**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10月29日

目录

[更新历史 3](#_Toc466237670)

[1. 引言 4](#_Toc466237671)

[1.1编制目的 4](#_Toc466237672)

[1.2词汇表 5](#_Toc466237673)

[1.3参考资料 5](#_Toc466237674)

[2. 产品概述 5](#_Toc466237675)

[3. 体系结构设计概述 6](#_Toc466237676)

[4. 结构视角 6](#_Toc466237677)

[4.1 业务逻辑层的分解 6](#_Toc466237678)

[4.1.1 orderbl模块 6](#_Toc466237680)

[4.1.2 userbl模块 16](#_Toc466237683)

[4.1.3 hotelbl模块 36](#_Toc466237686)

[4.1.4 promotionbl模块 43](#_Toc466237689)

[4.1.5 searchbl模块 49](#_Toc466237692)

[5. 依赖视角 58](#_Toc466237695)

# 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 |

# 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.

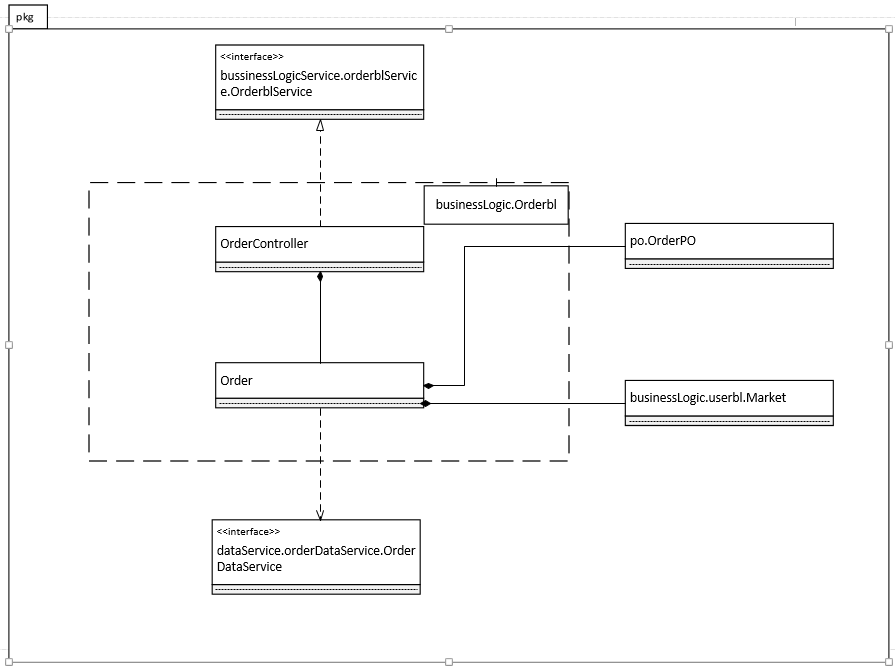


图4.1.1-1order模块类的设计

|  |  |
| --- | --- |
| 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.handleAbnormalOrder | 语法 | public boolean handleAbnormalOrder(OrderVO order); | |
| 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 | |
| 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 | 提交订单，持久化保存订单信息 | |
| OrderController.personOrders | 语法 | Public ArrayList <OrderVO> personOrders(String username); | |
| 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 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Order.handleAbnormalOrder(OrderVO order) | | | 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 |

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); | |
| 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 username); | |
| 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 | |
| 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 | |
| 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 |
| PriceCalc.priceCut(hotelPO price ,orderVO number,orderVO type) | | | 返回折后价格 |

### （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。

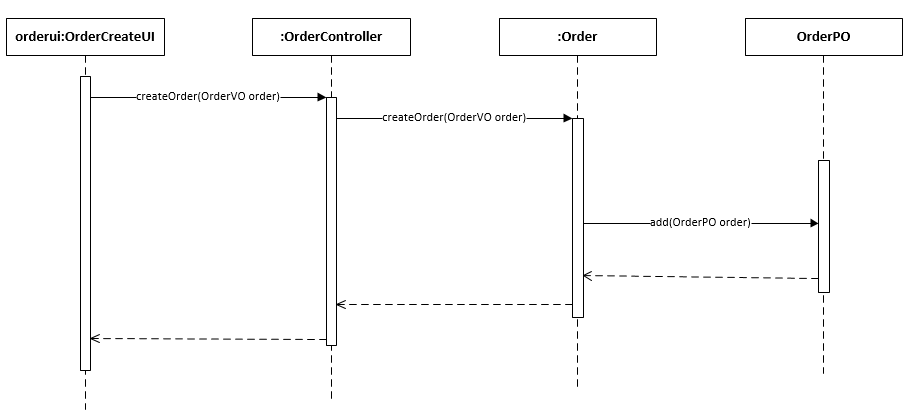


图4.1.1-2 订单生成的顺序图

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。

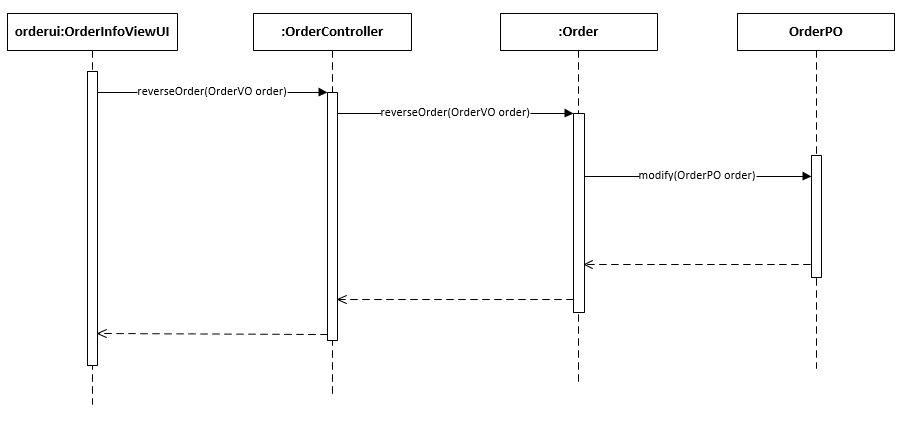


图4.1.1-3 订单撤销的顺序图

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。

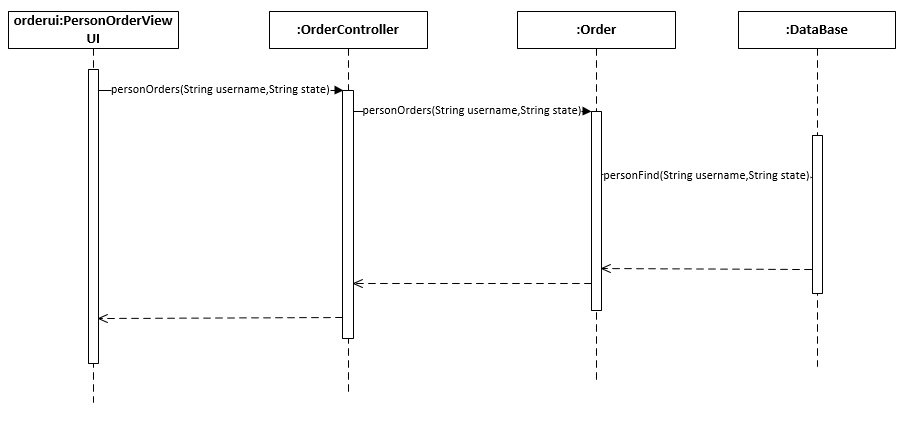


图4.1.1-4 个人订单查看的顺序图

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。

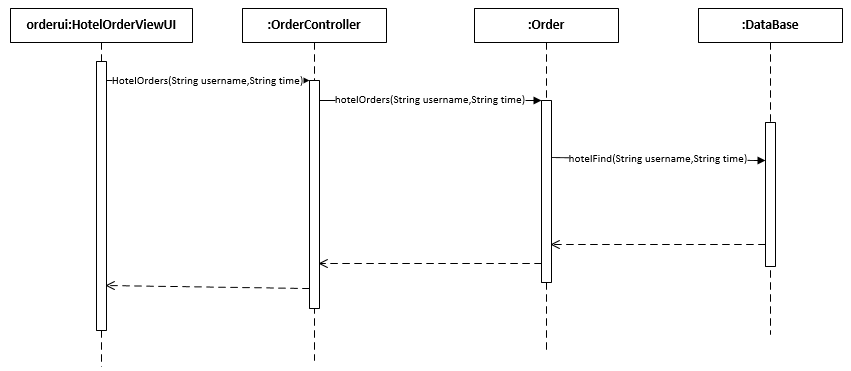


图4.1.1-5 酒店订单浏览的顺序图

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。

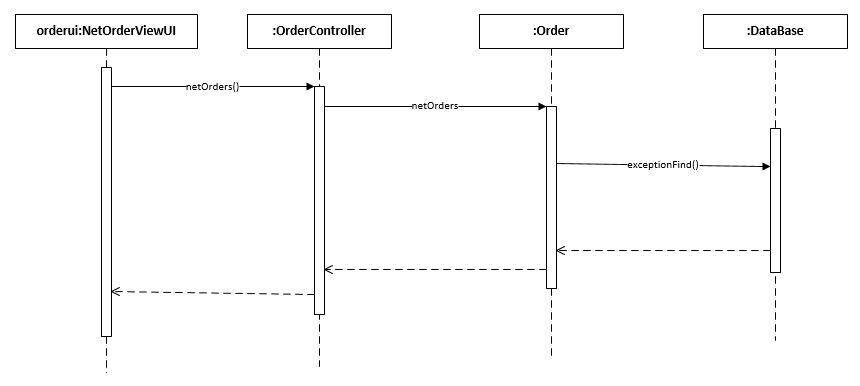


图4.1.1-6 网站订单浏览的顺序图

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。

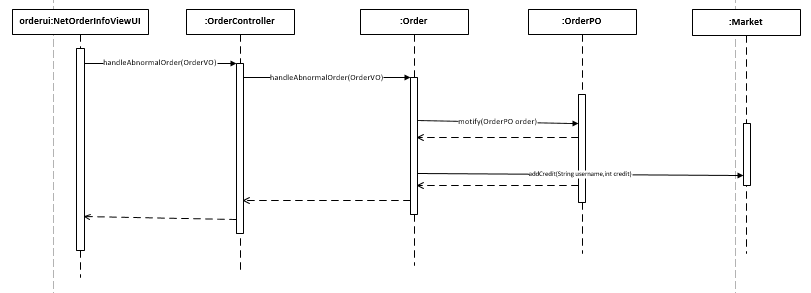


图4.1.1-7 异常订单处理的顺序图

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。

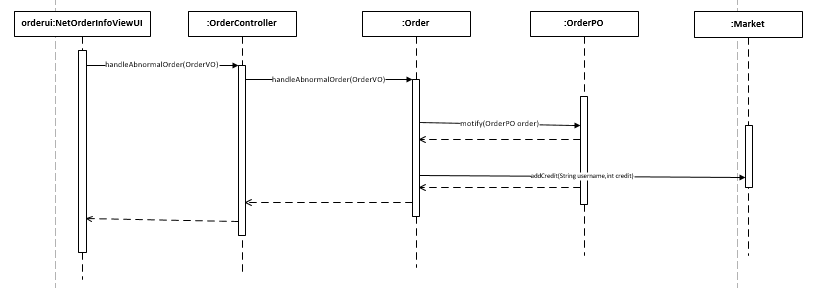


图4.1.1-8 订单执行的顺序图

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。

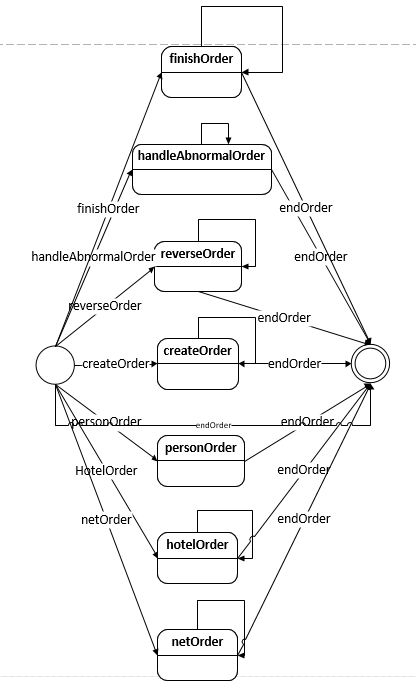


图4.1.1-9 Order对象的状态图

### 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。

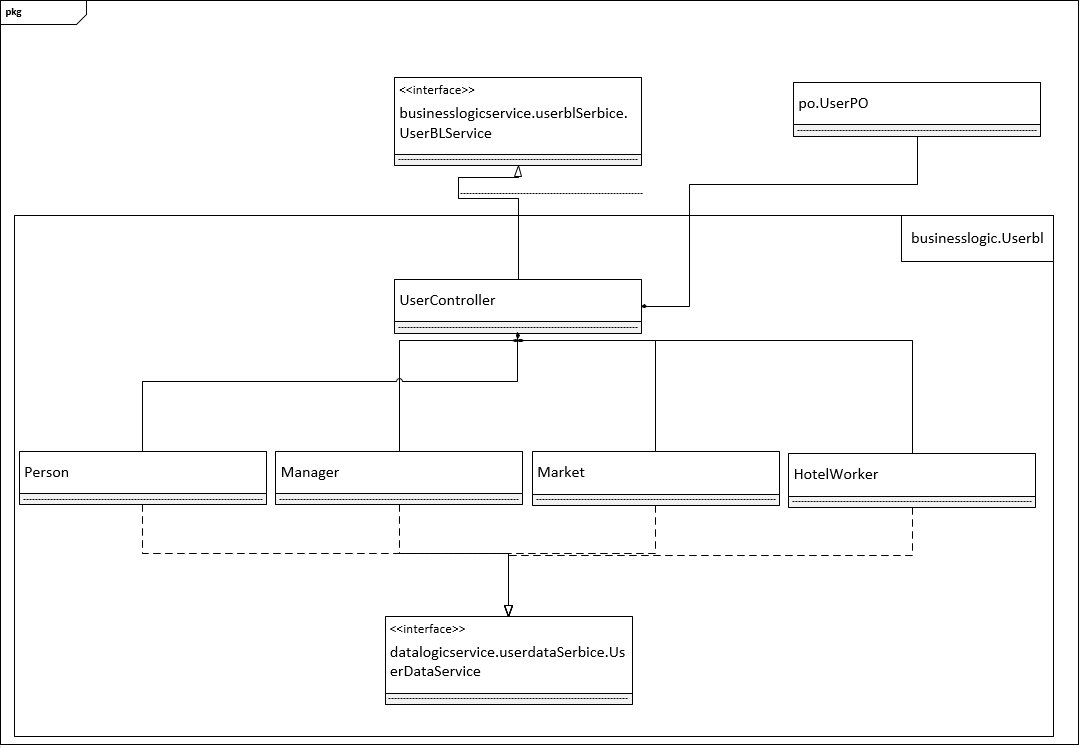


图4.1.2-1user模块类的设计

|  |  |
| --- | --- |
| 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.addCredit | 语法 | public boolean addCredit (String username,int credit) |
| 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 registerMember(PersonVO personinfo); |
| 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 PersonPO getPersonInfo(String username); |
| 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.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(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 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 |
| 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 |
| 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 |
| 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 |
| 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. 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 | |
| Person.registerMember | 语法 | public boolean registerMember(PersonVO personinfo); | |
| 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.personLogin | 语法 | public boolean Person.login(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 | |
| 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 |

表4.1.2-4 Market类的的接口规范

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 提供的服务（供接口） | | | | | |
| Market.getPersonInfo | | 语法 | public PersonPO getPersonInfo(String username); | |
| 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 MarketPO getMarketInfo(String username); | | |
| 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.addCredit | | 语法 | public boolean addCredit (String personname,int credit) | |
| 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.marketLogin | 语法 | | public boolean Market.login(String marketname,String password) | | |
| 前置条件 | | 网站营销人员已经输入了网站营销人员名和密码 | | |
| 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.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.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

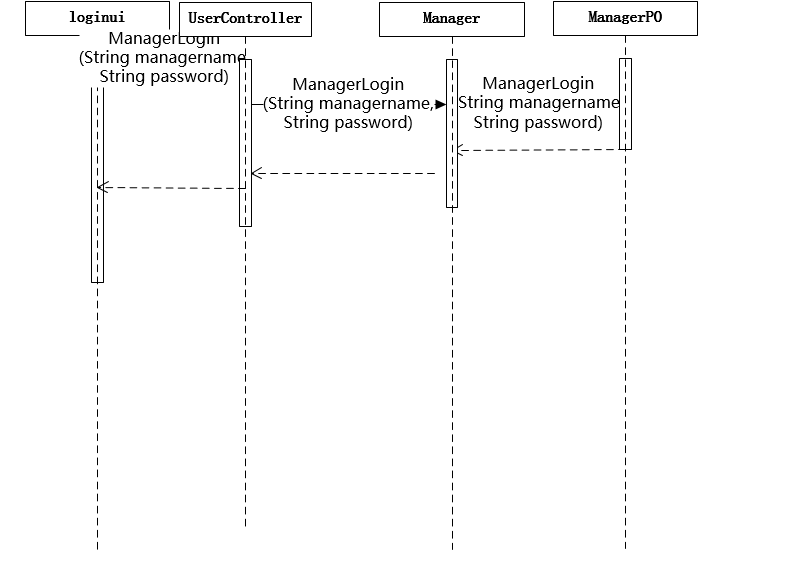


图4.1.2-2网站管理人员登陆顺序图

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

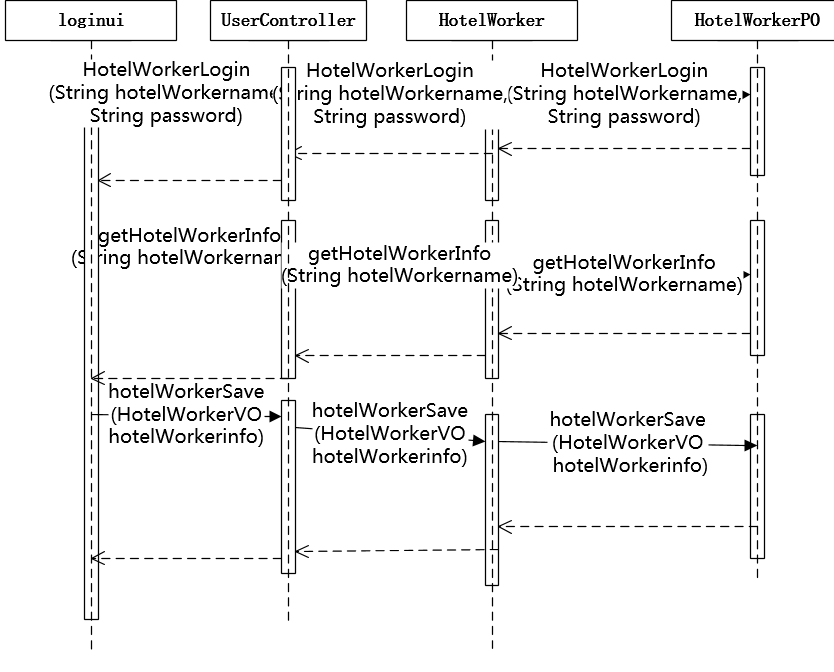


图4.1.2-3酒店工作人员登陆顺序图

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。

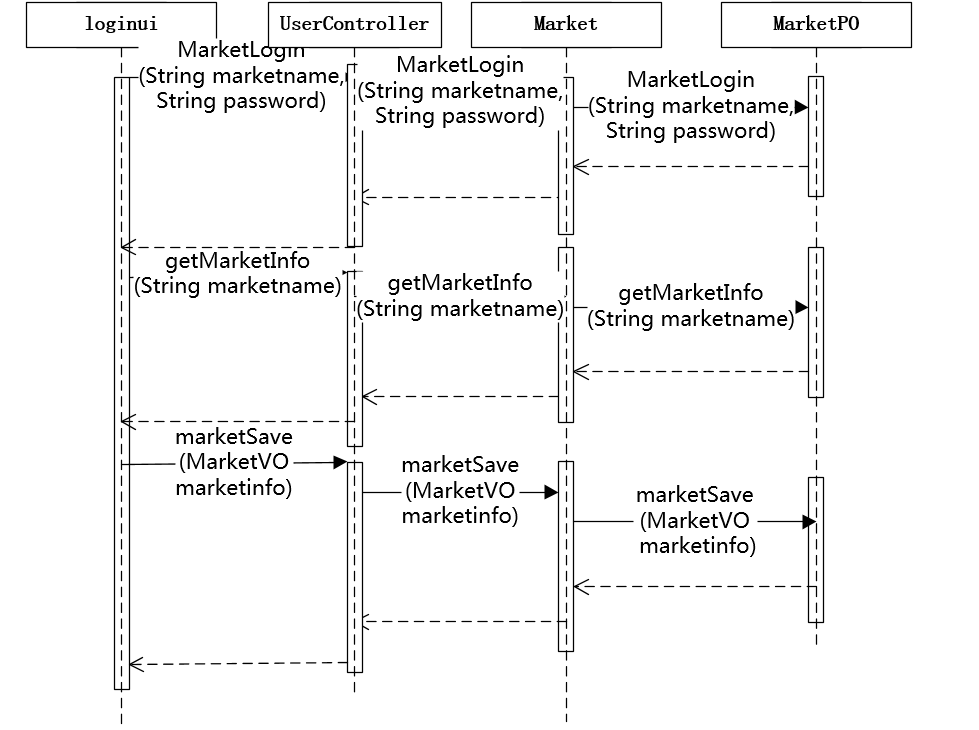


图4.1.2-4网站营销人员登陆、查看顺序图

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

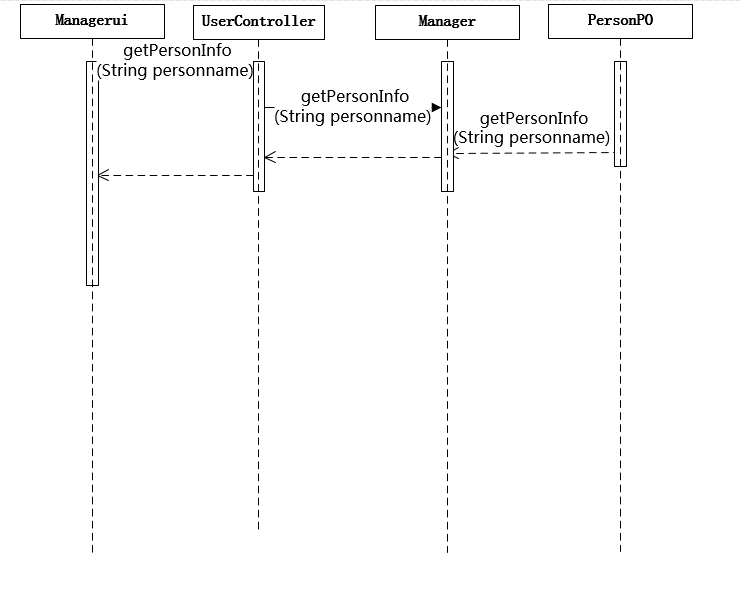


图4.1.2-5网站管理人员查看客户信息顺序图

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

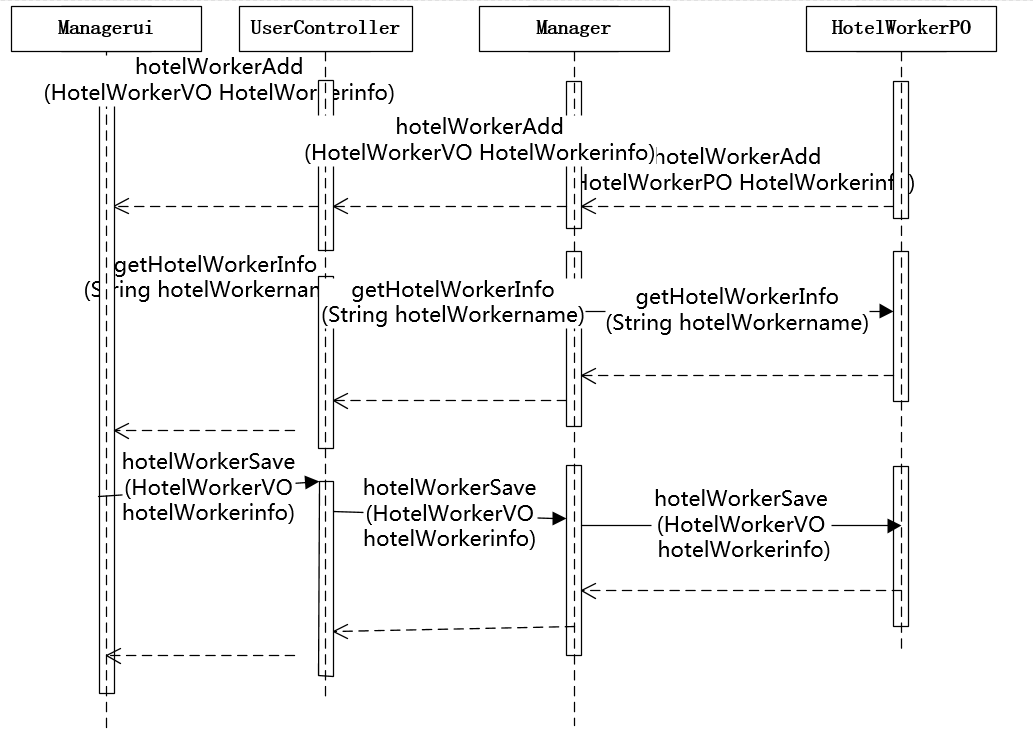


图4.1.2-6网站管理人员管理酒店工作人员顺序图

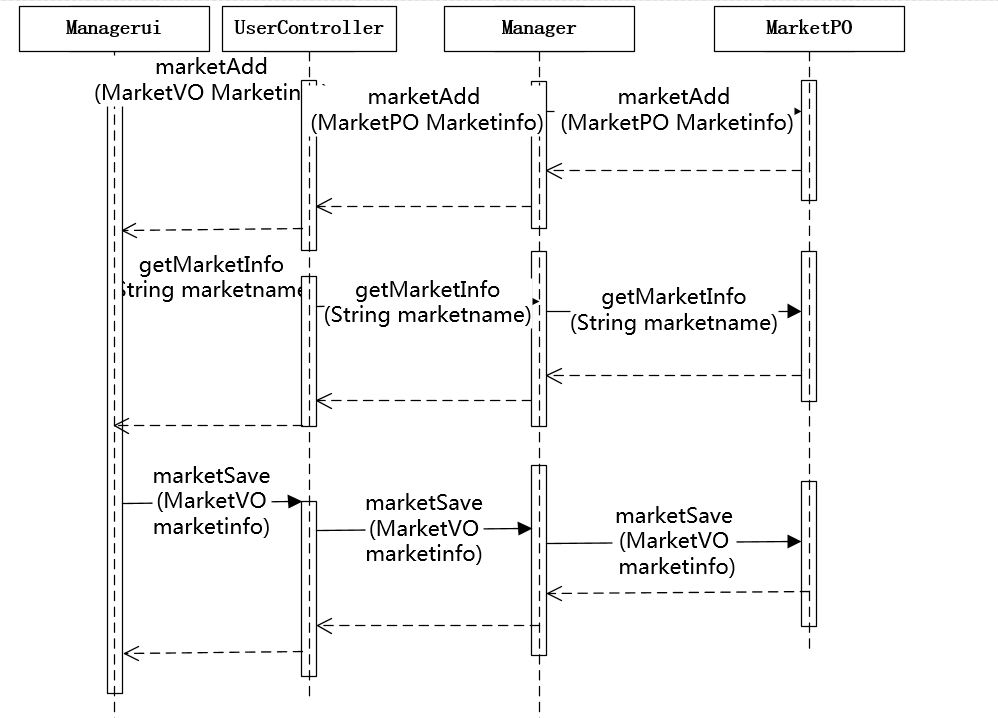
图4.1.2-7是网站管理人员新增、查看、修改网站营销人员信息时各个对象协作的顺序图

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

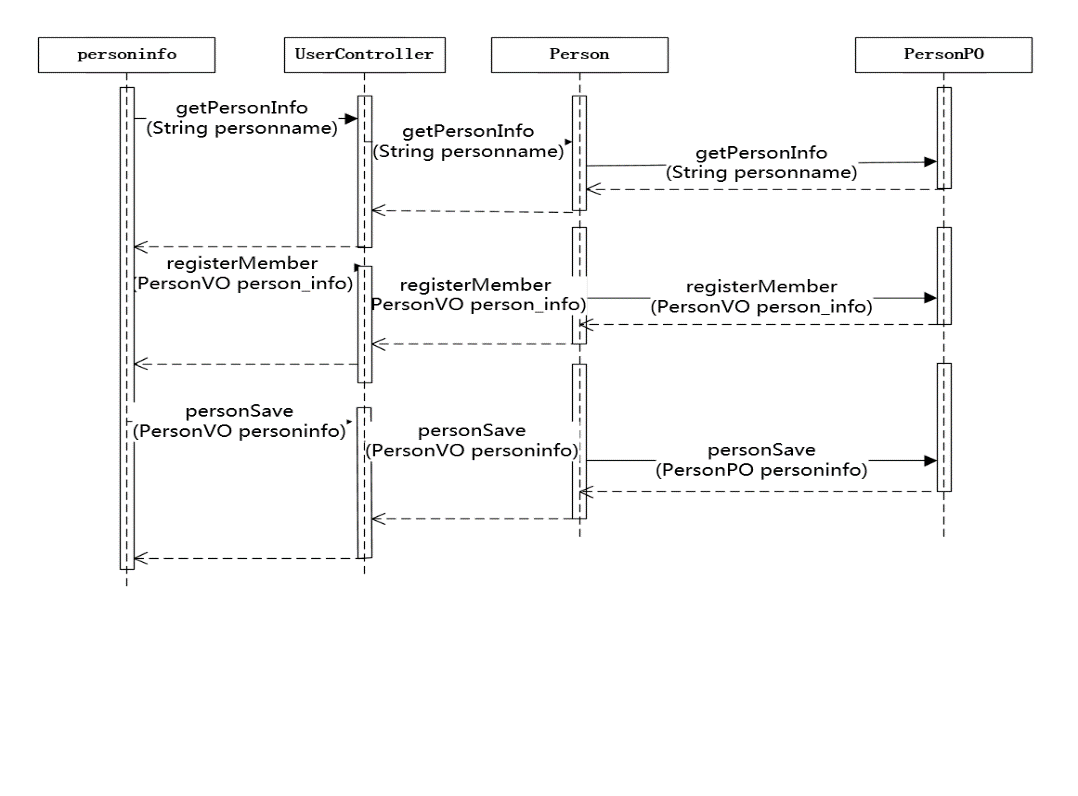


图4.1.2-8客户查看、修改个人信息顺序图

图4.1.2-9是客户注册、登陆时各个对象协作的顺序图

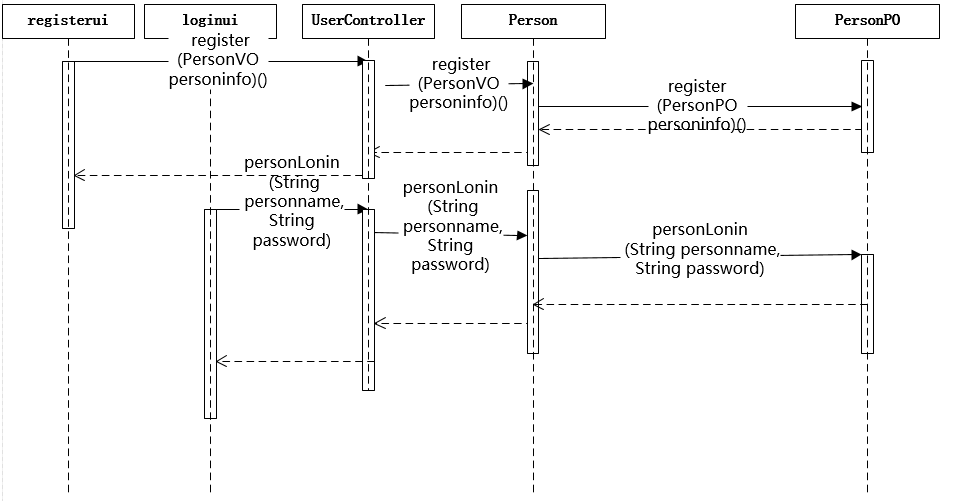


图4.1.2-9客户注册、登陆顺序图

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。

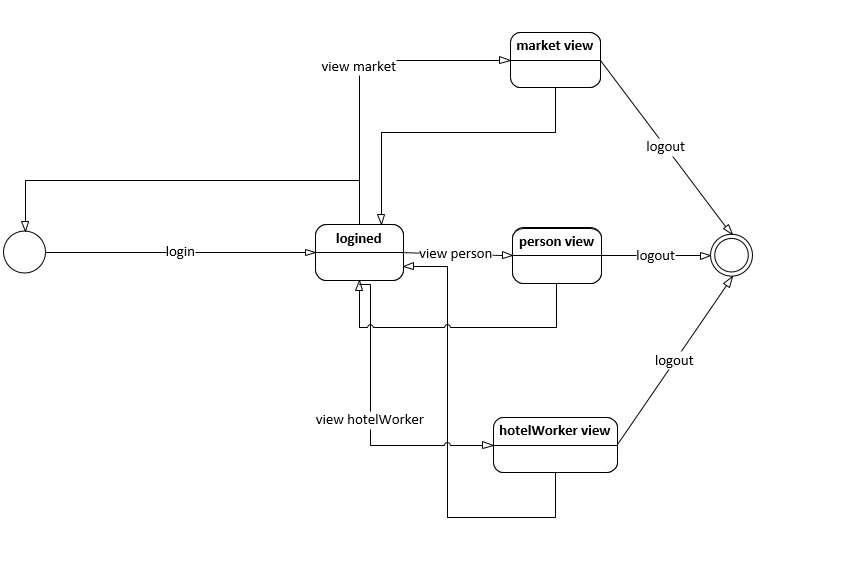


图4.1.2-10Manager对象状态图

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

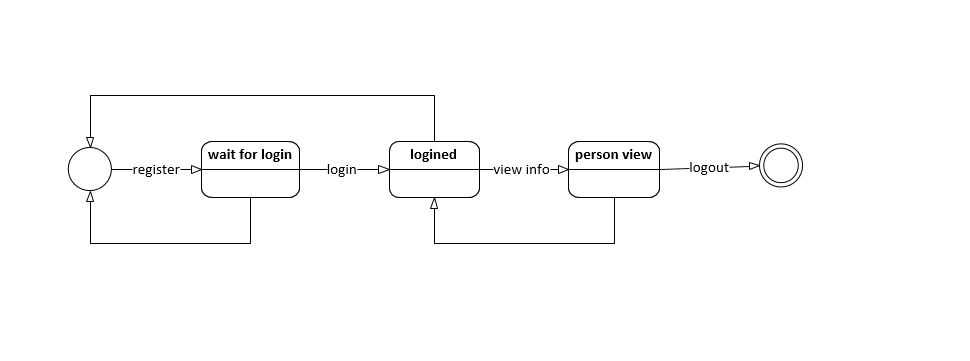


图4.1.2-11Person对象状态图

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

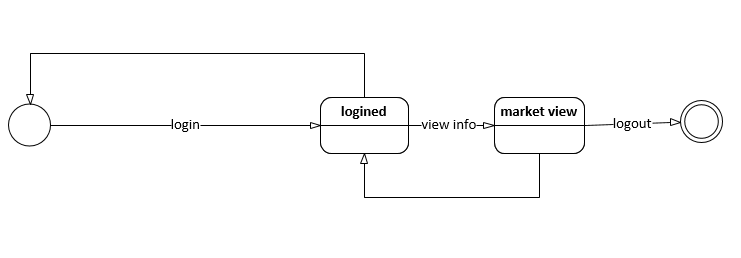


图4.1.2-12Market对象状态图

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

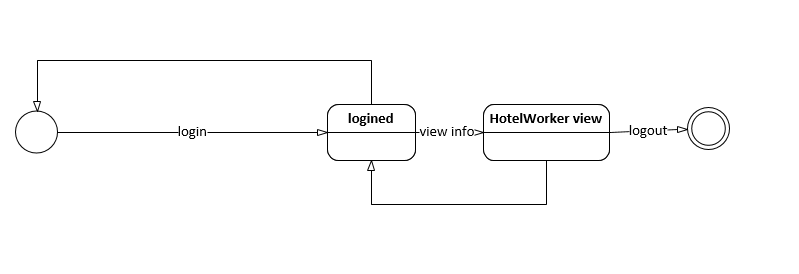


图4.1.2-13HotelWorker对象状态图

### 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.

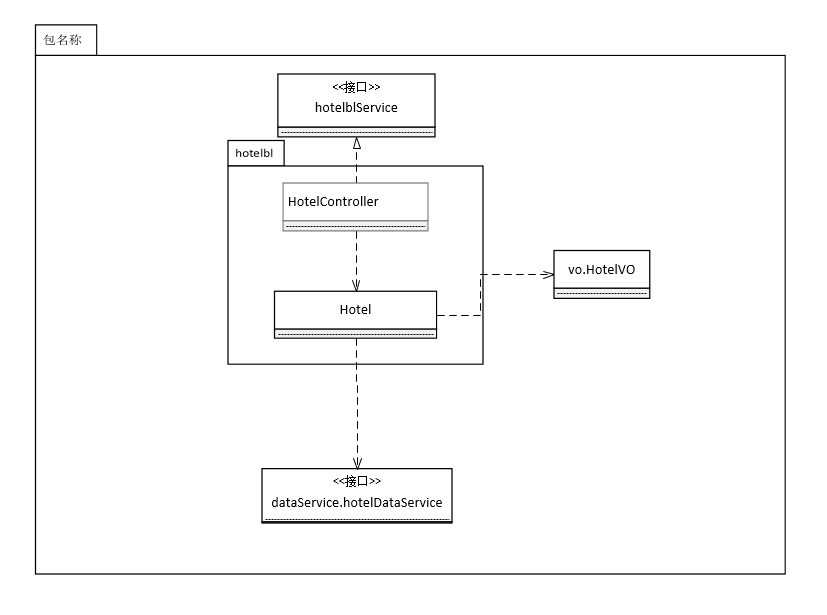


图4.1.3-1hotel模块类的设计

|  |  |
| --- | --- |
| 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.showHotelInfo | 语法 | public HotelPO showHotelInfo(String hotelname); | |
| 前置条件 | 已知酒店编号，需要返回酒店的信息 | |
| 后置条件 | 返回酒店的PO类 | |
| 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 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 | |
| HotelController.addComment | 语法 | public boolean addcomment(String comment，String username,String hotelname); | |
| 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 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Hotel.showHotelInfo(String Hotelname); | | | 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(String comment,String username,String hotelname) | | | 增加评论 |

表4.1.3-3 hotel类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Hotel.showHotelInfo | 语法 | public HotelPO getHotel(String hotelname); | |
| 前置条件 | 已知酒店编号，需要返回酒店的信息 | |
| 后置条件 | 返回酒店的PO类 | |
| 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.addComment | 语法 | public boolean addcomment(String comment，String username,String hotelname); | |
| 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 | |
| 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。

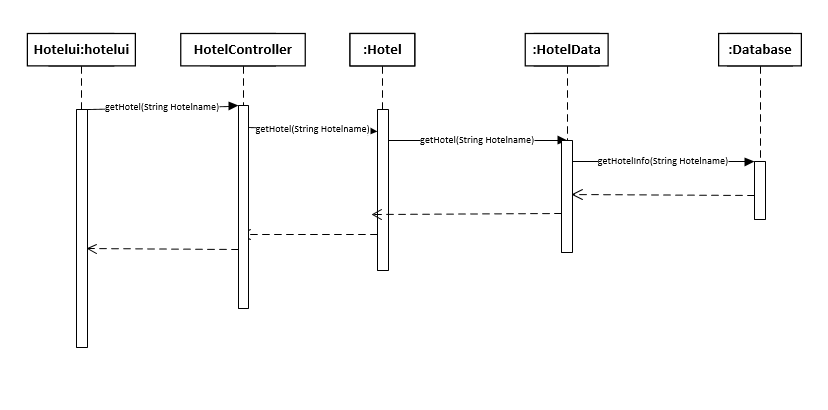


图4.1.3-2 酒店详情查看的顺序图

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。

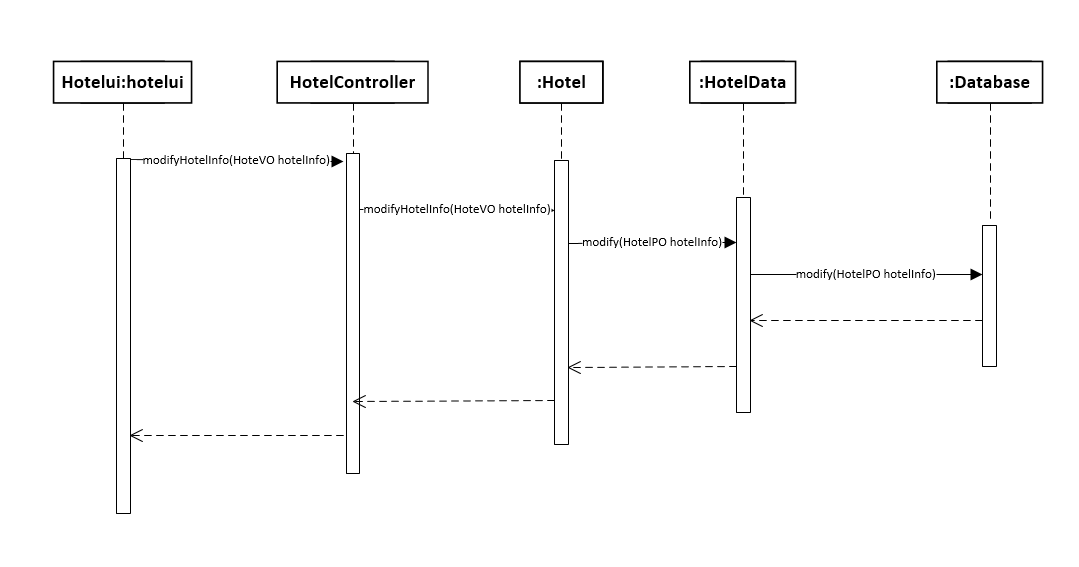


图4.1.3-3 酒店信息维护的顺序图

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。

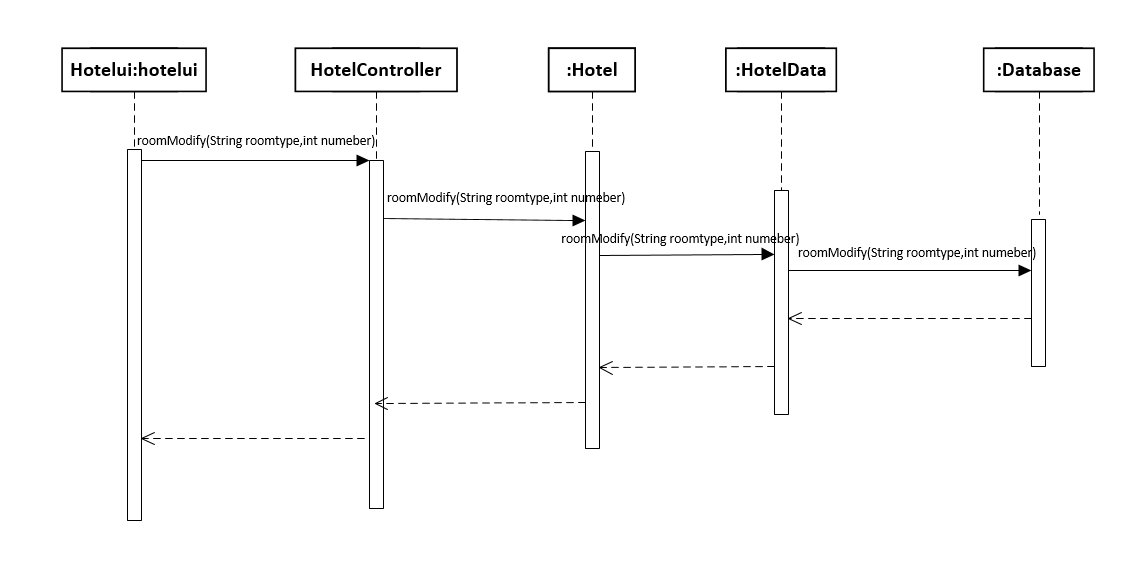


图4.1.3-4酒店房间信息修改的顺序图

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。

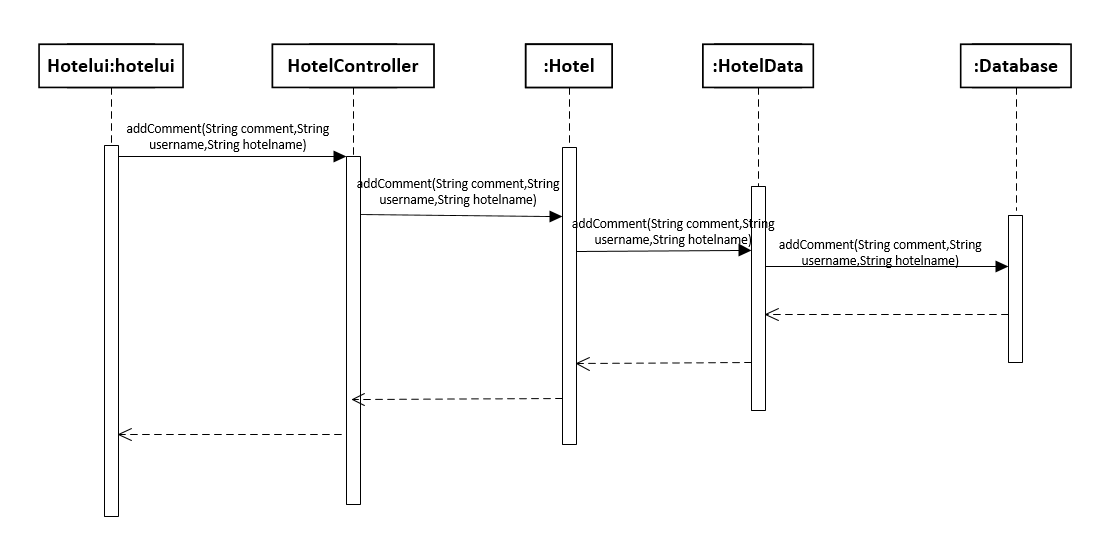


图4.1.3-5评价酒店的顺序图

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

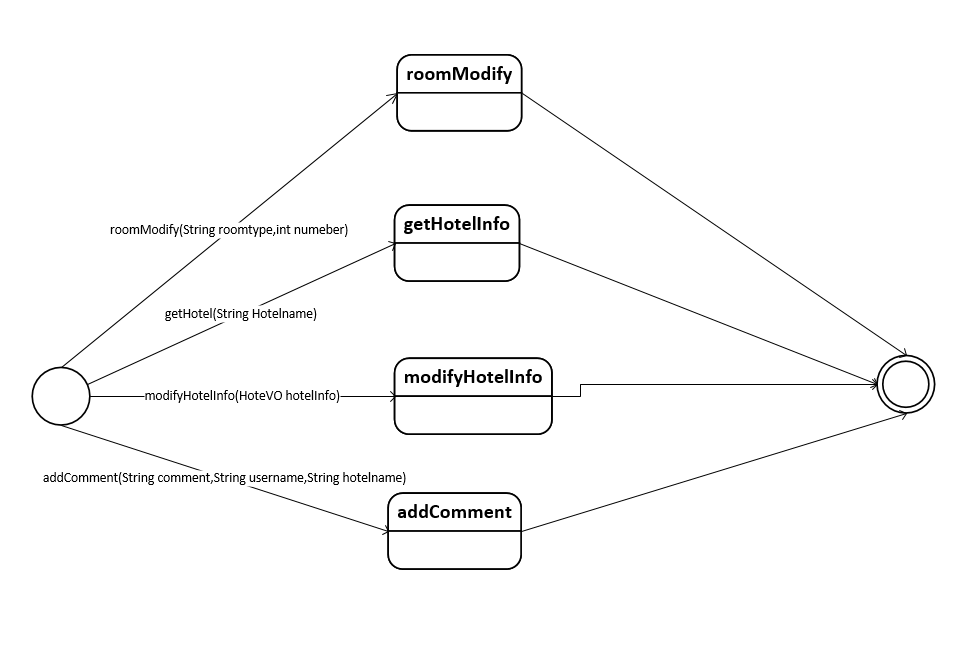


图4.1.3-6 hotel对象的状态图

### 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.

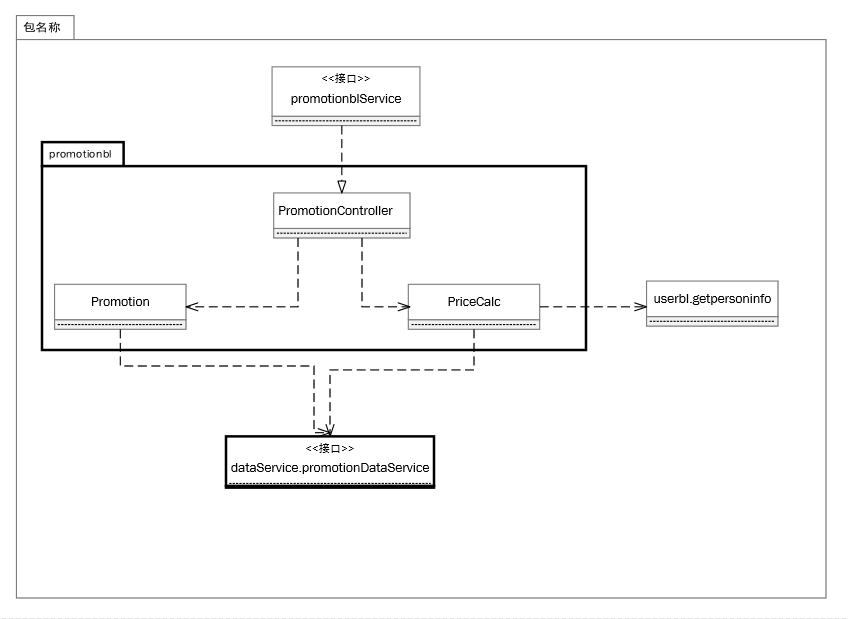


图4.1.4-1promotionbl模块类的设计

|  |  |
| --- | --- |
| 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 |
| PriceCalcu | 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(HotelPO 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  addProm(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 | |
| 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<PromotionPO>getProm(String hotelname); | |
| 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 |
| 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<PromotionPO>getProm(String hotelname); | |
| 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 |

### （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。

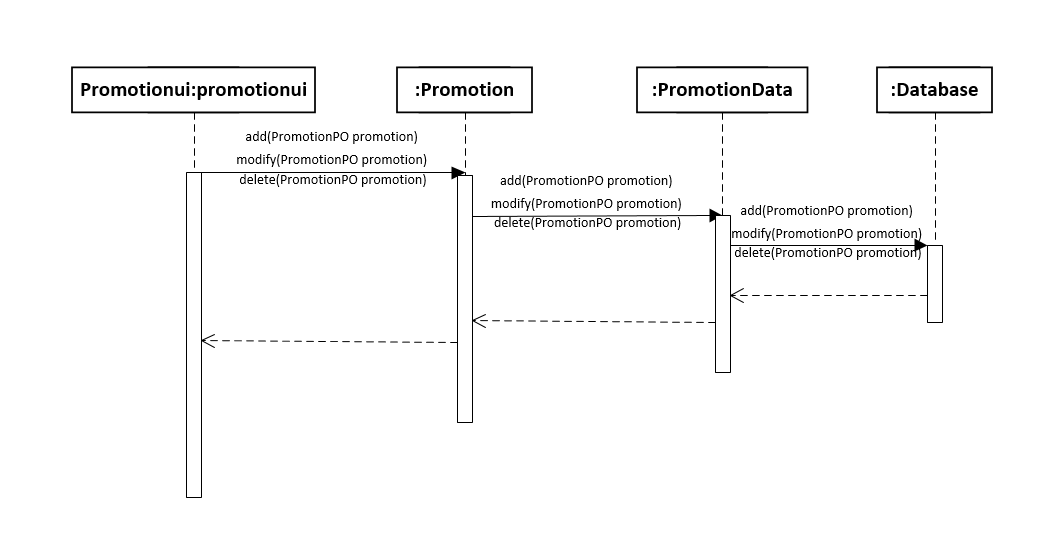


图4.1.4-2 促销策略更改的顺序图

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。

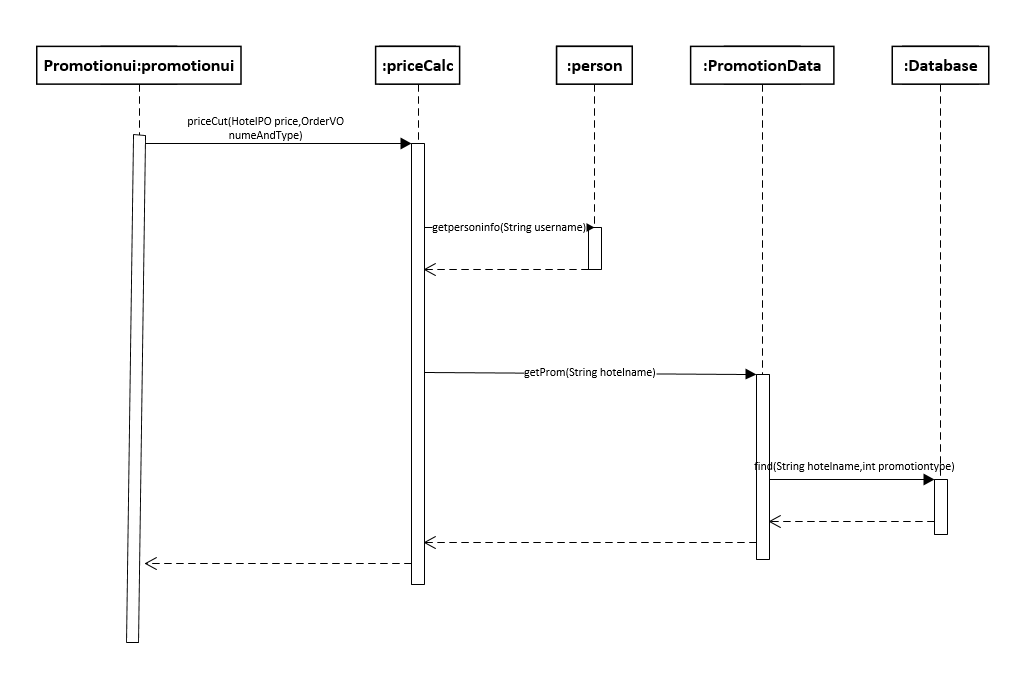


图4.1.4-3 读取促销策略并计算价格的顺序图

### 4.1.5 searchbl模块

（1）模块概述

Searchbl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。

Searchbl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searchblservice, searchdataservice两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了SearchController，这样SearchController将会将用户管理相关的业务逻辑职责和逻辑控制委托给searchbl对象。SearchPO是作为管理信息的持久化对象被添加到设计模型中的。

searchbl模块的设计如图4.1.5-1所示.

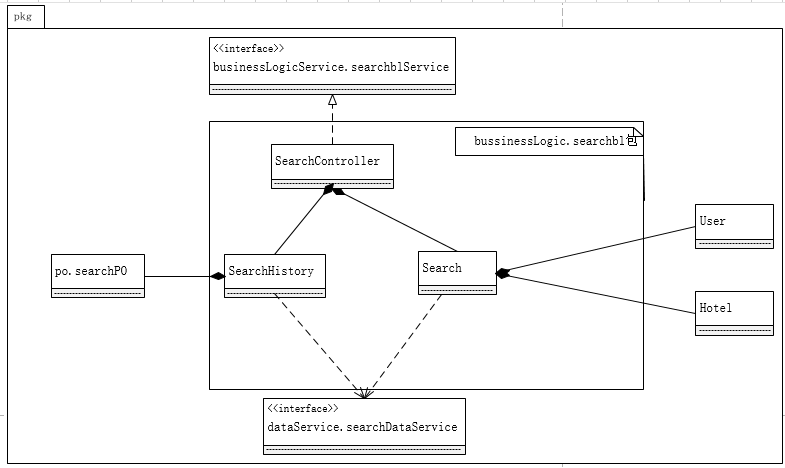


图4.1.5-1search模块类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| SearchController | 负责实现搜索界面所需的服务 |
| Search | 搜索的领域模型对象，拥有搜索所持有的用户、酒店等信息，可以帮助完成搜索界面所需要的服务 |
| SearchHistory | 负责实现与酒店搜索记录相关的服务 |

表4.1.5-1 searchbl模块的各个类的职责

（2）模块内部类的接口规范

### 表4.1.5-2 SearchController类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| SearchController.findHotel | 语法 | public ArrayList<HotelPO> findHotel(HotelPO condition) | |
| 前置条件 | 已创建一个Search领域对象，已添加筛选条件 | |
| 后置条件 | 调用Search领域的findHotel方法 | |
| SearchController.searchPersonInfo | 语法 | public PersonPO searchPeraonInfo (String personname) | |
| 前置条件 | 已创建一个Search领域对象，并且输入符合语法规则 | |
| 后置条件 | 调用Search领域的searchPersonInfo方法 | |
| SearchController.searchHotelWorkerInfo | 语法 | public HotelWorkerPO searchHotelWorkerInfo (String hotelWorkername) | |
| 前置条件 | 已创建一个Search领域对象，并且输入符合语法规则 | |
| 后置条件 | 调用Search领域的searchHotelWorkerInfo方法 | |
| SearchController.searchMarketInfo | 语法 | public MarketPO searchMarketInfo (String M  marketname) | |
| 前置条件 | 已创建一个Search领域对象，并且输入符合语法规则 | |
| 后置条件 | 调用Search领域的searchMarketInfo方法 | |
| SearchController.showHistory | 语法 | public ArrayList<SearchPO> showHistory(String username) | |
| 前置条件 | 已知客户名称 | |
| 后置条件 | 调用SearchHistory领域的showHistory方法 | |
| SearchController.saveHistory | 语法 | public boolean saveHistory (SearchPO SearchHistory); | |
| 前置条件 | 已创建一个SearchHistory领域对象，已有新的搜索记录 | |
| 后置条件 | 调用SearchHistory领域的saveHistory方法 | |
| SearchController.deleteHistory | 语法 | public boolean deleteHistory(SearchPO searchHistory) | |
| 前置条件 | 已选择SearchHistory领域对象 | |
| 后置条件 | 调用SearchHistory领域的deleteHistory方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Search.findHotel | | | 依据条件查找酒店 |
| Search.searchPersonInfo | | | 依据条件查找客户 |
| Search.searchHotelWorkerInfo | | | 依据条件查找酒店工作人员 |
| Search.searchMarketInfo | | | 依据条件查找网站营销人员 |
| SearchHistory.showHistory | | | 显示酒店搜索记录 |
| SearchHistory.saveHistory | | | 保存搜索酒店记录 |
| SearchHistory.deleteHistory | | | 删除搜索酒店记录 |

表4.1.5-3 Search类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Search.findHotel | 语法 | public ArrayList<HotelPO>findHotel(HotelPO condition); | |
| 前置条件 | 已知客户的筛选条件,需要查找符合条件的酒店 | |
| 后置条件 | 返回符合条件的酒店VO列表 | |
| Search.searchPersonInfo | 语法 | public PersonVO searchPersonInfo (String personname) | |
| 前置条件 | 已知需要查询的客户名称，需要返回查找结果 | |
| 后置条件 | 返回查找的对象的VO类 | |
| Search.searchHotelWorkerInfo | 语法 | public HotelWorkerVO searchHotelWorkerInfo (String hotelWorkername) | |
| 前置条件 | 已知需要查询的酒店工作人员名称，需要返回查找结果 | |
| 后置条件 | 返回查找的对象的VO类 | |
| Search.searchMarketInfo | 语法 | public MarketVO searchMarketInfo (String marketname) | |
| 前置条件 | 已知需要查询的网站营销人员名称，需要返回查找结果 | |
| 后置条件 | 返回查找的对象的VO类 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Hotel.findWithReq(HotelPO condition); | | | 依据条件查找酒店 |
| Person.getPersonInfo(String personname) | | | 依据条件查找客户信息 |
| HotelWorker.getHotelWorkerInfo(String hotelWorkername) | | | 依据条件查找酒店工作人员信息 |
| Market.getMarketInfo(String marketname) | | | 依据条件查找网站营销人员信息 |

表4.1.5-4 SearchHistory类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| SearchHistory.showHistory | 语法 | public ArrayList<SearchPO>showHistory(String username); | |
| 前置条件 | 已知客户名称 | |
| 后置条件 | 返回客户酒店搜索记录 | |
| SearchHistory.saveHistory | 语法 | public boolean saveHistory (SearchPO searchHistory); | |
| 前置条件 | 保存搜索酒店记录 | |
| 后置条件 | 持久化保存搜索酒店的记录 | |
| SearchHistory.deleteHistory | 语法 | public boolean deleteHistory(SearchPO searchHistory) | |
| 前置条件 | 已知客户需要删除的酒店编号 | |
| 后置条件 | 数据层删除选定的搜索记录 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| SearchDataService.showSearchHistory(String username); | | | 显示酒店搜索记录 |
| SearchDataService.addSearchHistory (SearchPO searchHistory); | | | 增加酒店搜索记录 |
| SearchDataService.deleteSearchHistory(SearchPO searchHistory) | | | 删除酒店搜索记录 |

### （4） 业务逻辑层的动态模型

如图4.1.5-2表明了互联网酒店预订系统中，当客户输入酒店筛选条件后，搜索酒店相关对象之间的协作。

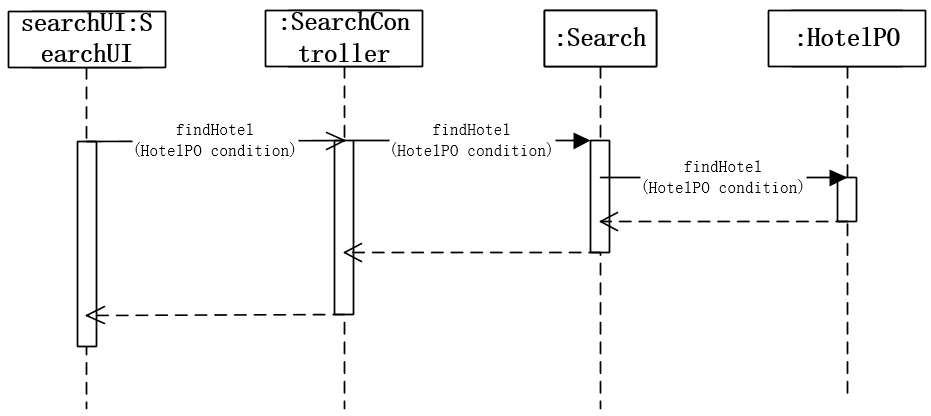


图4.1.5-2 搜索酒店的顺序图

如图4.1.5-3表明了互联网酒店预订系统中，当网站管理人员请求搜索客户后，搜索客户业务逻辑处理相关对象之间的协作。

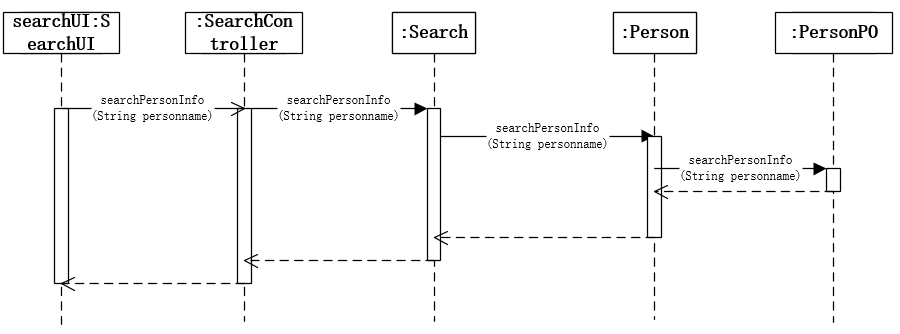


图4.1.5-3 搜索客户的顺序图

如图4.1.5-4表明了互联网酒店预订系统中，当网站管理人员请求搜索酒店工作人员后，搜索酒店工作人员业务逻辑处理相关对象之间的协作。

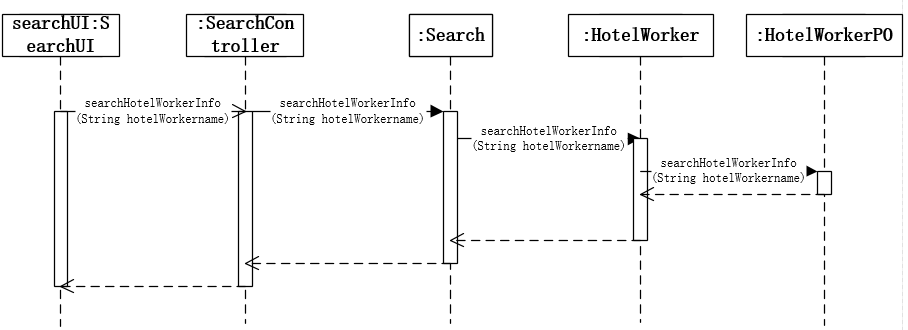


图4.1.5-4 搜索酒店工作人员的顺序图

如图4.1.5-5表明了互联网酒店预订系统中，当网站管理人员请求搜索网站营销人员后，搜索网站营销人员业务逻辑处理相关对象之间的协作。

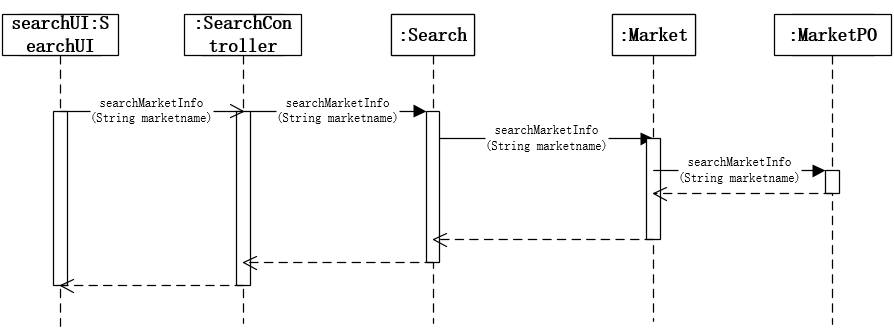


图4.1.5-5 搜索网站营销人员的顺序图

如图4.1.5-6表明了互联网酒店预订系统中，当客户请求查看酒店搜索记录后，酒店搜索记录浏览业务逻辑处理相关对象之间的协作。

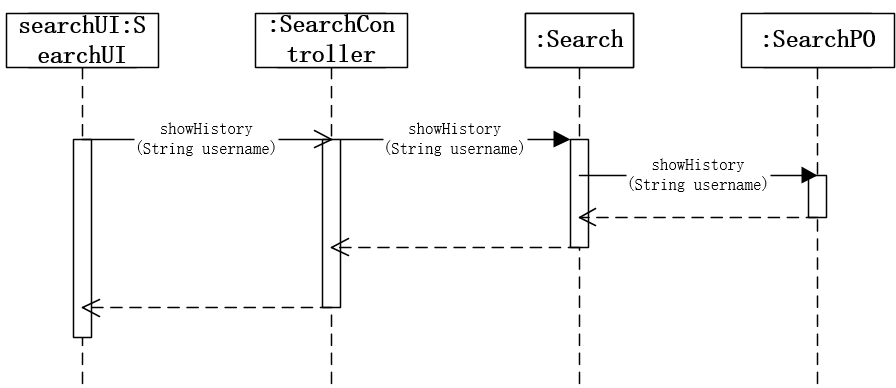


图4.1.5-6 酒店搜索记录的顺序图

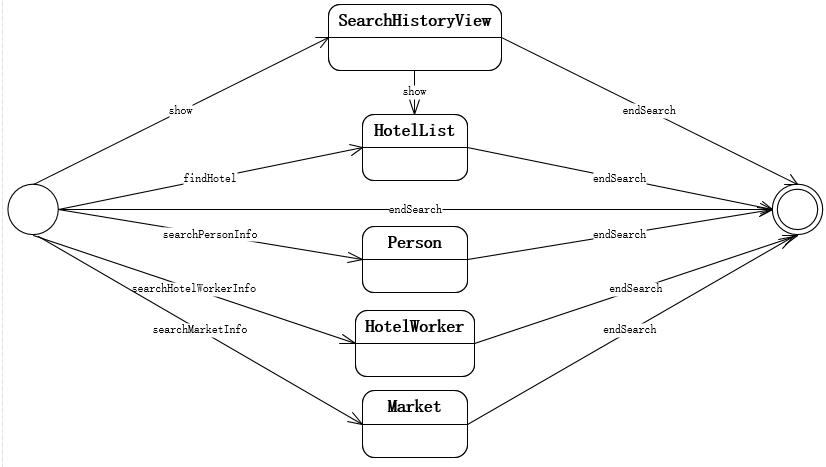


图4.1.5-7 Promotion对象的状态图

# 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 |