|  |
| --- |
| 互联网酒店管理系统  Hotel Property Management System |
| 软件详细设计描述文档 |
| V2.0 |

|  |
| --- |
| 焦立群、金灵益、黎奥邦、李恩铭  2017-1-1 |

更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| **黎奥邦** | 2016-10-28 | 最初文档模板 | V0.0 |
| **黎奥邦** | 2016-10-29 | 初步完成文档，整合了李恩铭以及金灵益的贡献内容，主要完成了业务逻辑层的编写 | V0.1 |
| **焦立群** | 2016-11-1 | 一些微小的工作，完成了ui层的分解初稿 | V0.2 |
| **金灵益** | 2016-11-6 | 完成了数据层的分解编写 | V1.0 |
| **焦立群** | 2016-11-13 | 界面层完善，order模块有较大改动 | V1.1 |
| **金灵益** | 2016-11-13 | 增加HotelRoom、HotelRoomList接口规范，修改酒店顺序图，增加HotelList接口内多个方法 | V1.2 |
| **黎奥邦** | 2016-11-13 | 删除editDiscount里的参数为int的重载 | V1.7 |
| **李恩铭** | 2016-12-05 | 在编码阶段对供需接口方法进行了改善设计，由李恩铭整合修改了各个接口的表格 | V1.8 |
| **金灵益** | 2017-01-01 | 项目完成之前对所有文档进行回顾，进行了全面修正和完善 | V2.0最终版 |

目录

[更新历史 - 1 -](#_Toc471053579)

[1.引言 - 3 -](#_Toc471053580)

[1.1编制目的 - 3 -](#_Toc471053581)

[1.2词汇表 - 3 -](#_Toc471053582)

[1.3参考资料 - 3 -](#_Toc471053583)

[2.产品概述 - 3 -](#_Toc471053584)

[3.体系结构设计描述 - 3 -](#_Toc471053585)

[4.结构视角 - 3 -](#_Toc471053586)

[4.1用户界面层的分解 - 3 -](#_Toc471053587)

[4.1.1UI层设计概要 - 3 -](#_Toc471053588)

[4.1.2UI层内部包的职责与划分 - 4 -](#_Toc471053589)

[4.1.3界面跳转示意 - 5 -](#_Toc471053590)

[4.2业务逻辑层的分解 - 5 -](#_Toc471053591)

[4.2.1 Orderbl模块 - 5 -](#_Toc471053592)

[4.2.2 Memberbl模块 - 12 -](#_Toc471053593)

[4.2.3 Hotelbl模块 - 15 -](#_Toc471053594)

[4.2.4 Accountbl模块 - 20 -](#_Toc471053595)

[4.2.5 HotelPromotionbl模块 - 24 -](#_Toc471053596)

[4.2.6 MarketPromotionbl模块 - 27 -](#_Toc471053597)

[4.2.7 Creditbl模块 - 30 -](#_Toc471053598)

[4.3数据层的分解 - 32 -](#_Toc471053599)

[4.3.1.DataFactoryService - 32 -](#_Toc471053600)

[4.3.2.OrderDataService - 33 -](#_Toc471053601)

[4.3.3.MemberDataService - 34 -](#_Toc471053602)

[4.3.4.HotelDataService - 35 -](#_Toc471053603)

[4.3.5.AccountDataService - 36 -](#_Toc471053604)

[4.3.6.HotelPromotionDataService - 37 -](#_Toc471053605)

[4.3.7.MarketPromotionDataService - 38 -](#_Toc471053606)

[4.3.8.CreditDataService - 39 -](#_Toc471053607)

[5.依赖视角 - 39 -](#_Toc471053608)

1.引言

1.1编制目的

本报告详细完成对互联网酒店管理系统的详细设计，达到指导后续软件构造目的，同时实现和测试人员和用户的沟通。

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。

1.2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 |
| **Order** | 订单 | 无 |
| **Member** | 客户信息 | 企业、普通客户 |
| **Hotel** | 酒店 | 无 |
| **Account** | 账户 | 无 |
| **HotelPromotion** | 酒店促销策略 | 无 |
| **MarketPromotion** | 网站促销策略 | 无 |
| **Credit** | 客户信用值 | 无 |

1.3参考资料

1. 互联网酒店管理系统HPMS用例文档\_V2.0

2. 互联网酒店管理系统HPMS软件需求规格说明\_V2.0

3. 互联网酒店管理系统HPMS软件体系结构描述文档\_V2.0

2.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文档中对产品的概括描述。

3.体系结构设计描述

参考互联网酒店管理系统概要设计文档中对体系结构设计的概述。

4.结构视角

4.1用户界面层的分解

4.1.1UI层设计概要

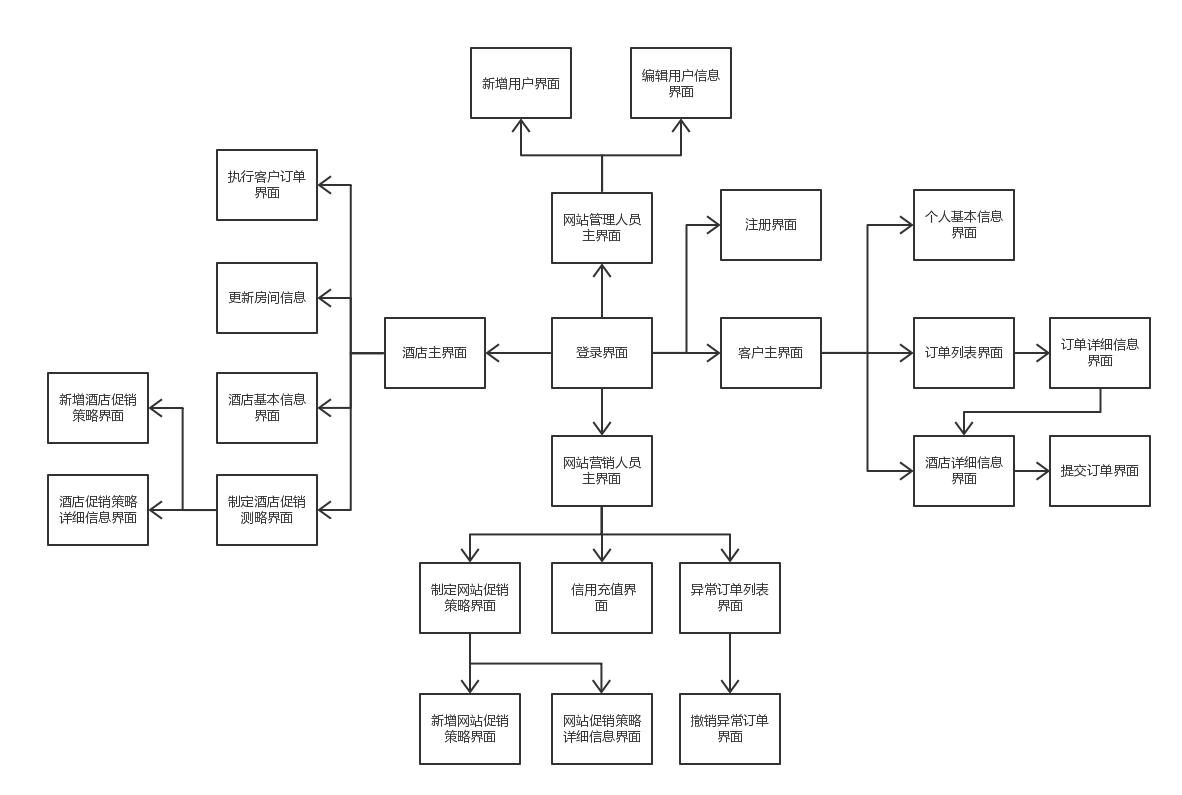
UI层主要以用户的性质来划分，包括会员、酒店工作人员、网站管理人员、网站营销人员。

UI层的入口是mainframe，显示的是打开系统的login界面，然后通过不同权限的用户账户登录，跳转到不同用户的ui包内，之后不同用户所对应的各个功能的界面可以最大程度在包内跳转，更好地实现高内聚、低耦合。

4.1.2UI层内部包的职责与划分

|  |  |  |  |
| --- | --- | --- | --- |
| **模块** | **包名** | **类名** | **功能** |
| **Common** | Common | SingletonItem | 创建一个单例，实现界面间参数的传递 |
| ViewFxmlPath | 保存所有的fxml文件路径 |
| **Controller** | Controller | HotelController | Hotel界面控制器 |
| MemberController | Member界面控制器 |
| MarketerController | Marketer界面控制器 |
| ManagerController | Manager界面控制器 |
| SignViewController | 登录、登出、注册界面控制器 |
| Controller.Service | HotelControllerService | Hotel界面控制器接口 |
| MemberControllerService | Member界面控制器接口 |
| MarketerControllerService | Marketer界面控制器接口 |
| ManagerControllerService | Manager界面控制器接口 |
| SignViewControllerService | 处理登录、登出、注册事件 |
| Controller.unity | Credit | 信用类型的property类，用于tableview的数据导入 |
| Hotel | 酒店类型的property类，用于tableview的数据导入 |
| Order | 订单类型的property类，用于tableview的数据导入 |
| Promotion | 促销策略类型的property类，用于tableview的数据导入 |
| **Util** | Alert | AlertPopUp | 提示弹出窗口 |
| Controller | PanelController | 界面跳转控制器 |
| State | ButtonState | 按钮状态 |
| **View** | view.member | MemberRootView | 用户主界面 |
| MemberInfoView | 查看/修改个人信息界面 |
| MemberOrderInfoView | 订单详细信息界面 |
| MemberOrderView | 客户订单主界面 |
| MemberOrderComment | 订单评价界面 |
| MemberHotelInfo | 酒店详细信息界面 |
| MemberHotelListView | 搜索酒店界面 |
| MemberHotelOrderView | 提交订单界面 |
| MemberCreditView | 客户查看信用记录界面 |
| view.hotel | HotelRootView | 酒店主界面 |
| HotelInfoView | 查看/修改酒店信息界面 |
| HotelOrderExeView | 执行订单界面 |
| HotelOrderView | 查看酒店订单界面 |
| HotelProAddView | 增加促销策略界面 |
| HotelProInfoView | 查看/修改促销策略界面 |
| HotelPromotionView | 酒店促销策略界面 |
| HotelRoomView | 查看/修改酒店房间界面 |
| view.marketer | MarketerRootView | 网站营销人员界面 |
| MarketerPromotionView | 网站促销策略列表界面 |
| MarketerProAddView | 新增策略界面 |
| MarketerProInfoView | 删改查策略界面 |
| MarketerCreditView | 信用充值界面 |
| MarketerOrderRevokeView | 撤销异常订单界面 |
| MarketerOrderView | 异常订单列表界面 |
| view.manager | ManagerRootView | 网站管理人员主界面 |
| ManagerUserAddView | 新增用户界面 |
| ManagerUserEditView | 删改用户信息界面 |
| ManagerUserSearchView | 搜索用户界面 |
| view.sign | SignInView | 登录主界面 |
| view.unity | PromotionView | 公共促销策略界面 |

4.1.3界面跳转示意



各界面跳转示意图

4.2业务逻辑层的分解

4.2.1 Orderbl模块

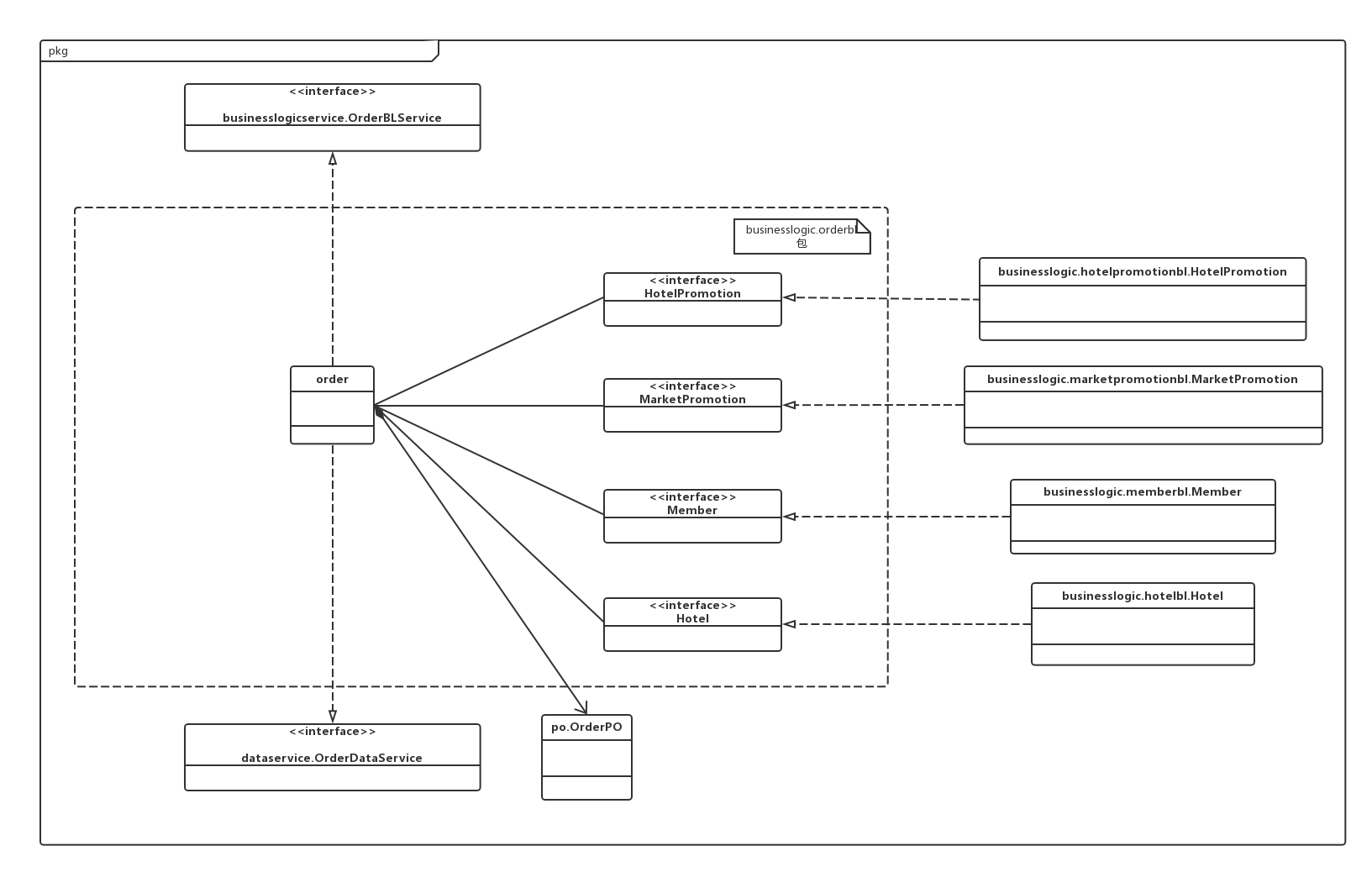
（1）模块概述

Orderbl模块承担的具体功能需求和非功能需求可参见需求规格说明文档；Orderbl模块的职责及接口参见体系结构设计文档。

（2）整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如展示层和业务逻辑层之间，我们添加businesslogicservice.OrderBLService接口。业务逻辑层和数据层之间添加dataservice.OrderDataService接口。OrderPo是作为订单的持久化对象被添加到设计模型中去的。

Order模块的设计如下图所示。



Order模块各个类的职责如下表所示。

|  |  |
| --- | --- |
| **模块** | **职责** |
| **Order** | 订单的领域模型对象，拥有订单的信息，可以帮助完成订单管理所需要的服务 |

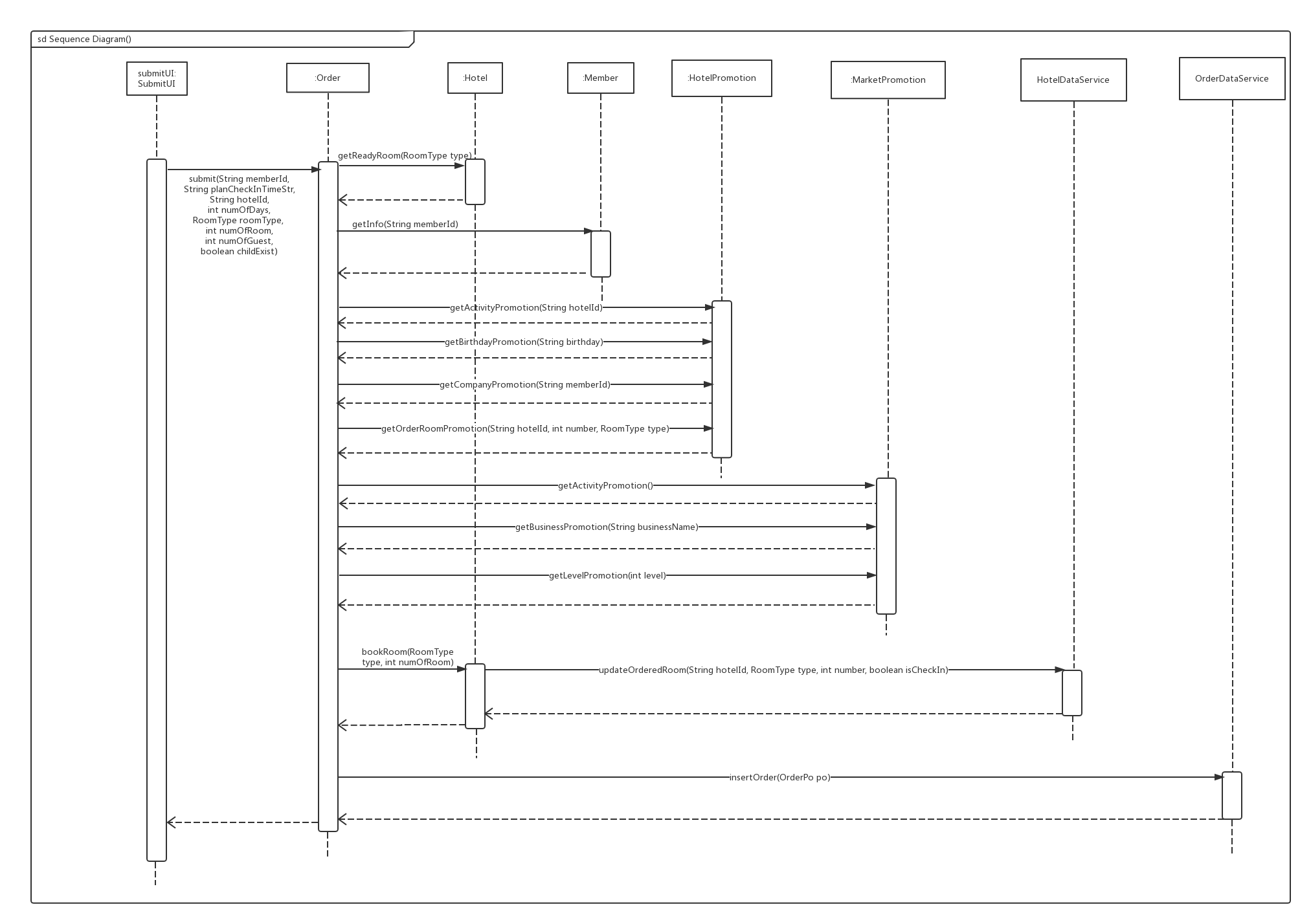
（3）模块内部类的接口规范

**Order**的接口规范

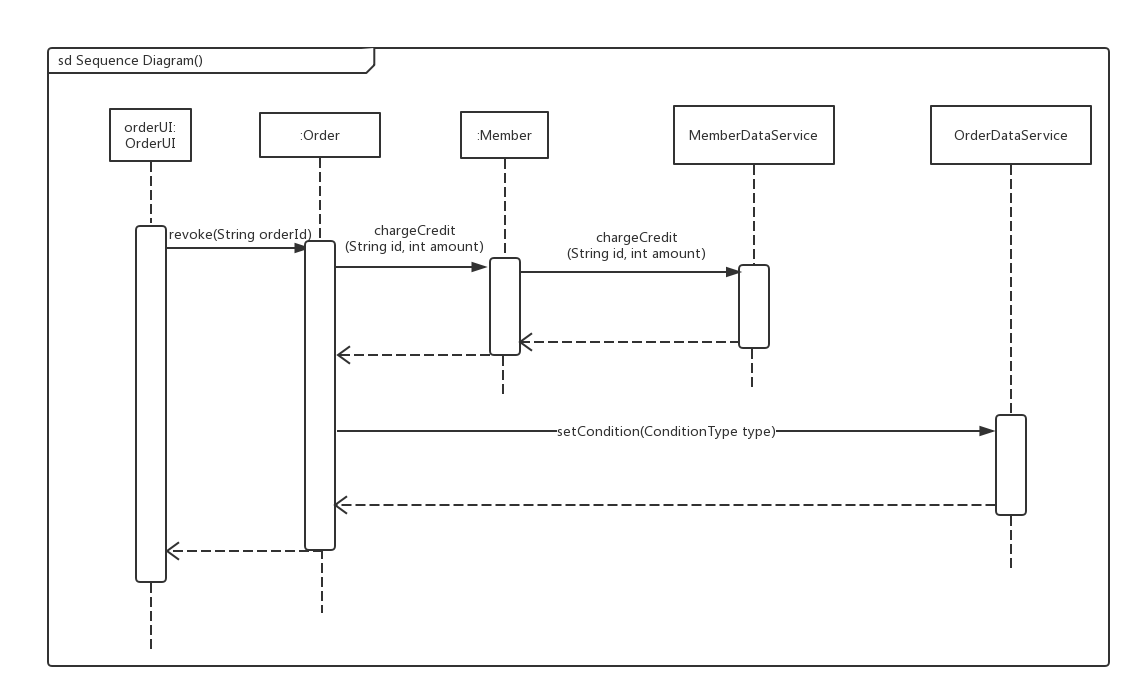
|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| **Order.getOrder** | 语法 | Public OrderVo getOrderVo(String orderId) | |
| 前置条件 | 客户获取订单 | |
| 后置条件 | 无 | |
| **Order.getOrderId** | 语法 | Public String getOrderId(String orderId) | |
| 前置条件 | 客户获取订单编号 | |
| 后置条件 | 无 | |
| **Order.getOrderCondition** | 语法 | Public OrderCondition getOrderCondition(String orderId) | |
| 前置条件 | 客户获取订单状态 | |
| 后置条件 | 无 | |
| **Order.getMemberId** | 语法 | public String getMemberId(String orderId); | |
| 前置条件 | 获取订单上的用户ID | |
| 后置条件 | 无 | |
| **Order.getMemberName** | 语法 | public String getMemberName(String orderId); | |
| 前置条件 | 获取订单上的用户名字 | |
| 后置条件 | 无 | |
| **Order.getCreateTime** | 语法 | public Date getCreateTime(String orderId); | |
| 前置条件 | 获取订单的生成时间 | |
| 后置条件 | 无 | |
| **Order.getCheckInTime** | 语法 | public Date getCheckInTime(String orderId); | |
| 前置条件 | 获取订单的执行时间 | |
| 后置条件 | 无 | |
| **Order.getCheckOutTime** | 语法 | public Date getCheckOutTime(String orderId); | |
| 前置条件 | 获取订单的结束时间 | |
| 后置条件 | 无 | |
| **Order.getHotelName** | 语法 | public String getHotelName(String orderId); | |
| 前置条件 | 获取订单的酒店名称 | |
| 后置条件 | 无 | |
| **Order.getHotelId** | 语法 | public String getHotelId(String orderId); | |
| 前置条件 | 获取订单的酒店ID | |
| 后置条件 | 无 | |
| **Order.getRoomType** | 语法 | public RoomType getRoomType(String orderId); | |
| 前置条件 | 获取订单的房间类型 | |
| 后置条件 | 无 | |
| **Order.getNumOfRoom** | 语法 | public int getNumOfRoom(String orderId); | |
| 前置条件 | 获取订单的房间数量 | |
| 后置条件 | 无 | |
| **Order.getNumOfGuest** | 语法 | public int getNumOfGuest(String orderId); | |
| 前置条件 | 获取订单的客人数量 | |
| 后置条件 | 无 | |
| **Order.getChildExist** | 语法 | public boolean getChildExist(String orderId); | |
| 前置条件 | 获取是否有儿童 | |
| 后置条件 | 无 | |
| **Order.getOriginalPrice** | 语法 | public int getOriginalPrice(String orderId); | |
| 前置条件 | 获取订单房间的原价 | |
| 后置条件 | 无 | |
| **Order.getDiscount** | 语法 | public double getDiscount(String orderId); | |
| 前置条件 | 获取促销策略 | |
| 后置条件 | 无 | |
| **Order.getDiscountedPrice** | 语法 | public int getDiscountedPrice(String orderId); | |
| 前置条件 | 获取折后价格 | |
| 后置条件 | 无 | |
| **Order.getAbnormalOrders** | 语法 | public List<OrderVo> getAbnormalOrders() | |
| 前置条件 | 网站营销人员获取异常订单列表 | |
| 后置条件 | 获取异常订单列表 | |
| **Order.getAllOrdersByHotel** | 语法 | public List<OrderVo> getAllOrdersByHotel  (String hotelId) | |
| 前置条件 | 酒店工作人员获取所有订单 | |
| 后置条件 | 得到酒店的所有订单列表 | |
| **Order.getAllOrdersByMember** | 语法 | public List<OrderVo> getAllOrdersByMember  (String memberId) | |
| 前置条件 | 客户获取所有订单 | |
| 后置条件 | 得到客户的所有订单列表 | |
| **Order.getNotExecutedOrdersByWebsite** | 语法 | public List<OrderVo> getNotExecutedByWebsite() | |
| 前置条件 | 网站营销人员获取未执行订单列表 | |
| 后置条件 | 得到所有未执行订单列表 | |
| **Order.getOrdersInConditionByHotel** | 语法 | public List<OrderVo> get**Orders**InConditionByHotel  (String hotelId, OrderCondition condition) | |
| 前置条件 | 酒店工作人员按订单类型获取订单列表 | |
| 后置条件 | 根据订单类型获取酒店的订单列表 | |
| **Order.getOrdersInConditionByMember** | 语法 | public List<OrderVo> get**Orders**InConditionByMember  (String memberId, OrderCondition condition) | |
| 前置条件 | 客户按订单类型获取订单列表 | |
| 后置条件 | 根据订单类型获取客户的订单列表 | |
| **Order.setToAbnormal** | 语法 | public ResultMessage setToAbnormal() | |
| 前置条件 | 客户在超过最晚订单执行时间未办理入住 | |
| 后置条件 | 将订单状态置为异常，更新客户信用值 | |
| **Order.submit** | 语法 | public ResultMessage submit  (String memberId, String planCheckInTime, String hotelId, int numOfDays, RoomType roomType, int numOfRoom, int numOfGuest, Boolean childExist) | |
| 前置条件 | 客户提交订单 | |
| 后置条件 | 更新订单、客户、酒店房间信息 | |
| **Order.cancel** | 语法 | public ResultMessage cancel(String orderId) | |
| 前置条件 | 客户取消订单 | |
| 后置条件 | 修改订单类型信息 | |
| **Order.checkIn** | 语法 | public ResultMessage checkIn(String orderId) | |
| 前置条件 | 酒店工作人员执行订单 | |
| 后置条件 | 更新订单状态为正在执行，客户信用值 | |
| **Order.abnormalCheckIn** | 语法 | public ResultMessage abnormalCheckIn  (String orderId) | |
| 前置条件 | 订单状态为异常 | |
| 后置条件 | 将订单状态置为正在执行，更新客户信用值 | |
| **Order.checkOut** | 语法 | public ResultMessage checkout(String orderId) | |
| 前置条件 | 酒店工作人员为订单执行退房操作 | |
| 后置条件 | 更新订单状态为已执行 | |
| **Order.revoke** | 语法 | public ResultMessage revoke  (String orderId, double allOrHalf) | |
| 前置条件 | 网站营销人员撤销异常订单 | |
| 后置条件 | 更新订单状态为已撤销，更新客户信用值 | |
| **Order.delete** | 语法 | public ResultMessage delete(String orderId) | |
| 前置条件 | 存在该id的订单 | |
| 后置条件 | 删除订单 | |
| **Order.setToFinished** | 语法 | public ResultMessage setToFinished(String orderId) | |
| 前置条件 | 客户已执行评分操作 | |
| 后置条件 | 更新订单状态为已完成 | |
| **需要的服务（需接口）** | | | |
| 服务名 | | | 服务 |
| **DatabaseFactory.getOrderData()** | | | 得到Order数据库的服务的引用 |
| **Member.chargeCredit(String id, int amount)** | | | 修改客户信用值 |
| **Member.getInfo(String memberId)** | | | 得到客户信息 |
| **Promotion.getPromotion()** | | | 得到促销策略 |
| **Hotel.bookRoom(RoomType type, int number)** | | | 预定房间，更新订单、酒店房间信息 |
| **Hotel.checkoutRoom**  **(RoomType type, int number)** | | | 退房，更新订单、酒店房间信息 |
| **OrderDataService.inset(OrderPo po)** | | | 在数据库中增加一条po记录 |
| **OrderDataService.find(String orderId)** | | | 按ID查找返回相应的OrderPo结果 |
| **OrderDataService.update(OrderPo po)** | | | 更新一个po |
| **OrderDataService.delete(String orderId)** | | | 删除该po |
| **OrderDataService.getAllByHotel(String hotelId)** | | | 得到该酒店的所有订单 |
| **OrderDataService.getAllByMember**  **(String memberId)** | | | 得到该客户的所有订单 |
| **OrderDataService.getNotExecuted()** | | | 得到所有未执行订单 |
| **OrderDataService.getAbnormal()** | | | 得到所有异常订单 |

（4）业务逻辑层的动态模型

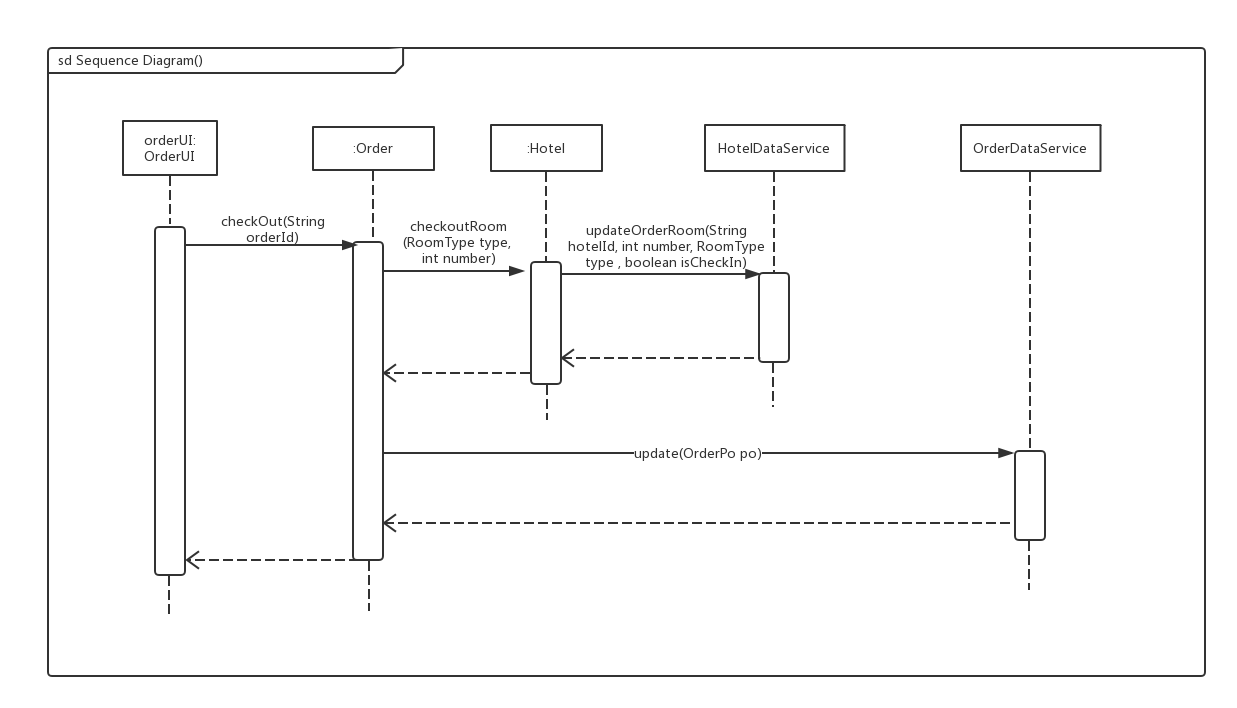
如图表明了当客户提交订单时，订单业务逻辑处理的相关对象之间的协作。



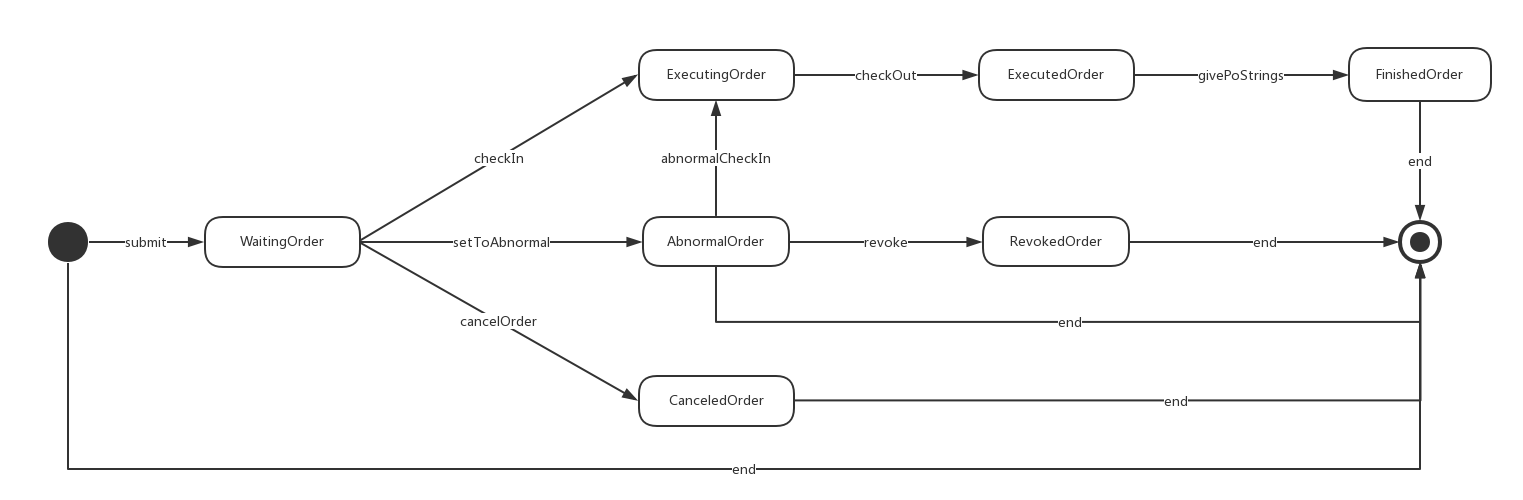
如图表明了当网站营销人员想要撤销异常订单时，订单业务逻辑处理的相关对象之间的协作。



如图表明了当酒店工作人员为订单执行退房时，订单业务逻辑处理的相关对象之间的协作。



如图所示的状态图描述了order对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。



4.2.2 Memberbl模块

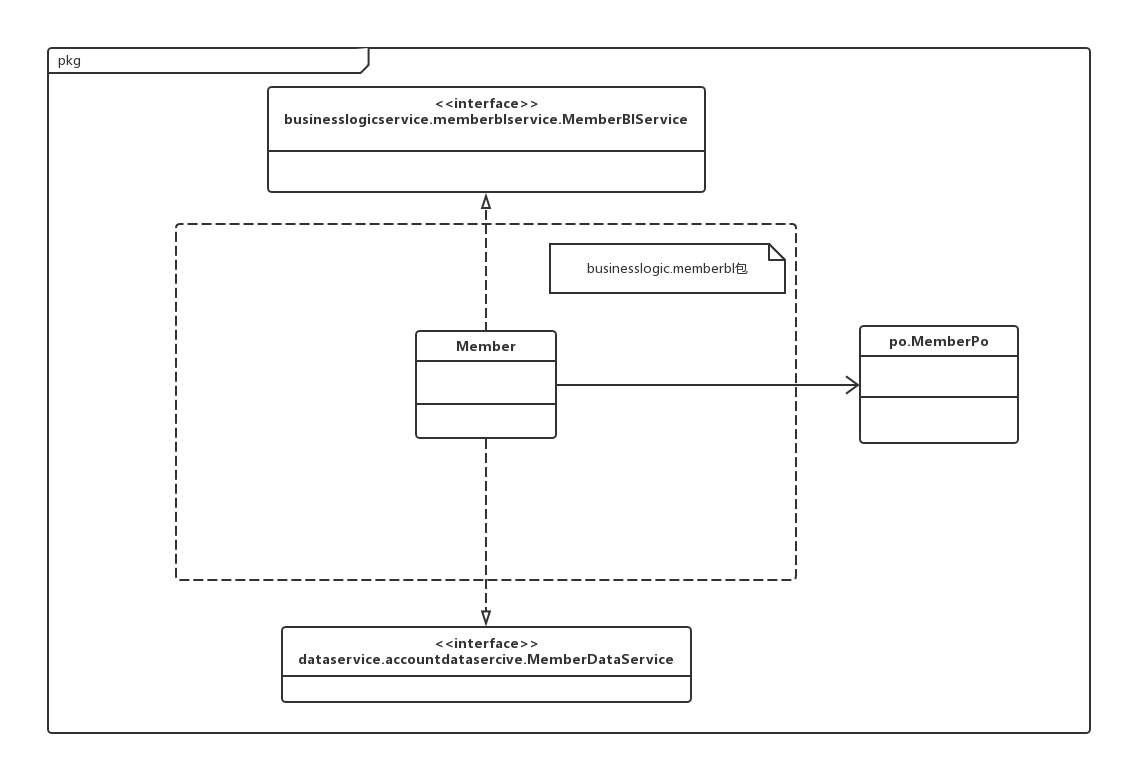
（1）模块概述

Memberbl模块负责实现对客户的账号信息的管理，客户可以对自己的账号信息进行查询，酒店可以对每一个客户所有的信息进行查看。具体功能需求和非功能需求可参见需求规格说明文档和体系结构设计文档。

（2）整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如展示层和业务逻辑层之间，我们添加bussinesslogicservice.MemberBLService接口。业务逻辑层和数据层之间添加dataservice.MemberDataService接口。MemberPo是作为账号信息的持久化对象被添加到设计模型中去的。

Memberbl模块的设计如下图所示



Memberbl模块各个类的职责如下表所示。

|  |  |
| --- | --- |
| **模块** | **职责** |
| **Member** | 客户信息的领域模型对象，拥有客户账号的信息，可以帮助完成账号查询、信用充值和酒店评价界面所需要的服务 |

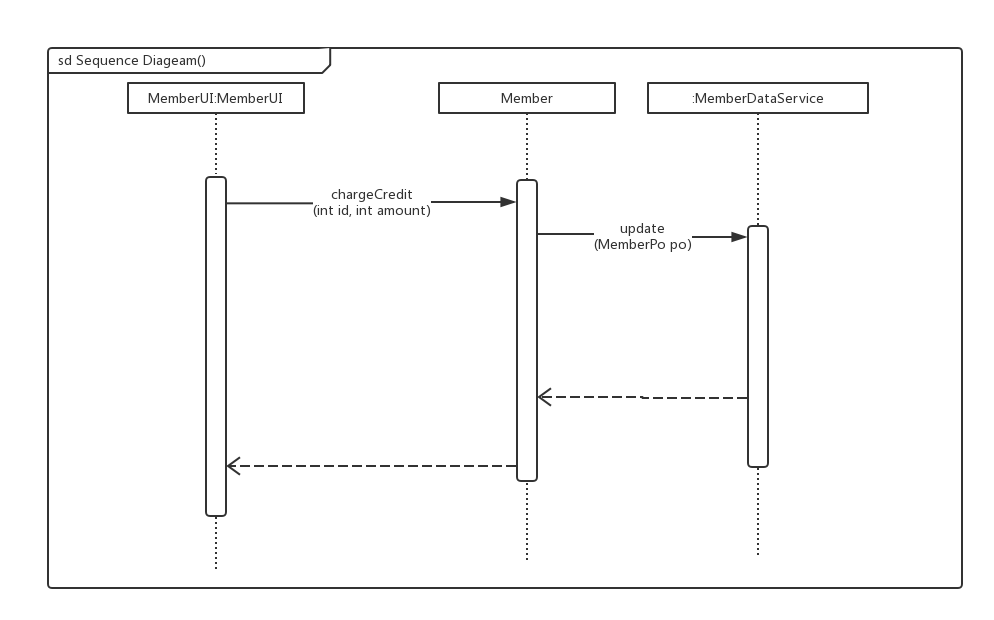
（3）模块内部类的接口规范

**Member** 的接口规范

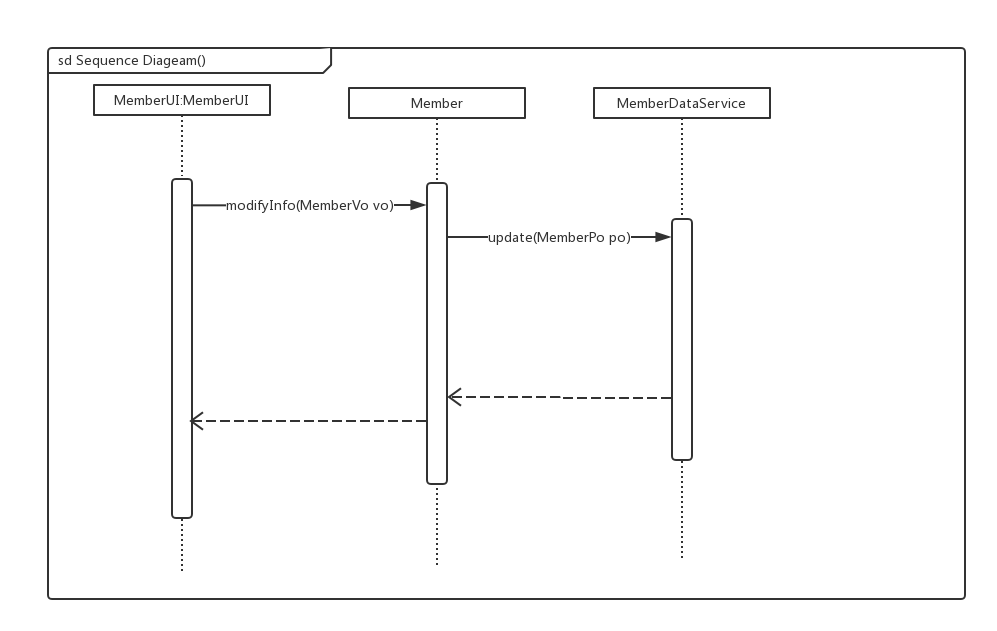
|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| **Member.getInfo** | 语法 | | public MemberVo getInfo(String id) |
| 前置条件 | | Id合法 |
| 后置条件 | | 返回用户vo |
| **Member.checkInfo** | 语法 | | public ResultMessage checkInfo  (String info, InfoType type) |
| 前置条件 | | 无 |
| 后置条件 | | 返回需检查信息的检查结果 |
| **Member. modifyInfo** | 语法 | | public ResultMessage modifyInfo  (MemberVo vo) |
| 前置条件 | | vo合法 |
| 后置条件 | | 返回修改成功/失败信息，持久化更新所涉及的领域对象的数据 |
| **Member.chargeCredit** | 语法 | | public ResultMessage chargeCredit  (String id, int amount) |
| 前置条件 | | id合法 |
| 后置条件 | | 对客户信用进行充值 |
| **Member.getCredit** | 语法 | | Public int getCredit(String id) |
| 前置条件 | | Id合法 |
| 后置条件 | | 得到该账号的信用值 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| **DatabaseFactory.getMemberData()** | | 得到Member数据层的服务的引用 | |
| **MemberDataService.getCredit**  **(String memberId)** | | 根据账号得到该账号的信用值 | |
| **MemberDataService.chargeCredit**  **(String memberId, int amount)** | | 对客户信用进行充值,更新一条po | |
| **MemberDataService.getInfo**  **(String memberId)** | | 根据账号在数据库查找po | |
| **MemberDataService.modifyInfo**  **(MemberPo po)** | | 修改账号信息，更新一条po | |

（4）业务逻辑层的动态模型

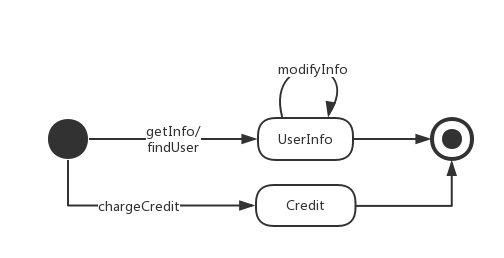
如图表明了当客户进行信用充值账号，用户信息业务逻辑处理的相关对象之间的协作。



如图表明了当客户想要修改个人基本信息的时候，用户信息业务逻辑处理的相关对象之间的协作。



如图所示的状态图描述了Member对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作



4.2.3 Hotelbl模块

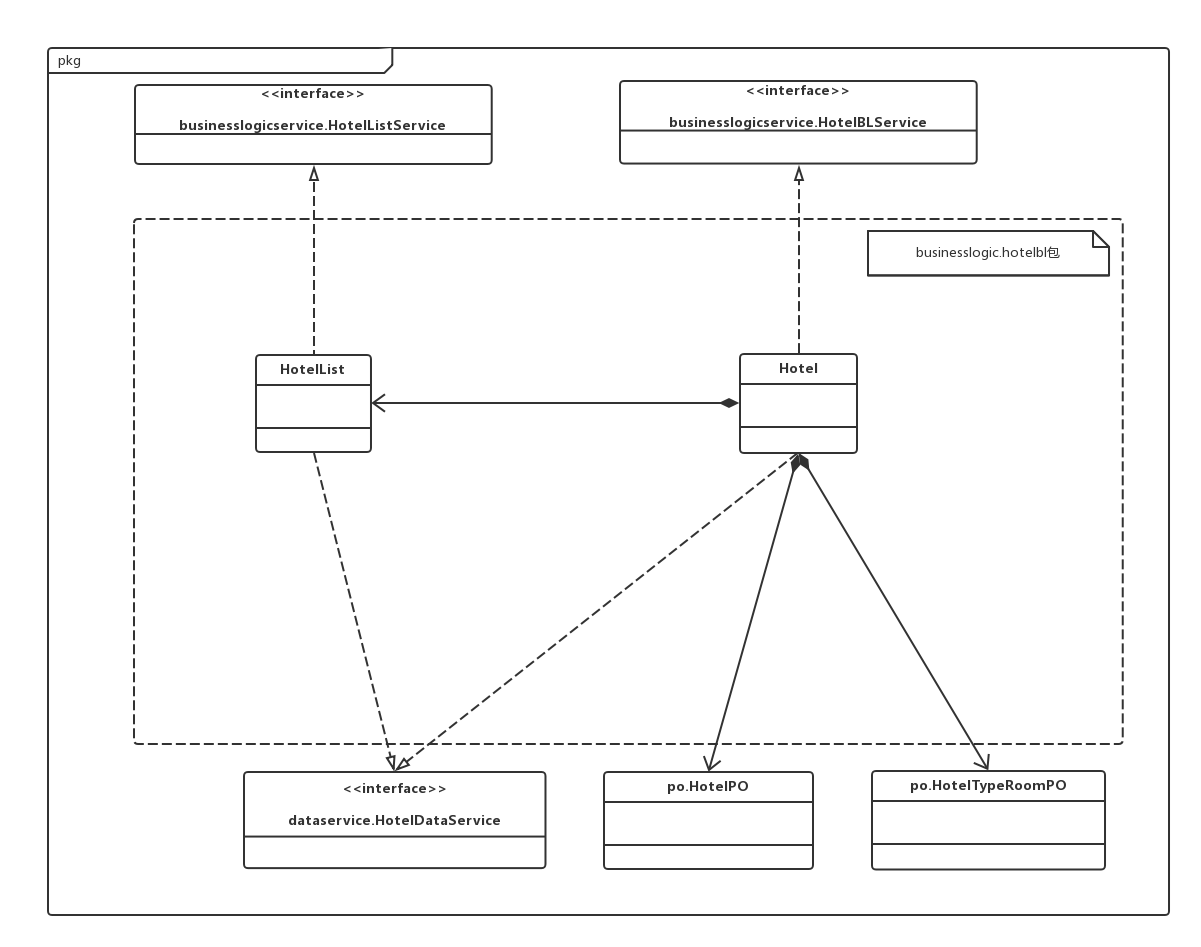
（1）模块概述

Hotelbl模块负责酒店的管理，查找酒店，显示酒店信息，酒店分类、排序，修改酒店。具体功能需求和非功能需求可参见需求规格说明文档和体系结构设计文档。

（2）整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如展示层和业务逻辑层之间，我们添加businesslogicservice.HotelBLService,businesslogicservice.HotelListService接口。业务逻辑层和数据层之间添加dataservice.HotelDataService接口。HotelPo,HotelTypeRoomPo是作为酒店的持久化对象被添加到设计模型中去的。

Hotelbl模块的设计如图



Hotelbl模块各个类的职责

|  |  |
| --- | --- |
| **模块** | **职责** |
| **Hotel** | 酒店的领域模型对象，拥有酒店的信息，可以帮助完成酒店管理所需要的服务 |
| **HotelList** | 酒店列表信息，提供酒店搜索、排序所需要的服务 |

（3）模块内部类的接口规范

**Hotel**的接口规范

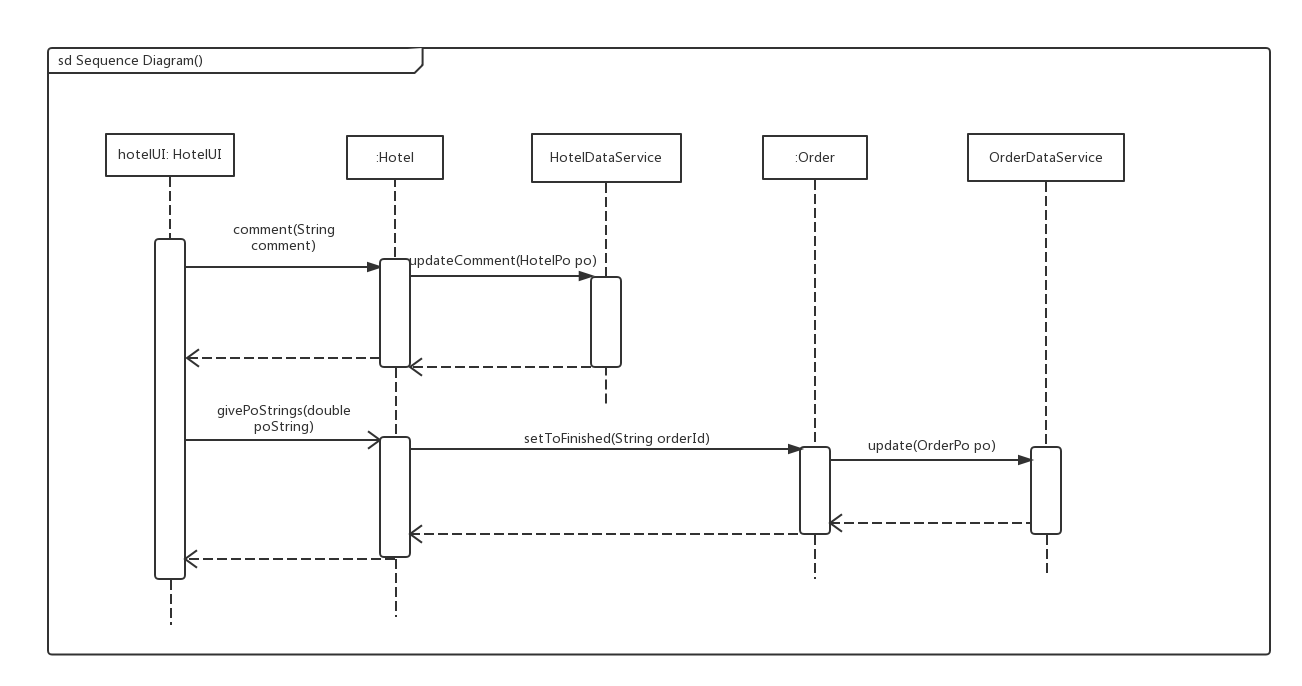
|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| **Hotel.getHotel** | 语法 | public HotelVo getHotel(String hotelId) | |
| 前置条件 | 无 | |
| 后置条件 | 按照酒店编号得到酒店vo | |
| **Hotel.updateBasicInfo** | 语法 | public ResultMessage updateBasicInfo(HotelVo vo) | |
| 前置条件 | 创建一个hotel对象 | |
| 后置条件 | 更新酒店基本信息 | |
| **Hotel.addHotel** | 语法 | public ResultMessage addHotel(String hotelId) | |
| 前置条件 | 不存在该ID的酒店 | |
| 后置条件 | 增加一个酒店 | |
| **Hotel.deleteHotel** | 语法 | public ResultMessage deleteHotel(String hotelId) | |
| 前置条件 | 存在该酒店 | |
| 后置条件 | 删除一个酒店 | |
| **Hotel.initializeRoom** | 语法 | public ResultMessage initializeRoom(String hotelId, RoomType type, int number, int price) | |
| 前置条件 | 输入合法 | |
| 后置条件 | 初始化客房信息 | |
| **Hotel.bookRoom** | 语法 | public ResultMessage bookroom(RoomType type, int number) | |
| 前置条件 | 该类型房间有空余数量 | |
| 后置条件 | 更新客房数量信息 | |
| **Hotel.checkoutRoom** | 语法 | public ResultMessage checkoutRoom(RoomType type, int number) | |
| 前置条件 | 创建一个hotel对象 | |
| 后置条件 | 更新客房数量信息 | |
| **Hotel.getReadyRoom** | 语法 | public int getReadyRoom(RoomType type) | |
| 前置条件 | 创建一个hotel对象 | |
| 后置条件 | 得到该类型房间的空余数量 | |
| **Hotel.comment** | 语法 | public ResultMessage comment(String comment) | |
| 前置条件 | 客户给予文字评价 | |
| 后置条件 | 更新酒店文字评论信息 | |
| **Hotel.givePoStrings** | 语法 | public ResultMessage givePoStrings(String orderId, double poString) | |
| 前置条件 | 客户给予评分 | |
| 后置条件 | 更新酒店评分、评分人数信息，更新订单状态 | |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| **DatabaseFactory.getHotelData** | | | 得到Hotel数据层的服务的引用 |
| **Order.setToFinished(String orderId)** | | | 将订单状态置为已评价状态 |
| **HotelDataService.addHotelPo(HotelPo po)** | | | 添加一个酒店po |
| **HotelDataService.deleteHotelPo(String hotelId)** | | | 删除一个酒店po |
| **HotelDataService.updateHotelList(HotelPo po)** | | | 更新一个酒店po |
| **HotelDataService.find(String id)** | | | 按ID进行查找返回相应的HotelPo结果 |
| **HotelDataService.updateComment(HotelPo po)** | | | 更新酒店文字评论 |
| **HotelDataService.initHotelTypeRoom(String hotelId, RoomType type, int number, int price)** | | | 初始化酒店房间信息 |
| **HotelDataService.getReadyRoom(String hotelId, RoomType type)** | | | 得到某类型房间的空余数量 |
| **HotelDataService.updateOrderedRoom(String hotelId, RoomType type, int number,**  **boolean ischeckI0n)** | | | 更新酒店房间预订数量 |

**HotelList**的接口规范

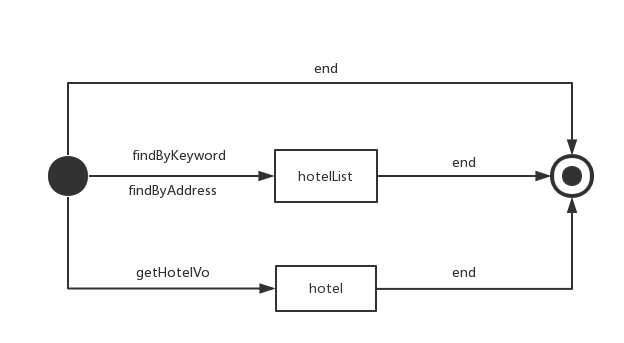
|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| **HotelList.sortByName** | 语法 | public void sortByName(List<HotelVo> list,  boolean isRise) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按酒店名称排序 |
| **HotelList.sortByStar** | 语法 | public void sortByStar(List<HotelVo> list,  boolean isRise) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按星级排序 |
| **HotelList.sortByPoint** | 语法 | public void sortByPoint(List<HotelVo> list,  boolean isRise) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按评分排序 |
| **HotelList.sortByPrice** | 语法 | public void sortByPrice(List<HotelVo> list,  RoomType type, boolean isRise) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按房间类型原始价格排序 |
| **HotelList.findByRoomType** | 语法 | public List<HotelVo> findByRoomType  (RoomType type, List<HotelVo> list) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按房间类型搜索酒店 |
| **HotelList.findByOriginalPrice** | 语法 | public List<HotelVo> findByOriginalPrice(RoomType type, int low, int high, List<HotelVo> list) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按房间类型原始价格区间搜索酒店 |
| **HotelList.findByAddress** | 语法 | public List<HotelVo> findByAddress(String province, String city, String business) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按省份、城市、商圈搜索酒店 |
| **HotelList.findByPoint** | 语法 | public List<HotelVo> findByPoint(double least,double max, List<HotelVo> list) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按评分区间搜索酒店 |
| **HotelList.findByStars** | 语法 | public List<HotelVo> findByStars(int least,int max, List<HotelVo> list) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按星级区间搜索酒店 |
| **HotelList.findByKeyword** | 语法 | public List<HotelVo> findByKeyword(String info) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 按关键字搜索酒店 |
| **HotelList.getProvinceList** | 语法 | public List<String> getProvinceList() |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 得到所有省份列表 |
| **HotelList.getCityList** | 语法 | public List<String> getCityList(String province) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 得到某省份下的所有城市列表 |
| **HotelList.getBusinessList** | 语法 | public List<String> getBusinessList(String province, String city) |
| 前置条件 | 创建一个HotelList对象 |
| 后置条件 | 返回某省某市的所有商圈列表 |
| **需要的服务（需接口）** | | |
| 服务名 | | 服务 |
| **DatabaseFactory.getHotelData()** | | 得到Hotel数据层的服务的引用 |
| **HotelDataService.find(String id)** | | 按ID进行查找返回相应的HotelPo结果 |
| **HotelDataService.keyFind(String info)** | | 按关键字进行查找返回相应的所有结果 |
| **HotelDataService.getProvinceList()** | | 得到数据层省份列表 |
| **HotelDataService.getCityList(String province)** | | 得到某省份的所有城市列表 |
| **HotelDataService.getBusinessList**  **(String province, String city)** | | 得到某省某市的所有商圈列表 |

（4）业务逻辑层的动态模型

如图表明了客户在进行酒店评分时，酒店业务逻辑处理的相关对象之间的协作。



如图所示的状态图描述了Hotel对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。



4.2.4 Accountbl模块

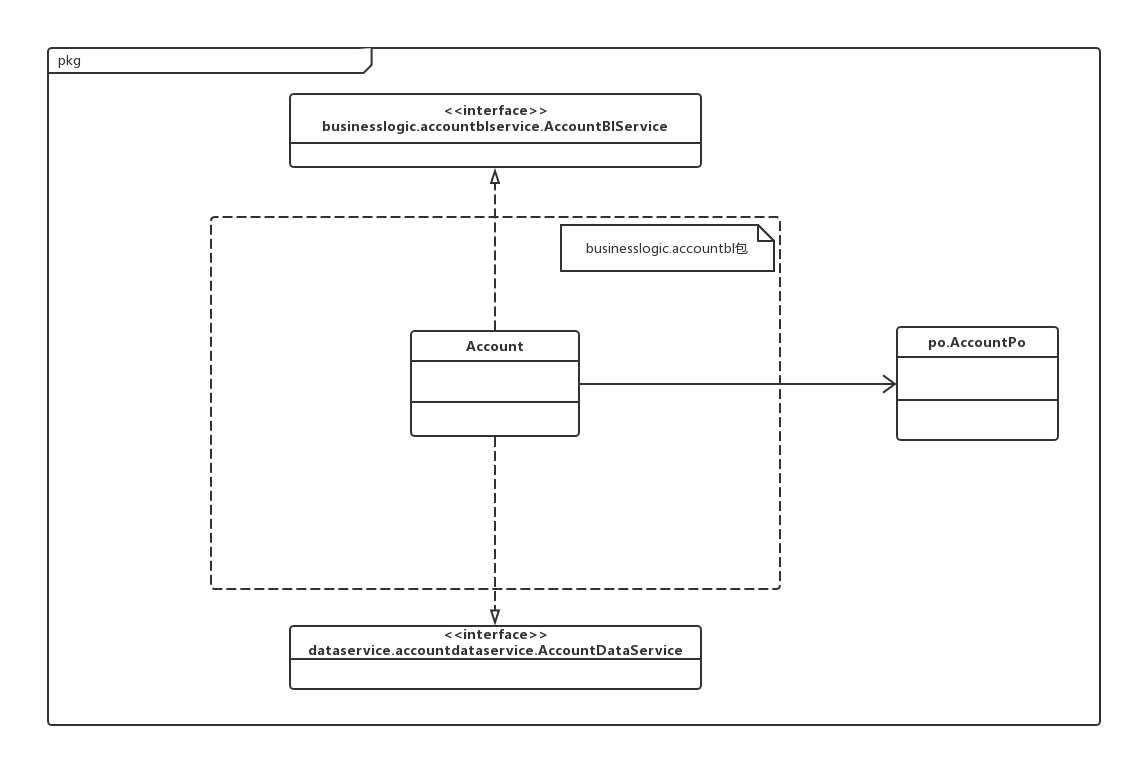
（1）模块概述

accountbl模块负责实现对四种类型用户（客户、酒店工作人员、酒店营销人员、酒店管理人员）账号的管理，对账号进行注册、查找、增加、删除和登入、登出验证，对账号密码进行修改。具体功能需求和非功能需求可参见需求规格说明文档和体系结构设计文档。

（2）整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如展示层和业务逻辑层之间，我们添加bussinesslogicservice.accountBLService接口。业务逻辑层和数据层之间添加dataservice.accountDataService接口。AccountPo是作为账号的持久化对象被添加到设计模型中去的。

accountbl模块的设计如图



accountbl模块各个类的职责

|  |  |
| --- | --- |
| **模块** | **职责** |
| **Account** | 账号的领域模型对象，拥有账号的账号名和密码，可以帮助完成账号管理和登入登出界面所需要的服务 |

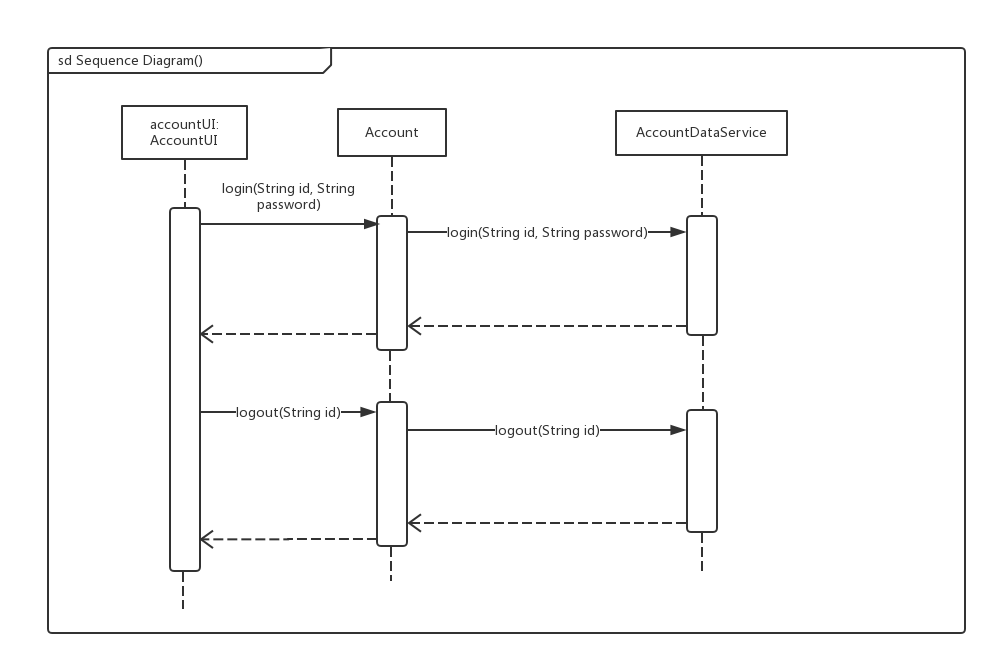
（3）模块内部类的接口规范

**Account** 的接口规范

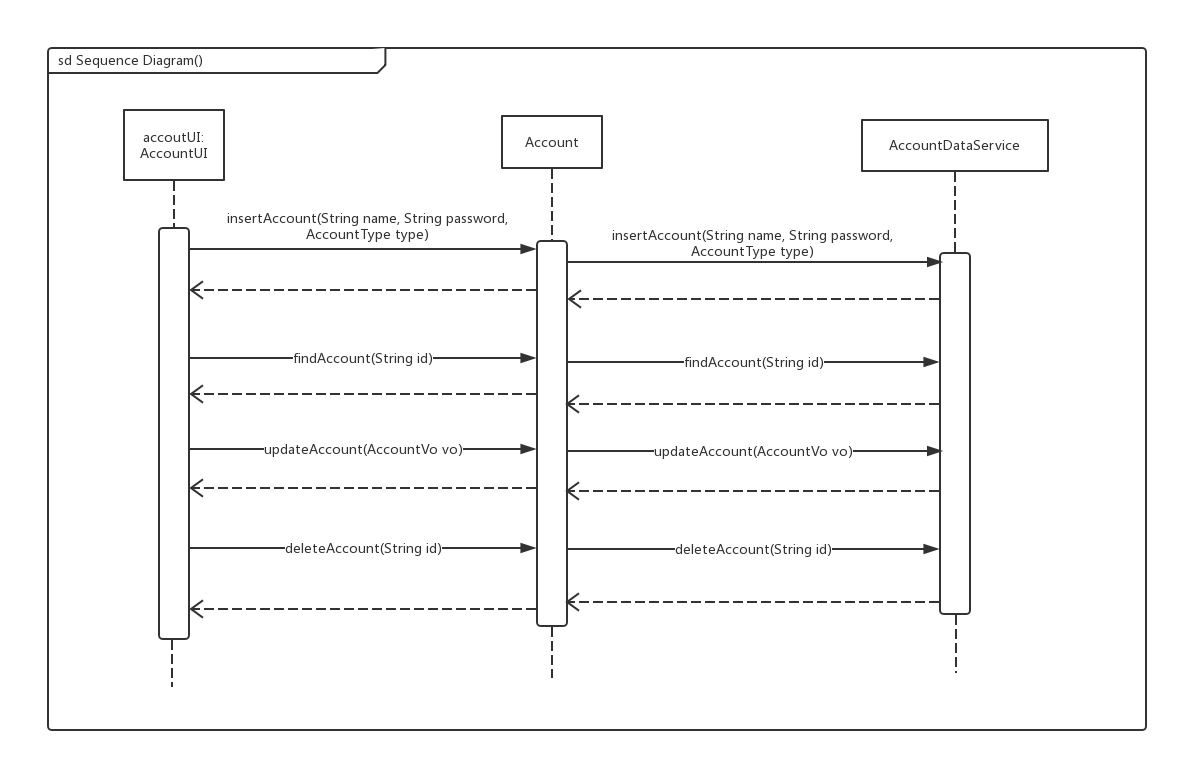
|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| **Account.login** | 语法 | | public AccountType login  (String id, String password) |
| 前置条件 | | 账号密码均正确且账号未登录 |
| 后置条件 | | 将账号状态置为已登录 |
| **Account.logout** | 语法 | | public ResultMessage logout(String id) |
| 前置条件 | | 账号状态为已登录 |
| 后置条件 | | 将账号状态置为未登录 |
| **Account.register** | 语法 | | public String register(String name,  String password, boolean isMember) |
| 前置条件 | | 用户名、密码合法 |
| 后置条件 | | 生成用户账号 |
| **Account.modifyPassword** | 语法 | | public ResultMessage modifyPassword  (String id, String newPassword) |
| 前置条件 | | 该账号存在且新密码合法 |
| 后置条件 | | 修改该账号的密码 |
| **Account.checkInput** | 语法 | | public ResultMessage checkInput(String input) |
| 前置条件 | | 无 |
| 后置条件 | | 检查输入是否合法 |
| **Account.findAccount** | 语法 | | public AccountVo findAccount(String id) |
| 前置条件 | | Id合法 |
| 后置条件 | | 根据id查找账号 |
| **Account.insertAccount** | 语法 | | public String insertAccount(String name,  String password, AccountType type) |
| 前置条件 | | 网站管理人员增加任意类型的用户 |
| 后置条件 | | 生成该用户的账号 |
| **Account.updateAccount** | 语法 | | public ResultMessage updateAccount  (AccountVo vo) |
| 前置条件 | | 用户更新账号信息 |
| 后置条件 | | 更新账号信息 |
| **Account.deleteAccount** | 语法 | | public ResultMessage deleteAccount(String id) |
| 前置条件 | | 网站管理人员删除账号 |
| 后置条件 | | 删除账号 |
| **Account.getAccountTypeById** | 语法 | | public AccountType getAccountTypeById  (String id) |
| 前置条件 | | 无 |
| 后置条件 | | 根据账号得到用户类型 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| **DatabaseFactory.getAccountData()** | | 得到Account数据层的服务的引用 | |
| **AccountDateService.insert(String username, String password, AccountType type)** | | 在数据库中插入一个单一持久化对象 | |
| **AccountDateService.delete(String id)** | | 在数据库中删除单一持久化对象 | |
| **AccountDateService.update(AccountPo po)** | | 在数据库中更新单一持久化对象 | |
| **AccountDateService.find(String id)** | | 根据账号查找单一持久化对象 | |
| **AccountDateService.modifyPassword(String id, String newPassword)** | | 更新账号的密码 | |
| **AccountDateService.login**  **(String id, String password)** | | 将账号置为登录状态 | |
| **AccountDateService.logout(String id)** | | 账将号置为未登录状态 | |

（4）业务逻辑层的动态模型

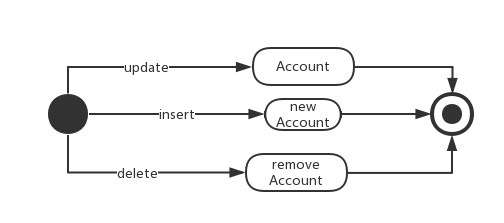
如图表明了当用户进行登入登出账号，账号业务逻辑处理的相关对象之间的协作。



如图表明了当网站管理人员想要对账号进行管理（包括查找、修改、增加和删除账号）时，账号业务逻辑处理的相关对象之间的协作。



如图所示的状态图描述了Account对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。



4.2.5 HotelPromotionbl模块

（1）模块概述

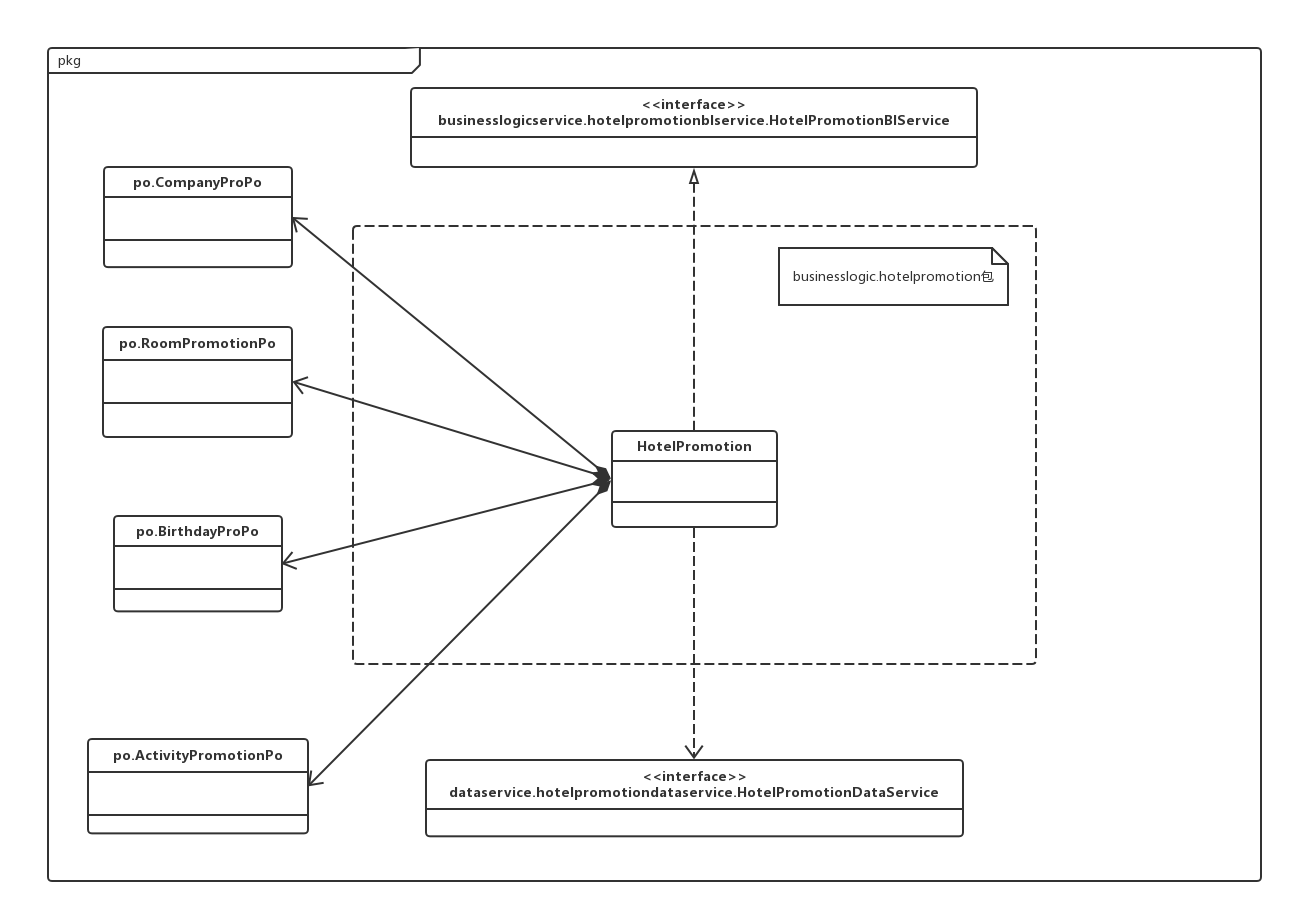
HotelPromotionbl模块承担编辑并实行促销策略的需求，由酒店工作人员维护，需要允许酒店工作人员编辑促销策略信息（活动促销策略、客户生日促销策略、合作企业促销策略、房间预订数量促销策略）。具体功能需求和非功能需求参见软件需求规格说明文档。

HotelPromotionbl模块的职责及接口参见软件体系结构描述文档。

（2）整体结构

根据体系结构的设计，我们将系统分为用户界面层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如用户界面层和业务逻辑层之间，我们添加businesslogicservice.hotelpromotionblservice.HotelPromotionBLService接口。业务逻辑层和数据层之间添加dataService.hotelpromotiondataservice.HotelPromotionDataService接口。ActivityPromotionPo，BirthdayProPo，CompanyProPo，RoomPromotionPo是作为促销策略的持久化对象被添加到设计模型中去的。

HotelPromotionbl模块的设计如图所示。



HotelPromotionbl模块各个类的职责如下表所示。

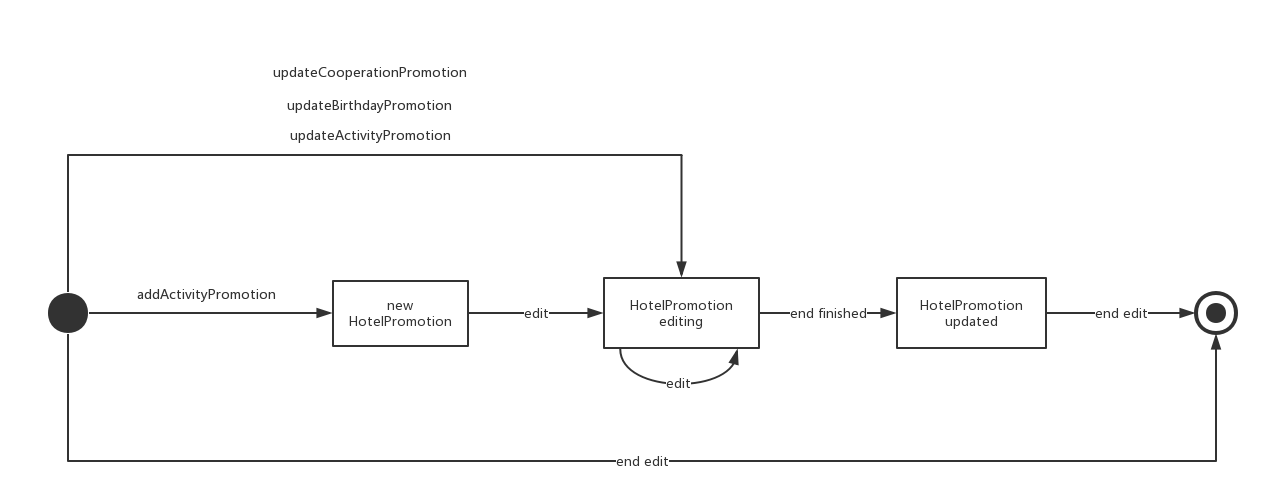
|  |  |
| --- | --- |
| **模块** | **职责** |
| **HotelPromotion** | 提供酒店促销策略所需服务，增、删、改促销策略 |

（3）模块内部类的接口规范

**HotelPromotion**的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| **HotelPromotion.addActivityStrategy** | 语法 | public ResultMessage addActivityStrategy  (String hotelId, ActivityPromotionVo vo) | |
| 前置条件 | 酒店工作人员增加活动促销策略 | |
| 后置条件 | 增加一条活动促销策略 | |
| **HotelPromotion.upActivityStrategy** | 语法 | public ResultMessage upActivityStrategy  (String hotelId, ActivityPromotionVo vo) | |
| 前置条件 | 酒店工作人员更新活动促销策略 | |
| 后置条件 | 更新一条活动促销策略 | |
| **HotelPromotion.delActivityStrategy** | 语法 | public ResultMessage delActivityStrategy  (String hotelId, ActivityPromotionVo vo) | |
| 前置条件 | 酒店工作人员删除一条活动策略 | |
| 后置条件 | 删除一条活动促销策略 | |
| **HotelPromotion.getCurrentActStrategy** | 语法 | public List<ActivityPromotionVo> getCurrentActStrategy(String hotelId) | |
| 前置条件 | 无 | |
| 后置条件 | 得到当前的有效活动促销策略列表 | |
| **HotelPromotion.upCoopStrategy** | 语法 | public ResultMessage upCoopStrategy  (String hotelId, CompanyProVo vo) | |
| 前置条件 | 酒店工作人员更新合作企业促销策略 | |
| 后置条件 | 更新企业促销策略 | |
| **HotelPromotion.getCoopStrategy** | 语法 | public double getCoopStrategy  (String hotelId, String memberId) | |
| 前置条件 | 无 | |
| 后置条件 | 得到企业在酒店的促销折扣 | |
| **HotelPromotion.upOrderRoomStrategy** | 语法 | public ResultMessage upOrderRoomStrategy  (String hotelId, RoomPromotionVo vo) | |
| 前置条件 | 酒店工作人员更新房间数量预订促销策略 | |
| 后置条件 | 更新房间预订促销策略 | |
| **HotelPromotion.getOrderRoomStrategy** | 语法 | public RoomPromotionVo getOrderRoomStrategy  (String hotelId, int numOfRoom) | |
| 前置条件 | 无 | |
| 后置条件 | 得到酒店房间数量预定促销策略 | |
| **HotelPromotion.upBirthStrategy** | 语法 | public ResultMessage upBirthStrategy  (String hotelId, BirthdayProVo vo) | |
| 前置条件 | 酒店工作人员更新客户生日促销策略 | |
| 后置条件 | 更新客户生日促销策略 | |
| **HotelPromotion.getBirthStrategy** | 语法 | public double getBirthStrategy  (String hotelId, Date birthday) | |
| 前置条件 | 无 | |
| 后置条件 | 得到客户生日当日酒店促销折扣 | |
| **需要的服务（需接口）** | | | |
| 服务名 | | | 服务 |
| **DatabaseFactory.getHotelPromotionData()** | | | 得到HotelPromotion数据层的服务的引用 |
| **HotelPromotionDataService.upActPromotion(String hotelId, ActivityPromotionPo po)** | | | 更新活动策略 |
| **HotelPromotionDataService.delActPromotion(String hotelId, ActivityPromotionPo po)** | | | 删除活动策略 |
| **HotelPromotionDataService.getActProList(String hotelId)** | | | 得到活动促销策略列表 |
| **HotelPromotionDataService.upBirthPromotion(String hotelId, BirthdayProPo po)** | | | 更新客户生日优惠策略 |
| **HotelPromotionDataService.getBirthPromotion(String hotelId)** | | | 得到客户生日优惠策略 |
| **HotelPromotionDataService.updateCooperPro(String hotelId, CompanyProPo po)** | | | 更新合作企业优惠策略 |
| **HotelPromotionDataService.getCooperPro(String hotelId)** | | | 得到合作企业优惠策略 |
| **HotelPromotionDataService.upRoomPromotion(String hotelId, RoomPromotionPo po)** | | | 更新预订房间数量优惠策略 |
| **HotelPromotionDataService.getRoomPromotion(String hotelId)** | | | 得到预订房间数量优惠策略 |

（4）业务逻辑层的动态模型



4.2.6 MarketPromotionbl模块

（1）模块概述

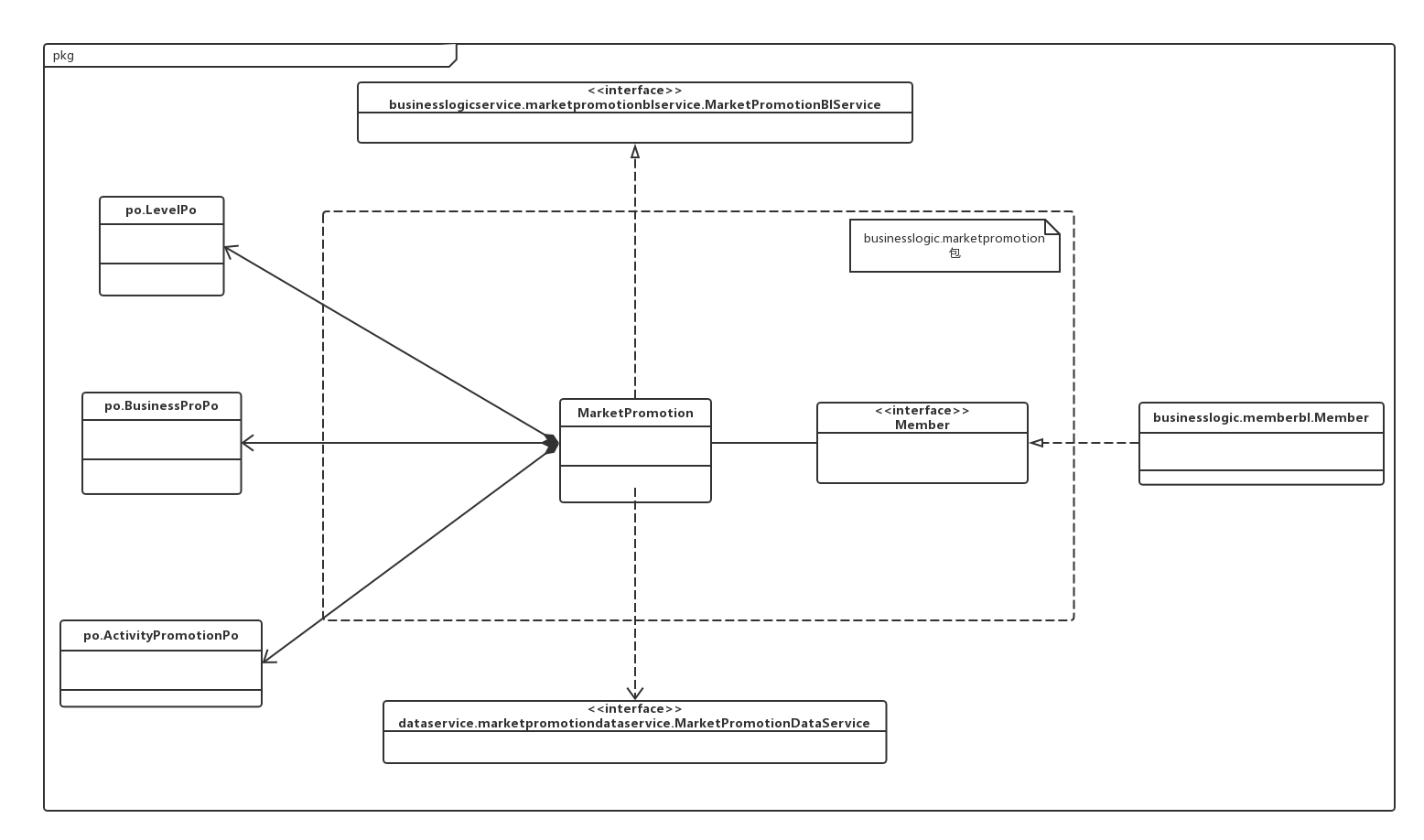
MarketPromotionbl模块承担编辑并实行促销策略的需求，由网站营销人员维护，需要允许网站营销人员编辑促销策略信息（活动促销策略、等级制定及促销策略、特定商圈专属促销策略）。具体功能需求和非功能需求参见软件需求规格说明文档。

MarketPromotionbl模块的职责及接口参见软件体系结构描述文档。

（2）整体结构

根据体系结构的设计，我们将系统分为用户界面层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如用户界面层和业务逻辑层之间，我们添加businesslogicservice.marketpromotionblservice.MarketPromotionBLService接口。业务逻辑层和数据层之间添加dataService.marketpromotiondataservice.MarketPromotionDataService接口。ActivityPromotionPo，BusinessProPo，LevelPo是作为促销策略的持久化对象被添加到设计模型中去的。

MarketPromotionbl模块的设计如图所示。



MarketPromotionbl模块各个类的职责如下表所示。

|  |  |
| --- | --- |
| **模块** | **职责** |
| **MarketPromotion** | 提供网站促销策略所需服务，增、删、改促销策略 |

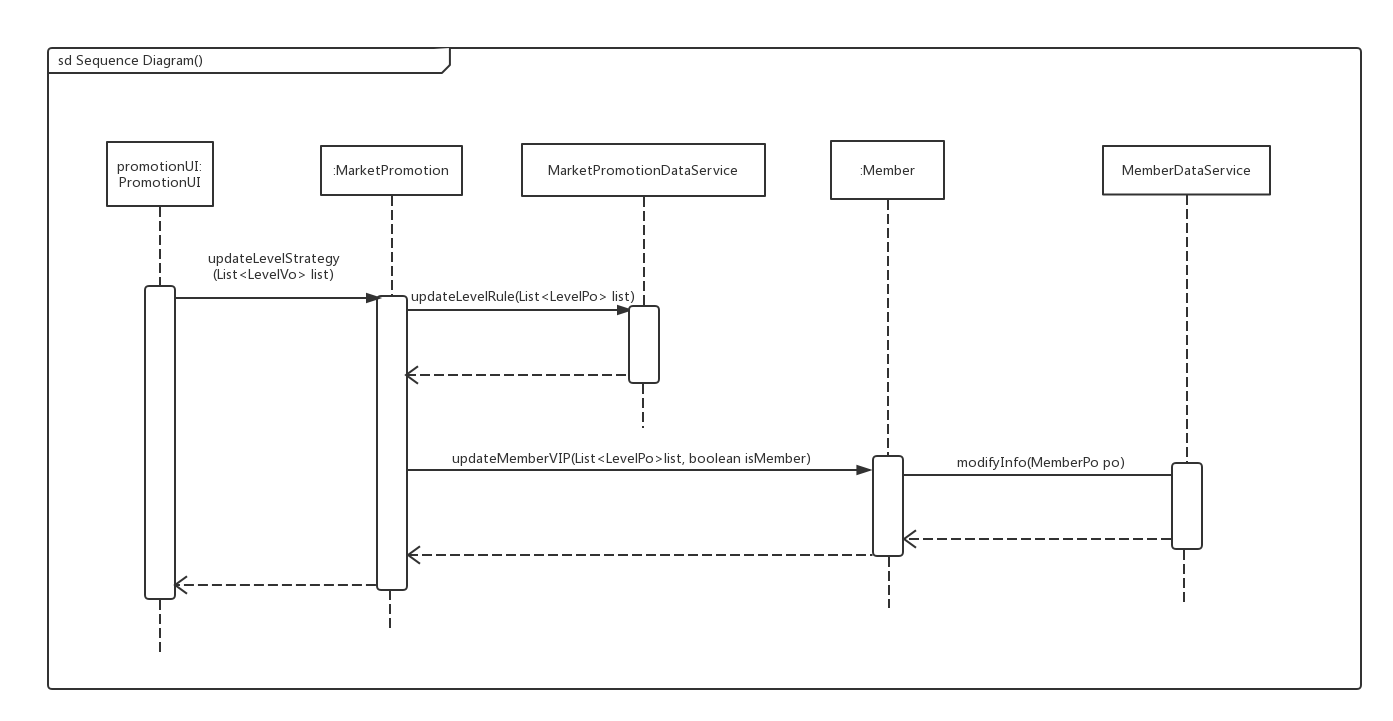
（3）模块内部类的接口规范

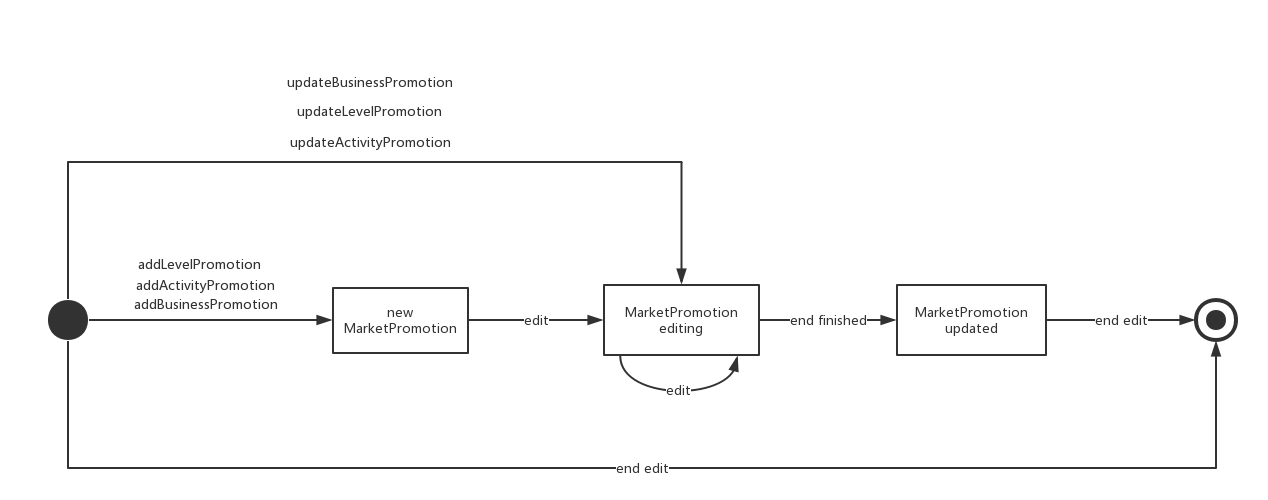
**MarketPromotion**的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| **MarketPromotion.addActivityStrategy** | 语法 | | public ResultMessage addActivityStrategy  (ActivityPromotionVo vo) |
| 前置条件 | | 网站营销人员增加活动促销策略 |
| 后置条件 | | 增加一条活动促销策略 |
| **MarketPromotion.upActivityStrategy** | 语法 | | public ResultMessage upActivityStrategy  (ActivityPromotionVo vo) |
| 前置条件 | | 网站营销人员更新活动促销策略 |
| 后置条件 | | 更新一条活动促销策略 |
| **MarketPromotion.delActivityStrategy** | 语法 | | public ResultMessage delActivityStrategy  (ActivityPromotionVo vo) |
| 前置条件 | | 网站营销人员删除一条活动策略 |
| 后置条件 | | 删除一条活动促销策略 |
| **MarketPromotion.getCurrentActStrategy** | 语法 | | public List<ActivityPromotionVo> getCurrentActStrategy() |
| 前置条件 | | 无 |
| 后置条件 | | 得到当前的有效活动促销策略列表 |
| **MarketPromotion.addBusinessStrategy** | 语法 | | public ResultMessage addBusinessStrategy  (BusinessProVo vo) |
| 前置条件 | | 网站营销人员增加一条商圈策略 |
| 后置条件 | | 增加一条商圈策略 |
| **MarketPromotion.upBusinessStrategy** | 语法 | | public ResultMessage upBusinessStrategy  (BusinessProVo vo) |
| 前置条件 | | 网站营销人员更新一条商圈策略 |
| 后置条件 | | 更新一条商圈策略 |
| **MarketPromotion.delBusinessStrategy** | 语法 | | public ResultMessage delBusinessStrategy  (BusinessProVo vo) |
| 前置条件 | | 网站营销人员删除一个商圈策略 |
| 后置条件 | | 删除一个商圈促销策略 |
| **MarketPromotion.getBusinessStrategy** | 语法 | | public BusinessProVo getBusinessStrategy  (String businessName) |
| 前置条件 | | 无 |
| 后置条件 | | 得到当前商圈专属促销策略 |
| **MarketPromotion.upLevelStrategy** | 语法 | | public ResultMessage upLevelStrategy  (List<LevelVo> list) |
| 前置条件 | | 网站营销人员更新等级促销策略 |
| 后置条件 | | 更新等级促销策略列表 |
| **MarketPromotion.getLevelStrategy** | 语法 | | public double getLevelStrategy(int level) |
| 前置条件 | | 无 |
| 后置条件 | | 得到客户等级对应的折扣 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| **DatabaseFactory.getMarketPromotionData()** | | 得到MarketPromotion数据层的服务的引用 | |
| **MarketPromotionDataService.upActPromotion**  **(ActivityPromotionPo po)** | | 更新活动策略 | |
| **MarketPromotionDataService.delActPromotion**  **(ActivityPromotionPo po)** | | 删除活动策略 | |
| **MarketPromotionDataService.getActProList()** | | 得到活动促销策略列表 | |
| **MarketPromotionDataService.updateLevelRule** | | 更新等级促销策略 | |
| **MarketPromotionDataServicegetLevelList()** | | 得到客户等级促销策略 | |
| **MarketPromotionDataService.updateBusiness**  **(BusinessProPo po)** | | 更新特定商圈促销策略 | |
| **MarketPromotionDataService.deleteBusiness**  **(BusinessProPo po)** | | 删除一条商圈策略 | |
| **MarketPromotionDataService.getBusinessList()** | | 得到特定商圈促销列表 | |
| **MemberDataService.modifyInfo(Member vo)** | | 当等级策略变化，更新所有客户等级 | |

（4）业务逻辑层的动态模型

如图表明了互联网酒店管理系统中，当网站营销人员更新等级策略时，促销策略业务逻辑处理的相关对象之间的协作。





4.2.7 Creditbl模块

（1）模块概述

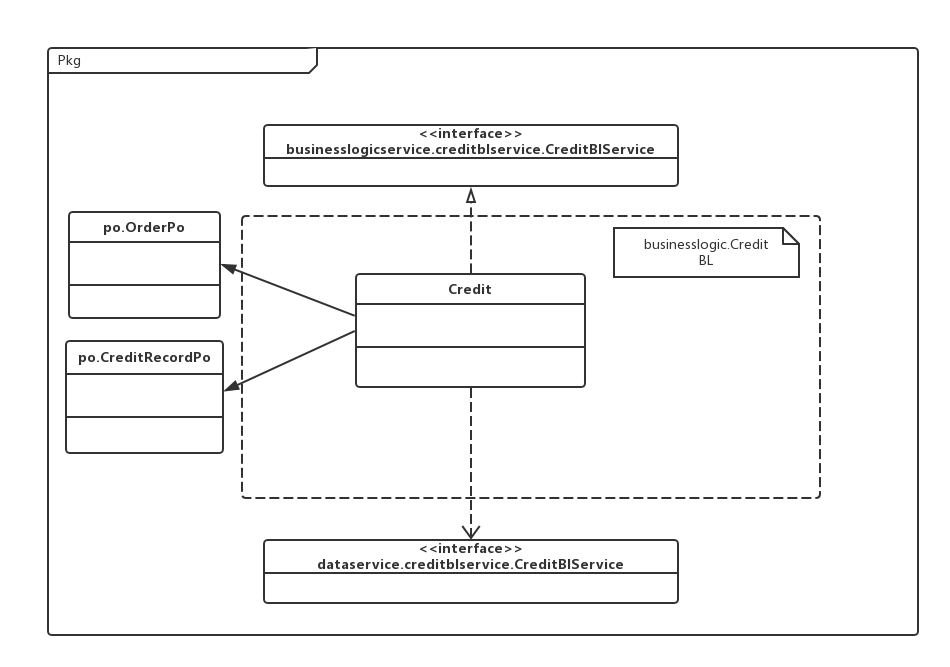
Creditbl模块记录信用值变化的需求。具体功能需求和非功能需求参见软件需求规格说明文档。

Creditbl模块的职责及接口参见软件体系结构描述文档。

（2）整体结构

根据体系结构的设计，我们将系统分为用户界面层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如用户界面层和业务逻辑层之间，我们添加businesslogicservice.Creditblservice.CreditBLService接口。业务逻辑层和数据层之间添加dataService.Creditdataservice.CreditDataService接口。CreditRecordPo是作为信用记录的的持久化对象被添加到设计模型中去的。

Creditbl模块的设计如图所示。



creditbl模块各个类的职责如下表所示。

|  |  |
| --- | --- |
| **模块** | **职责** |
| **credit** | 提供一个用户的信用值记录 |

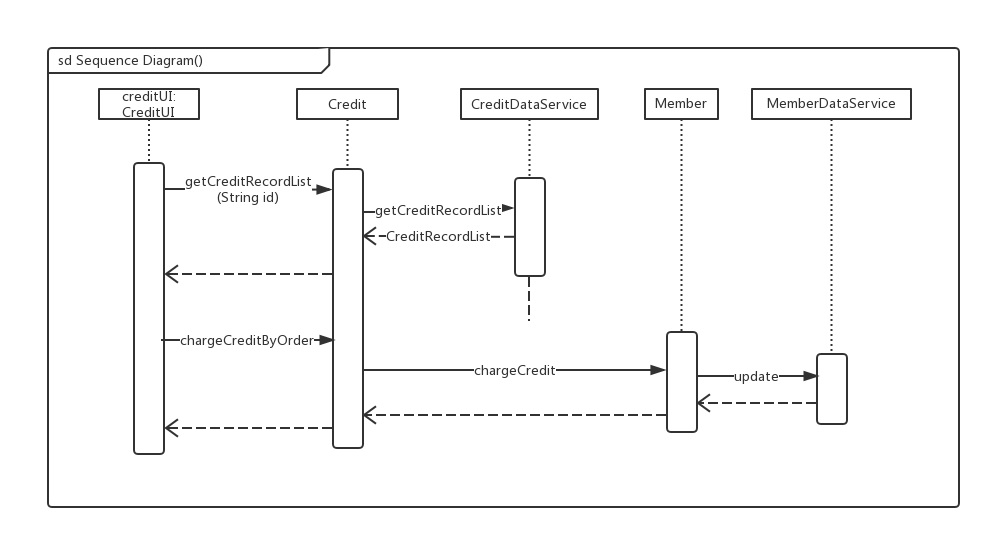
（3）模块内部类的接口规范

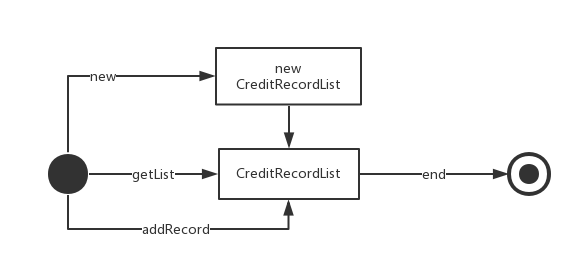
Credit的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| **Credit.addCreditByOrder** | 语法 | | public ResultMessage addCreditByOrder  (String id,int amount,OrderVo vo) |
| 前置条件 | | Id,订单,信用变化合法 |
| 后置条件 | | 更新信用值和信用值记录 |
| **Credit.getCreditRecordList** | 语法 | | public List<CreditRecordVo> get CreditRecordList(String id) |
| 前置条件 | | id合法 |
| 后置条件 | | 得到用户的信用值记录表 |
| **需要的服务（需接口）** | | | |
| 服务名 | | 服务 | |
| **DatabaseFactory.getCreditData()** | | 得到Credit数据层的服务的引用 | |
| **CreditDataService.updateCreditRecordData**  **(List<CreditRecordPo> list,String id)** | | 更新用户的信用记录 | |

（4）业务逻辑层的动态模型

如图表明了互联网酒店管理系统中，当通过订单改变用户的信用值时，信用逻辑处理的相关对象之间的协作。





4.3数据层的分解

数据层的开发包图参见 互联网酒店管理系统HPMS软件体系结构描述文档\_V2.4 图13

4.3.1.DataFactoryService

(1)模块描述

DataFactoryService模块是一个工厂类，用来创建其他的DataService实例，并将其统一管理起来。

(2)模块内部的接口规范如下表

**DataFactory的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| **DataFactory.getOrderData** | **语法** | public DataService getOrderData() throws RemoteException |
| **前置条件** | 无 |
| **后置条件** | 返回OrderData对象 |
| **DataFactory.getMemberData** | **语法** | public DataService getMemberData() throws RemoteException |
| **前置条件** | 无 |
| **后置条件** | 返回MemberData对象 |
| **DataFactory.getHotelData** | **语法** | public DataService getHotelData() throws RemoteException |
| **前置条件** | 无 |
| **后置条件** | 返回HotelData对象 |
| **DataFactory.getAccountData** | **语法** | public DataService getAccountData() throws RemoteException |
| **前置条件** | 无 |
| **后置条件** | 返回AccountData对象 |
| **DataFactory.getHotelPromotionData** | **语法** | public DataService getHotelPromotionData() throws RemoteException |
| **前置条件** | 无 |
| **后置条件** | 返回HotelPromotionData对象 |
| **DataFactory.getMarketPromotionData** | **语法** | public DataService getMarketPromotionData() throws RemoteException |
| **前置条件** | 无 |
| **后置条件** | 返回MarketPromotionData对象 |
| **DataFactory.getCreditData** | **语法** | public DataService getCreditData() throws RemoteException |
| **前置条件** | 无 |
| **后置条件** | 返回CreditData对象 |

4.3.2.OrderDataService

(1)模块描述

OrderDataService模块提供订单数据的集体载入、保存、增加、查找、修改服务

(2)模块内类接口规范

**OrderDataService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** | 相同ID的po在数据库中不存在 |
| **后置条件** | 在数据库中增加一条po记录 |
| **OrderDataService.find** | **语法** | public OrderPo find(String orderId) |
| **前置条件** | 订单号合法 |
| **后置条件** | 按ID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.delete** | **语法** | public ResultMessage deleteOrder  (String orderId) |
| **前置条件** | 数据库中存在该po |
| **后置条件** | 删除该po |
| **OrderDataService.getAllByHotel** | **语法** | public List<OrderPo> getAllByHotel  (String hotelId) |
| **前置条件** | 酒店账号存在 |
| **后置条件** | 得到该酒店的所有订单 |
| **OrderDataService.getAllByMember** | **语法** | public List<OrderPo> getAllByMember  (String memberId) |
| **前置条件** | 客户账号存在 |
| **后置条件** | 得到该客户的所有订单 |
| **OrderDataService.getNotExecuted** | **语法** | public List<OrderPo> getNotExecuted() |
| **前置条件** | 无 |
| **后置条件** | 得到所有未执行订单 |
| **OrderDataService.getAbnormal** | **语法** | public List<OrderPo> getAbnormal() |
| **前置条件** | 无 |
| **后置条件** | 得到所有异常订单 |

4.3.3.MemberDataService

(1)模块描述

MemberDataService模块提供用户信息数据集体载入、保存、增加、查找、修改服务

(2)模块内类接口规范

**MemberDataService接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| **MemberDataService.getCredit** | **语法** | public int getCredit(String memberId) |
| **前置条件** | 该客户账号存在 |
| **后置条件** | 得到该客户的信用值 |
| **MemberDataService.chargeCredit** | **语法** | public ResultMessage chargeCredit  (String memberId, int amount) |
| **前置条件** | 该客户账号存在，修改的信用值合法 |
| **后置条件** | 修改客户信用值 |
| **MemberDataService.getInfo** | **语法** | public MemberVo getInfo  (String memberId) |
| **前置条件** | 该客户账号存在 |
| **后置条件** | 得到memberVo |
| **MemberDataService.modifyInfo** | **语法** | public ResultMessage modifyInfo  (MemberPo po) |
| **前置条件** | 数据库中存在该po |
| **后置条件** | 更新一条po |

4.3.4.HotelDataService

(1)模块描述

HotelDataService提供酒店数据的集体载入、保存、增加、查找、修改、删除服务

(2)模块内类接口规范

**HotelDataService接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| **HotelDataService.insert** | **语法** | public ResultMessage insert(HotelPo po) |
| **前置条件** | 同样ID的po在数据库中不存在 |
| **后置条件** | 在数据库中增加一条po |
| **HotelDataService.updateHotelList** | **语法** | public ResultMessage updateHotelList(HotelPo po) |
| **前置条件** | 在数据库中存在同样ID的po |
| **后置条件** | 更新酒店列表信息 |
| **HotelDataService.delete** | **语法** | public ResultMessage delete(String hotelId) |
| **前置条件** | 在数据库中存在同样ID的po |
| **后置条件** | 删除一个po |
| **HotelDataService.initHotelTypeRoom** | **语法** | public ResultMessage initHotelTypeRoom(String hotelId, RoomType type, int number, int price) |
| **前置条件** | 酒店工作人员录入客房信息 |
| **后置条件** | 初始化房间信息 |
| **HotelDataService.findHotel** | **语法** | public HotelPo findHotel(String hotelId) |
| **前置条件** | 客户查看酒店信息 |
| **后置条件** | 按ID返回相应的HotelPo结果 |
| **HotelDataService.keyFind** | **语法** | public List<HotelPo> keyFind(String info) |
| **前置条件** | 客户输入关键字搜索酒店 |
| **后置条件** | 按关键字进行查找并返回所有符合条件的HotelPo结果 |
| **HotelDataService.getReadyRoom** | **语法** | public int getReadyRoom(String hotelId, RoomType type) |
| **前置条件** | 酒店工作人员查看酒店空余房间数 |
| **后置条件** | 返回酒店某类型房间空余数 |
| **HotelDataService.updateComment** | **语法** | public ResultMessage updateComment(HotelPo po) |
| **前置条件** | 得到客户的文字评价 |
| **后置条件** | 更新酒店文字评论 |
| **HotelDataService.updateOrderedRoom** | **语法** | public ResultMessage updateOrderedRoom(String hotelId, RoomType type, int number,  Boolean isCheckIn) |
| **前置条件** | 客户预定酒店或退房 |
| **后置条件** | 更新酒店房间信息 |
| **HotelDataService.getProvinceList** | **语法** | public List<String> getProvinceList() |
| **前置条件** | 客户通过地址筛选搜索酒店 |
| **后置条件** | 得到所有省份列表 |
| **HotelDataService.getCityList** | **语法** | Public List<String> getCityList  (String province) |
| **前置条件** | 客户已选择一个省份 |
| **后置条件** | 得到该省份所有的城市列表 |
| **HotelDataService.getBusinessList** | **语法** | Public List<String> getBusinessList  (String province, String city) |
| **前置条件** | 客户已选择一个城市 |
| **后置条件** | 得到该城市所有的商圈 |

4.3.5.AccountDataService

(1)模块描述

AccountDataService模块提供对账号数据的载入、查找、增加、删除、修改服务。

(2)模块内类接口规范

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| **AccountDataService.insert** | **语法** | public String insert(String username, String password, AccountType type) |
| **前置条件** | 数据库中不存在该用户名，密码合法 |
| **后置条件** | 增加一条po，生成账号 |
| **AccountDataService.update** | **语法** | public ResultMessage update  (AccountPo po) |
| **前置条件** | 数据库中存在该po |
| **后置条件** | 更新一条po |
| **AccountDataService.delete** | **语法** | public ResultMessage delete(String id) |
| **前置条件** | 数据库中存在该po |
| **后置条件** | 删除一条po |
| **AccountDataService.find** | **语法** | public AccountPo find(String id) |
| **前置条件** | 数据库中存在该id的po |
| **后置条件** | 得到一条po |
| **AccountDataService.**  **modifyPassword** | **语法** | public ResultMessage modifyPassword  (String id, String newPassword) |
| **前置条件** | 数据库中存在该id的po，新密码合法 |
| **后置条件** | 更新一条po |
| **AccountDataService.login** | **语法** | public AccountType login  (String id, String password) |
| **前置条件** | 账号和密码正确，账号未在登录状态 |
| **后置条件** | 账号置为登录状态 |
| **AccountDataService.logout** | **语法** | public ResultMessage logout(String id) |
| **前置条件** | 该账号为已登录状态 |
| **后置条件** | 账号置为未登入状态 |

4.3.6.HotelPromotionDataService

(1)模块描述

HotelPromotionDataService模块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。

(2)模块内类接口规范

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| **HotelPromotionDataService.**  **upActPromotion** | **语法** | public ResultMessage upActPromotion  (String hotelId, ActivityPromotionPo po) |
| **前置条件** | 输入折扣合法，开始日期在结束日期前 |
| **后置条件** | 更新一条po |
| **HotelPromotionDataService.**  **addActPromotion** | **语法** | public ResultMessage addActPromotion  (String hotelId, ActivityPromotionPo po) |
| **前置条件** | 输入折扣合法，开始日期在结束日期前 |
| **后置条件** | 增加一条po |
| **HotelPromotionDataService.**  **delActPromotion** | **语法** | public ResultMessage delActPromotion(String hotelId, ActivityPromotionPo po) |
| **前置条件** | 数据库中存在该活动促销策略 |
| **后置条件** | 删除一条po |
| **HotelPromotionDataService.**  **getActProList** | **语法** | public List<String> getActProList  (String hotelId) |
| **前置条件** | 无 |
| **后置条件** | 得到该酒店的活动促销策略列表 |
| **HotelPromotionDataService.**  **upBirthPromotion** | **语法** | public ResultMessage upBirthPromotion  (String hotelId, BirthdayProPo po) |
| **前置条件** | 输入折扣合法 |
| **后置条件** | 更新一条po |
| **HotelPromotionDataService.**  **getBirthPromotion** | **语法** | public BirthdayProPo getBirthPromotion  (String hotelId) |
| **前置条件** | 无 |
| **后置条件** | 得到该酒店客户生日促销策略 |
| **HotelPromotionDataService.**  **updateCooperPro** | **语法** | public ResultMessage updateCooperPro  (String hotelId, CompanyProPo po) |
| **前置条件** | 输入折扣合法 |
| **后置条件** | 更新一条po |
| **HotelPromotionDataService.**  **getCooperPro** | **语法** | public CompanyProPo getCooperPro  (String hotelId) |
| **前置条件** | 无 |
| **后置条件** | 得到酒店合作企业促销策略 |
| **HotelPromotionDataService.**  **upRoomPromotion** | **语法** | public ResultMessage upRoomPromotion  (String hotelId, RoomPromotionPo po) |
| **前置条件** | 输入房间数量合法，折扣合法 |
| **后置条件** | 更新一条po |
| **HotelPromotionDataService.**  **getRoomPromotion** | **语法** | public RoomPromotionPo getRoomPromotion(String hotelId) |
| **前置条件** | 无 |
| **后置条件** | 得到酒店房间预订促销策略 |

4.3.7.MarketPromotionDataService

(1)模块描述

MarketPromotionDataService模块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。

(2)模块内类接口规范

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| **MarketPromotionDataService.**  **upActPromotion** | **语法** | public ResultMessage upActPromotion  (ActivityPromotionPo po) |
| **前置条件** | 输入折扣合法，开始日期在结束日期前 |
| **后置条件** | 更新一条po |
| **MarketPromotionDataService.**  **addActPromotion** | **语法** | public ResultMessage addActPromotion  (ActivityPromotionPo po) |
| **前置条件** | 输入折扣合法，开始日期在结束日期前 |
| **后置条件** | 增加一条po |
| **MarketPromotionDataService.**  **delActPromotion** | **语法** | public ResultMessage delActPromotion  (ActivityPromotionPo po) |
| **前置条件** | 数据库中存在该活动促销策略 |
| **后置条件** | 删除一条po |
| **MarketPromotionDataService.**  **getActProList** | **语法** | public List<String> getActProList() |
| **前置条件** | 无 |
| **后置条件** | 得到网站促销活动策略 |
| **MarketPromotionDataService.**  **updateLevelRule** | **语法** | public ResultMessage updateLevelRule  (List<LevelPo> list) |
| **前置条件** | 每个等级所需信用值随着等级递增而增加，折扣合法 |
| **后置条件** | 更新等级促销列表 |
| **MarketPromotionDataService.**  **getLevelList** | **语法** | public List<LevelPo> getLevelList() |
| **前置条件** | 无 |
| **后置条件** | 得到等级促销策略列表 |
| **MarketPromotionDataService.**  **updateBusiness** | **语法** | public ResultMessage updateBusiness  (BusinessProPo po) |
| **前置条件** | 输入折扣合法 |
| **后置条件** | 增加或更新一条po |
| **MarketPromotionDataService.**  **deleteBusiness** | **语法** | public ResultMessage deleteBusiness  (BusinessProPo po) |
| **前置条件** | 数据库中存在该po |
| **后置条件** | 删除一条po |
| **MarketPromotionDataService.**  **getBusinessList** | **语法** | public List<BusinessProPo> getBusinessList() |
| **前置条件** | 无 |
| **后置条件** | 得到有特定促销策略的商圈列表 |

4.3.8.CreditDataService

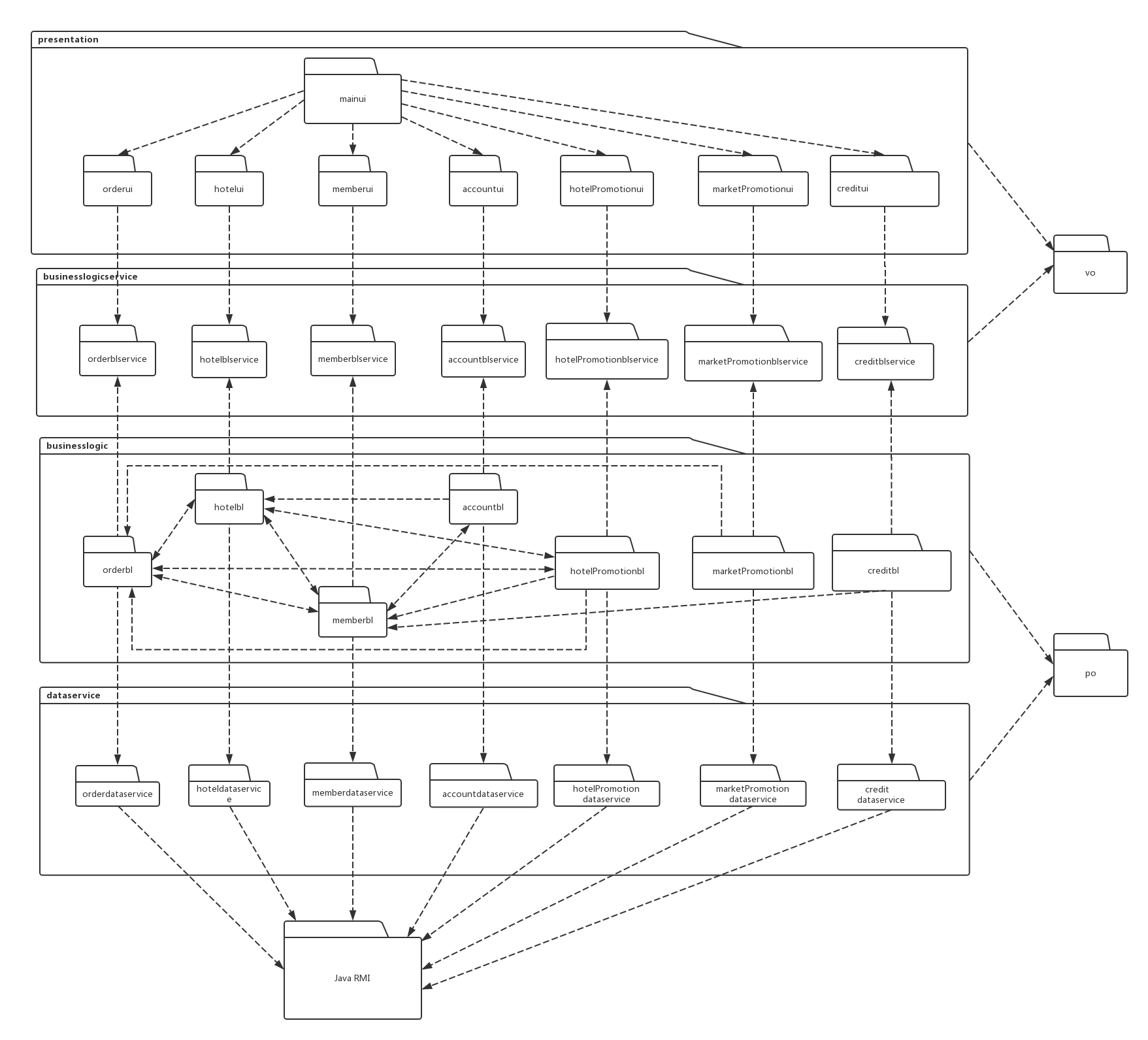
(1)模块描述

CreditDataService模块提供信用记录列表的查看服务。

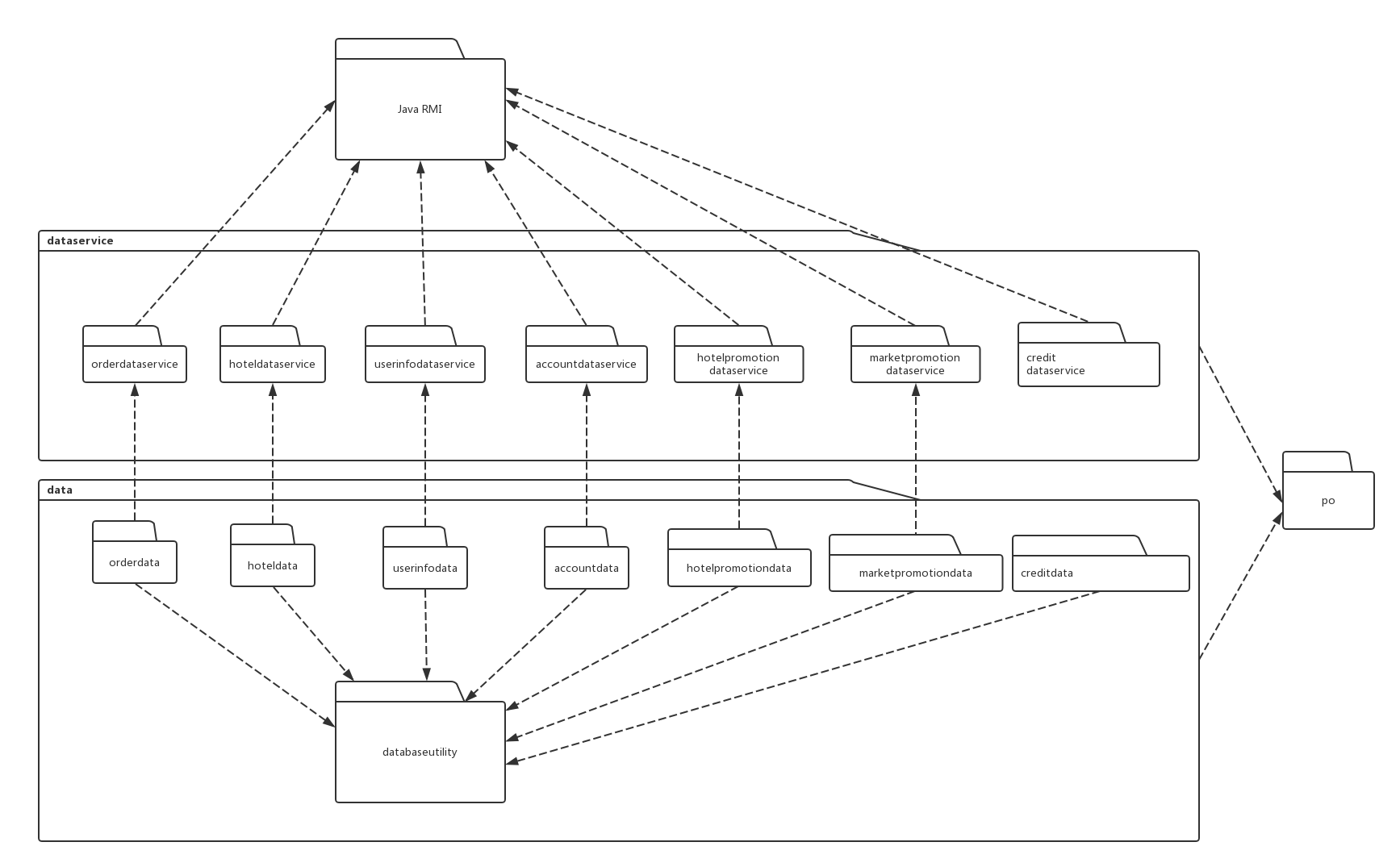
(2)模块内类接口规范

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| **CreditDataService.**  **addCreditByOrder** | **语法** | public ResultMessage addCreditByOrder  (String id,int amount,OrderVo vo) |
| **前置条件** | Id,订单号,数额合法 |
| **后置条件** | 增加或更新一条CreditRecordPo |
| **CreditDataService.**  **getCreditRecordList** | **语法** | public List<CreditRecordPo> getCreditRecordList(String id) |
| **前置条件** | id合法且存在 |
| **后置条件** | 得到该id的信用记录列表 |

5.依赖视角

****

客户端包图



服务器端包图