HRMS

酒店预订管理系统

体系结构设计文档

学 院：南京大学软件学院

LLLH小组

成 员：卢忆卿 李紫欣 刘钰 黄飘

完成日期：2016/10/15

目录

更新历史 5

1.引言 6

1.1编制目的 6

1.2词汇表 6

1.3参考资料 6

2.产品概述 7

3.逻辑视角 8

4.组合视角 9

4.1开发包图 11

4.2运行时进程 16

4.3物理部署 17

5.接口视角 18

5.1模块的职责 18

5.2用户界面层的分解 21

5.2.1 用户界面层模块的职责 23

5.2.2用户界面层模块的接口规范 24

**表5.2.2-1 用户界面层模块的接口规范** 24

**表5.2.2-2 用户界面层模块需要的服务接口** 25

5.2.3 用户界面模块设计原理 27

5.3 业务逻辑层的分解 27

**表5.3.1-1 业务逻辑层模块的职责** 28

5.3.2 业务逻辑层模块的接口规范 29

**表5.3.2-1Loginbl模块的接口规范** 29

**表5.3.2-2Clientbl模块的接口规范** 29

**表5.3.2-3Orderbl模块的接口规范** 33

**表5.3.2-4Sitemanagerbl模块的接口规范** 39

**表5.3.2-5Logbl模块的接口规范** 44

**表5.3.2-6HotelStaffbl模块的接口规范** 45

**表5.3.2-7Hotelinfobl模块的接口规范** 50

**表5.3.2-8Marketingbl模块的接口规范** 53

**表5.3.2-9Strategybl模块的接口规范** 59

5.4 数据层的分解 62

5.4.1 数据层模块的职责 63

5.4.2 数据层模块的接口规范 64

**表5.4.2-1数据层Clientdata模块的接口规范** 64

**表5.4.2-2数据层Orderdata模块的接口规范** 66

**表5.4.2-3数据层Logdata模块的接口规范** 70

**表5.4.2-4数据层Sitemanagerdata模块的接口规范** 70

**表5.4.2-5数据层HotelStaffdata模块的接口规范** 70

**表5.4.2-6数据层Hotelinfodata模块的接口规范** 71

**表5.4.2-7数据层Marketingdata模块的接口规范** 73

**表5.4.2-8数据层Strategydata模块的接口规范** 76

6、信息视角 78

6.1 数据持久化对象 78

**表6.1 PO类简单介绍** 78

6.2 数据库表 81

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 修改日期 | 修改原因 | 版本号 |
| 全体成员 | 2016/10/13 | 最初草稿 | V0.0.0 |
| 李紫欣 | 2016/10/15 | 排版 | V0.0.1 |
| 刘钰 | 2016/10/16 | 增加了signup包 | V0.0.2 |
| 全体人员 | 2016/10/20 | 删除了signup包，放到了login里面 | V0.0.3 |

# 1.引言

## 1.1编制目的

本报告详细完成对酒店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 | 词汇含义 | 备注 |
| HRMS | 酒店管理系统 |  |
| 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

参考酒店管理系统用例文档和酒店管理系统需求规格说明文档中对产品的概括描述。酒店管理系统主要功能见用例图如下。

# 3.逻辑视角

酒店管理系统中，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2166]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：

* Order和Client:将部分Order类抽象接口orderInfoService置入Client包，这样Order单向依赖于Client（实现接口+调用）
* Marketing和Client:将部分Marketing类抽象接口marketingInfoService置入Client包，这样Marketing单向依赖于Client（实现接口+调用）

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，最终建立的酒店管理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 | clientui,hotelstuffui,marketingui,sitemanagerui,orderui,strategyui,hotelinfoui, signupui,VO |
| loginui | loginblservice,界面类库包,VO |
| loginblservice |  |
| loginbl | loginblservice,PO,clientdataservice,hotelstaffdataservice,marketingdataservice, sitemanagerdataservice |
| clientui | orderblservice,loginblservice,界面类库包,VO |
| clientblservice |  |
| clientbl | clientblservice,orderblservice,hotelInfoblservice |
| clientdataservice | Java RMI,PO |
| clientdata | PO, clientdataservice, databaseutility,orderdataservice |
| hotelstaffui | hotelstaffblservice,loginblservice,界面类库包, VO |
| hotelstaffblservice |  |
| hotelstaffbl | Hotelstaffblservice,hotelstaffdataservice, PO  strategybl,orderbl,hotelinfobl |
| hotelstaffdataservice | JavaRMI, PO |
| hotelstaffdata | databaseutility,po,JavaRMI |
| marketingui | marketingblservice,loginblservice界面类库包, VO |
| matketingblservice |  |
| marketingbl | marketingblservice,marketingdataservice,po,clientbl,orderbl,strategybl |
| marketingdataservice | Java RMI, PO |
| marketingdata | Java RMI,po,databaseutility |
| sitemanagerui | sitemanagerblservice,loginblservice,界面类库包包, VO |
| sitemanagerblservice |  |
| sitemanagerbl | sitemanagerblservice,sitemanagerdataservice,po,clientbl,hotelstaffbl,hotelinfobl,marketingbl |
| sitemanagerdataservice | Java RMI, PO |
| sitemanagerdata | Java RMI, PO, databaseutility |
| orderui | orderblservice,界面类库包, VO |
| orderblservice |  |
| orderbl | orderblservice,orderdataservice,PO,clientbl,hotelinfobl, marketingbl,strategybl |
| orderdataservice | Java RMI, PO |
| orderdata | Java RMI, PO, databaseutility |
| hotelinfoui | hotelinfoblservice,界面类库包, VO |
| hotelinfoblservice |  |
| hotelinfobl | hotelinfoblservice,hotelinfodataservice, PO |
| hotelinfodataservice | Java RMI, PO |
| hotelinfodata | databaseutility,po,Java RMI |
| strategyui | strategyblservice, 界面类库包, VO |
| strategyblservice |  |
| strategybl | strategyblservice,strategydataservice,po,hotelinfobl,clientbl |
| strategydataservice | Java RMI, PO |
| strategydata | Java RMI,po,databaseutility |
| logblservice |  |
| logbl | logblservice,logdataservice,po |
| logdataservice | Java RMI, PO |
| logdata | Java RMI,po,databaseutility |
| vo |  |
| po |  |
| Utilitybl |  |
| 界面类库包 |  |
| Java RMI |  |
| databaseutility | JDBC |

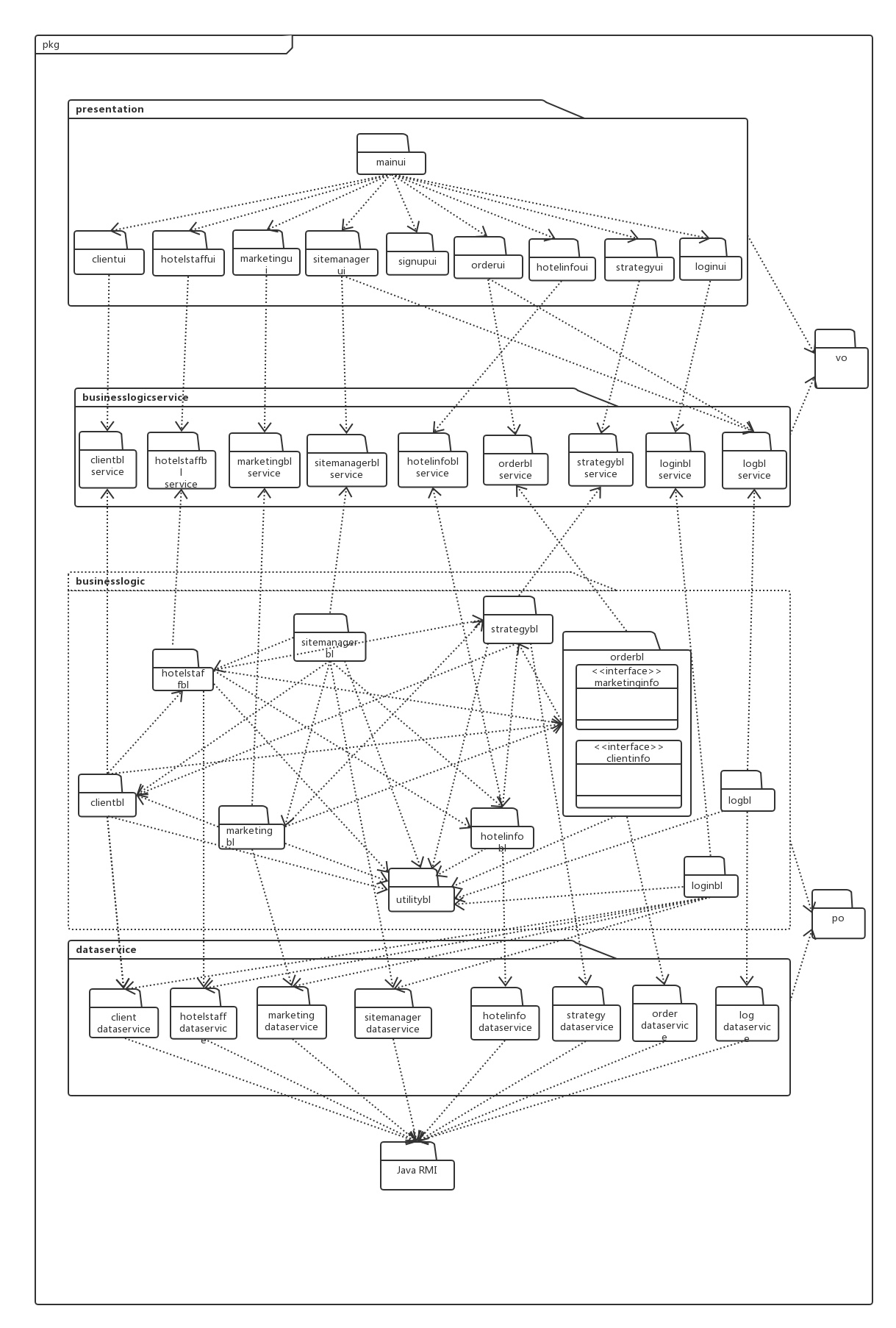


图4.1-1 酒店管理系统客户端开发包图



图4.1-2 酒店管理系统服务器端开发包图

## 4.2运行时进程

如图4-2，在酒店管理系统中，会有多个客户端进程和一个服务器端进程。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

酒店管理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JDK8.0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。



图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 |
| clientblservice  hotelstaffblservice  marketingblservice  sitemanagerblservice  orderblservice  strategyblservice  hotelinfoblservice  loginblservice  signupblservice  logblservice | 客户端展示层 | 客户端业务逻辑层 |
| clientdataservice  hotelstaffdataservice  marketingdataservice  sitemanagerdataservice  orderdataservice  strategydataservice  hotelinfodataservice  logdataservice | 客户端业务逻辑层 | 服务器端数据层 |

借用客户查询订单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）。ClientBLService提供了Client界面所需要的所有业务逻辑功能ClientDataService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

## 5.2用户界面层的分解

根据需求，系统存在36个用户界面：登录界面，客户界面，查看个人订单界面，查询酒店界面，异常订单界面，未完成正常订单界面，已完成正常订单界面，已撤销订单界面，历史预订酒店界面，注册会员界面，评价酒店界面，客户个人信息界面，酒店工作人员界面，酒店工作人员个人信息界面，酒店工作人员制订策略界面，生日折扣界面，三间及以上优惠界面，合作企业优惠界面，新建优惠策略界面，酒店信息界面，房间信息界面，处理订单界面，网站营销人员主界面，制订双十一促销策略界面，制订VIP会员特定商圈专属折扣界面界面，制订新促销策略界面，管理普通会员福利界面，管理企业会员福利界面，制订会员等级界面，管理客户信用值界面，处理异常订单界面，网站管理人员主界面，管理客户账户界面，添加酒店界面，查看修改酒店工作人员账户界面，管理网站营销人员账户界面。

界面跳转如图5.2所示。

图5.2 界面跳转图

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 |
| Mainui | 主界面Frame，负责其他业务界面Panel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**

|  |  |  |
| --- | --- | --- |
| Mainui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示Frame |
| Clientui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示客户服务的主界面 |
| Loginui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示客户登录的界面 |
| Signupui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示客户注册的界面 |
| Orderui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示OrderPanel |
| Sitemanagerui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示SitemanagerPanel |
| Hotelstaffui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示hotelstaffPanel |
| Hotelinfoui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示hotelinfoPanel |
| Marketing\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示MarketingPanel |
| Strategy\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 酒店工作人员或网站营销人员提出制定促销策略请求 |
| 后置条件 | 显示StrategyPanel |

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.loginblservice | 登陆界面的业务逻辑接口 |
| businesslogicservice.signupblservice | 注册界面的业务逻辑接口 |
| businesslogicservice.clientblservice | 处理客户相关业务的接口 |
| businesslogicservice.orderblservice | 处理与单据信息的接口 |
| businesslogicservice.sitemanager blservice | 处理与账户管理相关的接口 |
| businesslogicservice.logblservice | 记录系统重要操作的接口 |
| businesslogicservice.hotelstaffblservice | 提供酒店工作人员修改酒店工作人员信息的接口 |
| businesslogicservice.hotelinfoblservice | 提供酒店工作人员管理酒店信息，客户、网站营销人员、网站管理人员查看酒店信息的接口 |
| businesslogic.marketing\_blservice | 处理网站营销人员的操作信息的接口 |
| businessslogic.strategy\_blservice | 负责处理酒店管理人员或网站营销人员制定的促销策略的接口 |

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.LoginBLService | 登陆界面的业务逻辑接口 |
| businesslogicservice.\*BLService | 每个界面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Java的javafx来实现。



## 5.3 业务逻辑层的分解

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 |
| Clientbl | 负责用户相关的服务 |
| Loginbl | 负责验证用户登录的服务 |
| Signupbl | 负责处理与用户注册相关的服务 |
| Orderbl | 负责处理与订单相关的服务 |
| Sitemanagerbl | 提供客户账户管理、酒店信息管理、网站营销人员账户管理功能的服务 |
| Logbl | 记录系统重要操作，见需求规格说明中的log |
| Hotelstaffbl | 负责处理与酒店工作人员信息相关的服务 |
| Hotelinfobl | 负责处理与酒店信息相关的服务 |
| Marketingbl | 负责网站营销人员的操作信息相关的服务 |
| Strategybl | 负责处理与促销策略相关的服务，为其他包提供查询的接口 |

### 5.3.2 业务逻辑层模块的接口规范

#### **表5.3.2-1Loginbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Loginbl.login | 语法 | public boolean login(String user\_id,String password); | |
| 前置条件 | 启动登录服务，已知用户名和密码 | |
| 后置条件 | 验证用户名和密码，返回登录信息（登录是否成功、登录权限） | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Clientdata.checkAccount(String client\_id,String password); | | | 查找客户的账号是否存在 |
| Logbl.addLog(String logInfo) | | | 保存操作日志持久化序列——持久化序列 |
| HotelStaffbl.checkAccount(String id) | | | 查找酒店工作人员的账户是否存在 |
| Marketingbl.checkAccount(String id) | | | 查找网站营销人员的账户是否存在 |
| Sitemanagerbl.checkAccount(String id) | | | 查找网站管理人员的账户是否存在 |

#### **表5.3.2-2Clientbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Clientbl.getClient | 语法 | Public clientVO getclient (String clientID); |
| 前置条件 | 已知客户ID，需要客户的个人基本信息 |
| 后置条件 | 返回客户的基本信息 |
| Clientbl.createClient | 语法 | Public Boolean  createClient(ClientVO vo); |
| 前置条件 | 已有新建客户所需信息 |
| 后置条件 | 创建一个新用户 |
| Clientbl.setCredit | 语法 | Public Boolean  setCredit(String clientId，int recharge); |
| 前置条件 | 已知客户ID，需要修改客户的信用值信息 |
| 后置条件 | 修改并保存客户的信用值，并将修改条目计入信用记录 |
| Clientbl.setAllClientLevel | 语法 | Public Boolean  setAllClientLevel(LevelVO vo); |
| 前置条件 | 需要修改客户的会员等级信息 |
| 后置条件 | 修改并保存客户的会员等级 |
| Clientbl.getSearchHotel | 语法 | public List<hotelinfoVO> getSearchHotel（OrderRequirementVO vo） |
| 前置条件 | 已有合法（完整正确）的所需酒店范围 |
| 后置条件 | 返回符合条件的酒店的列表 |
| Clientbl.getSearchDateOrder | 语法 | Public List <OrderVO>  getSearchDateOrder（String clientID，Date date） |
| 前置条件 | 已有完整正确的所需订单时间范围 |
| 后置条件 | 返回符合条件的订单的列表 |
| Clientbl.getSearchHotelOrder | 语法 | Public List <OrderVO>  getSearchHotelOrder（String clientID，String hotelID） |
| 前置条件 | 已有完整正确的酒店名称 |
| 后置条件 | 返回客户在该酒店的订单列表 |
| Clientbl.updateInfo | 语法 | Public ResultMessage  updateInfo（ClientVO vo） |
| 前置条件 | 已有合法（完整正确）的个人信息修改条目 |
| 后置条件 | 返回是否成功修改的提示信息 |
| Clientbl.repealOrder | 语法 | Public Boolean repealOrder（String orderID） |
| 前置条件 | 客户已经确认要撤销订单 |
| 后置条件 | 撤销订单并修改订单状态 |
| Clientbl.getOrderList | 语法 | Public List<orderVO>  getOrderList(orderTpe order\_type) |
| 前置条件 | 已选择订单列表的类型 |
| 后置条件 | 显示相应类型的订单列表 |
| Clientbl.getCreditRecord | 语法 | Public File  getCreditRecord(String client ID) |
| 前置条件 | 已知客户ID，需要查看信用记录 |
| 后置条件 | 显示信用记录列表 |
| Clientbl.getHistoryList | 语法 | Public List<HotelInfoVO>  getHistoryList(String client ID) |
| 前置条件 | 需要查看历史预订酒店记录 |
| 后置条件 | 显是历史预订酒店列表 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Hotelinfobl.getBasicInfo(String hotelID) | | 查找并显示酒店列表 |
| Clientdata.findOrderList | | 查找客户某一类型的订单列表 |
| Clientdata.findHistoryList(String clientID) | | 查找客户历史酒店列表 |
| Clientdata.findCreditRecord(String clientID) | | 查找客户信用记录 |
| Clientdata.findPersonalInfo(String clientID) | | 查找客户个人基本信息 |
| Clientdata.modifyPersonalInfo(String clientID,clientPO po) | | 修改客户的个人基本信息 |
| Clientdata.createClient(clientPO po) | | 创建新客户 |
| Orderbl.createOrder(orderVO order) | | 新建一个订单 |
| Orderbl.findSpecificClinetOrder(String clientID) | | 查找某一个时期的订单 |
| Orderbl.findSpecificHotelClientOrder(String clientID,String hotelID) | | 查找客户在某一个酒店的订单 |
| Orderbl.findCUserOrderList(String clientID) | | 查找客户所有订单 |
| Orderbl.findClienTypeOrder(OrderType type,String clientID) | | 查找客某一类型订单 |
| Orderbl.cancelOrder（String order ID） | | 撤销客户的订单 |

#### **表5.3.2-3Orderbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Orderbl.  createOrder | 语法 | public ResultMessage createOrder (OrderVO orderInfo) |
| 前置条件 | 订单信息符合规则且齐全 |
| 后置条件 | 系统创建该订单 |
| Orderbl.  saveOrder | 语法 | public ResultMessage saveOrder (OrderVO orderInfo) |
| 前置条件 | 已知订单信息，需要保存该订单 |
| 后置条件 | 系统保存该订单 |
| Orderbl.  cancelOrder | 语法 | public ResultMessage cancelOrder (String orderId) |
| 前置条件 | 已知订单号，需要取消该订单 |
| 后置条件 | 订单置为取消状态 |
| Orderbl.  findSpecificOrderList | 语法 | public OrderVO findSpecificOrderList(String orderID) |
| 前置条件 | 已知订单号 |
| 后置条件 | 返回指定的订单 |
| Orderbl.  findUserOrderList | 语法 | public List<OrderVO> findUserOrderList (String userID) |
| 前置条件 | 已知用户种类 |
| 后置条件 | 返回该用户的所有订单 |
| Orderbl.  findSpecificDayClientOrder | 语法 | public List<OrderVO> findSpecificHotelClientOrder (String clientId, String hoteIId) |
| 前置条件 | 已知客户帐号，酒店帐号 |
| 后置条件 | 返回客户在某个酒店的所有订单 |
| Orderbl. findSpecificDayClientOrder | 语法 | public List<OrderVO> findSpecificDayClientOrder(String clientId, Date date) |
| 前置条件 | 已知请求查找的客户的帐号，要查找的日期 |
| 后置条件 | 返回客户该天的订单 |
| Orderbl.  findClientTypeOrder | 语法 | public List<OrderVO> findClientTypeOrder (OrderType type,String clientId) |
| 前置条件 | 已知请求查找的订单种类和客户帐号 |
| 后置条件 | 返回客户该种类（正常未执行、正常已执行、取消、异常）的订单 |
| Orderbl.  findSpecificHotelOrder | 语法 | Public OrderVO findSpecificHotelOrder  (String hotelId, String orderId) |
| 前置条件 | 已知请求查找酒店某一订单的酒店帐号和订单帐号 |
| 后置条件 | 返回酒店某个具体订单 |
| Orderbl.  findAbnormalOrderList | 语法 | Public List<OrderVO> findAbnormalOrderList  (Date date ) |
| 前置条件 | 已知请求查找异常订单的时间 |
| 后置条件 | 返回异常订单列表 |
| Orderbl.  checkTime | 语法 | Public boolean checkTime (OrderVO order, Date date) |
| 前置条件 | 晚上12点检查订单执行情况 |
| 后置条件 | 订单已修改的状态持久化保存 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Orderdata.add(OrderPO order) | | 新增订单一持久化序列 |
| Orderdata.save(OrderPO order) | | 保存订单一持久化序列 |
| Orderdata.cancel(String orderId) | | 取消订单－持久化序列 |
| Orderdata. findUserOrderList(String userId) | | 查找某种类用户的某种类订单一持久化序列 |
| Orderdata. findUserOrderList(String userId) | | 查找某用户的所有订单一持久化序列 |
| Orderdata. findClientTypeOrder (OrderType orderType,String clientId) | | 查找某客户的某个种类（正常未执行、正常已执行、取消、异常）的订单一持久化序列 |
| Orderdata.findAbnormalOrderList(Date date) | | 网站营销人员查找当天异常订单一持久化序列 |
| Order.checkTime(String orderId) | | 修改订单状态－持久化序列 |
| Logbl.addLog(String logInfo) | | 保存操作日志－持久化序列 |

#### **表5.3.2-4Sitemanagerbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Sitemanagerbl.  clientAccountUpdate | 语法 | public ResultMessage clientAccountUpdate (ClientVO clientVO) | |
| 前置条件 | 请求修改某客户账户 | |
| 后置条件 | 如果对应clientId存在，系统修改并保存该订单，返回修改成功的消息；否则，返回修改失败的消息 | |
| Sitemanagerbl.  accountDelete | 语法 | public ResultMessage clientAccountDelete (String userId) | |
| 前置条件 | 请求删除某用户账户 | |
| 后置条件 | 如果对应userId存在，系统删除该账户，返回删除成功的消息；否则，返回删除失败的消息 | |
| Sitemanagerbl.  clientAccountFind | 语法 | Public clientVO clientAccountFind (String findInfo) | |
| 前置条件 | 请求查找某客户账户 | |
| 后置条件 | 如果findInfo（clientId或phoneNumber）对账户存在，返回该客户账户，否则，返回null | |
| Sitemanagerbl.  MarketingAccountAdd | 语法 | public ResultMessage MarketingAccountAdd (MarketingVO marketing) | |
| 前置条件 | 请求增加某网站营销人员账户 | |
| 后置条件 | 如果系统中没有相同电话的网站营销人员账户，增加了一个网站营销人员账户，返回添加成功的消息，否则，返回添加失败的消息 | |
| Sitemanagerbl.  MarketingAccountUpdate | 语法 | public resultMessage MarketingAccountUpdate ( MarketingVO marketing) | |
| 前置条件 | 请求修改网站营销人员账户 | |
| 后置条件 | 如果系统中MarketingVO 里的marketingId存在，系统修改并保存账户，返回修改成功的消息，否则，返回修改失败的消息 | |
| Sitemanagerbl.  MarketingAccountFind | 语法 | public marketingVO MarketingAccountFind (String findInfo) | |
| 前置条件 | 请求查找某营销人员账户 | |
| 后置条件 | 如果findInfo（marketingId或phoneNumber）对应的账户存在，系统返回该账户信息；否则，返回null | |
| Sitemanagerbl.  HotelAccountAdd | 语法 | public ResultMessage MarketingAccountAdd (HotelInfoVO hotelinfo) | |
| 前置条件 | 请求添加某酒店 | |
| 后置条件 | 如果系统不存在同名酒店，则添加并保存该酒店，返回添加成功的消息，否则，返回添加失败的消息 | |
| Sitemanagerbl.  HotelStaffAccountFind | 语法 | public HotelStaffVO HotelStaffAccountFind  (String hotelId) | |
| 前置条件 | 请求查找某酒店工作人员账户 | |
| 后置条件 | 如果系统中存在hotelId对应的账户，返回该账户信息，否则，返回null | |
| Sitemanagerbl.  HotelStaffAccountUpdate | 语法 | public ResultMessage HotelStaffAccountUpdate  (HotelstaffVO hotelstaffVO) | |
| 前置条件 | 请求修改某酒店工作人员账户 | |
| 后置条件 | 如果系统中存在hotelstaffVO 里hotelId对应的账户，修改并保存该账户信息，返回修改成功的消息，否则，返回修改失败的消息 | |
| Sitemanagerbl.checkAccount | 语法 | public boolean checkAccount  (SitemanagerVO accountVO) | |
| 前置条件 | 请求检查酒店工作人员登陆信息是否正确 | |
| 后置条件 | 如果帐号密码正确，则返回true，否则，返回false | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Marketing.saveSitemanagerAdd | | | 网站营销人员保存网站管理人员新增的网站营销人员账户信息一持久化序列 |
| Marketing.saveSitemanagerUpdate | | | 网站营销人员保存网站管理人员修改的网站营销人员账户信息一持久化序列 |
| Marketing.saveSitemanagerDelete | | | 网站营销人员保存网站管理人员删除的网站营销人员账户信息一持久化序列 |
| Marketing.returnSitemanagerAccount | | | 网站营销人员查找网站管理人员请求提供的网站营销人员账户信息一持久化序列 |
| Client.saveSitemanagerUpdate | | | 客户保存网站管理人员修改的网站营销人员账户信息一持久化序列 |
| Client.saveSitemanagerDelete | | | 客户保存网站管理人员删除的网站营销人员账户信息一持久化序列 |
| Client.returnSitemanagerAccount | | | 网站营销人员查找网站管理人员请求提供的客户账户信息一持久化序列 |
| HotelStaff.saveSitemanagerUpdate | | | 酒店管理人员保存网站管理人员修改的酒店工作人员账户信息一持久化序列 |
| HotelStaff.returnSitemanagereAccount | | | 酒店管理人员查找网站管理人员请求提供的酒店管理人员账户信息一持久化序列 |
| HotelInfo.saveSitemanagerAdd | | | 酒店保存网站管理人员新增的就酒店账户信息一持久化序列 |
| Logbl.addLog(String logInfo) | | | 保存操作日志－持久化序列 |

#### **表5.3.2-5Logbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Logbl.getLog | 语法 | Public List<LogVO> getLog() |
| 前置条件 | 查看日志功能 |
| 后置条件 | 返回日志列表 |
| Logbl.addLog |  | Public resultMessage addLog (String logInfo) |
| 前置条件 | 已知请求增加一条日志的日志信息 |
| 后置条件 | 系统增加该日志 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Logdata.getLogList | | 得到log日志的所有记录PO |
| Logdata.addLog(LogPO) | | 增加一条LogPO |

#### **表5.3.2-6HotelStaffbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| hotelstaffbl.getBasicInfo | 语法 | Public HotelstaffBasicinfoVO getBasicInfo (String hotelID) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求获得自己的信息 |
| 后置条件 | 返回ID对应的酒店工作人员的信息 |
| hotelstaffbl.setPassword | 语法 | Public ResultMessage setPassword（String hotelID,String password) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求将新密码持久化保存 |
| 后置条件 | 持久化保存新密码 |
| Hotelstaffbl.updatehotelStrategy | 语法 | Public ResultMessage updatehotelstrategy (HotelStrategyVO vo) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求将酒店的促销策略信息持久化保存 |
| 后置条件 | 持久化保存ID对应的酒店的策略信息 |
| Hotelstaffbl.gethotelStrategy | 语法 | Public HotelStrategyVO gethotelStrategy(String hotelID) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求获得酒店的促销策略信息 |
| 后置条件 | 返回ID对应的酒店的策略信息 |
| Hotelsatffbl.gethotelinfo | 语法 | Public HotelinfoVO gethotelInfo(String hotelID) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求获得酒店基本信息 |
| 后置条件 | 返回hotelID对应的酒店信息 |
| Hotelstaffbl. updatehotelinfo | 语法 | Public ResultMessage updatehotelinfo(HotelinfoVO vo) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求将酒店基本信息持久化保存 |
| 后置条件 | 持久化保存ID对应的酒店信息 |
| Hotelstaffbl.getroominfo | 语法 | Public RoominfoVO getroominfo(String hotelID,String roomID) |
| 前置条件 | 酒店工作人员或客户身份被识别和授权，请求更新房间基本信息 |
| 后置条件 | 返回roomID对应的房间信息 |
| Hotelstaffbl.updateroominfo | 语法 | Public ResultMessage updateroominfo(HotelinfoVO vo) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求将房间基本信息持久化保存 |
| 后置条件 | 持久化保存房间信息 |
| Hotelstaffbl.gethotelOrderList | 语法 | Public List<HotelOrderVO> gethotelOrderList(String hotelID) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求获得hotelID所拥有的全部订单 |
| 后置条件 | 持久化保存roomID对应的房间信息 |
| Hotelstaffbl.updateOrderState | 语法 | Public ResultMessage updateOrderState(String orderID,OrderType orderType) |
| 前置条件 | 酒店工作人员身份被识别和授权，请求将订单状态信息持久化保存 |
| 后置条件 | 持久化保存订单状态信息信息 |
| HotelStaffbl.saveSitemanagerUpdate | 语法 | Public boolean saveSitemanagerUpdate  (HotelStaffVO vo) |
| 前置条件 | 网站管理人员身份被识别和授权，请求将对酒店工作人员信息（电话）持久化保存 |
| 后置条件 | 持久化保存订单状态酒店工作人员信息 |
| Hotelstaffbl. returnSitemanagereAccount | 语法 | Public HotelstaffVO returnSitemanagerAccount(String hotelID) |
| 前置条件 | 网站管理人员身份被识别和授权，请求获得酒店工作人员信息 |
| 后置条件 | 返回酒店工作人员信息 |
| Hotelstaffbl. checkAccount | 语法 | Public boolean checkAccount(String hotelID,String password) |
| 前置条件 | 登录界面请求检查登录名和密码 |
| 后置条件 | 查找持久化信息 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Hotelstaffdata.findBasicInfo(String hotelStaffID) | | 查找单一持久化序列 |
| Hotelstaffdata.updatePassword  (HotelstaffPO po) | | 更新单一持久化序列 |
| Strategybl.gethotelStrategy  (String hotelID) | | 请求strategybl查找酒店工作人员制定的策略信息 |
| Strategybl.getmarketingStrategy() | | 请求strategybl查找网站管理人员制定的策略信息 |
| Strategybl.updatehotelStrategy  (HotelStrategyVO vo) | | 请求strategybl更新酒店策略信息 |
| Strategybl.deletehotelStrategy（HotelStrategyVO vo） | | 请求strategybl删除策略信息 |
| Hotelinfobl.getBasicinfo(String hotleID) | | 请求hotelinfobl查找酒店基本信息 |
| Hotelinfobl.updateBasicinfo(hotelinfoVO) | | 请求hotelinfobl更新酒店基本信息 |
| Hotelinfobl.getroominfo(Stirng roomID,String htoelID) | | 请求hotelinfobl查找房间信息 |
| Hotelinfobl.updateroominfo(RoominfoVO vo) | | 请求hotelinfo更新房间信息 |
| Orderbl.findUserOrderList(String userid) | | 请求orderbl查找酒店的所有订单列表 |
| Orderbl.findSpecificHotelOrder(String hotelD,String orderID) | | 请求orderbl查找某个订单信息 |
| Orderbl.saveOrder(OrderVO order) | | 请求orederbl更新订单状态 |

#### **表5.3.2-7Hotelinfobl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| hotelinfobl.getBasicInfo | 语法 | Public HotelinfoVO getBasicInfo (String hotelID) | |
| 前置条件 | 酒店工作人员或客户或网站营销人员的身份已经被验证和授权，请求需要获得已知hotelID对应的酒店信息 | |
| 后置条件 | 返回要求的酒店信息 | |
| hotelinfobl.getBasicInfoList | 语法 | Public List<HotelinfoVO> getBasicInfoList (ClientRequirementVO vo) | |
| 前置条件 | 客户的身份已经被验证和授权，请求获得符合客户条件的酒店信息列表 | |
| 后置条件 | 返回符合客户条件酒店信息列表 | |
| hotelinfobl.updateBasicInfo | 语法 | Public ResultMessage updateBasicInfo(HotelInfoVO vo) | |
| 前置条件 | 酒店工作人员确认酒店信息修改完毕 | |
| 后置条件 | 持久化保存酒店信息 | |
| hotelinfobl.getRoominfo | 语法 | Public RoominfoVO getRoominfo(String hotelID, String roomtype) | |
| 前置条件 | 请求获得某房间类型对应的房间信息 | |
| 后置条件 | 返回某类型的房间信息 | |
| hotelinfobl.getRoominfoList | 语法 | Public List<RoominfoVO> getRoominfoList(String hotelID) | |
| 前置条件 | 请求获得某酒店的酒店信息列表 | |
| 后置条件 | 返回某酒店的所有房间信息列表 | |
| hotelinfobl.getfinalprice | 语法 | Public double calculatePrice(List<StartegyVO> list，ClientVO vo，double originalPrice) | |
| 前置条件 | 已知策略VO,订单VO,请求获得使用促销策略后的房间价格 | |
| 后置条件 | 返回使用完策略后的价格信息 | |
| hotelinfobl.saveSitemanagerAdd | 语法 | Public Boolean saveSitemanagerAdd (HotelinfoVO vo) | |
| 前置条件 | 网站管理人员的身份已被验证和识别，请求插入单一持久化序列 | |
| 后置条件 | 持久化保存酒店信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Hotelinfodata.findRoomInfo（String roomtype） | | | 查找单一持久化序列 |
| Hotelinfodata.findRoomInfoList（String hotelID） | | | 查找单一持久化序列 |
| Hotelinfodata.findhotelInfoList(String hotelID) | | | 查找单一持久化序列 |
| Hotelinfodata.findhotelInfo(String hotelID) | | | 查找单一持久化序列 |
| Hotelinfodata.updatehotelInfo(HotelInfoPO po) | | | 更新单一持久化序列 |
| Hotelinfodata.updateroomInfo(RoomInfoPO po) | | | 更新单一持久化序列 |
| Hotelinfodata.insert(RoomInfoPO po) | | | 插入单一持久化序列 |
| Hotelinfodata.delete(RoomInfoPO po) | | | 删除单一持久化序列 |
| Hotelinfodata.addlog(String loginfo) | | | 保存操作日志-持久化序列 |
| Clientbl.createOrder(String message) | | | 获得客户确认创建订单信息 |

#### **表5.3.2-8Marketingbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Marketingbl.addMarketingStrategy | 语法 | Public ResultMessage addMarketingStrategy(marketingStrategyVO vo) |
| 前置条件 | 已知网站营销人员的促销策略,需要将促销信息持久化保存 |
| 后置条件 | 持久化保存促销信息 |
| Marketingbl.getmarketingStrategy | 语法 | Public List<marketingStrategyVO> getmarketingStrategy(String id) |
| 前置条件 | 需要获得已存在的制定策略 |
| 后置条件 | 返回已存在的制定策略 |
| Marketingbl.deleteMarketingStrategy | 语法 | Public ResultMessage deleteMarketingStrategy(marketingStrategyVO vo) |
| 前置条件 | 需要删除制定策略 |
| 后置条件 | 持久化删除制定策略 |
| Marketingbl.addLevel | 语法 | Public ResultMessage addLevel(LevelVO vo) |
| 前置条件 | 已知网站营销人员设定的会员等级，需将等级信息保存 |
| 后置条件 | 持久化保存等级信息 |
| Marketingbl.findAllLevel | 语法 | Public List<LevelVO> findAllLevel() |
| 前置条件 | 需要获得已存在的等级信息 |
| 后置条件 | 返回已存在的等级信息 |
| Marketingbl.deleteLevel | 语法 | Public ResultMessage deleteLevel(LevelVO vo) |
| 前置条件 | 需要删除已存在的等级信息 |
| 后置条件 | 持久化删除已存在的等级信息 |
| Marketingbl.addPrivilege | 语法 | Public ResultMessage addPrivilege(PrivilegeVO vo) |
| 前置条件 | 已知营销人员的会员优惠策略，需将优惠信息持久化保存 |
| 后置条件 | 将优惠信息持久化保存 |
| Marketingbl.findAllPrivilege | 语法 | Public List<PrivilegeVO> findAllPrivilege(String VIPType) |
| 前置条件 | 需要获得已存在的会员优惠信息 |
| 后置条件 | 返回优惠策略列表 |
| Marketingbl.deletePrivilege | 语法 | Public ResultMessage deletePrivilege(PrivilegeVO vo) |
| 前置条件 | 需要删除已存在的会员优惠信息 |
| 后置条件 | 持久化删除会员优惠信息 |
| Marketingbl. findAbnormalOrderList | 语法 | Public List<OrderVO> showAbnormal(Date date) |
| 前置条件 | 需要获得当日异常订单列表 |
| 后置条件 | 返回当日异常订单列表 |
| Marketingbl.findSpecificAbnormalOrder | 语法 | Public OrderVO findSpecificAbnormalOrder(String id) |
| 前置条件 | 需要获得某个异常订单的信息 |
| 后置条件 | 返回该异常订单信息 |
| Marketingbl.savaAbnormalOrder | 语法 | Public ResultMessage handleAbnormal(OrderVO vo) |
| 前置条件 | 需要改变订单状态以及客户信用值 |
| 后置条件 | 持久化保存订单状态和客户信用值，建立订单状态修改记录 |
| Marketingbl.getClient | 语法 | Public ClientVO getClient(String clientID) |
| 前置条件 | 需要查看客户的信息 |
| 后置条件 | 返回客户详细信息 |
| Marketingbl.setCredit | 语法 | Public ResultMessage setCredit(String clientID,int recharge) |
| 前置条件 | 需要修改客户信用值 |
| 后置条件 | 持久化保存客户信用值的修改并保存修改记录 |
| Marketingbl.update | 语法 | Public ResultMessage update(MarketingVO vo); |
| 前置条件 | 网站营销人员需要修改自己的信息 |
| 后置条件 | 持久化修改该网站营销人员的信息 |
| Marketingbl.siteManagerUpdate | 语法 | Public boolean siteManagerUpdate(MarketingVO vo); |
| 前置条件 | 网站管理人员需要修改网站营销人员的信息 |
| 后置条件 | 持久化保存所需要修改的信息 |
| Marketingbl.saveSitemanagerAdd | 语法 | Public boolean saveSitemanagerAdd(MarketingVO vo) |
| 前置条件 | 网站管理人员需要增加网站营销人员及其信息 |
| 后置条件 | 持久化保存该网站营销人员的信息 |
| Marketingbl.saveSitemanagerDelete | 语法 | Public Boolean saveSitemanagerDelete(MarketingVO vo) |
| 前置条件 | 网站管理人员需注销某网站营销人员的账号 |
| 后置条件 | 该网站营销人员的账号被注销 |
| Marketingbl.return SitemanagerAccoutn | 语法 | Public MarketingVO returnSitemanagerAccount(String marketingID) |
| 前置条件 | 网站管理人员提供已存在的网站营销人员账号 |
| 后置条件 | 返回账号信息 |
| Marketingbl.checkAccount | 语法 | Public Boolean checkAccount (String marketingID,String password) |
| 前置条件 | 网站营销人员请求登录 |
| 后置条件 | 返回check账号与password的结果 |
| Marketingbl.init | 语法 | Public MarketingVO init(String id) |
| 前置条件 | 网站营销人员账户验证成功 |
| 后置条件 | 返回网站营销人员个人信息 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Marketingdata.addLevel(LevelPO po) | | 生成持久化等级信息对象 |
| Marketingdata.findAllLevel() | | 返回等级持久化对象列表 |
| Marketingdata.deleteLevel(LevelPO po) | | 删除持久化等级信息对象 |
| Marketingdata.updateAndAdd(MarketingPO po) | | 修改网站营销人员序列化信息 |
| Marketingdata.delete(MarketingPO po) | | 删除网站营销人员账号 |
| Marketingdata.getInfo(String id) | | 返回网站营销人员个人信息 |
| Order.findAbnormalOrderList(Date date) | | 返回异常订单对象列表 |
| Order.findSpecificAbnormalOrder(String orderID) | | 返回具体异常订单对象 |
| Order.savaAbnormalOrder(OrderPO order) | | 设置异常订单对象状态 |
| Client.getClient(String clientID) | | 返回客户详细序列化信息 |
| Client.setAllClientLevel(LevelPO po) | | 修改所有客户的会员等级与种类信息 |
| Client.setCredit(String clientID,int recharge) | | 修改客户信用值 |
| Strategy.addMarketingStrategy(marketingStrategyPO po) | | 生成促销策略持久化对象 |
| Strategy.getmarketingStrategy(String id) | | 返回单一所有网站营销人员的制定的促销策略持久化对象 |
| Strategy.deleteMarketingStrategy(marketingStrategyPO po) | | 删除单一持久化促销策略对象 |
| Strategy.addPrivilege(PrivilegePO po) | | 生成持久化优惠信息对象 |
| Strategy.findAllPrivilege(String VIPType) | | 返回优惠信息持久化对象 |
| Strategy.deletePrivilege(PrivilegePO po) | | 删除持久化优惠信息对象 |
| Log.addLog(String logInfo) | | 生成持久化日志记录信息 |

#### **表5.3.2-9Strategybl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Strategybl.addMarketingStrategy | 语法 | Public boolean addMarketingStrategy(marketingStrategyVO vo) |
| 前置条件 | 已知网站营销人员的促销策略,需要将促销信息持久化保存 |
| 后置条件 | 持久化保存促销信息 |
| Strategybl.getmarketingStrategy | 语法 | Public List<marketingStrategyVO> getmarketingStrategy(String id) |
| 前置条件 | 根据账户id需要获得已存在的制定策略 |
| 后置条件 | 返回已存在的制定策略 |
| Strategybl.deleteMarketingStrategy | 语法 | Public boolean deleteMarketingStrategy(marketingStrategyVO vo) |
| 前置条件 | 需要删除制定策略 |
| 后置条件 | 持久化删除制定策略 |
| Strategybl.gethotelStrategy | 语法 | Public List<hotelStrategyVO> getHotelStrategy(String hotelID) |
| 前置条件 | 酒店管理人员需要获得策略 |
| 后置条件 | 返回酒店促销策略 |
| Strategybl.updatehotelStrategy | 语法 | Public Boolean  updatehotelStrategy(hotelStrategyVO vo) |
| 前置条件 | 酒店人员需要更新酒店策略 |
| 后置条件 | 更新策略 |
| Strategybl.addPrivilege | 语法 | Public boolean addPrivilege(PrivilegeVO vo) |
| 前置条件 | 已知营销人员的会员优惠策略，需将优惠信息持久化保存 |
| 后置条件 | 将优惠信息持久化保存 |
| Strategybl.findAllPrivilege | 语法 | Public List<PrivilegeVO> findAllPrivilege(String VIPType) |
| 前置条件 | 需要获得已存在的会员优惠信息 |
| 后置条件 | 返回优惠策略列表 |
| Strategybl.deletePrivilege | 语法 | Public boolean delete(PrivilegeVO vo) |
| 前置条件 | 需要删除单一已存在的会员优惠信息 |
| 后置条件 | 持久化删除会员优惠单一信息 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Strategydata.addMarketingStrategy(StrategyPO po) | | 生成单一持久化对象 |
| Strategydata.getmarketingStrategy(String id) | | 返回单一持久化对象 |
| Strategydata.deleteMarketingStrategy(StrategyPO po) | | 删除单一持久化对象 |
| Strategydata.gethotelStrategy() | | 返回促销策略 |
| Strategydata.updatehotelStrategy(HotelStrategyPO po) | | 更新单一持久化对象 |
| Strategydata.addPrivilege(PrivilegePO po) | | 生成单一持久化对象 |
| Strategydata.findAllPrivilege(String VIPType) | | 返回已存在的会员优惠信息单一持久化对象 |
| Strategydata.deletePrivilege(PrivilegePO po) | | 删除单一持久化对象 |

## 5.4 数据层的分解

数据层主要给业务逻辑层提供数据防伪服务，包括对于持久化数据的增、删、改、查。Client业务逻辑需要的服务由ClientDataService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 |
| DataService | 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。 |
| DataServiceTxtImpl | 基于TXT文件的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 |
| DataServiceDatabaseImpl | 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Clientdata模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Clientdata.findCreditRecord | 语法 | Public File  findCreditRecord(String clientID); |
| 前置条件 | 查询客户的信用记录 |
| 后置条件 | 返回客户的信用记录 |
| Clientdata.setCredit | 语法 | Public Boolean  setCredit(String clientID，int recharge); |
| 前置条件 | 已知客户ID，需要修改客户的信用值信息 |
| 后置条件 | 修改并保存客户的信用值 |
| Clientdata.findPersonalInfo | 语法 | Public clientPO  findPersonalInfo (String clientID); |
| 前置条件 | 查询客户的个人基本信息 |
| 后置条件 | 返回客户的个人基本信息 |
| Clientdata.modifyPersonalInfo | 语法 | Public ResultMessage  modifyPersonalInfo (String clientID,clientInfoVO); |
| 前置条件 | 已知完整正确的待更新个人信息 |
| 后置条件 | 修改客户的个人基本信息并返回是否修改成功的信息 |
| Clientdata.checkAccount | 语法 | Public Boolean  checkAccount(String client\_id,String password); |
| 前置条件 | 查询客户账户是否存在 |
| 后置条件 | 返回客户账户是否存在的boolean值 |
| Clientdata.setAllLevel | 语法 | Public boolean  setAllLevel(levelPO po); |
| 前置条件 | 已知要更新的会员等级策略 |
| 后置条件 | 按照需更新的会员等级制度更新所有客户的会员等级 |
| Clientdata.createClient | 语法 | Public Boolean createClient（ClientPO po） |
| 前置条件 | 已知新建客户需要的信息 |
| 后置条件 | 新建一个客户 |

#### **表5.4.2-2数据层Orderdata模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Orderdata.add | 语法 | Public boolean add(OrderPO order) |
| 前置条件 | 与order的id相同的order在数据库中不存在 |
| 后置条件 | 数据库中添加一条order记录 |
| Orderdata.save | 语法 | Public boolean save(OrderPO order) |
| 前置条件 | 新增了订单，要对订单进行持久化保存 |
| 后置条件 | 数据库增加order记录 |
| Orderdata.cancel | 语法 | Public boolean cancel(String orderId) |
| 前置条件 | 要求取消该订单，要对取消结果进行持久化保存 |
| 后置条件 | 如果对应orderId存在，则在数据库中将该订单置为取消状态，否则，返回false |
| Orderdata.  findSpecificUserOrder | 语法 | Public OrderPO findSpecificUserOrder(UserType userType, String orderId) |
| 前置条件 | 逻辑层请求获得么某用户的某一具体订单 |
| 后置条件 | 返回对应orderId的OrderPO，如果orderId不存在，返回null |
| Orderdata.  findUserOrderList | 语法 | public List<OrderPO> findUserOrderList(String userId) |
| 前置条件 | 逻辑层请求获得某用户的所有订单列表 |
| 后置条件 | 返回对应userId的历史订单列表List<OrderPO>，如果userId不存在，返回null |
| Orderdata.find findClientTypeOrder | 语法 | public List<OrderPO> findClientTypeOrder (OrderType orderType,String clientId) |
| 前置条件 | 逻辑层请求获得某客户的某一种类订单 |
| 后置条件 | 返回对应clientId的符合OrderType的订单列表List<OrderPO>，如果clientId不存在，返回null |
| Orderdata.  findAbnormalOrderList | 语法 | Public List<OrderPO> findAbnormalOrderList(Date date) |
| 前置条件 | 逻辑层请求获得某天的所有异常订单 |
| 后置条件 | 返回对应date的异常订单列表List<OrderPO>，如果date不在数据条件范畴内，返回null |
| Orderdata.  checkTime | 语法 | Public void checkTime (String orderId) |
| 前置条件 | 逻辑层请求将超时订单置为异常 |
| 后置条件 | 如果超过晚上12点，订单仍未执行，则将订单置为异常状态，否则，不进行操作 |
| 提供的服务（供接口） | | |
| Orderdata.add | 语法 | Public boolean add(OrderPO order) |
| 前置条件 | 与order的id相同的order在数据库中不存在 |
| 后置条件 | 数据库中添加一条order记录 |
| Orderdata.save | 语法 | Public boolean save(OrderPO order) |
| 前置条件 | 新增了订单，要对订单进行持久化保存 |
| 后置条件 | 数据库增加order记录 |
| Orderdata.cancel | 语法 | Public boolean cancel(String orderId) |
| 前置条件 | 要求取消该订单，要对取消结果进行持久化保存 |
| 后置条件 | 如果对应orderId存在，则在数据库中将该订单置为取消状态，否则，返回false |
| Orderdata.  findSpecificUderOrder | 语法 | Public OrderPO findSpecificUserOrder(UserType userType, String orderId) |
| 前置条件 | 无 |
| 后置条件 | 返回对应orderId的OrderPO，如果orderId不存在，返回null |
| Orderdata.  findUserOrderList | 语法 | public List<OrderPO> findUserOrderList(String userId) |
| 前置条件 | 无 |
| 后置条件 | 返回对应userId的历史订单列表List<OrderPO>，如果userId不存在，返回null |
| Orderdata.find findClientTypeOrder | 语法 | public List<OrderPO> findClientTypeOrder (OrderType orderType,String clientId) |
| 前置条件 | 无 |
| 后置条件 | 返回对应clientId的符合OrderType的订单列表List<OrderPO>，如果clientId不存在，返回null |
| Orderdata.  findAbnormalOrderList | 语法 | Public List<OrderPO> findAbnormalOrderList(Date date) |
| 前置条件 | 无 |
| 后置条件 | 返回对应date的异常订单列表List<OrderPO>，如果date不在数据条件范畴内，返回null |
| Orderdata.  checkTime | 语法 | Public void checkTime (String orderId) |
| 前置条件 | 无 |
| 后置条件 | 如果超过晚上12点，订单仍未执行，则将订单置为异常状态，否则，不进行操作 |

#### **表5.4.2-3数据层Logdata模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| LogData.getLogList | 语法 | Public List<LogPO>getLogList() |
| 前置条件 | 逻辑层请求获得log记录 |
| 后置条件 | 返回log列表 |
| LogData.addLog | 语法 | Public boolean addLog (LogPO) |
| 前置条件 | 逻辑层请求增加一条log记录 |
| 后置条件 | 在数据库中增加该log记录 |

#### **表5.4.2-4数据层Sitemanagerdata模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Sitemanager.getAccount | 语法 | public SitemanagerPo getAccount() |
| 前置条件 | 逻辑层请求获得网站管理人员账户记录 |
| 后置条件 | 返回网站管理人员账户记录 |
| SitemanagerAccountUpdate(SitemanagerPO sitemanagerPO) | 语法 | public boolean SitemanagerAccountUpdate(SitemanagerPO sitemanagerPO) |
| 前置条件 | 逻辑层修改数据库中网站管理人员账户信息 |
| 后置条件 | 在数据层中修改网站管理人员账户信息 |

#### **表5.4.2-5数据层HotelStaffdata模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotelstaffdata.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 |
| Hotelstaffdata.findBasicInfo | 语法 | Public HotelstaffPO findBasicInfo(String hotelStaffID) |
| 前置条件 | 无 |
| 后置条件 | 按ID返回酒店工作人员信息 |
| Hotelstaffdata.updatePassword | 语法 | Public boolean updatePassword  (HotelstaffPO po) throws RemoteException |
| 前置条件 | 数据库中存在一个同样ID的PO |
| 后置条件 | 更新一个PO |
| Hotelstaffdata.checkAccount | 语法 | Public boolean checkAccount(hotelstaffPO po) throws RemoteException |
| 前置条件 | 数据库中存在一个同样ID的PO |
| 后置条件 | 返回酒店工作人员账户密码是否存在 |
| Hotelstaffdata.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 |

#### **表5.4.2-6数据层Hotelinfodata模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotelinfodata.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 |
| hotelinfodata.getRoomInfo | 语法 | Public RoomInfoPO getRoomInfo（String roomtype） |
| 前置条件 | 无 |
| 后置条件 | 按roomtype返回酒店信息 |
| hotelinfodata.getRoomInfoList | 语法 | Public List<RoomInfoPO> getRoomInfoList（String hotelID） |
| 前置条件 | 无 |
| 后置条件 | 按hotelID返回酒店信息列表 |
| hotelinfodata.findhotelInfoList | 语法 | Public HotelInfoPO findhotelInfoList(String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 按hotelID返回酒店信息列表 |
| hotelinfodata.findhotelInfo | 语法 | Public HotelInfoPO findhotelInfo(String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 按hotelID返回酒店信息 |
| hotelinfodata.updatehotelInfo | 语法 | Public boolean updatehotelInfo(HotelInfoPO po) |
|  | 前置条件 | 数据库里存在一个同样ID的PO |
| 后置条件 | 更新一个酒店信息PO |
| hotelinfodata.updateroomInfo | 语法 | Public boolean updateroomInfo(RoomInfoPO po) |
| 前置条件 | 数据库里存在一个同样ID的PO |
| 后置条件 | 更新一个房间信息PO |
| hotelinfodata.insertroomInfo | 语法 | Public boolean insert(RoomInfoPO po) |
| 前置条件 | 同样ID的PO在数据库中不存在 |
| 后置条件 | 增加一个房间信息PO |
| hotelinfodata.deleteroomInfo | 语法 | Public boolean delete(RoomInfoPO po) |
| 前置条件 | 数据库里存在一个同样ID的PO |
| 后置条件 | 删除一个房间信息PO |
| Hotelinfodata.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 |

#### **表5.4.2-7数据层Marketingdata模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Marketingdata.addLevel | 语法 | Public boolean addLevle(LevelPO po) throws RemoteException; |
| 前置条件 | 有新增会员等级需持久化保存 |
| 后置条件 | 在数据库中保存该等级持久化对象 |
| Marketingdata.findAllLevel | 语法 | Public List<LevelPO> findAllLevel()throws RemoteException; |
| 前置条件 | 需要获取所有等级信息 |
| 后置条件 | 从数据库中返回所有等级信息持久化对象列表 |
| Marketingdata.deleteLevel | 语法 | Public Boolean deleteLevel(LevelPO po) throws RemoteException; |
| 前置条件 | 需删除某等级持久化对象 |
| 后置条件 | 在数据库中删除该等级持久化对象信息 |
| Marketingdata.getInfo | 语法 | Public MarketingPO getInfo(String id) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 返回网站营销人员基本信息 |
| Marketingdata.updateAndAdd | 语法 | Public Boolean updateAndAdd(MarketingPO po) |
| 前置条件 | 更新或修改网站营销人员的信息 |
| 后置条件 | 更新该序列化信息 |
| Marketingdata.delete | 语法 | Public Boolean delete(MarketingPO po) |
| 前置条件 | 无 |
| 后置条件 | 在序列化信息中删除该网站营销人员账号信息 |

#### **表5.4.2-8数据层Strategydata模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Strategydata.addPrivilege | 语法 | Public boolean addPrivilege(PrivilegePO po) throws RemoteException; |
| 前置条件 | 有新增会员福利信息需持久化保存 |
| 后置条件 | 在数据库中保存该会员福利持久化对象 |
| Strategydata.findAllPrivilege | 语法 | Public List<PrivilegePO> findAllPrivilege(String VIPType)throws RemoteException; |
| 前置条件 | 需要获取所有VIPType会员信息 |
| 后置条件 | 从数据库中返回所有VIPType会员福利信息持久化对象列表 |
| Strategydata.deletePrivilege | 语法 | Public boolean deletePrivilege(PrivilegePO po) throws RemoteException; |
| 前置条件 | 需删除某会员福利信息持久化对象 |
| 后置条件 | 在数据库中删除该会员福利信息持久化对象信息 |
| Strategdata.addMarketingStrategy | 语法 | Public boolean addMarketingStrategy(MarkeetingStrategyPO po) throws RemoteException; |
| 前置条件 | 已知网站营销人员或酒店管理人员的促销策略,需要将促销信息持久化保存 |
| 后置条件 | 持久化保存促销信息 |
| Strategydata.getMarketingStrategy | 语法 | Public List<StrategyPO> getMarketingStrategy(String id) throws RemoteException; |
| 前置条件 | 根据账户id需要获得已存在的制定策略 |
| 后置条件 | 从数据库中返回已存在的制定策略 |
| Strategydata.deleteMarketingStrategy | 语法 | Public boolean (MarketingStrategyPO po) throws RemoteException; |
| 前置条件 | 需要删除制定策略 |
| 后置条件 | 持久化删除制定策略 |
| Strategydata.gethotelStrategy | 语法 | Public List<HotelStrategyPO> gethotelStrategy()throws RemoteException; |
| 前置条件 | 需调用数据库中的促销策略 |
| 后置条件 | 返回数据库中的促销策略 |
| Strategydata.updatehotelStrategy | 语法 | Public List<HotelStrategyPO> updatehotelStrategy()throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 在数据层更新酒店促销策略 |

# 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 |
| ClientPO | 客户ID，姓名，密码，手机号，会员类别（等级），信用记录，生日，所属企业 |
| OrderPO | 订单号、订单的所属客户帐号、名字、联系电话、订单的下达时间、订单的状态（正常未执行、正常已执行、撤销、异常）、订单的酒店、订单的房间信息、订单价格、享受的策略 |
| LogPO | 操作发起人的用户种类、帐号、操作发起的时间、具体操作 |
| SiteManagerPO | 网站管理人员的帐号、联系电话、密码 |
| roominfoPO | 酒店房间信息：包括房间的类型，房间号，价格，状态 |
| hotelinfoPO | 酒店信息：包括酒店的名称，地址，商圈，电话，房间，星级，评价，服务设施，酒店工作人员信息 |
| hotelstaffPO | 酒店工作人员信息：包括酒店工作人员的账号，密码 |
| MarketingPO | 姓名，账号，电话号码，登录密码 |
| LevelPO | 等级名称，等级层次，所需信用值 |
| PrivilegePO | 会员种类，折扣，类型（日常或生日特惠） |
| StrategyPO | 种类，活动时间，折扣，酒店，海报，客户界面位置，商圈，最低消费金额，会员类型和种类，房间数， |

持久化用户对象如ClientPO的定义如图6.1，更多定义见原型代码









## 6.2 数据库表

|  |  |
| --- | --- |
| 表名 | 内容 |
| Client | 使用系统的客户信息 |
| Credit | 信用记录 |
| HotelStaff | 使用系统的酒店工作人员的信息 |
| Marketing | 使用系统的网站营销人员的信息 |
| Sitemanager | 使用系统的网站管理人员的信息 |
| Order | 订单信息 |
| Log | 日志 |
| HotelStrategy | 酒店营销策略 |
| MarketingStrategy | 网站营销策略 |
| HotelInfo | 酒店信息 |
| RoomInfo | 房间信息 |