**酒店预订系统**

**体系结构设计文档**

学 院：南京大学软件学院

成 员：王友运卞纯源赵德宇王雪

完成日期：2016年10月10日

**目录**

[更新历史 4](#_Toc470033471)

[1、引言 5](#_Toc470033472)

[1.1 编制目的 5](#_Toc470033473)

[1.2词汇表 5](#_Toc470033474)

[1.3 参考资料 5](#_Toc470033475)

[2、产品概述 6](#_Toc470033476)

[3、逻辑视角 8](#_Toc470033477)

[4、组合视角 9](#_Toc470033478)

[4.1开发包图 11](#_Toc470033479)

[4.2运行时进程 15](#_Toc470033480)

[4.3物理部署 16](#_Toc470033481)

[5、接口视角 16](#_Toc470033482)

[5.1模块的职责 16](#_Toc470033483)

[5.2用户界面层的分解 19](#_Toc470033484)

[5.2.1用户界面层模块的职责 21](#_Toc470033485)

[5.2.2用户界面层模块的接口规范 21](#_Toc470033486)

[5.2.3 用户界面模块设计原理 23](#_Toc470033487)

[5.3业务逻辑层的分解 23](#_Toc470033488)

[5.3.1业务逻辑层模块的职责 24](#_Toc470033489)

[5.3.2业务逻辑层模块的接口规范 25](#_Toc470033490)

[5.4 数据层的分解 48](#_Toc470033491)

[5.4.1 数据层模块的职责 49](#_Toc470033492)

[5.4.2 数据层模块的接口规范 50](#_Toc470033493)

[6、信息视角 62](#_Toc470033494)

[6.1 数据持久化对象 62](#_Toc470033495)

[6.2 数据库表 65](#_Toc470033496)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 全体人员 | 2016-10-11 | 最初草稿（创建文档模板） | V1.0 |
| 王雪 | 2016-12-19 | 根据需求变化修改文档，完成界面和逻辑层部分 | V1.1 |
| 卞纯源 | 2016-12-20 | 根据需求变化修改文档，完成数据层部分 | V1.2 |

# 1、引言

## 1.1 编制目的

本报表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，同时实现和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而编写，是了解系统的导航。

## 1.2词汇表

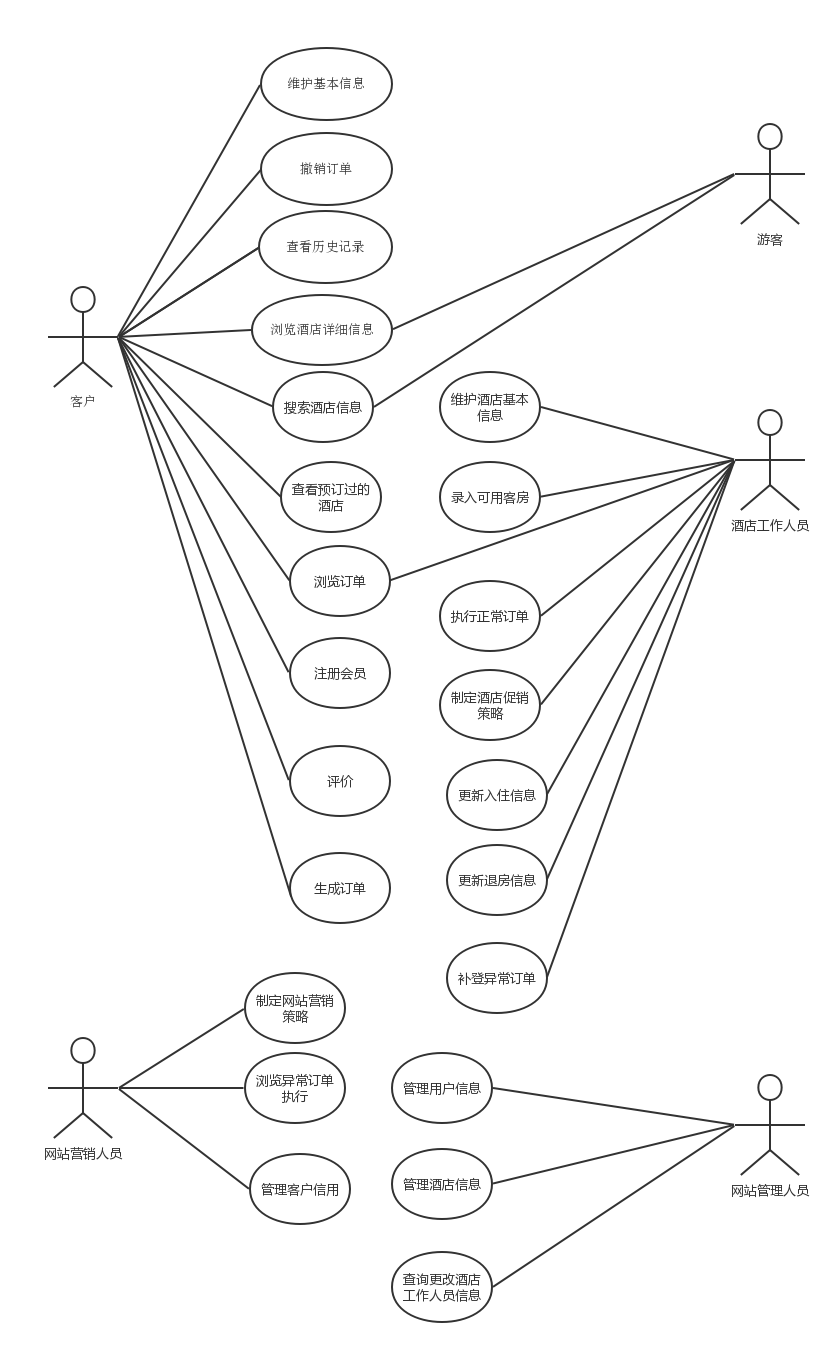
|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| \_ui | 表示某展示层 |  |
| \_bl | 表示某逻辑层 |  |
| \_data | 表示某数据层 |  |
| RMI | 表示远程方法调用 |  |

## 1.3 参考资料

1. IEEE std 1471-2000
2. 丁二玉，刘钦.计算与软件工程（卷二）[M]机械工业出版2012：134—182

# 2、产品概述

参考酒店预订系统用例文档和酒店预订系统软件需求规格说明文档中队产品的概括描述。酒店预订系统主要是应用于网上预定远程酒店订单的在线系统，主要功能见用例图如下。



# 3、逻辑视角

酒店预订系统中，选择了分层体系结构风格，将系统分为3层(展示层、

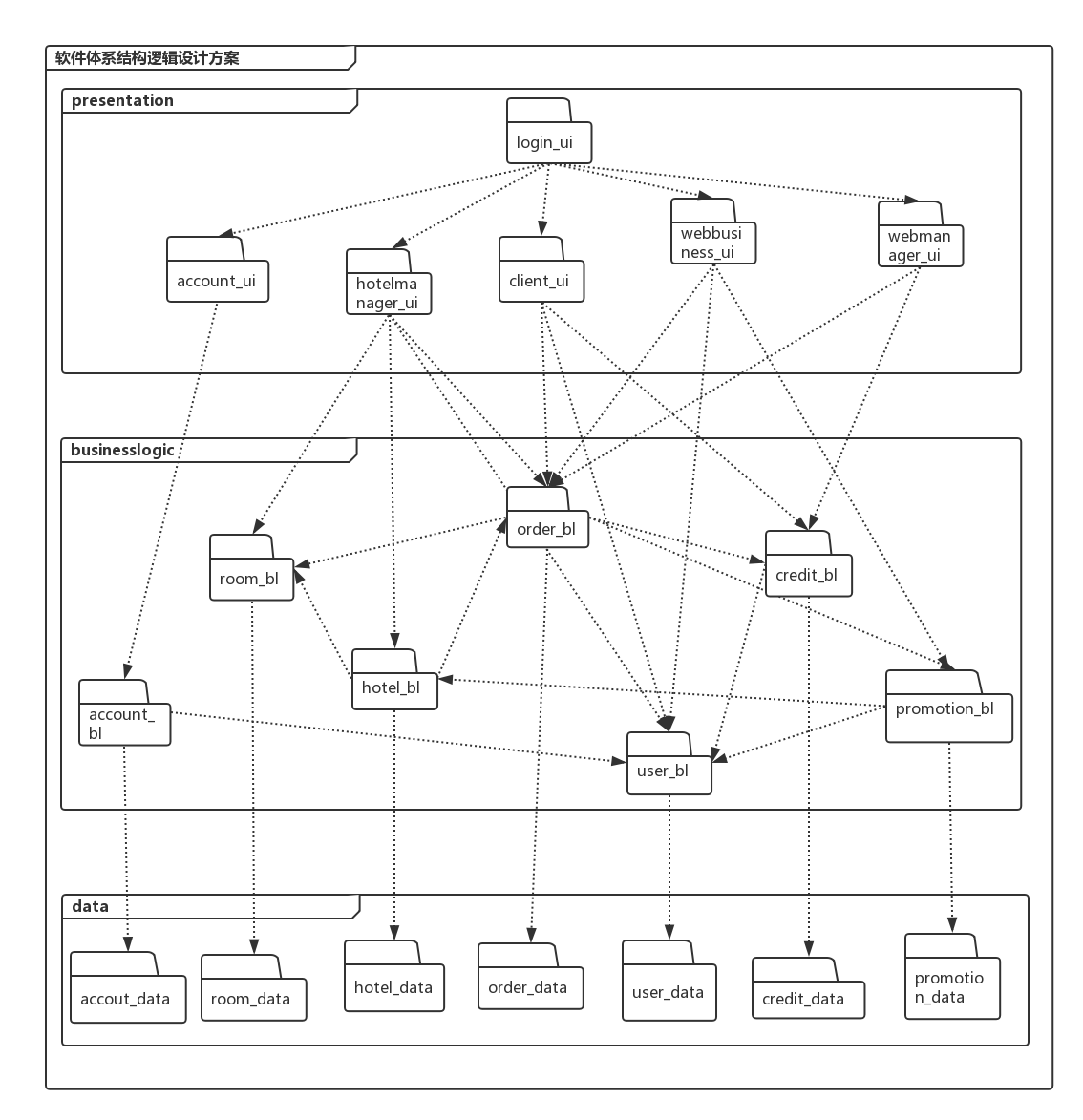
业务逻辑层、数据层)能够很好地示意整个高层抽象。展示层包含GUI页面

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。



图一 参照体系结构风格的包图表达逻辑视角



图二 软件体系结构逻辑设计方案

# 4、组合视角

[软工2 166]与抽象的逻辑设计相比，实现物理设计要考虑更多的实现细节，这些细节有：

1. presentation层与logic层被置于客户端，data层被置于服务器端，那么logic层的开发包不可能依赖于data层的开发包。使用RMI技术，将data层开发包分解为置于客户端的dataservice接口包和置于服务器端的data层开发包。这样一来，logic层开发包依赖于dataservice包，dataservice包和data层的开发包都依赖于RMI类库包。
2. 所有的data层开发包都需要进行数据持久化（例如读写数据库、读写文件等），所以它们会有一些重复代码，可以将重复代码独立为新的开发包，然后所有的data层开发包都依赖于databasetility。datavaseutility会依赖于JDBC类库包。
3. 所有的presentation层开发包都需要使用图形类型建立界面，都要依赖于图形界面类库包。
4. 此外，presentation层实现时，由mainui包负责整个页面之间的跳转逻辑。其他各包负责各自页面自身的功能。
5. 在分层风格的典型设计中，不希望高层直接依赖于低层，而是为低层建立接口包，实现依赖倒置原则，所以应该调整为：各presentation层开发包（调用）依赖于logic层接口包businesslogicservice包，logic层开发包也依赖于（实现了）logic层接口包businesslogicservice包。
6. 在分层风格的典型设计中，presentation层与logic层之间、logic层与data层之间可能会传递复杂数据对象，那么相邻两层都需要使用数据对象声明，所以需要将数据对象声明独立为开发包（VO包和PO包）。
7. 使用依赖倒置原则消除包的循环依赖现象，将循环依赖变为单向依赖：

8）在logic层中，初始化和业务逻辑层上下文的工作被分配到utility包中。

经过细节改进，最终建立的酒店预订系统开发包设计如表4.1-1，其局部包图如图4.1-1和4.1-2所示：

## 4.1开发包图

表4.1-1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 |
| login\_ui | 界面类库包,VO |
| Account\_ui | Account\_blservice界面类库包，VO |
| client\_ui | Order\_blservice,credit\_blservice,user\_BLService,界面类库包,VO |
| hotelmanager\_ui | Hotel\_Blservice,room\_blservice,order\_blservice,user\_blservice,界面类库包,VO |
| webmanager\_ui | User\_blservice,界面类库包,VO |
| webbussiness\_ui | Credit\_blservice,user\_blservice,promotion\_blservice,order\_blservice,界面类库包,VO |
| vistor\_ui | 界面类库包,VO |
| credit\_blservice |  |
| credit\_bl | credit\_data\_service,PO |
| user\_blservice |  |
| user\_bl | order\_bl,hotel\_bl  user\_data\_service, PO |
| account\_blservice |  |
| account\_bl | account\_data\_service,PO |
| room\_blservice |  |
| room\_bl | room\_data\_service, PO |
| hotel\_blservice |  |
| hotel\_bl | room\_bl, promotion\_bl,hotel\_data\_service,PO |
| order\_blservice |  |
| order\_bl | credit\_bl, hotel\_bl,  promotion\_bl,  order\_data\_service, PO |
| promotion\_blservice |  |
| promotion\_bl | Promotion\_data\_service |
| user\_data | databaseutility,PO |
| user\_data\_serivce | JAVA RMI,PO |
| accout\_data | databaseutility,PO |
| accout\_data\_service | JAVA RMI,PO |
| hotel\_data | databaseutility,PO |
| hotel\_data\_service | JAVA RMI,PO |
| room\_data | databaseutility,PO |
| room\_data\_service | JAVA RMI,PO |
| order\_data | databaseutility,PO |
| order\_data\_  service | JAVA RMI,PO |
| promotion\_data | databaseutility,PO |
| promotion\_data\_  service | JAVA RMI,PO |
| credit\_data | databaseutility,PO |
| credit\_data\_service | JAVA RMI,PO |
| VO |  |
| PO |  |
| databaseutility | java.JDBC |
| 界面类库包 |  |
| JAVA RMI | javafx |

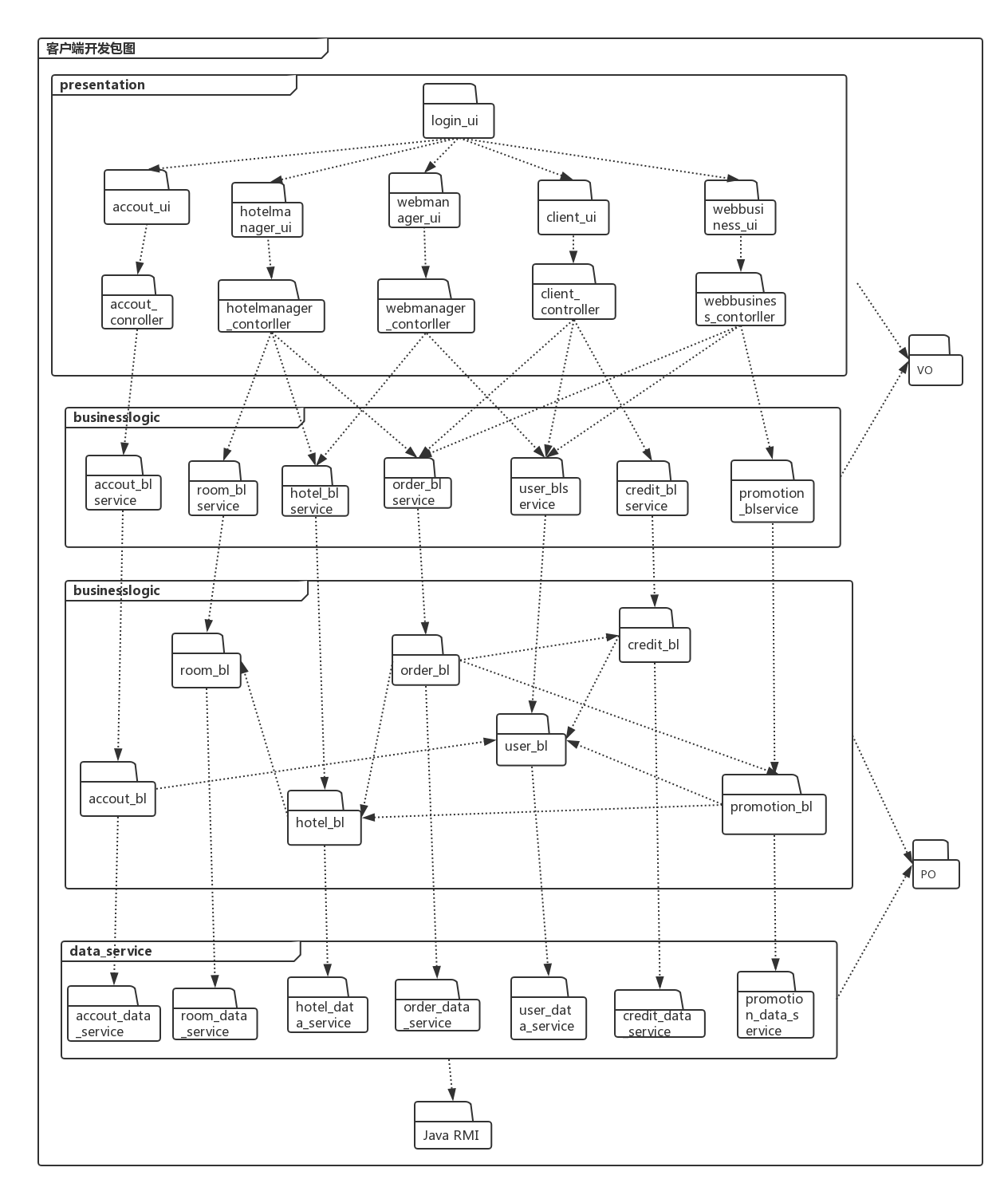


图4.1-1酒店预订系统客户端开发包图



图4.1-2酒店预订系统服务器端开发包图

## 4.2运行时进程

在酒店预订系统中，会有多个客户端进程和一个服务器端进程，其进程图

如图4.2所示。结合部署图，客户端进程实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4.2进程图

## 4.3物理部署

酒店预订系统中客户端构件是放在客户端机器上，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JavaRMI构件属于JDK6.0的一部分。所以，在系统JDK环境已经设置好的情况下，不需要再独立部署。部署图如图4.3所示。



4.3 部署图

# 5、接口视角

## 5.1模块的职责

客户端模块和服务器端模块视图分别如图5.1-1和图5.1-2所示。客户端各层和服务器端各层的职责分别如表5.1-1和表5.1-2所示。

图5.1-1 客户端模块视图



图5.1-2 服务器端模块视图

表5.1-1 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响应和业务处理逻辑。 |
| 客户端网络模块 | 利用Java RMI机制查找RMI服务,检测网络连接状态，进行断线重连 |

表5.1-2 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5.1-3所示。

表5.1-3 层之间调用的接口

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| credit\_blservice  user\_blservice  accout\_blservice  room\_blservice  hotel\_blservice  order\_blservice  promotion\_blservice | 客户端展示层 | 客户端业务逻辑层 |
| credit\_data\_service  user\_data\_service  acccout\_data\_service  hotel\_data\_service  room\_data\_service  order\_data\_service  evaluation\_data\_service  promotion\_data\_service | 客户端业务逻辑层 | 服务器端数据层 |

借用用户管理用例来说明层之间的调用，如图5.1-3所示。每一层之间都是由上层依赖了一个接口（需接口），而下层实现这个接口（供接口）。UserBLService提供了User界面所需要的所有业务逻辑功能

UserDataService提供了对数据库的增、删、改、查等操作。

这样的实现就大大降低了层与层之间的耦合。

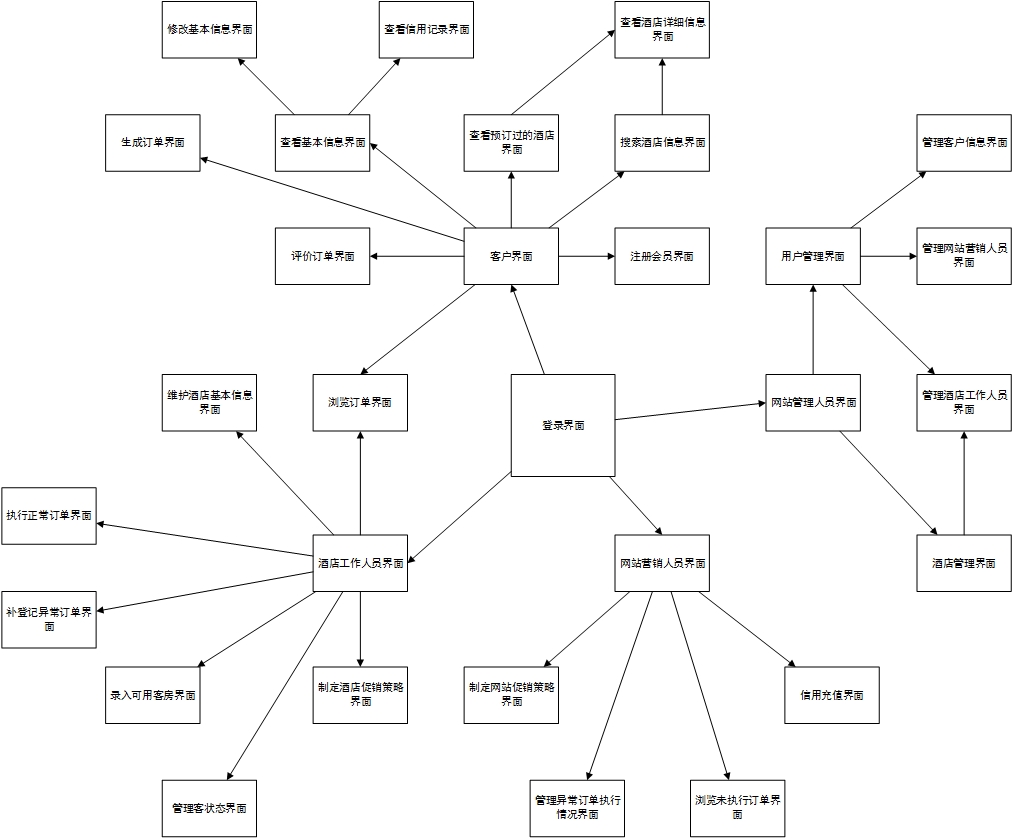


图5.1-3 用户管理用例层之间调用的接口

## 5.2用户界面层的分解

根据需求，系统存在30个用户界面：客户界面，酒店工作人员界面，网站管理人员界面，网站营销人员界面，注册会员界面，搜索酒店信息界面，查看酒店详细信息界面，查看预订过的酒店信息界面，浏览订单界面，生成订单界面，评价订单界面，查看信用记录界面，查看基本信息界面，修改基本信息界面，查看信用记录界面，执行正常订单界面，补登记异常订单界面，维护酒店基本信息界面，录入可用客房信息界面，管理客房状态界面，制定酒店促销策略界面，制定网站促销策略界面，管理异常订单执行情况界面，查看未执行订单界面，信用充值界面，酒店管理界面，用户管理界面，管理客户信息界面，管理酒店工作人员信息界面，管理网站营销人员界面。

界面跳转如图5.2所示



服务器端和客户端的用户界面设计接口是一致的，只是具体的页面不一样。用户界面类如图5.2-2所示。



图5.2-2 用户界面类

### 5.2.1用户界面层模块的职责

如表5.2.1-1所示为用户界面层模块的职责。

表5.2.1-1 用户界面层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| MainStage | 界面Stage,负责界面的显示和界面的跳转。 |

### 5.2.2用户界面层模块的接口规范

用户界面层模块的接口规范如表5.2.2-1所示。

表5.2.2-1 用户界面层模块的接口规范

|  |  |  |
| --- | --- | --- |
| Login\_ui | 语法 | Start(Stage stage) |
| 前置条件 | 无 |
| 后置条件 | 显示Stage以及accountScene |
| account\_ui | 语法 | Start(Stage stage) |
| 前置条件 | 无 |
| 后置条件 | 显示accountScene |
| Webmanager\_ui | 语法 | Start(Stage stage) |
| 前置条件 | 网站管理人员登录 |
| 后置条件 | 显示WebManagerScene |
| Vistor\_ui | 语法 | Start(Stage stage) |
| 前置条件 | 无 |
| 后置条件 | 显示vistorScene |
| Hotelmanager\_ui | 语法 | Start(Stage stage) |
| 前置条件 | 酒店工作人员登录 |
| 后置条件 | 显示HotelmanagerScene |
| Webbusiness\_ui | 语法 | Start(Stage stage) |
| 前置条件 | 网站营销人员登录 |
| 后置条件 | 显示WebbusinessScene |
| client\_ui | 语法 | Start(Stage stage) |
| 前置条件 | 客户登录 |
| 后置条件 | 显示clientScene |

用户界面层需要的服务接口如表5.2.2-2所示。

表5.2.2-2 用户界面层模块需要的服务接口

|  |  |
| --- | --- |
| 服务名 | 服务 |
| businesslogicservice.credit\_blservice | 提供信用增加和信用扣除 |
| businesslogicservice.user \_blservice | 处理用户信息的接口 |
| businesslogicservice.account\_blservice | 处理登录注册退出登录的接口 |
| businesslogicservice.room\_blservice | 处理酒店客房信息的接口 |
| businesslogicservice.hotel\_blservice | 处理酒店信息的接口（包括搜索酒店和查看酒店） |
| businesslogicservice.order\_blservice | 处理订单信息（浏览、查看订单）的接口 |
| businesslogicservice. promotion\_blservice | 处理网站和酒店营销策略的接口 |

### 5.2.3 用户界面模块设计原理

用户界面利用JavaFX与CSS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的业务逻辑处理对象。例如，Account对象负责处理登陆界面的业务逻辑；User对象负责处理管理用户的业务逻辑。业务逻辑层的设计如图5.3-1所示。

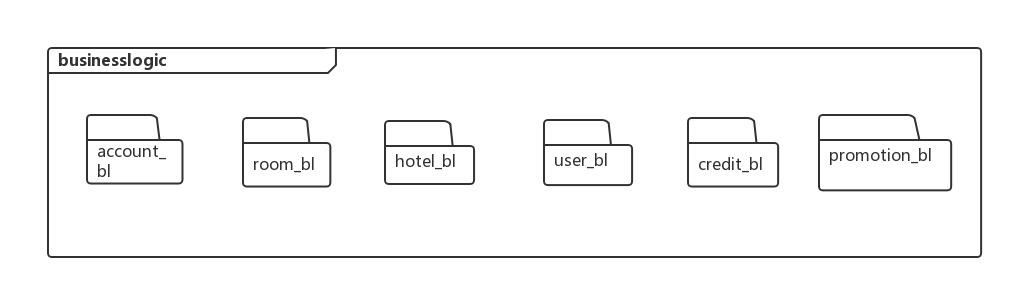


图5.3-1 业务逻辑层

### 5.3.1业务逻辑层模块的职责

业务逻辑层模块的职责如表5.3.1-1所示。

表5.3.1-1 业务逻辑层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| credit\_bl | 提供信用增加和信用扣除 |
| user \_bl | 处理用户信息的接口 |
| account\_bl | 处理登录注册退出登录的接口 |
| room\_bl | 处理酒店客房信息的接口 |
| hotel\_bl | 处理酒店信息的接口（包括搜索酒店和查看酒店） |
| order\_bl | 处理订单信息（浏览、查看订单）的接口 |
| promotion\_bl | 处理网站和酒店营销策略的接口 |

### 5.3.2业务逻辑层模块的接口规范

表5.3.2-1 credit\_bl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Credit.getCredit | 语法 | public int getCredit (String user\_id); | |
| 前置条件 | 已知客户id，需要获得客户的信用值 | |
| 后置条件 | 返回信用值 | |
| Credit.setVipCredit | 语法 | public ResultMessage setVIPCredit(, int credit\_num); | |
| 前置条件 | 网站营销人员已经登录 | |
| 后置条件 | 返回成功与否 | |
| Credit.getVIPCredit | 语法 | public int getVIPCredit ( ) | |
| 前置条件 | 需要获得升级所需的信用值 | |
| 后置条件 | 返回信用值 | |
| CreditHistory.changeCredit | 语法 | Public ResultMessage changeCredit(CreditChangeVO vo) | |
| 前置条件 | 用户执行订单或者撤销订单或者充值，或者网站营销人员撤销异常订单 | |
| 后置条件 | 系统更新用户的信用记录和信用值 | |
| CreditChange.getCreditHistory | 语法 | public ArrayList<CreditChangeVO> getCreditHistory (String userID); | |
| 前置条件 | 用户登陆，需要查看信用记录 | |
| 后置条件 | 数据从数据层调用至界面 | |
| CreditChange.rechargeCredit | 语法 | Public ResultMessage rechargeCredit(CreditChangeVO vo) | |
| 前置条件 | 用户向网站营销人员进行线下充值，网站营销人员调用该方法给用户充值 | |
| 后置条件 | 调用同类中的changeCredit方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| User.UpdateClientVip.updateClientVip(String userID, int level) | | | （当信用值的改变导致用户vip等级的改变时）改变用户的vip等级 |
| Data层 | | |  |
| CreditDao. getCredit（String userID） | | | 获得某一用户的信用值 |
| CreditDao.setVIPCredit(int level, int credit\_num) | | | 设置vip升级所需要的信用值 |
| CreditDao.getVIPCredit(int level) | | | 获得升级指定VIP所需要的信用值 |
| CreditDao.changeCredit(CreditChangePO po) | | | 向数据层添加信用记录 |
| CreditDao.getCreditHistory(String userID) | | | 根据ID获取用户的所有信用记录 |
|  | | |  |

表5.3.2-2 user\_bl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Client.getClientInfo | 语法 | Public ClientVO getClientInfo(String userID); | |
| 前置条件 | 已获得客户ID，需要查看基本信息 | |
| 后置条件 | 返回对应的用户基本信息 | |
| Clilent.updateClientInfo | 语法 | Public ResultMessage updateClientInfo(ClientVO vo); | |
| 前置条件 | 已知用户ID，需要修改对应的基本信息 | |
| 后置条件 | 更新用户数据并且返回是否修改成功 | |
| Client.registerVIP | 语法 | Public ResultMessage registerVIP (VIPVO vo); | |
| 前置条件 | 已获得用户ID和注册会员信息 | |
| 后置条件 | 更新用户vip信息并且返回注册会员是否成功 | |
| Client.getVIPInfo | 语法 | Public VIPVO getVIPInfo(String userID); | |
| 前置条件 | 客户已注册为会员，已获得客户ID，需要查看注册会员信息 | |
| 后置条件 | 查看数据并且返回用户注册会员信息 | |
| Client.updateClientVip | 语法 | Public boolean updateClientVip (String userID, int level); | |
| 前置条件 | 用户信用值改变并且达到了vip升级或者降级的标准 | |
| 后置条件 | 更新数据并且返回成功与否 | |
| Client.isVip | 语法 | Public boolean isVip(String userID) | |
| 前置条件 | 用户已经登录，需要查看自己是否是vip | |
| 后置条件 | 查看数据并且返回是与否 | |
| HotelManager.addHotelManager | 语法 | Public ResultMessage addHotelManager (HotelManagerVO hotelManager); | |
| 前置条件 | 已添加酒店，需要添加对应的酒店工作人员信息 | |
| 后置条件 | 更新数据并且返回是否添加成功 | |
| HotelManager.getHotelManagerInfo | 语法 | Public HotelManagerVO getHotelManagerInfo(String hotelID); | |
| 前置条件 | 需要获得酒店工作人员信息 | |
| 后置条件 | 查找数据返回酒店工作人员信息 | |
| HotelManager.updateHotelManagerInfo | 语法 | public ResultMessage updateHotelManagerInfo (HotelManagerVO hotelManagerInfo) | |
| 前置条件 | 已知酒店ID，需要更改的酒店工作人员信息 | |
| 后置条件 | 更新数据并返回是否修改成功 | |
| WebBusiness.addWebBusiness | 语法 | Public ResultMessage addWebBusiness(WebBusinessVO webBusinessInfo); | |
| 前置条件 | 需要添加网站营销人员 | |
| 后置条件 | 返回是否添加成功 | |
| WebBusiness.getWebBusinessInfo | 语法 | publicboolean getWebBusinessInfo(String webBusiness\_ID); | |
| 前置条件 | 需要获得网站营销人员信息 | |
| 后置条件 | 返回网站营销人员信息 | |
| WebBusiness.updateWebBusinessInfo | 语法 | Public ResultMessage updateWebBusinessInfo(WebBusinessVO ,webBusinessInfo); | |
| 前置条件 | 已知网站营销人员ID，需要修改网站营销人员信息 | |
| 后置条件 | 返回是否修改成功 | |
| WebManager.addHotelManager | 语法 | Public ResultMessage addHotelManager (HotelManagerVO hotelManager); | |
| 前置条件 | 已添加酒店，需要添加对应的酒店工作人员信息 | |
| 后置条件 | 调用HotelManger类的addHotelManager方法 | |
| WebManager.getHotelManagerInfo | 语法 | Public HotelManagerVO getHotelManagerInfo(String hotelID); | |
| 前置条件 | 需要获得酒店工作人员信息 | |
| 后置条件 | 调用HotelManger类的getHotelManagerInfo方法 | |
| WebManager.updateHotelManagerInfo | 语法 | public ResultMessage updateHotelManagerInfo (HotelManagerVO hotelManagerInfo) | |
| 前置条件 | 已知酒店ID，需要更改的酒店工作人员信息 | |
| 后置条件 | 调用HotelManger类的updateHotelManagerInfo方法 | |
| WebManager.addWebBusiness | 语法 | Public ResultMessage addWebBusiness(WebBusinessVO webBusinessInfo); | |
| 前置条件 | 需要添加网站营销人员 | |
| 后置条件 | 调用WebBusiness类的addWebBusiness方法 | |
| WebManager.getWebBusinessInfo | 语法 | publicboolean getWebBusinessInfo(String webBusiness\_ID); | |
| 前置条件 | 需要获得网站营销人员信息 | |
| 后置条件 | 调用WebBusiness类的addWebBusiness方法 | |
| WebManager.updateWebBusinessInfo | 语法 | Public ResultMessage getWebBusinessInfo (WebBusinessVO ,webBusinessInfo); | |
| 前置条件 | 已知网站营销人员ID，需要修改网站营销人员信息 | |
| 后置条件 | 调用WebBusiness类的updateWebBusinessInfo方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| ClientDao.updateClientInfo(ClientPO user) | | | 修改用户信息 |
| ClientDao.getClientInfo(Stirng userID) | | | 根据ID获取用户的信息 |
| ClientDao.addClient(ClientPO po) | | | 新增用户信息 |
| HotelManagerDao.getHotelManagerInfo(Stirng hotel\_ID) | | | 根据ID查找酒店管理人员的信息 |
| HotelManagerDao.addHotelManager (HotelManagerPO hotelManager) | | | 新增单一持久化对象：酒店管理人员 |
| HotelManagerDao.updateHotelManagerInfo(HotelManagerPO hotelManager) | | | 修改单一持久化对象：酒店管理人员的信息 |
| WebBusinessDao.getWebBusinessInfo(Stirng user\_ID) | | | 根据ID查找网站管理人员的信息 |
| WebBusinessDao.addWebBusiness (WebBusinessPO webBusiness) | | | 新增网站管理人员 |
| WebBusinessDao.updateWebBusinessInfo(WebBusinessPO webBusiness) | | | 修改网站管理人员的信息 |
|  | | |  |

表5.3.2-3 accout\_bl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Account.register | 语法 | publicboolean register(AccountVO account); | | |
| 前置条件 | 用户没有账号 | | |
| 后置条件 | 返回注册结果，增加一条帐号记录 | | |
| Account.login | 语法 | Public Boolean loginr(AccountVO account); | | |
| 前置条件 | 用户输入用户名和密码，希望登录 | | |
| 后置条件 | 返回登录成功，更改帐号在线状态 | | |
| Account.logout | 语法 | Public Boolean logout (String order\_id); | | |
| 前置条件 | 用户处于登录状态，希望退出登录 | | |
| 后置条件 | 返回登出结果，更改帐号在线状态 | | |
| Account.modifyPassword | 语法 | Public boolean modifyPassword (AccountVO account); | | |
| 前置条件 | 已获得需要修改帐号信息 | | |
| 后置条件 | 返回修改结果，更新帐号信息 | | |
| Account.getIdentity | 语法 | Publicstring getIdentity (String userID); | | |
| 前置条件 | 已知帐号信息 | | |
| 后置条件 | 查询数据返回帐号身份 | | |
| Account.userIDExist | 语法 | Public boolean userIDExist (String userID); | | |
| 前置条件 | 用户注册，确定用户名是否合法 | | |
| 后置条件 | 查询数据返回帐号是否存在 | | |
| 需要的服务（需接口） | | | | |
| 服务名 | | | 服务 | |
| Data层 | | |  | |
| AccountDao.addAccount | | | 添加新注册的账户 | |
| AccountDao.updateAccount | | | 更新帐户信息 | |
| AccountDao.getAccountInfo | | | 获取账户信息 | |
| AccountDao.userIDExists | | | 判断用户名是否存在 | |
| AccountDao.hasLogin | | | 判断用户是否处于登录状态，不能在不同地方同时登录一个账号 | |
| AccountDao.setLogin | | | 登录后设置登录状态 | |
| AccountDao.setLogout | | | 登出后设置登出状态 | |
|  | | |  | |

表5.3.2-4 room\_bl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Room. addRoomInfo | 语法 | Public ResultMessage addRoomInfo(RoomVO room); | |
| 前置条件 | 已获取要增加的房间信息 | |
| 后置条件 | 返回增加结果，增加一个房间信息 | |
| Room. updateRoomInfo | 语法 | Public ResultMessage updateRoomInfo(RoomVO room); | |
| 前置条件 | 已获取要更新的房间信息 | |
| 后置条件 | 返回更新成功与否，更改房间信息 | |
| Room. getRoomInfo | 语法 | Public RoomVO getRoomInfo(String hotelID, RoomType roomType); | |
| 前置条件 | 酒店管理人员已经登录，想要获取酒店的房间信息 | |
| 后置条件 | 返回房间信息 | |
| Room. getRoomList | 语法 | publicArrayList<RoomVo room>getRoomList(String hotelID); | |
| 前置条件 | 已知酒店id | |
| 后置条件 | 查找信息并返回改酒店的所有房间信息 | |
| Room. getRemainingRoomNums | 语法 | Public int getRemainingRoomNums (String hotelID , RoomType roomType); | |
| 前置条件 | 酒店管理人员已经登录，想要获得所有对应类型的房间的空房数量 | |
| Room. getSpecificTimeRemainingRoomNums | 语法 | Public int getSpecificTimeRemainingRoomNums (String hotelID , RoomType roomType， String time); | |
| 前置条件 | 酒店管理人员已经登录，想要获得所有对应类型的房间特定日期的空房数量 | |
| 后置条件 | 查找信息并返回改酒店对应类型房间特定日期的剩余数量 | |
| UpdateRoom.UpdateRoomInSpecificTime | 语法 | Public ResultMessage UpdateRoomInSpecificTime (String hotelID, RoomType roomType, int changeNum, String time); | |
| 前置条件 | 订单执行成功或者退房成功 | |
| 后置条件 | 更改房间数量信息并返回成功与否 | |
| UpdateRoom.UpdateRoom | 语法 | Public ResultMessage UpdateRoom (String hotelID, RoomType roomType, int changeNum); | |
| 前置条件 | 订单执行成功或者退房成功 | |
| 后置条件 | 时间设置为当天，调用UpdateRoomInSpecificTime方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Data层 | | |  |
| RoomDao.addRoom | | | 添加新录入的房间 |
| RoomDao.updateRoom | | | 更新房间信息 |
| RoomDao.getRoomInfo | | | 获取房间信息 |
| RoomDao.getHotelRooms | | | 获取指定酒店的所有房间信息 |

表5.3.2-5 hotel\_bl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| SearchHotel.getTradingArea | 语法 | publicArrayList<String> getTradingArea (String city); | |
| 前置条件 | 已获取城市名，需要获取城市对应的商圈列表 | |
| 后置条件 | 查看数据并返回商圈列表 | |
| SearchHotel.getInitialHotelList | 语法 | publicArrayList<HotelInfoVO>getInitialHotelList (String location,String tradingArea ); | |
| 前置条件 | 已知地址，商圈 | |
| 后置条件 | 返回该地址该商圈的酒店列表 | |
| SearchHotel.getSortedList | 语法 | publicArrayList<HotelInfoVO>getSortedList (HotelSearchCondition condition, ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知排序条件 | |
| 后置条件 | 返回符合条件的酒店信息 | |
| SearchHotel.getBookedHotelList | 语法 | Public ArrayList<HotelInfoVO>getBookedHotelList (String user\_id); | |
| 前置条件 | 用户已经登录，已知用户id | |
| 后置条件 | 返回预定过的酒店列表 | |
| SearchHotel.search | 语法 | PublicHotelInfoVO search(HotelSearchVO search); | |
| 前置条件 | 已知搜索条件 | |
| 后置条件 | 返回符合条件的酒店信息 | |
| SearchHotel.getCities | 语法 | publicArrayList<String> getCities (); | |
| 前置条件 | 用户已经登录想要搜索酒店 | |
| 后置条件 | 返回所有城市列表 | |
| PriceSort.getSortedList | 语法 | publicArrayList<HotelInfoVO>getSortedList (ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合价格排序条件的列表 | |
| PriceSort.getSpecificSectionHotelList | 语法 | publicArrayList<HotelInfoVO>getSortedList (double startPrice, double endPrice, ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合价格区间条件的酒店列表 | |
| StarSort.getSortedList | 语法 | publicArrayList<HotelInfoVO>getSortedList (ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合星级排序条件的列表 | |
| StarSort.getSpecificSectionHotelList | 语法 | publicArrayList<HotelInfoVO>getSortedList (double start, double end, ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合星级区间条件的酒店列表 | |
| GradeSort.getSortedList | 语法 | publicArrayList<HotelInfoVO>getSortedList (ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合评分排序条件的列表 | |
| GradeSort.getSpecificSectionHotelList | 语法 | publicArrayList<HotelInfoVO>getSortedList (double start, double end, ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合评分区间条件的酒店列表 | |
| UpOrDownSort.compare | 语法 | public boolean compare(double a, double b) | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合星级排序条件的列表 | |
| CheckHotel.getHotelInfo | 语法 | Public HotelVO getHotelInfo (String hotelID); | |
| 前置条件 | 已知酒店名 | |
| 后置条件 | 查看数据返回该酒店信息 | |
| CheckHotel.getHotelTradingArea | 语法 | Public String getHotelTradingArea (String hotel\_id);; | |
| 前置条件 | 促销策略需要获取酒店的商圈信息 | |
| 后置条件 | 返回该酒店的商圈信息 | |
| CheckHotel.getHotelEvalutaions | 语法 | Public ArrayList<EvalutionVO> getHotelEvalutaions (Hotel hotel); | |
| 前置条件 | 客户查看酒店评价信息 | |
| 后置条件 | 返回酒店的评价信息 | |
| CheckHotel.getRoomPrice | 语法 | Public double getRoomPrice (String hotelID ,RoomType roomType); | |
| 前置条件 | 生成订单时需要确定订单的原始价格 | |
| 后置条件 | 调用room类的getRoomPrice的方法 | |
| UpdateHotel.updateHotelInfo | 语法 | public ResultMessage updateHotelInfo (HotelInfoVO hotel); | |
| 前置条件 | 酒店信息需要更改 | |
| 后置条件 | 更新数据库该酒店信息 | |
| UpdateHotel.addHotel | 语法 | public ResultMessage addHotel (HotelInfoVO hotel); | |
| 前置条件 | 网站管理人员输入添加的酒店信息 | |
| 后置条件 | 更新数据并返回成功与否 | |
| UpdateHotel.hotelIDExist | 语法 | public boolean hotelIDExist (String hotelID); | |
| 前置条件 | 添加酒店信息判断酒店id是否存在 | |
| 后置条件 | 查看数据并返回是与否 | |
| UpdateHotel.hotelHasManager | 语法 | public boolean hotelHasManager (String hotelID); | |
| 前置条件 | 添加酒店管理人员判断酒店管理人员是否存在 | |
| 后置条件 | 查看数据并返回是与否 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderHotelInfo.getHotelEvalutions | | | 获取酒店评价信息 |
| OrderHotelInfo.getBookedHotelList | | | 获取酒店评价信息 |
| PriceSort.getSortedList | | | 按价钱排序 |
| StarSort.getSortedList | | | 按星级排序 |
| GradeSortSort.getSortedList | | | 按评分排序 |
| UpOrDownSort | | | 升序或降序 |
| RoomInfo.getHotelLowestPrice | | | 获取酒店房间的最低价格 |
| RoomInfo.getRoomPrice | | | 获得指定酒店指定房间类型的价格 |
| OrderHotelInfo.getHotelEvalutions | | | 获取酒店评价信息 |
| OrderHotelInfo.getBookHotelList | | | 获取制定客户预订过的酒店的id |
| Data层 | | |  |
| HotelDao.searchHotelList | | | 获取酒店列表 |
| HotelDao.getHotelInfoByHotelID | | | 获取酒店详细 |
| HotelDao.addHotel | | | 添加酒店信息 |
| HotelDao.updateHotel | | | 更改酒店信息 |
| HotelDao.getCities | | | 获取所有城市 |
| HotelDao.getTradingAreas | | | 获取指定城市的商圈列表 |
| HotelDao.getHotelByHotelId | | | 获取酒店详细 |
| HotelDao.getHotelByHotelId | | | 获取酒店信息 |
| HotelDao.updateHotel | | | 更新酒店信息 |
| HotelDao.addHotel | | | 添加酒店信息 |

表5.3.2-6 order\_bl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Order.getOrderInfo | 语法 | public OrderVO getOrderInfo(String orderID); | |
| 前置条件 | 系统已创建订单 | |
| 后置条件 | 查询订单并返回订单信息 | |
| Order.getEvaluationInfo | 语法 | public EvaluationVO getEvaluationInfo(String orderID); | |
| 前置条件 | 客户对订单进行评价 | |
| 后置条件 | 查询评价并返回评价信息 | |
| Order. evaluate | 语法 | public ResultMessage evaluate(EvaluationVO evaluation); | |
| 前置条件 | 系统已创建订单 | |
| 后置条件 | 用户得到评价成功与否的结果 | |
| Order. getHotelEvaluations | 语法 | public ArrayList<EvaluationVO> getHotelEvaluations(String hotelID) | |
| 前置条件 | 酒店存在被评价记录 | |
| 后置条件 | 用户请求查看酒店评价 | |
| Order. getBookedHotelList | 语法 | public ArrayList<String> getBookedHotelList(String userID); | |
| 前置条件 | 被预定过酒店存在被评价记录 | |
| 后置条件 | 用户查询预定过酒店的评价记录 | |
| OrderList.filterUserOrderList | 语法 | public ArrayList<OrderVO> filterUserOrderList(String userID, OrderListCondition condition); | |
| 前置条件 | 用户有预定酒店的记录 | |
| 后置条件 | 网站营销人员查询今日订单 | |
| Order. filterHotelOrderList | 语法 | public ArrayList<OrderVO> filterHotelOrderList(String hotelID, OrderListCondition condition); | |
| 前置条件 | 用户有预定酒店的记录 | |
| 后置条件 | 酒店工作人员查询今日订单 | |
| Order. filterSpecificUserHotelOrderList | 语法 | public ArrayList<OrderVO> filterSpecificUserHotelOrderList(String userID, String hotelID); | |
| 前置条件 | 用户有预定酒店的历史纪录 | |
| 后置条件 | 用户查询历史订单记录 | |
| Order. filterWebDailyOrderList | 语法 | public ArrayList<OrderVO> filterWebDailyOrderList(OrderListCondition condition) | |
| 前置条件 | 用户预定酒店 | |
| 后置条件 | 网站营销人员查询当日订单 | |
| CreateOrder.createOrder | 语法 | Public OrderVO createOrder(OrderVO order); | |
| 前置条件 | 用户已注册 | |
| 后置条件 | 用户下订单 | |
| Order. judgeCreditCanCreateOrder | 语法 | public boolean judgeCreditCanCreateOrder(String userID) | |
| 前置条件 | 用户已注册 | |
| 后置条件 | 用户下订单 | |
| ExecuteOrder. hasCheckOut | 语法 | public ResultMessage hasCheckOut(String orderID) | |
| 前置条件 | 用户退房 | |
| 后置条件 | 酒店管理人员设置退房信息 | |
| ExecuteOrder. checkOut | 语法 | public ResultMessage checkOut(String orderID) | |
| 前置条件 | 用户退房 | |
| 后置条件 | 酒店管理人员更新退房信息 | |
| ExecuteOrder. normalExecute | 语法 | public ResultMessage normalExecute(String orderID, String[] roomIDs); | |
| 前置条件 | 用户入住 | |
| 后置条件 | 酒店管理人员执行订单 | |
| ExecuteOrder. supplyOrder | 语法 | public ResultMessage supplyOrder(String orderID, String[] roomIDs); | |
| 前置条件 | 用户超时未入住 | |
| 后置条件 | 酒店管理人员补登记订单 | |
| ExecuteOrder. undoAbnormalOrder | 语法 | public ResultMessage undoAbnormalOrder(String orderID, boolean recoverAllDeletedCredit) | |
|  | 前置条件 | 用户超时未执行订单 | |
|  | 后置条件 | 网站营销人员撤销异常订单 | |
| ExecuteOrder. undoUnexecutedOrder | 语法 | public ResultMessage undoUnexecutedOrder(String orderID); | |
|  | 前置条件 | 用户已下订单，未执行 | |
|  | 后置条件 | 用户撤销订单 | |
| OrderListFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 需要筛选指定条件的订单 | |
| 后置条件 | 返回订单列表 | |
| AbnormalFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 有全部订单列表，需要筛选异常的订单 | |
| 后置条件 | 返回异常订单列表 | |
| ExecutedFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 有全部订单列表，需要筛选已执行的订单 | |
| 后置条件 | 返回已执行订单列表 | |
| UnexecutedFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 有全部订单列表，需要筛选未执行的订单 | |
| 后置条件 | 返回未执行订单列表 | |
| UndoedFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 有全部订单列表，需要筛选撤销未执行的订单 | |
| 后置条件 | 返回撤销未执行订单列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Data层 | | |  |
| OrderDao.addEvaluation | | | 添加订单评价 |
| OrderDao.getOrderByOrderID | | | 获取订单信息 |
| OrderDao.getEvalutionByOrderID | | | 获取订单评价 |
| OrderDao.getBookedHotelList | | | 获取指定用户预订过的酒店id |
| OrderListDao.getOrderListByUserID | | | 根据用户id获取订单 |
| OrderListDao.getOrderListByHotelID | | | 根据酒店id获取订单 |
| OrderListDao.getAllDailyOrders | | | 获取当天全部订单 |
| CreditChange.changeCredit | | | 更新用户信用值和信用记录 |
| CalculatePromtionInfo.calculatePromotion | | | 生成订单时计算优惠信息 |
| UpdateRoom.updateRoom | | | 更新房间信息 |
| HotelInfo.getRoomPrice | | | 获取房间信息 |
| OrderDao.getOrderNum | | | 获取订单次序，用于生成订单id |
| OrderDao.addOrder | | | 添加订单 |
| ExecuteOrder.undoAbnormalOrder | | | 撤销异常订单 |
| OrderList.filterWebDailyOrderList | | | 筛选当天全部订单 |
|  | | |  |

表5.3.2-7 promotion\_bl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CalculatePromotion. addPromotion | 语法 | | public ResultMessage addPromotion(PromotionVO vo) |
| 前置条件 | | 网站营销人员和酒店工作人员添加促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion. getPromotion | 语法 | | public PromotionVO getPromotion(String promotionID) |
| 前置条件 | | 网站营销人员查看促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion. getHotelPromotions | 语法 | | public ArrayList<PromotionVO> getHotelPromotions(String hotelID, PromotionType promotionType) |
| 前置条件 | | 酒店工作人员查看酒店促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion. getWebPromotions | 语法 | | public ArrayList<PromotionVO> getWebPromotions(PromotionType promotionType) |
| 前置条件 | | 网站营销人员查看网站促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion. deletePromotion | 语法 | | public ResultMessage deletePromotion(String promotionID) |
| 前置条件 | | 网站营销人员和酒店工作人员删除促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion. calculatePromotion | 语法 | | public OrderVO calculatePromotion(OrderVO order) |
| 前置条件 | | 用户下订单 |
| 后置条件 | | 系统生成订单 |
| Promotion.judgePromotion | 语法 | | PublicBoolean judgePromotion(OrderVO orderVo); |
| 前置条件 | | 生成订单，需要计算促销 |
| 后置条件 | | 返回是否满足该促销的使用条件 |
| Promotion.calculate | 语法 | | Public doublecalculate\_Promotionid(Order order); |
| 前置条件 | | 生成订单，需要计算促销 |
| 后置条件 | | 计算订单促销策略后返回新的订单信息 |
| Promotion. changeToVO | 语法 | | public PromotionVO changeToVO() |
| 前置条件 | | 查看促销策略 |
| 后置条件 | | 生成促销策略内容返回新的策略信息 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| Data层 | |  | |
| PromotionDao.getPromotionNum | | 获取promotion的次序 | |
| PromotionDao.addPromotion | | 添加promotion | |
| PromotionDao.getHotelPromotions | | 获取指定酒店的promotion列表 | |
| PromotionDao.getWebPromtions | | 获取网站promotions | |
| PromotionDao.deletePromotion | | 删除指定促销策略 | |
| HotelBirthdayPromotion.calculate | | 计算酒店会员生日折扣 | |
| HotelBusinessPromotino. calculate | | 计算酒店会员合作企业折扣 | |
| HotelDoubleElevenPromotion. calculate | | 计算酒店双11折扣 | |
| HotelThreeRoomsPromotion. calculate | | 计算酒店预订3间房间及以上折扣 | |
| WebDoubleElevenPromotion. calculate | | 计算网站双11折扣 | |
| WebVipLevelPromtion. calculate | | 计算网站会员等级折扣 | |
| WebVipTradingAreaPromotion. calculate | | 计算网站不同会员等级不同商圈不同折扣的优惠 | |
| Data层 | |  | |
| PromotionDao.getWebPromotions | | 获取网站所有促销信息 | |
| PromotionDao.getHotelPromotions | | 获取指定酒店促销促销信息 | |
| HotelBirthdayPromotion.judgePromtion | | 判断订单是否满足酒店会员生日折扣 | |
| HotelBirthdayPromotion.calculate | | 计算酒店会员生日折扣 | |
| HotelBusinessPromotino. judgePromtion | | 判断订单是否满足酒店会员合作企业折扣 | |
| HotelBusinessPromotino. calculate | | 计算酒店会员合作企业折扣 | |
| HotelDoubleElevenPromotion. judgePromtion | | 判断订单是否满足酒店双11折扣 | |
| HotelDoubleElevenPromotion. calculate | | 计算酒店双11折扣 | |
| HotelThreeRoomsPromotion. judgePromtion | | 判断订单是否满足酒店预订3间房间及以上折扣 | |
| HotelThreeRoomsPromotion. calculate | | 计算酒店预订3间房间及以上折扣 | |
| WebDoubleElevenPromotion. judgePromtion | | 判断订单是否满足网站双11折扣 | |
| WebDoubleElevenPromotion. calculate | | 计算网站双11折扣 | |
| WebVipLevelPromtion. judgePromtion | | 判断订单是否满足网站会员等级折扣 | |
| WebVipLevelPromtion. calculate | | 计算网站会员等级折扣 | |
| WebVipTradingAreaPromotion. judgePromtion | | 判断订单是否满足网站不同会员等级不同商圈不同折扣的优惠 | |
| WebVipTradingAreaPromotion. calculate | | 计算网站不同会员等级不同商圈不同折扣的优惠 | |

## 5.4 数据层的分解

数据层主要给业务逻辑层提供数据防伪服务，包括对于持久化数据的增、删、改、查。User业务逻辑需要的服务由UserDataService接口提供。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。数据层模块的具体描述如图5.4所示。

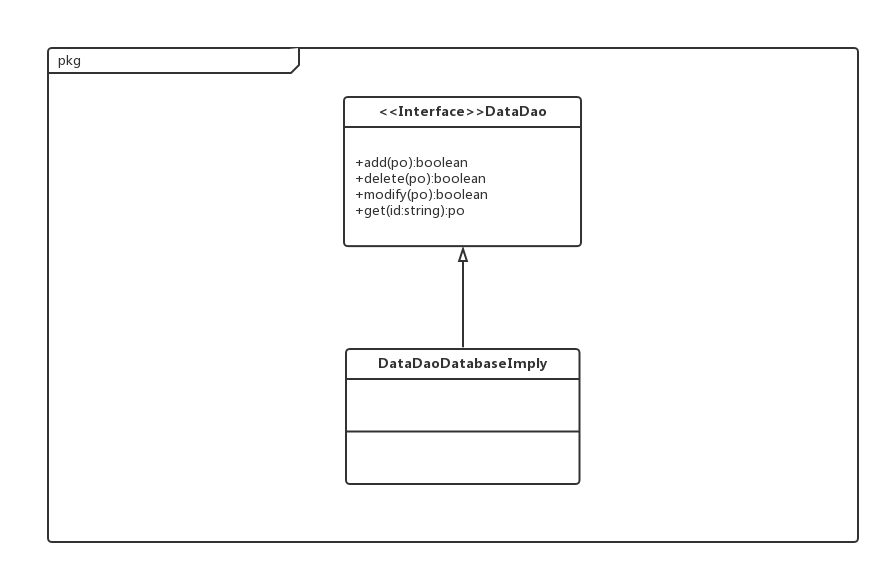


图5.4 数据层模块的描述

### 5.4.1 数据层模块的职责

数据层模块的职责如表5.4.1所示。

表5.4.1 数据层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| DataDao | 持久化数据库的接口，提供集体载入，集体保存、增、删、改、查服务。 |
| DataDaoDatabaseImpl | 基于数据库的持久化数据的接口，提供集体载入、集体保存、增、删、改、查服务 |

### 5.4.2 数据层模块的接口规范

表5.4.2-1数据层credit\_data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CreditDao.changeCredit | 语法 | public boolean changeCredit (CreditHistoryPO po) throws RemoteException; |
| 前置条件 | 已获得信用修改的po |
| 后置条件 | 在数据库中增加一个信用值修改记录 |
| CreditDao.getCreditHistory | 语法 | public ArrayList<CreditHistoryPO> getCreditHistory(String userID) throws RemoteException **;** |
| 前置条件 | 逻辑层获取账户的信用修改记录 |
| 后置条件 | 获取修改记录并返回 |
| CreditDao. getCredit | 语法 | public int getCredit(String userID) throws RemoteException**；** |
| 前置条件 | 逻辑层请求查看账户信用值 |
| 后置条件 | 返回当前客户的信用值 |
| CreditDao. | 语法 | public boolean setVIPCredit(int level, int credit\_num) throws RemoteException; |
| 前置条件 | 逻辑层请求修改会员信用等级 |
| 后置条件 | 数据库更新数据 |
| CreditDao. | 语法 | public int getVIPCredit(int level) throws RemoteException; |
| 前置条件 | 逻辑层请求获取会员等级对应的信用值 |
| 后置条件 | 返回会员等级对应的信用值 |

表5.4.2-2数据层user\_data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| ClientDao. addClient | 语法 | public boolean addClient(ClientPO po) throws RemoteException; |
| 前置条件 | 逻辑层请求增加一个用户 |
| 后置条件 | 数据库增加一个用户信息记录 |
| ClientDao. getClientInfo | 语法 | public ClientPO getClientInfo (String clientID) throws RemoteException; |
| 前置条件 | 逻辑层请求获取用户信息 |
| 后置条件 | 数据层返回用户信息 |
| ClientDao. updateClientInfo | 语法 | public boolean updateClientInfo (ClientPO clientPO) throws RemoteException; |
| 前置条件 | 逻辑层请求更新用户信息 |
| 后置条件 | 数据层更新用户信息 |
| HotelManagerDao. getHotelManagerInfo | 语法 | public HotelManagerPO getHotelManagerInfo(String hotel\_ID) throws RemoteException; |
| 前置条件 | 逻辑层请求获取酒店管理人员信息 |
| 后置条件 | 数据层返回酒店管理人员信息 |
| HotelManagerDao. updateHotelManagerInfo | 语法 | public boolean updateHotelManagerInfo (HotelManagerPO po) throws RemoteException; |
| 前置条件 | 逻辑层请求更新酒店管理人员信息 |
| 后置条件 | 数据层更新酒店管理人员信息 |
| HotelManagerDao. addHotelManager | 语法 | public boolean addHotelManager (HotelManagerPO hotelManagerPO) throws RemoteException; |
| 前置条件 | 逻辑层请求增加一个酒店管理人员 |
| 后置条件 | 数据库增加一个酒店管理人员信息记录 |
| WebBusinessDao. getWebBusinessInfo | 语法 | public WebBusinessPO getWebBusinessInfo(String webBusiness\_ID) throws RemoteException; |
|  | 前置条件 | 逻辑层请求获取网站营销人员信息 |
|  | 后置条件 | 数据层返回网站营销人员信息 |
| WebBusinessDao. updateWebBusinessInfo | 语法 | public boolean updateWebBusinessInfo(WebBusinessPO webBusinessInfo) throws RemoteException; |
|  | 前置条件 | 逻辑层请求更新网站营销人员信息 |
|  | 后置条件 | 数据层更新网站营销人员信息 |
| WebBusinessDao. addWebBusiness | 语法 | public boolean addWebBusiness(WebBusinessPO webBusinessPO) throws RemoteException; |
|  | 前置条件 | 逻辑层请求增加一个网站营销人员 |
|  | 后置条件 | 数据库增加一个网站营销人员信息记录 |

表5.4.2-3数据层account\_data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AccountDao.addAccount | 语法 | public Boolean addAccount (AccountPO po) throws RemoteException; |
| 前置条件 | 逻辑层请求增加一个账户 |
| 后置条件 | 数据库增加账户 |
| AccountDao. userIDExists | 语法 | public boolean userIDExists (String userID) throws RemoteException; |
| 前置条件 | 逻辑层获取该用户名账户是否已经存在 |
| 后置条件 | 数据库判断是否存在该帐号 |
| AccountDao.modifyPassword | 语法 | public boolean modifyPassword (AccountPO po) throws RemoteException; |
| 前置条件 | 逻辑层请求修改账户密码 |
| 后置条件 | 数据库修改账户密码 |
| AccountDao. getAccountInfo | 语法 | Public AccountPO getAccountInfo (String accountID) throws RemoteException; |
|  | 前置条件 | 逻辑层请求获取帐号信息 |
|  | 后置条件 | 数据库返回帐号信息 |
| AccountDao. hasLogin | 语法 | public boolean hasLogin(String userID) throws RemoteException; |
|  | 前置条件 | 逻辑层查询账户是否已经登录 |
|  | 后置条件 | 数据库判断是否已经登录 |
| AccountDao. setLogout | 语法 | public boolean setLogout(String userID) throws RemoteException; |
|  | 前置条件 | 逻辑层请求设置帐号为登出 |
|  | 后置条件 | 数据库修改帐号登录状态 |
| AccountDao. setLogin | 语法 | public boolean setLogin(String userID) throws RemoteException; |
|  | 前置条件 | 逻辑层请求设置帐号为登录 |
|  | 后置条件 | 数据库修改帐号登录状态 |

表5.4.2-4数据层room\_data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| RoomDao. getRoomInfo | 语法 | public RoomPO getRoomInfo (String hotelId, int roomType) throws RemoteException; |
| 前置条件 | 逻辑层请求调用酒店某一种类客房信息 |
| 后置条件 | 数据层返回客房信息 |
| RoomDao. addRoom | 语法 | public boolean addRoom (RoomPO roomPO) throws RemoteException; |
| 前置条件 | 逻辑层请求增加客房信息 |
| 后置条件 | 数据库增加客房信息 |
| RoomDao. updateRoom | 语法 | public boolean updateRoom (RoomPO po) throws RemoteException; |
| 前置条件 | 逻辑层请求更新客房信息 |
| 后置条件 | 数据库更新客房信息 |
| RoomDao. getHotelRooms | 语法 | public ArrayList<RoomPO> getHotelRooms(String hotelId) throws RemoteException; |
|  | 前置条件 | 逻辑层获取酒店房间信息 |
|  | 后置条件 | 数据层返回酒店房间信息 |

表5.4.2-5数据层hotel\_data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelDao. addHotel | 语法 | public boolean addHotel(HotelPO po) throws RemoteException; |
| 前置条件 | 逻辑层请求增加一家酒店 |
| 后置条件 | 数据库中增加一个hotel记录 |
| HotelDao. updateHotel | 语法 | public boolean updateHotel (HotelPO po) throws RemoteException; |
| 前置条件 | 逻辑层请求更新一家酒店信息 |
| 后置条件 | 数据库更新该酒店信息 |
| HotelDao. getHotelInfoByHotelID | 语法 | public HotelPO getHotelInfoByHotelID (String hotelID) throws RemoteException; |
| 前置条件 | 逻辑层请求符合条件的酒店信息 |
| 后置条件 | 返回具体酒店信息 |
| HotelDao. SearchHotelList | 语法 | public ArrayList<HotelPO> SearchHotelList (String city,String tradingArea) throws RemoteException; |
| 前置条件 | 逻辑层请求符合条件的酒店列表 |
| 后置条件 | 返回符合条件的酒店列表 |
| HotelDao. hotelIDExist | 语法 | public boolean hotelIDExist(String hotelID) throws RemoteException; |
| 前置条件 | 逻辑层请求判断酒店是否存在 |
| 后置条件 | 返回是否存在 |
| HotelDao. getCities | 语法 | public ArrayList<String> getCities() throws RemoteException; |
| 前置条件 | 逻辑层请求获取所有成熟 |
| 后置条件 | 数据层返回所有城市列表 |
| HotelDao. getTradingAreas | 语法 | public ArrayList<String> getTradingAreas (String city) throws RemoteException; |
| 前置条件 | 逻辑层请求获取该城市所有商圈 |
| 后置条件 | 数据层返回商圈列表 |

表5.4.2-6数据层order\_data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderDao. getOrderNum | 语法 | public int getOrderNum() throws RemoteException; |
| 前置条件 | 逻辑层请求获取订单总数量 |
| 后置条件 | 数据层返回订单总数量 |
| OrderDao. addOrder | 语法 | public boolean addOrder(OrderPO orderPO) throws RemoteException; |
| 前置条件 | 逻辑层请求增加订单 |
| 后置条件 | 数据库增加订单数据 |
| OrderDao.  updateOrder | 语法 | public boolean updateOrder (OrderPO orderPO) throws RemoteException; |
| 前置条件 | 逻辑层请求更新订单信息 |
| 后置条件 | 数据层更新订单信息 |
| OrderDao. getOrderByOrderID | 语法 | public OrderPO getOrderByOrderID (String orderID)throws RemoteException; |
| 前置条件 | 逻辑层请求获取订单信息 |
| 后置条件 | 数据层返回订单信息 |
| OrderDao. addEvaluation | 语法 | public boolean addEvaluation (EvaluationPO po) throws RemoteException; |
| 前置条件 | 逻辑层请求增加评价信息 |
| 后置条件 | 数据库增加评价信息 |
| OrderDao. getEvaluationByOrderID | 语法 | public EvaluationPO getEvaluationByOrderID(String orderID) throws RemoteException; |
| 前置条件 | 逻辑层请求获取订单评价信息 |
| 后置条件 | 数据层返回订单评价信息 |
| OrderDao. getEvaluationByHotelID | 语法 | public ArrayList<EvaluationPO> getEvaluationByHotelID(String hotelID) throws RemoteException; |
| 前置条件 | 逻辑层请求获取酒店评价信息 |
| 后置条件 | 数据层返回酒店评价信息 |
| OrderDao. getBookedHotelList | 语法 | public ArrayList<String> getBookedHotelList (String userID) throws RemoteException; |
| 前置条件 | 逻辑层请求获取用户预定过的酒店 |
| 后置条件 | 数据层返回用户预定过的酒店列表 |
| OrderListDao. getOrderListByUserId | 语法 | public ArrayList<OrderPO> getOrderListByUserId(String userID) throws RemoteException; |
| 前置条件 | 逻辑层请求获取用户的所有订单 |
| 后置条件 | 数据层返回用户的所有订单信息列表 |
| OrderListDao. getOrderListByHotelID | 语法 | public ArrayList<OrderPO> getOrderListByHotelID(String hotelID) throws RemoteException; |
| 前置条件 | 逻辑层请求获取酒店所有的订单 |
| 后置条件 | 数据层返回酒店所有订单信息列表 |
| OrderListDao. getAllDailyOrders | 语法 | public ArrayList<OrderPO> getAllDailyOrders(String time) throws RemoteException; |
|  | 前置条件 | 逻辑层请求获取酒店今日的所有订单 |
|  | 后置条件 | 数据层返回酒店今日的所有订单信息列表 |

表5.4.2-8数据层promotion\_data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionDao. getHotelPromotions | 语法 | public ArrayList<PromotionPO> getHotelPromotions(String hotel, int promotionType) throws RemoteException; |
| 前置条件 | 逻辑层请求获取酒店的所有相应类型的促销策略 |
| 后置条件 | 数据层返回酒店的所有相应类型的促销策略列表 |
| PromotionDao. getWebPromotions | 语法 | public ArrayList<PromotionPO> getWebPromotions(int promotionType) throws RemoteException; |
| 前置条件 | 逻辑层请求获取网站所有相应类型的促销策略 |
| 后置条件 | 数据层返回网站所有相应类型的促销策略列表 |
| PromotionDao. getPromotion | 语法 | public PromotionPO getPromotion (String promotionID) throws RemoteException; |
| 前置条件 | 逻辑层请求获取对应的促销策略信息 |
| 后置条件 | 数据层返回对应的促销策略信息 |
| PromotionDao. addPromotion | 语法 | public boolean addPromotion(PromotionPO promotionPO) throws RemoteException; |
| 前置条件 | 逻辑层请求添加促销策略 |
| 后置条件 | 数据库添加促销策略数据 |
| PromotionDao. updatePromotion | 语法 | public boolean updatePromotion(PromotionPO promotionPO) throws RemoteException; |
| 前置条件 | 逻辑层请求更新促销策略信息 |
| 后置条件 | 数据库更新促销策略信息 |
| PromotionDao. getPromotinoNum | 语法 | public int getPromotinoNum() throws RemoteException; |
| 前置条件 | 逻辑层请求获取是第几个订单，用于生成订单promotionID每次调用完data层数字加一 |
| 后置条件 | 数据层返回数据库促销策略数量 |

# 6、信息视角

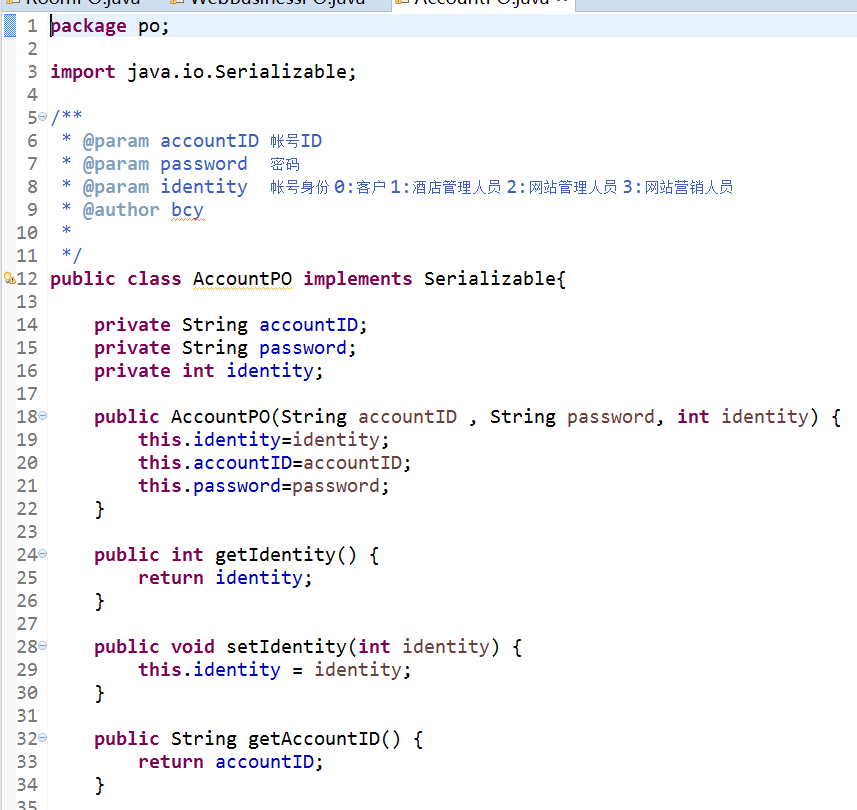
## 6.1 数据持久化对象

系统的PO类就是对应的相关的实体类，在此只做简单的介绍，如表6.1所示：

表6.1 PO类简单介绍

|  |  |
| --- | --- |
| 类名 | 包含的属性 |
| AccountPO | 帐号Id，密码，帐号身份 |
| ClientPO | 用户id，电话，真实姓名，身份证号，会员类型，会员等级，会员信息 |
| CreditHistoryPO | 用户ID，信用修改时间，订单ID，操作，信用变化值，当前信用 |
| EvaluationPO | 订单ID，评价星级，评价信息 |
| HotelManagerPO | 酒店Id，电话号码，真实姓名，身份证号 |
| HotelPO | 酒店ID，酒店名称，城市，所属商圈，酒店地址，评价等级，酒店星级，酒店介绍，酒店设施，合作企业 |
| OrderPO | 用户ID，订单号，酒店ID，开始时间，结束时间，分配的房间Id，酒店房间数量，房间类型，是否有小孩，入住人数，订单状态，促销前价格，促销后价格，使用的促销策略ID，客户入住时间，客户退房时间，异常订单撤销时间，订单被置为异常时间，未执行订单被用户撤销时间 |
| PromotionPO | 促销类型，促销Id，酒店ID,促销名称，折扣，开始时间，结束时间，合作企业，多间房间折扣，vip商圈折扣，vip等级折扣 |
| RoomPO | 酒店ID，房间类型，房间数量，价格，接下来30天对应的房间数量 |
| WebBusinessPO | 用户id，真实姓名，电话号码，身份证号 |

持久化用户对象如UserPO的定义如图6.1，更多定义见原型代码



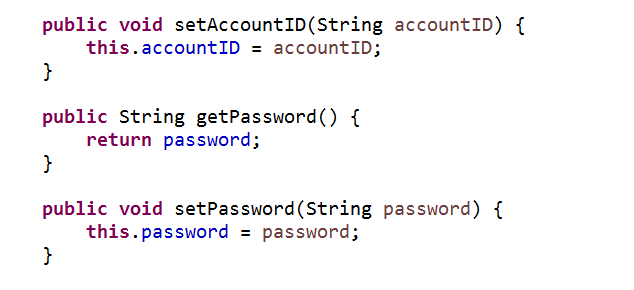


图6.1 持久化用户对象UserPO的定义

## 6.2 数据库表

|  |  |
| --- | --- |
| 表名 | 内容 |
| account | 帐号信息 |
| Client | 用户信息 |
| Credithistory | 信用修改记录 |
| Evaluation | 评价信息 |
| Hotel | 酒店信息 |
| Hotelmanager | 酒店管理人员信息 |
| Locations | 地址及商圈 |
| Orders | 订单信息 |
| Promotion | 促销策略 |
| Room | 酒店房间信息 |
| vipLevel | Vip等级对应的信用值 |
| Webbusiness | 网站营销人员信息 |