->promotion#truePrice |
| order. confirmReservation | 语法 | public ResultMessage confirmReservation(OrderVO orderVO) |
| 前置条件 | 数据库中存在状态为异常的订单 |
| 后置条件 | 根据（用户信用值信息）判断是否可以提交,不可提交 返回信息提示. 更新数据库 |
| order. addComment | 语法 | public ResultMessage addComment(String orderID, int grade, String comment) |
| 前置条件 | 用户评价订单,输入评分 |
| 后置条件 | 订单评分评论更新  酒店评分更新 |
| order. browseUnperformed | 语法 | public ArrayList<OrderVO> browseUnperformed() |
| 前置条件 | 网站营销人员浏览未执行订单 |
| 后置条件 | 返回所有未执行订单 |
| order. browseAbnormal | 语法 | public ArrayList<OrderVO> browseAbnormal() |
| 前置条件 | 网站营销人员查看异常订单 |
| 后置条件 | 返回所有异常订单 |
| order. webCancelAbnormal | 语法 | public ResultMessage webCancelAbnormal(String orderID, boolean isHalf) |
| 前置条件 | 用户仍在预计入住时期, 用户取消入住计划 |
| 后置条件 | 改变酒店的房间信息；  增加用户信用值（全部/一半）；  记录撤销订单时间、改变订单状态 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| orderdao. searchByID(String orderID) | | 根据订单编号查找订单 |
| orderdao. searchByUser(String userID) | | 根据用户编号查找订单 |
| orderdao. searchByUserWithHotel(String userID,String hotelID) | | 根据用户编号、酒店编号查找订单 |
| orderdao. searchByHotel(String hotelID) | | 根据酒店编号查找订单 |
| orderdao. searchByState(StateOfOrder state) | | 根据状态编号查找订单 |
| orderdao. insert(OrderPO orderPO) | | 添加订单 |
| orderdao. update(OrderPO orderPO) | | 更新订单 |
| hotel. plusRoomAvail(String hotelId, String roomType,int num, Date checkIn, Date checkOut) | | 线下提前离开，酒店工作人员手动增加可用客房数 |
| hotel. getCheckInDDL(String hotelID) | | 得到该酒店的最晚入住时间 |
| hotel. getRoomAvailNum(String hotelID,String roomType, Date checkIn, Date checkOut) | | 得到某酒店某房间类型特定时间段的可用客房数量 |
| hotel. updateGrade(String hotelID, int grade) | | 更新数据库中酒店的评分 |
| promotion. countPromotionOfRoom(OrderInfo orderInfo) | | 计算订单的价值 |

#### 5.3.2.5 promotion模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| promotion. getIDForNewPromotion | 语法 | String getIDForNewPromotion(String setterID) |
| 前置条件 | 无 |
| 后置条件 | 返回促销策略ID |
| promotion. addPromotion | 语法 | public ResultMessage addPromotion(PromotionVO promotionVO) |
| 前置条件 | 无 |
| 后置条件 | 在数据库中增加促销策略持久化对象 |
| promotion. delPromotion | 语法 | public ResultMessage delPromotion(String setterID,String promotionID) |
| 前置条件 | 无 |
| 后置条件 | 在数据库中删除促销策略持久化对象 |
| promotion. getPromotionOfHotel | 语法 | public ArrayList<PromotionVO> getPromotionOfHotel(String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 返回酒店所有促销策略 |
| Promotion.totalDiscount | 语法 | public ArrayList <PromotionVO> getPromotionOfDistrict (String district) |
| 前置条件 | 无 |
| 后置条件 | 返回网站营销人员制定过的地区策略 |
| promotion. countPromotionOfRoom | 语法 | public String countPromotionOfRoom(OrderInfo orderVO) |
| 前置条件 | 订单生成 |
| 后置条件 | 返回最优策略下价格 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| promotondao.getNewID(String setterID) | 得到新的促销策略的编号 | |
| promotiondao.insert(PromotionPO po) | 添加策略 | |
| promotiondao.delete(String setterID, String promotionID) | 删除制定者、策略编号 | |
| promotiondao.update(PromotionPO po) | 更新策略 | |
| promotiondao.find(String setter, String promotionID) | 查看制定者、策略编号 | |
| promotiondao.finds(String scope) | 根据地区或酒店id返回符合条件的促销策略 | |
| user.getInfo(String userID) | 返回用户信息 | |

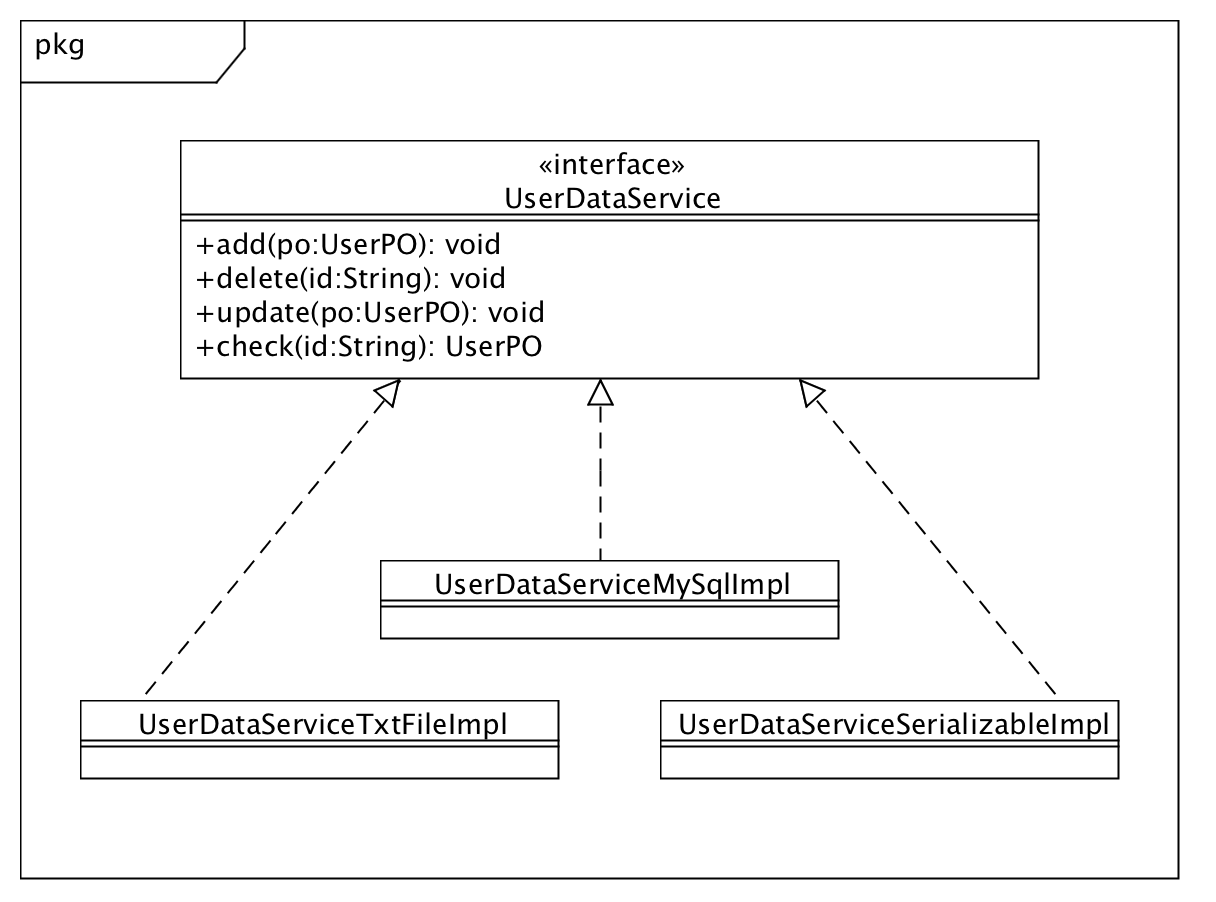
#### 5.3.2.6 webStaff模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| webStaff. getIDForWebsalesman | 语法 | public String getIDForWebsalesman() |
| 前置条件 | 无 |
| 后置条件 | 得到营销人员id |
| webStaff.addWebSalesman | 语法 | public ResultMessage addWebSalesman(String id, String password) |
| 前置条件 | 无 |
| 后置条件 | 添加网站营销人员 |
| webStaff. getAllWebSalesmen | 语法 | public ArrayList<WebSalesmanVO> getAllWebSalesmen() |
| 前置条件 | 无 |
| 后置条件 | 返回所有营销人员信息 |
| webStaff. changePassword | 语法 | public ResultMessage changePassword(String ID,String oldPassword, String newPassword) |
| 前置条件 | 无 |
| 后置条件 | 系统显示更新结果 |
| webStaff. updateWebSalesman | 语法 | public ResultMessage updateWebSalesman(WebSalesmanVO webSalesmanVO) |
| 前置条件 | 无 |
| 后置条件 | 更新网站营销人员信息 |
| webStaff. getWebSalesmanVO | 语法 | public WebSalesmanVO getWebSalesmanVO(String webSalesmanID) |
| 前置条件 | 无 |
| 后置条件 | 返回网站营销人员信息 |
| webStaff. checkPassword | 语法 | public static ResultMessage checkPassword(String ID, String password) |
| 前置条件 | id和密码格式正确 |
| 后置条件 | 检查用户名和密码是否匹配，返回验证结果 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| webManagerdao.updateManager (String managerID,String password) | 更新网站管理人员密码信息 | |
| webManagerdao. getManagerInstance(String managerID) | 根据网站管理人员ID 得到网站管理人员信息 | |
| websalesmandao.insert(WebSalesmanPO webSalesmanPO) | 增加网站营销人员 | |
| websalesmandao. update(WebSalesmanPO webSalesmanPO) | 根据websalesmanpo的ID 更新网站营销人员 | |
| websalesmandao.findByID(String webSalesmanID) | 根据ID 得到网站营销人员 | |
| websalesmandao.findByDistrict(String district) | 根据地址 得到辖区内的网站营销人员 | |
| websalesmandao.getAll() | 得到所有网站营销人员 | |
| websalesmandao.getIDForNewWebSalesman(String year) | 根据四位入职年份得到新的ID | |

## 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。由于持久化数据的保存可能存在多种形式：Txt文件、序列化文件、数据库等，所示抽象了数据服务。以UserDataService为例,

数据层模块的具体描述如下图所示：

数据层模块的描述

### 5.4.1 数据层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| logindao | 负责保存在线人员记录，进行登陆登出的操作 |
| userdao | 负责保存用户基本信息，进行增删改查的操作 |
| hoteldao | 负责保存酒店基本信息，并进行增删改查的操作 |
| orderdao | 负责保存订单信息，并进行增删改查的操作 |
| promotiondao | 负责保存促销策略信息，并进行增删改查的操作 |
| webStaffdao | 负责保存网站工作人员信息，并进行增删改查的操作 |

### 5.4.2 数据层模块的接口规范

#### 5.4.2.1 LoginDao

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| logindao.addOnline | 语法 | public ResultMessage addOnline(OnlinePersonPO po) |
| 前置条件 | 用户登陆 |
| 后置条件 | 在数据库中增加一个po记录 |
| logindao.deleteOnline | 语法 | public ResultMessage deleteOnline(Role role,String id) |
| 前置条件 | 数据库中含有该po |
| 后置条件 | 删除一个po |

#### 5.4.2.2 UserDao

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| userdao.getInfo | 语法 | public UserPO getInfo(String userID) |
| 前置条件 | id在数据库中存在 |
| 后置条件 | 根据用户编号 返回用户信息 |
| userdao.update | 语法 | public ResultMessage update(UserPO userPO) |
| 前置条件 | 数据库中含有该id的po记录 |
| 后置条件 | 更新一个po |
| userdao.insert | 语法 | public ResultMessage insert(UserPO userPO) |
| 前置条件 | 无 |
| 后置条件 | 在数据库中新增一个用户持久化对象 |
| userdao.setMemberLevel | 语法 | public ResultMessage setMemberLevel(int gradeWithCredit) |
| 前置条件 | 无 |
| 后置条件 | 制定会员等级 |
| userdao.getMemberLevel | 语法 | public int getMemberLevel() |
| 前置条件 | 无 |
| 后置条件 | 返回会员等级 |
| userdao. getAll() | 语法 | public ArrayList<UserPO> getAll() |
| 前置条件 | 无 |
| 后置条件 | 返回用户持久化信息列表 |
| userdao.getCreditRecordList | 语法 | public ArrayList<CreditRecordPO> getCreditRecordList(String userid) |
| 前置条件 | 无 |
| 后置条件 | 返回用户信用记录列表 |
| userdao.addCreditRecord | 语法 | public ResultMessage addCreditRecord(CreditRecordPO po) |
| 前置条件 | 用户信用值变化 |
| 后置条件 | 增加用户信用记录持久化对象 |

#### 5.4.2.3 HotelDao

|  |  |  |
| --- | --- | --- |
| 提供的接口（供接口） | | |
| hoteldao.getHotel | 语法 | public HotelPO getHotel(String id) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回酒店单一持久化对象 |
| hoteldao.getHotelList | 语法 | Public ArrayList<HotelPO> getHotelList(String district) |
| 前置条件 | 该酒店列表存在 |
| 后置条件 | 返回酒店持久化对象列表 |
| hoteldao. updateHotel | 语法 | Public ResultMessage updateHotel(HotelPO po) |
| 前置条件 | 无 |
| 后置条件 | 更新酒店单一持久化对象 |
| hoteldao.addHotel | 语法 | Public ResultMessage addHotel(HotelPO po) |
| 前置条件 | 无 |
| 后置条件 | 插入酒店单一持久化对象 |
| hoteldao.getAvaliableRoomList | 语法 | public ArrayList<RoomAvailPO> getAvailableRoomList(String id，Date checkIn) |
| 前置条件 | 该客房列表存在 |
| 后置条件 | 根据酒店ID、日期 返回酒店在该日期下的所有客房的可用信息 |
| hoteldao. changeRoomAvail | 语法 | public ResultMessage changeRoomAvail(String id, String roomType,boolean isPlus, int num, Date checkIn, Date checkOut) |
| 前置条件 | 酒店工作人员手动修改 |
| 后置条件 | 更改酒店 房间类型 房间可用数量的信息 |
| hoteldao.getNewHotelID | 语法 | public String getNewHotelID(String district) |
| 前置条件 | 添加酒店 |
| 后置条件 | 根据酒店传入地区 初始化酒店账号 |
| hoteldao.updateGrade | 语法 | public ResultMessage updateGrade(String hotelID,int grade) |
| 前置条件 | 用户输入评分 |
| 后置条件 | 更新酒店评分 |
| hoteldao. addSpecialRoom | 语法 | public ResultMessage addSpecialRoom(RoomPO roomPO) |
| 前置条件 | 无 |
| 后置条件 | 根据roompo里的酒店账号 添加酒店的客房信息 并将该房间类型可用客房数量置为总数 |
| hoteldao.deleteSpecialRoom | 语法 | public ResultMessage deleteSpecialRoom(RoomPO roomPO) |
| 前置条件 | 该房间存在 |
| 后置条件 | 根据roompo里的酒店账号 删除酒店的客房信息 |
| hoteldao.getRoomList | 语法 | public ArrayList<RoomPO> getRoomList(String hotelID) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 根据酒店ID 返回酒店的所有房间的信息 |
| hoteldao.updateRoom | 语法 | public ResultMessage updateRoom(RoomPO roomPO) |
| 前置条件 | 无 |
| 后置条件 | 根据roompo里的酒店账号 更新该房间类型的信息 |
| hoteldao. numOfRoomAvail | 语法 | public int numOfRoomAvail(String id, String roomType, Date checkIn, Date checkOut) |
| 前置条件 | 该酒店存在 |
| 后置条件 | 返回酒店 房间类型 一段日期 可用数量 |
| hoteldao.getAll | 语法 | public ArrayList<HotelPO> getAll() |
| 前置条件 | 无 |
| 后置条件 | 返回所有酒店的持久化对象列表 |
| hoteldao. getImageAddresses | 语法 | public ArrayList<String> getImageAddresses(String hotelID) |
| 前置条件 | 该id格式正确 |
| 后置条件 | 从数据库中查找该酒店id，返回该酒店的所有图片的url |

#### 5.4.2.4 OrderDao

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| orderdao.searchByID | 语法 | public OrderPO searchByID(String orderid) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按订单ID进行查找返回相应的OrderPO结果 |
| orderdao.searchByUser | 语法 | public OrderPO searchByUser(String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按用户ID进行查找返回相应的OrderPO结果 |
| orderdao.searchByUserWithHotel | 语法 | public OrderPO searchByUserWithHotel(String userid,String hotelid) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按用户编号、酒店编号进行查找返回相应的OrderPO结果 |
| orderdao.searchByHotel | 语法 | public OrderPO searchByHotel(String hotelid) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按酒店编号进行查找返回相应的OrderPO结果 |
| orderdao.searchState | 语法 | public OrderPO searchByState(StateOfOrder state) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按订单状态进行查找返回相应的OrderPO结果 |
| orderdao.insert | 语法 | public void insert(OrderPO po) throws RemoteException |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| orderdao.update | 语法 | public void update(OderPO po) throws RemoteExcptioin |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |

#### 5.4.2.5 PromotionDao

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| promotiondao.find | 语法 | public PromotionPO find(String setter, String promotionID)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按制定者、策略编号进行查找返回相应的PromotionPO结果 |
| promotiondao.insert | 语法 | public void insert(PromotionPO po) throws RemoteException |
| 前置条件 | 相同的ID在数据库中不存在 |
| 后置条件 | 在数据库中增加一个po数据 |
| promotiondao.update | 语法 | public void update(PromotionPO po) throws RemoteException |
| 前置条件 | 在数据库中存在相同ID的po |
| 后置条件 | 更新一个po数据 |
| PromotionDataService.delete | 语法 | public void delete(String setterID, String promotionID) throws RemoteException |
| 前置条件 | 在数据库中存在相同ID的po |
| 后置条件 | 删除一个po数据 |
| promotiondao.finds | 语法 | public ArrayList<PromotionPO> finds(String scope)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按地区编号进行查找返回相应的PromotionPO结果 |
| promotiondao. getNewID | 语法 | public String getNewID(String setterID)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回新的促销策略的编号 |

#### 5.4.2.6 WebStaffDao

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| webstaffdao.updateManager | 语法 | public ResultMessage updateManager (String managerID,String password)throws RemoteException |
| 前置条件 | 网管修改密码 |
| 后置条件 | 更新网站管理人员密码信息 |
| webstaffdao. getManagerInstance | 语法 | public WebManagerPO getManagerInstance(String managerID)throws RemoteException |
| 前置条件 | ID在数据库中存在 |
| 后置条件 | 根据网站管理人员ID 得到网站管理人员信息 |
| webstaffdao.insert | 语法 | public ResultMessage insert(WebSalesmanPO webSalesmanPO)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 增加一个po数据 |
| webstaffdao.update | 语法 | public ResultMessage update(WebSalesmanPO webSalesmanPO)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 更新一个po数据 |
| webstaffdao.findByID | 语法 | public WebSalesmanPO findByID(String webSalesmanID)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 根据ID返回网站营销人员 |
| webstaffdao.findByDistrict | 语法 | public ArrayList<WebSalesmanPO> findByDistrict(String district)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 根据地址 返回辖区内的网站营销人员 |
| webstaffdao.getAll | 语法 | public ArrayList<WebSalesmanPO> getAll()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有网站营销人员 |
| webstaffdao.getNewID | 语法 | public String getNewID() throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 根据四位入职年份得到新的ID |

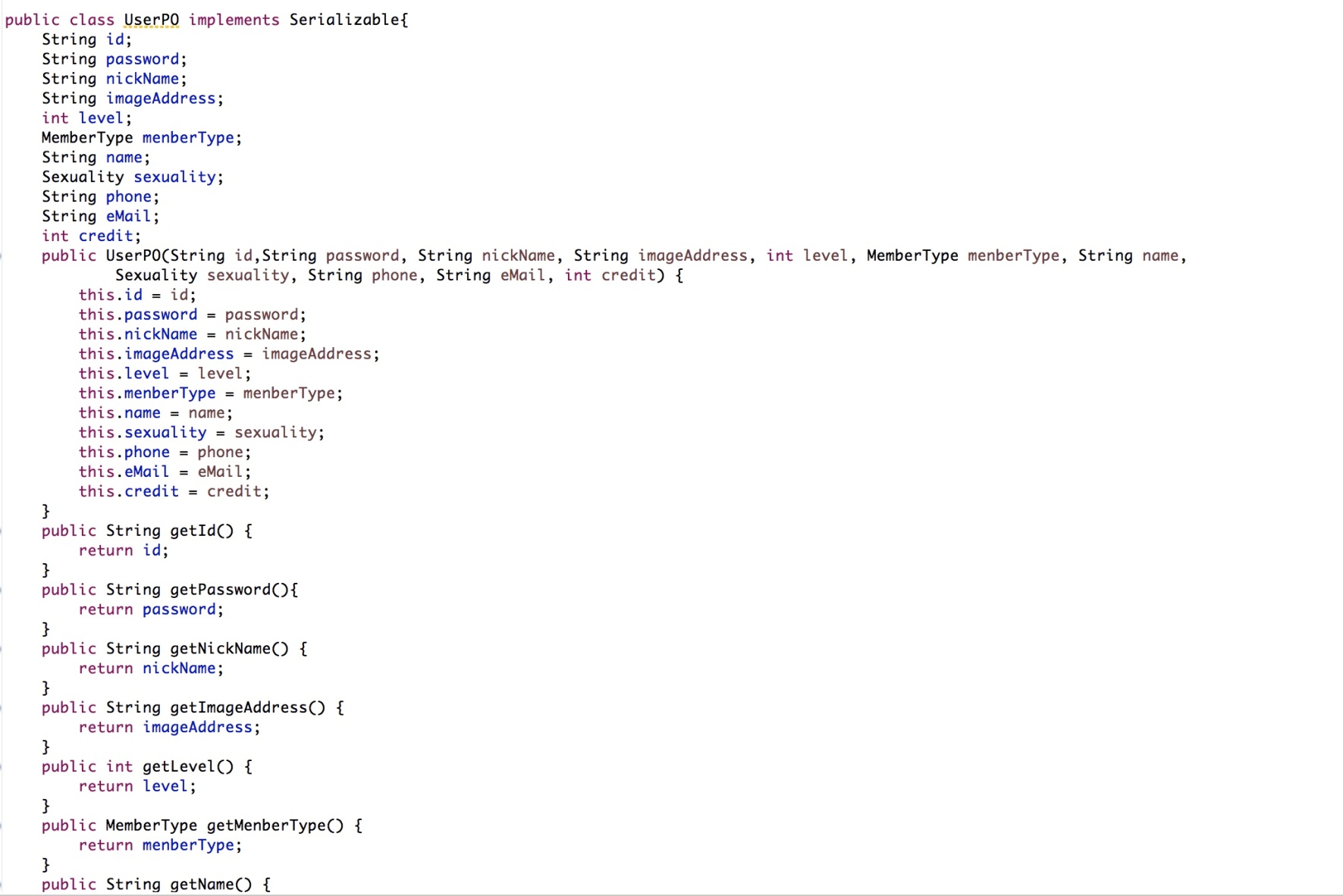
# 6.信息视角

## 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，包括ID、密码、昵称以及个人其他信息。
* HotelPO类包含酒店工作人员的账号，密码，酒店的名称、联系方式、地址、商圈、简介、设施、星级、评分、最晚入住时间属性
* RoomAvailPO类包含酒店账号、客房类型、可用数量、日期、价格、图片地址
* PromotionPO类包含制定策略的酒店编号（000000000表示网站）、酒店内策略编号、策略名称、策略满足条件、策略折扣
* WebSalesmanPO类包含网站营销人员账号、密码、姓名、联系电话
* WebManagerPO类包含网站管理人员账号、密码、姓名、联系电话
* OrderPO 类包含订单编号、酒店编号、用户编号、订单总额、总额促销策略、房间的类型、价格、入住日期、离开日期、房间的促销策略

持久化用户对象UserPO的定义如图所示：



## 6.2 Txt持久化格式

文件格式：本系统依据不同情况分别采用数据库、序列化、txt文件持久化保存三种方式保存文件。

Txt数据保持格式以MemberLevelSetting.txt为例。每行存储的是升一级的额度

## 6.3 数据库表

数据库表及其格式如下：

// 账号 密码 昵称 头像url

// 生日 会员等级 会员类型 企业名称 信用值

// 真实姓名 性别 邮箱 信用记录条数

UserInfo(userID blob,password blob, nickName char(10),image tinytext,birthday char(10),level tinyint, memberType tinyint,commerceName char(20),credit int, trueName blob,sex tinyint,eMail char(30),creditRecordNum int

// 编号 密码 电话 名称

// 地址 商圈 标准双人间价格 简介 设施

// 星级 评分 最晚入住时间 图片地址 评论人数 房间类型数量

HotelInfo:

hotelID char(10),password blob,phoneNumber char(11),name char(15), district char(6),addressDetail char(20),standardPrice double,briefIntro tinytext,facility char(4),level tinyint,grade double,latestCheckinTime char(8),imageAddress tinytext,commentNum int,roomTypeNum tinyint

// 酒店 类型 总量

// 价格 图片地址 可用数量日期列表 上次更新日期

RoomInfo:

hotelID char(10),roomType char(10),amountTotal int,price double,imageAddress tinytext,aList text,lastUpdate date

// 订单编号 用户编号

// 用户姓名 酒店编号 酒店名称 订单状态 房间类型 房间价格 房间数量

// 入住人数 是否有儿童 原价 折后 促销策略

// 评价 评分 预计入住、退房时间 酒店最晚时间

// 实际入住、退房时间 订单生成时间 撤销订单时间 撤销异常时间

OrderGeneral:

orderID char(26),userID blob, userName blob, hotelID char(10), hotelName char(15), state tinyint,roomType char(10),roomPrice double,roomNum tinyint,peopleNum tinyint,withChild tinyint,originValue double,trueValue double,promotion char(20),

comment tinytext,grade tinyint,checkIn date,checkOut date,hotelDDL char(8),

bornDate date,actCheckIn datetime,actCheckOut datetime,cancelTime datetime,cancelAbTime datetime

// 制定者 序号 策略名称

// 生效日期 失效日期(2010-01-01默认0点) 针对类型（地区，酒店，房间类型） // 具体条件（酒店id10位(前6位地区id)，房间类型Str）

// 适用条件（房间数量、价值、会员等级（企业会员）、用户生日）

// 折扣方式（打折，降价） 数字均为正数 满200减20(200,20) 满2间打8折(2,0.8) 满10级减10(10,10) 用户生日：85折(0,0.85)

PromotionInfo:

setter char(10),id char(3),name char(10), beginDate date,endDate date,scopeType tinyint,scopeNum char(20), conditionType tinyint,conditionNum double,deductionType tinyint,deductionNum double

// 账号 日期 订单号 信用更改原因 信用变化 变更后信用值

CreditRecordInfo:

userID blob,changeDate date,orderID char(26), creditAction tinyint,changeValue int,credit int

// 账号 密码 姓名 地区

WebStaffInfo:

id char(10), password blob,name char(10),district char(6)