**······**

**酒店预定系统**

**HRS（Hotel Reservation System）**

**体系结构设计文档**

**开发小组：darksky小组（NO.50）**

目录

[**更新历史** 3](#_Toc464375967)

[1.引言 5](#_Toc464375968)

[1.1编制目的 5](#_Toc464375969)

[1.2词汇表 5](#_Toc464375970)

[1.3参考资料 5](#_Toc464375971)

[2.产品概述 5](#_Toc464375972)

[3.逻辑视角 6](#_Toc464375973)

[4.组合视图 7](#_Toc464375974)

[4.1开发包图 7](#_Toc464375975)

[4.2运行时进程 12](#_Toc464375976)

[4.3物理部署 13](#_Toc464375977)

[5.接口视角 14](#_Toc464375978)

[5.1模块的职责 14](#_Toc464375979)

[5.2用户界面层的分解 17](#_Toc464375980)

[5.2.1用户界面模块的职责 18](#_Toc464375981)

[5.2.2用户界面模块的接口规范 18](#_Toc464375982)

[5.2.3用户界面模块的设计原理 19](#_Toc464375983)

[5.3业务逻辑层的分解 19](#_Toc464375984)

[5.3.1业务逻辑层模块的职责 19](#_Toc464375985)

[5.3.2业务逻辑层模块的接口规范 20](#_Toc464375986)

[5.4数据层的分解 27](#_Toc464375987)

[5.4.1数据层模块的职责 31](#_Toc464375988)

[5.4.2数据层模块的接口规范 34](#_Toc464375989)

[6.信息视角 43](#_Toc464375990)

[6.1数据持久化对象 43](#_Toc464375991)

[6.2TXT持久化格式 43](#_Toc464375992)

[6.3数据库表 43](#_Toc464375993)

**更新历史**

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 共同 | 2016-10-15 | 制定初稿 | V1.0 草稿 |
|  |  |  |  |
|  |  |  |  |

# 1.引言

## 1.1编制目的

## 1.2词汇表

|  |  |  |
| --- | --- | --- |
| **词汇名称** | **词汇含义** | **备注** |
| HRS | 酒店预定系统 |  |
|  |  |  |

## 1.3参考资料

《酒店预定系统用例文档》、《酒店预订系统软件需求规格说明》

# 2.产品概述

参考《酒店预定系统用例文档》和《酒店预订系统软件需求规格说明》中对产品的概括性描述。

# 3.逻辑视角

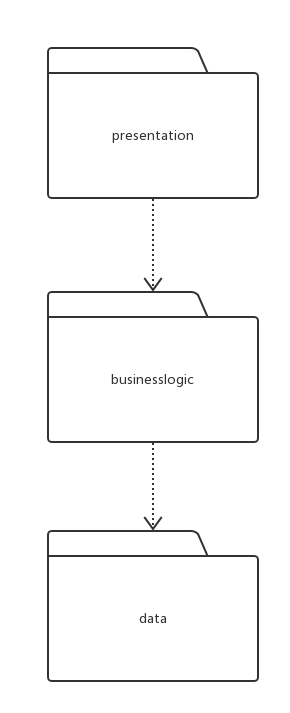
 在酒店预定系统中，选择了分层风格，将系统分为三层（展示层、业务逻辑层、数据层）能够很好地示意整个高层抽象。展示层包括GUI页面的实现，业务逻辑层包含业务逻辑处理的实现，数据层负责数据的持久化和访问。分层体系结构的逻辑视角和逻辑设计方案如图1和图2所示。

图1 参照体系结构风格的包图表达逻辑视角

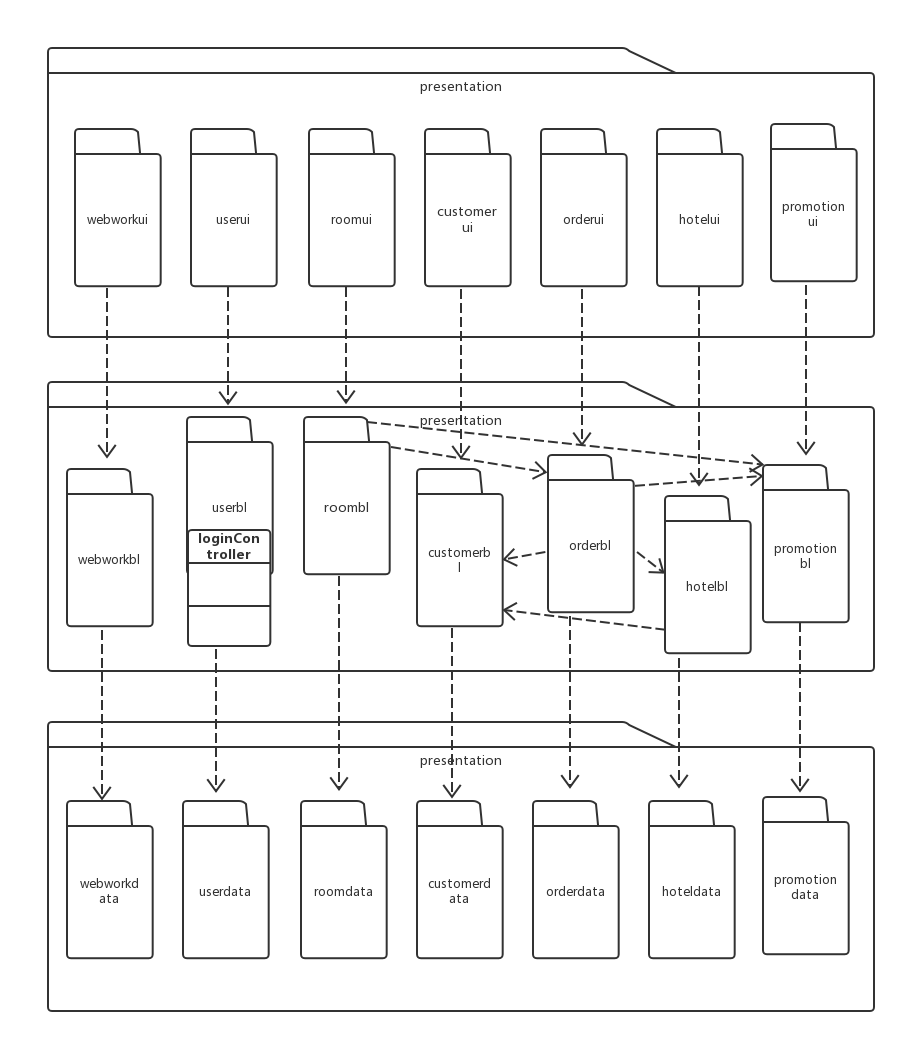


图2 软件体系结构逻辑设计方案

# 4.组合视图

## 4.1开发包图

酒店预定系统的开发包设计如表1所示。

**表1 酒店预订系统的开发包设计**

|  |  |
| --- | --- |
| mianui | Userui,customerui,hotelui,roomui,promotionui,orderui,webworkerui,vo |
| customerui | 界面类库包,customerblservice,vo |
| customerbl | Customerblservice,customerdataservice,po |
| customerblservice |  |
| customerdata | Customerdataservice,databaseseutility,po,java RMI |
| customerdataservice | Java RMI,po |
| hotelui | 界面类库包,hotelblservice,vo |
| hotelbl | Hotelblservice,hoteldataservice,po,customerbl |
| hotelblservice |  |
| hoteldata | Hoteldataservice,databaseseutility,java RMI,po |
| hoteldataservice | Java RMI,po |
| roomui | 界面类库包,roomblservice,vo |
| roombl | Roomblservice,roomdataservice,po,orderbl,promotionbl |
| roomblservice |  |
| roomdata | Databaseseutility,java RMI,po,roomdataservice |
| roomdataservice | Java RMI,po |
| promotionui | 界面类库包,vo,promotionblservice |
| promotionbl | Promotionblservice,promotiondataservice,po |
| Promotionblservice |  |
| promotiondata | Databaseseutility,java RMI,po,promotiondataservice |
| promotiondataservice | Po,java RMI |
| orderui | 界面类库包,vo,orderblservice |
| orderbl | Orderblservice,orderdataservice,po,customerbl,hotelbl,promotionbl |
| orderblservice |  |
| orderdata | Orderdataservice,databaseseutility,po,java RMI |
| orderdataservice | Java RMI,po |
| webworkerui | 界面类库包,vo,webworkerblservice |
| webworkerbl | Webworkblservice,webworkerdataservice,po |
| webworkerblservice |  |
| webworkerdata | Databaseseutility,java RMI,po,webworkerdataservice |
| webworkerdataservice | Java RMI,po |
| userui | 界面类库包,vo,userblservice |
| userbl | UserInterface,UserDataClient,UserPo |
| userblservice |  |
| userdata | Databaseseutility,java RMI,po,userdataservie |
| userdataservice | Java RMI,po |
| vo |  |
| po |  |
| utilitybl |  |
| 界面类库包 |  |
| Java RMI |  |
| databaseutility | JDBC |

酒店预订系统的客户端开发包图如图3所示，服务器端开发包图如图4所示。

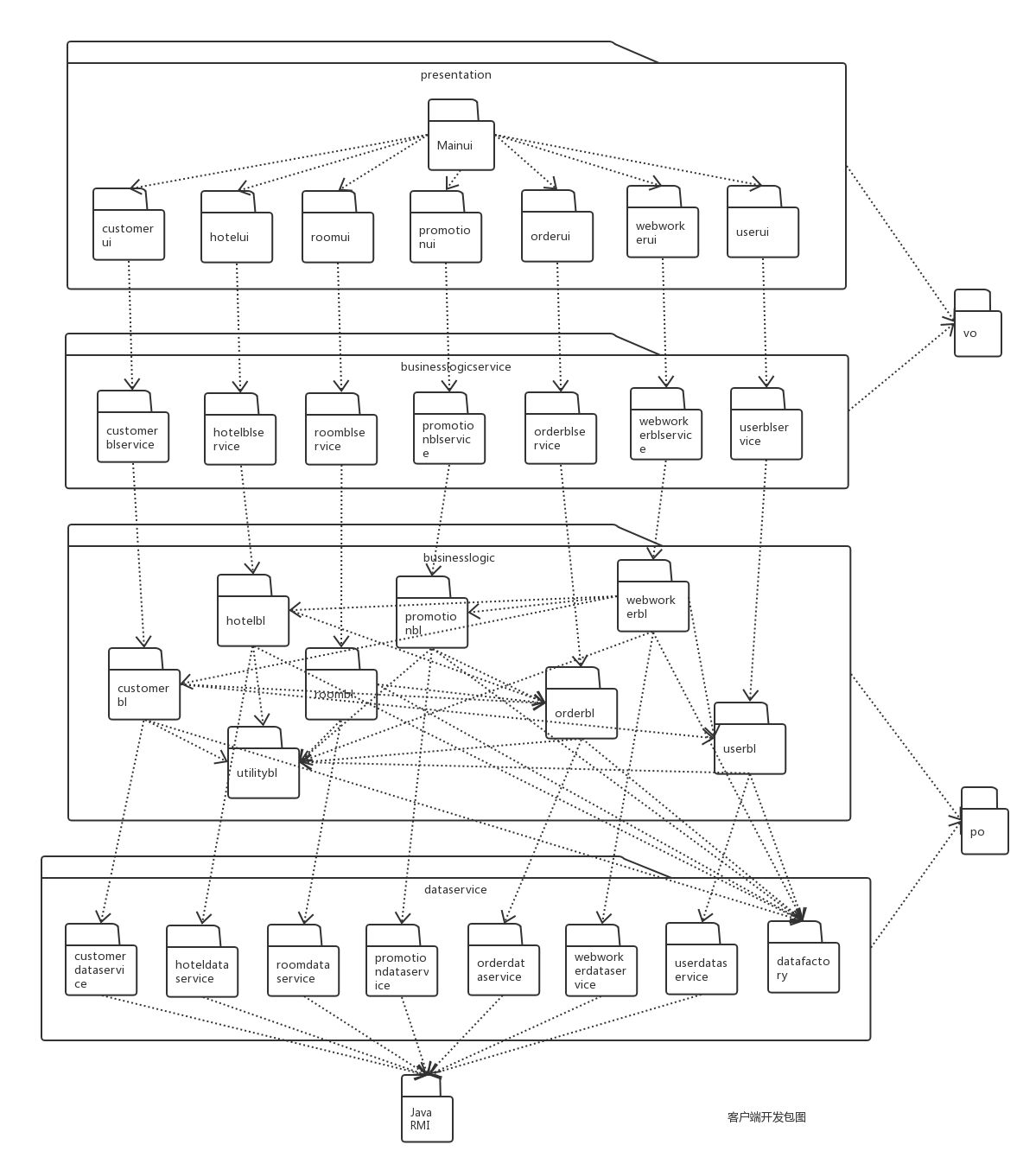


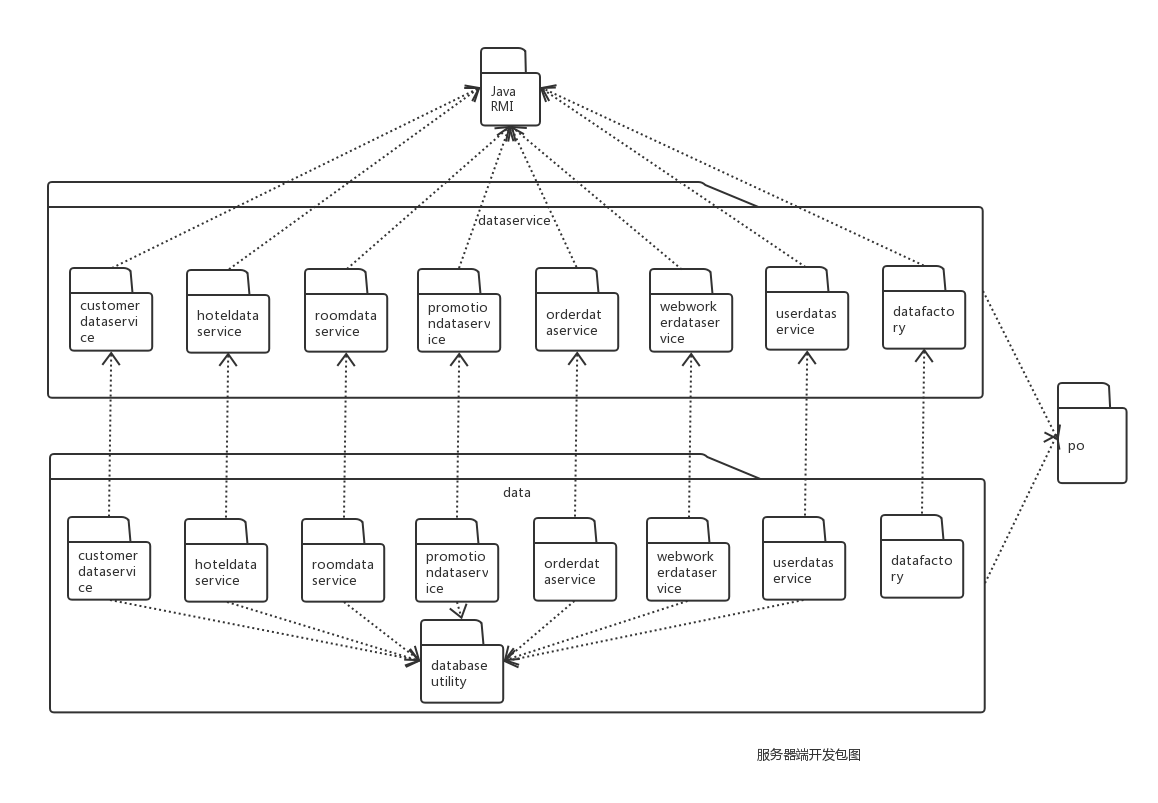
图3 酒店预定系统客户端开发包

图4 酒店预定系统服务器端包图

## 4.2运行时进程

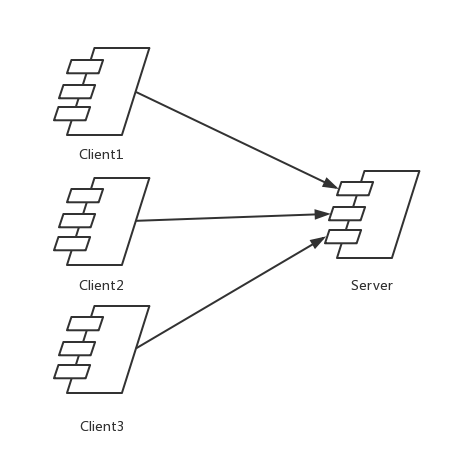
 在酒店预定系统中，会有多个 和一个服务器端进程，其进程图如图5所示。结合部署图，客户端进程实在客户端机器上运行，服务器端在服务器端机器上运行。

图5 进程图

## 4.3物理部署

酒店预定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java RMI构件属于JDK 6.0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6所示。

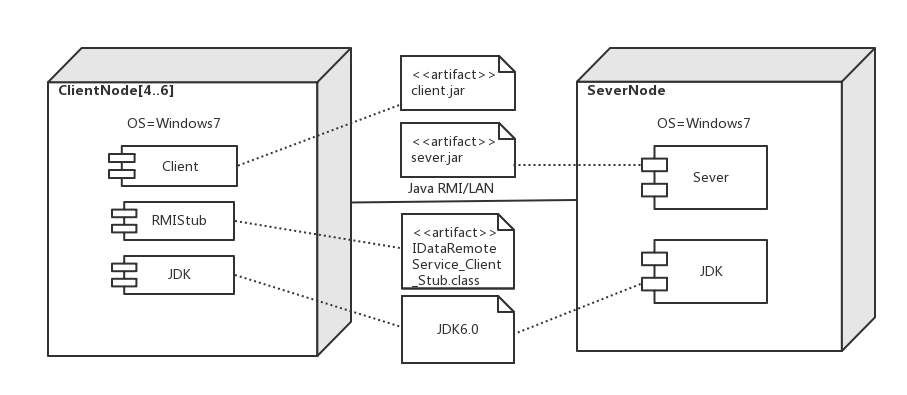


图6 部署图

# 5.接口视角

## 5.1模块的职责

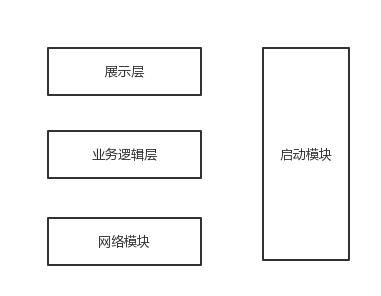
 客户端模块视图和服务器端模块视图分别如图7和图8所示。客户端各层和服务器端各层的职责分别如表2和表3所示。

图7 客户端模块视图

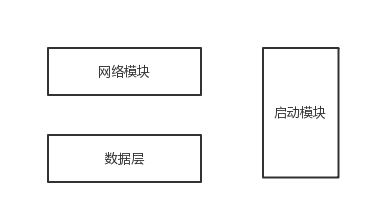


图8 服务器端模块视图

**表2 客户端各层的职责**

|  |  |
| --- | --- |
| **层** | **职责** |
| 启动模块 | 负责初始化网络通信机制 |
| 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javaRMI机制查找RMI服务 |

**表3 服务器端各层的职责**

|  |  |
| --- | --- |
| **层** | **职责** |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 数据层 | 负责数据的持久化及数据访问接口 |
| 服务器端网络模块 | 利用javaRMI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。层与层之间的接口如表4所示。

**表4 层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| **接口** | **服务调用方** | **服务提供方** |
| Customerblservice  Hotelblservice  Roomblservice  Promotionblservice  Orderblservice  Webworkerblservice  Userblservice | 客户端展示层 | 客户端业务逻辑层 |
| Customerdataservice  Hoteldataservice  Roomdataservice  Promotiondataservice  Orderdataservice  Webworkerdataservice  Userdataservice  Databasefactory | 客户端业务逻辑层 | 服务器端数据层 |

借用用户信用值充值这个用例来说明层与层之间的调用，如图9所示。每一层都是由上层依赖了一个接口，而下层实现了这个接口。Customerblservice提供了Customer界面所需要的所有业务逻辑功能。Customerdataservice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各层之间的耦合程度。

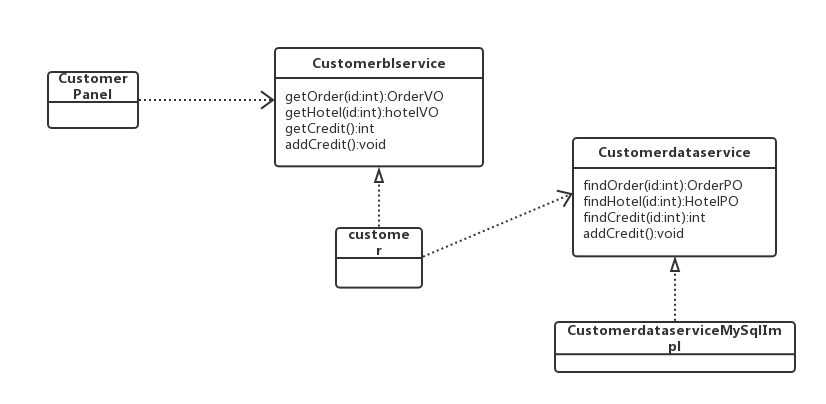


图9 信用充值用例之间调用的接口

## 5.2用户界面层的分解

根据需求，同存在如图10所示19个界面。界面跳转如图10所示。

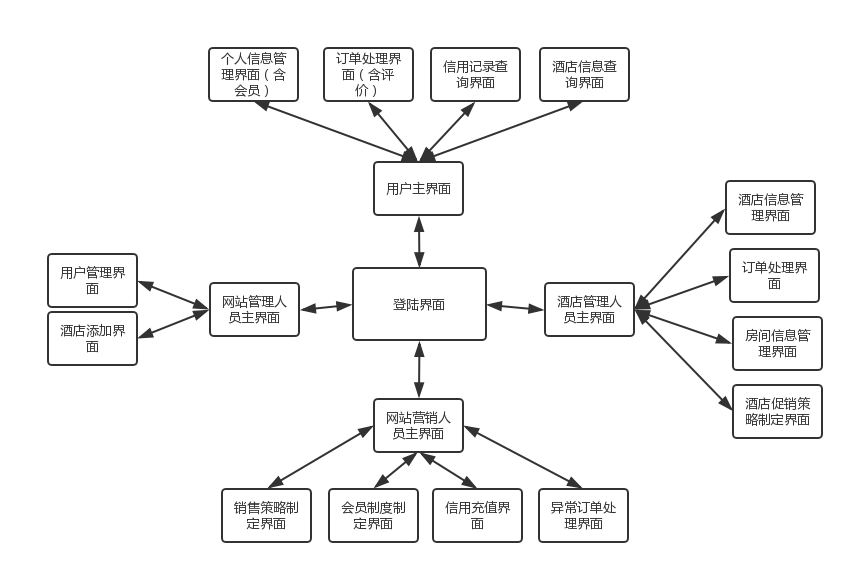
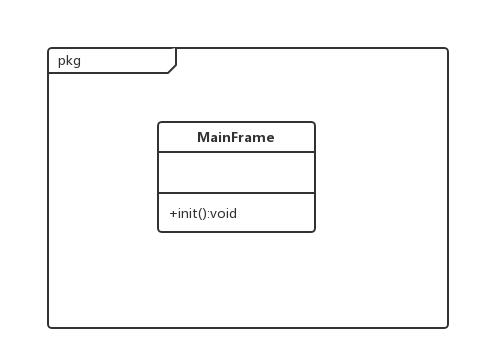


图10 用户界面跳转

服务器端和客户端的用户界面设计是一致的，只是具体的页面不一样。用户界面类如图11所示。

图11 用户界面类

### 5.2.1用户界面模块的职责

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**

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

用户界面层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** |
| Userloginservice.Loginblservice | 登陆界面的业务逻辑接口 |
| Userloginservice.\*blservice | 每个界面都有一个对应的业务逻辑接口 |

### 5.2.3用户界面模块的设计原理

用户界面利用Java的Swing和AWT库来实现。

## 5.3业务逻辑层的分解

### 5.3.1业务逻辑层模块的职责

|  |  |
| --- | --- |
| userbl | 负责实现对应与登录界面所需要的服务 |
| orderbl | 负责实现订单处理界面所需要的服务 |
| customerbl | 负责实现客户信息界面所需要的服务 |
| hotelbl | 负责实现酒店信息界面所对应的服务 |
| roombl | 负责实现房间信息界面所需要的服务 |
| promotionbl | 负责实现促销策略界面所需要的服务 |
| webworkrbl | 负责实现网站管理界面所需要的服务 |

### 5.3.2业务逻辑层模块的接口规范

**userbl模块的接口示范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| User.login | 语法 | public UserPO login(long account,String password) |
| 前置条件 | 输入有效的账户，密码 |
| 后置条件 | 根据id查找用户是否存在，根据用户信息验证密码是否正确，如果通过验证，则记录当前用户 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| UserDataService.find（int id） | | 根据ID查找用户信息 |
| UserDataService.finds(String field,int value) | | 根据两值查找多个持久化对象 |
| UserDataService.insert(UserPO po) | | 插入单一持久化对象 |
| UserDataService.delete(UserPO po) | | 删除单一持久化对象 |
| UserDataService.update(UserPO po) | | 更新单一持久化对象 |
| UserDataService.init | | 初始化持久化数据库 |
| UserDataService.finish | | 结束持久化数据库的使用 |

**Orderbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Orders.createOrder | 语法 | public void createOrder(OrderPO om) |
| 前置条件 | 启动一个预订服务 |
| 后置条件 | 在持久化数据库中添加一个订单 |
| Orders.getOrder | 语法 | public OrderPO getOrder(long id) |
| 前置条件 | 要获得一个订单的信息 |
| 后置条件 | 返回一个订单的信息 |
| Orders.cancelOrder | 语法 | public void cancelOrder(long id) |
| 前置条件 | 客户申请撤销订单 |
| 后置条件 | 系统将订单设置为已撤销状态 |
| Order.completeOrders | 语法 | public void completeOrder(long id) |
| 前置条件 | 客户按时办理入住 |
| 后置条件 | 系统将订单设置为已执行状态 |
| Order.recoverOrder | 语法 | Public void recoverOrder(long id) |
| 前置条件 | 客户延期入住，经过酒店或网站的审核 |
| 后置条件 | 系统将订单设为已执行状态 |
| 需要的服务（需接口） | | |
| OrdersDataService.find(long id) | | 根据id查找单一持久化对象 |
| OrdersDataService.finds(String field,int value) | | 根据两值查找多个持久化对象 |
| OrdersDataService.insert(SalesPO po) | | 插入单一持久化对象 |
| OrdersDataService.delete(SalesPo po) | | 删除单一持久化对象 |
| OrdersDataService.update(SalesPO po) | | 更新单一持久化对象 |
| OrdersDataService.init | | 初始化持久化数据库 |
| OrdersDataService.finish | | 结束持久化数据库的使用 |

**Hotelbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotel.getHotel | 语法 | public HotelPO getHotel(HotelMessage hm) |
| 前置条件 | 要获得酒店信息 |
| 后置条件 | 返回酒店信息 |
| Hotel.addEvaluation | 语法 | public void addEvaluation (long id，String str) |
| 前置条件 | 有客户给该酒店评价 |
| 后置条件 | 把得到的评分和评价插入到数据库中 |
| Hotel.changeHotel | 语法 | Public void changeHotel(HotelPO rm) |
| 前置条件 | 酒店信息发生更改 |
| 后置条件 | 更新持久化数据 |
| 需要的服务（需接口） | | |
| HotelDataService.find(long id) | | 根据id查找单一持久化对象 |
| HotelDataService.finds(String field,int value) | | 根据两值查找多个持久化对象 |
| HotelDataService.insert(HotelPO po) | | 插入单一持久化对象 |
| HotelDataService.delete(HotelPo po) | | 删除单一持久化对象 |
| HotelDataService.update(HotelPO po) | | 更新单一持久化对象 |
| HotelDataService.init | | 初始化持久化数据库 |
| HotelDataService.finish | | 结束持久化数据库的使用 |

**Roombl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Room.getRoom | 语法 | public HotelPO getRoom(long id) |
| 前置条件 | 要显示酒店的房间信息 |
| 后置条件 | 返回房间信息 |
| Room.changeRoom | 语法 | public void change(RoomPO rm) |
| 前置条件 | 房间状态改变 |
| 后置条件 | 改变持久化数据中的房间状态 |
| Room.addRoom | 语法 | Public void addRoom(RoomPO rm) |
| 前置条件 | 酒店申请添加房间 |
| 后置条件 | 在持久化数据库中添加一个房间的信息 |
| 需要的服务（需接口） | | |
| RoomDataService.find(long id) | | 根据id查找单一持久化对象 |
| RoomDataService.finds(String field,int value) | | 根据两值查找多个持久化对象 |
| RoomDataService.insert(RoomPO po) | | 插入单一持久化对象 |
| RoomDataService.delete(RoomPo po) | | 删除单一持久化对象 |
| RoomDataService.update(RoomPO po) | | 更新单一持久化对象 |
| RoomDataService.init | | 初始化持久化数据库 |
| RoomDataService.finish | | 结束持久化数据库的使用 |

**Webworkerbl模块的规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Webworker.getAccount | 语法 | public String getAccount(hotelPO hm) |
| 前置条件 | 有人注册会员 |
| 后置条件 | 有人注册会员后，记录其信息 |
| 需要的服务（需接口） | | |
| WebworkerDataService.find(long id) | | 根据id查找单一持久化对象 |
| WebworkerDataService.finds(String field,int value) | | 根据两值查找多个持久化对象 |
| WebworkerDataService.insert(WebworkerPO po) | | 插入单一持久化对象 |
| WebworkerDataService.delete(WebworkerPo po) | | 删除单一持久化对象 |
| WebworkerDataService.update(WebworkerPO po) | | 更新单一持久化对象 |
| WebworkerDataService.init | | 初始化持久化数据库 |
| WebworkerDataService.finish | | 结束持久化数据库的使用 |

**Customerbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Customer.getCustomer | 语法 | public CustomerPO getCustomer(long id) |
| 前置条件 | 当要获得用户信息时 |
| 后置条件 | 返回用户信息 |
| Customer.changeCustomer | 语法 | public void changeCustomer(long id) |
| 前置条件 | 客户请求改变信息 |
| 后置条件 | 更新客户信息 |
| 需要的服务（需接口） | | |
| CustomerDataService.find(long id) | | 根据id查找单一持久化对象 |
| CustomerDataService.finds(String field,int value) | | 根据两值查找多个持久化对象 |
| CustomerDataService.insert(CustomerPO po) | | 插入单一持久化对象 |
| CustomerDataService.delete(CustomerPo po) | | 删除单一持久化对象 |
| CustomerDataService.update(CustomerPO po) | | 更新单一持久化对象 |
| CustomerDataService.init | | 初始化持久化数据库 |
| CustomerDataService.finish | | 结束持久化数据库的使用 |

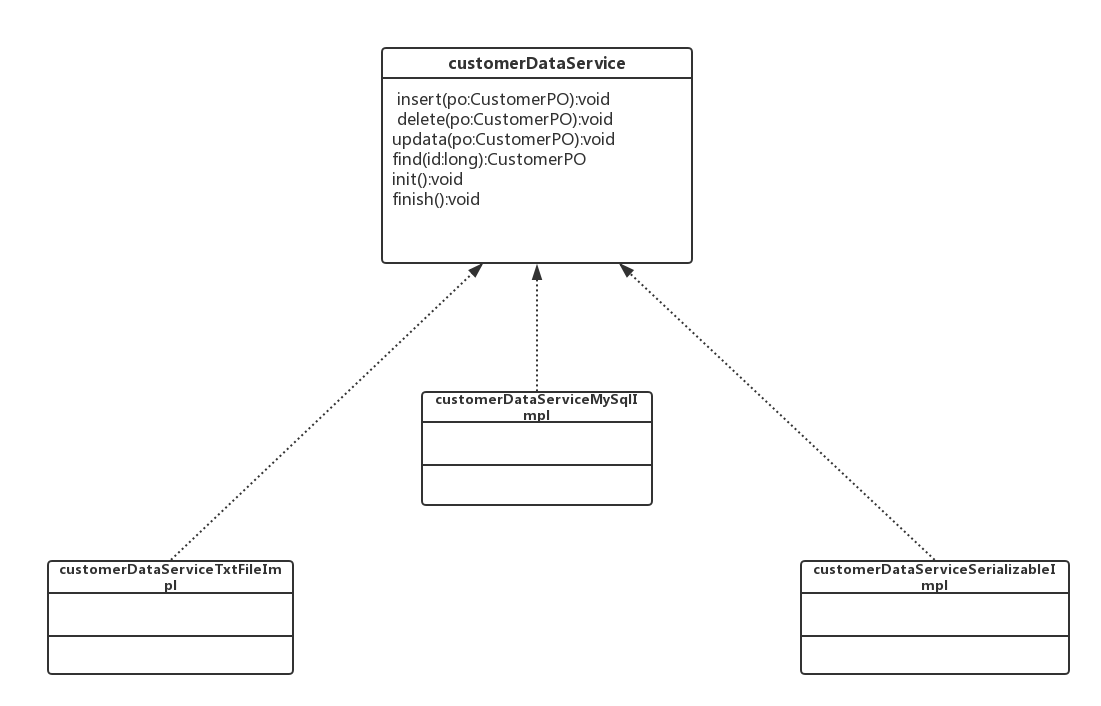
**Promotionbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Promotion.usePromotion | 语法 | public promotionPO usePromotion(long id) |
| 前置条件 | 下一个订单 |
| 后置条件 | 下一个订单后，使用促销 |
| Promotion.changePromotion | 语法 | public promotionPO changePromotion(long id) |
| 前置条件 | 有新的促销策略 |
| 后置条件 | 有新的促销策略后，更新促销的服务信息 |
| Promotion.getPromotion | 语法 | Public String getPromotion() |
| 前置条件 | 需要展示策略的文本信息时 |
| 后置条件 | 返回促销策略的文本信息 |
| 需要的服务（需接口） | | |
| PromotionDataService.find(long id) | | 根据id查找单一持久化对象 |
| PromotionDataService.finds(String field,int value) | | 根据两值查找多个持久化对象 |
| PromotionDataService.insert(WebworkerPO po) | | 插入单一持久化对象 |
| PromotionDataService.delete(WebworkerPo po) | | 删除单一持久化对象 |
| PromotionDataService.update(WebworkerPO po) | | 更新单一持久化对象 |
| PromotionDataService.init | | 初始化持久化数据库 |
| PromotionDataService.finish | | 结束持久化数据库的使用 |

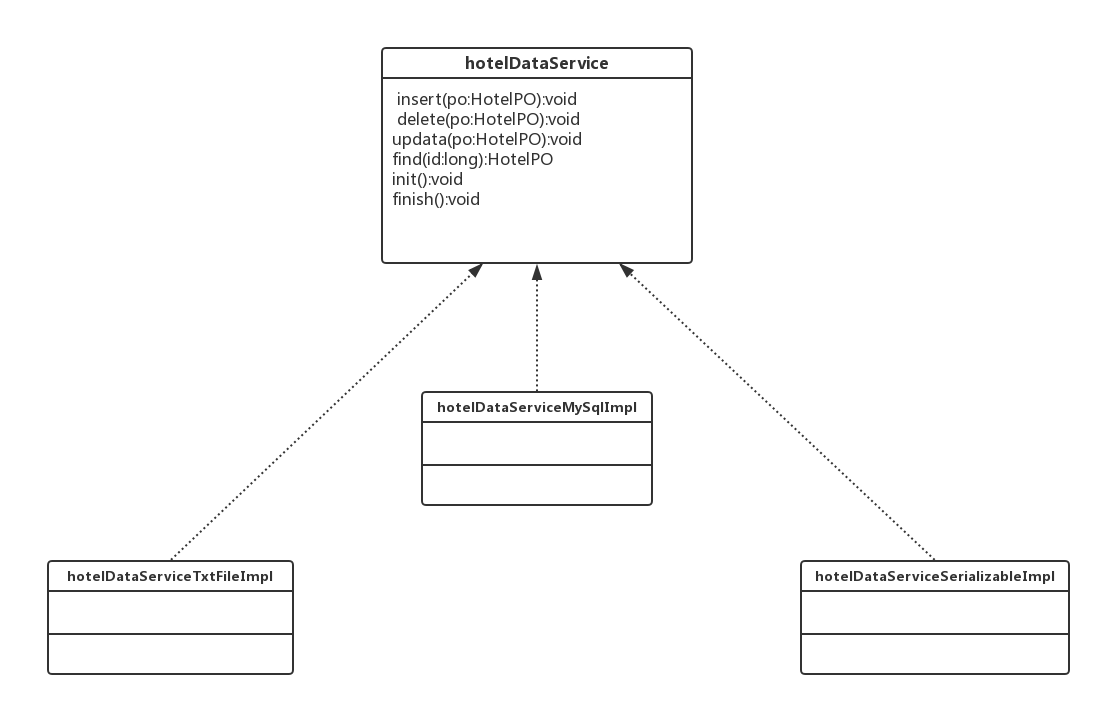
## 5.4数据层的分解

数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增、删、改、查。

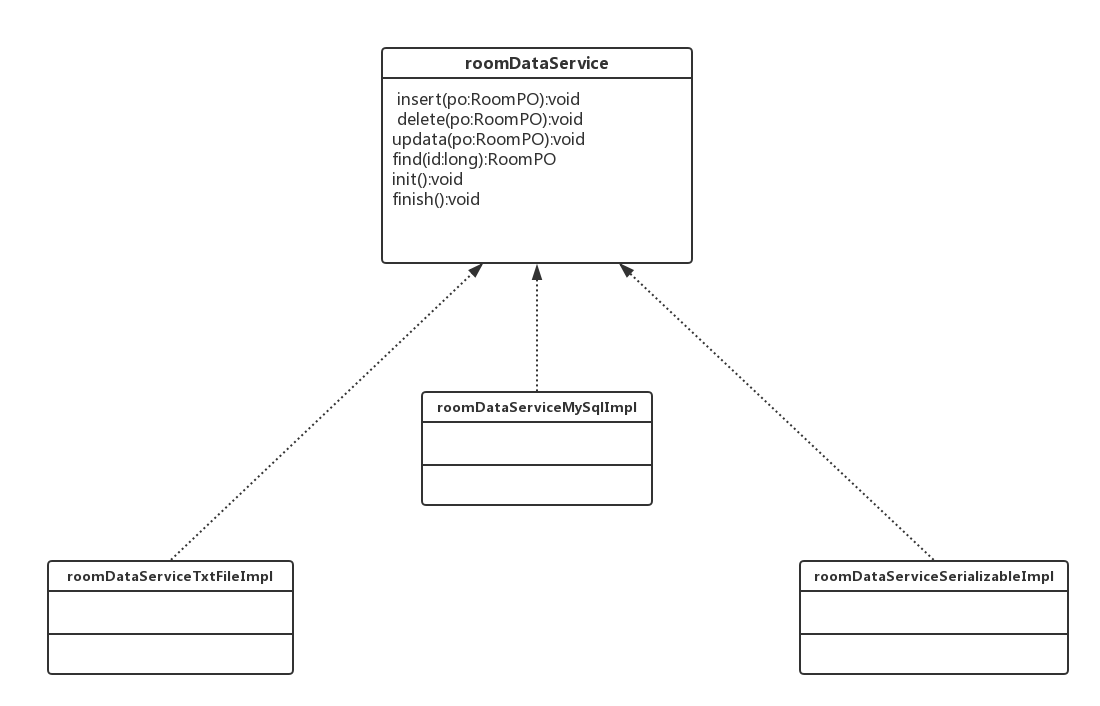
由于持久化数据的保存可能存在多种形式：txt文件，序列化文件，数据库等，所以抽象了数据服务，数据层模块的描述具体如下图所示。



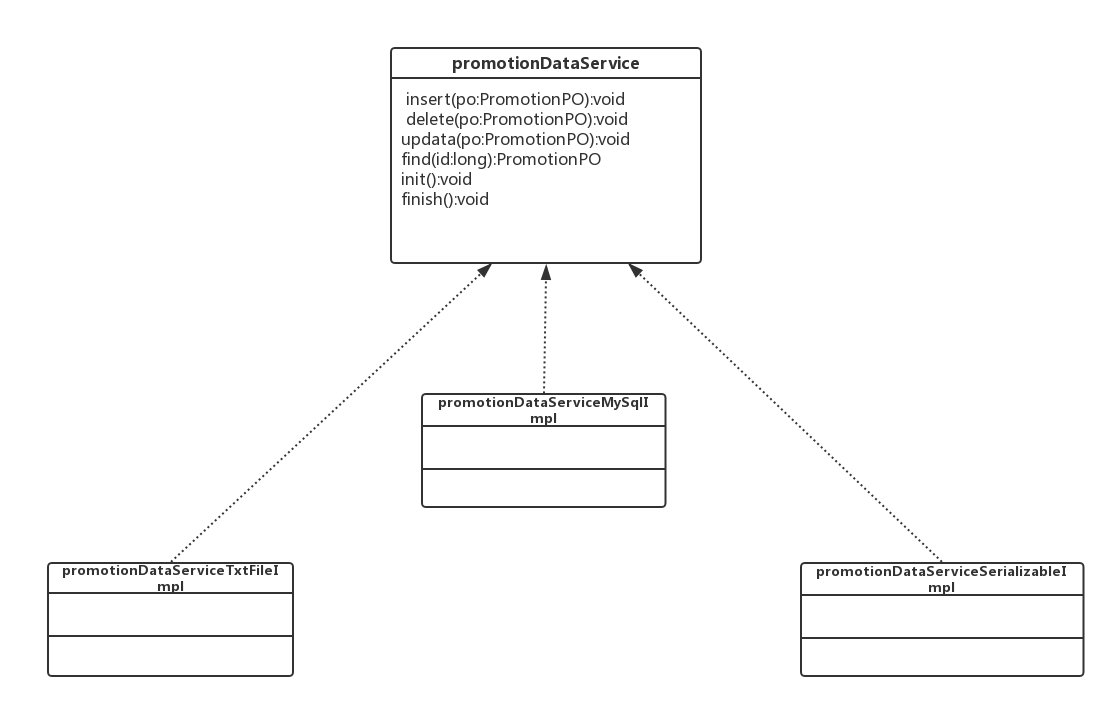
数据层模块的描述—customer



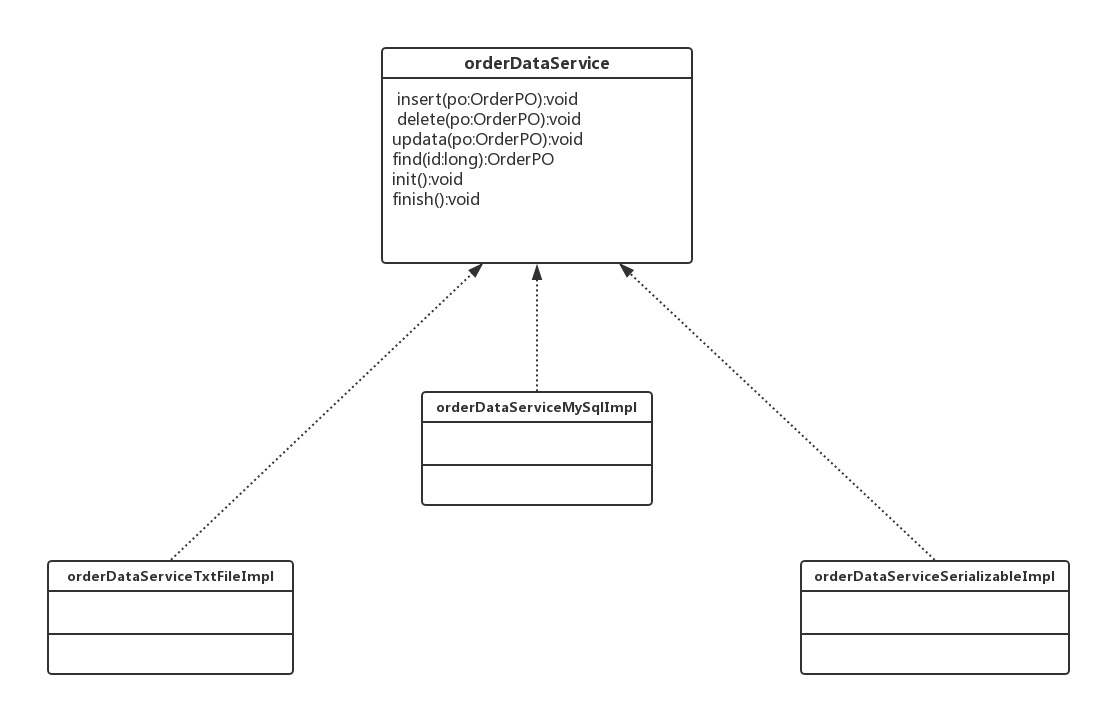
数据层模块的描述—hotel



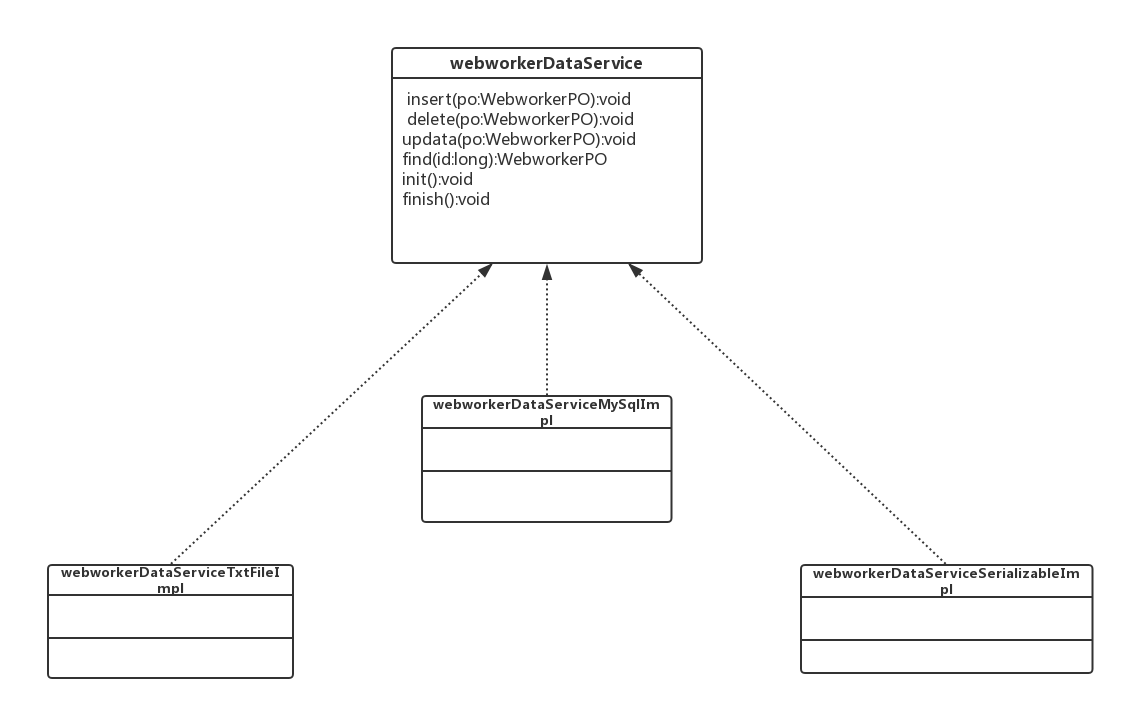
数据层模块的描述—room



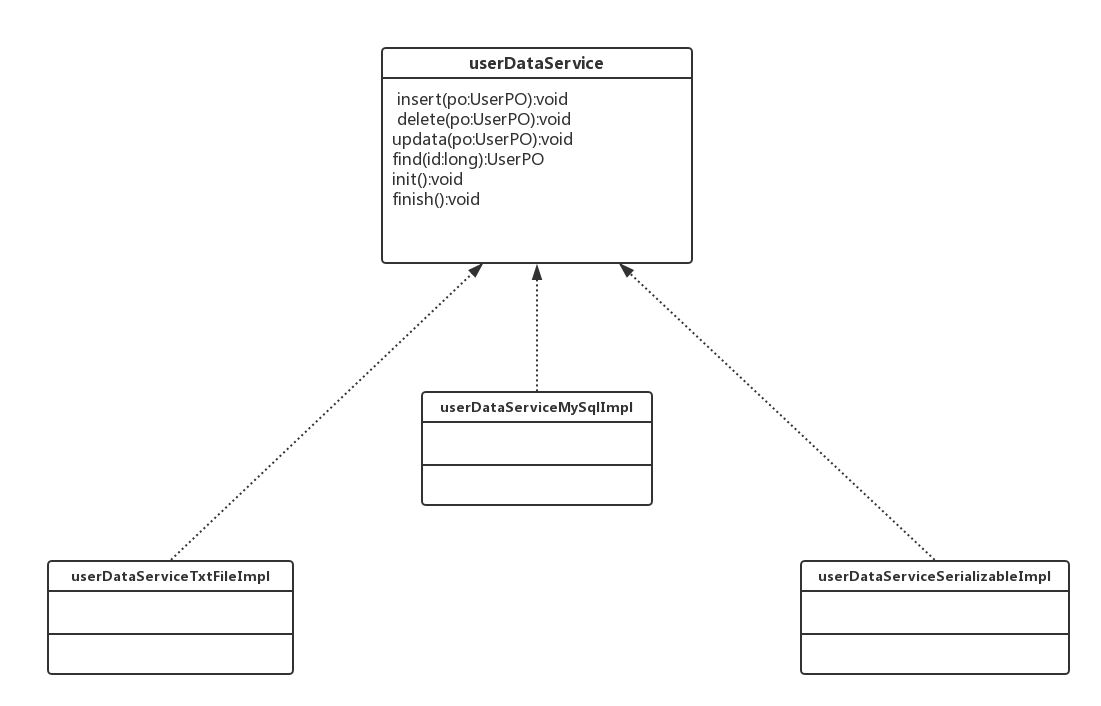
数据层模块的描述—promotion



数据层模块的描述—order



数据层模块的描述—webworker



数据层模块的描述—user

### 5.4.1数据层模块的职责

数据层模块的职责如下表所示

|  |  |
| --- | --- |
| 模块 | 职责 |
| CustomerDataService | 持久化客户信息数据库的接口，提供集体载入，集体保存、增、删、改、查服务 |
| CustomerDataServiceTxtFileImpl | 基于txt文件的持久化客户信息数据库的接口，提供集体载入、集体保存、增、删、改、查服务 |
| CustomerDataServiceSerializableImpl | 基于序列化文件的持久化客户信息数据库的接口，提供集体载入、集体保存、增、删、改、查服务 |
| CustomerDataServiceMySqlImpl | 基于MySql数据库的持久化客户信息数据库的接口，提供集体载入、集体保存、增、删、改、查 |
| HotelDataService | 持久化酒店信息数据库的接口，提供集体载入，集体保存、增、删、改、查服务 |
| HotelDataServiceTxtFileImpl | 基于txt文件的持久化酒店信息数据库的接口，提供集体载入、集体保存、增、删、改、查服务 |
| HotelDataServiceSerializableImpl | 基于序列化文件的持久化酒店信息数据库的接口，提供集体载入、集体保存、增、删、改、查服务 |
| HotelDataServiceMySqlImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存、增、删、改、查 |
|  |  |
| RoomDataService | 持久化客户信息数据库的接口，提供集体载入，集体保存、增、删、改、查服务 |
| RoomDataServiceTxtFileImpl | 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 |
| RoomDataServiceSerializableImpl | 基于序列化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 |
| RoomDataServiceMySqlImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存、增、删、改、查 |
| PromotionDataService | 持久化客户信息数据库的接口，提供集体载入，集体保存、增、删、改、查服务 |
| PromotionDataServiceTxtFileImpl | 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 |
| PromotionDataServiceSerializableImpl | 基于序列化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 |
| PromotionDataServiceMySqlImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存、增、删、改、查 |
| OrderDataService | 持久化客户信息数据库的接口，提供集体载入，集体保存、增、删、改、查服务 |
| OrderDataServiceTxtFileImpl | 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 |
| OrderDataServiceSerializableImpl | 基于序列化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 |
| OrderDataServiceMySqlImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存、增、删、改、查 |
| WebworkerDataService | 持久化客户信息数据库的接口，提供集体载入，集体保存、增、删、改、查服务 |
| WebworkerDataServiceTxtFileImpl | 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 |
| WebworkerDataServiceSerializableImpl | 基于序列化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 |
| WebworkerDataServiceMySqlImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存、增、删、改、查 |
| UserDataService | 持久化客户信息数据库的接口，提供集体载入，集体保存、增、删、改、查服务 |
| UserDataServiceTxtFileImpl | 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 |
| UserDataServiceSerializableImpl | 基于序列化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 |
| UserDataServiceMySqlImpl | 基于MySql数据库的持久化数据库的接口，提供集体载入、集体保存、增、删、改、查 |

### 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所示

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CustomerDataService.find | 语法 | Public CustomerPo find(long id) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的CustomerPO结果 |
| CustomerDataService.find | 语法 | Public CustomerPo find(String field,int value)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按字段名和值查找多个CustomerPO结果 |
| CustomerDataService.insert | 语法 | Public void insert(CustomerPO po) throws RemoteException; |
| 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 |
| CustomerDataService.delete | 语法 | Public void delete(CustomerPO po) throws RemoteException; |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 删除一个PO |
| CustomerDataService.updata | 语法 | Public void update(CustomerPO po) throws RemoteException; |
| 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 |
| CustomerDataService.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 |
| CustomerDataService.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 |
|  |  |  |
| HotelDataService.find | 语法 | Public HotelPo find(long id) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的HotelPO结果 |
| HotelDataService.find | 语法 | Public HotelPo find(String field,int value)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按字段名和值查找多个HotelPO结果 |
| HotelDataService.insert | 语法 | Public void insert(HotelPO po) throws RemoteException; |
| 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.delete | 语法 | Public void delete(HotelPO po) throws RemoteException; |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 删除一个PO |
| HotelDataService.updata | 语法 | Public void update(HotelPO po) throws RemoteException; |
| 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() throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| HotelDataService.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 |

|  |  |  |
| --- | --- | --- |
| RoomDataService.find | 语法 | Public RoomPo find(long id) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的RoomPO结果 |
| RoomDataService.find | 语法 | Public RoomPo find(String field,int value)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按字段名和值查找多个RoomPO结果 |
| RoomDataService.insert | 语法 | Public void insert(RoomPO po) throws RemoteException; |
| 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 |
| RoomDataService.delete | 语法 | Public void delete(RoomPO po) throws RemoteException; |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 删除一个PO |
| RoomDataService.updata | 语法 | Public void update(RoomPO po) throws RemoteException; |
| 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 |
| RoomDataService.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 |
| RoomDataService.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 |

|  |  |  |
| --- | --- | --- |
| PromotionDataService.find | 语法 | Public PromotionPo find(long id) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的PromotionPO结果 |
| PromotionDataService.find | 语法 | Public PromotionPo find(String field,int value)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按字段名和值查找多个PromotionPO结果 |
| PromotionDataService.insert | 语法 | Public void insert(PromotionPO po) throws RemoteException; |
| 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.delete | 语法 | Public void delete(PromotionPO po) throws RemoteException; |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 删除一个PO |
| PromotionDataService.updata | 语法 | Public void update(PromotionPO po) throws RemoteException; |
| 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() throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| PromotionDataService.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 |
| OrderDataService.find | 语法 | Public OrderPo find(long id) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的OrderPO结果 |
| OrderDataService.find | 语法 | Public OrderPo find(String field,int value)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按字段名和值查找多个OrderPO结果 |
| OrderDataService.insert | 语法 | Public void insert(OrderPO po) throws RemoteException; |
| 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.delete | 语法 | Public void delete(OrderPO po) throws RemoteException; |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 删除一个PO |
| OrderDataService.updata | 语法 | Public void update(OrderPO po) throws RemoteException; |
| 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() throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| OrderDataService.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 |
| WebworkerDataService.find | 语法 | Public WebworkerPo find(long id) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的WebworkerPO结果 |
| WebworkerDataService.find | 语法 | Public WebworkerPo find(String field,int value)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按字段名和值查找多个WebworkerPO结果 |
| WebworkerDataService.insert | 语法 | Public void insert(WebworkerPO po) throws RemoteException; |
| 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 |
| WebworkerDataService.delete | 语法 | Public void delete(WebworkerPO po) throws RemoteException; |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 删除一个PO |
| WebworkerDataService.updata | 语法 | Public void update(WebworkerPO po) throws RemoteException; |
| 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 |
| WebworkerDataService.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 |
| WebworkerDataService.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 |
| UserDataService.find | 语法 | Public UserPo find(long id) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的UserPO结果 |
| UserDataService.find | 语法 | Public UserPo find(String field,int value)throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 按字段名和值查找多个UserPO结果 |
| UserDataService.insert | 语法 | Public void insert(UserPO po) throws RemoteException; |
| 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 |
| UserDataService.delete | 语法 | Public void delete(UserPO po) throws RemoteException; |
| 前置条件 | 同样ID的PO在数据库中存在 |
| 后置条件 | 删除一个PO |
| UserDataService.updata | 语法 | Public void update(UserPO po) throws RemoteException; |
| 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 |
| UserDataService.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 |
| UserDataService.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 |

# 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。

* UserPO类包含用户的用户名、密码属性。
* OrderPO类则包含订单的信息如时间、客户、酒店及状态等属性。
* CustomerPO类包含用户的ID、信用值、消费记录等属性。
* HotelPO类则包括酒店的位置、星级、好评等属性。
* RoomPO类则包含酒店房间的数量与位置信息。
* PromotionPO类则包含酒店与网站营销策略信息。
* WebeorkerPO类则包含用户身份数据信息。

## 6.2TXT持久化格式

数据持久化格式以UserLog.txt为例。每行分别对应ID、密码。中间用”：“隔开，如下所示：

1339527：1258056789

1831632：6688224466

## 6.3数据库表

数据库中包含：User表、Hotel表、Room表、Promotion表、Order表、Webwork表、Customer表。