酒店管理系统

Management System of Hotels

软件体系结构描述文档

V1.1

南京大学软件学院 Octopus小组

成员：周沁涵、桑田、钱柯宇、潘潇睿

2016-10-09

目录

[更新历史 3](#_Toc464344590)

[1. 引言 4](#_Toc464344591)

[1.1 编制目的 4](#_Toc464344592)

[1.2 词汇表 4](#_Toc464344593)

[1.3 参考资料 4](#_Toc464344594)

[2. 产品概述 5](#_Toc464344595)

[3. 逻辑视角 6](#_Toc464344596)

[3. 组合视角 7](#_Toc464344597)

[4.1 开发包图 9](#_Toc464344598)

[4.2 运行时进程 13](#_Toc464344599)

[4.3 物理部署 14](#_Toc464344600)

[5. 接口视角 14](#_Toc464344601)

[5.1 模块的职责 14](#_Toc464344602)

[5.2 用户界面层的分解 18](#_Toc464344603)

[5.2.1 用户界面层模块的职责 20](#_Toc464344604)

[5.2.2 用户界面层模块的接口规范 20](#_Toc464344605)

[5.2.3 用户界面模块设计原理 22](#_Toc464344606)

[5.3 业务逻辑层的分解 23](#_Toc464344607)

[5.3.1 业务逻辑层模块的职责 23](#_Toc464344608)

[5.3.2 业务逻辑层模块的接口规范 24](#_Toc464344609)

[5.4 数据层的分解 48](#_Toc464344610)

[5.4.1 数据层模块的职责 48](#_Toc464344611)

[5.4.2 数据层模块的接口规范 49](#_Toc464344612)

[6. 信息视角 63](#_Toc464344613)

[6.1 数据持久化对象 63](#_Toc464344614)

[6.2 数据库表 67](#_Toc464344615)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 全体成员 | 2016-10-9 | 最初草稿 | V0.0.0 草稿 |
| 桑田 | 2016-10-10 | 修正排版 | V1.0.0 |
| 周沁涵 | 2016-10-11 | 修改“逻辑视角”部分内容 | V1.0.1 |
| 钱柯宇，潘潇睿 | 2016-10-15 | 修正逻辑接口和数据接口部分内容 | V1.1.0 |
| 潘潇睿 | 2016-10-16 | 删改Order逻辑层接口中的两个方法 | V1.1.1 |
| 周沁涵 | 2016-10-16 | 删除数据层接口中的Customer\_Data\_Service.delete，修改promotionPO | V1.1.2 |

# 1. 引言

## 1.1 编制目的

本报表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

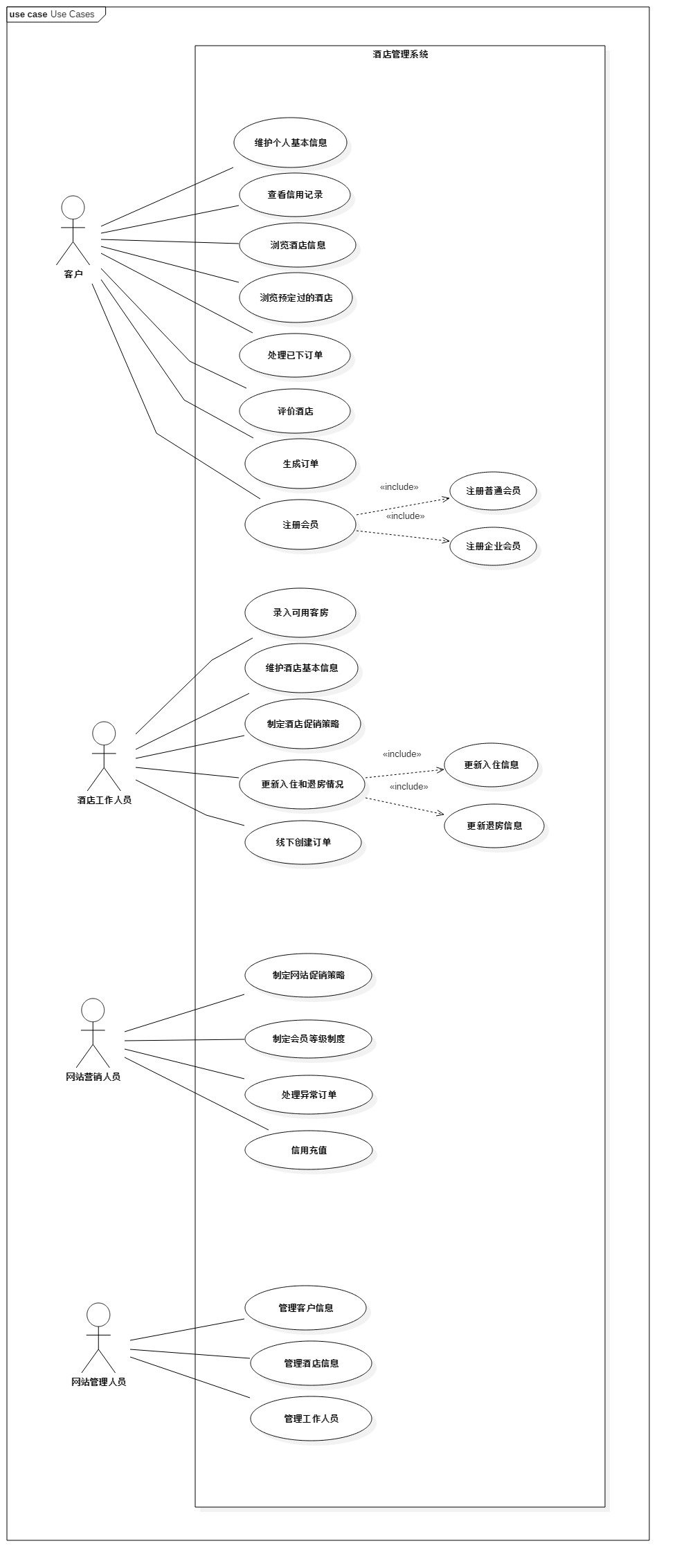
|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| \_ui | 表示某展示层 |  |
| \_bl | 表示某逻辑层 |  |
| \_data | 表示某数据层 |  |
| RMI | 表示远程方法调用 |  |
| utility\_bl | 表示初始化和业务逻辑上下文的工作 |  |
| datafactory | 表示调用其他数据库的方法 |  |

## 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软件需求规格说明文档中队产品的概括描述。酒店管理系统主要是应用于建立线上酒店预订系统，并添加营销策略与管理,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、

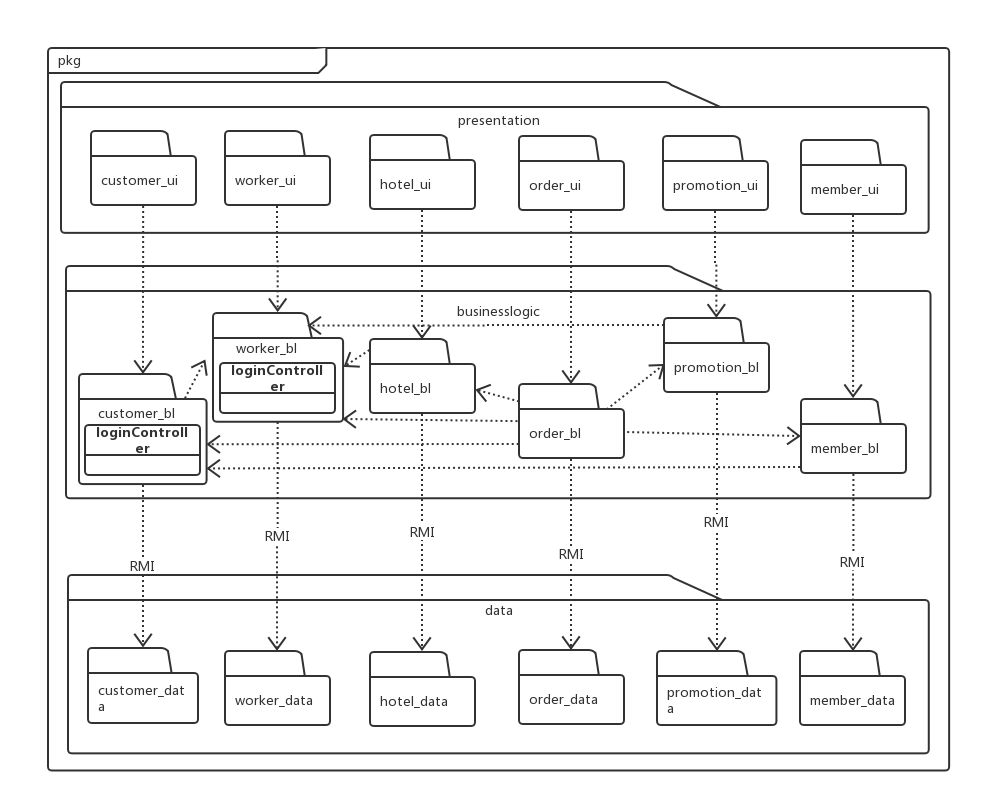
业务逻辑层、数据层)能够很好地示意整个高层抽象。展示层包含GUI页面

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一和图二所示。



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

# 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data\_service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data\_service包，data\_service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databaseutility。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main\_ui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Promotion:将部分Order类抽象接口置入Promotion包，这样Order单向依赖于Promotion（实现接口+调用）
* Hotel和Order:将部分Order类抽象接口orderInfoService置入Hotel包，这样Order单向依赖于Hotel（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\_bl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和图4.1-2所示：

## 4.1 开发包图

表4.1-1 开发包设计

|  |  |
| --- | --- |
| 开发（物理）包 | 依赖的其他开发包 |
| main\_ui | login\_ui, customer\_ui, clerk\_ui, marketer\_ui, manager\_ui, search\_ui, hotel\_ui, order\_ui, member\_ui, promotion\_ui, vo |
| login\_ui | login\_bl\_service,vo,界面类库包 |
| login\_bl\_service |  |
| login\_bl | login\_bl\_service, data\_factory,po, utlity\_bl |
| customer\_ui | customer\_bl,vo,界面类库包 |
| cutomer\_bl\_service |  |
| customer\_bl | cutomer\_bl\_service,utlity,po, customer\_data\_service |
| customer\_data\_service | Java RMI,po |
| customer\_data |  |
| clerk\_ui | clerk\_bl\_service,vo,界面类库包 |
| clerk\_bl\_service |  |
| clerk\_bl | clerk\_data\_service, data\_factory,utility\_bl,po |
| clerk\_data\_service | Java RMI,po |
| clerk\_data |  |
| marketer\_ui | marketer\_bl\_service,vo,界面类库包 |
| marketer\_bl\_service |  |
| marketer\_bl | marketer\_bl\_service, data\_factory, utility\_bl,po |
| marketer\_data\_service | Java RMI,po |
| marketer\_data |  |
| manager\_ui | manager\_bl\_service, vo,界面类库包 |
| manager\_bl\_service |  |
| manager\_bl | manager\_data\_service, manager\_bl\_service, utility\_bl,po |
| manager\_data\_service | Java RMI,po |
| manager\_data |  |
| search\_ui | search\_bl\_service,vo,界面类库包 |
| search\_bl\_service |  |
| search\_bl | search\_bl\_service,data\_factory, utility\_bl,po |
| hotel\_ui | hotel\_bl\_service, vo,界面类库包 |
| hotel\_bl\_service |  |
| hotel\_bl | manager\_bl,clerk\_bl,utility,data\_factory, hotel\_data\_service |
| hotel\_data\_service | Java RMI,po |
| hotel\_data |  |
| order\_ui | order\_bl\_service, vo,界面类库包 |
| order\_bl\_service |  |
| order\_bl | order\_bl\_service ,customer\_bl,clerk\_bl,marketer\_bl,hotel\_bl,member\_bl,promoteon\_bl,utility,po,data\_factory, order\_data\_service |
| order\_data\_service | Java RMI,po |
| order\_data |  |
| member\_ui | member\_bl\_service, vo,界面类库包 |
| member\_bl\_service |  |
| member\_bl | member\_bl\_service,customer\_bl,utility,po, member\_data\_service |
| member\_data\_service | Java RMI,po |
| member\_data |  |
| promotion\_ui | promotion\_bl\_service, vo,界面类库包 |
| promotion\_bl\_service |  |
| promotion\_bl | promotion\_bl\_service, promotion\_data\_service |
| promotion\_data\_service | Java RMI,po |
| promotion\_data |  |
| vo |  |
| po |  |
| utility\_bl |  |
| 界面类库包 |  |
| Java RMI |  |
| Datafactory\_service |  |
| databaseutility | JDBC |

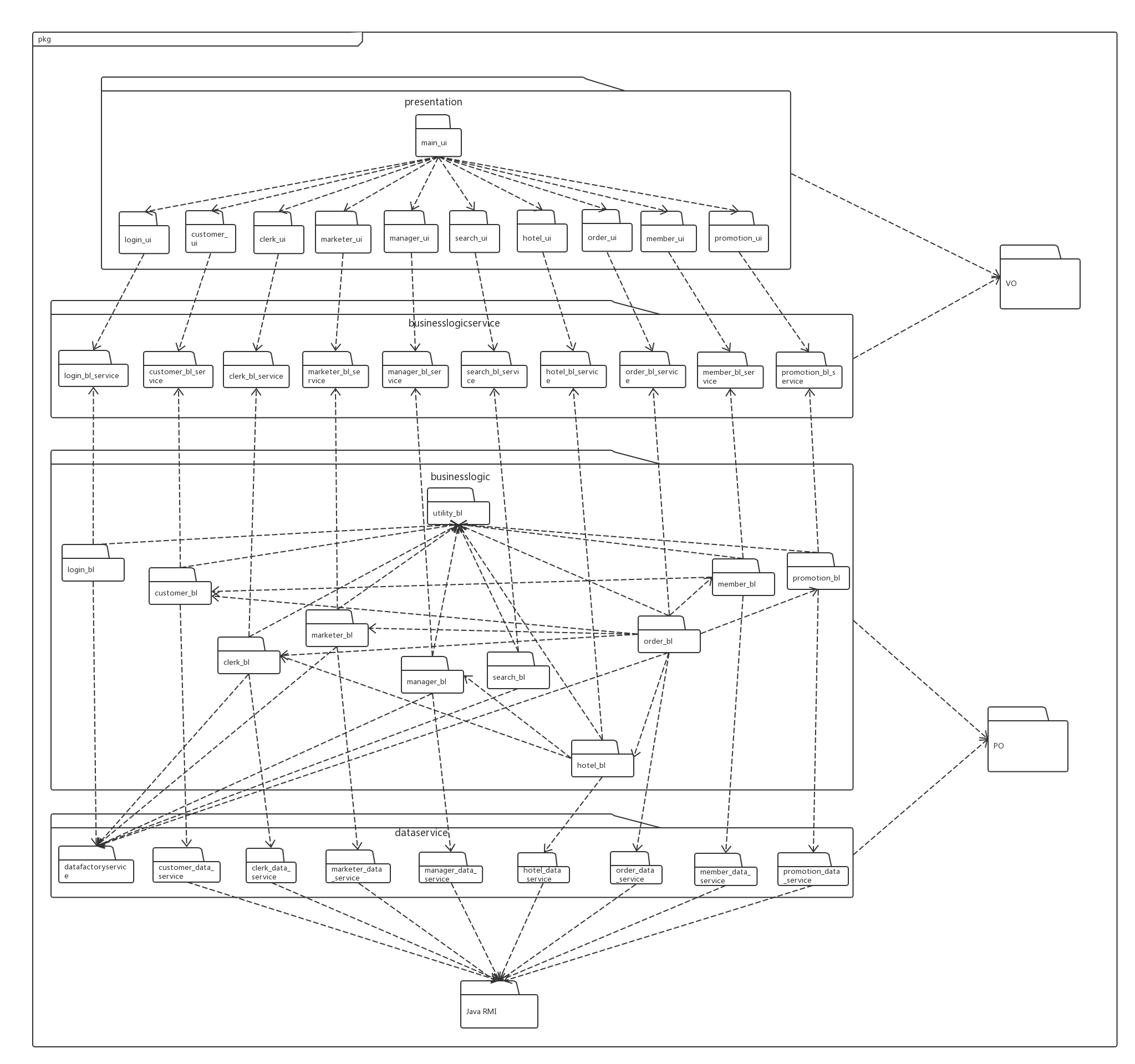


图4.1-1 酒店管理系统客户端开发包图

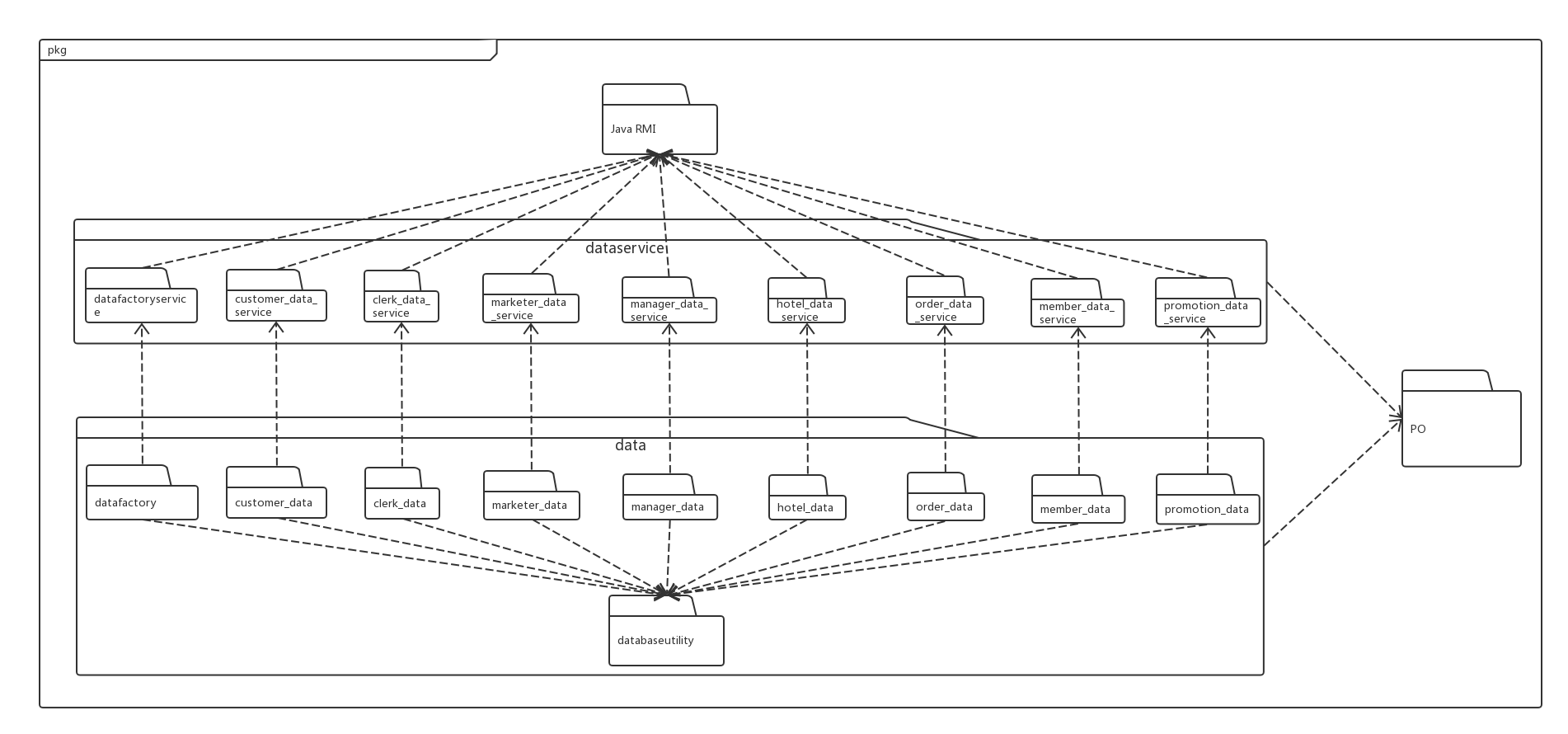


图4.1-2 酒店管理系统服务器端开发包图

## 4.2 运行时进程

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，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。



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5.2-2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5.2-2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。 |
| 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 |
| login\_bl\_service  customer\_bl\_service  clerk\_bl\_service  marketer\_bl\_service  manager\_bl\_service  search\_bl\_service  hotel\_bl\_service  order\_bl\_service  member\_bl\_service  promotion\_bl\_service | 客户端展示层 | 客户端业务逻辑层 |
| customer\_data\_service  clerk\_data\_service  marketer\_data\_service  manager\_data\_service  hotel\_data\_service  order\_data\_service  member\_data\_service  promotion\_data\_service  datafactoryservice | 客户端业务逻辑层 | 服务器端数据层 |

借用客户维护个人信息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）。customer\_bl\_service提供了customer界面所需要的所有业务逻辑功能customer\_data\_service提供了对数据库的查询、修改等操作。这样的实现就大大降低了层与层之间的耦合。

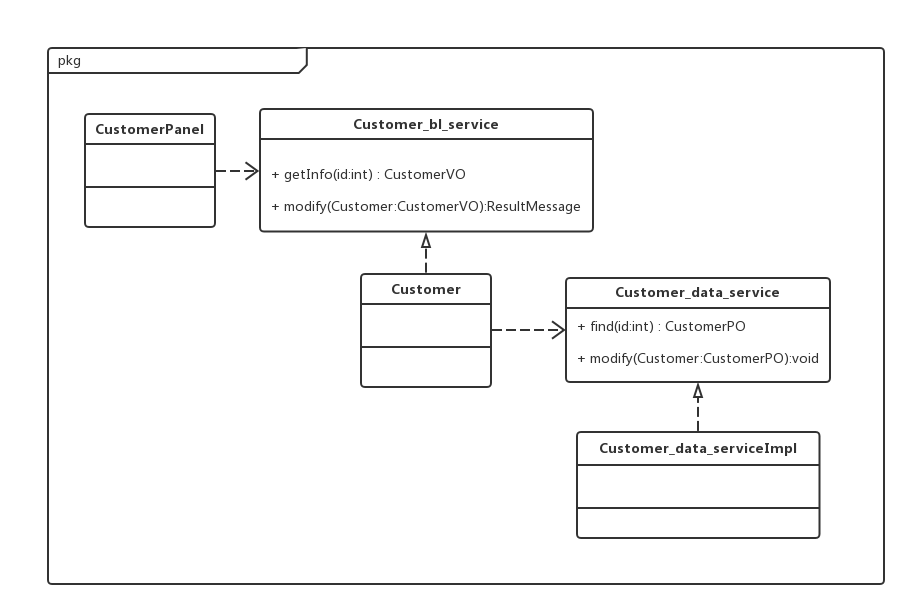


图5.1-3 客户维护个人基本信息用例层之间调用的接口

## 5.2 用户界面层的分解

根据需求，系统存在42个用户界面：登录界面，客户界面，客户注册界面，客户个人信息维护界面，客户密码修改界面，信用记录查看界面，注册普通会员界面，注册企业会员界面，我的订单查看界面，订单详细信息查看界面，评价酒店界面，预定过的酒店查看界面，酒店搜索结果界面，酒店详细信息查看界面，生成订单界面，酒店工作人员界面，酒店工作人员个人信息维护界面，酒店工作人员密码修改界面，录入可用客房界面，酒店基本信息维护界面，制定酒店促销策略界面，酒店订单列表查看界面，更新入住信息界面，更新退房信息界面，订单详细信息查看界面，网站营销人员界面，网站营销人员个人信息维护界面，网站营销人员密码修改界面，制定网站促销策略界面，制定会员等级制度界面，异常订单列表查看界面，订单详细信息查看界面，信用充值界面，网站管理人员界面，用户查询界面，用户基本信息维护界面，酒店查询界面，酒店信息维护界面，增加酒店界面，工作人员查询界面，工作人员信息维护界面，增加工作人员界面。

用户界面跳转如图5.2-1所示。

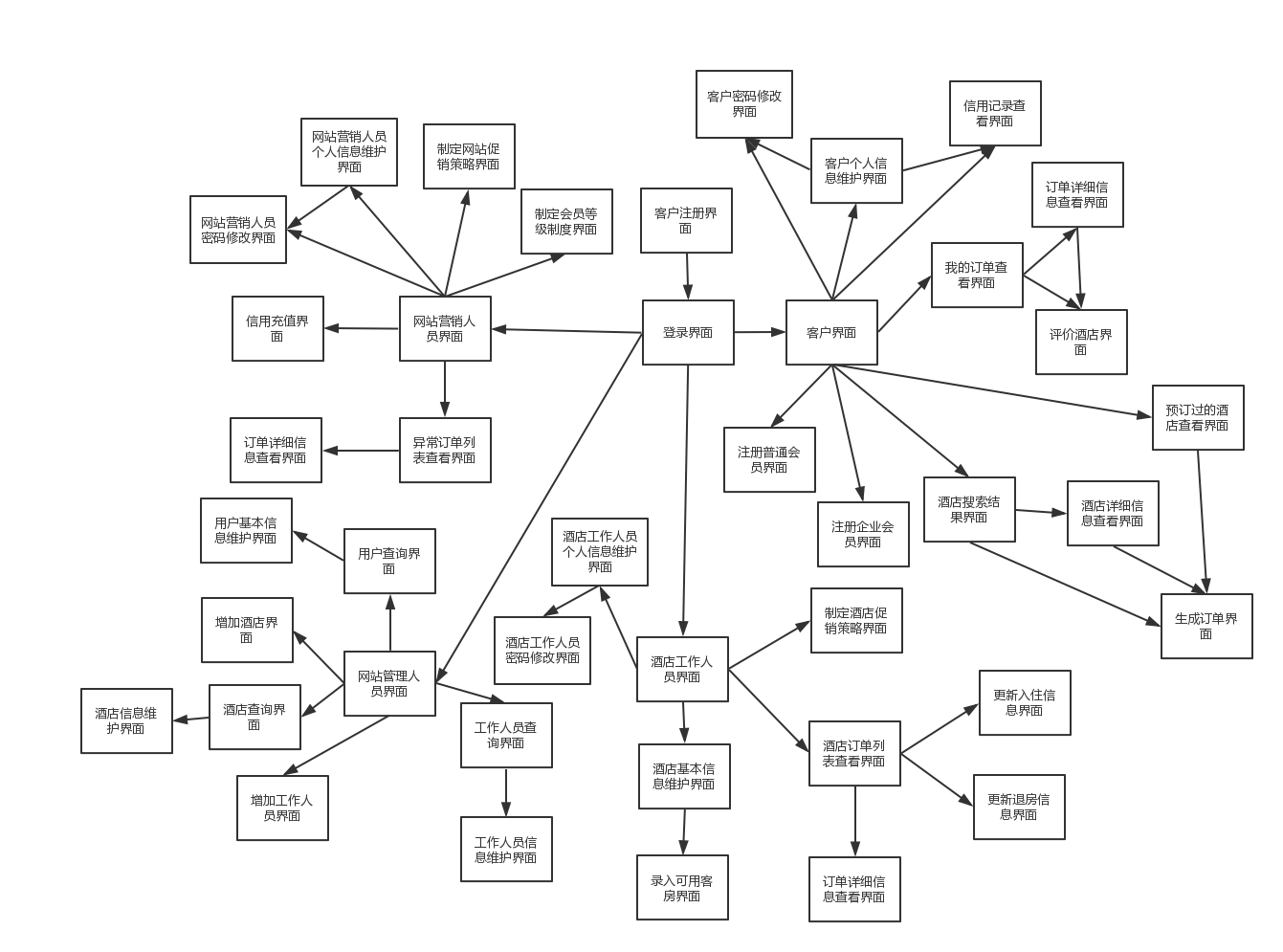


图5.2-1 用户界面跳转

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所示为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 用户界面层模块的接口规范

用户界面层模块的接口规范如表6所示。

表6 用户界面层模块的接口规范

|  |  |  |
| --- | --- | --- |
| Main\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示Frame及LoginPanel |
| Login\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示LoginPanel |
| Customer\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示CustomerPanel |
| Clerk\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示ClerkPanel |
| Marketer\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示MaketerPanel |
| Manager\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示ManagerPanel |
| Search\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示SearchPanel |
| Hotel\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示HotelPanel |
| Order\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示OrderPanel |
| Member\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示MemberPanel |
| Promotion\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示PromotionPanel |

用户界面层模块需要的服务接口如表7所示。

表7 用户界面层模块需要的服务接口

|  |  |
| --- | --- |
| 服务名 | 服务 |
| businesslogicservice.login\_bl\_service | 登录界面的业务逻辑接口 |
| businesslogicservice.customer\_bl\_service | 处理客户信息的接口 |
| businesslogicservice.clerk \_bl\_service | 处理酒店工作人员信息的接口 |
| businesslogicservice.maketer \_bl\_service | 处理网站营销人员信息的接口 |
| businesslogicservice.manager \_bl\_service | 处理网站管理人员信息的接口 |
| businesslogicservice.search\_bl\_service | 提供搜索功能的接口 |
| businesslogicservice.hotel\_bl\_service | 处理酒店信息的接口 |
| businesslogicservice.order \_bl\_service | 处理订单信息的接口 |
| businesslogicservice.member\_bl\_service | 处理会员信息的接口 |
| businesslogicservice.promotion\_bl\_service | 负责处理营销策略信息的接口 |

### 5.2.3 用户界面模块设计原理

用户界面利用Java的JavaFx库来实现。

## 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，Customer对象负责处理登陆界面的业务逻辑；Customer对象负责处理维护个人信息的业务逻辑。业务逻辑层的设计如图5.3-1所示。

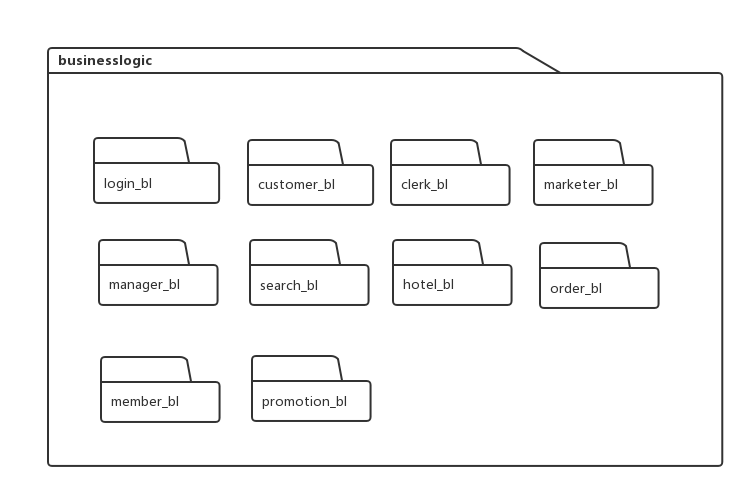


图5.3-1 业务逻辑层

### 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8所示。

表5 业务逻辑层模块的职责

|  |  |  |
| --- | --- | --- |
| 模块 | 职责 | |
| login\_bl | 负责提供验证客户登陆的服务 | |
| customer\_bl | 负责提供注册用户账号和维护基本信息功能的服务 | |
| marketer\_bl | 负责提供注册营销人员账号和维护基本信息的服务 | |
| clerk\_bl | 负责提供注册酒店工作人员账号的服务 | |
| manager\_bl | 负责提供增删改用户、工作人员、营销人员信息的服务 | |
| hotel\_bl | 负责提供录入客房、增减酒店房间数量和维护酒店基本信息的服务 | |
| order\_bl | 负责提供订单的生成、浏览和执行的服务 | |
| member\_bl | 负责提供注册会员的服务 | |
| promotion\_bl | 负责提供酒店促销策略和网站营销策略的制定与执行服务 | |
| search\_bl | 负责提供搜索订单与用户的服务 | |
| utility\_bl | 负责初始化和业务逻辑上下层 |

### 5.3.2 业务逻辑层模块的接口规范

Customer\_bl模块的接口规范如表5.3.2-1所示。

表5.3.2-1 customer\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Customer.signUp | 语法 | public ResultMessage signUp(CustomerVO customerVO) |
| 前置条件 | 必要信息全都按规范填写 |
| 后置条件 | 查找是否存在相应的ID、手机、邮箱，并返回注册的结果，如果 注册成功，则新增持久化对象 |
| Customer.changeInfo | 语法 | public ResultMessage changeInfo( CustomerVO customerVO) |
| 前置条件 | 修改的信息格式符合规范 |
| 后置条件 | 返回修改是否成功的结果 |
| Customer.changePassword | 语法 | public ResultMessage changePassword(String ID, String oldPw, String newPw1, String newPw2); |
| 前置条件 | 客戶已登錄 |
| 后置条件 | 更改客戶密碼 |
| Customer.getSingle | 语法 | public CustomerVO getSingle(String ID) |
| 前置条件 | 输入的ID存在 |
| 后置条件 | 返回该ID的Customer |
| Customer.getAll | 语法 | public List<CustomerVO> getAll() |
| 前置条件 | 无 |
| 后置条件 | 返回所有的Customer 信息 |
| Customer.getCreditRecord | 语法 | public CreditRecordVO getCreditRecord (CustomerVO customerVO) |
| 前置条件 | 无 |
| 后置条件 | 返回该用户的信用记录 |
| Customer.getHistoryHotel | 语法 | public List<HotelVO> getHistoryHotel (CustomerVO customerVO)； |
| 前置条件 | 无 |
| 后置条件 | 返回该用户的预定过的酒店 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Customer\_data\_service. modify(CustomerPO po) | 修改单一持久化对象 | |
| Customer\_data\_service.  add(CustomerPO po) | 新增单一持久化对象 | |
| Customer\_data\_service. find(String ID) | 根据ID查找用户 | |
| Customer\_Data\_Service.getCurrentCredit(String ID) | 获取用户当前信用值 | |
| Customer\_Data\_Service.getReservedHotel(String user\_id) | 获取用户预订过的酒店名列表 | |

Clerk\_bl模块的接口规范如表5.3.2-2所示。

表5.3.2-2Clerk\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Clerk.getSingleByID | 语法 | public ClerkVO getSingleByID(String ID) |
| 前置条件 | 输入的ID存在 |
| 后置条件 | 返回该ID的Clerk信息 |
| Clerk.getSingleByName | 语法 | public ClerkVO getSingleByName(String name) |
| 前置条件 | 输入的name存在 |
| 后置条件 | 返回该name的Clerk信息 |
| Clerk.getAll | 语法 | public List<ClerkVO> getAll() |
| 前置条件 | 无 |
| 后置条件 | 返回所有的Clerk 信息 |
| Clerk.changeInfo | 语法 | public ResultMessage changeInfo(ClerkVO clerkVO); |
| 前置条件 | 修改的信息格式符合规范 |
| 后置条件 | 返回修改是否成功的结果 |
| Clerk.changePassword | 语法 | public ResultMessage changePassword(String ID, String oldPW, String newPW1. String newPW2); |
| 前置条件 | 酒店工作人員已登錄 |
| 后置条件 | 更改酒店工作人員密碼 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Hotel\_data\_service. Find(String ID) | 根据ID查找酒店 | |
| Clerk\_Data\_Service.findByClerkID (String ID) | 根据ID查找工作人员 | |
| Clerk\_Data\_Service.findByClerk (String name) | 根据姓名查找工作人员 | |

Marketer\_bl模块的接口规范如表5.3.2-3所示。

表5.3.2-3Marketer\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Marketer.getSingleByID | 语法 | public MarketerVO getSingleByID(String ID) |
| 前置条件 | 输入的ID存在 |
| 后置条件 | 返回该ID的Marketer信息 |
| Marketer.getSingleByName | 语法 | public MarketerVO getSingleByName(String name) |
| 前置条件 | 输入的name存在 |
| 后置条件 | 返回该name的Marketer信息 |
| Marketer.getAll | 语法 | public List<MarketerVO> getAll() |
| 前置条件 | 无 |
| 后置条件 | 返回所有的Marketer 信息 |
| Marketer.CreditCharge | 语法 | public ResultMessage creditCharge(String ID, CustomerVO vo) |
| 前置条件 | 无 |
| 后置条件 | 输入的内容正确，返回充值信用是否成功的信息 |
| Marketer.changeInfo | 语法 | public ResultMessage changeInfo(MarketerVO marketerVO) |
| 前置条件 | 修改的信息格式符合规范 |
| 后置条件 | 返回修改是否成功的结果 |
| Marketer.changePassword | 语法 | public ResultMessage changePassword(String ID, String oldPw, String newPw1, String newPw2); |
| 前置条件 | 營銷人員已登錄 |
| 后置条件 | 更改營銷人員密碼 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Marketer\_Data\_Service.findBy MarketerName (String name) | 根据name查找营销人员 | |
| Marketer\_Data\_Service.findByMarketerID（String ID） | 根据ID查找营销人员 | |
| Marketer\_Data\_Service.findByPosition(WorkerPosition position) | 根据职位查找营销人员 | |
| DataFactory\_Data\_Service.addCreditRecord(CreditRecordPO crPO) | 增加用户信用记录 | |

Manager\_bl模块的接口规范如表5.3.2-4所示。

表5.3.2-4Manager\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Manager.addCustomer | 语法 | public ResultMessage addCustomer( CustomerVO customerVO) |
| 前置条件 | 无 |
| 后置条件 | 检查输入的用户信息，返回修改是否成功的结果 |
| Manager.changeCustomerInfo | 语法 | public ResultMessage changeCustomerInfo( CustomerVO customerVO) |
| 前置条件 | 更改的用户存在 |
| 后置条件 | 检查更改的用户信息，返回更改是否成功的结果 |
| Manager.addClerk | 语法 | public ResultMessage addClerk( ClerkVO clerkVO) |
| 前置条件 | 无 |
| 后置条件 | 检查输入的工作人员信息，返回修改是否成功的结果 |
| Manager.deleteClerk | 语法 | public ResultMessage deleteClerk( ClerkVO clerkVO) |
| 前置条件 | 删除的工作人员存在 |
| 后置条件 | 返回删除是否成功的结果 |
| Manager.changeClerkInfo | 语法 | public ResultMessage changeClerkInfo( ClerkVO clerkVO) |
| 前置条件 | 更改的工作人员存在 |
| 后置条件 | 检查更改的工作人员信息，返回更改是否成功的结果 |
| Manager.addMarketer | 语法 | public ResultMessage addMarketer (MarketerVO marketerVO) |
| 前置条件 | 无 |
| 后置条件 | 检查输入的营销人员信息，返回修改是否成功的结果 |
| Manager.deleteMarketer | 语法 | public ResultMessage deleteMarketer (MarketerVO marketerVO) |
| 前置条件 | 删除的营销人员存在 |
| 后置条件 | 返回删除是否成功的结果 |
| Manager.changeMarketerInfo | 语法 | public ResultMessage changeMarketerInfo(MarketerVO marketerVO) |
| 前置条件 | 更改的营销人员存在 |
| 后置条件 | 检查更改的营销人员信息，返回更改是否成功的结果 |
| Manager.addHotel | 语法 | public ResultMessage addHotel (HotelVO hotelVO) |
| 前置条件 | 无 |
| 后置条件 | 检查输入的酒店信息，返回修改是否成功的结果 |
| Manager.deleteHotel | 语法 | public ResultMessage deleteHotel (HotelVO hotelVO) |
| 前置条件 | 删除的酒店存在 |
| 后置条件 | 返回删除是否成功的结果 |
| Manager.changeHotelInfo | 语法 | public ResultMessage changeHotelInfo(HotelVO hotelVO) |
| 前置条件 | 更改的营销人员存在 |
| 后置条件 | 检查更改的营销人员信息，返回更改是否成功的结果 |
| Manager.addClerk | 语法 | public ResultMessage addClerk(HotelVO hotelVO,ClerkVO clerkVO) |
| 前置条件 | 该酒店已添加，没有工作人员 |
| 后置条件 | 检查添加的工作人员信息，返回添加是否成功的结果 |
| Manager.addMarketer | 语法 | public ResultMessage addMarketer(MarketerVO marketerVO) |
| 前置条件 | 该酒店已添加，没有工作人员 |
| 后置条件 | 检查添加的工作人员信息，返回添加是否成功的结果 |
| Manager.changePassword | 语法 | public ResultMessage changePassword(String ID, String oldPW, String newPW1, String newPW2) |
| 前置条件 | 该管理人员已存在 |
| 后置条件 | 更改管理人员的密码 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Customer\_data\_service.modify(CustomerPO po) | 修改单一持久化对象 | |
| Customer\_data\_service.add(CustomerPO po) | 新增单一持久化对象 | |
| Customer\_data\_service.delete(CustomerPO po) | 删除单一持久化对象 | |
| Clerk\_data\_service.modify(ClerkPO po) | 修改单一持久化对象 | |
| Clerk\_data\_service.add(ClerkPO po) | 新增单一持久化对象 | |
| Clerk\_data\_service.delete(ClerkPO po) | 删除单一持久化对象 | |
| Marketer\_data\_service.modify(MarketerPO po) | 修改单一持久化对象 | |
| Marketer\_data\_service.add(MarketerPO po) | 新增单一持久化对象 | |
| Marketer\_data\_service.delete(MarketerPO po) | 删除单一持久化对象 | |
| Manager\_data\_service.find(StringID ) | 按ID查找网站管理人员 | |

Login\_bl模块的接口规范如表5.3.2-5所示。

表5.3.2-5Login\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Login.login | 语法 | public ResultMessage login(String ID, String password) |
| 前置条件 | 启动登陆服务，已知账号和密码 |
| 后置条件 | 查找账号密码是否对应，返回登录信息（登录是否成功） |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Customer\_data\_service. find(String ID) | 根据ID查找用户 | |
| Clerk\_data\_service. find(String ID) | 根据ID查找酒店工作人员 | |
| Marketer\_data\_service. find(String ID) | 根据ID查找网站营销人员 | |
| Manager\_data\_service. find(String ID) | 根据ID查找网站管理人员 | |

Hotel\_bl模块的接口规范如表5.3.2-6所示。

表5.3.2-6Hotel\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotel.addRoom | 语法 | public ResultMessage addRoom(RoomVO roomVO) |
| 前置条件 | 无 |
| 后置条件 | 检查输入信息是否符合，返回增加房间结果 |
| Hotel.changeAvailableRoom | 语法 | public ResultMessage changeAvailableRoom(String ID, String type, int number，DailyRoomInfoVO dailyRoomInfoVO) |
| 前置条件 | 无 |
| 后置条件 | 检查房间数量是否正确，返回改变空闲房间结果 |
| Hotel.changeReservedRoom | 语法 | public ResultMessage changeReservedRoom(String type, int number, DailyRoomInfoVO dailyRoomInfoVO) |
| 前置条件 | 无 |
| 后置条件 | 检查房间数量是否正确，返回变化已预订房间结果 |
| Hotel.changeOccupiedRoom | 语法 | public ResultMessage changeOccupiedRoom(String type, int number, DailyRoomInfoVO dailyRoomInfovo) |
| 前置条件 | 无 |
| 后置条件 | 检查房间数量是否正确，返回变化已入住房间结果 |
| Hotel.getRoom | 语法 | public List<RoomVo> getRoom(String ID) |
| 前置条件 | 无 |
| 后置条件 | 检查输入ID是否存在，返回酒店房间信息 |
| Hotel.getSingle | 语法 | public HotelVo getSingle(String ID) |
| 前置条件 | 无 |
| 后置条件 | 检查输入ID是否存在，返回酒店信息 |
| Hotel.getAll | 语法 | public List<HotelVo> getAll() |
| 前置条件 | 无 |
| 后置条件 | 返回所有酒店信息 |
| Hotel.addComment | 语法 | public void addComment(CommentVO commentVO, OrderVO orderVO) |
| 前置条件 | 订单已被评价 |
| 后置条件 | 评价信息输入正确，在酒店评价列表中增加评价，计算评分 |
| Hotel.sortByPrice | 语法 | public List<HotelVO> sortByPrice(List<HotelVO> list) |
| 前置条件 | 无 |
| 后置条件 | 将酒店按照价格排序 |
| Hotel.sortByStar | 语法 | public List<HotelVO> sortByStar(List<HotelVO> list) |
| 前置条件 | 无 |
| 后置条件 | 将酒店按照星级排序 |
| Hotel.sortByScore | 语法 | public List<HotelVO> sortByScore(List<HotelVO> list) |
| 前置条件 | 无 |
| 后置条件 | 将酒店按照评分排序 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Hotel\_Data\_Service.addRoomType（RoomPO po） | 新增单一持久化对象 | |
| Hotel\_Data\_Service.modifyRoomType(RoomPo po) | 修改房间信息 | |
| Hotel\_Data\_Service.getDailyRoomInfo（Date date） | 获取当日房间信息 | |
| Hotel\_Data\_Service.setDailyRoomInfo(List<DailyRoomInfoPo>) | 设置房间信息 | |
| Hotel\_Data\_Service.addComment(CommentPO po) | 增加酒店评价 | |
| Hotel\_Data\_Service.getComment(String HotelID) | 获取酒店评价 | |

Order\_bl模块的接口规范如表5.3.2-7所示。

表5.3.2-7Order\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Order.createOrder | 语法 | public ResultMessage createOrder(OrderVO orderVO) |
| 前置条件 | 酒店有可预定房间，信用值满足订房条件 |
| 后置条件 | 检查输入信息是否正确，返回创建订单结果;若创建成功，则生成订单对象 |
| Order.cancelOrder | 语法 | public ResultMessage cancelOrder(OrderVO orderVO) |
| 前置条件 | 该订单已生成但未执行 |
| 后置条件 | 返回撤销订单结果 |
| Order.executeOrder | 语法 | public ResultMessage executeOrder(OrderVO orderVO) |
| 前置条件 | 线上用户实际入住 |
| 后置条件 | 检查信息是否输入正确，返回执行订单结果 |
| Order.endOrder | 语法 | public ResultMessage endOrder(OrderVO orderVO) |
| 前置条件 | 用户实际退房 |
| 后置条件 | 检查信息是否输入正确，返回结束订单结果 |
| Order.setAbnormal | 语法 | public void setAbnormal(OrderVO orderVO) |
| 前置条件 | 用户超过预定时间仍未入住 |
| 后置条件 | 更改订单状态为异常 |
| Order.renewOrder | 语法 | public void renewOrder(OrderVO orderVO) |
| 前置条件 | 用户申诉合理 |
| 后置条件 | 更改订单状态为已撤销 |
| Order.getSingle | 语法 | public OrderVo getSingle(OrderVO orderVO) |
| 前置条件 | 输入信息合法 |
| 后置条件 | 返回该订单信息 |
| Order.getAll | 语法 | public List<OrderVO> getAll() |
| 前置条件 | 输入信息合法 |
| 后置条件 | 返回所有订单信息 |
| Order.getDiscount | 语法 | public int getDiscount(List<PromotionVO> list，OrderVO orderVO) |
| 前置条件 | 当前存在促销策略 |
| 后置条件 | 若符合促销策略条件，返回折扣后的价格 |
| Order.addCreditRecord | 语法 | public void addCreditRecord (OrderVO orderVO, CreditRecordVO creditRecordVO) |
| 前置条件 | 无 |
| 后置条件 | 根据订单状态，增加信用记录 |
| Order.changeStatus | 语法 | public void changeStatus (OrderVO orderVO) |
| 前置条件 | 无 |
| 后置条件 | 根据订单信息，更改订单状态 |
| Order.getPrice | 语法 | public double getPrice (OrderVO orderVO， OrderStatus orderStatus) |
| 前置条件 | 无 |
| 后置条件 | 根据订单信息，生成订单价格 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Datafactory\_Data\_Service.add(CreditRecordPO crPO) | 增加用户信用记录 | |
| Order\_Data\_Service.find | 获取订单信息 | |
| Order\_Data\_Service.add(OrderPo po) | 新增单一持久化对象 | |
| Order\_Data\_Service.changeStatus（OrderPO po, OrderCondition condition） | 更改订单状态 | |
| Order\_Data\_Service.getCondition（OrderPO po） | 获取订单状态 | |
| Datafactory\_Data\_Service.getCurrentCredit(String ID) | 获取用户当前信用值 | |
| Order\_Data\_Service.getPrice（OrderPO po） | 获取订单价格 | |
| Order\_Data\_Service.getStatus(OrderPO po) | 获取订单状态 | |
| Order\_Data\_Service.setActualCheckinTime (OrderPO po, Date actual\_date\_of\_arrival) | 设置实际到达时间 | |
| Order\_Data\_Service.setActualCheckOutTime (OrderPO po, Date actual\_date\_of\_departure) | 设置实际离开时间 | |
| Order\_Data\_Service.getLatestExecutedTime (OrderPO po) | 获取最晚入住时间 | |
| Order\_Data\_Service.findByCustomerID(String customerID) | 根据用户ID查找订单 | |
| Order\_Data\_Service.findCustomerIDAndOrderStatus(String customerID, OrderStatus status) | 根据用户和订单状态查找订单 | |
| Order\_Data\_Service.findByOrderStatus(OrderStatus Status) | 根据订单状态查找订单 | |

Member\_bl模块的接口规范如表5.3.2-8所示。

表5.3.2-8Member\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Member.signUp | 语法 | public ResultMessage signUp(MemberVO memberVO, CustomerVO customerVO) |
| 前置条件 | 用户的信用值符合条件 |
| 后置条件 | 检查输入信息是否符合，返回注册会员结果 |
| Member.upGrade | 语法 | public void upGrade(MemberVO memberVO, CustomerVO customerVO) |
| 前置条件 | 用户的信用值足够升级 |
| 后置条件 | 无 |
| Member.deGrade | 语法 | public void deGrade(MemberVO memberVO, CustomerVO customerVO) |
| 前置条件 | 用户的信用值不够当前等级 |
| 后置条件 | 无 |
| Member.getMember | 语法 | public MemberVO getMember(String ID) |
| 前置条件 | 无 |
| 后置条件 | 输入ID存在，返回该ID会员信息 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Member\_Data\_Service.add | 新增单一持久化对象 | |
| Member\_Data\_Service.Upgrade | 会员等级上升 | |
| Member\_Data\_Service.Degrade | 会员等级下降 | |
| Member\_Data\_Service.Delete | 删除会员信息 | |

Promotion\_bl模块的接口规范如表5.3.2-9所示。

表5.3.2-9Promotion\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Promotion.addPromotion | 语法 | public ResultMessage addPromotion(PromtionVO promotionVO) |
| 前置条件 | 无 |
| 后置条件 | 检查输入信息是否符合，返回新增促销策略结果 |
| Promotion.getAll | 语法 | public List<PromotionVO> getAll (Date date) |
| 前置条件 | 无 |
| 后置条件 | 返回当前时间内所有促销策略 |
| Promotion.getSingle | 语法 | public PromotionVO getSingle(Date date, String name) |
| 前置条件 | 无 |
| 后置条件 | 返回当前时间内的具体的促销策略 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Promotion\_Data\_Service.add | 新增单一持久化对象 | |
| Promotion\_Data\_Service.getTargetAera(PromotionPO po) | 获取营销策略针对地区 | |
| Promotion\_Data\_Service.getTargetHotel(PromotionPO po) | 获取营销策略针对酒店 | |
| Promotion\_Data\_Service.getTargetUser(PromotionPO po) | 获取营销策略针对用户 | |
| Promotion\_Data\_Service.getStartTime(PromotionPO po) | 获取营销策略开始时间 | |
| Promotion\_Data\_Service.getEndTime(PromotionPO po) | 获取营销策略结束时间 | |
| Promotion\_Data\_Service.getDiscount(PromotionPO po) | 获取营销策略折扣 | |
| Promotion\_Data\_Service.getMinRoom(PromotionPO po) | 获取营销策略最少预定房间 | |

Search\_bl模块的接口规范如表5.3.2-10所示。

表5.3.2-10Search\_bl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Search. searchByPosition | 语法 | public List<WorkerVO> searchByPosition(Position position) |
| 前置条件 | 管理工作人员界面 |
| 后置条件 | 输入的职位符合规范，返回该职位的用户列表 |
| Search. searchCustomer | 语法 | public List<CustomerVO> searchCustomer(String ID) |
| 前置条件 | 选择用户搜索界面， |
| 后置条件 | 输入的ID符合规范，返回搜索结果 |
| Search. searchWorkerByID | 语法 | public List<WorkerVO> searchWorkerByID(String ID) |
| 前置条件 | 管理工作人员界面 |
| 后置条件 | 输入的ID符合规范，返回搜索结果 |
| Search. searchWorkerByName | 语法 | public List<WorkerVO> searchWorkerByName(String name) |
| 前置条件 | 管理工作人员界面 |
| 后置条件 | 输入的名字符合规范，返回搜索结果 |
| Search. searchHotelByID | 语法 | public List<HotelVO> searchHotelByID(String ID) |
| 前置条件 | 管理酒店界面 |
| 后置条件 | 输入的ID符合规范，返回搜索结果 |
| Search. searchHotelByAddress | 语法 | public List<HotelVO> searchHotelAddress(String address) |
| 前置条件 | 管理酒店界面 |
| 后置条件 | 输入的地址符合规范，返回搜索结果 |
| Search. searchOrderByStatus | 语法 | public List<OrderVO> searchOrderByStatus(OrderSatus status) |
| 前置条件 | 无 |
| 后置条件 | 输入的酒店状态符合规范，返回搜索结果 |
| Search. searchOrderByCustomerName | 语法 | public List<OrderVO> searchOrderByCustomerName (String customerName) |
| 前置条件 | 无 |
| 后置条件 | 输入的用户名字符合规范，返回搜索结果 |
| Search. searchOrderByHotelName | 语法 | public List<OrderVO> searchOrderByHotelName(String hotelName) |
| 前置条件 | 无 |
| 后置条件 | 输入的酒店名符合规范，返回搜索结果 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Customer\_Data\_Service.find（String ID） | 按照ID查找用户 | |
| Clerk\_Data\_Service.findByClerk Name (String name) | 按照名字查找酒店工作人员 | |
| Clerk\_Data\_Service.findByClerkID (String ID) | 按照ID查找酒店工作人员 | |
| Marketer\_Data\_Service.findBy MarketerName (String name) | 按照名字查找网站营销人员 | |
| Marketer\_Data\_Service.findByMarketerID (String ID) | 按照ID查找网站营销人员 | |
| Order\_Data\_Service.find(String CustomerName) | 按照用户名字查找订单 | |
| Order\_Data\_Service.findByHotelID (String HotelID) | 按照酒店名查找订单 | |
| Order\_Data\_Service.findByOrderStatus (String status) | 按照订单状态查找订单 | |
| Order\_Data\_Service.findCustomerIDAndOrderStatus(String customerID, OrderStatus status) | 按照用户ID及订单状态查找酒店 | |

## 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。Customer业务逻辑需要的服务由CustomerDataService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Customer\_Data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Customer\_Data\_Service.add | 语法 | Public ResultMessage add(CustomerPO po) |
| 前置条件 | 同样ID的po在数据文件中不存在 |
| 后置条件 | 在数据文件中增加一个po记录 |
| Customer\_Data\_Service.modify | 语法 | Public ResultMessage modify (CustomerPO po) |
| 前置条件 | 逻辑层请求修改一条客户账户信息 |
| 后置条件 | 数据库修改账户 |
| Customer\_Data\_Service.find | 语法 | Public List<CustomerPO> find(String id) |
| 前置条件 | 相同ID的po在数据文件中已存在 |
| 后置条件 | 返回相应的CustomerPO结果 |
| Customer\_Data\_Service.getID | 语法 | Public String getID (CustomerPO po) |
| 前置条件 | 相同ID的po在数据文件中已存在 |
| 后置条件 | 返回相应的身份证号信息 |
| Customer\_Data\_Service.getCurrentCredit | 语法 | Public int getCurrentCredit (String user\_id) |
| 前置条件 | 数据文件中存在po记录 |
| 后置条件 | 返回该客户当前信用值 |
| Customer\_Data\_Service.getReservedHotel | 语法 | Public List<HotelPO> getReservedHotel (CustomerPO customerPO) |
| 前置条件 | 数据文件中存在po记录 |
| 后置条件 | 返回该客户预订过的酒店列表 |

表5.4.2-2数据层Clerk\_Data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Clerk\_Data\_Service.add | 语法 | Public ResultMessage add(ClerkPO po) |
| 前置条件 | 同样ID的po在数据文件中不存在 |
| 后置条件 | 在数据文件中增加一个po记录 |
| Clerk\_Data\_Service.modify | 语法 | Public ResultMessage modify (ClerkPO po) |
| 前置条件 | 逻辑层请求修改一条酒店工作人员账户信息 |
| 后置条件 | 数据库修改账户 |
| Clerk\_Data\_Service.findByClerk Name | 语法 | Public List<ClerkPO> findByClerkName (String name) |
| 前置条件 | 相同星系信息的po在数据文件中已存在 |
| 后置条件 | 返回相应的ClerkPO结果 |
| Clerk\_Data\_Service.findByClerkID | 语法 | Public List<ClerkPO> findByClerkID (String id) |
| 前置条件 | 相同ID信息的po在数据文件中已存在 |
| 后置条件 | 返回相应的ClerkPO结果 |
| Clerk\_Data\_Service.delete | 语法 | Public ResultMessage delete(ClerkPO po) |
| 前置条件 | 相同ID的po在数据文件中已存在 |
| 后置条件 | 删除该po记录 |

表5.4.2-3数据层Marketer\_Data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Marketer\_Data\_Service.add | 语法 | Public ResultMessage add(MarketerPO po) |
| 前置条件 | 相同ID的po在数据文件中不存在 |
| 后置条件 | 在数据文件中增加一个po记录 |
| Marketer\_Data\_Service.modify | 语法 | Public ResultMessage modify (Marketer PO po) |
| 前置条件 | 逻辑层请求修改一条网站营销人员账户信息 |
| 后置条件 | 数据库修改账户 |
| Marketer\_Data\_Service.findBy MarketerName | 语法 | Public List<MarketerPO> findByMarketerName (String name) |
| 前置条件 | 相同name信息的po在数据文件中已存在 |
| 后置条件 | 返回相应的MarketerPO结果 |
| Marketer\_Data\_Service.findByMarketerID | 语法 | Public List<MarketerPO> findByMarketerID (String id) |
| 前置条件 | 相同id信息的po在数据文件中已存在 |
| 后置条件 | 返回相应的MarketerPO结果 |
| Marketer\_Data\_Service.findByPosition | 语法 | public List<MarketerPO> findByPosition (WorkerPosition position) |
| 前置条件 | 数据文件中存在该类职位的工作人员 |
| 后置条件 | 返回数据文件中所有该职位人员的信息 |
| Marketer\_Data\_Service.delete | 语法 | Public ResultMessage delete(MarketerPO po) |
| 前置条件 | 相同ID的po在数据文件中已存在 |
| 后置条件 | 删除该po记录 |

表5.4.2-4数据层Manager\_Data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Manager\_Data\_Service.modify | 语法 | Public ResultMessage modify (Manager PO po) |
| 前置条件 | 逻辑层请求修改一条网站管理人员账户信息 |
| 后置条件 | 数据库修改账户 |

表5.4.2-5数据层Member\_Data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Member\_Data\_Service.add | 语法 | Public ResultMessage add(MemberPO po) |
| 前置条件 | 同样ID的po在会员数据文件中不存在 |
| 后置条件 | 在数据文件中增加一个po记录 |
| Member\_Data\_Service.upgrade | 语法 | Public ResultMessage upgrade (int grade) |
| 前置条件 | 会员数据文件中，该ID信用值满足升级条件 |
| 后置条件 | 数据库修改会员等级（上升1） |
| Member\_Data\_Service.degrade | 语法 | Public ResultMessage degrade (int grade) |
| 前置条件 | 会员数据文件中，该ID信用值低于其当前等级的要求信用值数 |
| 后置条件 | 数据库修改会员等级（下降1） |
| Member\_Data\_Service.getGrade | 语法 | Public int getGrade (MemberPO po) |
| 前置条件 | 数据文件中存在该po |
| 后置条件 | 返回该po会员等级 |
| Member\_Data\_Service.delete | 语法 | Public ResultMessage delete(MemberPO po) |
| 前置条件 | 会员数据文件中，该ID信用值低于会员最低信用值要求 |
| 后置条件 | 数据库将该会员po记录删除 |
| Member\_Data\_Service.getType | 语法 | Public MemberType getType (MemberPO po) |
| 前置条件 | 数据文件中存在该po |
| 后置条件 | 返回该po会员类型 |
| Member\_Data\_Service.getMember | 语法 | Public MemberPO getMember (String id) |
| 前置条件 | 数据文件中存在相同ID的po |
| 后置条件 | 返回该MemberPO |
| Member\_Data\_Service.addMemberLevel | 语法 | Public ResultMessage addMemberLevel (MemberLevelPO po) |
| 前置条件 | 逻辑层请求添加会员等级制度 |
| 后置条件 | 在数据文件中增加一个MemberLevelPO记录 |
| Member\_Data\_Service.modifyMemberLevel | 语法 | Public ResultMessage modifyMemberLevel (MemberLevelPO po) |
| 前置条件 | 数据文件中已存在MemberLevelPO记录 |
| 后置条件 | 在数据文件中更改该MemberLevelPO记录 |
| Member\_Data\_Service. getNumberOfMemberLevel | 语法 | Public int getNumberOfMemberLevel(MemberLevelPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回会员等级个数 |

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） | | |
| Order\_Data\_Service.add | 语法 | Public ResultMessage add(OrderPO po) |
| 前置条件 | 逻辑层请求增加一条订单记录 |
| 后置条件 | 在数据文件中增加一个po记录 |
| Order\_Data\_Service.findByCustomerID | 语法 | Public List<OrderPO> findByCustomerID(String customerID) |
| 前置条件 | 逻辑层要求通过客户ID获得订单列表 |
| 后置条件 | 返回对应的OrderPO |
| Order\_Data\_Service.findCustomerID  AndOrderStatus | 语法 | Public List<OrderPO> findCustomerIDAndOrderStatus (String customerID, OrderStatus status) |
| 前置条件 | 逻辑层要求通过客户名和订单状态获得订单列表 |
| 后置条件 | 返回对应的OrderPO |
| Order\_Data\_Service.findByHotelID | 语法 | Public List<OrderPO> findByHotelID(String HotelID) |
| 前置条件 | 逻辑层要求通过酒店ID获得订单列表 |
| 后置条件 | 返回对应的OrderPO |
| Order\_Data\_Service.findByOrderStatus | 语法 | Public List<OrderPO> findByOrderStatus(OrderPO po, OrderStatus Status) |
| 前置条件 | 逻辑层要求通过订单状态获得订单列表 |
| 后置条件 | 返回对应的OrderPO |
| Order\_Data\_Service.getPrice | 语法 | Public int getPrice(OrderPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回该po中的价格值 |
| Order\_Data\_Service.changeStatus | 语法 | Public ResultMessage changeStatus (OrderPO po, OrderStatus condition) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 更改该po记录的订单状态值 |
| Order\_Data\_Service.getStatus | 语法 | Public OrderStatus getStatus (OrderPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回该po中的订单状态值 |
| Order\_Data\_Service.setActualCheckinTime | 语法 | Public ResultMessage setActualDateOfArrival (OrderPO po, Date actualActualTime) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 数据文件中更新实际入住时间 |
| Order\_Data\_Service.setActualCheckOutTime | 语法 | Public ResultMessage setActualDateOfDeparture (OrderPO po, Date actualCheckoutTime) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 数据文件中更新实际离开时间 |
| Order\_Data\_Service.getLatestExecutedTime | 语法 | Public Date getLatestExecutedTime (OrderPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回该po最晚执行时间 |

表5.4.2-7数据层Hotel\_Data模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotel\_Data\_Service.add | 语法 | Public ResultMessage add(HotelPO po) |
| 前置条件 | 数据文件中不存在相同ID的po |
| 后置条件 | 在数据文件中增加一个po记录 |
| Hotel\_Data\_Service.modify | 语法 | Public ResultMessage modify (HotelPO po) |
| 前置条件 | 数据文件中存在相同ID的po |
| 后置条件 | 在数据文件中修改po记录 |
| Hotel\_Data\_Service.delete | 语法 | Public ResultMessage delete (HotelPO po) |
| 前置条件 | 数据文件中存在相同ID的po |
| 后置条件 | 删除该po数据 |
| Hotel\_Data\_Service.find | 语法 | Public List<HotelPO> find(String address, String area, Date expected\_date\_of\_arrival, Date expected\_date\_of\_departure， int star, double score) |
| 前置条件 | 逻辑层要求所有符合条件的po |
| 后置条件 | 返回符合条件的所有po |
| Hotel\_Data\_Service.getHotel | 语法 | Publice HotelPo getHotel(String id) |
| 前置条件 | 数据文件中存在相同ID的po |
| 后置条件 | 返回该hotelpo |
| Hotel\_Data\_Service. addRoomType | 语法 | Public ResultMessage addRoomType (RoomPO po) |
| 前置条件 | 逻辑层要求增加一个房间类型记录 |
| 后置条件 | 在数据文件中增加一个RoomPO记录 |
| Hotel\_Data\_Service.modifyRoomType | 语法 | Public ResultMessage modifyRoomType (RoomPO po) |
| 前置条件 | 数据文件中存在该房间类型对应的po记录 |
| 后置条件 | 数据文件中修改该po记录 |
| Hotel\_Data\_Service.deleteRoomType | 语法 | Public ResultMessage deleteRoomType (RoomPO po) |
| 前置条件 | 数据文件中存在该房间类型对应的po记录 |
| 后置条件 | 数据文件中删除该po记录 |
| Hotel\_Data\_Service.getRoomPrice | 语法 | Public double getRoomPrice (RoomPO po) |
| 前置条件 | 数据文件中存在该房间类型对应的po记录 |
| 后置条件 | 返回该RoomPO的价格值 |
| Hotel\_Data\_Service.getRoomType | 语法 | Public String getRoomType (RoomPO po) |
| 前置条件 | 数据文件中存在该房间类型对应的po记录 |
| 后置条件 | 返回该RoomPO的房间类型 |
| Hotel\_Data\_Service.getDailyRoomInfo | 语法 | Public DailyRoomInfoPo getDailyRoomInfo (Date date) |
| 前置条件 | 数据文件中存在该日期的po记录 |
| 后置条件 | 返回该DailyRoomInfo Po |
| Hotel\_Data\_Service.setDailyRoomInfo | 语法 | Public ResultMessage setDailyRoomInfo (List<DailyRoomInfoPo> list) |
| 前置条件 | 逻辑层请求更改每日房间信息 |
| 后置条件 | 数据文件中存储所有DailyRoomInfoPo |
| Hotel\_Data\_Service.addComment | 语法 | Public ResultMessage addComment (CommentPO po) |
| 前置条件 | 逻辑层要求增加一个评价记录 |
| 后置条件 | 在数据文件中增加一个po记录 |
| Hotel\_Data\_Service. getComment | 语法 | Public List<CommentPO> getCommentPo (String HotelID) |
| 前置条件 | 数据文件中存在相同ID的Commentpo |
| 后置条件 | 返回符合条件的所有CommentPo |
| Hotel\_Data\_Service.addTo  ListOfHotelReservedByCustomer | 语法 | Public ResultMessage addTo  ListOfHotelReservedByCustomer (HotelPO hotelPO, CustomerPO customerPO) |
| 前置条件 | 数据文件中存在该CustomerPO |
| 后置条件 | 将该酒店添加至该用户的预订过的酒店列表中并存储至数据文件中 |

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） | | |
| Promotion\_Data\_Service.add | 语法 | Public ResultMessaage add (PromotionPO po) |
| 前置条件 | 逻辑层要求增加一个促销策略记录 |
| 后置条件 | 在数据文件中增加一个po记录 |
| Promotion\_Data\_Service.getTargetAera | 语法 | Public String getTargetAera (PromotionPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回策略适用商圈信息 |
| Promotion\_Data\_Service.getTargetHotel | 语法 | Public List<HotelPO> getTargetHotel (PromotionPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回策略适用酒店信息 |
| Promotion\_Data\_Service.getTargetUser | 语法 | Public MemberType getTargetUser (PromotionPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回策略适用客户类型 |
| Promotion\_Data\_Service.getStartTime | 语法 | Public Date getStartTime (PromotionPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回策略生效时间 |
| Promotion\_Data\_Service.getEndTime | 语法 | Public Date getEndTime (PromotionPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回策略过期时间 |
| Promotion\_Data\_Service.getDiscount | 语法 | Public double getDiscount (PromotionPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回策略折扣信息 |
| Promotion\_Data\_Service.getMinRoom | 语法 | Public int getMinRoom (PromotionPO po) |
| 前置条件 | 数据文件中存在该po记录 |
| 后置条件 | 返回策略生效至少需要的房间数 |

表5.4.2-9数据层DataFactory模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| DataFactory. increaseCredit | 语法 | Public ResultMessage addCredit (CustomerPO customerPO, int increase) |
| 前置条件 | 逻辑层请求增加客户信用值 |
| 后置条件 | 在数据文件中增加一条CreditRecordPO |
| DataFactory.decreaseCredit | 语法 | Public ResultMessage decreaseCredit (CustomerPO customerPO int decrease) |
| 前置条件 | 逻辑层请求减少客户信用值 |
| 后置条件 | 在数据文件中增加一条CreditRecordPO |
| DataFactory.getCreditRecord | 语法 | Public List<CreditRecordPO> getCreditRecord (String user\_name) |
| 前置条件 | 数据文件中存在该客户名称对应的CreditRecordPO 记录 |
| 后置条件 | 返回该客户所CreditRecordPO |
| DataFactory.addCreditRecord | 语法 | public ResultMessage addCreditRecord( CreditRecordPO crPO) |
| 前置条件 | 无 |
| 后置条件 | 在数据文件中新增该信用记录 |

# 6. 信息视角

## 6.1 数据持久化对象

|  |  |
| --- | --- |
| 类名 | 包含的属性 |
| CustomerPO | 客户信息：姓名/名称，手机号，邮箱号，信用值，个人头像，身份证号，密码，会员类型. |
| MemberPO | 会员信息：身份证号，会员类型，会员等级。如果是普通会员，还有一项客户生日；如果是企业会员，还有一项企业名称。 |
| MemberLevelPO | 会员等级制度：制定者姓名，制定日期，等级个数，会员信用值界限。 |
| OrderPO | 订单信息：客户姓名/名称，手机号，身份证号，酒店ID，酒店名称，预计订单开始时间，实际订单开始时间，预计退房时间，实际退房时间，最晚订单执行时间，房间类型及数量，预计入住人数，有无儿童，备注，使用的优惠策略名称，折扣前价格，折后价格，订单状态，订单号。 |
| CreditRecordPO | 信用记录：包括信用变化值，更改时间，客户姓名/名称，客户ID，更改后信用值。如果是与订单相关的信用值变化（正常订单增加信用值，撤销订单和异常订单扣除信用值），还有一项订单号；如果是与充值相关的，还有一项营销人员姓名。 |
| ClerkPO | 酒店工作人员信息：姓名/名称，手机号，密码，身份证号，个人头像，所在酒店名称，所在酒店ID，工作人员职位。 |
| HotelPO | 酒店信息：酒店名称，酒店地址，酒店所属商圈，简介，设施服务，酒店星级，经营许可证号，酒店图片，系统中该酒店工作人员的名称和手机号，酒店ID，若干个DailyRoomInfoPO，酒店评分，若干个CommentPO。 |
| DailyRoomInfoPO | 每日房间信息：酒店ID， 日期，若干个RoomPO。 |
| RoomPO | 房间信息：酒店ID，房间类型，已入住房间数量，已预订房间数量，剩余空房数量，房间价格。 |
| CommentPO | 评价信息：评分，评价描述，评分客户姓名/名称，评分客户ID，评价对应酒店名称，评价对应酒店ID，评价对应订单号，评价时间。 |
| MarketerPO | 网站营销人员信息：姓名，身份证号，手机号，密码，个人头像，工作人员职位。 |
| PromotionPO | 促销策略信息：策略制定者姓名，制定时间，策略名称，策略适用客户，策略适用商圈，策略适用酒店，策略生效时间，策略过期时间，策略折扣，策略生效至少需要的房间数，策略编号。 |
| ManagerPO | 网站管理人员信息：姓名，身份证号，手机号，密码，个人头像，工作人员职位。 |

持久化对象如CustomerPO的定义如图6.1，更多定义见原型代码。

1. **package** po;
3. **import** java.awt.Image;
4. **import** java.io.Serializable;
5. **import** util.MemberType;
7. **public** **class** CustomerPO **implements** Serializable {
8. // 用户名
9. **private** String userName;
10. // 用户密码
11. **private** String password;
12. // 用户手机号
13. **private** String phone;
14. // 用户邮箱账号
15. **private** String email;
16. // 用户信用值
17. **private** **int** credit;
18. // 用户头像
19. **private** Image picture;
20. // 用户身份证号
21. **private** String ID;
22. // 会员类型
23. **private** MemberType memberType;
25. **public** CustomerPO() {
26. }
28. **public** CustomerPO(String userName, String password, String phone, String email, **int** credit, Image picture,
29. String ID,MemberType memberType) {
30. **this**.userName = userName;
31. **this**.password = password;
32. **this**.phone = phone;
33. **this**.email = email;
34. **this**.credit = credit;
35. **this**.picture = picture;
36. **this**.ID = ID;
37. **this**.memberType=memberType;
38. }
40. **public** String getUserName() {
41. **return** userName;
42. }
44. **public** **void** setUserName(String userName) {
45. **this**.userName = userName;
46. }
48. **public** String getPassword() {
49. **return** password;
50. }
52. **public** **void** setPassword(String password) {
53. **this**.password = password;
54. }
56. **public** String getPhone() {
57. **return** phone;
58. }
60. **public** **void** setPhone(String phone) {
61. **this**.phone = phone;
62. }
64. **public** String getEmail() {
65. **return** email;
66. }
68. **public** **void** setEmail(String email) {
69. **this**.email = email;
70. }
72. **public** **int** getCredit() {
73. **return** credit;
74. }
76. **public** **void** setCredit(**int** credit) {
77. **this**.credit = credit;
78. }
80. **public** Image getPicture() {
81. **return** picture;
82. }
84. **public** **void** setPicture(Image picture) {
85. **this**.picture = picture;
86. }
88. **public** String getID() {
89. **return** ID;
90. }
92. **public** MemberType getMemberType(){
93. **return** memberType;
94. }
96. **public** **void** setMemberType(MemberType memberType){
97. **this**.memberType=memberType;
98. }
99. }

图6.1 持久化对象CustomerPO的定义

## 6.2 数据库表

|  |  |
| --- | --- |
| 表名 | 内容 |
| Customer | 使用系统的客户信息 |
| Customer\_Credit | 客户的信用记录 |
| Customer\_Hotel | 客戶已經預定過的酒店 |
| Clerk | 使用系统的酒店工作人员信息 |
| Marketer | 使用系统的网站营销人员信息 |
| Manager | 使用系统的网站管理人员信息 |
| Order | 订单 |
| Hotel | 酒店 |
| Hotel\_Room | 酒店房间信息 |
| Member | 会员 |
| MemberLevel | 会员等级制度 |
| Promotion | 营销策略 |