**酒店线上预订系统OBSH**

**软件体系结构描述文档**

**V3.0草稿版**

**2016-10-11**

**更新历史**

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 全体成员 | **2016/10/13** | **增加逻辑层接口** | **V1.0草稿** |
| 全体成员 | **2016/10/14** | **增加数据层接口** | **V2.0草稿** |
| 全体成员 | **2016/10/15** | **添加信息视角和目录** | **V3.0草稿** |
| 全体成员 | **2016/10/16** | **修改数据层接口** | **V4.0正式版** |

16组成员：毕潇晗 顾忻 何阳 胡一凡

**目录**

1. **引言…………………………………………………………………………………………P1**

**1.1编制目的…………………………………………………………………………………P1**

**1.2词汇表……………………………………………………………………………………P1**

**1.3参考资料…………………………………………………………………………………P1**

1. **产品概述……………………………………………………………………………………P1**
2. **逻辑视角……………………………………………………………………………………P1**
3. **组合视角……………………………………………………………………………………P2**

**4.1开发图包…………………………………………………………………………………P2**

**4.2运行时进程………………………………………………………………………………P5**

**4.3物理部署…………………………………………………………………………………P6**

1. **接口视角……………………………………………………………………………………P6**

**5.1模块的职责………………………………………………………………………………P6**

**5.2用户界面层的分解………………………………………………………………………P8**

**5.3业务逻辑层的分解………………………………………………………………………P9**

**5.4数据层的分解……………………………………………………………………………P16**

1. **信息视角……………………………………………………………………………………P20**

**6.1数据持久化对象…………………………………………………………………………P20**

**6.2数据库表…………………………………………………………………………………P23**

1. **引言**

**1.1编制目的**

本报告详细完成对酒店线上预定系统的概要设计，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、测试人员以及最终用户而编写，是了解系统的导航。

**1.2词汇表**

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| OBSH | 酒店线上预订系统 | 无 |

**1.3参考资料**

1.酒店线上用例文档V2.1正式版

2.酒店线上预订系统OBSH需求文档V3.0正式版

1. **产品概述**

参考酒店线上预定系统用例文档和酒店线上预定系统软件需求规格说明中对产品的概括描述。酒店线上预订系统OBSH是为传统的线下酒店开发的业务系统，开发的目标是帮助酒店处理线上的酒店预订，包括酒店的预订、酒店促销、入住信息处理、客房管理。通过酒店线上预订系统OBSH的应用，期望为酒店提高酒店工作人员的工作效率、提高营销额和利润、提升酒店知名度、降低运营成本、吸引回头客并提高满意度。

1. **逻辑视角**

酒店线上预定系统OBSH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3.1和图3.2所示。

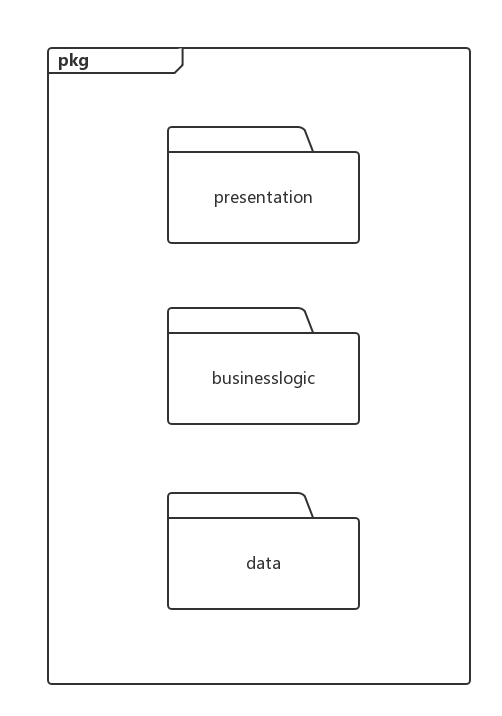


图3.1 参照体系结构风格的包图表达逻辑视角

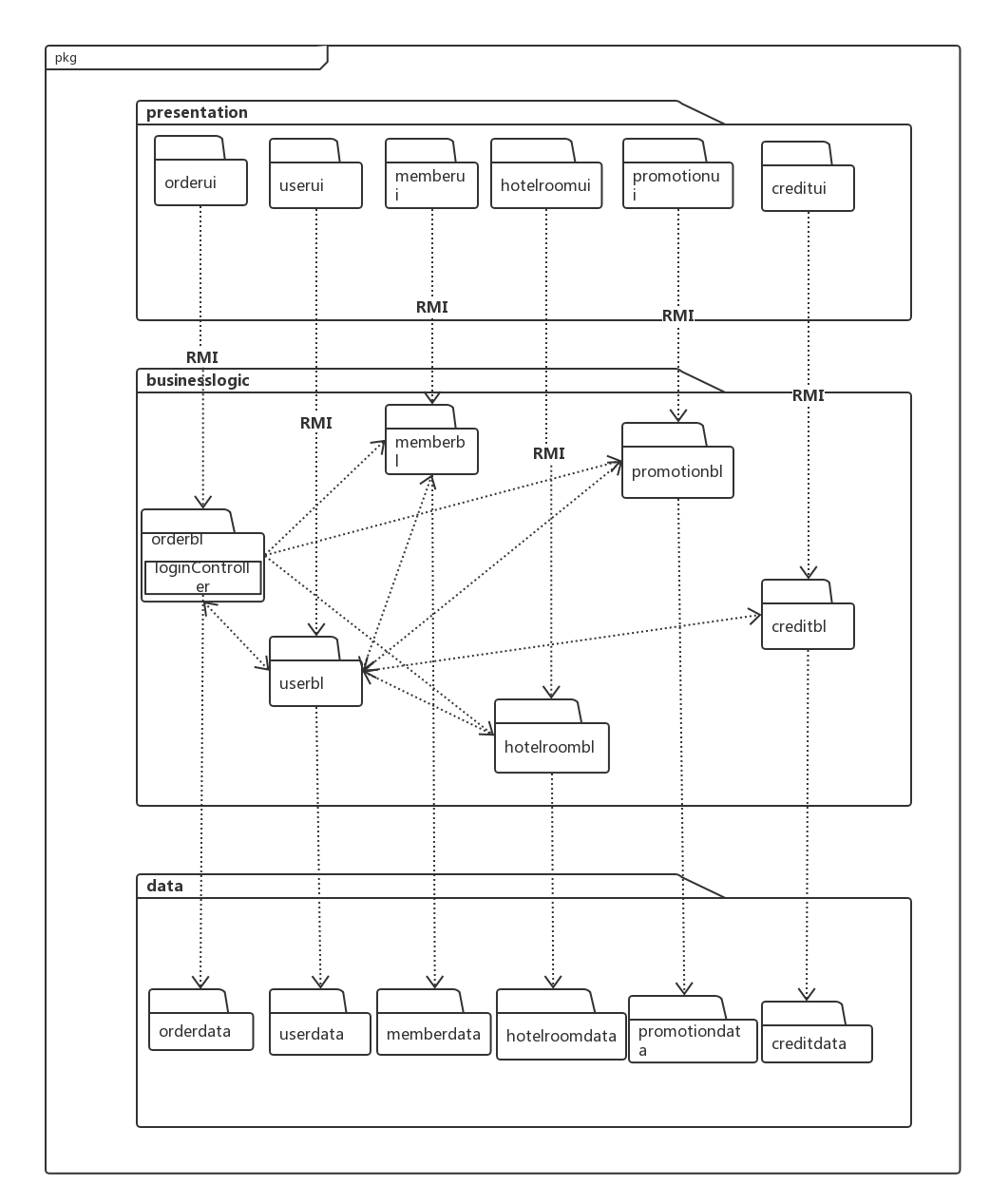


图3.2 软件体系结构逻辑设计方案

1. **组合视角**

**4.1开发包图**

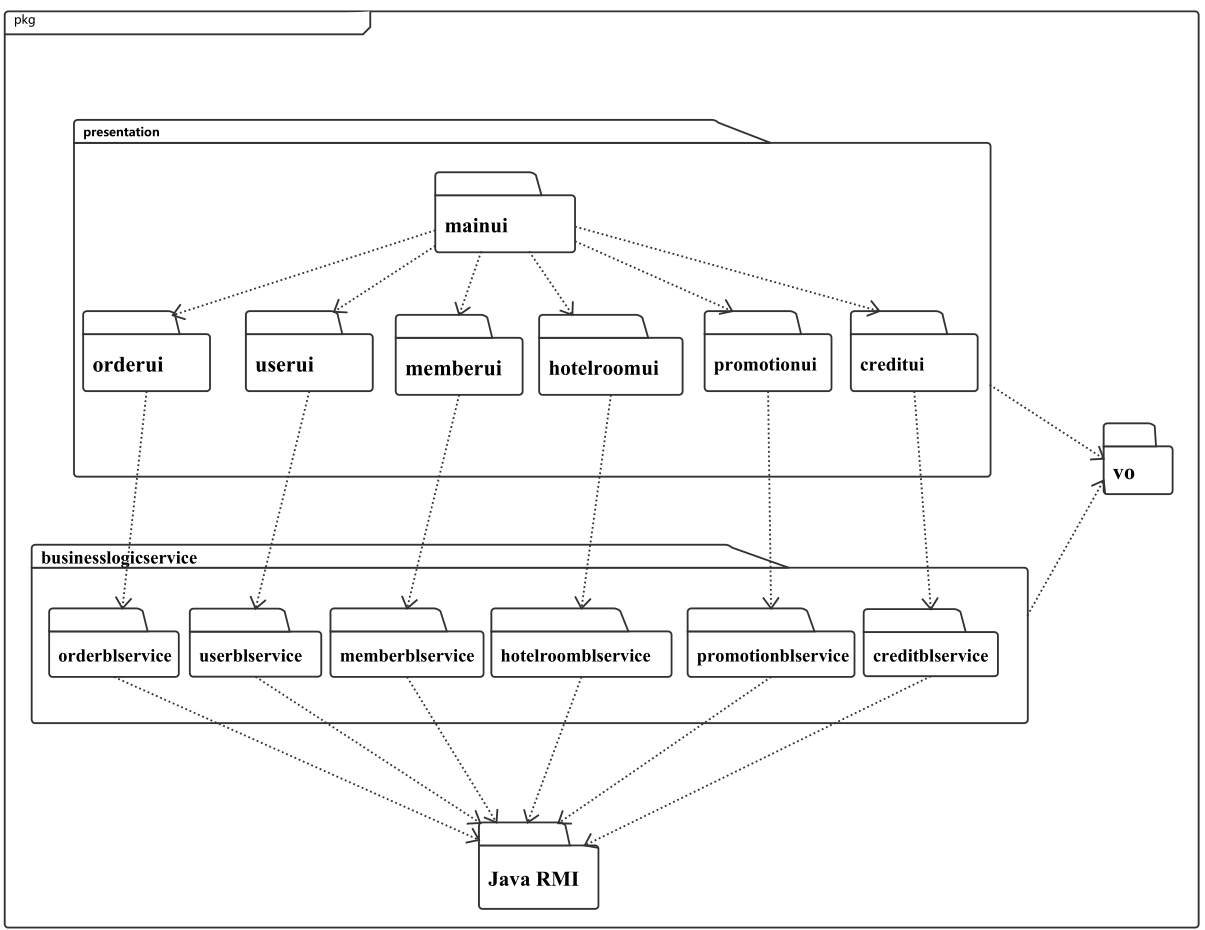
酒店线上预定系统的最终开发包设计如表4.1.1所示

表4.1.1 酒店线上预定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,userui,memberui,promotionui,hotelroomui,creditui,vo |
| orderui | orderblservice,界面类库包,vo |
| orderbl | orderblservice,orderdataservice,po,userbl,promotionbl |
| orderblservice | JAVA RMI |
| orderdata | databaseutility,po |
| orderdataservice |  |
| userui | userblservice,界面类库包 |
| userbl | Userinterface,userdataclient,userpo |
| userblservice | JAVA RMI |
| userdata | databaseutility,po |
| userdataservice |  |
| memberui | memberblservice,界面类库包 |
| memberbl | memberblservice,memberdataservice,po,userbl |
| memberblservice | JAVA RMI |
| memberdata | databaseutility,po |
| memberdataservice |  |
| hotelroomui | hotelroomblservice,界面类库包 |
| hotelroombl | hotelroomblservice,hotelroomdataservice,po,orderbl |
| hotelroomblservice | JAVA RMI |
| hotelroomdata | databaseutility,po |
| hotelroomdataservice |  |
| promotionui | promotionblservice,界面类库包 |
| promotionbl | hotelroomblservice,hotelroomdataservice,po |
| promotionblservice | JAVA RMI |
| promotiondata | databaseutility,po |
| promotiondataservice |  |
| creditui | creditblservice,界面类库包 |
| creditbl | creditblservice,creditdataservice,po,userbl |
| creditblservice | JAVA RMI′ |
| creditdata | databaseutility,po |
| creditdataservice |  |
| vo |  |
| po |  |
| utilitybl |  |
| 界面类库包 |  |
| Java RMI |  |
| databaseutility | JDBC |

酒店线上预定系统客户端开发包图如图4.1.1所示，服务器端开发包图如图4.1.2所示

图4.1.1 酒店线上预订系统客户端开发包图



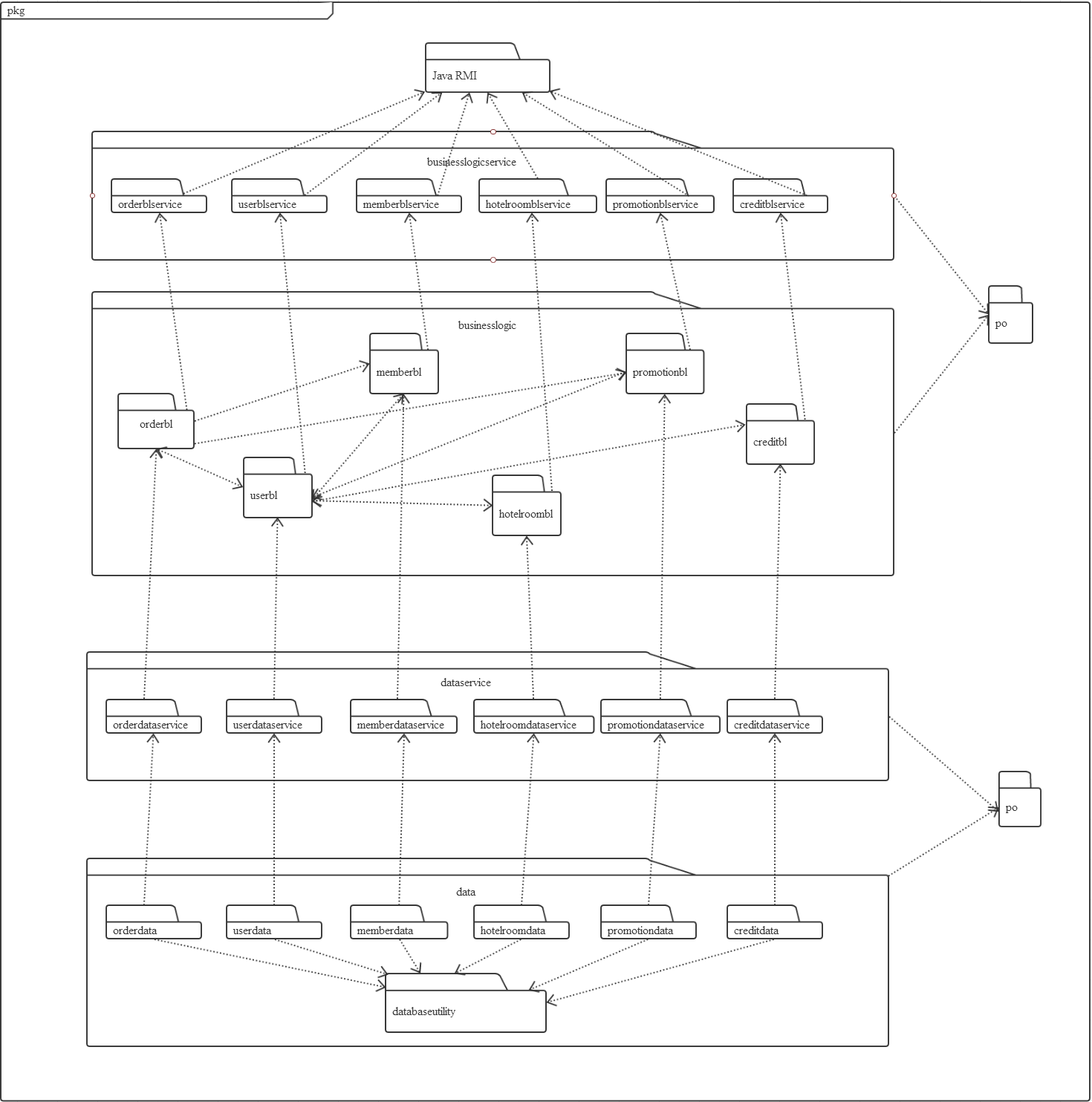


图4.1.2酒店线上预订系统服务器端开发包图

**4.2运行时进程**

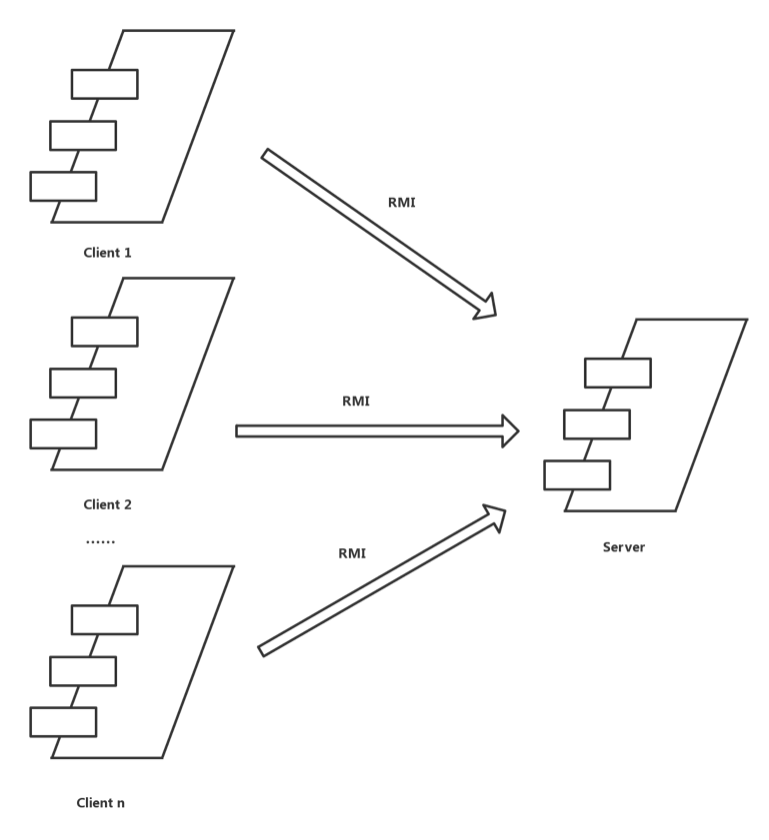
 在酒店线上预定系统中，会有多个客户端进程和一个服务器端进程，其进程图如图4.2.1所示。结合部署图，客户端进程是在客户端机器上运行，服务器端进程在服务器端机器上运行。

图4.2.1 进程图

**4.3物理部署**

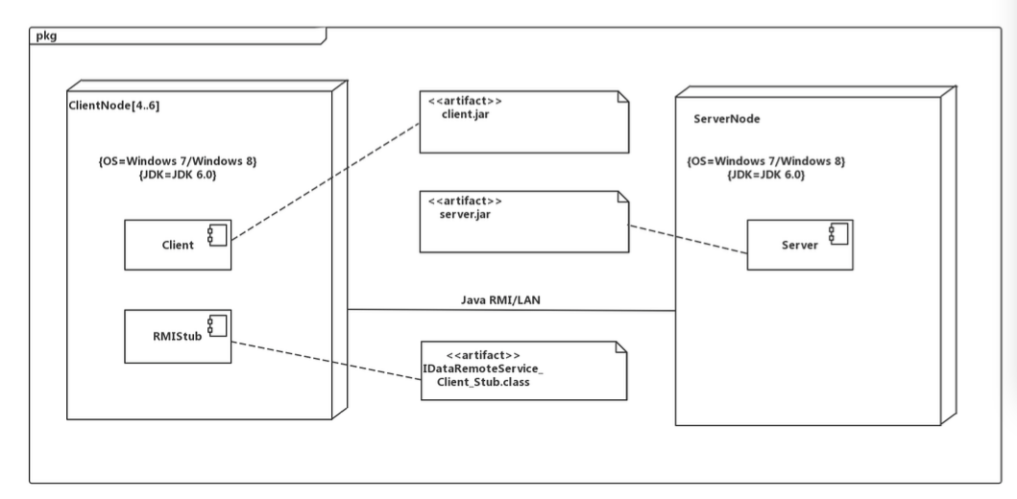
 酒店线上预定系统中客户端构件是放在客户端机器上，服务器端构件是放在服务器端机器上。在客户端节点上，还要部署RMIStub构件。由于JAVA RMI构件属于JDK7.0的一部分。所以，在系统JDK环境已经设置好的情况下，不需要再独立部署。部署图如图4.3.1所示。

图4.3.1 部署图

1. **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如图5.1.1所示。客户端各层和服务器端各层的职责如表5.1.1和表5.1.2所示。

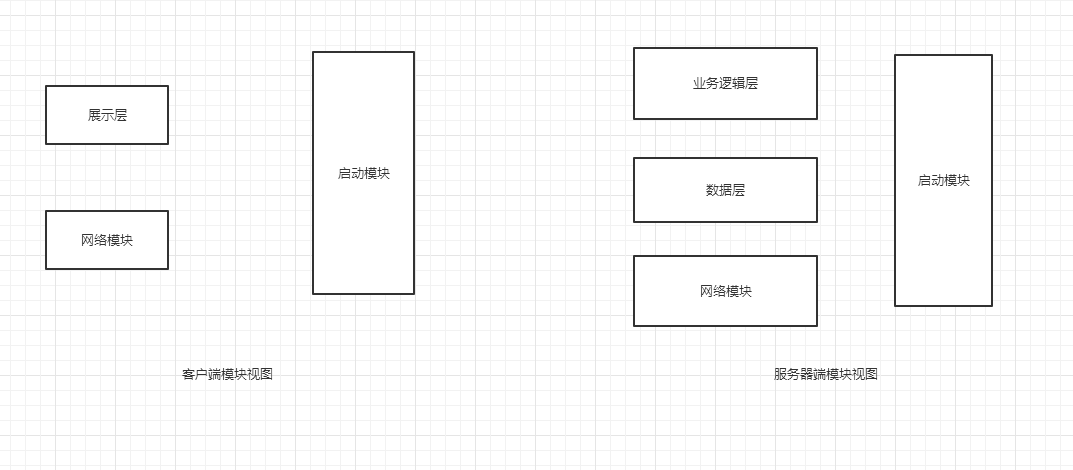


图5.1.1

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 |
| 客户界面层 | 基于窗口的酒店线上预订客户端用户界面 |
| 客户端网络模块 | 利用Java RMI机制查找RMI服务 |

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 |
| 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接口 |
| 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 |
| OrderblService  UserblService  MemberblService  HotelroomblService  PromotionblService  CreditblService | 客户端展示层 | 服务器端逻辑层 |
| OrderDataService  UserDataService  MemberDataService  HotelroomDataService  PromotionDataService  CreditDataService  Databasefactory | 服务器端逻辑层 | 服务器端数据层 |

**5.2用户界面层的分解**

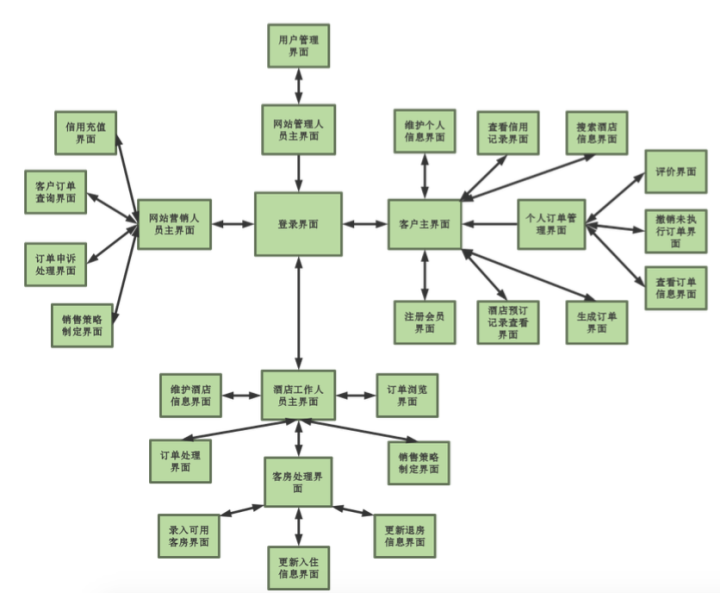
 根据需求，系统存在28个用户界面：登录界面、网站营销人员主界面、信用充值界面、客户订单查询界面、订单申诉处理界面、（网站营销人员的）销售策略制定界面、网站管理人员主界面、用户管理界面、客户主界面、维护个人信息界面、注册会员界面、查看信用记录界面、酒店预定记录查看界面、个人订单管理界面、生成订单界面、搜索酒店信息界面、评价界面、撤销未执行订单界面、查看订单信息界面、酒店工作人员主界面、维护酒店信息界面、订单处理界面、订单浏览界面、客房处理界面、销售策略制定界面、录入可用客房界面、更新入住信息界面、更新退房信息界面。界面跳转如图5.2.1所示。

图5.2.1 用户界面跳转

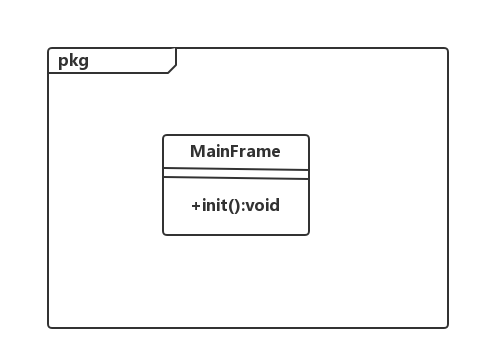
 服务器端和客户端的用户界面设计接口是一致的，只是具体的页面不一样。用户界面类如图5.2.2所示。

图5.2.2 用户界面类

**5.2.1用户界面层模块的职责**

如表5.2.1.2所示为用户界面层模块的职责。

表5.2.1.2 用户界面层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| MainFrame | 界面MainFrame，负责界面的显示和界面的跳转 |

**5.2.2用户界面层模块的接口规范**

用户界面层模块的接口规范如表5.2.2.1所示。

表5.2.2.1 用户界面层模块的接口规范

|  |  |  |
| --- | --- | --- |
| 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 | 登录界面的服务逻辑接口 |
| 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JavaFX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，User对象负责处理登录界面的业务逻辑；Order对象负责订单界面的业务逻辑。业务逻辑层的设计如图5.3.1所示。

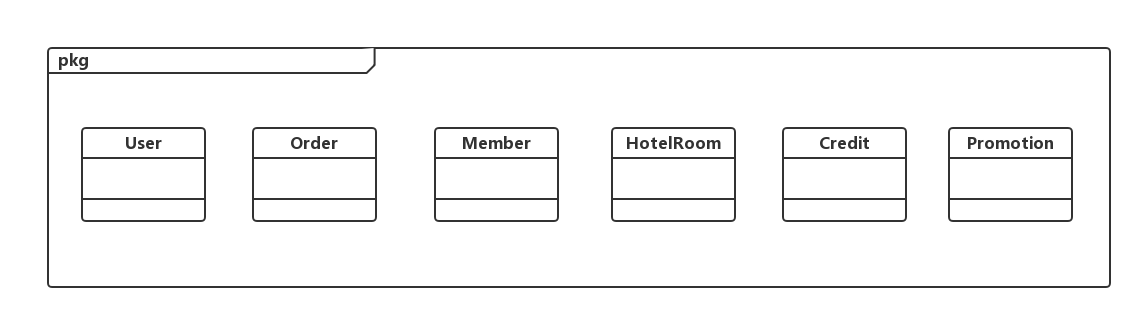


图5.3.1 业务逻辑层的设计

**5.3.1 业务逻辑层模块的职责**

业务逻辑层模块的职责如表5.3.1.1所示

|  |  |
| --- | --- |
| 模块 | 职责 |
| userbl | 负责实现对应于登录界面所需要的服务 |
| orderbl | 负责实现订单界面所需要的服务 |
| hotelroombl | 负责实现客房管理界面所需要的服务 |
| memberbl | 负责实现会员管理界面所需要的服务 |
| promotionbl | 负责实现销售策略界面所需要的服务 |
| creditbl | 负责实现信用管理界面所需要的服务 |

表5.3.1.1 业务逻辑层模块的职责

**5.3.2业务逻辑层模块的接口规范**

表5.3.2.1 userbl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| User.login | 语法 | public ResultMessage login(long id, String password); |
| 前置条件 | password符合输入规则 |
| 后置条件 | 系统查找是否存在相应的User，根据输入的password返回登录验证的结果 |
| User.views | 语法 | public ArrayList<Hotel> views(String address,String commercialDistrict); |
| 前置条件 | address,commercialDistrict符合输入规则 |
| 后置条件 | 系统根据客户输入的地址商圈查找相应范围内的酒店，返回查找的结果 |
| User.getCredit | 语法 | public ResultMessage getCredit(long id); |
| 前置条件 | 客户已登录 |
| 后置条件 | 根据输入的id返回信用值 |
| User.modifyMessage | 语法 | public void modifyMessage(long id); |
| 前置条件 | 无 |
| 后置条件 | 系统更新客户个人信息 |
| User.modifyHotelMessage | 语法 | public ResultMessage modifyHotelMessage(long id,Hotel hotel); |
| 前置条件 | 酒店工作人员已登录 |
| 后置条件 | 系统根据hotel，返回酒店基本信息 |
| User.Administration | 语法 | public void Administration(long id); |
| 前置条件 | 网站管理人员已登录 |
| 后置条件 | 系统更新用户信息 |
| User.modifyPassword | 语法 | public void modifyPassword(long id); |
| 前置条件 | 无 |
| 后置条件 | 系统更新用户密码 |
| User.cancel | 语法 | public void cancel(UserVO vo); |
| 前置条件 | 无 |
| 后置条件 | 系统持续化更新用户信息 |
| User.filter | 语法 | public ArrayList<Hotel> filter(String condition); |
| 前置条件 | 客户已输入地址商圈 |
| 后置条件 | 系统返回按condition排列的酒店信息 |
| User.sort | 语法 | public ArrayList<Hotel>sort(ArrayList<Hotel>); |
| 前置条件 | 无 |
| 后置条件 | 系统返回排序后的酒店基本信息 |
| User.viewDetail | 语法 | public ResultMessage viewDetail(Hotel hotel); |
| 前置条件 | 无 |
| 后置条件 | 系统返回酒店详细信息 |
| User.endSearch | 语法 | public void endSearch(UserVO vo); |
| 前置条件 | 无 |
| 后置条件 | 系统结束搜索酒店，持久化更新涉及的领域对象的数据 |
| User.add | 语法 | public ResultMessage add(long id,String password); |
| 前置条件 | 网站管理人员已登录 |
| 后置条件 | 系统持久化更新用户信息 |
| User.getOrderHistory | 语法 | public ResultMessage getOrderHistory (long id); |
| 前置条件 | 客户已登录 |
| 后置条件 | 根据输入的id返回酒店预订记录 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| DatabaseFacory.getUserDatabase | 得到User数据库的服务的引用 | |
| UserDataService.find(long id) | 根据用户ID获得单一持久化对象 | |
| UserDataService.finds(String address,String commercialDistrict) | 根据地址、商圈获得多个酒店对象 | |
| UserDataService.findbycondition(UserPO po) | 根据条件获得多个酒店对象 | |
| 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.finds(String address,String commercialDistrict) | 根据获得用户对象 | |

表5.3.2.2 orderbl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Order.Cancellation | 语法 | Public void Cancellation(OrderVo vo) |
| 前置条件 | 无 |
| 后置条件 | 系统持久化修改该订单对象的状态 |
| Order.CancelTime | 语法 | Public Time CancelTime(OrderVo vo) |
| 前置条件 | 无 |
| 后置条件 | 系统计算下订单到现在的时间差并返回 |
| Order.CancelKillCredit | 语法 | Public void CancelKillCredit(OrderVo vo) |
| 前置条件 | 无 |
| 后置条件 | 系统持久化修改该订单对应用户的信用值 |
| Order.Assess | 语法 | Public ResultMessage Assess(int score,String comment，OrderVo vo) |
| 前置条件 | 已输入评分/评语并提交 |
| 后置条件 | 系统持久化更新该订单对应酒店的信息 |
| Order.Add | 语法 | Public ResultMessage Add(OrderVo vo) |
| 前置条件 | 客户输入订单信息并且格式正确 |
| 后置条件 | 系统持久化更新该订单的用户信息 |
| Order.CreditCheck | 语法 | Public ResultMessage CreditCheck(OrderVo vo) |
| 前置条件 | 已经进入订单生成环节 |
| 后置条件 | 系统根据用户的信用值返回是否通过 |
| Order.CalPromotion | 语法 | Public PromotionVo CalPromotion(int id) |
| 前置条件 | 输入的id符合规范 |
| 后置条件 | 系统返回该客户满足的最优优惠政策 |
| Order.CalPrice | 语法 | Public OrderVo CalPrice(PromotionVo vo1,OrderVo vo2) |
| 前置条件 | 无 |
| 后置条件 | 系统根据优惠政策计算促销后的价格 |
| Order.Solve | 语法 | Public ResultMessage Solve(OrderVo vo) |
| 前置条件 | 无 |
| 后置条件 | 系统持久化更新该订单的信息 |
| Order.ComplainDeal | 语法 | Public ResultMessage Complain(OrderVo vo) |
| 前置条件 | 无 |
| 后置条件 | 系统持久化更新该订单的用户信息 |
| Order.ViewByID | 语法 | Public Order ViewByID (int id) |
| 前置条件 | 输入的订单id符合输入规范 |
| 后置条件 | 系统返回订单的信息 |
| Order.Views | 语法 | Public ArrayList<Order> Views(int id) |
| 前置条件 | 输入的用户id符合输入规范 |
| 后置条件 | 系统返回订单列表 |
| Order.ViewByCustom | 语法 | Public ArrayList<Order> ViewByCustom(int id) |
| 前置条件 | 输入的id符合输入规范 |
| 后置条件 | 系统返回客户的订单列表 |
| Order.ViewByDaily | 语法 | Public ArrayList<Order> ViewByDaily(Date date) |
| 前置条件 | 输入的日期符合输入规范 |
| 后置条件 | 系统返回对应日期的订单列表 |
| Order.ViewByState | 语法 | Public ArrayList<Order> ViewByState(OrderState state) |
| 前置条件 | 输入的订单状态符合输入规范 |
| 后置条件 | 系统返回对应状态的订单列表 |
| Order.TimeSort | 语法 | Public ArrayList<Order> TimeSort ( ArrayList <Order> ) |
| 前置条件 | 无 |
| 后置条件 | 系统返回按时间排序的订单列表 |
| Order.CreateID | 语法 | Public String CreateID() |
| 前置条件 | 无 |
| 后置条件 | 系统结合当前时间创建一个订单号 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| UserDataService.find(int id) | 根据用户ID获得单一持久化对象 | |
| OrderDataService.find(int id) | 根据订单号ID获得单一持久化对象 | |
| OrderDataService.findbydaily(int id) | 根据时间得到订单对象 | |
| OrderDataService.findbycustom(Date date) | 根据用户ID获得订单对象 | |
| OrderDataService.finds(String hotel,UserPO po) | 根据酒店以及用户得到订单对象 | |
| OrderDataService.findbystate(State state) | 得到根据状态的订单对象 | |
| OrderDataService.delete(OrderPo po) | 删除单一持久化对象 | |
| OrderDataService.updata(OrderPo po) | 更新单一持久化对象 | |
| OrderDataService.insert(OrderPo po) | 插入单一持久化对象 | |
| OrderDataService.finds |  | |
| DatabaseFacory.getOrderDatabase | 得到Order数据库的服务的引用 | |
| UserDataService.update(UserPo po) | 更新单一持久化对象 | |
| HotelRoomDataService.update(HotelRoomPo po) | 更新单一持久化对象 | |
| CreditDataService.find(CreditPo po) | 得到单一持久化对象 | |

表5.3.2.3 hotelroombl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hotel.Addroom | 语法 | Public ResultMessage Addroom (int room) |
| 前置条件 | 启动一个录入客房的回合 |
| 后置条件 | 在一个录入客房的回合中，增加可用客房 |
| Hotel.Changnumber | 语法 | Public ResultMessage Changenumber (int roomnumber) |
| 前置条件 | 启动一个录入客房的回合 |
| 后置条件 | 在一个录入客房的回合中，增加可用客房数量 |
| Hotel.Updateroom | 语法 | Public void Ppdateroom () |
| 前置条件 | 启动一个录入客房的回合 |
| 后置条件 | 结束此次录入客房，持久化更新涉及的领域对象的数据 |
| Hotel.Checkin | 语法 | Public ResultMessage Checkin (HotelroomVo vo) |
| 前置条件 | 无 |
| 后置条件 | 系统持久化更新入住信息 |
| Hotel.Timein | 语法 | Public ResultMessage Timein (Date datein) |
| 前置条件 | 开始更新入住信息 |
| 后置条件 | 系统更新房间入住的日期 |
| Hotel.IDin | 语法 | Public ResultMessage IDin (int id) |
| 前置条件 | 开始更新入住信息 |
| 后置条件 | 系统记录入住房间的房间号 |
| Hotel.TimeOutpro | 语法 | Public ResultMessage TimeOutpro (Date dateoutpro) |
| 前置条件 | 开始更新入住信息 |
| 后置条件 | 系统记录该房的预计离开时间 |
| Hotel.Timeout | 语法 | Public ResultMessage Timeout (Date timeout) |
| 前置条件 | 开始更新退房信息 |
| 后置条件 | 系统更新退房日期 |
| Hotel.IDout | 语法 | Public ResultMessage IDout (int id) |
| 前置条件 | 开始更新退房信息 |
| 后置条件 | 系统记录退房的房间号 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| HotelroomDataService.find(int id) | 根据房间号进行查找单一持久化对象 | |
| HotelroomDataService.find (State state) | 根据状态进行查找多个持久化对象 | |
| HotelroomDataService.insert(HotelroomPO po) | 插入单一持久化对象 | |
| HotelroomDataSerivce.delete(HotelroomPO po) | 删除单一持久化对象 | |
| HotelroomDataService.update(HotelroomPO po) | 更新单一持久化对象 | |
| DatabaseFacory.getHotelroomDatabase | 得到Hotelroom数据库的服务的引用 | |

表5.3.2.4 Promotionbl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Promotion. createNewItem | 语法 | public void createItem(ItemVo vo) |
| 前置条件 | 无 |
| 后置条件 | 更新营销策略列表 |
| Promotion.Del  这个和update是单纯的功能需求接口。。。 | 语法 | public void Del(promotion vo) |
| 前置条件 | 已添加营销策略 |
| 后置条件 | 系统更新营销策略列表 |
| Promotion. update | 语法 | public void update (promotion vo) |
| 前置条件 | 已添加营销策略 |
| 后置条件 | 系统修改营销策略信息 |
| Promotion.getCalulate | 语法 | public double getCalulate(double originalPrice) |
| 前置条件 | 已添加订单房间信息 |
| 后置条件 | 返回经营销策略优惠后的订单价格 |
| 需要的接口（需接口） | | |
| DatabaseFacory.getPromotionDatabase | 得到promotion数据库的服务的引用 | |
| PromotionDataService.insert(PromotionPO po) | 插入单一持久化对象 | |
| PromotionDataService.delete(PromotionPO po) | 删除单一持久化对象 | |
| PromotionDataService.update(PromotionPO po) | 更新单一持久化对象 | |
| PromotionDataService.show(PromotionPO po) | 得到所有的单一持久化对象 | |

表5.3.2.5 Memberbl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Member.createByPersonal | 语法 | public void createByPersonal (long id, string birthday) |
| 前置条件 | 已添加客户信息 |
| 后置条件 | 增加该客户个人会员信息 |
| Member.createByBusiness | 语法 | public void createByBusiness (long id, string name) |
| 前置条件 | 已添加客户信息 |
| 后置条件 | 增加该客户企业会员信息 |
| Member.calRank | 语法 | public int calRank(long id, double credit) |
| 前置条件 | 已添加会员信息、信用信息 |
| 后置条件 | 根据会员信用值多少，计算会员等级 |
| 需要的接口（需接口） | | |
| DatabaseFacory.getMemberDatabase | 得到member数据库的服务的引用 | |
| 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 | |
| MemberDataService.show(MemberPO po) | 得到所有的单一持久化对象 | |
| MemberDataService.find(int id) | 根据ID进行查找单一持久化对象 | |

表5.3.2.6 Creditbl模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Credit.addValue | 语法 | public void addValue (Credit vo) |
| 前置条件 | 已添加客户信息 |
| 后置条件 | 增加客户信用值 |
| 需要的接口（需接口） | | |
| DatabaseFacory.getCreditDatabase | 得到Credit数据库的服务的引用 | |
| CreditDataService.insert(CreditPO po) | 插入单一持久化对象 | |
| CreditDataService.delete(CreditPO po) | 删除单一持久化对象 | |
| CreditDataService.update(CreditPO po) | 更新单一持久化对象 | |
| CreditDataService.find(int id) | 根据ID进行查找单一持久化对象 | |

**5.4数据层的分解**

数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增、删、改、查。各业务逻辑需要的服务由各DataService接口提供。持久化数据的保存采用数据库形式进行保存

数据层模块的描述具体如图5.4.1所示(以orderdataservice为例)

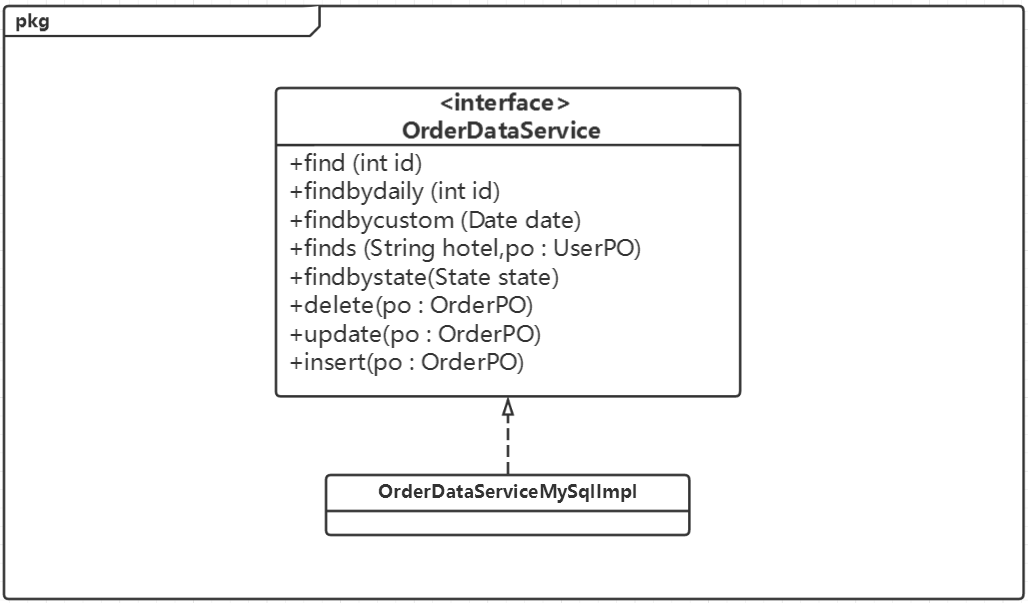


图5.4.1

**5.4.1数据层模块的职责**

表5.4.1.1数据层模块(UserDataService)的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| UserDataService | 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、查服务 |

表5.4.1.2数据层模块(MemberDataService)的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| MemberDataService | 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 |
| MemberDataServiceMySqlImpl | 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、查服务 |

表5.4.1.3数据层模块(OrderDataService)的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| OrderDataService | 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、查服务 |

表5.4.1.4数据层模块(CreditDataService)的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| OrderDataService | 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、查服务 |

表5.4.1.5数据层模块(HotelroomDataService)的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| HotelroomDataService | 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 |
| HotelroomDataServiceMySqlImpl | 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、查服务 |

表5.4.1.6数据层模块(PromotionDataService)的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| PromotionDataService | 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、查服务 |

**5.4.2数据层模块的接口规范**

表5.4.2.1MemberDataService的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| MemberDataService.find | 语法 | public MemberPO find (long 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.insert | 语法 | public void insert (MemberPO po) |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| MemberDataService.delete | 语法 | public void delete(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.finish | 语法 | public void finish() |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库 |

表5.4.2.2 CreditDataService的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CreditDataService.find | 语法 | public CreditPO find (long id) |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的CreditPO结果 |
| CreditDataService.insert | 语法 | public void insert (CreditPO po) |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| CreditDataService.delete | 语法 | public void delete(CreditPO 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 |
| CreditDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| CreditDataService.finish | 语法 | public void finish() |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库 |

表5.4.2.3 PromotionDataService的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionDataService.show | 语法 | Public void show (PromotionPO po) |
| 前置条件 | 无 |
| 后置条件 | 显示所有PromotionPO 的信息列表 |
| PromotionDataService.insert | 语法 | public void insert (PromotionPO po) |
| 前置条件 | 同样ID的po在Mapper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) |
| 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.finish | 语法 | public void finish() |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库 |

表5.4.2.4 UserDataService的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserDataService.show | 语法 | Public void show(UserPo po) |
| 前置条件 | 无 |
| 后置条件 | 显示所有UserPo记录 |
| UserDataService.find | 语法 | Public UserPo find(int id) |
| 前置条件 | 无 |
| 后置条件 | 根据ID返回对应的UserPo |
| UserDataService.insert | 语法 | Public void insert(UserPo po) |
| 前置条件 | 同样的po在Mapper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) |
| 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() |
| 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() |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库 |

表5.4.2.5 OrderDataService的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderDataService.show | 语法 | Public void show(OrderPo po) |
| 前置条件 | 无 |
| 后置条件 | 显示所有OrderPo记录 |
| OrderDataService.find | 语法 | Public OrderPo find(int id) |
| 前置条件 | 无 |
| 后置条件 | 根据ID返回对应的OrderPo |
| OrderDataService.insert | 语法 | Public void insert(OrderPo po) |
| 前置条件 | 同样的po在Mapper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) |
| 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.finish | 语法 | Public void finish() |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库 |

表5.4.2.6 HotelRoomDataService的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelRoomDataService.show | 语法 | Public void show(HotelRoomPo po) |
| 前置条件 | 无 |
| 后置条件 | 显示所有HotelRoomPo记录 |
| HotelRoomDataService.find | 语法 | Public HotelRoomPo find(int id) |
| 前置条件 | 无 |
| 后置条件 | 根据ID返回对应的HotelRoomPo |
| HotelRoomDataService.delete | 语法 | Public void delete(HotelRoomPo 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 |
| HotelRoomDataService.insert | 语法 | Public void insert(HotelRoomPo po) |
| 前置条件 | 同样的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| HotelRoomDataService.init | 语法 | Public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久化数据库 |
| HotelRoomDataService.finish | 语法 | Public void finish() |
| 前置条件 | 无 |
| 后置条件 | 结束持久化数据库 |

**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，系统的PO类如下所示：

·UserPO：用户类包含用户的账号、密码、联系方式、是否会员

·OrderPO：订单类包含订单的订单号、订单状态、订单开始时间、折扣、订单价格、下订单的客户账号、酒店名称、房间号

·HotelRoomPO：客房类包含酒店客房的所属酒店名称、商圈、地址、星级、客房的类型、房间号、是否占用、入住的时间、退房的时间、房间的基础价格

·MemberPO：会员类包括会员的生日/企业名称、账号、信用等级

·CreditPO：信用值类包含客户的信用值、信用值变更记录(包括增加或减少的数量、时间、变更原因)

·PromotionPO：销售策略类包含销售策略的折扣内容、适用的时间、折扣的额度、适用的酒店

持久化用户对象UserPO的定义如图所示

public class UserPO implements Serializable{

private int id;

private String password;

private String phone;

private boolean ifVIP=false;

public UserPO (int i; String pa; String ph){

this.id=i;

this.password=pa;

this.phone=ph;

}

public int getID(){

return id;

}

public String getPassword(){

return password;

}

public String setPassword(String pa){

this.password=pa;

}

public String getPhone(){

return phone;

}

public String setPhone(String ph){

this.phone=ph;

}

public setifVIP(boolean b){

this.ifVIP=b;

}

}

持久化订单对象OrderPo的定义如图所示

public class OrderPO implements Serializable {

private long orderID;

private OrderState state;

private Date date;

private Promotion promotion;

private double orderPrice;

private int id;

private String hotel;

private String roomnum;

public OrderPO(long id, Order state, Date date) {

this. orderID = id;

this. state = state;

this.date = date;

}

public long getOrderID() {

return orderID;

}

public OrderState getState() {

return state;

}

public OrderState setState(OrderState state) {

this.state = state;

}

public Date getDate() {

return date;

}

public Date setDate(Date date) {

this.date = date;

}

public Promotion getPromotion(){

return promotion;

}

public Promotion setPromotion(Promotion promotion){  
 this.promotion = promotion;

}

public double getOrderPrice(){

return orderPrice;

}  
 public double setOrderPrice(double price){  
 this.orderPrice = price;

}  
 public int getID(){  
 return id;

}

public String getHotel(){  
 return hotel;

}

public String setHotel(String hotel){  
 this.hotel = hotel;

}

public String getRoomNum(){  
 return roomnum;

}

public String setRoomNum(String room){

this.roomnum = room;

}

}

**6.2数据库表**

数据库中包含User表、Member表、Hotel表、Hotelroom表、Promtion表