|  |
| --- |
| 互联网酒店管理系统  Hotel Property Management System |
| 软件体系结构描述文档 |
| V1.4 草稿 |

|  |
| --- |
| 焦立群、金灵益、黎奥邦、李恩铭  2016-10-11 |

更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 焦立群 | 2016-10-11 | 最初文档模板 | V1.0 |
| 焦立群 | 2016-10-11 | 添加模块职责，用户界面层的分解，memberbl模块的接口规范 | V1.1 |
| 李恩铭 | 2016-10-12 | 添加逻辑视角，组合视角，业务逻辑层accountbl模块的接口规范 | V1.2 |
| 金灵益 | 2016-10-12 | 添加数据层的分解，信息视角，业务逻辑层hotelbl模块的接口规范 | V1.3 |
| 焦立群 | 2016-10-12 | 修改了文档的排版 | V1.4 |

目录

[更新历史 1](#_Toc464077648)

[1. 引言 3](#_Toc464077649)

[1.1 编制目的 3](#_Toc464077650)

[1.2 词汇表 3](#_Toc464077651)

[1.3 参考资料 3](#_Toc464077652)

[2. 产品概述 3](#_Toc464077653)

[3. 逻辑视角 3](#_Toc464077654)

[4. 组合视角 5](#_Toc464077655)

[4.1 开发包图 5](#_Toc464077656)

[4.2 运行时进程 8](#_Toc464077657)

[4.3 物理部署 9](#_Toc464077658)

[5. 接口视角 9](#_Toc464077659)

[5.1 模块的职责 9](#_Toc464077660)

[5.2 用户界面层的分解 11](#_Toc464077661)

[5.3 业务逻辑层的分解 14](#_Toc464077662)

[5.4 数据层的分解 16](#_Toc464077663)

[6. 信息视角 21](#_Toc464077664)

[6.1 数据持久化对象 21](#_Toc464077665)

[6.2 Txt持久化格式 21](#_Toc464077666)

[6.3 数据库表 21](#_Toc464077667)

1. 引言
   1. 编制目的

本报告详细完成对互联网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编写，是了解系统的导航。

* 1. 词汇表

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| HPMS | 互联网酒店管理系统 | Hotel Property Management System |

* 1. 参考资料

互联网酒店管理系统HPMS软件需求规格说明\_V1.0正式版

互联网酒店管理系统HPMS用例文档\_V1.1正式版

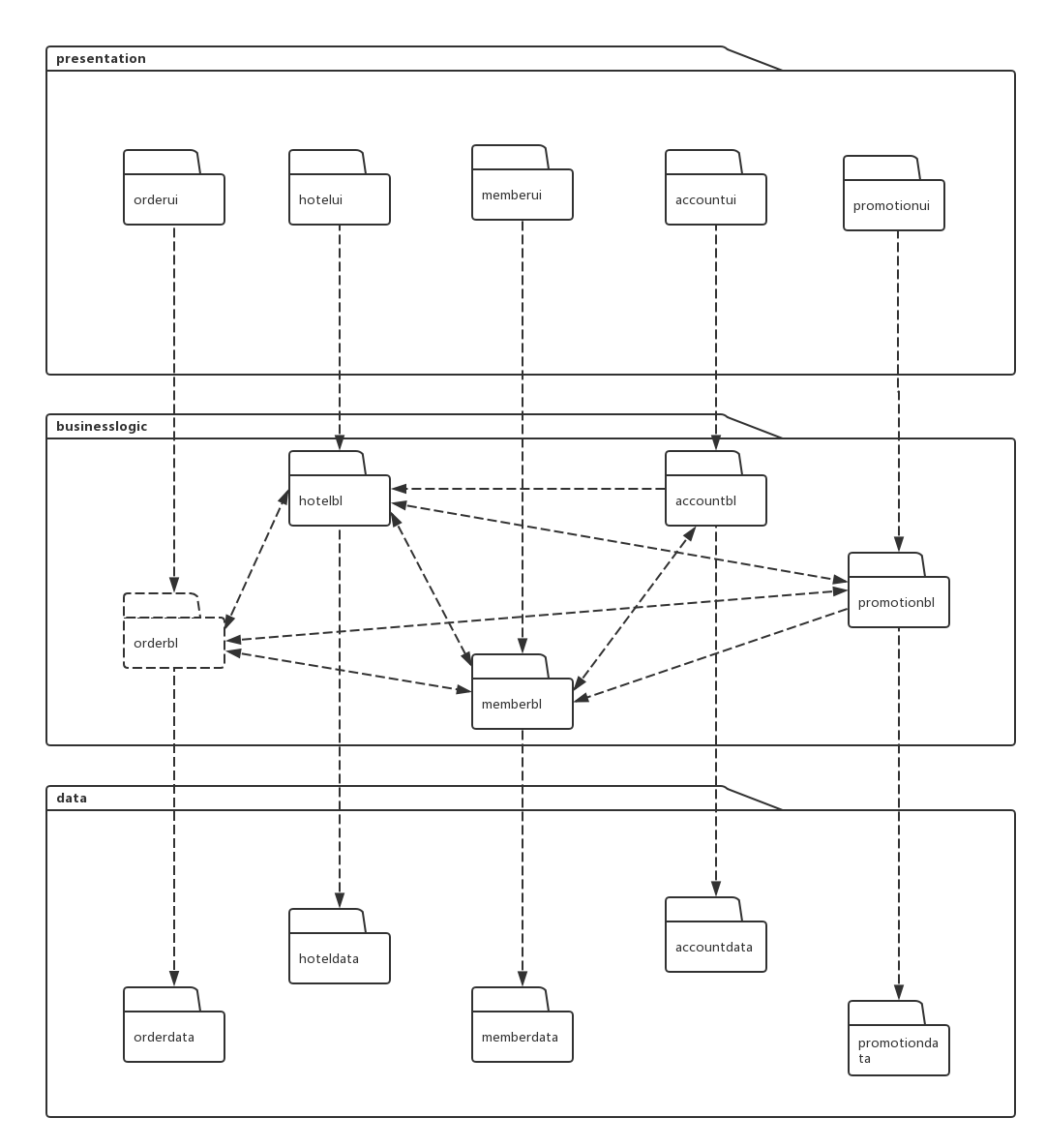
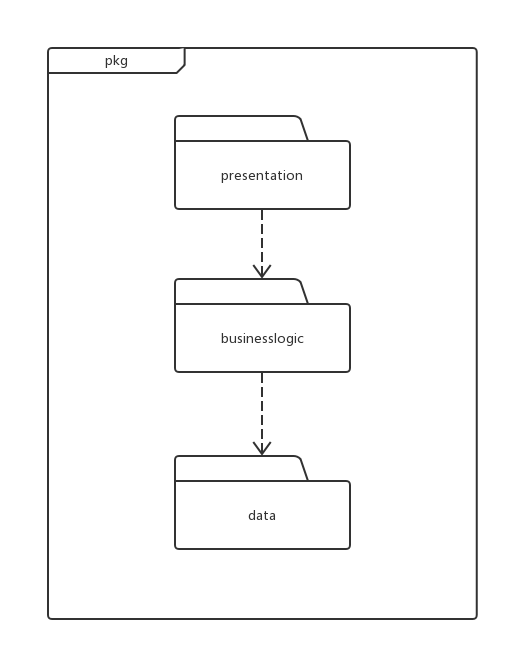
1. 产品概述

参考互联网酒店管理系统用例文档和互联网酒店管理系统软件需求规格说明中对产品的概括描述。

1. 逻辑视角

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

**图 1参照体系结构风格的包图表达逻辑视角**



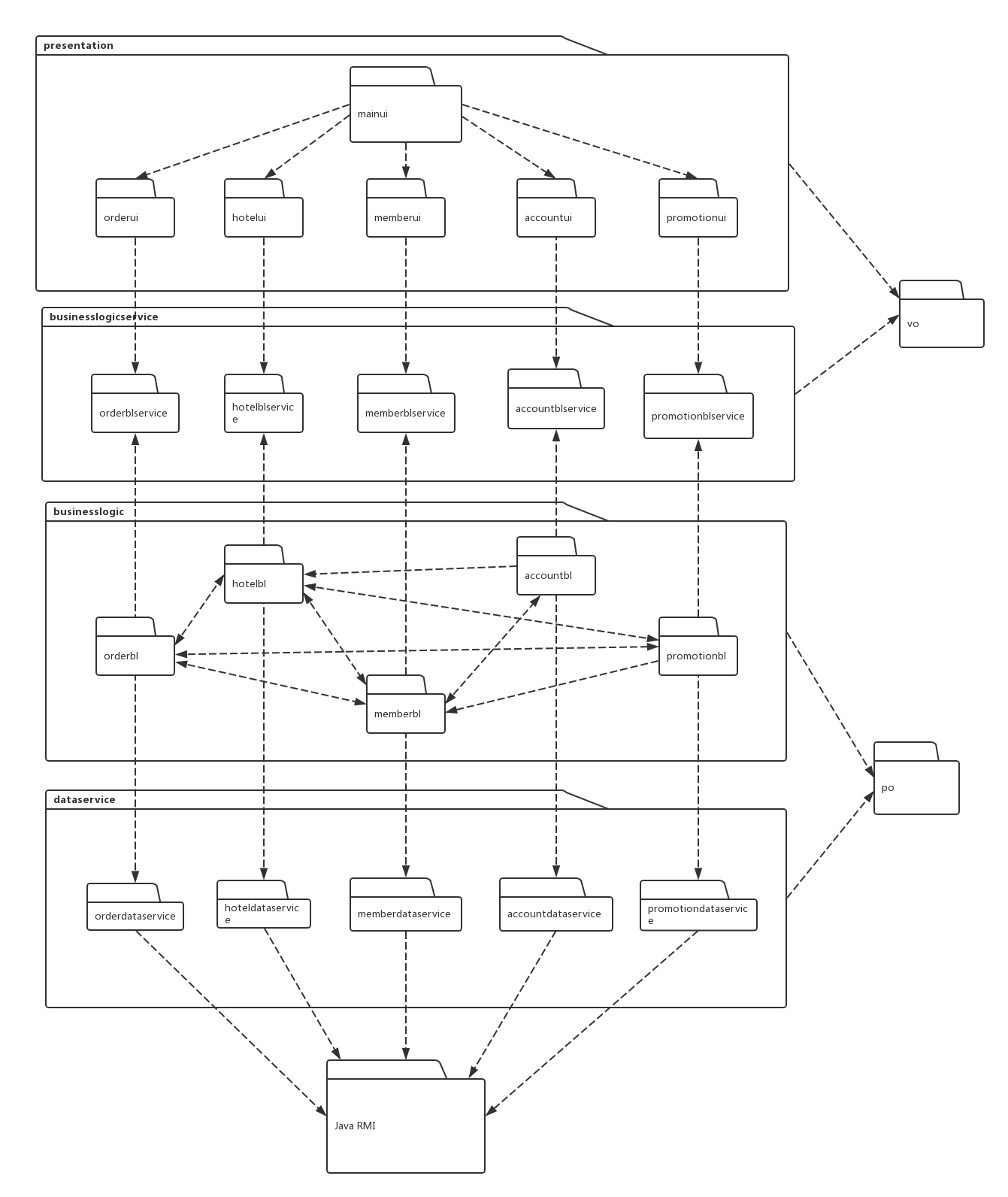
**图2 软件体系结构逻辑设计方案**

1. 组合视角
   1. 开发包图

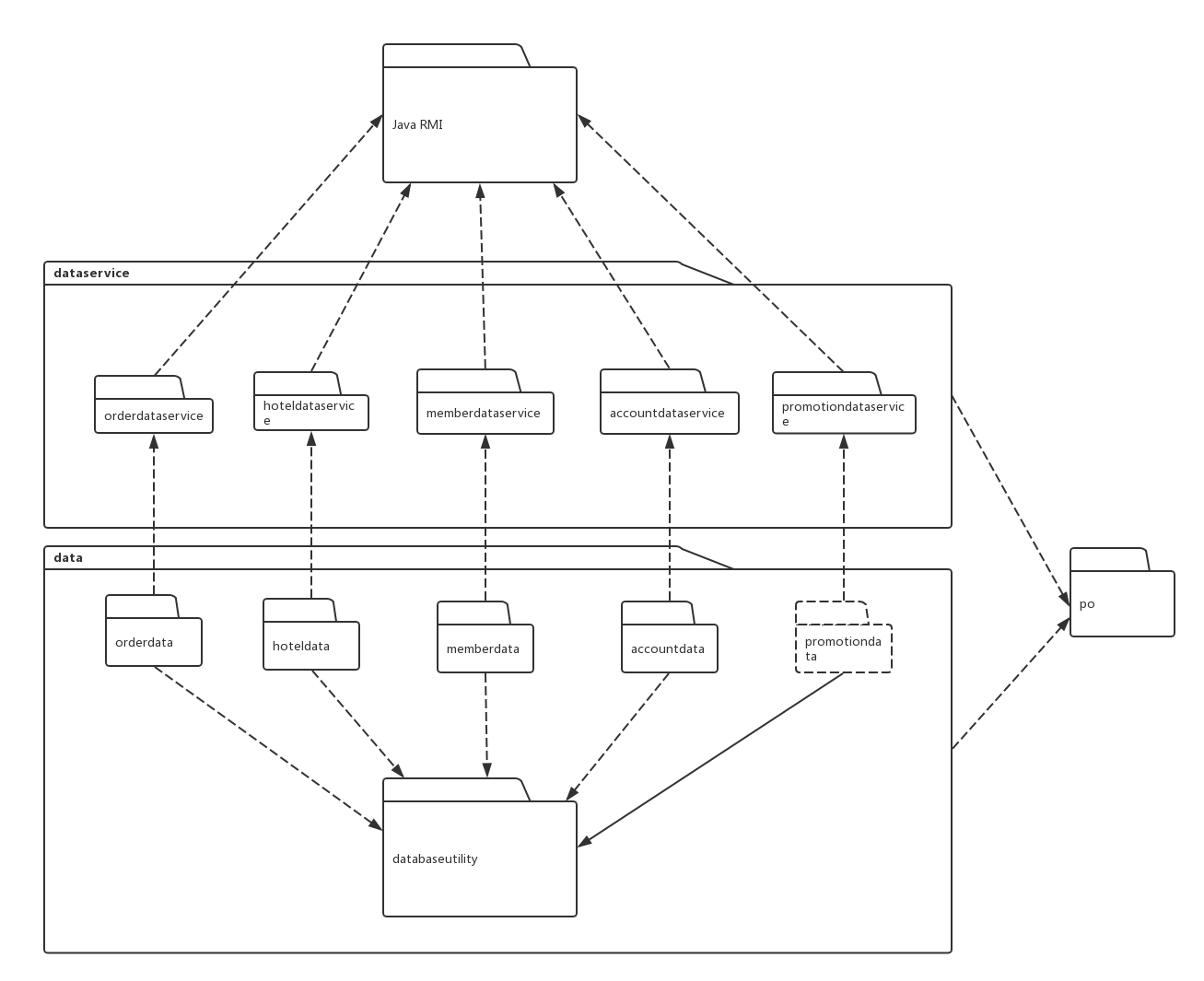
互联网酒店管理系统的最终开发包设计如表1所示

**表1 互联网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** | orderui, hotelui, memberui, accountui, promotionui |
| **orderui** | orderblservice, 界面类库包 |
| **orderblservice** |  |
| **orderbl** | orderblservice, hotelbl, memberbl, promotionbl, po |
| **orderdataservice** | Java RMI, po |
| **orderdata** | orderdataservice, databaseutility |
| **hotelui** | hotelblservice, 界面类库包 |
| **hotelblservice** |  |
| **hotelbl** | hotelblservice, orderbl, memberbl, promotionbl, po |
| **hoteldataservice** | Java RMI, po |
| **hoteldata** | hoteldataservice, databaseutility |
| **memberui** | memberblservice, 界面类库包 |
| **memberblservice** |  |
| **memberbl** | memberblservice, accountbl, orderbl, hotelbl, po |
| **memberdataservice** | Java RMI, po |
| **memberdata** | memberdataservice, databaseutility |
| **accountui** | accountblservice, 界面类库包 |
| **accountblservice** |  |
| **accountbl** | accountblservice, memberbl, hotelbl, po |
| **accountdataservice** | Java RMI, po |
| **accountdata** | accountdataservice, databaseutility |
| **promotionui** | promotionblservice, 界面类库包 |
| **promotionblservice** |  |
| **promotionbl** | promotionblservice, orderbl, hotelbl, po |
| **promotiondataservice** | Java RMI, po |
| **promotiondata** | promotiondataservice, databaseutility |
| **vo** |  |
| **po** |  |
| **utilitybl** |  |
| **界面类库包** |  |
| **Java RMI** |  |
| **database utility** |  |



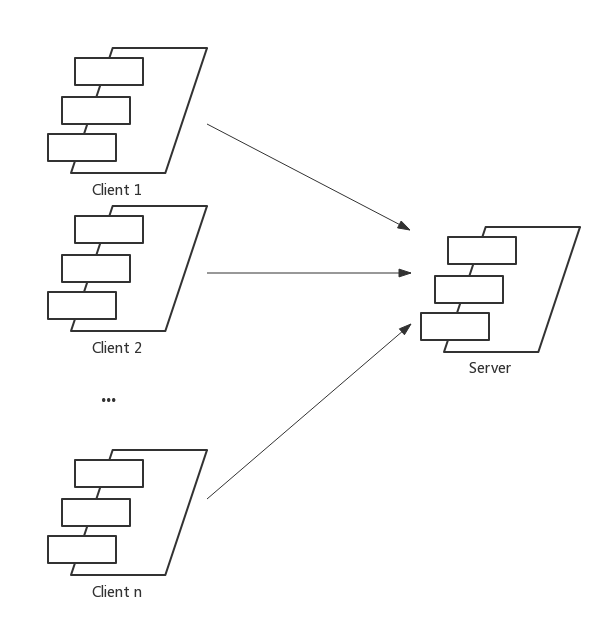
**图 3 客户端开发包图**



**图 4 服务器端开发包图**

* 1. 运行时进程

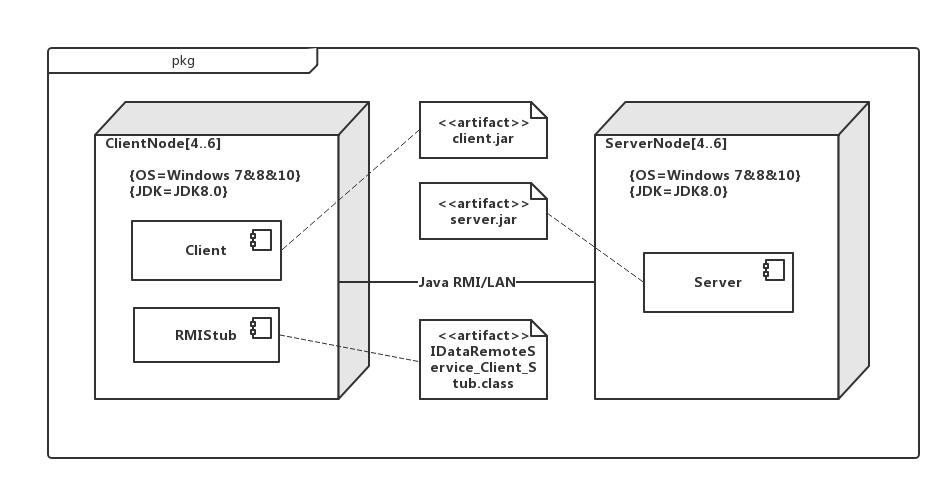
在互联网酒店管理系统中，会有多个客户端进程和一个服务器端进程，其进程图如图5所示。结合部署图，客户端进程是在客户端机器上运行，服务器端进程是在服务器端机器上运行。



**图 进程图**

* 1. 物理部署

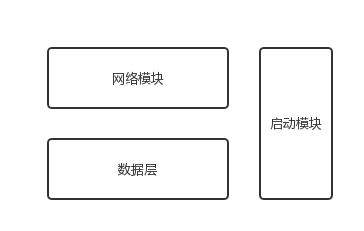
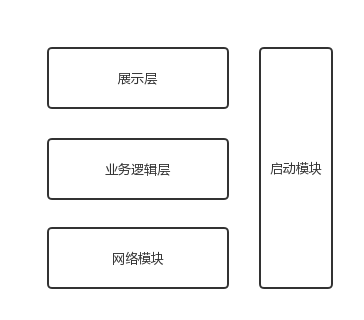
互联网酒店管理系统中客户端构件是放在客户端机上，服务器端构件是放在服务器端机器上。在客户端节点上，还要部署RMIStub构件。由于Java RMI 构件属于JDK 1.8的一部分。所以，在系统JDK环境已经设置好的情况下，不需要独立部署。部署图如图6所示。



**图 6 部署图**

1. 接口视角
   1. 模块的职责

客户端模块和服务器端模块视图分别如图和图所示。客户端各层和服务器端各层的职责分别如表和表所示。



**图 客户端模块视图 图 服务器端模块视图**

**表 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 | 基于窗口的互联网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进行响应并进行业务处理逻辑 |
| 客户端网络模块 | 利用Java RMI机制查找RMI服务 |

**表 服务器端各层的职责**

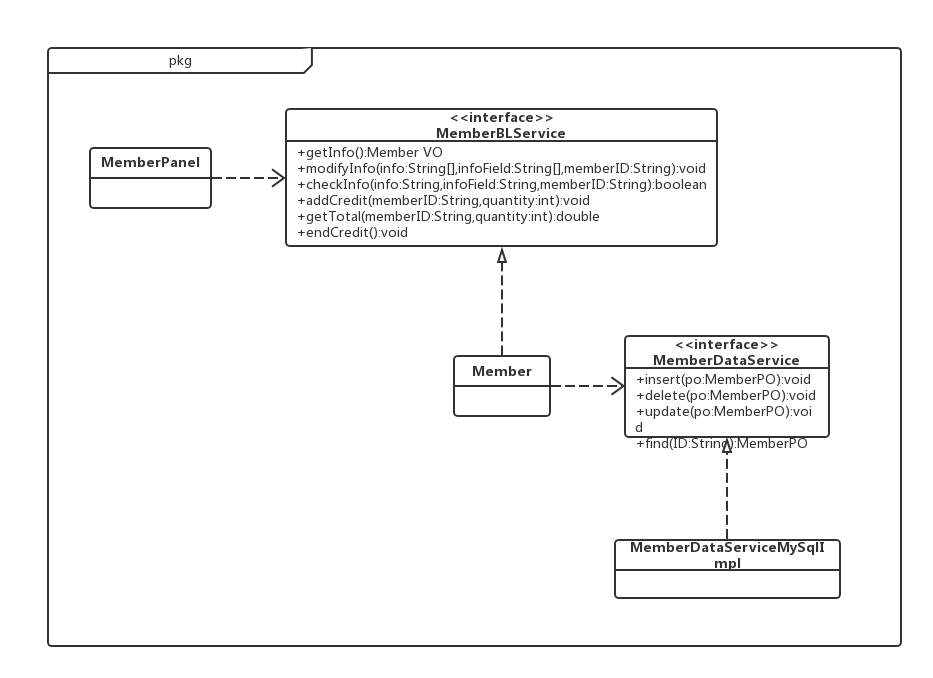
|  |  |
| --- | --- |
| 层 | 职责 |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 数据层 | 负责数据的持久化及数据访问接口 |
| 服务器端网络模块 | 利用Java RMI机制开启RMI服务，注册RMI服务 |

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所示。

**表 层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| OrderBLService  HotelBLService  MemberBLService  AccountBLService  PromotionBLService | 客户端展示层 | 客户端业务逻辑层 |
| OrderDataService  HotelDataService  MemberDataService  AccountDataService  PromotionDataService  DatabaseFactory | 客户端业务逻辑层 | 服务器端数据层 |

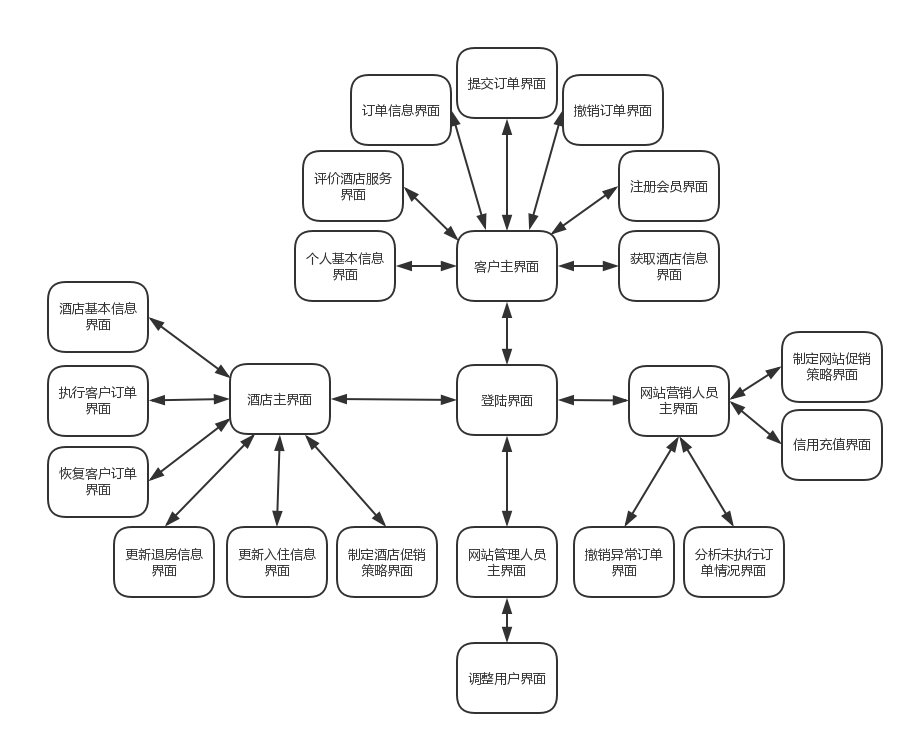
借用客户用例来说明层之间的调用，如图所示。每一层之间都是由上层依赖了一个接口（需接口），而下层实现这个接口（供接口）。MemberBLService提供了Member界面所需要的所有业务逻辑功能。MemberDataService提供了对数据库的增、删、改、查等操作。这样的实现就大大降低了层与层之间的耦合。



**图 客户用例层之间调用的接口**

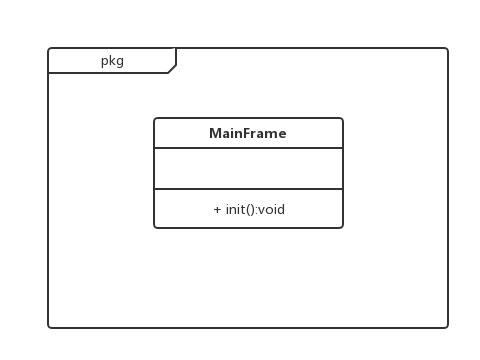
* 1. 用户界面层的分解

根据需求，系统存在23个用户界面：登陆界面、客户主界面、酒店主界面、网站营销人员主界面、网站管理人员主界面、个人基本信息界面、评价酒店服务界面、订单信息界面、提交订单界面、撤销订单界面、注册会员界面、获取酒店信息界面、酒店基本信息、执行客户订单界面、恢复客户订单界面、更新退房信息界面、更新入住信息界面、制定酒店促销策略界面、制定网站促销策略界面、信用充值界面、撤销异常订单界面、分析未执行订单情况界面、调整用户界面。界面跳转如图所示。



**图 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。用户界面类如图所示。



**图 用户界面类**

* + 1. **用户界面层模块的职责**

如表所示为用户界面层模块的职责

**表 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 |

* + 1. **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所示。

**表 用户界面层模块的接口规范**

|  |  |  |
| --- | --- | --- |
| MainFrame | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示Frame以及LoginPanel |

用户界面层需要的服务接口如表所示。

**表 用户界面层模块需要的服务接口**

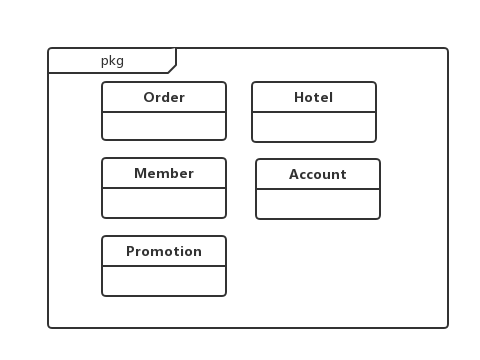
|  |  |
| --- | --- |
| 服务名 | 服务 |
| businesslogicservice.LoginBLService | 登陆界面的业务逻辑接口 |
| Businessloginservice.\*BLService | 每个界面都有一个相应的业务逻辑接口 |

* + 1. **用户界面模块设计原理**

用户界面利用JavaFX的API来实现

* 1. 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，Order对象负责处理订单界面的业务逻辑。业务逻辑层的设计如图所示。



**图 业务逻辑层的设计**

* + 1. **业务逻辑层模块的职责**

业务逻辑层模块职责如表所示

**表 业务逻辑层模块的职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| orderbl |  |
| hotelbl |  |
| memberbl |  |
| accountbl |  |
| promotionbl |  |

* + 1. **业务逻辑层模块的接口规范**

orderbl、hotelbl、memberbl、accountbl和promotionbl模块的接口规范分别如表、表、表、表和表所示

**表 memberbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Member.modifyInfo | 语法 | Public ResultMessage modifyInfo(String[] info, String[] infoField, String memberID) | |
| 前置条件 | info符合输入规则 | |
| 后置条件 | 修改用户信息，持久化更新涉及的领域对象的数据 | |
| Member.checkInfo | 语法 | Public ResultMessage checkInfo(String Info, String InfoField, String memberID) | |
| 前置条件 | Info被修改或特定infoField为空 | |
| 后置条件 | 返回检查结果 | |
| Member.addCredit | 语法 | Public ResultMessage addCredit(String memberID, int quantity) | |
| 前置条件 | 启动一个信用充值回合 | |
| 后置条件 | 在一个信用充值回合中，增加信用充值的数量 | |
| Member.getTotal | 语法 | Public ResultMessage getTotal(String memberID, int quantity) | |
| 前置条件 | 已添加充值者信息、充值信用值数量 | |
| 后置条件 | 返回此信用充值回合中需要支付的总额 | |
| Member.endCredit | 语法 | Public void endCredit() | |
| 前置条件 | 已支付 | |
| 后置条件 | 结束此次信用充值回合，持久化更新涉及的领域对象的数据 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | **服务** |
| MemberDataService.find(String ID) | | | 根据ID进行查找单一持久化对象 |
| MemberDataService.insert(MemberPO po) | | | 插入单一持久化对象 |
| MemberDataService.delete(MemberPO po) | | | 删除单一持久化对象 |
| MemberDataService.update(MemberPO po) | | | 更新单一持久化对象 |
| DataBaseFactory.getMemberDatabase | | | 得到Member数据库的服务的引用 |

**表 accountbl模块的接口规范**

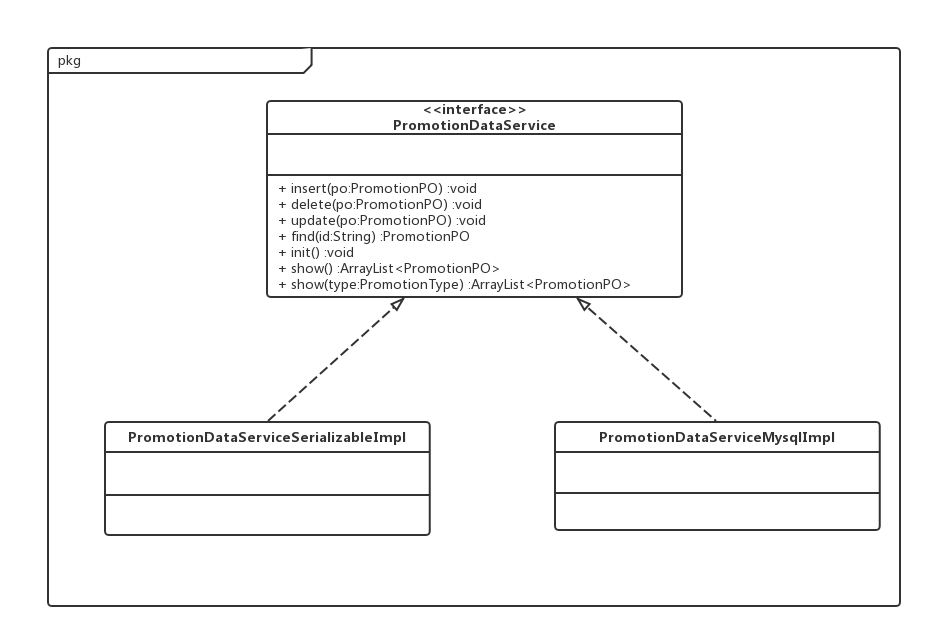
|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Account.register | **语法** | public boolean tryRegister(String name,String password); | |
| **前置条件** | name和password符合输入规则 | |
| **后置条件** | 查找name是否已存在，据此返回注册结果 | |
| Account.login | **语法** | public boolean tryLogin(String name,String password); | |
| **前置条件** | name和password符合输入规则，相应的account未登录 | |
| **后置条件** | 查找是否存在相应的account，根据输入的password返回登录验证的结果 | |
| Account.logout | **语法** | public boolean tryLogout(String name); | |
| **前置条件** | 相应的account已经登录 | |
| **后置条件** | 根据相应的account是否已登录返回登出的结果 | |
| Account.modify | **语法** | Public Boolean tryModify(String name,String oldPassword,String newPassword); | |
| **前置条件** | 无 | |
| **后置条件** | 查找是否存在相应的account，根据输入的oldPassword返回修改密码的结果 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | **服务** |
| DatabaseFactory.getAccountDatabase | | | 得到Account数据库的服务的引用 |
| AccountDataService.insert(AccountPO po) | | | 在数据库中插入一个单一持久化对象 |
| AccountDataService.update(AccountPO po) | | | 在数据库中更新一个单一持久化对象 |
| AccountDataService.find(String name) | | | 根据用户名查找单一持久化对象 |

**表 hotelbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotel.addHotel | **语法** | public ResultMessage addHotel(String name, String address, String business, String tel, String info) |
| **前置条件** | 名称、地址、商圈、联系方式、简介输入合法 |
| **后置条件** | 添加新的酒店，持久化更新设计的对象数据 |
| Hotel.deleteHotel | **语法** | public ResultMessage deleteHotel(String id) |
| **前置条件** | 编号输入合法 |
| **后置条件** | 删除酒店，持久化更新涉及的对象数据 |
| Hotel.updateHotel | **语法** | public ResultMessage updateHotel(String name, String address, String business, String tel, String info) |
| **前置条件** | 名称、地址、商圈、联系方式、简介输入合法 |
| **后置条件** | 酒店更新，持久化更新设计的对象数据 |
| Hotel.findHotel | **语法** | public ResultMessage findHotel(String id) |
| **前置条件** | 编号输入合法 |
| **后置条件** | 查找到该酒店 |
| Hotel.getReadyRoom | **语法** | public int getReadyRoom() |
| **前置条件** | 已计算空余的酒店房间数量 |
| **后置条件** | 返回空余房间数量 |
| Hotel.getPoints | **语法** | public double getPoints(double point) |
| **前置条件** | 客户已进行酒店评价 |
| **后置条件** | 计算并返回酒店的评分 |
| 需要的服务（需接口） | | |
| 服务名 | | **服务** |
| DatabaseFactory.getHotelDatabase | | 得到Hotel数据库的服务的引用 |
| HotelDataService.insert(HotelPO po) | | 在数据库中插入持久化数据 |
| HotelDataService.find(String id) | | 按ID进行查找返回相应持久化数据 |
| HotelDataService.update(HotelPO po) | | 在数据库中更新一个持久化对象 |
| HotelDataService.delete(String id) | | 在数据库中删除一个持久化对象 |
| HotelDataService.getReadyRoom() | | 返回酒店空余房间数 |
| HotelDataService.getBusiness() | | 返回酒店所属商圈 |

* 1. 数据层的分解

数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增、删、改、查。数据层模块的描述具体如下图所示。



**5.4.1．数据层模块的职责**

数据层模块的职责如下表所示：

**表 数据层模块职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| OrderDataService | 提供订单数据的集体载入、保存、增加、查找、修改服务 |
| HotelDataService | 提供酒店数据的集体载入、保存、增加、查找、修改、删除服务 |
| MemberDataService | 提供客户数据集体载入、保存、增加、查找、修改、删除服务 |
| AccountDataService | 提供账号数据查找服务 |
| PromotionDataService | 提供促销策略数据的集体载入、保存、增加、查找、修改、删除服务 |

**5.4.2．数据层模块的接口规范**

数据层模块的接口规范如下表

**数据层模块的接口规范**

**OrderDataService类接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderDataService.insert | **语法** | public void insert(OrderPO po) |
| **前置条件** | 同样ID的po在数据库中不存在 |
| **后置条件** | 在数据库中增加一个po记录 |
| OrderDataService.find | **语法** | public OrderPO find(String id) |
| **前置条件** | 无 |
| **后置条件** | 按ID进行查找返回相应的OrderPO结果 |
| OrderDataService.update | **语法** | public void update(OrderPO po) |
| **前置条件** | 在数据库中存在同样ID的po |
| **后置条件** | 更新一个po |
| OrderDataService.init | **语法** | public void init() |
| **前置条件** | 无 |
| **后置条件** | 初始化持久化数据库 |
| OrderDataService.show | **语法** | public ArrayList<OrderPO> show  (String memberID) |
| **前置条件** | 无 |
| **后置条件** | 根据客户编号返回所有订单数据 |
| OrderDataService.getFinished | **语法** | public ArrayList<OrderPO> getFinished() |
| **前置条件** | 无 |
| **后置条件** | 返回已执行订单数据 |
| OrderDataService.getFinished | **语法** | public ArrayList<OrderPO> getFinished  (String memberID) |
| **前置条件** | 无 |
| **后置条件** | 根据客户编号返回已执行订单数据 |
| OrderDataService.getSubmitted | **语法** | public ArrayList<OrderPO> getSubmitted() |
| **前置条件** | 无 |
| **后置条件** | 返回未执行订单数据 |
| OrderDataService.getSubmitted | **语法** | public ArrayList<OrderPO> getSubmitted  (String memberID) |
| **前置条件** | 无 |
| **后置条件** | 根据客户编号返回未执行订单数据 |
| OrderDataService.getFailed | **语法** | public ArrayList<OrderPO> getFailed() |
| **前置条件** | 无 |
| **后置条件** | 返回异常订单数据 |
| OrderDataService.getFailed | **语法** | public ArrayList<OrderPO> getFailed  (String memberID) |
| **前置条件** | 无 |
| **后置条件** | 根据客户编号返回异常订单数据 |
| OrderDataService.getCanceled | **语法** | public ArrayList<OrderPO> getCanceled  (String memberID) |
| **前置条件** | 无 |
| **后置条件** | 根据客户编号返回已撤销订单数据 |

**HotelDataService接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelDataService.insert | **语法** | public void insert(HotelPO po) |
| **前置条件** | 同样ID的po在数据库中不存在 |
| **后置条件** | 在数据库中增加一个po记录 |
| HotelDataService.find | **语法** | public HotelPO find(String id) |
| **前置条件** | 无 |
| **后置条件** | 按ID进行查找返回相应的HotelPO结果 |
| HotelDataService.update | **语法** | public void update(HotelPO po) |
| **前置条件** | 在数据库中存在同样ID的po |
| **后置条件** | 更新一个po |
| HotelDataService.init | **语法** | public void init() |
| **前置条件** | 无 |
| **后置条件** | 初始化持久化数据库 |
| HotelDataService.delete | **语法** | public void delete (String id) |
| **前置条件** | 同样的po在数据库中存在 |
| **后置条件** | 删除po |
| HotelDataService.show | **语法** | public ArrayList<HotelPO> show() |
| **前置条件** | 无 |
| **后置条件** | 返回所有酒店数据 |
| HotelDataService.getName | **语法** | public String getName() |
| **前置条件** | 无 |
| **后置条件** | 返回酒店名称 |
| HotelDataService.getAddress | **语法** | public String getAddress() |
| **前置条件** | 无 |
| **后置条件** | 返回酒店地址 |
| HotelDataService.getBusiness | **语法** | public String getBusiness() |
| **前置条件** | 无 |
| **后置条件** | 返回酒店所属商圈 |
| HotelDataService.getPoint | **语法** | public double getPoint() |
| **前置条件** | 无 |
| **后置条件** | 返回酒店的评分 |
| HotelDataService.getReadyRoom | **语法** | public int getReadyRoom() |
| **前置条件** | 无 |
| **后置条件** | 返回酒店空余房间数 |

**MemberDataService接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| MemberDataService.insert | **语法** | public void insert(MemberPO po) |
| **前置条件** | 同样ID的po在数据库中不存在 |
| **后置条件** | 在数据库中增加一个po记录 |
| MemberDataService.find | **语法** | public MemberPO find(String id) |
| **前置条件** | 无 |
| **后置条件** | 按ID进行查找返回相应的MemberPO结果 |
| MemberDataService.update | **语法** | public void update(MemberPO po) |
| **前置条件** | 在数据库中存在同样ID的po |
| **后置条件** | 更新一个po |
| MemberDataService.init | **语法** | public void init() |
| **前置条件** | 无 |
| **后置条件** | 初始化持久化数据库 |
| MemberDataService.show | **语法** | public ArrayList<MemberPO> show() |
| **前置条件** | 无 |
| **后置条件** | 返回所有客户数据 |
| MemberDataService.getID | **语法** | public String getID() |
| **前置条件** | 无 |
| **后置条件** | 返回客户的编号 |
| MemberDataService.getUsername | **语法** | public String getUsername() |
| **前置条件** | 无 |
| **后置条件** | 返回客户的用户名 |
| MemberDataService.getTel | **语法** | public String getTel() |
| **前置条件** | 无 |
| **后置条件** | 返回客户的联系方式 |
| MemberDataService.getCredit | **语法** | public long getCredit() |
| **前置条件** | 无 |
| **后置条件** | 返回客户的信用值 |

**AccountDataService接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AccountDataService.find | **语法** | public AccountPO find(String id) |
| **前置条件** | 无 |
| **后置条件** | 按ID进行查找返回相应的AccountPO结果 |
| AccountDataService.init | **语法** | public void init() |
| **前置条件** | 无 |
| **后置条件** | 初始化持久化数据库 |

**PromotionDataService接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionDataService.insert | **语法** | public void insert(PromotionPO po) |
| **前置条件** | 同样ID的po在数据库中不存在 |
| **后置条件** | 在数据库中增加一个po记录 |
| PromotionDataService.find | **语法** | public PromotionPO find(String id) |
| **前置条件** | 无 |
| **后置条件** | 按ID进行查找返回相应的PromotionPO结果 |
| PromotionDataService.update | **语法** | public void update(PromotionPO po) |
| **前置条件** | 在数据库中存在同样ID的po |
| **后置条件** | 更新一个po |
| PromotionDataService.init | **语法** | public void init() |
| **前置条件** | 无 |
| **后置条件** | 初始化持久化数据库 |
| PromotionDataService.show | **语法** | public ArrayList<PromotionPO> show() |
| **前置条件** | 无 |
| **后置条件** | 得到所有促销策略数据 |
| PromotionDataService.show | **语法** | public ArrayList<PromotionPO> show  (PromotionType type) |
| **前置条件** | 无 |
| **后置条件** | 根据类型返回特定的促销策略数据 |

1. 信息视角
   1. 数据持久化对象

系统的PO类就是对应的相关的实体类，如下所示。

* OrderPO类包括订单的编号（订单号为13位0~9的数字，前8位为当前日期，后5位由系统随机生成，日期格式为：yyyy-mm-dd，因此一天最多可以生成100,000条订单），开始时间，退房时间，最晚订单执行时间，房间类型，房间数量，预计入住人数，儿童数，订单状态（包括已执行、未执行、已撤销、异常），价格（大于等于0、精确到小数点后2位的浮点数），折扣，折扣后总额
* HotelPO类包括酒店名称，地址，所属商圈，联系方式，简介，评分，评价，空余房间数，房间总数
* MemberPO类包括客户的编号（格式为一个字母加5位0~9数字，普通客户以“N”开头，企业客户以“E”开头），用户名，密码，分类（普通客户、企业客户），联系方式，信用值，企业名称，生日（日期格式为：yyyy-mm-dd）
* AccountPO类包括用户的编号（用户账号统一采用一个字母加5位0~9数字格式，网站管理人员账号以“A”开头，网站营销人员账号以“M”开头，酒店以“H”开头，普通会员账号以“N”开头，企业会员账号以“E”开头），密码
* PromotionPO类是HotelPromotionPO和MarketPromotionPO的父类
* HotelPromotionPO类包括促销类型，开始日期，结束日期（日期格式为：yyyy-mm-dd），活动名称，活动折扣；生日折扣；房间数，房间数折扣；企业折扣
* MarketPromotionPO类包括促销类型，开始日期，结束日期（日期格式为：yyyy-mm-dd），活动名称，活动折扣，活动简介；商圈名称，商圈等级，商圈折扣；会员等级，每次升级所需的信用值，每个等级对应的折扣
  1. Txt持久化格式

统一用\*.ser格式

* 1. 数据库表

数据库中包含Order表，Hotel表,Member表,Account表,HotelPromotion表,MarketPromotion表