**互联网酒店预订系统**

**详细设计描述文档**

学 院：南京大学软件学院

成 员：谢铠联 谢凯航 武秀峰 夏沐天

完成日期：2016年1月1日

目录

[更新历史 5](#_Toc470469092)

[1. 引言 7](#_Toc470469093)

[1.1编制目的 7](#_Toc470469094)

[1.2词汇表 7](#_Toc470469095)

[1.3参考资料 7](#_Toc470469096)

[2. 产品概述 7](#_Toc470469097)

[3. 体系结构设计概述 8](#_Toc470469098)

[4. 结构视角 8](#_Toc470469099)

[4.1业务逻辑层的分解 8](#_Toc470469100)

[业务逻辑层的开发包图参见软件体系结构说明文档图4.1 8](#_Toc470469101)

[4.1.1 orderbl模块 8](#_Toc470469102)

[4.1.2 userbl模块 22](#_Toc470469105)

[4.1.3 hotelbl模块 44](#_Toc470469108)

[4.1.4 promotionbl模块 55](#_Toc470469111)

[4.2数据层的分解 61](#_Toc470469114)

[4.1.1 orderdata模块 61](#_Toc470469116)

[4.2.2 userdata模块 64](#_Toc470469118)

[4.2.3 hoteldata模块 69](#_Toc470469120)

[4.2.4 promotionbl模块 72](#_Toc470469122)

[4.2.5 searchdata模块 74](#_Toc470469124)

[4.3界面展示层的分解 77](#_Toc470469126)

[4.3.1 personui模块 77](#_Toc470469128)

[4.3.2 hotelworkerui模块 81](#_Toc470469130)

[4.3.3marketui模块 84](#_Toc470469132)

[4.3.4 managerui模块 86](#_Toc470469134)

[5. 依赖视角 89](#_Toc470469136)

[6. 耦合和内聚度量 90](#_Toc470469137)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 全体人员 | 2016-10-25 | 最初草稿（创建文档模板） | V1.0 |
| 全体人员 | 2016-10-27 | 最初定稿（成员加入具体内容） | V1.1 |
| 谢铠联、武秀峰 | 2016-10-28 | 发现Search类的方法返回值问题，并且进行拆分调整。同时调整了格式，把所有内容 | V2.0 |
| 谢铠联 | 2016-10-29 | 发现hotel、promotion包类名问题进行调整，并且加入了依赖视角 | V2.1 |
| 全体成员 | 2016-11-6 | 修正一些小错误，修正search模块所需方法的命名错误 | V3.0 |
| 夏沐天 | 2016-12-16 | 增加了data层的设计 | V3.1 |
| 武秀峰 | 2016-12-21 | 修改逻辑层的语法 | V3.2 |
| 谢铠联 | 2016-12-24 | 增加了ui层的设计 | V3.3 |
| 武秀峰 | 2016-12-25 | 修改逻辑层的语法 | V4.0 |
| 武秀峰 | 2016-1-1 | 逻辑层Hotel模块新增方法 | V4.1 |

# 引言

## 1.1编制目的

本报告详细完成对快递物流系统的详细设计，达到指导后续软件构造的目的，同时实现和测试人员及用户的沟通。

本报告面向开发人员、测试人员及最终用户而编写，是了解系统的导航。

## 1.2词汇表

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| UI | 表示某展示层 |  |
| bl | 表示某逻辑层 |  |
| Data | 表示某数据层 |  |

## 1.3参考资料

1. IEEE std 1471-2000
2. 丁二玉，刘钦.计算与软件工程（卷二）[M]机械工业出版2012：134—182

# 产品概述

随着工作和生活节奏的加快，人们的差旅出行需要迅速提升，但是传统上的线下酒店预订模式有着很多不方便之处：因为目的地往往是外地，无法全面了解当地的酒店情况，选择范围大大受限，难以选择到最适合的酒店；提前预订的双方都有风险，酒店担心恶意预订，顾客担心酒店不符合预期；酒店要及时掌握预订顾客的行程变化也会比较麻烦，需要频繁的长途联系；虽然过去的体验能够给未来带来参考，但是时间一长很多信息就会被遗忘掉，过去的参考作用就受到限制。（事实上，过去的出行体验会大大影响人们未来的选择）

为了解决上述问题，方便人们出行中的酒店预订，一家创业公司决定开发一个互联网酒店预订系统，包括下列功能：全面展现任一个地域内的酒店信息，帮助顾客进行更好地比较与选择；执行信用累计制度，防范恶意预订；进行房客评价，反映酒店与描述的相符程度；让酒店可以及时了解到所有预订订单的变化动态；帮助顾客建立体验历史记录，在顾客预订时发挥历史记录的作用。另外，为了进行网站推广，吸引酒店和顾客的入驻，该网站会在系统中补贴费用帮助酒店进行促销。

产品功能如下：

SF1：全面展现任一个地域内的酒店信息，包括房型、房间价格、商圈、星级、酒店特色、设施服务、评分

SF2：执行信用累计制度，通过客户的信用值来减少恶意预订

SF3：进行房客评价，反映酒店与描述的相符程度

SF4：酒店与网站数据实时更新，让酒店可以及时了解到所有预订订单的变化动态

SF5：酒店可以自行制定促销手段进行销售竞争

SF6：网站通过推广吸引客户和酒店，对酒店和客户进行补贴

SF7：发展会员，提高客户折扣和回头率

SF8：建立浏览历史和订单历史，帮助客户通过历史信息来更好的预订酒店

# 体系结构设计概述

# 结构视角

## 4.1业务逻辑层的分解

## 业务逻辑层的开发包图参见软件体系结构说明文档图4.1

### 4.1.1 orderbl模块

（1）模块概述

orderbl模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

orderbl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，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加的接口包括orderblservice , orderdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加OrderController，这样OrderController 将会将订单相关的业务逻辑职责和逻辑控制委托给orderbl 对象。orderPO是做为管理信息的持久化对象被添加到设计模型中的。

orderbl模块的设计如图4.1.1-1所示.

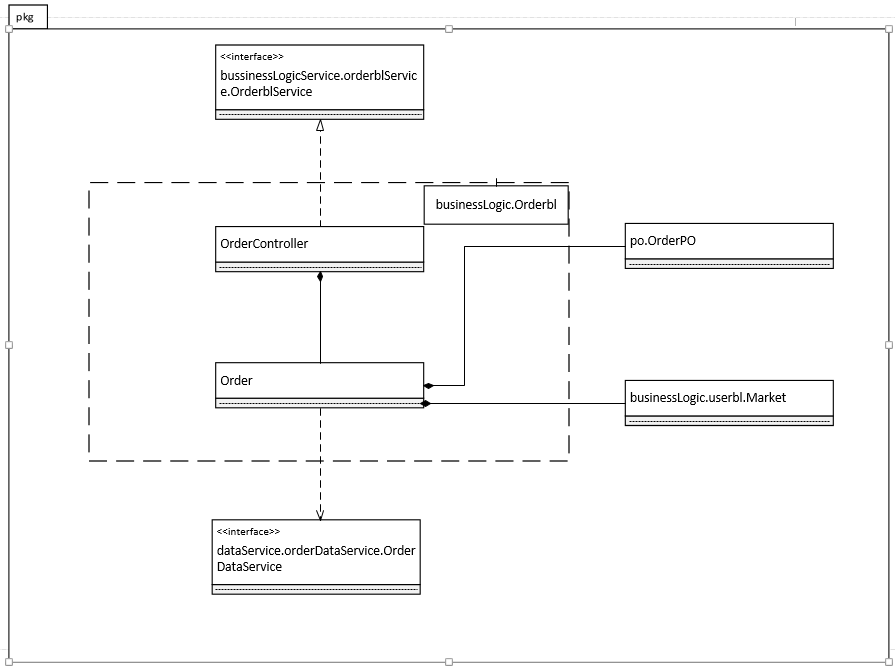


图4.1.1-1order模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| orderController | 负责实现所有与订单有关的界面所需的服务 |
| order | 订单的领域模型对象，拥有订单编号、开始时间、退房时间、最晚订单执行时间、房间类型及数量、预计入住人数、有无儿童等信息，可以帮助完成与订单有关界面的服务 |

表4.1.1-1 orderbl模块的各个类的职责

（2）模块内部类的接口规范

### 表4.1.1-2 OrderController类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| OrderController. returnBestPrice | 语法 | public OrderVO returnBestPrice (OrderVO order); | |
| 前置条件 | 有订单信息，需要根据订单信息返回最优价格 | |
| 后置条件 | 返回订单最优价格 | |
| OrderController.handleAbnormalOrder | 语法 | public boolean handleAbnormalOrder(OrderVO order，int percentOfCredit); | |
| 前置条件 | 处理异常订单，需要改变订单状态和恢复信用值的百分比 | |
| 后置条件 | 将订单状态变为已执行并且增加用户被扣除的信用值 | |
| OrderController.reverseOrder | 语法 | public boolean reverseOrder(OrderVO order); | |
| 前置条件 | 撤销订单，需要改变订单状态和用户信息 | |
| 后置条件 | 将订单状态变为已撤销，根据撤销时间减少用户信用值 | |
| OrderController.finishOrder | 语法 | public boolean finishOrder(OrderVO order); | |
| 前置条件 | 执行订单，需要改变订单状态和用户信息 | |
| 后置条件 | 将订单状态变为已执行并且增加客户信用值 | |
| OrderController.createOrder | 语法 | public boolean  createOrder(OrderVO order); | |
| 前置条件 | 生成订单，已知完整订单信息 | |
| 后置条件 | 返回是否成功生成订单 | |
| OrderController.personOrders | 语法 | Public ArrayList <OrderVO> personOrders(String personname); | |
| 前置条件 | 查看个人订单，已知用户名 | |
| 后置条件 | 根据用户名返回个人订单列表 | |
| OrderController.hotelOrders | 语法 | public ArrayList<OrderVO> hotelOrders(String hotelname); | |
| 前置条件 | 浏览酒店订单，已知酒店名称 | |
| 后置条件 | 根据酒店名称返回酒店订单列表 | |
| OrderController.netOrders | 语法 | public ArrayList<OrderVO> netOrders(); | |
| 前置条件 | 浏览网站订单，已经登录验证 | |
| 后置条件 | 返回网站未执行和异常订单列表 | |
| OrderController.personStateOrders | 语法 | public ArrayList<OrderVO> personStateOrders(String personname, String state); | |
| 前置条件 | 客户在个人订单查看过程中，需要进一步查看某个状态（未执行，已执行，已撤销，异常）的订单 | |
| 后置条件 | 返回客户符合条件的订单 | |
| OrderController.hotelStateOrders | 语法 | public ArrayList<OrderVO> hotelStateOrders(String hotelname, String state) | |
| 前置条件 | 酒店工作人员在酒店订单查看过程中，进一步查看某状态订单 | |
| 后置条件 | 返回符合条件的酒店订单 | |
| OrderController.netNumOrders | 语法 | public ArrayList<OrderVO> netNumOrders(String date)； | |
| 前置条件 | 网站营销人员在浏览网站订单的过程中，进一步查看某日订单 | |
| 后置条件 | 返回符合条件的网站订单 | |
| OrderController. getOrderInfo | 语法 | public OrderVO getOrderInfo(String orderID)； | |
| 前置条件 | 有根据订单ID得到订单详细信息的需求 | |
| 后置条件 | 返回相应订单详细信息 | |
| OrderController. changeOrderState | 语法 | public boolean changeOrderState(String orderID, String newState)； | |
| 前置条件 | 酒店工作人员需要改变订单状态 | |
| 后置条件 | 修改相应订单状态 | |
| OrderController. leaveRoom | 语法 | public boolean leaveRoom(String orderID)； | |
| 前置条件 | 客户离开酒店，酒店工作人员增添订单中的离开时间 | |
| 后置条件 | 修改相应订单状态，返回是否操作成功 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Order.returnBestPrice(OrderVO order) | | | 得到酒店最优价格 |
| Order.handleAbnormalOrder(OrderVO order，int percentOfCredit) | | | 处理一个异常订单 |
| Order.reverseOrder(OrderVO order) | | | 撤销一个未执行订单 |
| Order. finishOrder(OrderVO order) | | | 执行一个异常或未执行订单 |
| Order.createOrder(OrderVO order) | | | 生成并保存一个订单 |
| Order.personOrders(String username) | | | 查看个人订单 |
| Order.hotelOrders(String hotelname) | | | 浏览酒店订单 |
| Order.netOrders() | | | 浏览网站订单 |
| Order.personStateOrders(String personname, String state) | | | 浏览客户某状态的订单 |
| Order.hotelStateOrders(String hotelname, String state) | | | 浏览酒店某状态的订单 |
| Order.netNumOrders(String date) | | | 浏览网站某天的订单 |
| Order. getOrderInfo(String orderID) | | | 得到相应ID订单信息 |
| Order. changeOrderState(String orderID, String newState) | | | 改变某订单状态 |
| Order.leaveRoom(String orderID) | | | 改变订单中客户离开时间 |

表4.1.1-2 Order类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Order.handleAbnormalOrder | 语法 | public boolean handleAbnormalOrder(OrderVO order , int percentOfCredit); | |
| 前置条件 | 处理异常订单，需要改变订单状态和用户信息 | |
| 后置条件 | 将订单状态变为已执行并且增加用户被扣除的信用值 | |
| Order.reverseOrder | 语法 | public boolean reverseOrder(OrderVO order); | |
| 前置条件 | 撤销订单，需要改变订单状态和用户信息 | |
| 后置条件 | 将订单状态变为已撤销，根据撤销时间减少用户信用值 | |
| Order.finishOrder | 语法 | public boolean finishOrder(OrderVO order); | |
| 前置条件 | 执行订单，需要改变订单状态和用户信息 | |
| 后置条件 | 将订单状态变为已执行并且增加客户信用值 | |
| Order.createOrder | 语法 | public boolean  createOrder(OrderVO order); | |
| 前置条件 | 生成订单，已知完整订单信息 | |
| 后置条件 | 提交订单，持久化保存订单信息 | |
| Order.personOrders | 语法 | Public ArrayList <OrderVO> personOrders(String personname); | |
| 前置条件 | 查看个人订单，已知客户名 | |
| 后置条件 | 根据用户名返回个人订单列表 | |
| Order.hotelOrders | 语法 | public ArrayList<OrderVO> hotelOrders(String hotelname); | |
| 前置条件 | 浏览酒店订单，已知酒店名称 | |
| 后置条件 | 根据酒店名称返回酒店订单列表 | |
| Order.netOrders | 语法 | public ArrayList<OrderVO> netOrders(); | |
| 前置条件 | 浏览网站订单，已经登录验证 | |
| 后置条件 | 返回网站未执行和异常订单列表 | |
| Order.personStateOrders | 语法 | public ArrayList<OrderVO> personStateOrders(String personname, String state); | |
| 前置条件 | 客户在个人订单查看过程中，需要进一步查看某个状态（未执行，已执行，已撤销，异常）的订单 | |
| 后置条件 | 返回客户符合条件的订单 | |
| Order.hotelStateOrders | 语法 | public ArrayList<OrderVO> hotelStateOrders(String hotelname, String state) | |
| 前置条件 | 酒店工作人员在酒店订单查看过程中，进一步查看某状态订单 | |
| 后置条件 | 返回符合条件的酒店订单 | |
| Order.netNumOrders | 语法 | public ArrayList<OrderVO> netNumOrders(String date)； | |
| 前置条件 | 网站营销人员在浏览网站订单的过程中，进一步查看某日订单 | |
| 后置条件 | 返回符合条件的网站订单 | |
| Order. getOrderInfo | 语法 | public OrderVO getOrderInfo(String orderID)； | |
| 前置条件 | 有根据订单ID得到订单详细信息的需求 | |
| 后置条件 | 返回相应订单详细信息 | |
| Order.changeOrderState | 语法 | public boolean changeOrderState(String orderID, String newState)； | |
| 前置条件 | 酒店工作人员需要改变订单状态 | |
| 后置条件 | 修改相应订单状态 | |
| Order.leaveRoom | 语法 | public boolean leaveRoom(String orderID)； | |
| 前置条件 | 客户离开酒店，酒店工作人员增加订单中客户离开时间 | |
| 后置条件 | 修改相应订单状态 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderDataService.add(OrderPO order) | | | 新增一个订单信息 |
| OrderDataService.persoFind(String personname) | | | 返回个人订单列表 |
| OrderDataService.modify(OrderPO order) | | | 改变一个订单的状态 |
| Market.addCredit(String personname,int credit) | | | 改变个人信息中的信用值 |
| OrderDataService.hotelFind (String hotelname) | | | 返回单酒店订单列表 |
| OrderDataService.exceptionFind () | | | 返回未执行和异常订单列表 |
| orderDataService.getOrderInfo(orderID) | | | 返回订单信息 |

### （4） 业务逻辑层的动态模型

如图4.1.1-2表明了互联网酒店预订系统中，当客户填写完订单信息后，订单生成业务逻辑处理相关对象之间的协作。

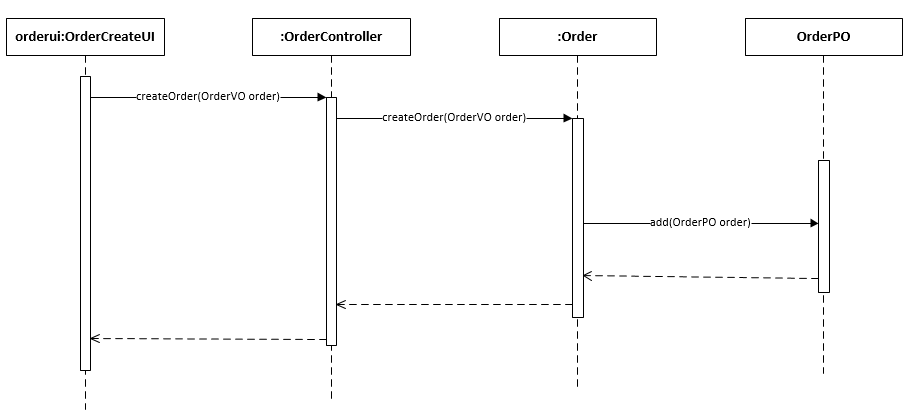


图4.1.1-2 订单生成的顺序图

如图4.1.1-3表明了互联网酒店预订系统中，当客户请求撤销未执行订单后，订单撤销业务逻辑处理相关对象之间的协作。

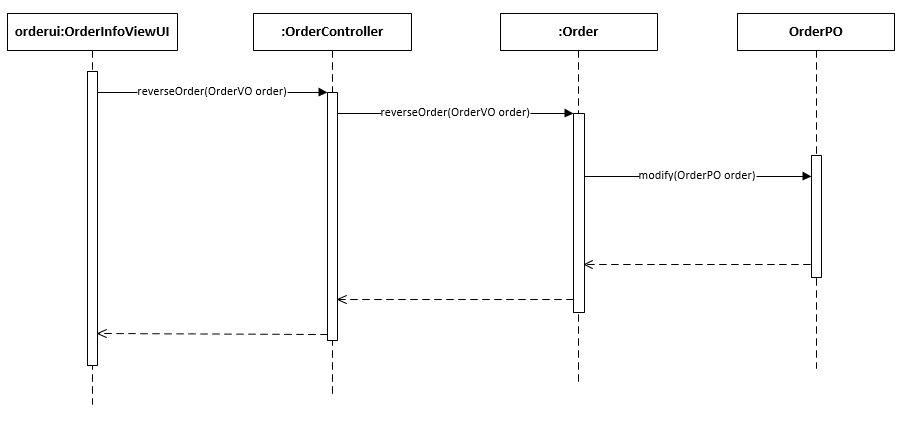


图4.1.1-3 订单撤销的顺序图

如图4.1.1-4表明了互联网酒店预订系统中，当客请求查看个人订单后，个人订单查看业务逻辑处理相关对象之间的协作。

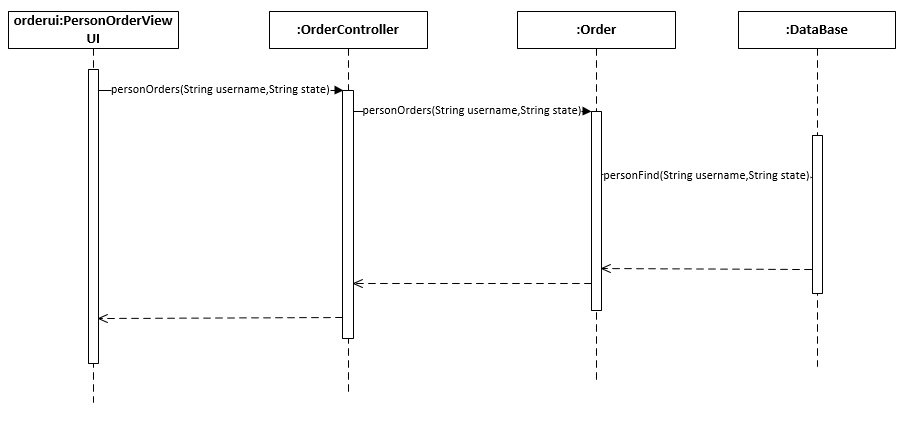


图4.1.1-4 个人订单查看的顺序图

如图4.1.1-5表明了互联网酒店预订系统中，当酒店工作人员请求浏览酒店订单后，酒店订单浏览业务逻辑处理相关对象之间的协作。

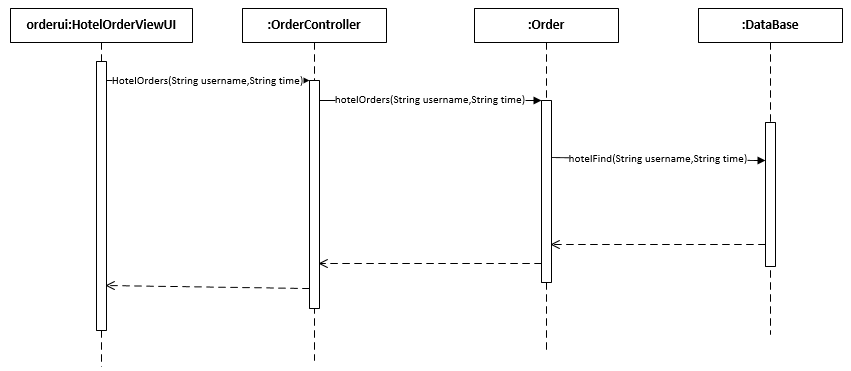


图4.1.1-5 酒店订单浏览的顺序图

如图4.1.1-6表明了互联网酒店预订系统中，当网站营销人员请求浏览订单后，网站订单浏览业务逻辑处理相关对象之间的协作。

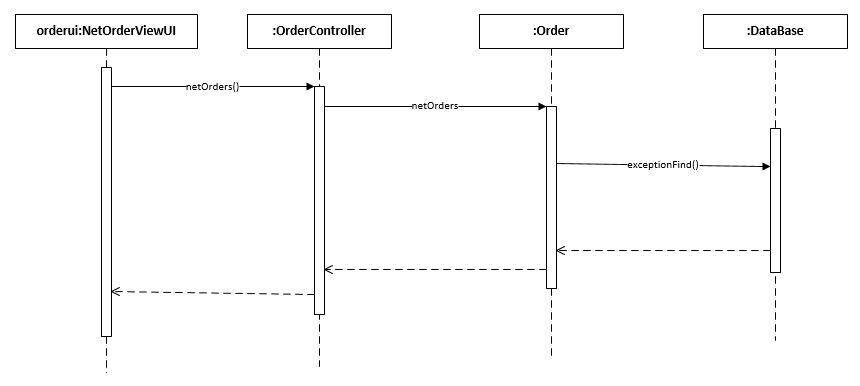


图4.1.1-6 网站订单浏览的顺序图

如图4.1.1-7表明了互联网酒店预订系统中，当网站营销人员请求处理异常订单后，异常订单处理业务逻辑处理相关对象之间的协作。

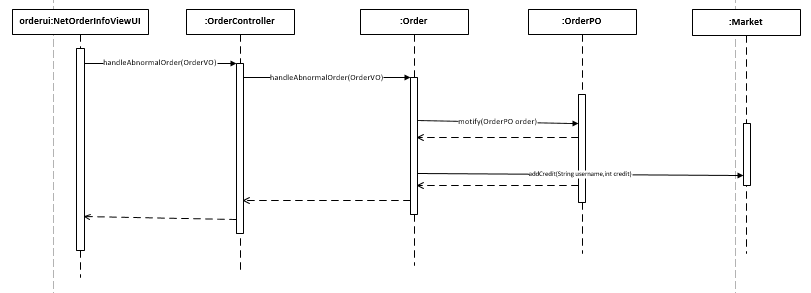


图4.1.1-7 异常订单处理的顺序图

如图4.1.1-2表明了互联网酒店预订系统中，当酒店工作人员请求执行订单后，订单执行业务逻辑处理相关对象之间的协作。

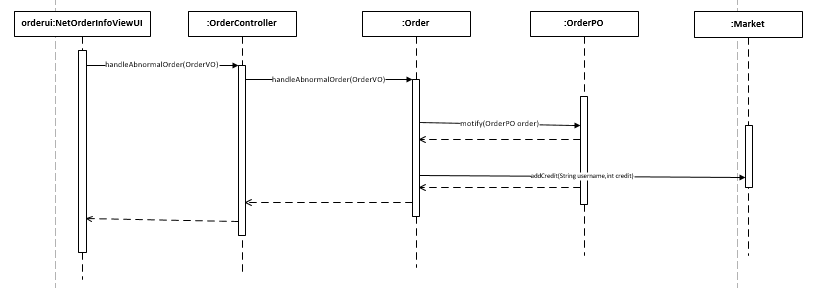


图4.1.1-8 订单执行的顺序图

图4.1.1-9描述了order对象生存时的状态序列。在login后对象进入loinged方法，并且可以由finishOrder、handleAbnormalOrder、reverseOrder、createOrder、personOrders、hotelOrders、netOrders的七个方法分别进入finishOrder、handleAbnormalOrder、reverseOrder、createOrder、personOrder、hotelOrder、netOrder的状态，最后在logout后对象生命结束。

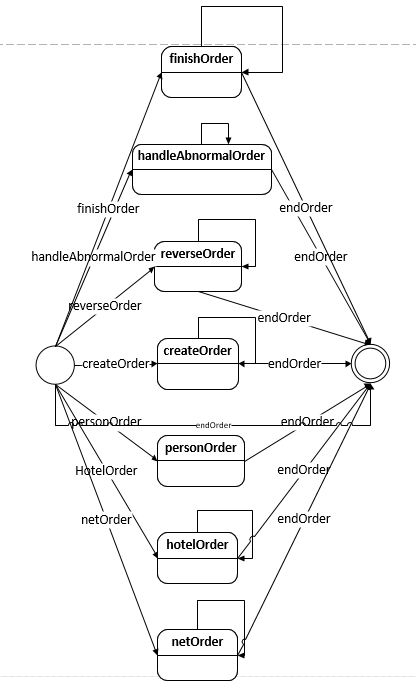


图4.1.1-9 Order对象的状态图

### 4.1.2 userbl模块

（1）模块概述

userbl模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

userbl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

userbl模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。userbl模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。为了隔离业务逻辑职责和逻辑控制职责，我们添加了UserController，这样UserController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给userbl 对象。UserPO是做为管理信息的持久化对象被添加到设计模型中的。

userbl的模块设计见图4.1.2-1。

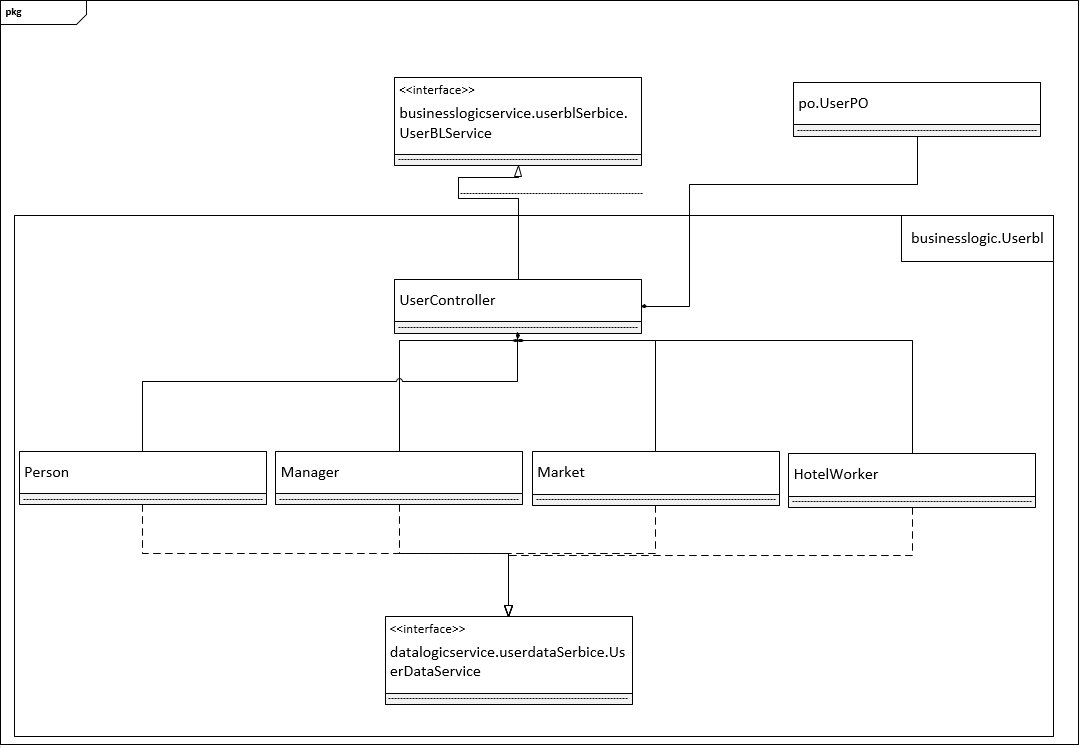


图4.1.2-1user模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| UserController | 负责实现对应于有关用户界面所需的方法 |
| Person | 负责处理有关客户的功能 |
| Manager | 负责处理有关网站管理人员的功能 |
| Market | 负责处理有关网站营销人员的功能 |
| HotelWorker | 负责处理有关酒店工作人员的功能 |

表4.1.2-1 userbl模块的各个类的职责

（2）模块内部类的接口规范

### 表4.1.2-2 UserController类的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserController.  Register | 语法 | public boolean register(PersonVO personinfo) |
| 前置条件 | 客户注册，已知注册信息 |
| 后置条件 | 根据注册信息更新网站用户信息并返回注册结果 |
| UserController. personSave | 语法 | public boolean personSave(PersonVO personinfo); |
| 前置条件 | 保存更改的客户信息，已知更改后客户信息 |
| 后置条件 | 更新客户信息并返回保存结果 |
| UserController.marketSave | 语法 | public boolean marketSave(MarketVO marketinfo); |
| 前置条件 | 保存更改的网站营销人员信息，已知更改后网站营销人员信息 |
| 后置条件 | 更新网站营销人员信息并返回保存结果 |
| UserController. hotelWorkerSave | 语法 | public boolean hotelWorkerSave(HotelWorkerVO hotelWorkerinfo); |
| 前置条件 | 保存更改的酒店工作人员信息，已知更改后酒店工作人员信息 |
| 后置条件 | 更新酒店工作人员信息并返回保存结果 |
| UserController.changeCredit | 语法 | public boolean changeCredit (String username,RecordVO recordvo) |
| 前置条件 | 已知增加的信用值，需要更新个人信息内的信用值 |
| 后置条件 | 更新个人信息并返回结果 |
| UserController.registerMember | 语法 | public boolean registeMember(PersonVO personinfo, String vipType, String vipInfo); |
| 前置条件 | 已知会员信息，需要向个人信息中加入会员信息 |
| 后置条件 | 更新个人信息并返回结果 |
| UserController.getPersonInfo | 语法 | public PersonVO getPersonInfo(String personname); |
| 前置条件 | 已知客户名，需要取得用户信息 |
| 后置条件 | 根据客户名返回客户信息 |
| UserController.getPersonInfo | 语法 | public PersonVO getPersonInfo(int personID); |
| 前置条件 | 已知客户ID，需要取得用户信息 |
| 后置条件 | 根据客户ID返回客户信息 |
| UserController.getHotelWorkerInfo | 语法 | public HotelWorkerVO getHotelWorkerInfo(String hotelWorkername); |
| 前置条件 | 已知酒店工作人员名，需要取得酒店工作人员信息 |
| 后置条件 | 根据酒店工作人员名返回(String personname)酒店工作人员信息 |
| UserController.userLogin | 语法 | public boolean userLonin (String username,String password，String usertype) | |
| 前置条件 | 用户已经输入了用户名和密码 | |
| 后置条件 | 返回登陆是否通过 | |
| UserController. marketAdd | 语法 | public boolean marketAdd (MarketVO Marketinfo); |
| 前置条件 | 增加一个网站营销人员信息，已知新增的网站营销人员信息 |
| 后置条件 | 新增酒店工作人员信息并返回保存结果 |
| UserController. hotelWorkerAdd | 语法 | public boolean hotelWorkerAdd (HotelWorkerVO Marketinfo); |
| 前置条件 | 增加一个酒店工作人员信息，已知新增的酒店工作人员信息 |
| 后置条件 | 新增酒店工作人员信息并返回保存结果 |
| UserController.getMarketInfo | 语法 | public PersonVO getMarketInfo(String marketname); |
| 前置条件 | 已知网站营销人员名，需要取得网站营销人员信息 |
| 后置条件 | 根据网站营销人员名返回网站营销人员信息 |
| UserController.isExist | 语法 | public boolean isExist(String username, String usertype); |
| 前置条件 | 有判断某用户是否存在的需求 |
| 后置条件 | 返回是否存在结果 |
| UserController.writeRecord | 语法 | public boolean writeRecord(String personname, RecordVO record)； |
| 前置条件 | 有写入信用记录的需求 |
| 后置条件 | 写入信用记录 |
| UserController.getRecord | 语法 | public ArrayList<RecordVO> getRecord(String personname)； |
| 前置条件 | 有获取客户信用记录的需求 |
| 后置条件 | 返回获取的信用记录 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Person.register(PersonVO personinfo) | | 客户登陆 |
| Person. personSave(PersonVO personinfo) | | 客户信息保存 |
| Person.registerMember(PersonVO personinfo) | | 客户注册会员 |
| Person.getPersonInfo(String personname) | | 获得客户信息 |
| Person.personLogin(String personname,String password) | | 客户登陆 |
| Person.writeRecord(String personname, RecordVO record) | | 写入客户信用记录 |
| Person.getRecord(String personname) | | 获取客户信用记录 |
| Person.isExist(String personname) | | 判断客户是否存在 |
| Market. marketSave(MarketVO marketinfo) | | 保存网站营销人员信息 |
| Market.getPersonInfo(String username) | | 获得客户信息 |
| Market.getMarketInfo(String username) | | 获得网站营销人员信息 |
| Market.addCredit String personname,int credit) | | 修改客户信用值 |
| Market.marketLogin(String marketname,String password) | | 网站营销人员登陆 |
| Market.isExist(String marketname) | | 判断网站营销人员是否存在 |
| HotelWorker. hotelWorkerSave(HotelWorkerVO hotelWorkerinfo) | | 保存酒店工作人员信息 |
| HotelWorker.getHotelWorkerInfo(String hotelWorkername) | | 获得酒店工作人员信息 |
| HotelWorker.hotelWorkerLogin(String hotelWorkername,String password) | | 酒店工作人员登陆 |
| HotelWorker.isExist(String hotelworkername) | | 判断酒店工作人员是否存在 |
| Manager. hotelWorkerSave(HotelWorkerVO hotelWorkerinfo) | | 保存酒店工作人员信息 |
| Manager.getPersonInfo(String personname) | | 获得酒店工作人员信息 |
| Manager.getHotelWorkerInfo(String hotelWorkername) | | 获得酒店工作人员信息 |
| Manager.managerLogin(String managername,String password) | | 网站营销人员登陆 |
| Manager. marketAdd(MarketVO Marketinfo) | | 新增网站营销人员信息 |
| Manager. hotelWorkerAdd(HotelWorkerVO Marketinfo) | | 新增酒店工作人员信息 |
| Manager.getMarketInfo(String marketname) | | 获得网站营销人员信息 |

表4.1.2-3 Person类的的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Person.register | 语法 | public boolean register(PersonVO personinfo) | |
| 前置条件 | 客户注册，已知注册信息 | |
| 后置条件 | 根据注册信息更新网站用户信息并返回注册结果 | |
| Person. modifyPerson | 语法 | public boolean modifyPerson (PersonVO personinfo); | |
| 前置条件 | 保存更改的客户信息，已知更改后客户信息 | |
| 后置条件 | 更新客户信息并返回保存结果 | |
| Person.registerMember | 语法 | public boolean registerMember(PersonVO personvo, String vipType, String vipInfo); | |
| 前置条件 | 已知会员信息，需要向个人信息中加入会员信息 | |
| 后置条件 | 更新个人信息并返回结果 | |
| Person.getPersonInfo | 语法 | public PersonPO getPersonInfo(String personname); | |
| 前置条件 | 已知客户名，需要取得用户信息 | |
| 后置条件 | 根据客户名返回客户信息 | |
| Person.getPersonInfo | 语法 | public PersonVO getPersonInfo(int personID); | |
| 前置条件 | 已知客户ID，需要取得用户信息 | |
| 后置条件 | 根据客户ID返回客户信息 | |
| Person.personLogin | 语法 | public boolean personLogin(String personname,String password) | |
| 前置条件 | 客户已经输入了客户名和密码 | |
| 后置条件 | 返回登陆是否通过 | |
| Person.isExist | 语法 | public boolean isExist(String personname); | |
| 前置条件 | 有判断某客户是否存在的需求 | |
| 后置条件 | 返回是否存在结果 | |
| Person.writeRecord | 语法 | public boolean writeRecord(String personname, RecordVO record)； | |
| 前置条件 | 有写入信用记录的需求 | |
| 后置条件 | 写入信用记录 | |
| Person.getRecord | 语法 | public ArrayList<RecordVO> getRecord(String personname)； | |
| 前置条件 | 有获取客户信用记录的需求 | |
| 后置条件 | 返回获取的信用记录 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| userData.addPerson (personPO personInfo) | | | 新增客户信息 |
| userData.findPerson (String personname) | | | 寻找客户信息 |
| userData.modifyPerson (personPO personInfo) | | | 改变客户信息 |
| userData.personLogin(String personname,String password) | | | 验证登陆是否通过 |
| userData.writeRecord(String personname, RecordPO record); | | | 写入客户信用记录 |
| userData. getRecord(String personname); | | | 获得客户信用记录 |

表4.1.2-4 Market类的的接口规范

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 提供的服务（供接口） | | | | | |
| Market.merketLogin | | 语法 | public boolean marketlogin(String marketname,String password); | |
| 前置条件 | 已知用户名和密码，进行登陆 | |
| 后置条件 | 网站营销人员登陆 | |
| Market.getPersonInfo | | 语法 | public PersonPO getPersonInfo(String personname); | |
| 前置条件 | 已知客户名，需要取得用户信息 | |
| 后置条件 | 根据客户名返回客户信息 | |
| Market.getMarketInfo | 语法 | | public MarketVO getMarketInfo(String marketname); | | |
| 前置条件 | | 已知网站营销人员名，需要取得网站营销人员信息 | | |
| 后置条件 | | 根据网站营销人员名返回网站营销人员信息 | | |
| Manager.changeCredit | | 语法 | public boolean changeCredit (String personname,RecordVO recordvo) | |
| 前置条件 | 已知增加的信用值，需要更新个人信息内的信用值 | |
| 后置条件 | 更新个人信息并返回结果 | |
| Market.isExist | 语法 | | public boolean isExist(String marketname); | | |
| 前置条件 | | 已知网站营销人员名称，判断是否存在 | | |
| 后置条件 | | 返回判断结果 | | |
| 需要的服务（需接口） | | | | | |
| 服务名 | | | | 服务 | |
| userData.findMarket (String marketname) | | | | 寻找网站营销人员信息 | |
| userData.modifyMarket (marketPO marketInfo) | | | | 改变网站营销人员信息 | |
| userData.marketLogin(String marketname,String password) | | | | 验证登陆是否通过 | |
| userData.addCredit(personPO personInfo) | | | | 增加客户信用值 | |

表4.1.2-5 hotelWorker类的的接口规范

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 提供的服务（供接口） | | | | |
| HotelWorker.getHotelWorkerInfo | 语法 | | public HotelWorkerPO getHotelWorkerInfo(String hotelWorkername); | |
| 前置条件 | | 已知酒店工作人员名，需要取得酒店工作人员信息 | |
| 后置条件 | | 根据酒店工作人员名返回酒店工作人员信息 | |
| HotelWorker .isExist | 语法 | public boolean isExist(String hotelworkername); | | |
| 前置条件 | 已知酒店工作人员名称，判断是否存在 | | |
| 后置条件 | 返回判断结果 | | |
| HotelWorker.hotelWorkerLogin | 语法 | | public boolean HotelWorkerLogin(String hotelWorkername,String password) | |
| 前置条件 | | 酒店工作人员已经输入了酒店工作人员名和密码 | |
| 后置条件 | | 返回登陆是否通过 | |
| 需要的服务（需接口） | | | | |
| 服务名 | | | | 服务 |
| userData.findHotelWorker (String hotelWorkername) | | | | 寻找酒店工作人员信息 |
| userData.modifyHotelWorker (hotelWorkerPO hotelWorkerInfo) | | | | 改变酒店工作人员信息 |
| userData.hotelWorkerLogin(String hotelWorkername,String password) | | | | 验证登陆是否通过 |

表4.1.2-6 Manager类的接口设计

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Manager.marketSave | 语法 | public boolean marketSave(MarketVO marketinfo); |
| 前置条件 | 保存更改的网站营销人员信息，已知更改后网站营销人员信息 |
| 后置条件 | 更新网站营销人员信息并返回保存结果 |
| Manager. hotelWorkerSave | 语法 | public boolean hotelWorkerSave(HotelWorkerVO hotelWorkerinfo); |
| 前置条件 | 保存更改的酒店工作人员信息，已知更改后酒店工作人员信息 |
| 后置条件 | 更新酒店工作人员信息并返回保存结果 |
| Manager.getPersonInfo | 语法 | public PersonPO getPersonInfo(String personname); |
| 前置条件 | 已知客户名，需要取得用户信息 |
| 后置条件 | 根据客户名返回客户信息 |
| Manager.getHotelWorkerInfo | 语法 | public HotelWorkerPO getHotelWorkerInfo(String hotelWorkername); |
| 前置条件 | 已知酒店工作人员名，需要取得酒店工作人员信息 |
| 后置条件 | 根据酒店工作人员名返回酒店工作人员信息 |
| Manager.managerLogin | 语法 | public boolean personLonin (String managername,String password) | |
| 前置条件 | 网站管理人员已经输入了网站管理人员名和密码 | |
| 后置条件 | 返回登陆是否通过 | |
| Manager. marketAdd | 语法 | public boolean marketAdd (MarketVO Marketinfo); |
| 前置条件 | 增加一个网站营销人员信息，已知新增的网站营销人员信息 |
| 后置条件 | 新增酒店工作人员信息并返回保存结果 |
| Manager. hotelWorkerAdd | 语法 | public boolean hotelWorkerAdd (HotelWorkerVO Marketinfo); |
| 前置条件 | 增加一个酒店工作人员信息，已知新增的酒店工作人员信息 |
| 后置条件 | 新增酒店工作人员信息并返回保存结果 |
| Manager.getMarketInfo | 语法 | public PersonPO getMarketInfo(String marketname); |
| 前置条件 | 已知网站营销人员名，需要取得网站营销人员信息 |
| 后置条件 | 根据网站营销人员名返回网站营销人员信息 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| userData.addMarket(marketPO marketInfo) | | 新增网站营销人员信息 |
| userData.addHotelWorker(hotelWorkerPO hotelWorkerInfo) | | 新增酒店工作人员信息 |
| userData.findMarket(String marketname) | | 寻找网站营销人员信息 |
| userData.findHotelWorker(String hotelWorkername) | | 寻找酒店工作人员信息 |
| userData.findPerson(String personrname) | | 寻找客户信息 |
| userData.modifyMarket(marketPO marketInfo) | | 改变网站营销人员信息 |
| userData.modifyHotelWorker(hotelWorkerPO hotelWorkerInfo) | | 改变酒店工作人员信息 |

### （4） 业务逻辑层的动态模型

图4.1.2-2是网站管理人员输入用户名和密码登录时，各个对象之间的协作关系

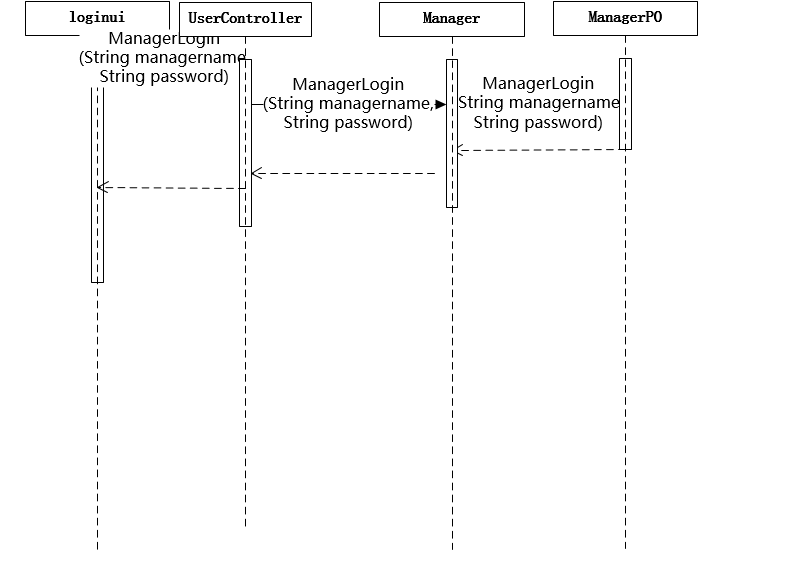


图4.1.2-2网站管理人员登陆顺序图

图4.1.2-3是酒店工作人员输入用户名和密码后登陆并查询酒店工作人员信息时各个对象的协作关系

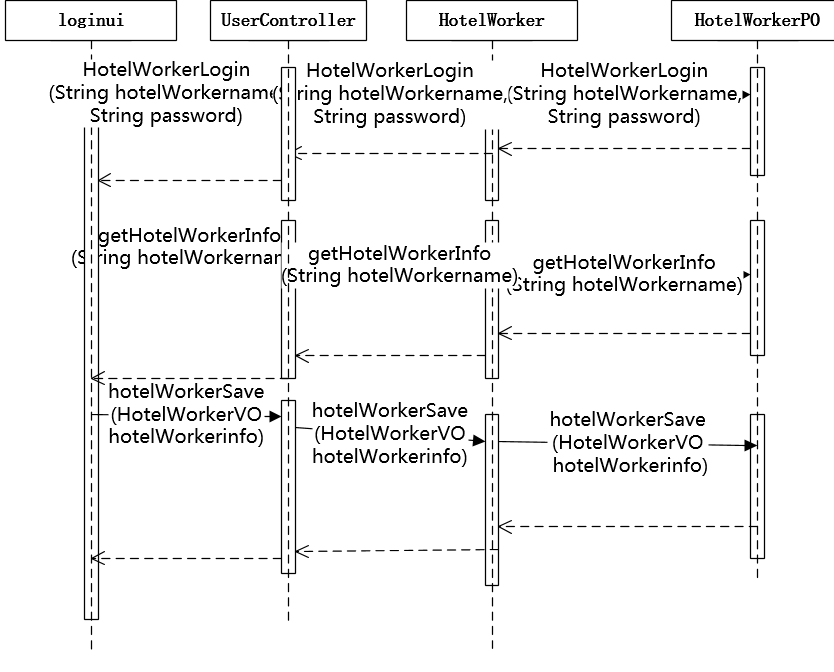


图4.1.2-3酒店工作人员登陆顺序图

图4.1.2-4是网站营销人员输入用户名和密码登陆、查询网站管理人员信息时各个对象的协作关系。

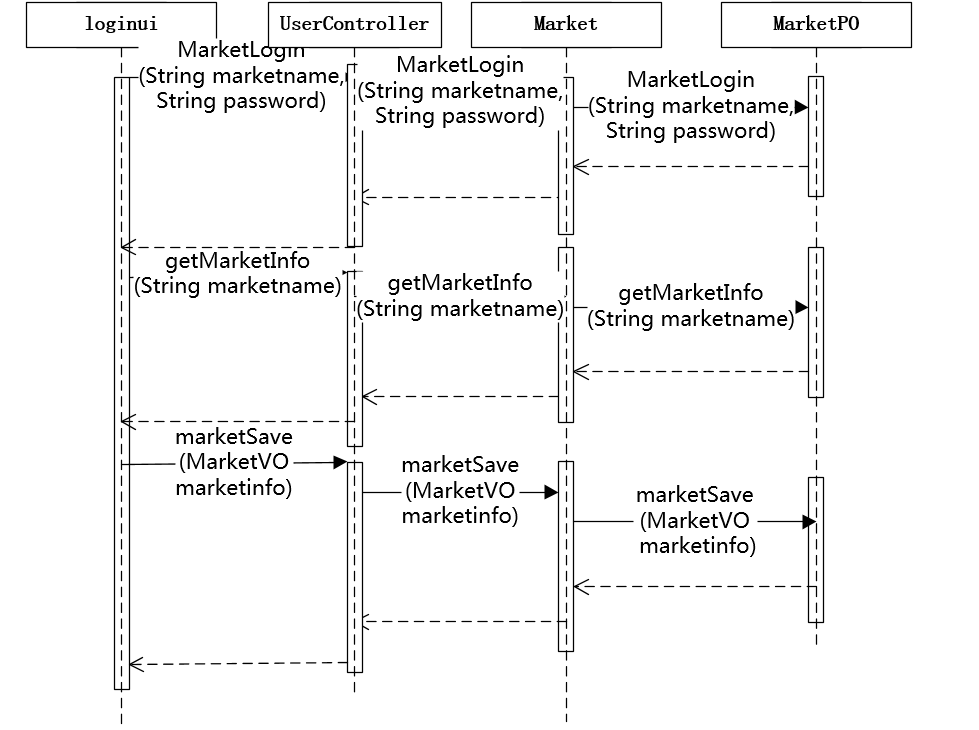


图4.1.2-4网站营销人员登陆、查看顺序图

图4.1.2-5是网站管理人员查看客户信息时各个对象的协作关系

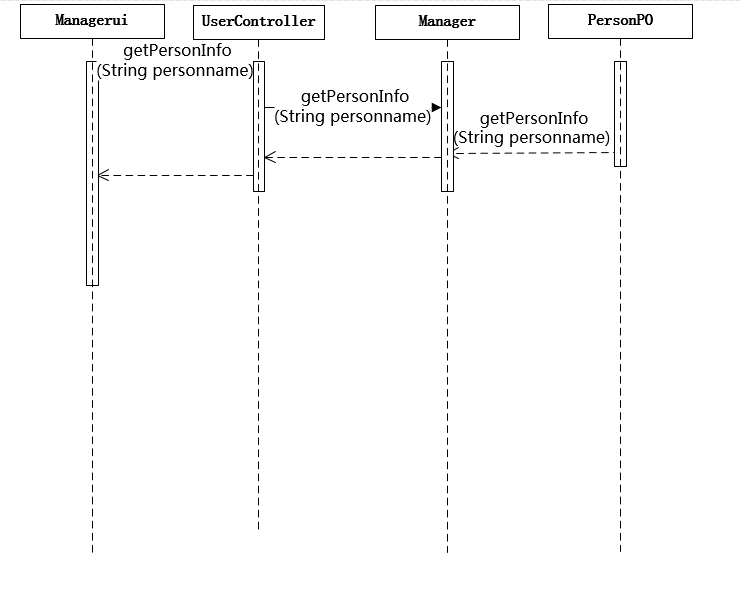


图4.1.2-5网站管理人员查看客户信息顺序图

图4.1.2-6是网站管理人员新增、查看、修改酒店工作人员信息时各个对象协作的顺序图

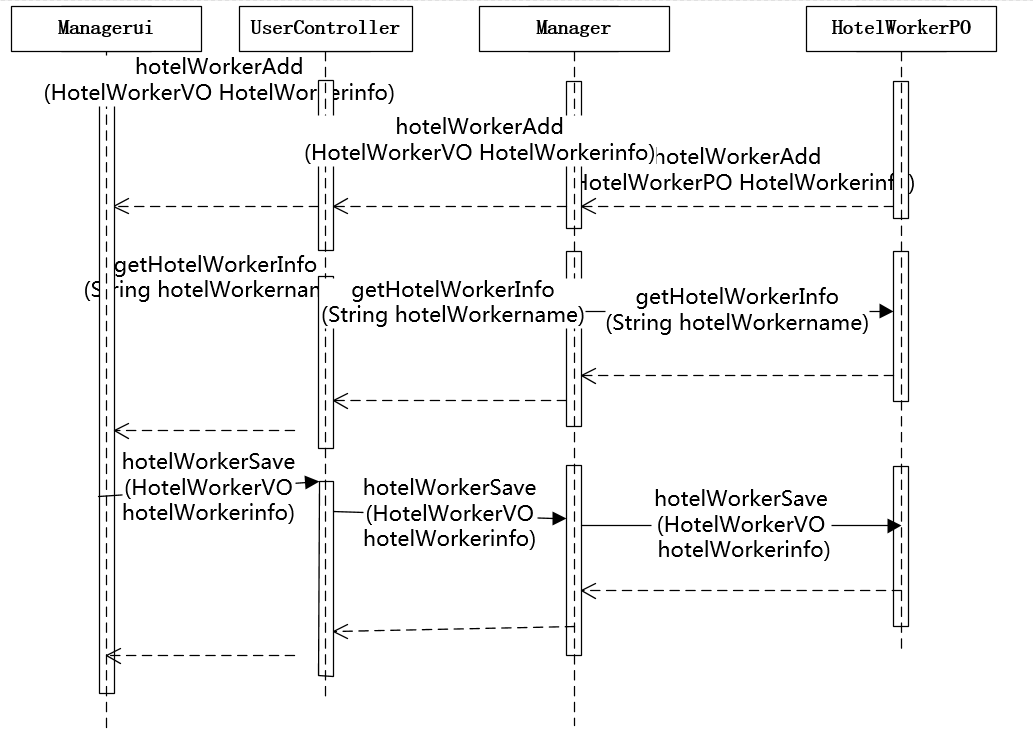


图4.1.2-6网站管理人员管理酒店工作人员顺序图

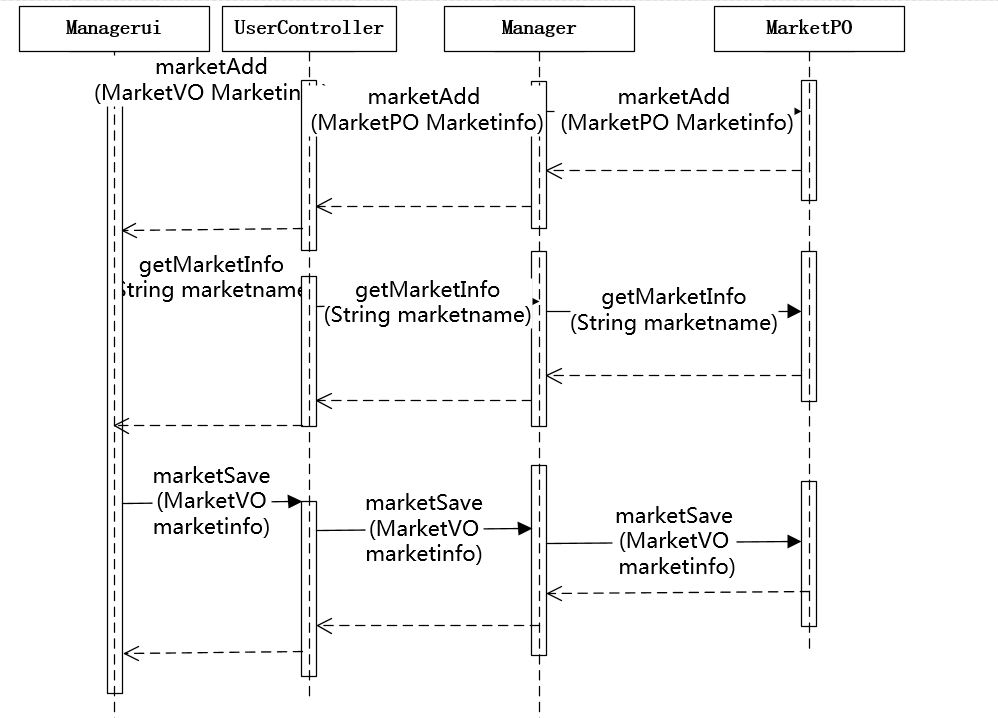
图4.1.2-7是网站管理人员新增、查看、修改网站营销人员信息时各个对象协作的顺序图

图4.1.2-7网站管理人员新增、查看、修改网站营销人员顺序图

图4.1.2-8是客户查看、修改个人信息时各个对象协作的顺序图

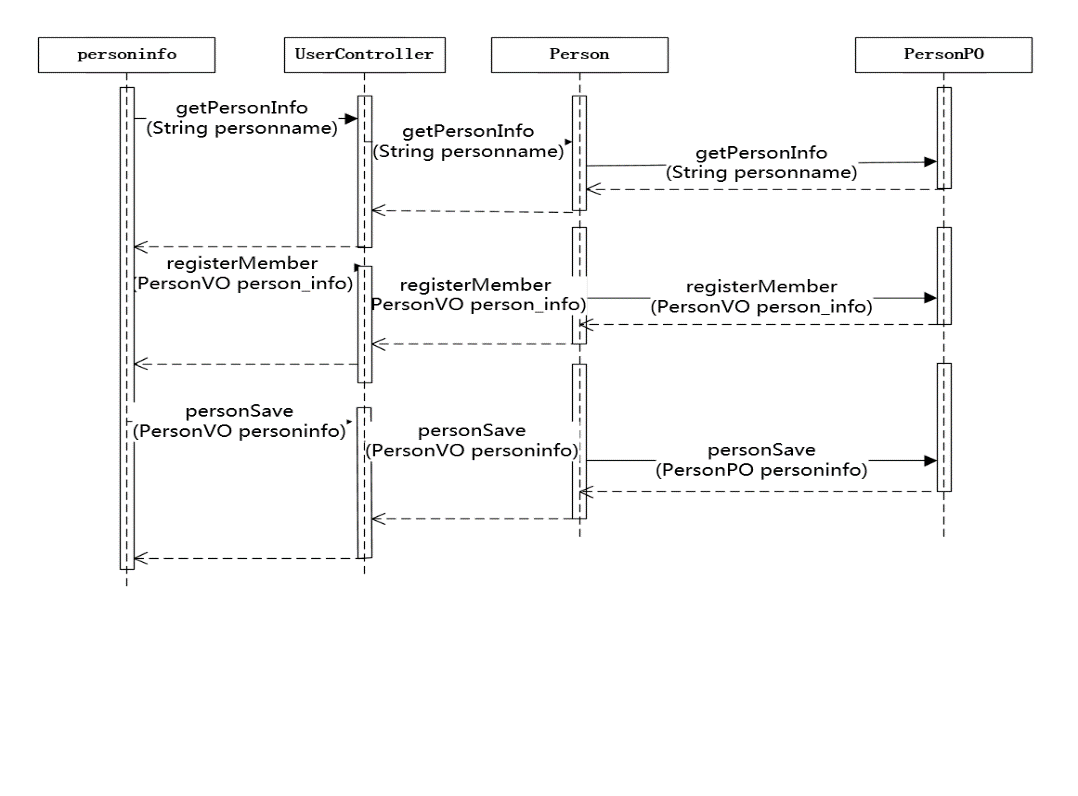


图4.1.2-8客户查看、修改个人信息顺序图

图4.1.2-9是客户注册、登陆时各个对象协作的顺序图

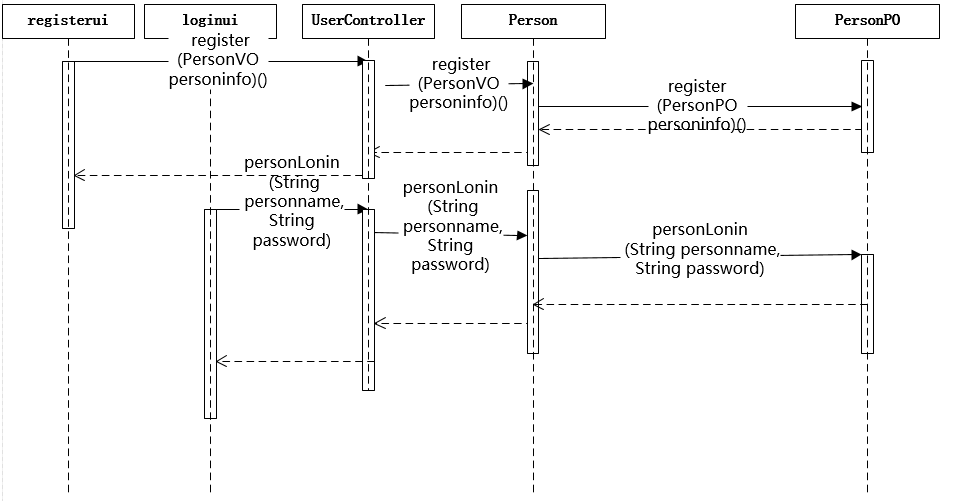


图4.1.2-9客户注册、登陆顺序图

图4.1.2-10描述了manager对象生存时的状态序列。在login后对象进入loinged方法，并且可以由view market、view person、view hotelworker的三个方法分别进入market view、person view、hotelworker view的状态，最后在logout后对象生命结束。

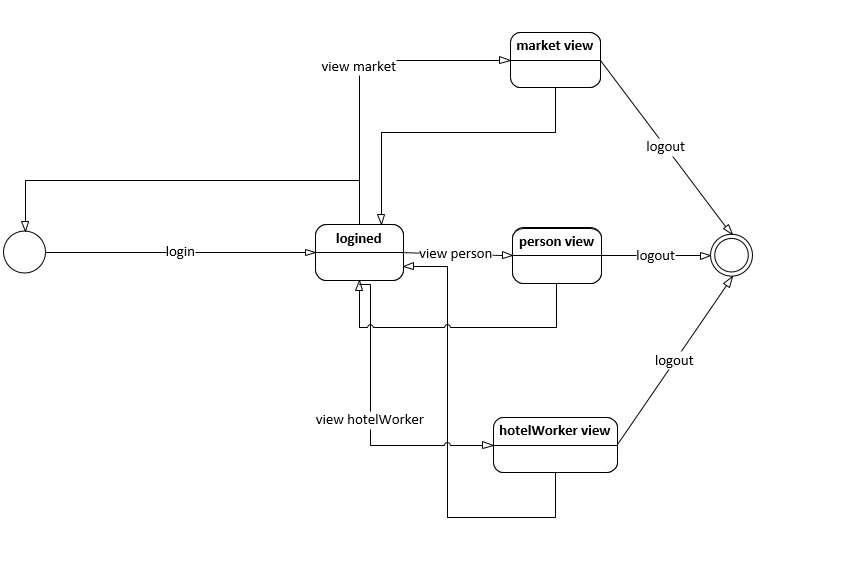


图4.1.2-10Manager对象状态图

图4.1.2-11描述了person对象在生存时的状态序列，person在register后进入wait for login状态，在login后进入logined状态，在view info后进入person view状态，并且在logout后结束生命

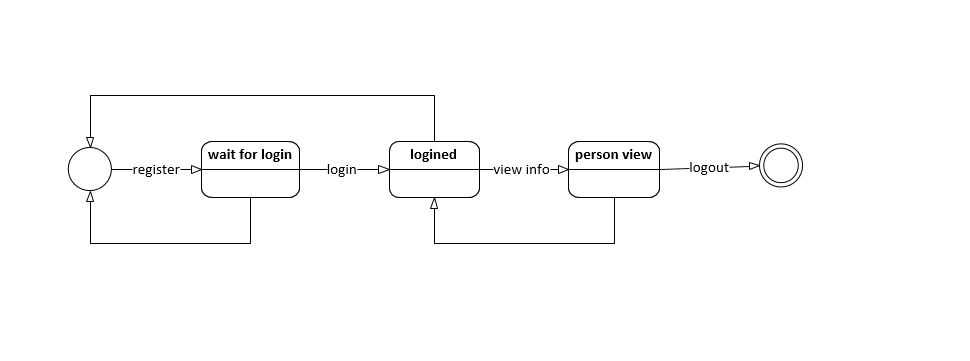


图4.1.2-11Person对象状态图

图4.1.2-12描述了Market对象在生存时的状态序列，在login后进入logined状态，在view info后进入market view状态，并且在logout后结束生命

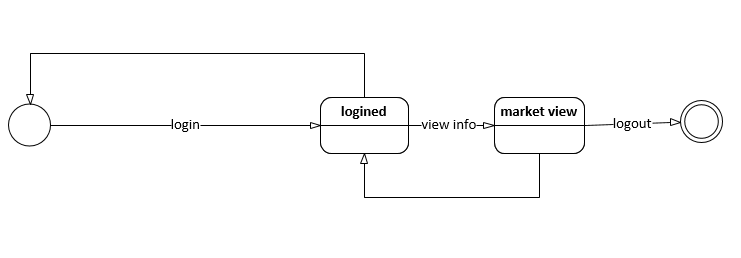


图4.1.2-12Market对象状态图

图4.1.2-13描述了HotelWorker对象在生存时的状态序列，在login后进入logined状态，在view info后进入HotelWorke view状态，并且在logout后结束生命

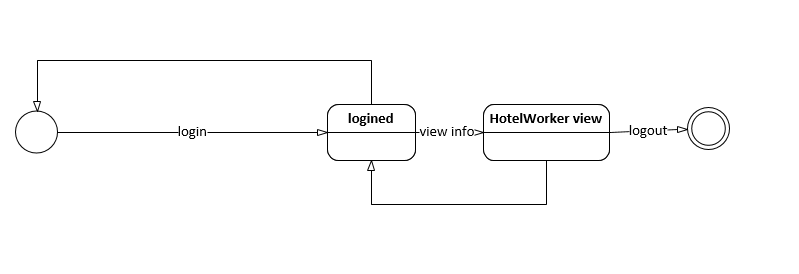


图4.1.2-13HotelWorker对象状态图

### 4.1.3 hotelbl模块

（1）模块概述

hotelbl模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

hotelbl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，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加的接口包括hotelblService , hotelDataService 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加HotelController，这样HotelController 将会将订单相关的业务逻辑职责和逻辑控制委托给hotel对象。

hotelbl模块的设计如图4.1.3-1所示.

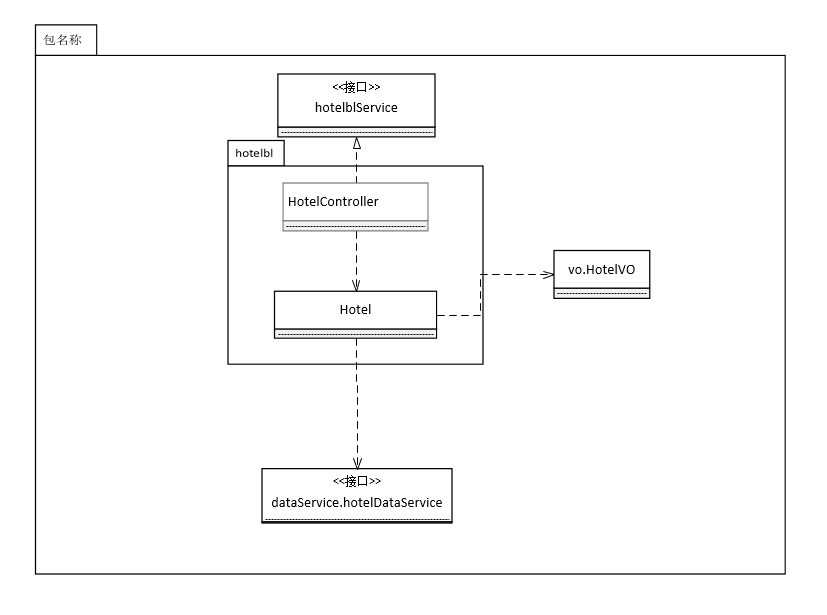


图4.1.3-1hotel模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| HotelController | 负责实现对应于有关酒店界面所需的方法 |
| Hotel | 负责处理有关酒店的功能 |

表4.1.3-1 hotelbl模块的各个类的职责

（2）模块内部类的接口规范

### 表4.1.3-2 HotelController类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelController. getHotelInfoByHotelworkerOrManager | 语法 | public HotelVO getHotelInfoByHotelworkerOrManager(String Hotelname)； | |
| 前置条件 | 酒店工作人员或网站管理人员，已知酒店编号，需要返回酒店的信息 | |
| 后置条件 | 返回酒店信息 | |
| HotelController. getHotelInfoByPerson | 语法 | public HotelVO getHotelInfoByPerson(String Hotelname)； | |
| 前置条件 | 客户已知酒店编号，需要返回酒店的信息 | |
| 后置条件 | 返回酒店信息 | |
| HotelController.modifyHotelInfo | 语法 | public boolean modifyHotelinfo(HotelVO hotelinfo); | |
| 前置条件 | 酒店工作人员维护酒店的信息 | |
| 后置条件 | 数据层修改酒店的信息 | |
| HotelController.roomModify | 语法 | public boolean roomModify(String hotelname, ArrayList<RoomVO> roomvoList) | |
| 前置条件 | 酒店工作人员修改酒店房间数量 | |
| 后置条件 | 更新酒店的房间数量 | |
| HotelController.addComment | 语法 | public boolean addcomment(CommentVO commentvo); | |
| 前置条件 | 用户需要评论酒店 | |
| 后置条件 | 数据层增加酒店评论 | |
| HotelController. findWithReq | 语法 | public ArrayList<HotelSearchVO> findWithReq(HotelConditionVO worstCondition, HotelConditionVO bestCondition) | |
| 前置条件 | 根据条件得到符合条件的酒店 | |
| 后置条件 | 返回符合条件的酒店 | |
| HotelController. addHotel | 语法 | public boolean addHotel(HotelVO hotelvo) | |
| 前置条件 | 需要增加酒店 | |
| 后置条件 | 增加酒店 | |
| HotelController. findReqRoom | 语法 | public RoomInfoVO findReqRoom(String hotelname, String roomtype, String starttime, String endtime)； | |
| 前置条件 | 有获得酒店符合要求房间的需求 | |
| 后置条件 | 返回符合条件的对应酒店的房间 | |
| HotelController. getAvailableNumber | 语法 | public int getAvailableNumber(String hotelname, String roomtype, String starttime, String endtime)； | |
| 前置条件 | 获得对应酒店某房间类型的可用房间数量 | |
| 后置条件 | 返回对应数量 | |
| HotelController. getHotelRoomInfo | 语法 | public ArrayList<RoomInfoVO> getHotelRoomInfo(String hotelname)； | |
| 前置条件 | 有根据酒店名称返回对应酒店的房间信息的需求 | |
| 后置条件 | 返回对应酒店的房间信息 | |
| HotelController. addRoom | 语法 | public boolean addRoom(String roomtype, int roomnum, int roomprice, String hotelname)； | |
| 前置条件 | 根据需求新增某酒店的某类型房间 | |
| 后置条件 | 增添对应房间信息，并返回是否增添成功 | |
| HotelController. editRoom | 语法 | public boolean editRoom(String roomtype, int roomnum, int roomprice, String hotelname)； | |
| 前置条件 | 酒店工作人员有修改酒店某类型的房间数量和价格的需求 | |
| 后置条件 | 修改酒店房间信息，并返回是否修改成功 | |
| HotelController. getHighestPrice | 语法 | public int getHighestPrice(String hotelname)； | |
| 前置条件 | 需要获取某酒店中最贵的房间价格 | |
| 后置条件 | 返回酒店中最贵的房间价格 | |
| HotelController. removeRoom | 语法 | public boolean removeRoom(String roomtype, String hotelName)； | |
| 前置条件 | 酒店工作人员有删除某酒店的某类型的所有房间的需求 | |
| 后置条件 | 删除对应房间信息，并返回是否删除成功 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Hotel.getHotelInfoByHotelworkerOrManager(Hotelname); | | | 非客户根据酒店名返回酒店信息 |
| Hotel. getHotelInfoByPerson(String Hotelname) | | | 客户根据酒店名返回酒店信息 |
| Hotel.modify(HotelVO hotelinfo) | | | 修改酒店信息 |
| Hotel. roomModify(String roomtype,int number) | | | 修改酒店房间信息 |
| Hotel.addComment（CommentPO commentpo) | | | 增加评论 |
| Hotel.findWithReq(HotelConditionVOworstCondition, HotelConditionVO bestCondition) | | | 根据条件区间搜索符合条件的酒店 |
| Hotel. addHotel(HotelVO hotelvo) | | | 增加酒店 |
| Hotel. findReqRoom(String hotelname, String roomtype, String starttime, String endtime) | | | 搜索符合条件的酒店房间 |
| Hotel. getAvailableNumber(String hotelname, String roomtype, String starttime, String endtime) | | | 返回对应可用房间数量 |
| Hotel. getHotelRoomInfo(String hotelname) | | | 搜索对应酒店房间信息 |
| Hotel. addRoom(String roomtype, int roomnum, int roomprice, String hotelname) | | | 新增酒店房间 |
| Hotel.editRoom(String roomtype, int roomnum, int roomprice, String hotelname) | | | 修改酒店房间 |
| Hotel. getHighestPrice(String hotelname) | | | 获取某酒店中最贵的房间价格 |
| Hotel.removeRoom(String roomtype, String hotelName) | | | 删除酒店房间 |

表4.1.3-3 hotel类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Hotel.getHotelInfoByHotelworkerOrManager | 语法 | public HotelVO getHotelInfoByHotelworkerOrManager(String Hotelname)； | |
| 前置条件 | 酒店工作人员或网站管理人员，已知酒店编号，需要返回酒店的信息 | |
| 后置条件 | 返回酒店信息 | |
| Hotel.getHotelInfoByPerson | 语法 | public HotelVO getHotelInfoByPerson(String Hotelname)； | |
| 前置条件 | 客户已知酒店编号，需要返回酒店的信息 | |
| 后置条件 | 返回酒店信息 | |
| Hotel.modifyHotelInfo | 语法 | public boolean modifyHotelinfo(HotelVO hotelinfo); | |
| 前置条件 | 酒店工作人员维护酒店的信息 | |
| 后置条件 | 数据层修改酒店的信息 | |
| Hotel.roomModify | 语法 | public boolean roomModify(String roomtype,int number) | |
| 前置条件 | 酒店工作人员修改酒店房间数量 | |
| 后置条件 | 更新酒店的房间数量 | |
| Hotel. findWithReq | 语法 | public ArrayList<HotelSearchVO> findWithReq(HotelConditionVO worstCondition, HotelConditionVO bestCondition) | |
| 前置条件 | 根据条件得到符合条件的酒店 | |
| 后置条件 | 返回符合条件的酒店 | |
| Hotel. addHotel | 语法 | public boolean addHotel(HotelVO hotelvo) | |
| 前置条件 | 需要增加酒店 | |
| 后置条件 | 增加酒店 | |
| Hotel.addComment | 语法 | public boolean addcomment(CommentVO commnetvo); | |
| 前置条件 | 用户需要评论酒店 | |
| 后置条件 | 数据层增加酒店评论 | |
| Hotel. findReqRoom | 语法 | public RoomInfoVO findReqRoom(String hotelname, String roomtype, String starttime, String endtime)； | |
| 前置条件 | 有获得酒店符合要求房间的需求 | |
| 后置条件 | 返回符合条件的对应酒店的房间 | |
| Hotel.getAvailableNumber | 语法 | public int getAvailableNumber(String hotelname, String roomtype, String starttime, String endtime)； | |
| 前置条件 | 获得对应酒店某房间类型的可用房间数量 | |
| 后置条件 | 返回对应数量 | |
| Hotel.getHotelRoomInfo | 语法 | public ArrayList<RoomInfoVO> getHotelRoomInfo(String hotelname)； | |
| 前置条件 | 有根据酒店名称返回对应酒店的房间信息的需求 | |
| 后置条件 | 返回对应酒店的房间信息 | |
| Hotel.addRoom | 语法 | public boolean addRoom(String roomtype, int roomnum, int roomprice, String hotelname)； | |
| 前置条件 | 根据需求新增某酒店的某类型房间n个 | |
| 后置条件 | 新增酒店房间，并返回是否新增成功 | |
| Hotel. editRoom | 语法 | public boolean editRoom(String roomtype, int roomnum, int roomprice, String hotelname)； | |
| 前置条件 | 酒店工作人员有修改酒店某类型的房间数量和价格的需求 | |
| 后置条件 | 修改酒店房间信息，并返回是否修改成功 | |
| Hotel. getHighestPrice | 语法 | public int getHighestPrice(String hotelname)； | |
| 前置条件 | 需要获取某酒店中最贵的房间价格 | |
| 后置条件 | 返回酒店中最贵的房间价格 | |
| Hotel. removeRoom | 语法 | public boolean removeRoom(String roomtype, String hotelName)； | |
| 前置条件 | 酒店工作人员有删除某酒店的某类型的所有房间的需求 | |
| 后置条件 | 删除对应房间信息，并返回是否删除成功 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| hotelDataService.getHotelInfo(String Hotelname); | | | 根据酒店名返回酒店信息 |
| hotelDataService.modify(HotelVO hotelinfo) | | | 修改酒店信息 |
| hotelDataService. roomModify(String roomtype,int number) | | | 修改酒店房间信息 |
| hotelDataService.addComment(String comment,String username,String hotelname) | | | 增加评论 |

### （4） 业务逻辑层的动态模型

如图4.1.3-2表明了互联网酒店预订系统中，当客户选择查看某个酒店信息后，酒店详情查看业务逻辑处理相关对象之间的协作。

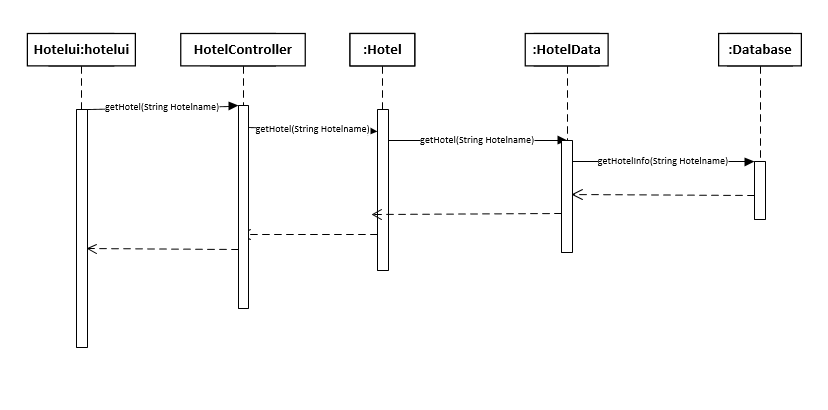


图4.1.3-2 酒店详情查看的顺序图

如图4.1.3-3表明了互联网酒店预订系统中，当酒店工作人员请求更改酒店信息后，酒店信息维护业务逻辑处理相关对象之间的协作。

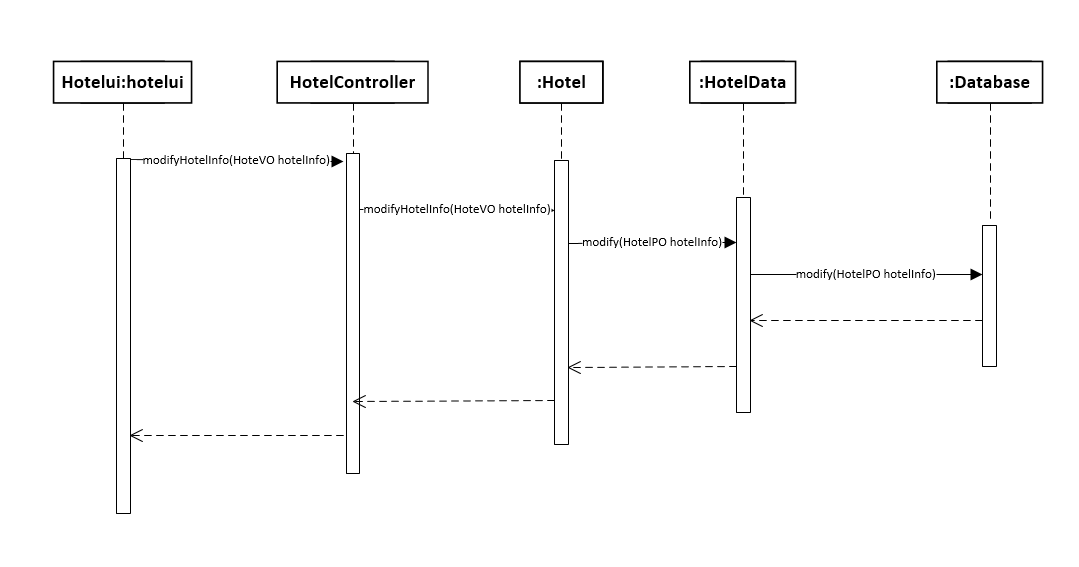


图4.1.3-3 酒店信息维护的顺序图

如图4.1.3-4表明了互联网酒店预订系统中，当酒店工作人员请求修改酒店房间信息后，修改酒店房间信息业务逻辑处理相关对象之间的协作。

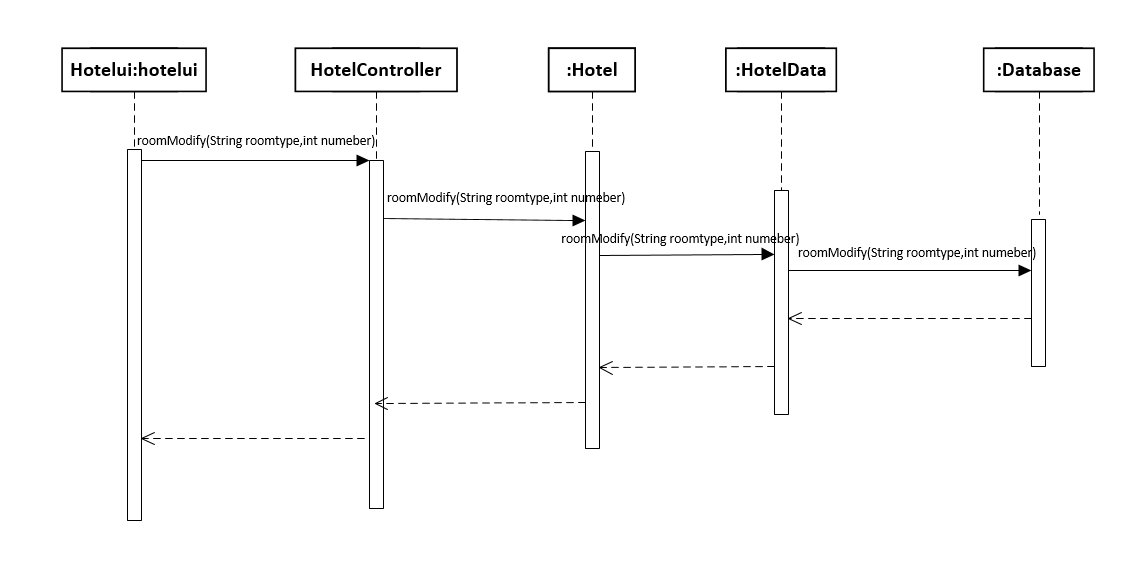


图4.1.3-4酒店房间信息修改的顺序图

如图4.1.3-5表明了互联网酒店预订系统中，当客户请求添加评价后，评价酒店业务逻辑处理相关对象之间的协作。

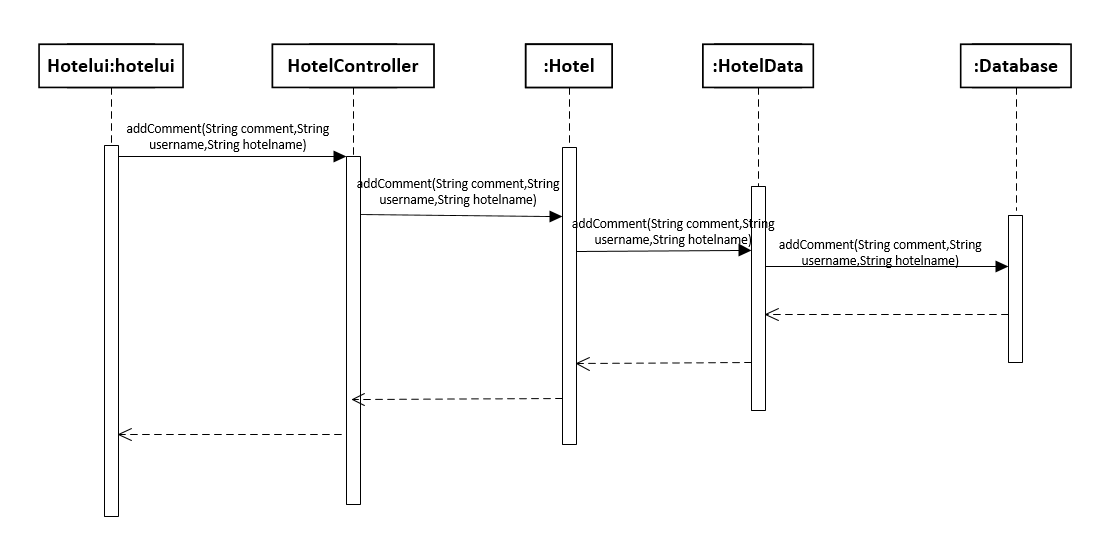


图4.1.3-5评价酒店的顺序图

图4.1.3-6描述了hotel对象在生存时的状态序列，在login后进入logined状态，在roomModify,getHotel,modifyhotelInfo,addComment后进入roomModify,getHotelInfo,modifyHotelInfo,addComent状态，并且在logout后结束生命

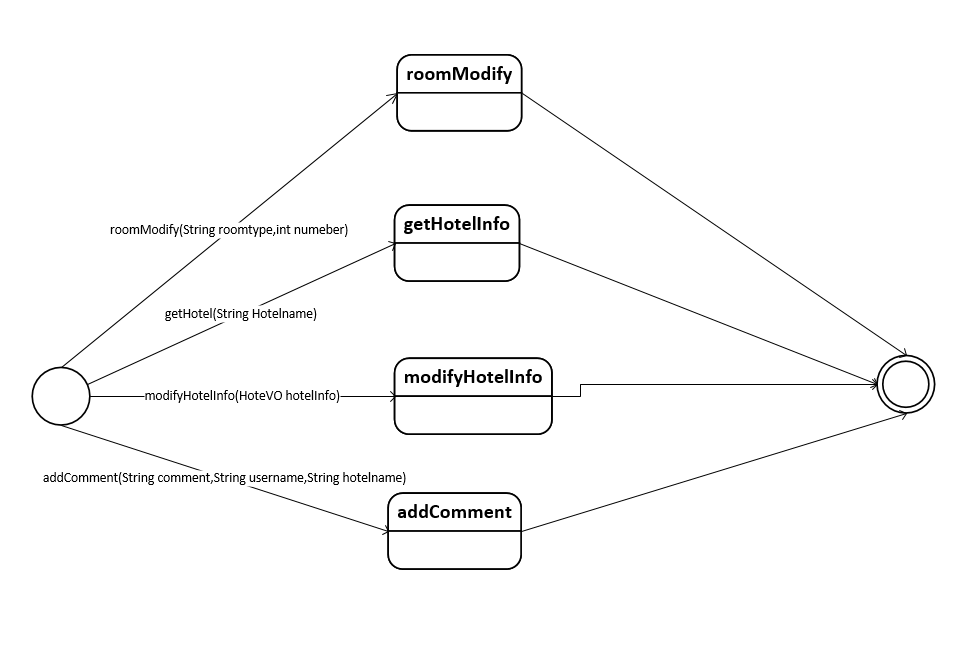


图4.1.3-6 hotel对象的状态图

### 4.1.4 promotionbl模块

（1）模块概述

promotionbl模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

promotionbl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，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加的接口包括promotionblService , promotionDataService 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加PromotionController，这样PromotionController 将会将促销策略相关的业务逻辑职责和逻辑控制委托给promotion对象和priceCalc。

promotionbl模块的设计如图4.1.4-1所示.

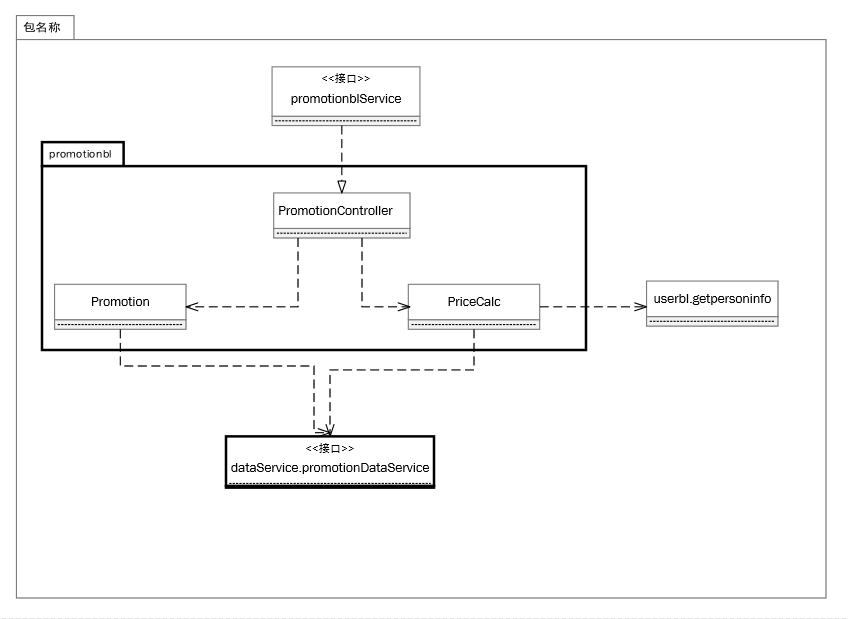


图4.1.4-1promotionbl模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| PromotionController | 负责实现对应于有关促销策略所需的方法 |
| Promotion | 负责处理有关促销策略的功能 |
| PriceCalc | 负责实现有关价格计算的方法 |

表4.1.4-1 promotionbl模块的各个类的职责

（2）模块内部类的接口规范

### 表4.1.4-2 PromotionController类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| PromotionController.priceCut | 语法 | public double priceCut(HotelVO price ,OrderVO numberAndType); | |
| 前置条件 | 已知客户所需房间数量与类型，已知该酒店房间价格 | |
| 后置条件 | 返回考虑优惠政策后的价格 | |
| PromotionController.addProm | 语法 | public boolean  addProm(PromotionVO prom); | |
| 前置条件 | 酒店工作人员或网站营销人员新增新优惠政策 | |
| 后置条件 | 改变存储的优惠政策 | |
| PromotionController.deleteProm | 语法 | public boolean  deleteProm(PromotionVO prom); | |
| 前置条件 | 酒店工作人员或网站营销人员删除优惠政策 | |
| 后置条件 | 删除存储的优惠政策 | |
| PromotionController.modifyProm | 语法 | public boolean  modifyProm(PromotionVO prom); | |
| 前置条件 | 酒店工作人员或网站营销人员修改优惠政策 | |
| 后置条件 | 改变存储的优惠政策 | |
| PromotionController.getProm | 语法 | public boolean ArrayList<PromotionVO>getProm(String hotelnameOrWeb); | |
| 前置条件 | 需要获取当前的优惠政策内容 | |
| 后置条件 | 返回当前优惠政策VO列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| PriceCalc.priceCut(PromotionPO promotion); | | | 计算订单价格 |
| Promotion.addProm（PromotionPO promotion） | | | 增加一个促销策略 |
| Promotion.deleteProm(PromotionPO promotion) | | | 删除一个促销策略 |
| Promotion.modifyProm(PromotionPO promotion) | | | 改变一个促销策略 |
| Promotion.getProm(String hotelname) | | | 获得促销策略 |

表4.1.4-3 **promotion类**的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| Promotion.addProm | 语法 | public boolean  addProm(PromotionVO prom); | |
| 前置条件 | 酒店工作人员或网站营销人员新增新优惠政策 | |
| 后置条件 | 改变存储的优惠政策 | |
| Promotion.deleteProm | 语法 | public boolean  deleteProm(PromotionVO prom); | |
| 前置条件 | 酒店工作人员或网站营销人员删除优惠政策 | |
| 后置条件 | 改变存储的优惠政策 | |
| Promotion.modifyProm | 语法 | public boolean  modifyProm(PromotionVO prom); | |
| 前置条件 | 酒店工作人员或网站营销人员修改优惠政策 | |
| 后置条件 | 改变存储的优惠政策 | |
| Promotion.getProm | 语法 | public boolean ArrayList<PromotionVO>getProm(String hotelnameOrWeb); | |
| 前置条件 | 需要获取当前的优惠政策内容 | |
| 后置条件 | 返回当前优惠政策PO列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| promotionDataService.add(PromotionPO promotion); | | | 增加一个促销策略 |
| promotionDataService.find(String hotelname,int promotiontype) | | | 查找一个促销策略 |
| promotionDataService.delete(PromotionPO promotion) | | | 删除一个促销策略 |
| promotionDataService.modify(PromotionPO promotion) | | | 改变一个促销策略 |
| Person.getPersonInfo (String personName) | | | 获得客户的信息 |

表4.1.4-4 **PriceCalc类**的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| PriceCalc.priceCut | 语法 | public double priceCut(HotelVO hotelvo,OrderVO ordervo); | |
| 前置条件 | 计算优惠政策后的订单价格 | |
| 后置条件 | 返回订单价格 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Promotion.getProm | | | 获得促销策略 |

### （4） 业务逻辑层的动态模型

如图4.1.4-2表明了互联网酒店预订系统中，当酒店工作人员选择增加/删除/修改促销策略后，促销策略更改业务逻辑处理相关对象之间的协作。

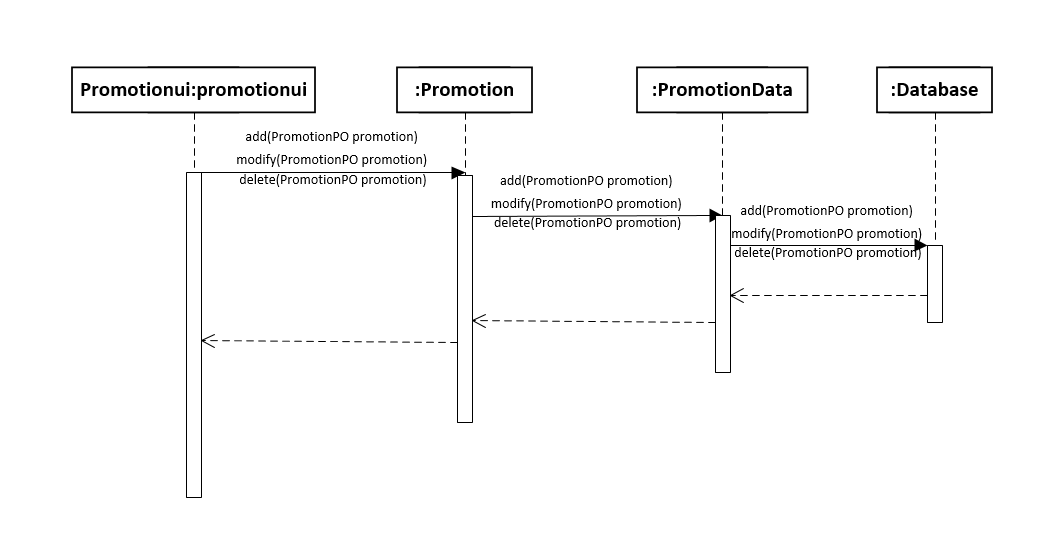


图4.1.4-2 促销策略更改的顺序图

如图4.1.4-2表明了互联网酒店预订系统中，当需要计算订单价格时后，促销策略所负责的业务逻辑处理相关对象之间的协作。

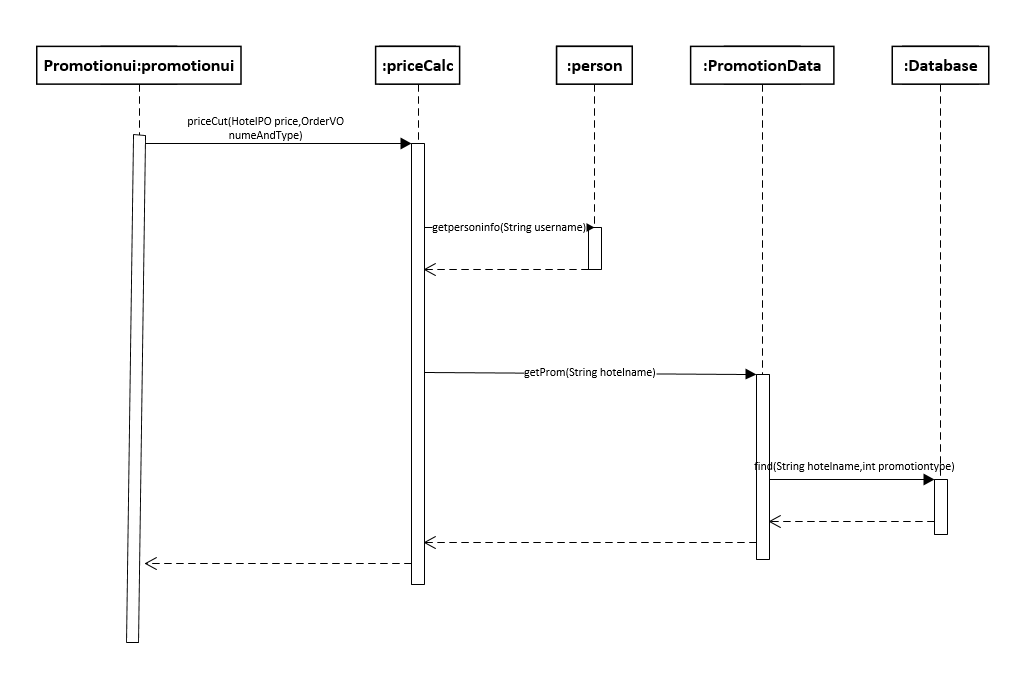


图4.1.4-3 读取促销策略并计算价格的顺序图

## 4.2数据层的分解

业务逻辑层的开发包图参见软件体系结构说明文档图4.1

4.1.1 orderdata模块

（1）模块概述

orderdata模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

orderdata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，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加的接口是orderdataservice 接口。orderPO是做为管理信息的持久化对象被添加到设计模型中的。

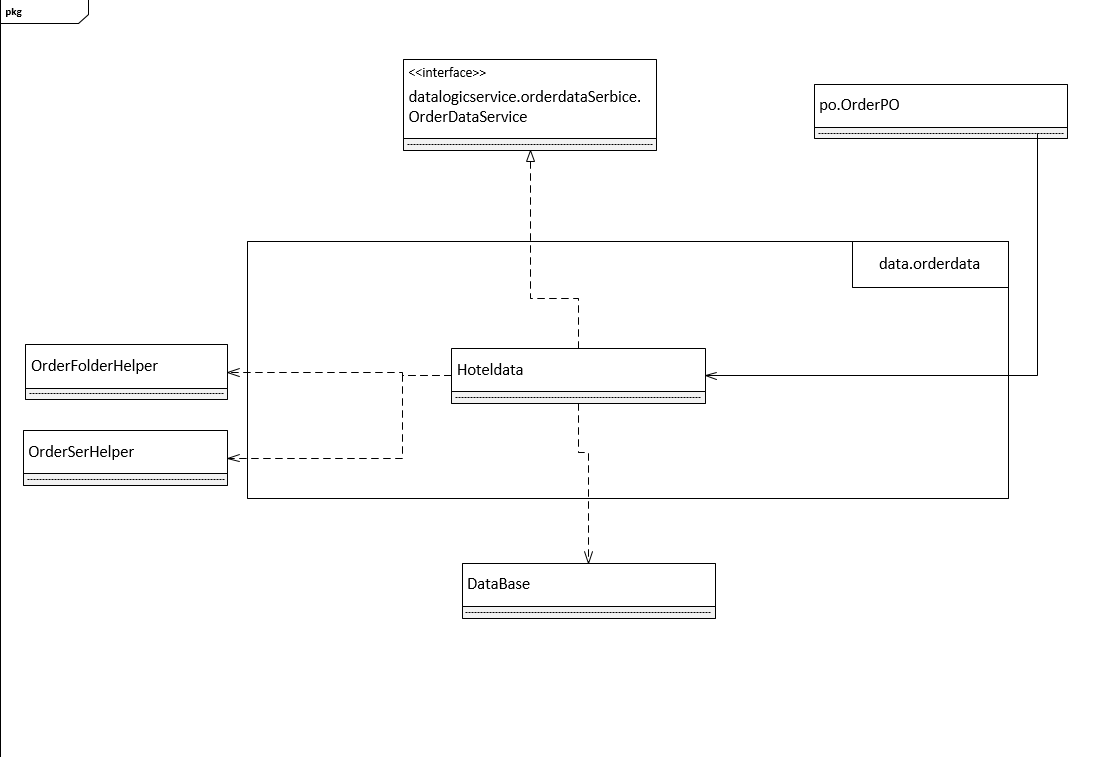
orderdata模块的设计如图4.2.1-1所示. 

图4.2.1-1orderdata模块的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| orderdata | 订单数据层的对象模型，拥有增加一条订单，寻找所有异常订单，寻找指定酒店的所有订单，修改已有订单信息，寻找制定客户的所有订单的方法 |

表4.2.1-1 orderdata模块的各个类的职责

（2）模块内部类的接口规范

### 表4.2.1-2 OrderData类的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderData.add | 语法 | public boolean add(OrderPO order); |
| 前置条件 | 数据库不存在同样ID的订单 |
| 后置条件 | 在数据库中增加一条记录，并且把序列化文件保存到本地 |
| OrderData. exceptionFind | 语法 | public ArrayList<OrderPO> exceptionFind (); |
| 前置条件 | 无 |
| 后置条件 | 无 |
| OrderData. hotelFind | 语法 | public ArrayList<OrderPO> hotelFind (String hotelname); |
| 前置条件 | 存在相同名字的酒店 |
| 后置条件 | 无 |
| OrderData. modify | 语法 | public boolean modify(OrderPO order) |
| 前置条件 | 存在相同的orderID的订单记录 |
| 后置条件 | 修改数据库的记录和相应的ser文件 |
| OrderData. personFind | 语法 | public ArrayList<OrderPO> personFind(String personname) |
| 前置条件 | 存在相同用户名的客户 |
| 后置条件 | 根据客户名称返回酒店订单列表 |
| OrderData. getOrderInfo | 语法 | public OrderPO getOrderInfo(String orderID) |
| 前置条件 | 存在相同ID的订单 |
| 后置条件 | 根据订单ID返回订单 |

4.2.2 userdata模块

（1）模块概述

userdata模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

userdata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

userdata模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。userdata模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层userbl 对象。PersonPO MarketPO ManagerPO HotelWorkerPO是做为管理信息的持久化对象被添加到设计模型中的。

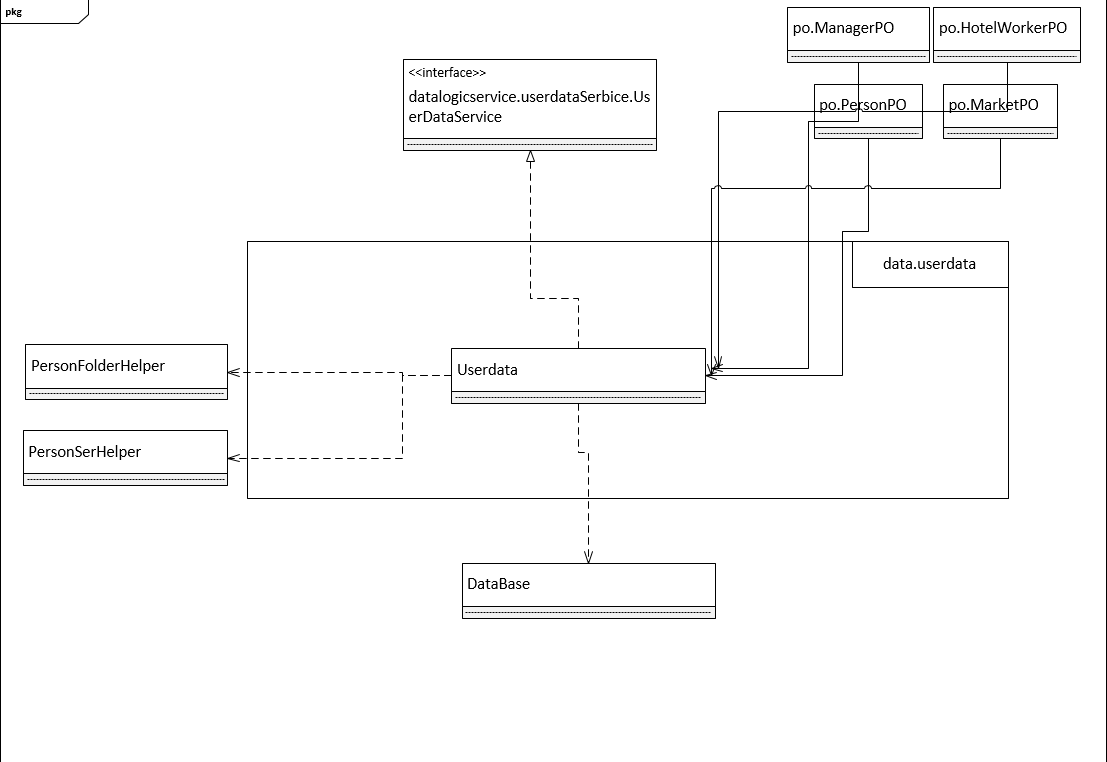
userdata的模块设计见图4.2.2-1

图4.2.2-1userdata的模块

图4.2.2-1user模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| UserData | 订单数据层的对象模型，拥有寻找用户、增加用户、修改用户信息、验证用户登录和增加客户信用记录的方法 |

表4.2.2-1 userbl模块的各个类的职责

（2）模块内部类的接口规范

表4.2.2-2 UserData类的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserData.  addPerson | 语法 | public boolean addPerson(PersonPO personInfo) |
| 前置条件 | 数据库中不存在相同客户名的记录 |
| 后置条件 | 数据库增加一条记录并保存ser文件 |
| UserData. addHotelWorker | 语法 | public boolean  addHotelWorker(HotelWorkerPO hotelworker) |
| 前置条件 | 数据库不存在相同名字的酒店工作人员记录 |
| 后置条件 | 数据库增加一条记录并保存ser文件 |
| UserData. addMarket | 语法 | public boolean addMarket(MarketPO marketInfo) |
| 前置条件 | 数据库不存在相同名字的网站营销人员记录 |
| 后置条件 | 数据库增加一条记录并保存ser文件 |
| UserData. changeCredit | 语法 | public boolean changeCredit(String username, int credit) |
| 前置条件 | 数据库存在相同客户名的记录 |
| 后置条件 | 更新信用值 |
| UserData. findHotelWorker | 语法 | public HotelWorkerPO findHotelWorker(String hotelWorkername) |
| 前置条件 | 无 |
| 后置条件 | 返回指定名字的酒店工作人员信息，不存在则返回null |
| UserData. findMarket | 语法 | public MarketPO findMarket(String marketname) |
| 前置条件 | 无 |
| 后置条件 | 返回指定名字的网站营销人员信息，不存在则返回null |
| UserData. findPerson | 语法 | public PersonPO findPerson(int personID) |
| 前置条件 | 无 |
| 后置条件 | 返回指定ID的客户信息，不存在则返回null |
| UserData. findPerson | 语法 | public PersonPO findPerson(String personname) |
| 前置条件 | 无 |
| 后置条件 | 返回指定名字的客户信息，不存在则返回null |
| UserData. hotelWorkerLogin | 语法 | public boolean hotelWorkerLogin(String hotelWorkername, String password) | |
| 前置条件 | 无 | |
| 后置条件 | 返回登陆是否通过 | |
| UserData. managerLogin | 语法 | public boolean managerLogin(String managername, String password) |
| 前置条件 | 无 |
| 后置条件 | 返回登陆是否通过 |
| UserData. marketLogin | 语法 | public boolean marketLogin(String marketname, String password) |
| 前置条件 | 无 |
| 后置条件 | 返回登陆是否通过 |
| UserData. modifyHotelWorker | 语法 | public boolean modifyHotelWorker(HotelWorkerPO hotelWorkerInfo) |
| 前置条件 | 数据库存在相同名字的酒店工作人员记录 |
| 后置条件 | 修改数据库记录和ser文件 |
| UserData. modifyMarket | 语法 | public boolean modifyMarket(MarketPO marketInfo) |
| 前置条件 | 数据库存在相同名字的网站营销人员记录 |
| 后置条件 | 修改数据库记录和ser文件 |
| UserData. modifyPerson | 语法 | public boolean modifyPerson(PersonPO personInfo) |
| 前置条件 | 数据库存在相同名字的客户记录 |
| 后置条件 | 修改数据库记录和ser文件 |
| UserData. personLogin | 语法 | public boolean personLogin(String personname, String password) |
| 前置条件 | 无 |
| 后置条件 | 返回登陆是否通过 |
| UserData. writeRecord | 语法 | public boolean writeRecord(String personname, RecordPO record) |
| 前置条件 | 数据库存在相同名字的客户记录 |
| 后置条件 | 修改ser文件 |
| UserData. readRecord | 语法 | public boolean readRecord (String personname) |
| 前置条件 | 数据库存在相同名字的客户记录 |
| 后置条件 | 返回相应记录，如果没有返回null |

4.2.3 hoteldata模块

（1）模块概述

hoteldata模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

hoteldata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

hoteldata模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。hoteldata模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层hoteldata对象。HotelPO是做为管理信息的持久化对象被添加到设计模型中的。

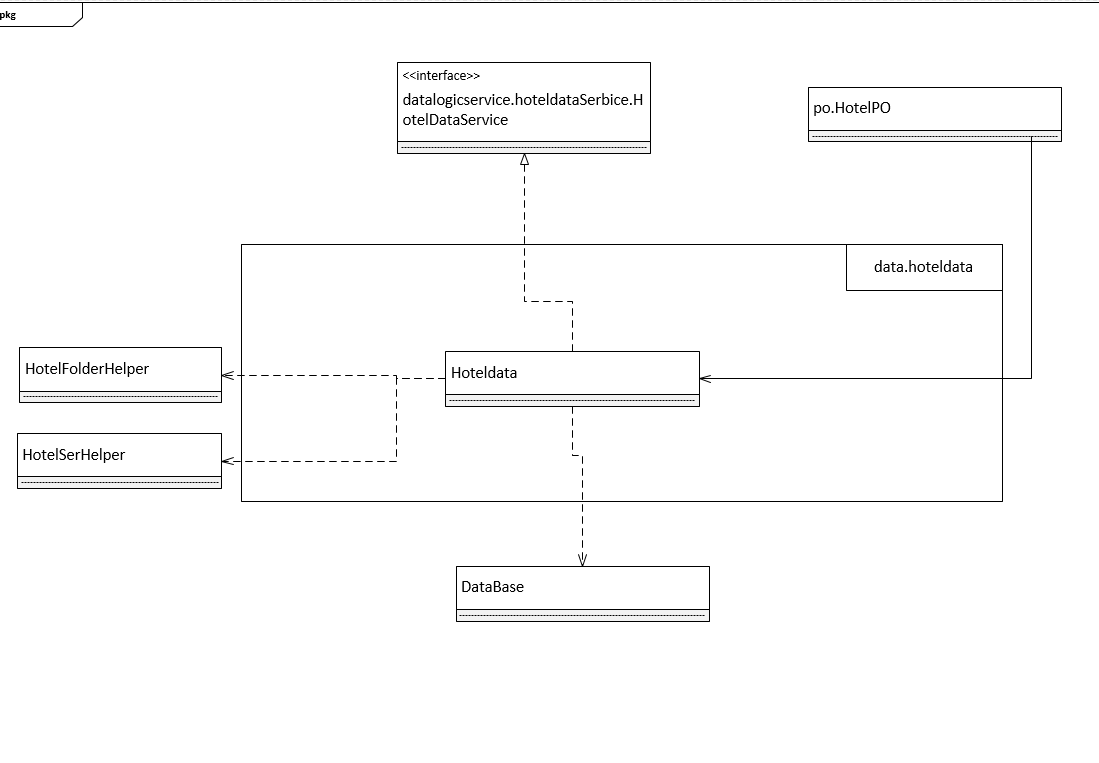
hoteldata的模块设计见图4.2.3-1。

图4.2.3-1 hoteldata的模块设计

|  |  |
| --- | --- |
| 类 | 职责 |
| Hoteldata | 酒店数据层的对象模型，拥有增加一个酒店评论，寻找酒店，增加酒店，修改酒店信息，修改酒店房间信息的方法 |

表4.2.3-1 hoteldata模块的各个类的职责

（2）模块内部类的接口规范

表4.2.3-2 HoteData类的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Hoteldata.  addComment | 语法 | public boolean addComment(CommentPO comment) |
| 前置条件 | 数据库中存在评论的酒店记录 |
| 后置条件 | 保存ser文件 |
| Hoteldata. addHotel | 语法 | public boolean addHotel(HotelPO hotel) |
| 前置条件 | 数据库不存在相同名字的酒店记录 |
| 后置条件 | 数据库增加一条记录并保存ser文件 |
| Hoteldata. findWithReq | 语法 | public ArrayList<HotelPO> findWithReq(HotelPO worstCondition,  HotelPO bestCondition) |
| 前置条件 | 无 |
| 后置条件 | 返回符合条件的酒店记录，不存在就返回null |
| Hoteldata. modify | 语法 | public boolean modify(HotelPO hotelinfo) |
| 前置条件 | 数据库存在相同名字的酒店记录 |
| 后置条件 | 数据库修改记录并保存ser文件 |
| Hoteldata. roomModify | 语法 | public boolean roomModify(String hotelname, ArrayList<RoomPO> rooms) |
| 前置条件 | 数据库存在相同酒店名的记录 |
| 后置条件 | 保存ser文件 |
| Hoteldata. showHotelinfo | 语法 | public HotelPO showHotelinfo(String Hotelname) |
| 前置条件 | 无 |
| 后置条件 | 返回指定名字的酒店信息，不存在则返回null |

4.2.4 promotionbl模块

（1）模块概述

promotionbl模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

promotionbl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，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加的接口promotionDataService。PromotionPO是做为管理信息的持久化对象被添加到设计模型中的。promotiondata模块的设计如图4.2.4-1所示.

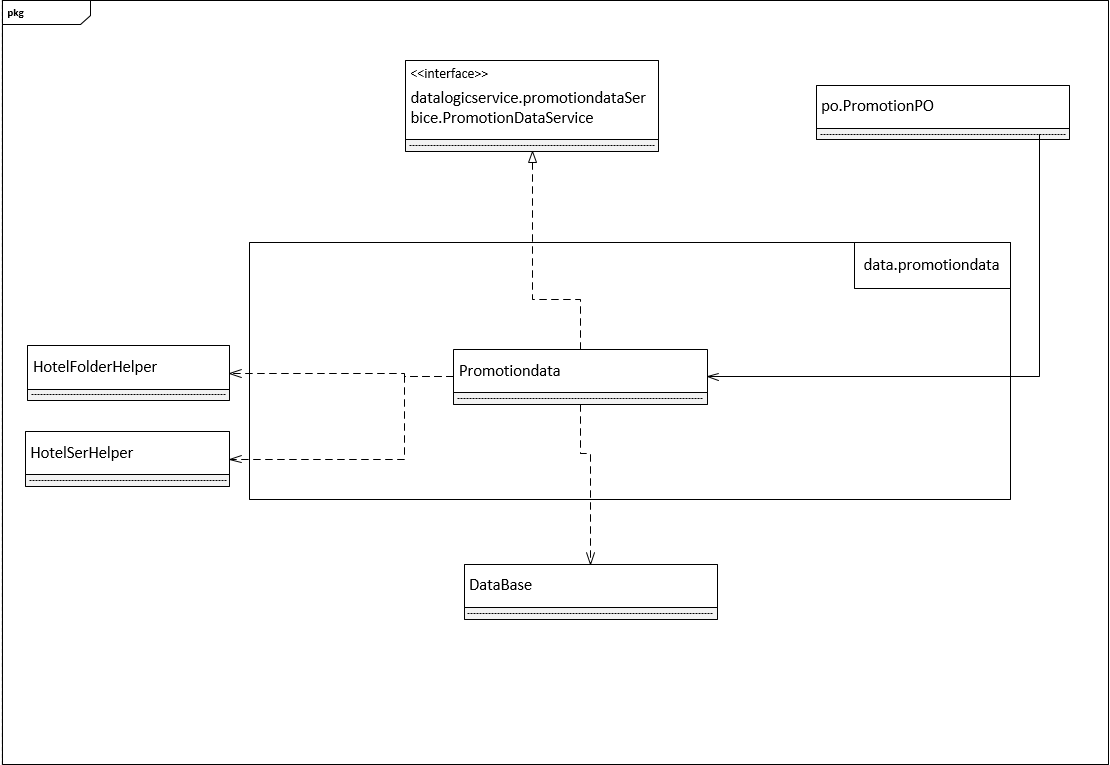


图4.2.4-1promotiondata模块的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| PromotionData | 促销策略数据层的对象模型，拥有增加一个促销策略，寻找酒店，增加酒店，修改酒店信息，修改酒店房间信息的方法 |

表4.2.4-1 promotionbl模块的各个类的职责

（2）模块内部类的接口规范

表4.2.4-2PromotionData类的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionData. add | 语法 | public boolean add(PromotionPO promotion) |
| 前置条件 | 数据库中不存在相同名字的促销策略记录 |
| 后置条件 | 数据库中新增一条记录 |
| PromotionData. delete | 语法 | public boolean delete(PromotionPO promotion) |
| 前置条件 | 数据库中存在相同名字的促销策略记录 |
| 后置条件 | 数据库中删除一条记录 |
| PromotionData. find | 语法 | public ArrayList<PromotionPO> find(String hotelnames) |
| 前置条件 | 数据库中存在有相同酒店名的记录 |
| 后置条件 | 返回所有酒店的促销策略，没有就返回null |
| PromotionData. modify | 语法 | public boolean modify(PromotionPO promotion) |
| 前置条件 | 数据库中存在相同ID的促销策略 |
| 后置条件 | 修改数据库中的记录 |

### 4.2.5 searchdata模块

（1）模块概述

Searchdata模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

Searchdata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，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了searchdataservice接口。为了隔离业务逻辑职责和逻辑控制职责SearchPO是作为管理信息的持久化对象被添加到设计模型中的。

searchdata模块的设计如图4.2.4-1所示.

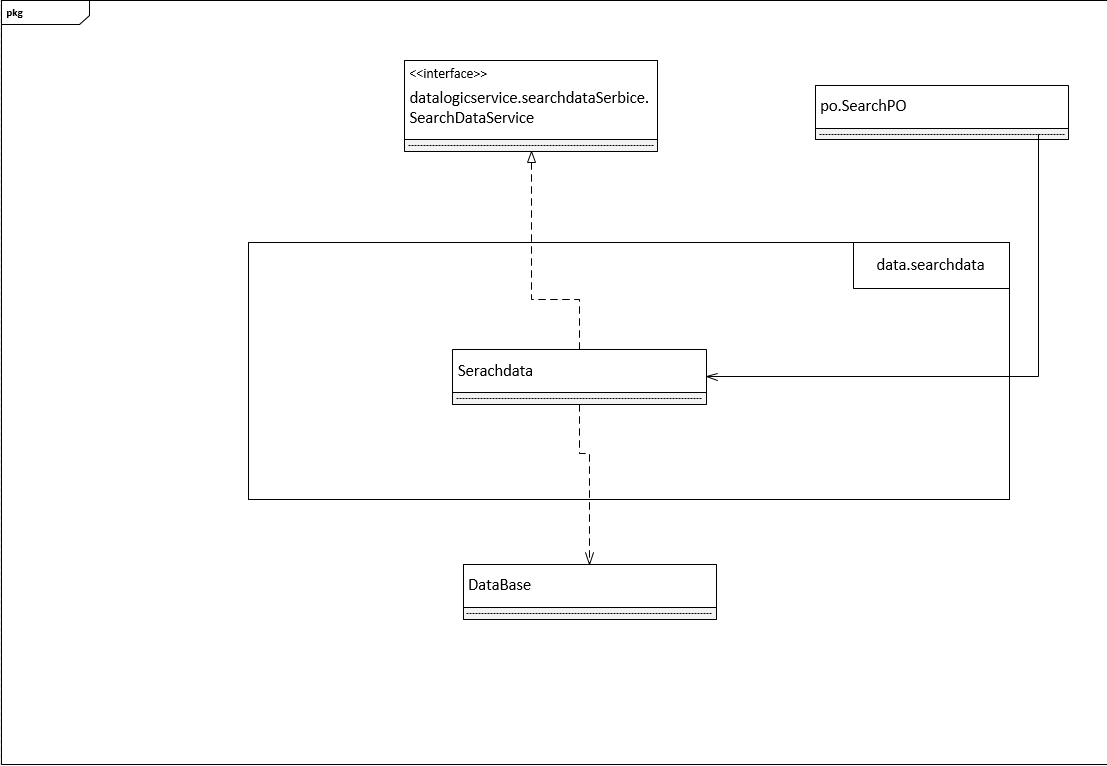


图4.2.5-1searchdata模块的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| SearchData | 订单数据层的对象模型，拥有增加一条搜索记录，删除一条搜索记录，显示指定用户的搜索记录的方法 |

表4.2.5-1 searchdata模块的各个类的职责

（2）模块内部类的接口规范

### 表4.2.5-2 SearchData类的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| SearchData. addSearchHistory | 语法 | public boolean addSearchHistory(SearchPO s) |
| 前置条件 | 数据库中存在搜索记录指定的客户记录 |
| 后置条件 | 数据库中新增一条记录 |
| SearchData. deleteSearchHistory | 语法 | public boolean deleteSearchHistory(SearchPO s) |
| 前置条件 | 数据库中存在搜索记录指定的客户记录 |
| 后置条件 | 数据库中删除一条记录 |
| SearchData. showSearchHistory | 语法 | public ArrayList<SearchPO> showSearchHistory(String username) |
| 前置条件 | 数据库中存在搜索记录指定的客户记录 |
| 后置条件 | 返回此用户的所有记录，如果不存在则返回null |

## 4.3界面展示层的分解

## 界面展示 层的开发包图参见软件体系结构说明文档图4.1

### 4.3.1 personui模块

（1）模块概述

personui模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

personui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

为了隔离界面展示职责和逻辑控制职责，我们添加personuiController，这样personuiController 将会承担客户相关的界面跳转并且调用逻辑层userbl、hotelbl和orderbl对象的逻辑方法。将personVO是做为管理传递和暂时持有的数据对象被添加到设计模型中的。

personui模块的设计如图4.3.1-1所示.

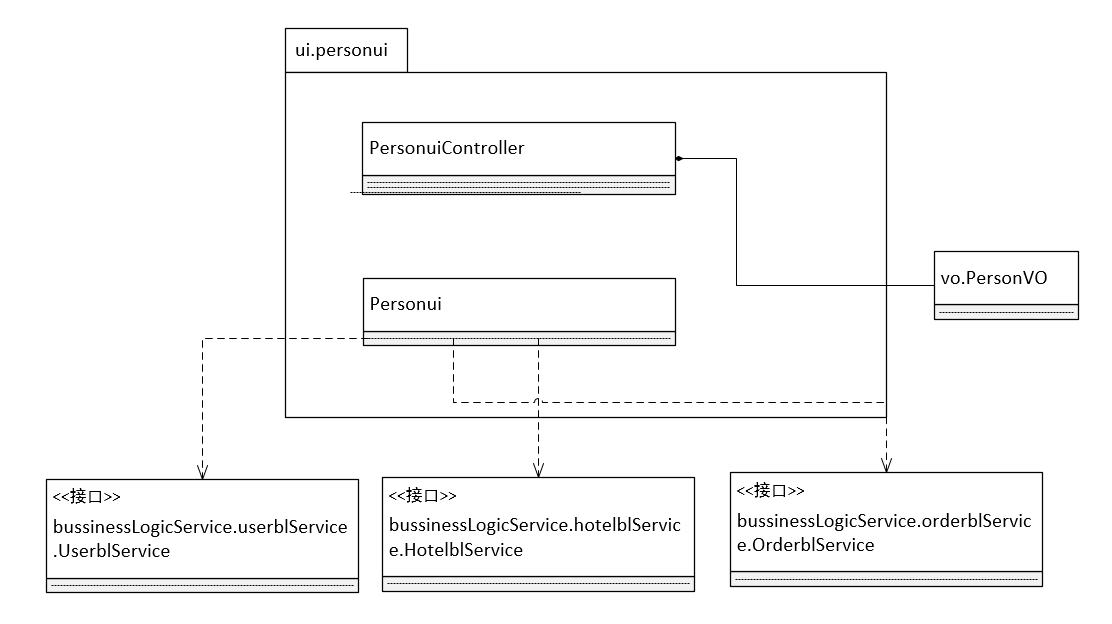


图4.3.1-1personui模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| PersonuiController | 负责实现调用逻辑层与客户有关的逻辑，并且实现界面间的跳转 |
| Personui | 负责客户的界面显示 |

表4.3.1-1 personui模块的各个类的职责

（2）模块内部类的接口规范

### 表4.3.1-2 PersonuiController类的接口规范

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| 服务名 | 服务 |
| OrderController.handleAbnormalOrder(OrderVO order，int percentOfCredit) | 处理一个异常订单 |
| OrderController.reverseOrder(OrderVO order) | 撤销一个未执行订单 |
| OrderController. finishOrder(OrderVO order) | 执行一个异常或未执行订单 |
| OrderController.createOrder(OrderVO order) | 生成并保存一个订单 |
| OrderController.personOrders(String username) | 查看个人订单 |
| OrderController.hotelOrders(String hotelname) | 浏览酒店订单 |
| OrderController.netOrders() | 浏览网站订单 |
| OrderController.personStateOrders(String personname, String state) | 浏览客户某状态的订单 |
| OrderController.hotelStateOrders(String hotelname, String state) | 浏览酒店某状态的订单 |
| OrderController.netNumOrders(String date) | 浏览网站某天的订单 |
| OrderController. getOrderInfo(String orderID) | 得到相应ID订单信息 |
| OrderController. changeOrderState(String orderID, String newState) | 改变某订单状态 |
| UserController.register(PersonVO userinfo) | 客户注册 |
| UserController.personSave(PersonVO personInfo) | 客户信息修改，信用不能修改，只能查看 |
| UserController.marketSave(MarketVO marketInfo) | 保存网站营销人员信息 |
| UserController.hotelWorkerSave(HotelWorkerVO hotelWorkerInfo) | 保存酒店工作人员信息 |
| UserController.changeCredit(String personname, RecordVO recordvo) | 修改信用值 |
| UserController.registeMember(PersonVO personInfo, String vipType, String vipInfo) | 会员注册 |
| UserController.getPersonInfo(String personname) | 根据客户名称获取客户信息 |
| UserController.getPersonInfo(int personID) | 根据客户ID获取客户信息 |
| UserController.getHotelWorkerInfo(String hotelWorkername) | 获取酒店工作人员信息 |
| UserController.userLogin(String username, String password, String usertype) | 用户登录 |
| UserController.marketAdd(MarketVO marketInfo) | 增加网站营销人员 |
| UserController.hotelWorkerAdd(HotelWorkerVO hotelworkerInfo) | 增加酒店工作人员 |
| UserController.getMarketInfo(String marketname) | 获取网站营销人员信息 |
| UserController.isExist(String username, String usertype) | 判断某用户是否存在 |
| UserController.writeRecord(String personname, RecordVO record) | 写入客户信用记录 |
| UserController.getRecord(String personname) | 获取客户信用记录 |
| HotelController.getHotelInfoByPerson(String Hotelname) | 客户根据酒店名称，获取酒店信息 |
| HotelController.addComment(CommentVO commentvo) | 写入客户评论 |
| HotelController.findWithReq(HotelConditionVO worstCondition, HotelConditionVO bestCondition) | 根据客户给出要求的房间条件区间，筛选符合条件的酒店，返回酒店列表 |
| HotelController.findReqRoom(String hotelname, String roomtype, String starttime, String endtime) | 返回符合条件的对应酒店的房间 |
| HotelController.getAvailableNumber(String hotelname, String roomtype, String starttime, String endtime) | 返回符合条件的对应酒店的可用房间数量 |
| HotelController.getHotelRoomInfo(String hotelname) | 根据酒店名称返回对应酒店的房间类型和价格的列表 |

### 4.3.2 hotelworkerui模块

（1）模块概述

hotelworkerui模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

hotelworkerui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

为了隔离界面展示职责和逻辑控制职责，我们添加hotelworkeruiController，这样hotelworkeruiController 将会承担客户相关的界面跳转并且调用逻辑层userbl、hotelbl和orderbl对象的逻辑方法。将hotelworkerVO是做为管理传递和暂时持有的数据对象被添加到设计模型中的。

hotelworkerui模块的设计如图4.3.2-1所示.

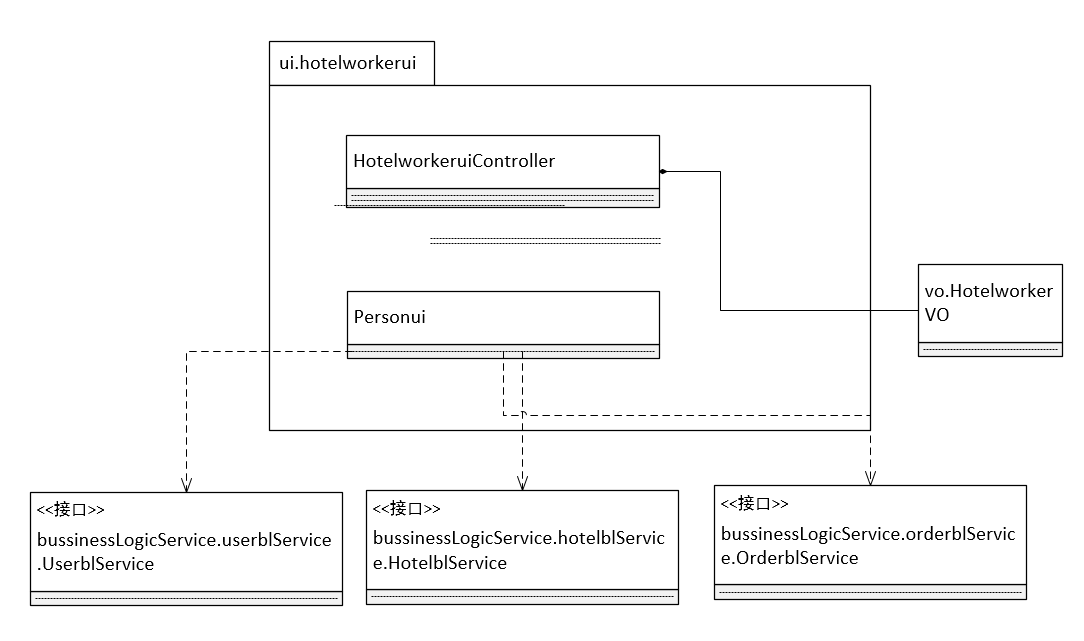


图4.3.2-1hotelworkerui模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| HotelworkeruiController | 负责实现调用逻辑层与客户有关的逻辑，并且实现界面间的跳转 |
| Hotelworkerui | 负责客户的界面显示 |

表4.3.2-1 hotelworkerui模块的各个类的职责

（2）模块内部类的接口规范

### 表4.3.2-2 HotelworkeruiController类的接口规范

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| 服务名 | 服务 |
| OrderController.handleAbnormalOrder(OrderVO order，int percentOfCredit) | 处理一个异常订单 |
| OrderController. finishOrder(OrderVO order) | 执行一个异常或未执行订单 |
| OrderController.hotelOrders(String hotelname) | 浏览酒店订单 |
| OrderController.hotelStateOrders(String hotelname, String state) | 浏览酒店某状态的订单 |
| OrderController. getOrderInfo(String orderID) | 得到相应ID订单信息 |
| OrderController. changeOrderState(String orderID, String newState) | 改变某订单状态 |
| UserController.register(PersonVO userinfo) | 客户注册 |
| UserController.userLogin(String username, String password, String usertype) | 用户登录 |
| UserController.hotelWorkerAdd(HotelWorkerVO hotelworkerInfo) | 增加酒店工作人员 |
| UserController.getMarketInfo(String marketname) | 获取网站营销人员信息 |
| UserController.isExist(String username, String usertype) | 判断某用户是否存在 |
| HotelController.getHotelInfoByHotelworkerOrManager(String Hotelname) | 酒店工作人员或网站管理人员，根据酒店名称，获取酒店信息 |
| HotelController.modifyHotelInfo(HotelVO hotelinfo) | 修改酒店信息 |
| HotelController.roomModify(String hotelname, ArrayList<RoomVO> roomvoList) | 房间内容修改 |
| HotelController.addHotel(HotelVO hotelvo) | 增加酒店 |
| HotelController.findReqRoom(String hotelname, String roomtype, String starttime, String endtime) | 返回符合条件的对应酒店的房间 |
| HotelController.getAvailableNumber(String hotelname, String roomtype, String starttime, String endtime) | 返回符合条件的对应酒店的可用房间数量 |
| HotelController.getHotelRoomInfo(String hotelname) | 根据酒店名称返回对应酒店的房间类型和价格的列表 |
| HotelController.addRoom(String roomtype, int roomnum, int roomprice, String hotelname) | 新增某酒店的某类型房间 |

### 4.3.3marketui模块

（1）模块概述

marketui模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

marketui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

为了隔离界面展示职责和逻辑控制职责，我们添加marketuiController，这样marketuiController 将会承担客户相关的界面跳转并且调用逻辑层userbl和orderbl对象的逻辑方法。将marketVO是做为管理传递和暂时持有的数据对象被添加到设计模型中的。

marketui模块的设计如图4.3.3-1所示.

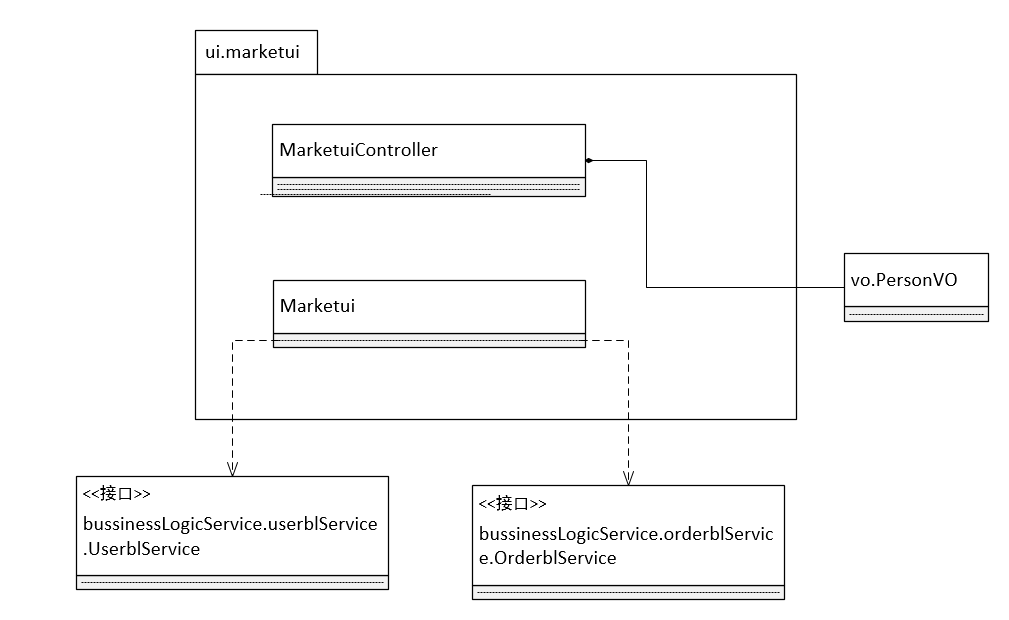


图4.3.3-1hotelworkerui模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| HotelworkeruiController | 负责实现调用逻辑层与客户有关的逻辑，并且实现界面间的跳转 |
| Hotelworkerui | 负责客户的界面显示 |

表4.3.3-1 hotelworkerui模块的各个类的职责

（2）模块内部类的接口规范

### 表4.3.3-2 HotelworkeruiController类的接口规范

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| 服务名 | 服务 |
| OrderController.handleAbnormalOrder(OrderVO order，int percentOfCredit) | 处理一个异常订单 |
| OrderController.netOrders() | 获得网站所有订单的列表 |
| OrderController. getOrderInfo(String orderID) | 得到相应ID订单信息 |
| OrderController. changeOrderState(String orderID, String newState) | 改变某订单状态 |
| UserController.register(PersonVO userinfo) | 客户注册 |
| UserController.userLogin(String username, String password, String usertype) | 用户登录 |
| UserController.isExist(String username, String usertype) | 判断某用户是否存在 |
| UserController.changeCredit(String personname, RecordVO recordvo) | 修改信用值 |

### 4.3.4 managerui模块

（1）模块概述

managerui模块承担的需求参见需求规格说明文档功能需求及相关功能需求。

managerui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

为了隔离界面展示职责和逻辑控制职责，我们添加manageruiController，这样manageruiController 将会承担客户相关的界面跳转并且调用逻辑层userbl对象的逻辑方法。MarketVO、HotelworkerVO、PersonVO是做为管理传递和暂时持有的数据对象被添加到设计模型中的。

managerui模块的设计如图4.3.2-1所示.

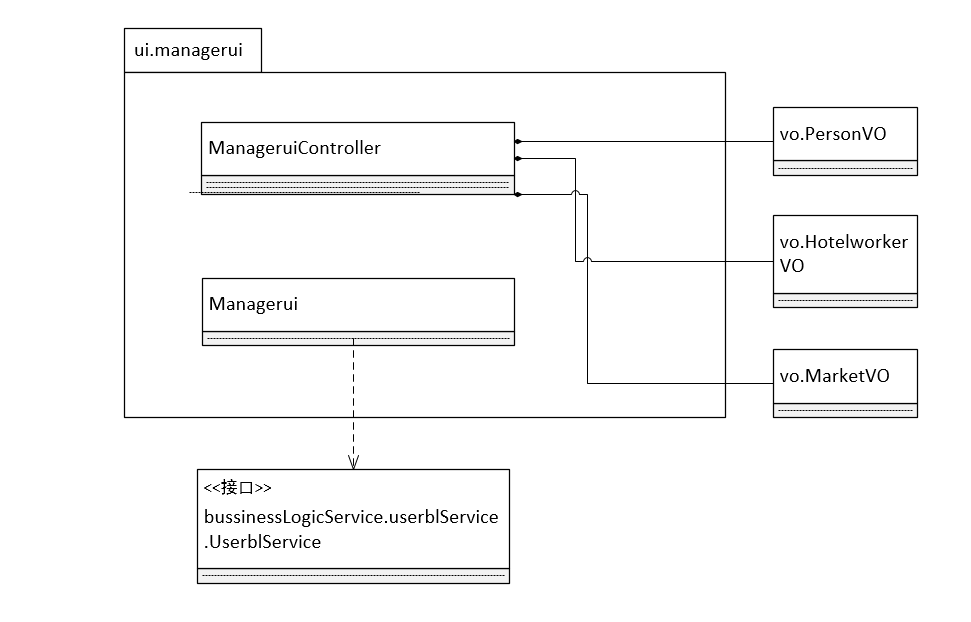
s

图4.3.4-1hotelworkerui模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| HotelworkeruiController | 负责实现调用逻辑层与客户有关的逻辑，并且实现界面间的跳转 |
| Hotelworkerui | 负责客户的界面显示 |

表4.3.4-1 hotelworkerui模块的各个类的职责

（2）模块内部类的接口规范

### 表4.3.4-2 HotelworkeruiController类的接口规范

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| 服务名 | 服务 |
| UserController.userLogin(String username, String password, String usertype) | 用户登录 |
| UserController.hotelWorkerAdd(HotelWorkerVO hotelworkerInfo) | 增加酒店工作人员 |
| UserController.getMarketInfo(String marketname) | 获取网站营销人员信息 |
| UserController.isExist(String username, String usertype) | 判断某用户是否存在 |
| HotelController.addHotel(HotelVO hotelvo) | 增加酒店 |
| UserController.marketAdd(MarketVO marketInfo) | 增加网站营销人员 |
| UserController.getHotelworkerInfo(String hotelworkerName) | 获取酒店工作人员信息 |
| UserController.getPersonInfo(String personname) | 获取客户信息 |
| UserController.personSave(PersonVO personInfo) | 客户信息修改，信用不能修改，只能查看 |
| UserController.marketSave(MarketVO marketInfo) | 保存网站营销人员信息 |
| UserController.hotelWorkerSave(HotelWorkerVO hotelWorkerInfo) | 保存酒店工作人员信息 |

# 依赖视角

图5-1和图5-2是客户端和服务端各自的包之间的依赖关系。

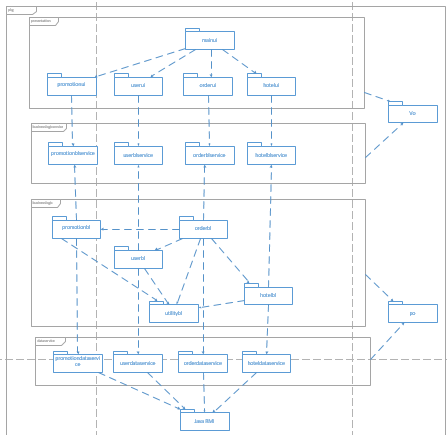
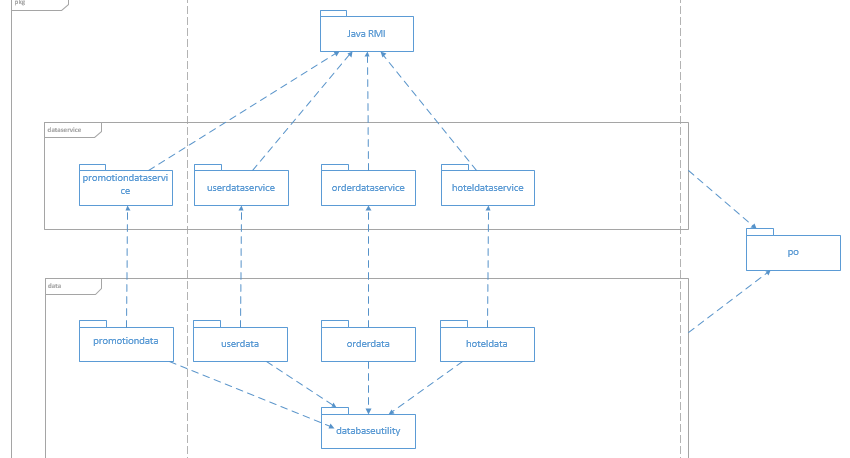


图5-1客户端包图

图5-2服务器端包图

# 耦合和内聚度量

1. Orderbl模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类名 | CBO | DAC | NOC | DIT |
| Order | 3 | 3 | 0 | 0 |
| 调用其他类2，被其他类调用1 | Member，  PriceCalc，  OrderPO |
| OrderController | 11 | 1 | 0 | 0 |
| 调用其他类7，被其他类调用4 | Order |

1. Userbl模块

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类名 | CBO | DAC | NOC | DIT |
| UserController | 12 | 4 | 0 | 0 |
| 调用其他类12 | Person,  Market,  Manager,  HotelWorker |
| 类名 | CBO | DAC | NOC | DIT |
| Manager | 8 | 4 | 0 | 0 |
| 被其他类调用8 | PersonPO  MarketPO  HotelWorkerPO  UserData |
| 类名 | CBO | DAC | NOC | DIT |
| Person | 6 | 2 | 0 | 0 |
| 被其他类调用6 | UserData  PersonPO |
| 类名 | CBO | DAC | NOC | DIT |
| HotelWorker | 3 | 2 | 0 | 0 |
| 被其他类调用3 | UserData  HotelWorkerPO |
| 类名 | CBO | DAC | NOC | DIT |
| Market | 4 | 2 | 0 | 0 |
| 被其他类调用4 | UserData  MarketPO |