**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：曹畅 顾逸飞 黄凯文 唐鑫

完成日期：2016年10月09日

目录

[更新历史 3](#_Toc466228622)

[1 引言 4](#_Toc466228623)

[1.1 编制目的 4](#_Toc466228624)

[1.2 词汇表 4](#_Toc466228625)

[1.3 参考资料 4](#_Toc466228626)

[2 产品概述 4](#_Toc466228627)

[3 体系结构设计概述 5](#_Toc466228628)

[4 结构视角 5](#_Toc466228629)

[4.1 业务逻辑层的分解 5](#_Toc466228630)

[4.1.1 Order模块 5](#_Toc466228631)

[4.1.2 User模块 19](#_Toc466228632)

[4.1.3 Hotel模块 33](#_Toc466228633)

[4.1.4 Vip模块 47](#_Toc466228634)

[4.1.5 Integral模块 51](#_Toc466228635)

[4.1.6 Promotionbl模块 55](#_Toc466228636)

[4.2 数据层的分解 60](#_Toc466228637)

[4.2.1 OrderDataService模块 60](#_Toc466228638)

[4.2.2 UserDataService模块 63](#_Toc466228639)

[4.2.3 HotelDataService模块 66](#_Toc466228640)

[4.2.4 VipDataService模块 68](#_Toc466228641)

[4.2.5 IntegralDataService模块 69](#_Toc466228642)

[4.2.6 PromotionDataService模块 70](#_Toc466228643)

[5 依赖视角 70](#_Toc466228644)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 全体人员 | 2016/10/29 | 详细设计初步模型 | V1.0 |
| 顾逸飞 | 2016/11/6 | 在第二次作业基础上修改 | V1.1 |

# 引言

## 编制目的

本报告详细完成对酒店预订系统的详细设计，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。

## 词汇表

|  |  |
| --- | --- |
| **缩写或单词** | **解释** |
| OrderOnUser | 用户订单 |
| OrderOnWeb | 酒店订单 |
| HotelReservedCheck | 检查预定过的酒店信息 |
| Integral | 信誉积分 |
| PromotionHotel | 酒店营销策略 |
| PromotionWeb | 网站营销策略 |

## 参考资料

《酒店预订系统用例文档V2.0》

《酒店预订系统需求规格说明文档V2.0》

《酒店预订系统软件体系结构描述文档V2.0》

# 产品概述

参考酒店预订系统用例文档及酒店预订系统需求规格说明文档对产品的概括描述。

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。客户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，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。

为了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，该系统能够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，并且能够通过网站推广，吸引酒店和顾客的入驻。它包括一个数据集中服务器和多个客户端。数据集中服务器将所有的数据储存起来进行维护。用户通过客户端完成日常任务，客户端与数据集中服务器才是实时通信的方式完成数据交换。

# 体系结构设计概述

参考酒店预订系统软件体系结构描述文档。

主要采用分层的架构方式，采用分布式的部署方式，将数据存储在服务器端，客户端

通过RMI方式调用获取数据。

# 结构视角

## 业务逻辑层的分解

业务逻辑层的开发包图请查看体系结构设计文档。

### Order模块

1、模块概述：

Order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。

Order模块的职责和接口参见软件系统结构描述文档内对该模块的描述。

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.OrderOnUserBLService接口。业务逻辑层和数据层之间添加dataservice.orderdataservice.OrderOnUserDataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加OrderOnUserController，这样OrderOnUserController会将对订单的业务逻辑处理委托给OrderOnUser对象。OrderOnUserPO是作为账户信息的持久性对象被添加到设计模型中去的。

Orderbl模块的设计如图1-1、图1-2、图1-3：

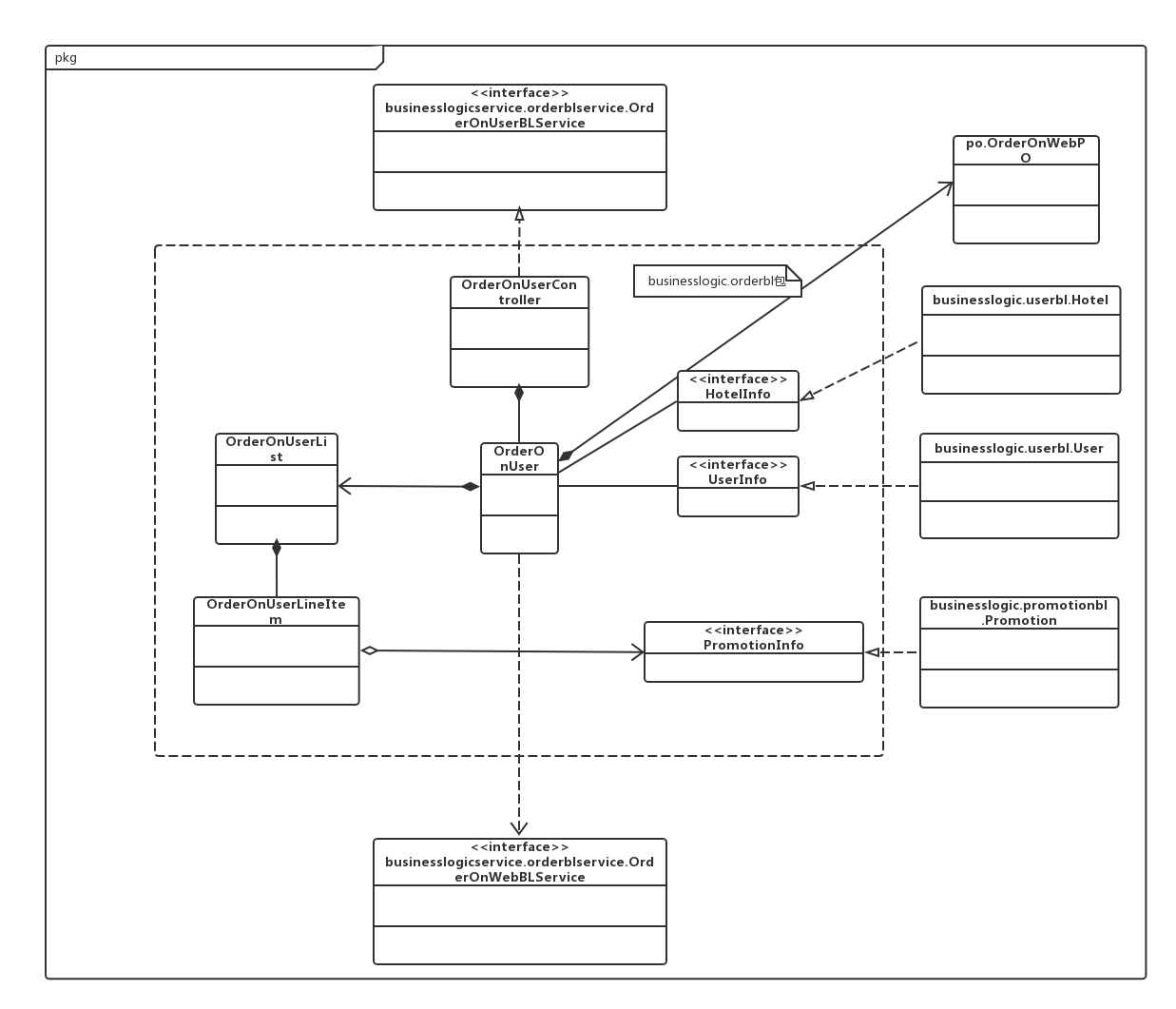


图1-1 Orderbl模块中各个类的设计

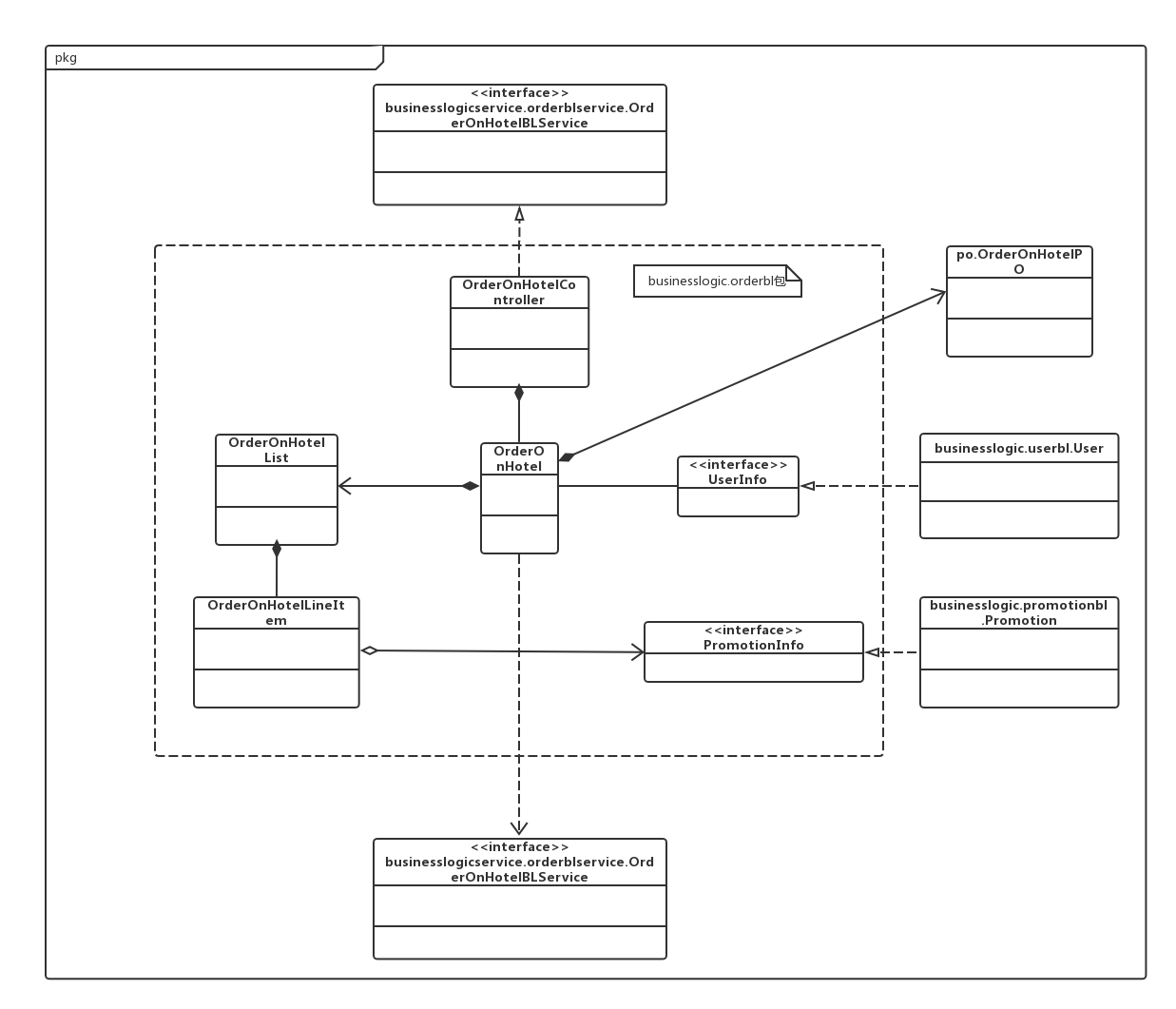


图1-2 Orderbl模块中各个类的设计

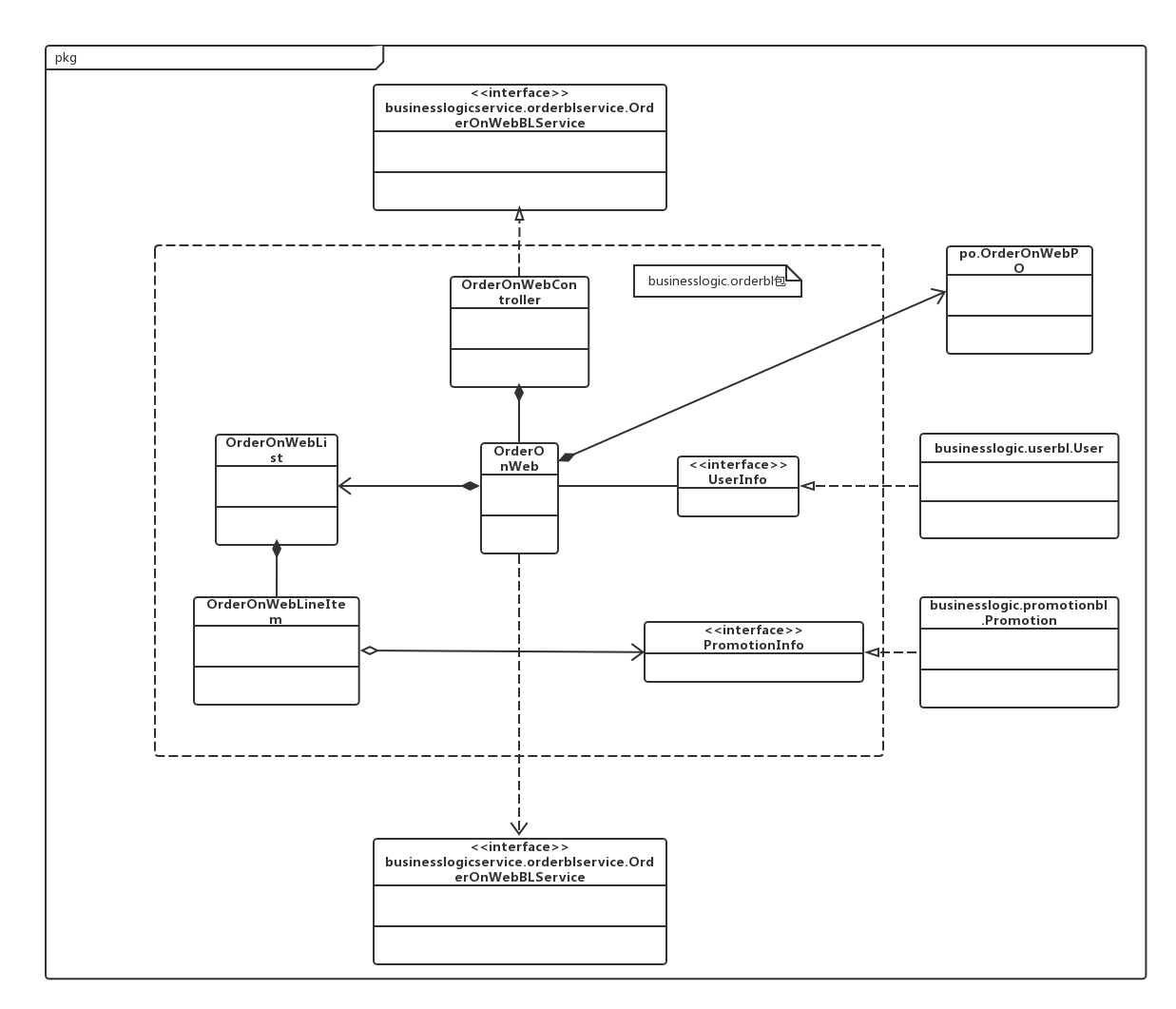


图1-3 Orderbl模块中各个类的设计

Orderbl模块各个类的职责如表1所示。

表1 Orderbl模块各个类的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| OrderOnUserController | 负责实现对应于个人订单界面所需要的服务 |
| OrderOnHotelController | 负责实现对应于酒店订单界面所需要的服务 |
| OrderOnWebController | 负责实现对应于网站订单界面所需要的服务 |
| OrderOnUser | 个人订单对象，拥有个人订单的信息，可以帮助完成个人订单界面、个人订单详情界面、个人订单撤销界面所需要的服务 |
| OrderOnHotel | 酒店订单对象，拥有酒店订单的信息，可以帮助完成酒店订单列表界面、酒店订单详情界面、酒店订单状态修改界面所需要的服务 |
| OrderOnWeb | 网站订单对象，拥有网站订单的信息，可以帮助完成异常订单界面、异常订单详情界面、申诉列表界面、申诉处理界面所需要的服务 |

3、模块内部类的接口规范

OrderOnUserController和OrderOnUser的接口规范如表2-1-1和表2-1-2所示。

表2-1-1 OrderOnUserController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderOnUserController.personalOrderScan | 语法 | public ArrayList<OrderOnUserVO> personalOrderScan(); |
| 前置条件 | 已经创建一个OrderOnUser领域对象 |
| 后置条件 | 调用OrderOnUser领域对象的personalOrderScan方法，持久化更新相关数据 |
| OrderOnUserController.personalOrderCancel | 语法 | public ResultMsg personalOrderCancel(OrderOnUserVO orderVO); |
| 前置条件 | 已经创建一个OrderOnUser领域对象 |
| 后置条件 | 调用OrderOnUser领域对象的personalOrderCancel方法，系统持久化更新相关数据 |
| OrderOnUserController.personalOrderDetail | 语法 | public OrderMsg personalOrderDetail(OrderOnUserVO orderVO); |
| 前置条件 | 已经创建一个OrderOnUser领域对象 |
| 后置条件 | 调用OrderOnUser领域对象的personalOrderDetail方法 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| OrderOnUser.personalOrderScan() | 查查看个人订单 | |
| OrderOnUser.personalOrderCancel(OrderOnUserVO vo) | 撤销个人订单 | |
| OrderOnUser.personalOrderDetail(OrderOnUserVO vo) | 查看个人订单详情 | |

表2-1-2 OrderOnUser的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderOnUser. personalOrderScan | 语法 | public ArrayList<OrderOnUserVO> personalOrderScan(); |
| 前置条件 | 订单信息符合规则且齐全 |
| 后置条件 | 系统显示该订单 |
| OrderOnUser. personalOrderCancel | 语法 | public ResultMsg personalOrderCancel(OrderOnUserVO orderVO); |
| 前置条件 | 订单信息符合规则且齐全 |
| 后置条件 | 系统提示撤销订单成功 |
| OrderOnUser. personalOrderDetail | 语法 | public ArrayList<OrderOnHotelVO> hotelOrderDetail(OrderOnUserVO orderVO); |
| 前置条件 | 账户信息符合规则且齐全 |
| 后置条件 | 系统显示个人订单详情 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| OrderOnUserDataService.  insert(OrderOnUserPO po) | 新增单一持久化对象 | |
| OrderOnUserDataService.  delete(OrderOnUserPO po) | 删除单一持久化对象 | |
| OrderOnUserDataService.  update(OrderOnUserPO po) | 更新单一持久化对象 | |
| OrderOnUserDataService.findByName  (OrderOnUesrPO po) | 根据姓名查找单一持久化对象 | |
| OrderOnUserDataService.findByCondition  (OrderOnUesrPO po) | 根据条件查找单一持久化对象 | |

OrderOnHotelController和OrderOnHotel的接口规范如表2-2-1和表2-2-2所示。

表2-2-1 OrderOnHotelController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderOnHotelController.hotelOrderScan | 语法 | public ArrayList<OrderOnHotelVO> hotelOrderScan(); |
| 前置条件 | 已经创建一个OrderOnHotel领域对象 |
| 后置条件 | 调用OrderOnHotel领域对象的hotelOrderScan方法，持久化更新相关数据 |
| OrderOnHotelController.hotelOrderModify | 语法 | public ResultMsg hotelOrderModify(OrderOnHotelVO orderVO); |
| 前置条件 | 已经创建一个OrderOnHotel领域对象 |
| 后置条件 | 调用OrderOnHotel领域对象的hotelOrderModify方法，系统持久化更新相关数据 |
| OrderOnHotelController.hotelOrderDetail | 语法 | public OrderMsg hotelOrderDetail(OrderOnHotelVO orderVO); |
| 前置条件 | 已经创建一个OrderOnHotel领域对象 |
| 后置条件 | 调用OrderOnHotel领域对象的hotelOrderDetail方法 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| OrderOnHotel.hotelOrderScan() | 查看酒店订单 | |
| OrderOnHotel.hotelOrderNodify(OrderOnHotelVO vo) | 修改酒店订单状态 | |
| OrderOnHotel.hotelOrderDetail(OrderOnHotelVO vo) | 查看酒店订单详情 | |

表2-2-2 OrderOnHotel的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderOnHotel. hotelOrderScan | 语法 | public ArrayList<OrderOnHotelVO> hotelOrderScan(); |
| 前置条件 | 订单信息符合规则且齐全 |
| 后置条件 | 系统显示该订单 |
| OrderOnHotel. hotelOrderModify | 语法 | public ResultMsg personalOrderModify(OrderOnHotelVO orderVO); |
| 前置条件 | 订单信息符合规则且齐全 |
| 后置条件 | 系统提示修改订单成功 |
| OrderOnHotel. hotelOrderDetail | 语法 | public ArrayList<OrderOnHotelVO> hotelOrderDetail(OrderOnHotelVO orderVO); |
| 前置条件 | 账户信息符合规则且齐全 |
| 后置条件 | 系统显示酒店订单详情 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| OrderOnHotelDataService.insert(OrderOnHotelPO po) | 新增单一持久化对象 | |
| OrderOnHotelDataService.delete(OrderOnHotelPO po) | 删除单一持久化对象 | |
| OrderOnHotelDataService.update(OrderOnHotelPO po) | 更新单一持久化对象 | |
| OrderOnHotelDataService.find  (OrderOnHotelPO po) | 查找单一持久化对象 | |
| OrderOnHotelDataService.show(OrderOnHotelPO po) | 显示单一持久化对象 | |

OrderOnWebController和OrderOnWeb的接口规范如表2-3-1和表2-3-2所示。

表2-3-1 OrderOnWebController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderOnWebController.complaintListScan | 语法 | public ArrayList<OrderOnWebVO> complaintListScan(); |
| 前置条件 | 已经创建一个OrderOnWeb领域对象 |
| 后置条件 | 调用OrderOnWeb领域对象的complaintListScan方法，持久化更新相关数据 |
| OrderOnWebController.complaintHandle | 语法 | public ResultMsg complaintHandle(OrderOnWebVO orderVO); |
| 前置条件 | 已经创建一个OrderOnWeb领域对象 |
| 后置条件 | 调用OrderOnWeb领域对象的complaintHandle方法，系统持久化更新相关数据 |
| OrderOnWebController.abnormalOrderScan | 语法 | public ArrayList<OrderOnWebVO> abnormalOrderScan(); |
| 前置条件 | 已经创建一个OrderOnWeb领域对象 |
| 后置条件 | 调用OrderOnWeb领域对象的abnormalOrderScan方法 |
| OrderOnWebController.abnormalOrderDetail | 语法 | public ArrayList<OrderOnWebVO> abnormalOrderDetail(OrderOnWebVO orderVO); |
| 前置条件 | 已经创建一个OrderOnWeb领域对象 |
| 后置条件 | 调用OrderOnWeb领域对象的abnormalOrderDetail方法 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| OrderOnWeb.complaintListScan() | 查看申诉列表 | |
| OrderOnWeb.conplaintHandle(OrderOnWebVO vo) | 处理申诉 | |
| OrderOnWeb.abnormalOrderScan() | 浏览异常订单 | |
| OrderOnWeb.abnormalOrderDetail(OrderOnWebVO vo) | 查看异常订单详情 | |

表2-3-2 OrderOnWeb的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderOnWeb. complaintListScan | 语法 | public ArrayList<OrderOnWebVO> complaintListScan(); |
| 前置条件 | 订单信息符合规则且齐全 |
| 后置条件 | 系统显示申诉列表 |
| OrderOnWeb. complaintHandle | 语法 | public ResultMsg complaintHandle(OrderOnWebVO orderVO); |
| 前置条件 | 订单信息符合规则且齐全 |
| 后置条件 | 系统处理申诉成功 |
| OrderOnWeb. abnormalOrderScan | 语法 | public ArrayList<OrderOnWebVO> abnormalOrderScan(); |
| 前置条件 | 账户信息符合规则且齐全 |
| 后置条件 | 系统显示异常订单 |
| OrderOnWeb. abnormalOrderDetail | 语法 | public OrderMsg abnormalOrderDetail(OrderOnWebVO orderVO); |
| 前置条件 | 账户信息符合规则且齐全 |
| 后置条件 | 系统显示异常订单详情 |
| 需要的服务（需接口） | | |
| 服务名 | 服务内容 | |
| OrderOnWebDataService.  insert(OrderOnWebPO po) | 新增单一持久化对象 | |
| OrderOnWebDataService.  delete(OrderOnWebPO po) | 删除单一持久化对象 | |
| OrderOnWebDataService.  update(OrderOnWebPO po) | 更新单一持久化对象 | |
| OrderOnWebDataService.findByName  (OrderOnWebPO po) | 根据姓名查找单一持久化对象 | |
| OrderOnWebDataService.findByCondition(OrderOnUesrPO po) | 根据条件查找单一持久化对象 | |
| OrderOnWebDataService.show(OrderOnWebPO po) | 显示单一持久化对象 | |

4、业务逻辑层的动态模型

图2-1-1表明了在酒店预订系统中，客户查看个人订单中相关对象之间的协作。

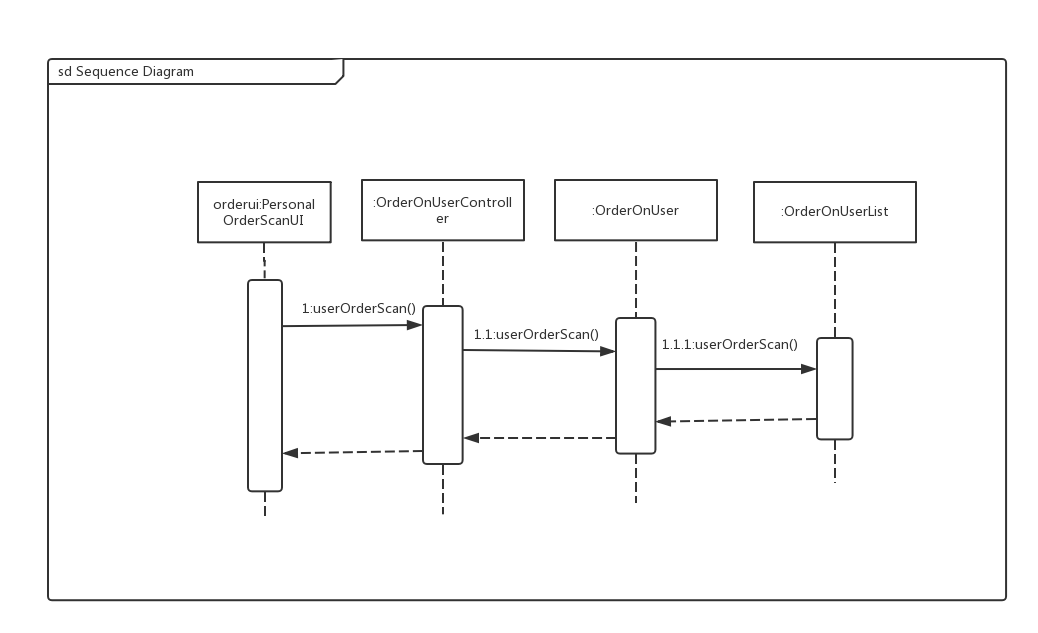


图2-1-1 查看个人订单的顺序图

图2-1-2表明了在酒店预订系统中，客户查看个人订单详情中相关对象之间的协作。

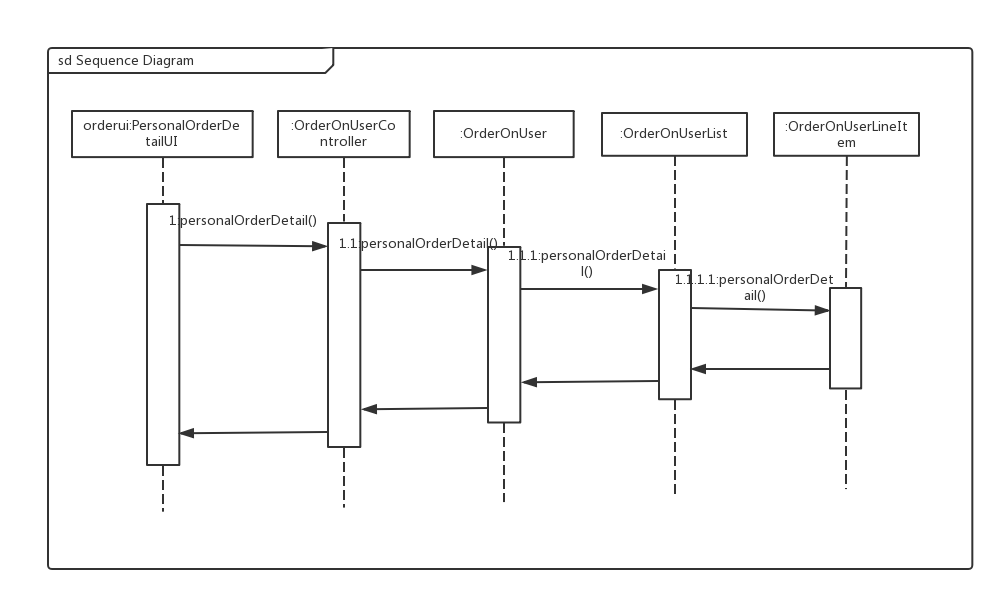


图2-1-2 查看个人订单详情的顺序图

图2-1-3表明了在酒店预订系统中，客户撤销订单中相关对象之间的协作。

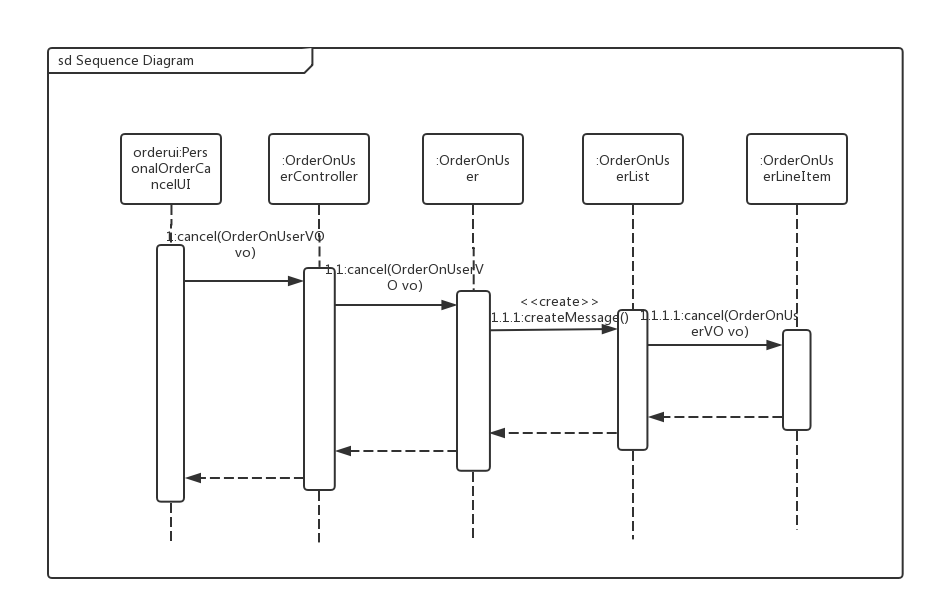


图2-1-3 撤销订单的顺序图

图2-2-1表明了在酒店预订系统中，酒店工作人员查看酒店订单信息中相关对象之间的协作。

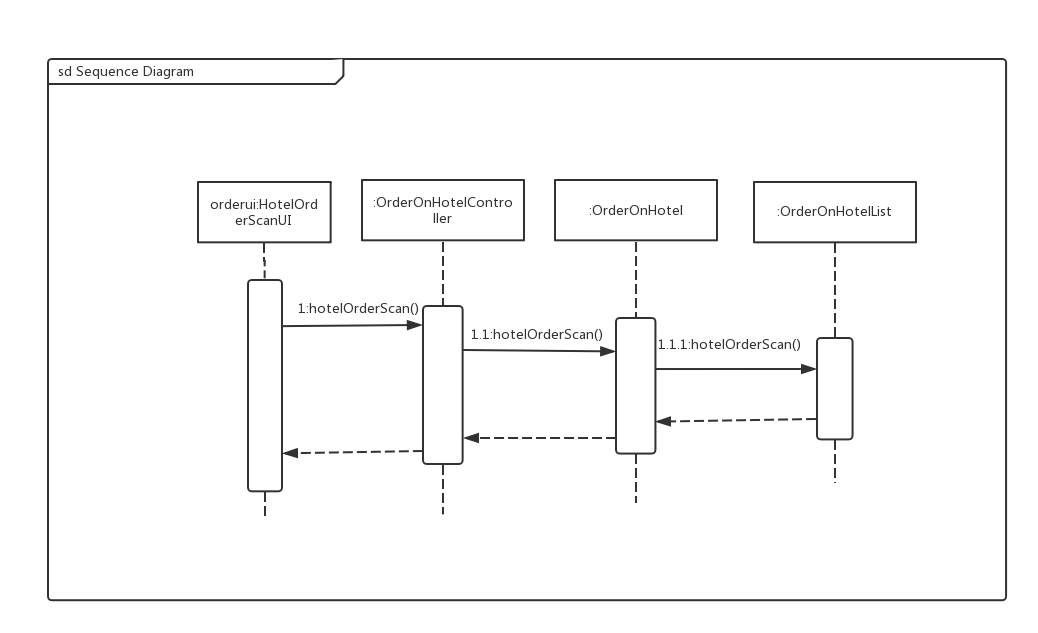


图2-2-1 查看酒店订单信息的顺序图

图2-2-2表明了在酒店预订系统中，酒店工作人员查看酒店订单详情中相关对象之间的协作。

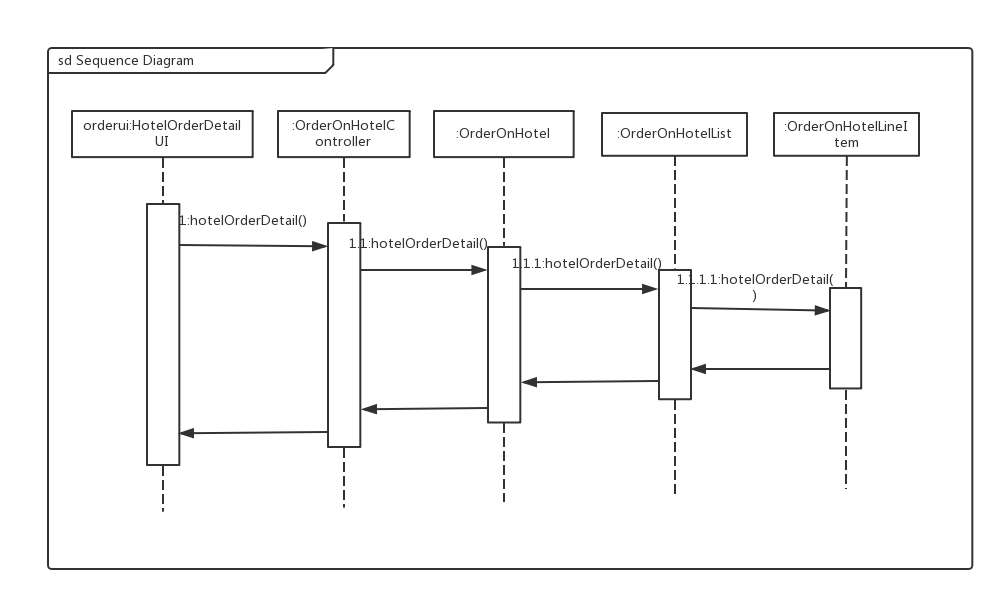


图2-2-2 查看酒店订单详情的顺序图

图2-2-3表明了在酒店预订系统中，酒店工作人员修改订单状态中相关对象之间的协作。

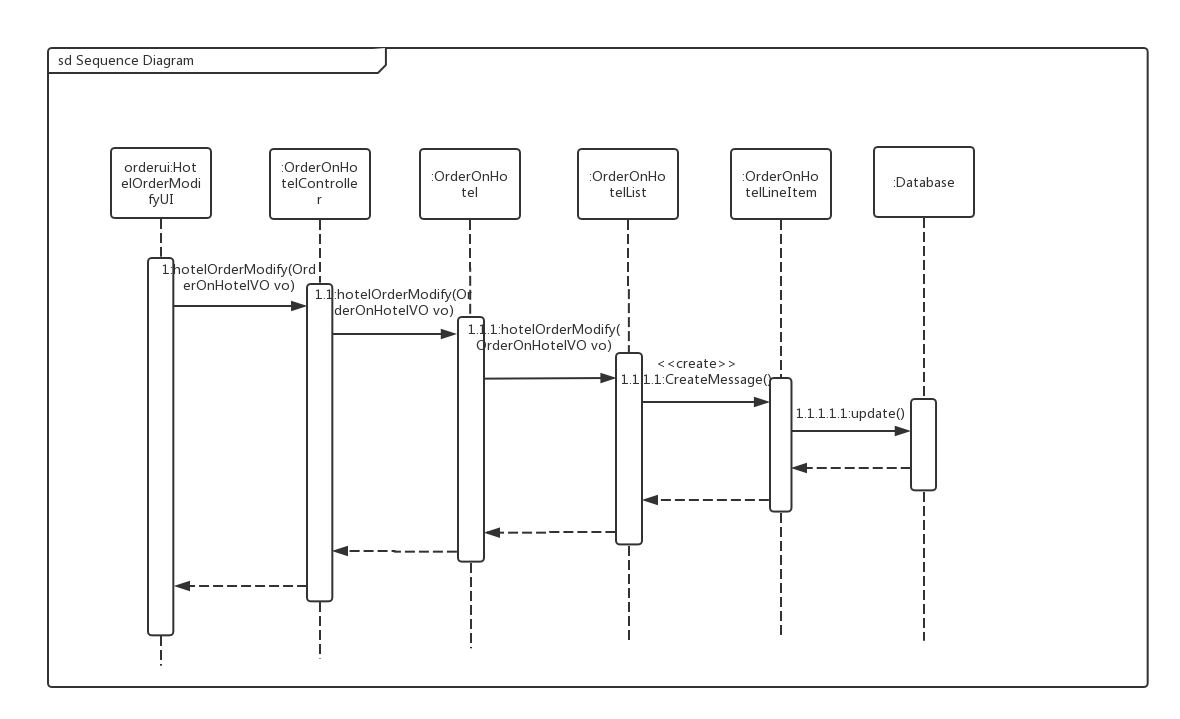


图2-2-3 修改订单状态的顺序图

图2-3-1表明了在酒店预订系统中，网站营销人员查看异常订单中相关对象之间的协作。

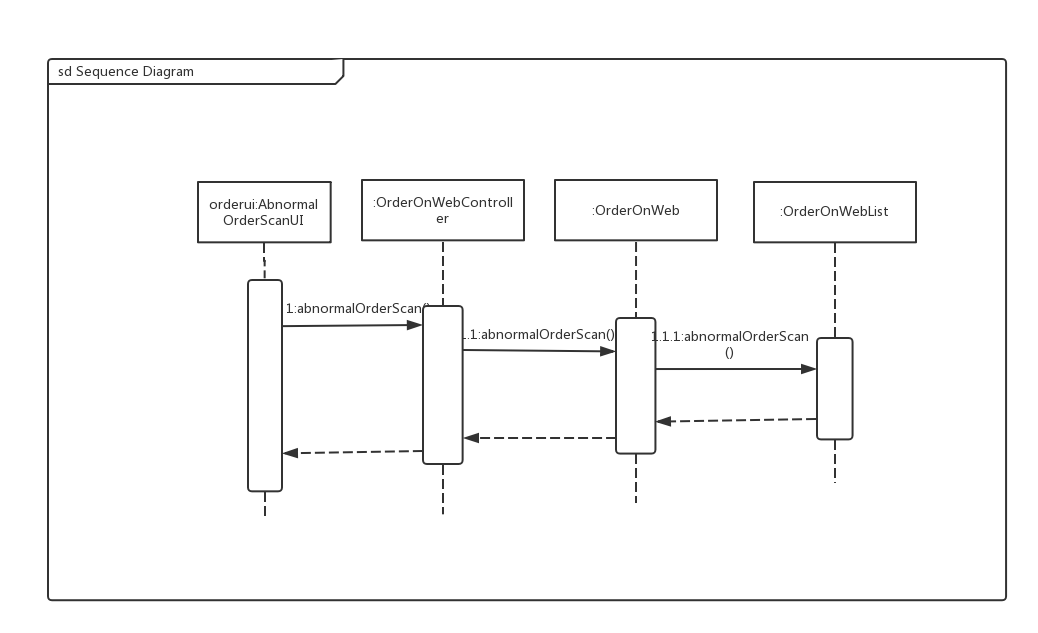


图2-3-1 查看异常订单的顺序图

图2-3-2表明了在酒店预订系统中，网站营销人员查看异常订单详情中相关对象之间的协作。

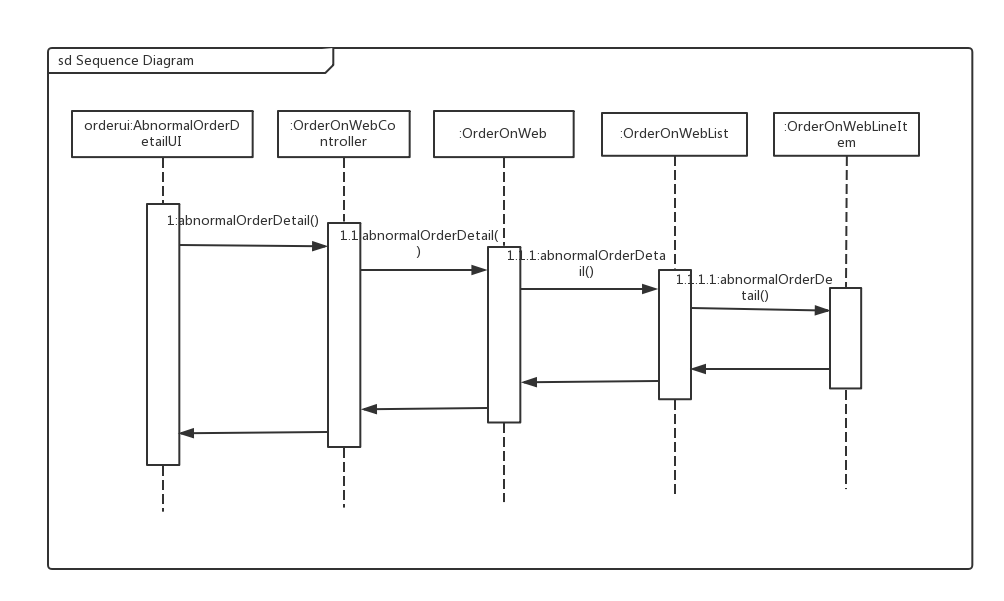


图2-3-2 查看异常订单详情的顺序图

图2-3-3表明了在酒店预订系统中，网站营销人员查看申诉列表中相关对象之间的协作。

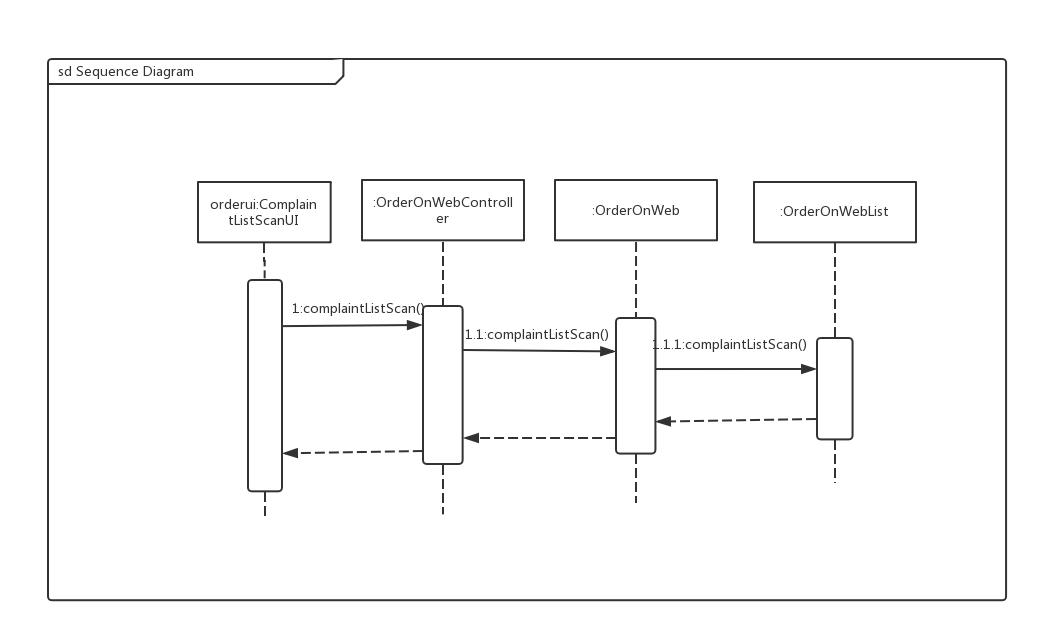


图2-3-3 查看申诉列表的顺序图

图2-3-4表明了在酒店预订系统中，网站营销人员处理申诉中相关对象之间的协作。

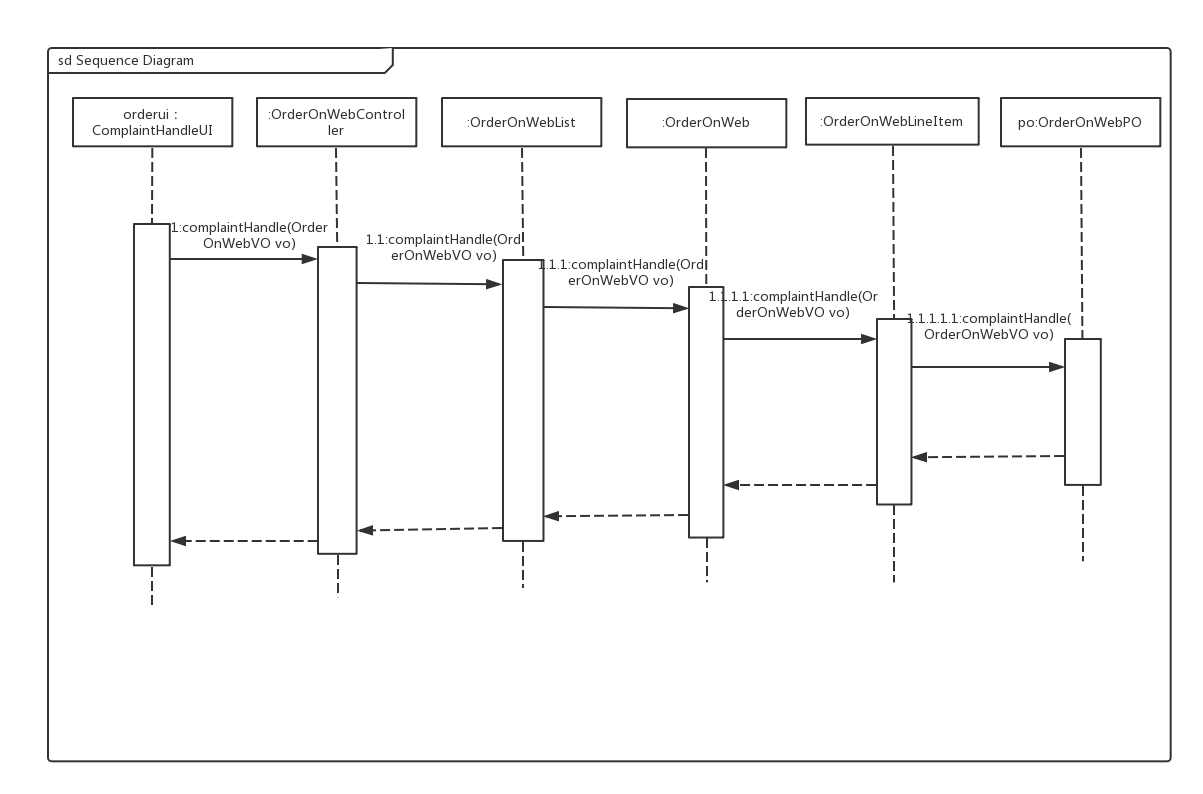


图2-3-4 处理申诉的顺序图

下图描述了OrderOnUser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。

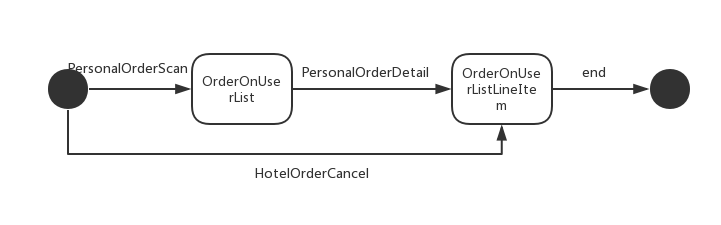


图3-1 OrderOnUser状态图

下图描述了OrderOnHotel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。

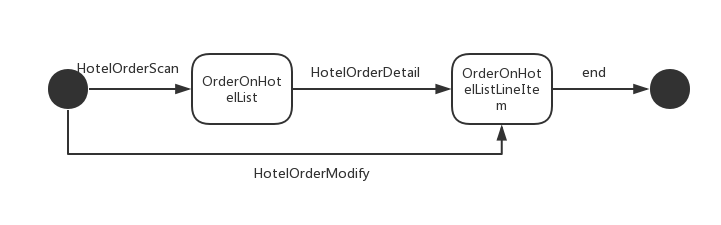


图3-1 OrderOnHotel状态图

下图描述了OrderOnWeb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。

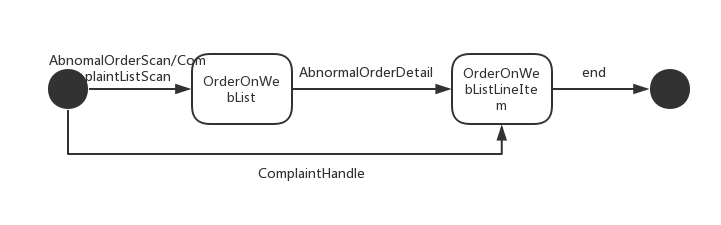


图3-1 OrderOnWeb状态图

5、业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### User模块

1. 模块概述

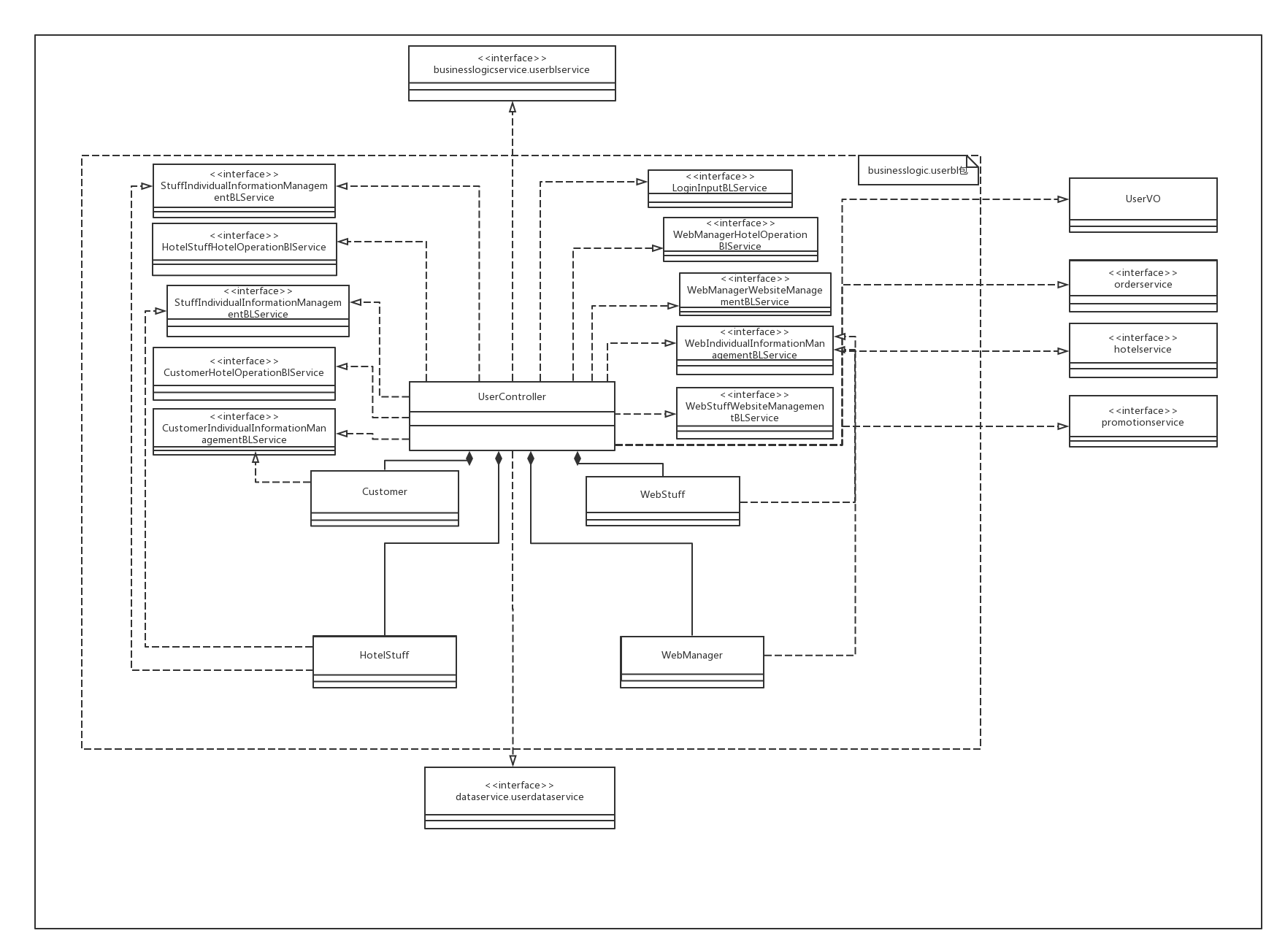
user模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

user模块的职责和接口参见软件系统结构描述文档内对该模块的描述。

1. 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。在展示层和业务逻辑层之间添加UserblService接口。在业务逻辑层和数据层之间添加UserdataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加UserController，这样UserController会将对商品管理的业务逻辑处理委托给相应的用户对象。UserPO是作为账户信息的持久性对象被添加到设计模型中去的。

Userbl模块的设计如图



Userbl模块中各个类的设计

|  |  |
| --- | --- |
| 类 | 职责 |
| UserController | 负责用户信息管理和维护的各种需求 |
| Customer | 客户对象，拥有客户信息，可以帮助完成客户界面所需要的服务 |
| HotelStuff | 酒店管理人员对象，拥有酒店管理人员信息和对应酒店部分信息，可以帮助完成酒店管理界面所需要的服务。 |
| WebStuff | 网站营销人员对象，拥有网站营销人员信息，有，有管理网站营销策略和网站订单信息的权限 |
| WebManager | 网站管理人员对象，又有网站管理人员信息，可以访问网站和管理网站所有用户及酒店的信息。 |

1. 模块内部类的接口规范

UserController的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserController.LogIn | 语法 | Public boolean LogIn(LoginInputVO vo); |
| 前置条件 | 已创建过一个用户对象。 |
| 后置条件 | 通过访问用户信息数据库检查账号和密码是否匹配，设置当前用户。 |
| UserController.LogOut | 语法 | Public boolean LogOut(); |
| 前置条件 | 程序的当前用户不为空。 |
| 后置条件 | 程序的当前用户为空。 |
| UserController.Register | 语法 | Public boolean Register(RegisterInputVO vo); |
| 前置条件 | 输入信息合法。 |
| 后置条件 | 将新的用户信息加入到数据库中。 |
| UserController.IndividualBaseInfoInquiry | 语法 | Public UserInfoVO IndividualBaseInfoInquiry (UserIDVO vo); |
| 前置条件 | 无 |
| 后置条件 | 根据用户ID返回不同种类的用户信息。 |
| UserController.IndividualBaseInfoModification | 语法 | Public ResultMessage IndividualBaseInfoModification(UserIDVO vo); |
| 前置条件 | 程序的当前用户就是传入UserIDVO对应的用户。 |
| 后置条件 | 改变数据库中对应的用户信息。 |
| UserController.IndividualOrderInquiry | 语法 | Public ArrayList<OrderVO> IndividualOrderInquiry(UserIDVO vo); |
| 前置条件 | 客户要求查看个人订单信息。 |
| 后置条件 | 系统返回客户个人订单信息表。 |
| UserController.IndividualHotelInquiry | 语法 | Public ArrayList<HotelVO> IndividualHotelInquiry(UserIDVO vo); |
| 前置条件 | 客户要求查看个人预订过的酒店信息。 |
| 后置条件 | 系统返回客户预订过的酒店信息表。 |
| UserController.IndividualCredictInquiry | 语法 | Public CreditVo IndividualCredictInquiry(UserIDVo vo); |
| 前置条件 | 要求查看个人信用信息。 |
| 后置条件 | 系统返回客户个人信用信息。 |
| UserController.HotelSearch | 语法 | Public ArrayList HotelSearch<HotelVO>(HotelConditionVO vo); |
| 前置条件 | 客户要求输入筛选条件搜索酒店。 |
| 后置条件 | 系统返回酒店列表。 |
| UserController.OrderCreat | 语法 | Public boolean OederCreat(HotelIDVO vo1,OrderVO vo2); |
| 前置条件 | 客户要求填写订单信息。 |
| 后置条件 | 返回预定结果并更新相应酒店的数据库。 |
| UserController.HotelEvaluate | 语法 | Public void HotelEvaluate(HotelEvaluateVO vo); |
| 前置条件 | 客户要求对已执行订单中的酒店进行评价。 |
| 后置条件 | 更新该酒店的信息。 |
| UserController.MemberRegisterApply | 语法 | Public void MemberRegisterApply(HotelIDVO vo1,MemberVO vo2); |
| 前置条件 | 客户要求注册会员。 |
| 后置条件 | 系统发行申请至网站营销人员。 |
| UserController.HotelInformationInquiry | 语法 | Public HotelVO HotelInformationInquiry(HotelIDVO vo); |
| 前置条件 | 要求查看酒店信息。 |
| 后置条件 | 系统返回酒店信息。 |
| UserController.HotelInformationModification | 语法 | Public boolean HotelInformationModification(HotelStuffHotelOperationVO vo); |
| 前置条件 | 酒店管理人员要求修改酒店信息。 |
| 后置条件 | 系统返回修改过后的酒店信息。 |
| UserController.HotelStrategeManage | 语法 | Public boolean HotelStrategeManage(HotelStuffHotelOperationVO vo); |
| 前置条件 | 酒店管理人员要求对酒店策略进行调整。 |
| 后置条件 | 系统更新酒店信息。 |
| UserController.OrderStateUpdate | 语法 | Public void OrderStateUpdate(HotelStuffHotelOperationVO vo); |
| 前置条件 | 客户或酒店管理人员要求更新订单状态或者订单到期后系统自动更新。 |
| 后置条件 | 系统更新客户个人信息和酒店信息。 |
| UserController.OrderScan | 语法 | Public ArrayList<OrderVO> OrderScan(HotelStuffHotelOperationVO vo); |
| 前置条件 | 酒店管理人员要求查看酒店订单信息。 |
| 后置条件 | 系统返回酒店订单信息。 |
| UserController.HotelAdd | 语法 | Public void HotelAdd(HotelVO vo); |
| 前置条件 | 网站管理人员要求增加酒店。 |
| 后置条件 | 系统更新数据库。 |
| UserController.StuffAdd | 语法 | Public void StuffAdd(HotelIDVO vo1,UserIDVO vo2); |
| 前置条件 | 网站管理人员要求增加酒店管理人员。 |
| 后置条件 | 系统更新相应酒店的信息。 |
| UserController.WebsitStrategeCreate | 语法 | Public void WebsiteStrategeCreate(WebStuffWebsiteManagementVO vo); |
| 前置条件 | 网站营销人员要求制定营销策略。 |
| 后置条件 | 系统更新网站数据库。 |
| UserController.AbnormalOrderScan | 语法 | Public ArrayList<OrderVO> AbnormalOrderScan(); |
| 前置条件 | 网站营销人员要求查看异常订单。 |
| 后置条件 | 系统返回当天异常订单信息。 |
| UserController.UserCreditModification | 语法 | Public boolean UserCreditModification(WebStuffWebsiteManagementVO vo); |
| 前置条件 | 网站营销人员要求修改用户信用信息。 |
| 后置条件 | 系统更新数据并通知用户。 |
| UserController.UserInformationInquiry | 语法 | Public UserVO UserInformationInquiry(UserIDVO vo); |
| 前置条件 | 网站管理人员要求查看用户信息。 |
| 后置条件 | 系统返回用户信息。 |
| UserController.UserInformationModification | 语法 | Public boolean UserInformationModification(UserIDVO vo1,UserInfoVO vo2); |
| 前置条件 | 网站管理人员要求修改用户信息。 |
| 后置条件 | 系统返回修改结果并更新数据库。 |
| UserController.WebsiteStuffAdd | 语法 | Public void WebsiteStuffAdd(UserIDVO vo); |
| 前置条件 | 网站管理人员要求增加网站营销人员。 |
| 后置条件 | 系统更新用户和网站数据库。 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| 服务名 | 服务 |
| OrderOnUserBLService.personalOrderScan | 客户查看用户个人订单信心。 |
| OrderOnUserBLService.personalOrderCancel | 客户撤销订单。 |
| OrderOnUserBLService.personalOrderDetail | 客户查看用户订单详情。 |
| OrderOnHotelBLService.hotelOrderScan | 酒店管理人员查看酒店订单列表 |
| OrderOnHotelBLService.hotelOrderDetail | 酒店管理人员查看酒店订单详情。 |
| OrderOnHotelBLService.hotelOrderModify | 酒店管理人员修改酒店订单状态信息。 |
| OrderOnWebBLService.complaintListScan | 网站营销人员查看系统的申诉列表。 |
| OrderOnWebBLService.complaintHandle | 网站营销人员处理申诉请求。 |
| OrderOnWebBLService.abnormalOrderScan | 网站营销人员查看网站异常订单列表。 |
| OrderOnWebBLService.abnormalOrderDetail | 网站营销人员查看异常订单详情。 |
| HotelSearchBLServiceselectCondition | 系统根据用户选择的条件搜索酒店。 |
| HotelSearchBLService.show | 系统显示特定条件的酒店搜索结果。 |
| HotelInfoCheckBLService.selectHotel | 客户查看酒店详细信息。 |
| HotelEvaluateBLService.inputEvaluate | 客户对已执行订单酒店进行评价。 |
| HotelInfoMaintainBLService.inputHotelInfo | 酒店工作人员维护酒店基本信息。 |
| RoomAddBLService.addRoom | 酒店管理人员录入可用客房。 |
| RoomAddBLService. updateRoom | 酒店管理人员更新客房状态。 |
| VipLevelBLService.searchLevel | 用户获得当前积分所对应的用户等级。 |
| VipLevelBLService. changeLevelNeed | 网站营销人员修改用户信用值。 |
| IntegralBLService. getIntegral | 查看当前用户信用值。 |
| IntegralBLService.addIntegral | 系统增加当前用户信用值。 |
| IntegralBLService. subIntegral | 系统减少当前用户信用值。 |
| PromotionHoelChangeBLService.changeBirthCut | 酒店工作人员修改酒店会员生日促销策略 |
| PromotionHoelChangeBLService. changeOverCut | 酒店工作人员修改酒店三间以上优惠策略 |
| PromotionHoelChangeBLService. changeJoinCut | 酒店工作人员修改酒店合作企业客户优惠策略。 |
| PromotionHoelChangeBLService.changeHotelCustomCut | 酒店工作人员自定义酒店促销策略。 |
| PromotionWebChangeBLService.changeLevelCut | 网站营销人员修改网站会员等级优惠。 |
| PromotionValue.getValue | 计算根据促销策略得出的价格。 |
| Customer.GetBaseInfo | 取得客户的基本信息 |
| Customer.SetBaseInfo | 修改客户的基本信息 |
| Customer.GetIndividualOrder | 取得客户的订单信息 |
| Customer.GetIndividualHotel | 取得客户住过的酒店信息 |
| Customer.GetCredit | 取得客户的信用信息 |
| UserManagementDataService.GetLoginInfo | 检验用户登录信息 |
| UserManagementDataService.AddUser | 处理注册事件 |
| HotelStuff.GetBaseInfo | 获取酒店管理人员基本信息 |
| HotelStuff.SetBaseInfo | 修改酒店管理人员基本信息 |
| HotelStuff.GetHotelInfo | 获取酒店管理人员所在酒店的信息 |
| HotelStuff.SetHotelInfo | 修改酒店管理人员所在酒店的信息 |
| HotelStuff.GetStratege | 获取酒店管理人员所在酒店的促销策略 |
| HotelStuff.SetStratege | 修改酒店管理人员所在酒店的促销策略 |
| HotelStuff.OrderScan | 获取酒店管理人员所在酒店的订单列表 |

Customer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的接口（供接口） | | |
| Customer.GetBaseInfo | 语法 | Public CustomerInfoVO GetBaseInfo(UserIDVO vo); |
| 前置条件 | 用户账号存在 |
| 后置条件 | 根据用户的种类返回不同种类的用户基本信息 |
| Customer.SetBaseInfo | 语法 | Public ResultMsg SetBaseInfo(UserIDVO vo1, UserInfoVO vo2); |
| 前置条件 | 被修改用户就是程序的当前用户 |
| 后置条件 | 更新数据库中的用户信息 |
| Customer.GetIndividualOrder | 语法 | Public ArrayList<OrderVO> GetIndividualOrder(UserIDVo vo); |
| 前置条件 | 用户已经登录 |
| 后置条件 | 返回当前用户的订单列表 |
| Customer.GetIndividualHotel | 语法 | Public ArrayList<HotelVO> GetIndividualHotel(UserIDVO vo); |
| 前置条件 | 用户已经登录 |
| 后置条件 | 返回当前用户的酒店信息列表 |
| Customer.GetCredit | 语法 | Public CreditVO GetCredit(UserIDVO vo); |
| 前置条件 | 用户已经登录 |
| 后置条件 | 返回用户信用信息 |
| Customer.ApplyMember | 语法 | Public ResultMsg ApplyMember(UserIDVO vo); |
| 前置条件 | 申请用户为当前登录用户 |
| 后置条件 | 更新用户数据库信息 |
| Customer.CreatOrder | 语法 | Public ResultMsg CreatOrder(OrderVO vo1,HotelIDVO vo2); |
| 前置条件 | 订单信息完整且对应时间酒店存在空房 |
| 后置条件 | 系统将订单信息记录入用户数据库和酒店数据库 |
| Customer.CancelOrder | 语法 | Public ResultMsg CancleOrder(OrderVO vo1,HotelIDVO vo2); |
| 前置条件 | 备操作订单状态为未执行 |
| 后置条件 | 更改订单数据库和用户信息数据库 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| 服务名 | 服务 |
| UserManagementDataService.GetUserBaseInfo | 从数据库得到用户的个人信息 |
| UserManagementDataService.SetUserBaseInfo | 更新数据库中的个人信息 |
| CustomerManagementDataService.GetCustomerOrders | 获得用户的订单信息 |
| CustomerManagementDataService.GetCustomerrHotel | 获得用户预订过的酒店信息 |
| CustomerManagementDataService.AddMembers | 处理会员注册事件 |
| CustomerManagementDataService.UpDateCustomerOrders | 更新数据库中的用户订单状态信息 |
| CustomerManagementDataService.SetCustomerHotel | 更新数据库中用户酒店信息 |

HotelStuff的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的接口（供接口） | | |
| HotelStuff.GetBaseInfo | 语法 | Public HotelStuffInfoVO GetBaseInfo(UserIDVO vo); |
| 前置条件 | 已经创建一个酒店管理人员对象 |
| 后置条件 | 返回对应输入参数中ID的酒店管理人员基本信息 |
| HotelStuff.SetBaseInfo | 语法 | Public ResultMsg SetBaseInfo(UserIDVO vo1,HotelStuffInfoVO vo2); |
| 前置条件 | 已经创建一个酒店管理人员对象，被修改对象为程序当前用户 |
| 后置条件 | 更新数据库中的酒店管理人员信息 |
| HotelStuff.GetHotelInfo | 语法 | Public HotelVO GetHotelInfo(HotelIDVO vo); |
| 前置条件 | 已经创建一个酒店对象 |
| 后置条件 | 返回对应酒店的信息 |
| HotelStuff.SetHotelInfo | 语法 | Public ResultMsg SetHotelInfo(HotelIDVO vo1,HotelVO vo2); |
| 前置条件 | 已经创建一个酒店对象，并且程序当前用户为该酒店对象的管理人员 |
| 后置条件 | 更新数据库中的酒店信息 |
| HotelStuff.GetStratege | 语法 | Public ArrayList<HotelStrategeVO >GetStratege(HotelIDVO vo); |
| 前置条件 | 已经创建一个酒店对象 |
| 后置条件 | 返回对应酒店的促销策略 |
| HotelStuff.SetStratege | 语法 | Public ResultMsg SetStratege(HotelIDVO vo1,HotelStrategeVO vo2); |
| 前置条件 | 已经创建一个酒店对象，并且程序当前用户为该酒店对象的管理人员。 |
| 后置条件 | 更新数据库中的酒店促销策略信息 |
| HotelStuff.OrderScan | 语法 | Public ArrayList<OrderVO> OrderScan(HotelIDVO vo); |
| 前置条件 | 已经创建一个酒店对象，并且程序当前用户为该酒店的管理人员 |
| 后置条件 | 返回酒店的订单信息列表 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| 服务名 | 服务内容 |
| UserManagementDataService.GetUserBaseInfo | 获取用户个人信息 |
| UserManagementDataService.SetUserBaseInfo | 修改用户个人信息 |

WebStuff的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的接口（供接口） | | |
| WebStuff.GetBaseInfo | 语法 | Public WebStuffInfoVO GetBaseInfo(UserIDVO vo); |
| 前置条件 | 已经创建网站营销人员对象并且被查询对象就是程序的当前用户 |
| 后置条件 | 返回网站营销人员的基本信息 |
| WebStuff.SetBaseInfo | 语法 | Public ResultMsg SetBaseInfo(UserIDVO vo1,WebStuffInfoVO vo2); |
| 前置条件 | 已经创建网站营销人员对象并且被修改对象就是程序的当前用户 |
| 后置条件 | 更新数据库中网站营销人员的个人信息 |
| WebStuff.GetUserCredit | 语法 | Public CredictVO GetUserCredit(UserIDVO vo); |
| 前置条件 | 已经创建网站营销人员对象并且程序的当前用户就是该网站营销人员对象 |
| 后置条件 | 返回被查询用户的信用值 |
| WebStuff.SetUserCredict | 语法 | Public ResultMsg SetUserCredit(UserIDVO vo,int n); |
| 前置条件 | 已经创建网站营销人员对象程序的当前用户就是该网站营销人员对象并且信用值输入合法。 |
| 后置条件 | 更新数据库中的用户信用值和信用记录信息。 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| 服务名 | 服务 |
| UserManagementDataService.GetUserBaseInfo | 获得用户基本信息 |
| UserManagementDataService.SetUserBaseInfo | 修改用户基本信息 |
| CustomerManagementDataService.SetCustomerCredit | 修改用户信用信息 |

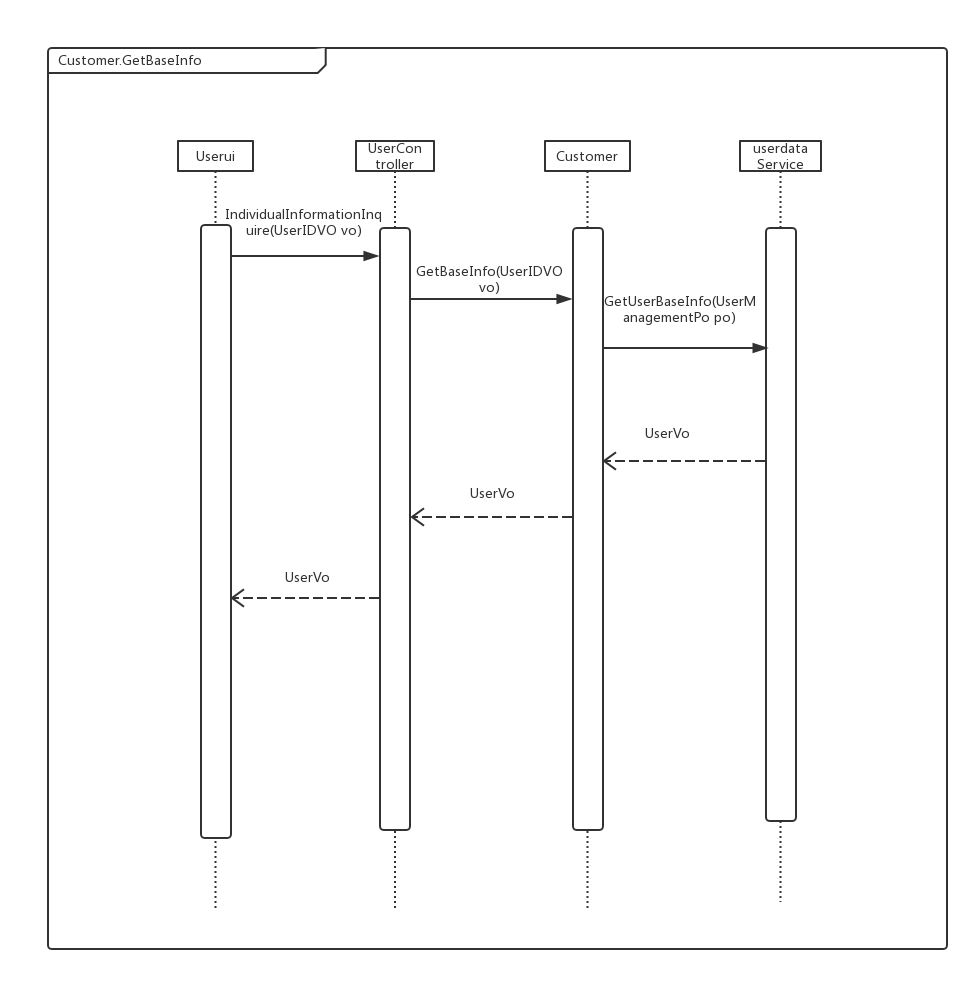
WebManager接口规范

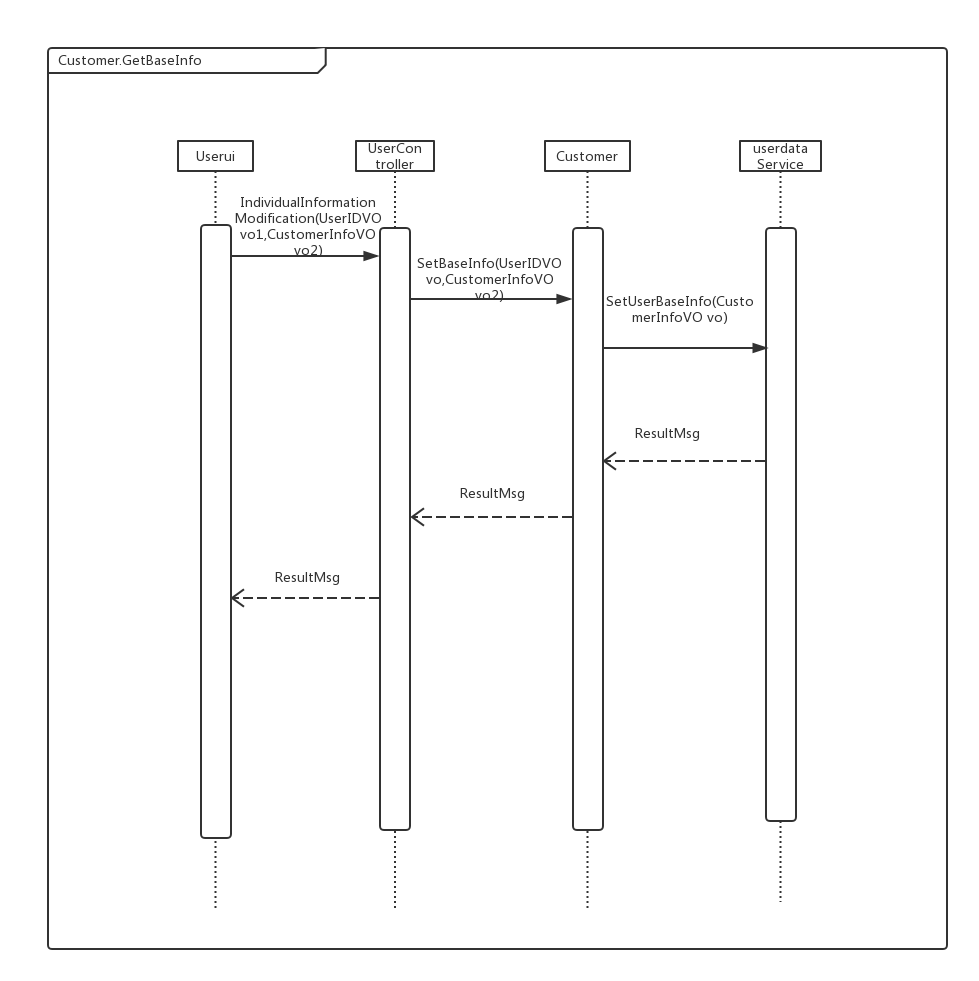
|  |  |  |
| --- | --- | --- |
| 提供的接口（供接口） | | |
| WebManager.GetBaseInfo | 语法 | Public WebManagerInfoVO GetBaseInfo(UserIDVO vo); |
| 前置条件 | 输入的用户ID对应的网站管理人员对象已经创建 |
| 后置条件 | 返回对应ID的网站管理人员信息 |
| WebManager.SetBaseInfo | 语法 | Public ResultMsg SetBaseInfo(UserIDVO vo1,WebManagerInfoVO vo2); |
| 前置条件 | 被修改的网站管理人员实例已经存在 |
| 后置条件 | 更新网站管理人员数据库信息 |
| WebManager.GetUserInfo | 语法 | Public UserInfo GetUserInfo(UserIDVO vo); |
| 前置条件 | 网站管理人员已经登录验证，并且是程序的当前用户 |
| 后置条件 | 返回对应用户的信息 |
| WebManager.SetUserInfo | 语法 | Public ResultMsg SetUserInfo(UserIDVO vo1,UserInfoVO vo2); |
| 前置条件 | 网站管理人员已经登录验证，并且是程序的当前用户 |
| 后置条件 | 更新数据库中的用户信息 |
| WebManager.GetHotelInfo | 语法 | Public HotelVO GetHotelInfo(HotelIDVO vo); |
| 前置条件 | 网站管理人员已经登录验证，且被查询的酒店对象已经存在 |
| 后置条件 | 返回对应酒店的信息 |
| WebManager.SetHotelInfo | 语法 | Public ResultMsg SetHotelInfo (HotelIDVO vo1,HotelVO vo2); |
| 前置条件 | 网站管理人员已经登录验证，且被修改的酒店对象已经存在 |
| 后置条件 | 更新酒店数据库中的信息 |
| WebManager.AddHotel | 语法 | Public ResultMsg AddHotel(HotelVO vo); |
| 前置条件 | 已经创建一个酒店对象 |
| 后置条件 | 更新酒店数据库，加入新对象信息。 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| 服务名 | 服务 |
| UserManagementDataService.GetUserBaseInfo | 获取网站管理人员的信息 |
| UserManagementDataService.SetUserBaseInfo | 修改网站管理人员的信息 |

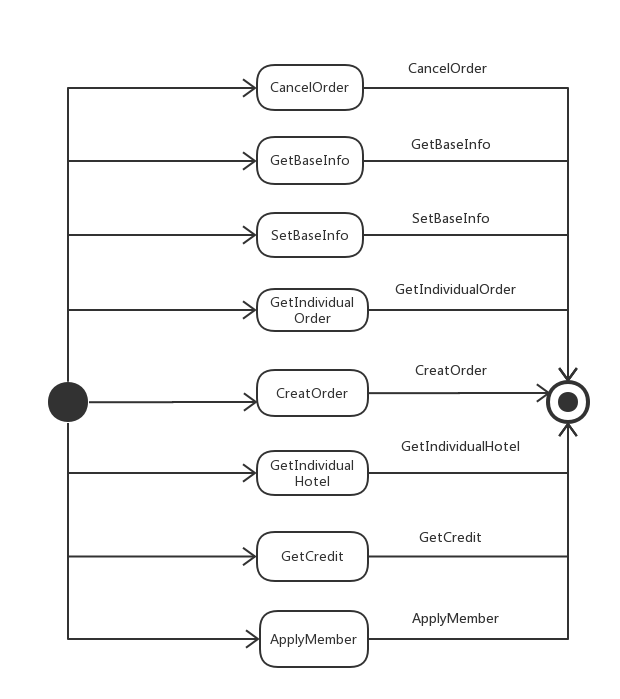
1. 业务逻辑层的动态模型

在用户模块中，查看个人信息、修改个人信息的顺序图如下图所示：





下图描述了Customer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。



5.业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### Hotel模块

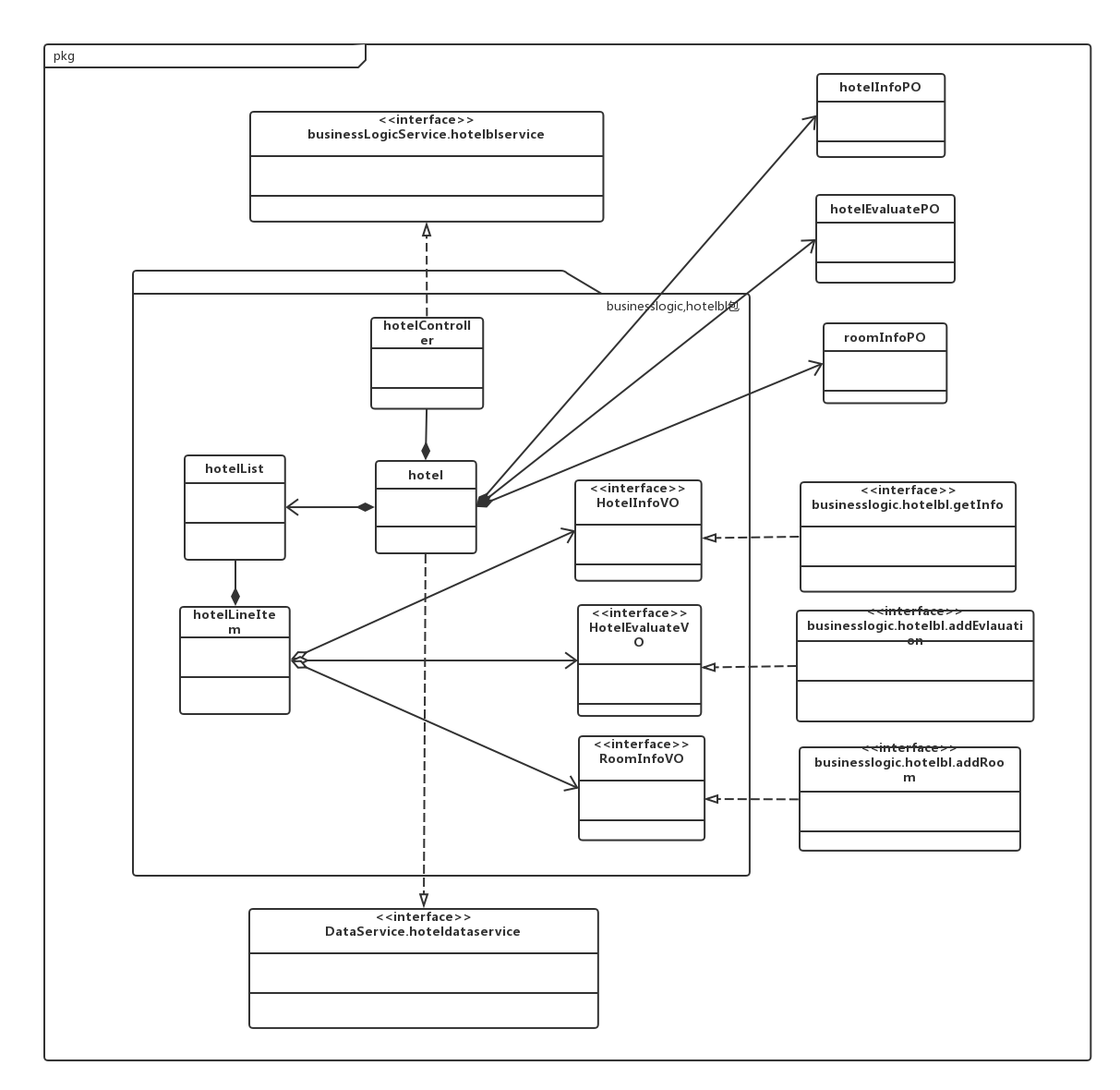
1、模块概述：

**hotel**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。

**hotel**模块的职责和接口参见软件系统结构描述文档内对该模块的描述。

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。在展示层和业务逻辑层之间添加HotelService接口。在业务逻辑层和数据层之间添加HotelDataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加HotelController，这样HotelController会将对酒店管理的业务逻辑处理委托给Hotel对象。HotelInfoPO、HotelEvaluatePO、RoomInfoPO是作为酒店信息的持久性对象被添加到设计模型中去的。



Hotelbl模块各个类的职责如表4-1-1-1所示。

**hotelbl模块各个类的职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| HotelController | 负责对应于酒店管理所需要的服务 |
| Hotel | 酒店管理的领域模型对象，拥有酒店的信息，可以帮助完成酒店管理界面所需要的服务 |

3、模块内部类的接口规范

Controller和Hotel的接口规范如下所示

**Controller的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelSearchController.selectCondition | 语法 | public HotelMsg selectCondition(HotelInfoVO hotelConditionVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的searchHotel方法 |
| HotelSearchController.show | 语法 | public HotelMsg show(HotelInfoVO hotelListInfoVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的show方法 |
| HotelInfoCheckController.checkHotelInfo | 语法 | public HotelMsg checkHotelInfo(HotelInfoVO selectHotelVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的checkHotelInfo方法，系统返回酒店信息 |
| HotelReservedCheckController.checkReserved | 语法 | public HotelMsg checkReserved( HotelInfoVO reservedHotelVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的checkReserved方法，系统返回判断结果 |
| HotelReserveController.reserveHotel | 语法 | public HotelMsg reserveHotel(HotelInfoVO reserveHotelVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的reserveHotel方法 |
| HotelReserveController.createOrder | 语法 | public HotelMsg createOrder(HotelInfoVO orderInfoVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的createOrder方法，系统生成表单存储 |
| HotelEvaluateController.inputEvaluate | 语法 | public HotelMsg inputEvaluate(HotelEvaluateVO evaluateInfoVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的inputEvaluate方法，系统生成表单存储 |
| HotelEvaluateController.checkOrder | 语法 | public HotelMsg checkOrder(HotelEvaluateVO orderListVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的checkOrder方法，系统生成表单存储 |
| HotelEvaluateController.completeEvaluation | 语法 | public HotelMsg completeEvaluation( HotelEvaluateVO completeEvaluationVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的completeEvaluation方法，系统生成表单存储 |
| HotelInfoMaintainController.inputHotelInfo | 语法 | public HotelMsg inputHotelInfo(HotelInfoVO hotelInfoVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的inputHotelInfo方法，系统生成表单存储 |
| HotelInfoMaintainController.submitHotelInfo | 语法 | public HotelMsg submitInfo(HotelInfoVO hotelInfoVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的submitHotelInfo方法，系统生成表单存储 |
| RoomAddController.addRoom | 语法 | public HotelMsg addRoom(RoomInfoVO roomInfoVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的addRoom方法，系统生成表单存储 |
| RoomAddController.updateRoom | 语法 | public HotelMsg updateRoom(RoomInfoVO roomInfoVO) |
| 前置条件 | 已经创建一个hotel领域对象，并且输入符合语法规则 |
| 后置条件 | 调用Hotel类的updateRoom方法，系统生成表单存储 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| HotelSearch.createSearch (HotelInfoVO vo) | 生成搜索信息 |
| HotelInfoCheck.getInfo(HotelInfoVO vo) | 获取酒店信息 |
| HotelReservedCheck.getReservedInfo(HotelInfoVO vo) | 获取已预订酒店信息 |
| HotelReserve.addReservation(HotelInfoVO vo) | 生成预定 |
| HotelReserve.createOrder(HotelInfoVO vo) | 生成订单 |
| HotelEvaluate.addEvaluation(HotelEvaluateVO vo) | 添加评论信息 |
| HotelInfoMaintain.correctHotelInfo(HotelInfoVO vo) | 修改酒店信息 |
| RoomAdd.addRoom(RoomInfoVO vo) | 添加客房信息 |

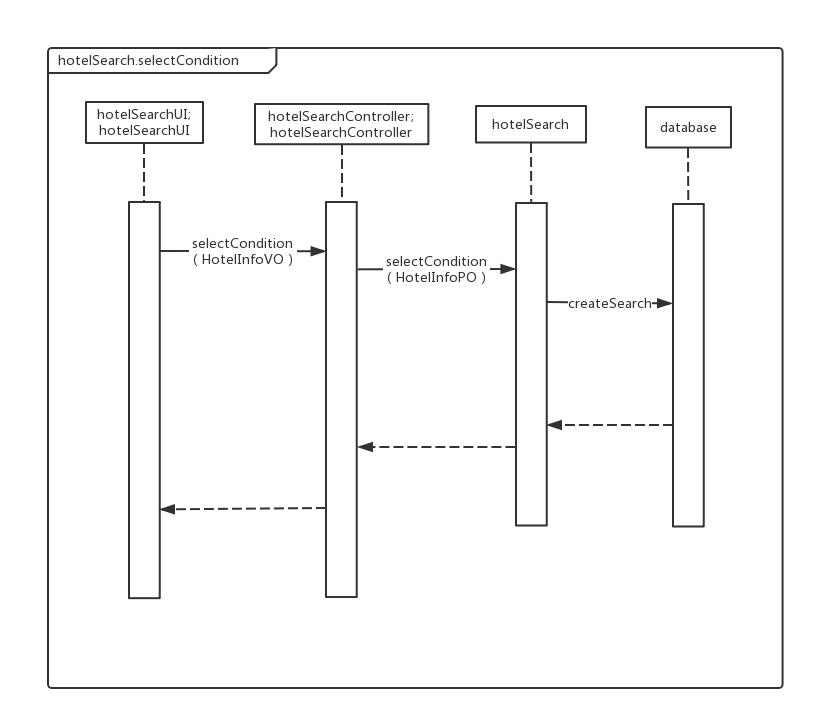
**hotelbl模块的接口规范**

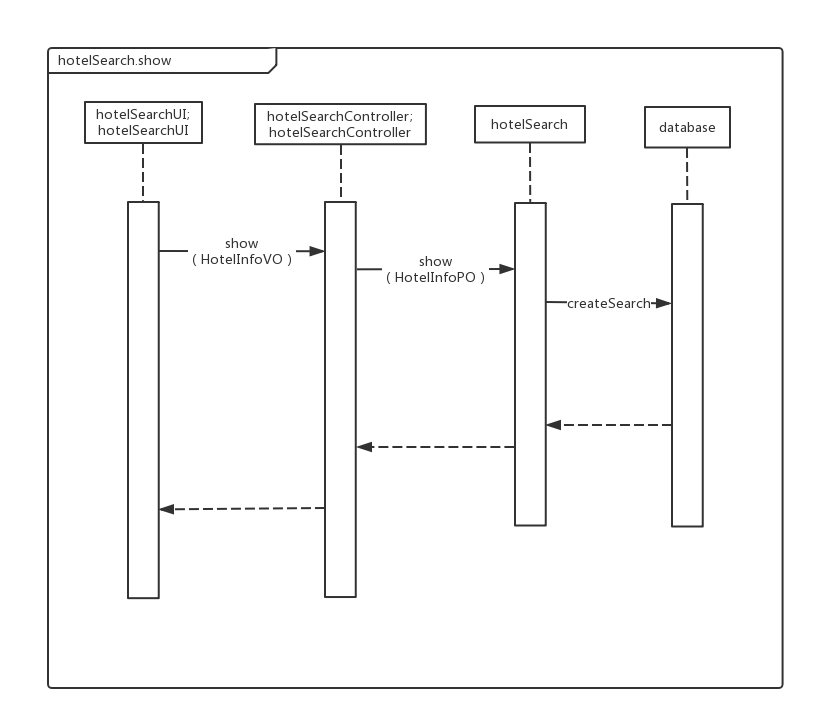
|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelSearch.selectCondition | 语法 | public HotelMsg selectCondition(HotelInfoVO hotelConditionVO) |
| 前置条件 | 客户选择酒店搜索条件，选择搜索酒店 |
| 后置条件 | 系统检查酒店条件，反馈检查结果 |
| HotelSearch.show | 语法 | public HotelMsg show(HotelInfoVO hotelListInfoVO) |
| 前置条件 | 酒店搜索条件符合 |
| 后置条件 | 系统反馈搜索结果 |
| HotelInfoCheck.checkHotelInfo | 语法 | public HotelMsg checkHotelInfo(HotelInfoVO selectHotelVO) |
| 前置条件 | 客户选择酒店，请求查看酒店信息 |
| 后置条件 | 系统反馈酒店信息 |
| HotelReservedCheck.checkReserved | 语法 | public HotelMsg checkReserved( HotelInfoVO reservedHotelVO) |
| 前置条件 | 客户请求查看预订过的酒店 |
| 后置条件 | 系统检查订单，反馈检查结果 |
| HotelReserve.reserveHotel | 语法 | public HotelMsg reserveHotel(HotelInfoVO reserveHotelVO) |
| 前置条件 | 客户选择酒店，提交预定请求 |
| 后置条件 | 系统检查客户信息与酒店客房信息，反馈检查结果 |
| HotelReserve.createOrder | 语法 | public HotelMsg createOrder(HotelInfoVO orderInfoVO) |
| 前置条件 | 检查结果符合要求 |
| 后置条件 | 系统生成预定订单，并反馈预定成功提示 |
| HotelEvaluate.inputEvaluate | 语法 | public HotelMsg inputEvaluate(HotelEvaluateVO evaluateInfoVO) |
| 前置条件 | 客户请求对酒店进行评价，并输入评价信息 |
| 后置条件 | 系统检查评价信息格式，反馈检查结果 |
| HotelEvaluate.checkOrder | 语法 | public HotelMsg checkOrder(HotelEvaluateVO orderListVO) |
| 前置条件 | 评价信息符合格式 |
| 后置条件 | 系统检查酒店是否在已执行订单中，并反馈检查结果 |
| HotelEvaluate.completeEvaluation | 语法 | public HotelMsg completeEvaluation( HotelEvaluateVO completeEvaluationVO) |
| 前置条件 | 检查结果酒店在已执行订单中 |
| 后置条件 | 系统更新评价信息，并提示评价成功 |
| HotelInfoMaintain.inputHotelInfo | 语法 | public HotelMsg inputHotelInfo(HotelInfoVO hotelInfoVO) |
| 前置条件 | 酒店工作人员输入维护酒店基本信息 |
| 后置条件 | 系统检查信息格式，并反馈检查结果 |
| HotelInfoMaintain.submitInfo | 语法 | public HotelMsg submitInfo(HotelInfoVO hotelInfoVO) |
| 前置条件 | 酒店基本信息符合格式 |
| 后置条件 | 系统更新信息，并反馈维护成功 |
| RoomAdd.addRoom | 语法 | public HotelMsg addRoom(RoomInfoVO roomInfoVO) |
| 前置条件 | 酒店工作人员添加可用客房 |
| 后置条件 | 系统检查客房信息是否符合要求，并反馈检查结果 |
| RoomAdd.updateRoom | 语法 | public HotelMsg updateRoom(RoomInfoVO roomInfoVO) |
| 前置条件 | 客房信息符合要求 |
| 后置条件 | 系统更新客房信息，并提示操作完成 |

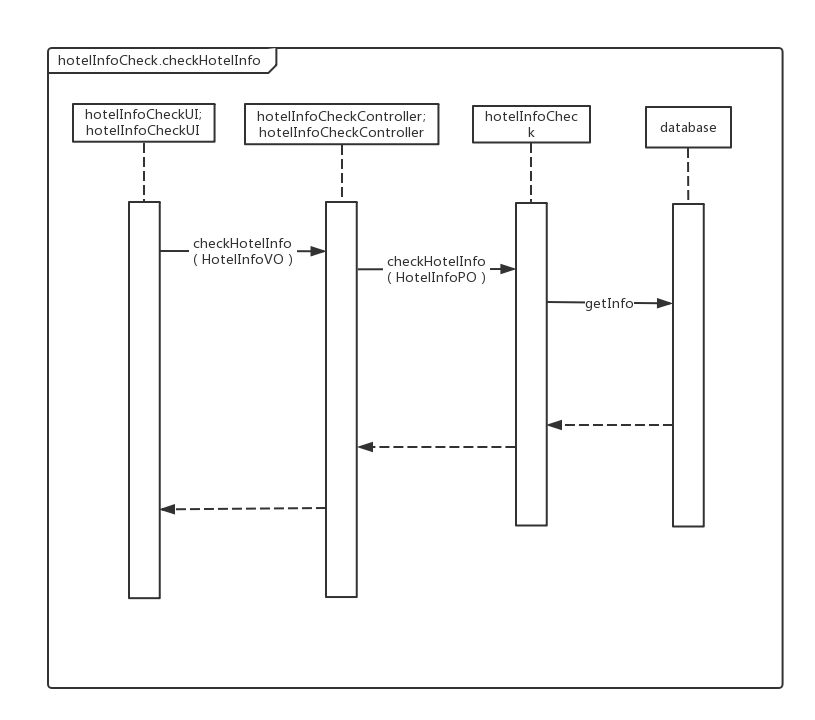
|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| HotelInfoDataService.insert(HotelInfoPO po) | 在数据库中新增一个酒店信息 |
| HotelInfoDataService.delete(HotelInfoPO po) | 在数据库中删除一个酒店信息 |
| HotelInfoDataService.update(HotelInfoPO po) | 在数据库中更新一个酒店信息 |
| HotelInfoDataService.find(HotelInfoPO po) | 在数据库中查找一个酒店信息 |
| HotelEvaluateDataService.insert(HotelEvaluatePO po) | 在数据库中新增一个酒店评价信息 |
| HotelEvaluateDataService.delete(HotelEvaluatePO po) | 在数据库中删除一个酒店评价信息 |
| HotelEvaluateDataService.find(HotelEvaluatePO po) | 在数据库中查找一个酒店评价信息 |
| RoomInfoDataService.insert(RoomInfoPO po) | 在数据库中新增一个可用客房信息 |
| RoomInfoDataService.delete(RoomInfoPO po) | 在数据库中删除一个可用客房信息 |
| RoomInfoDataService.update(RoomInfoPO po) | 在数据库中更新一个可用客房信息 |
| RoomInfoDataService.find(RoomInfoPO po) | 在数据库中查找一个可用客房信息 |

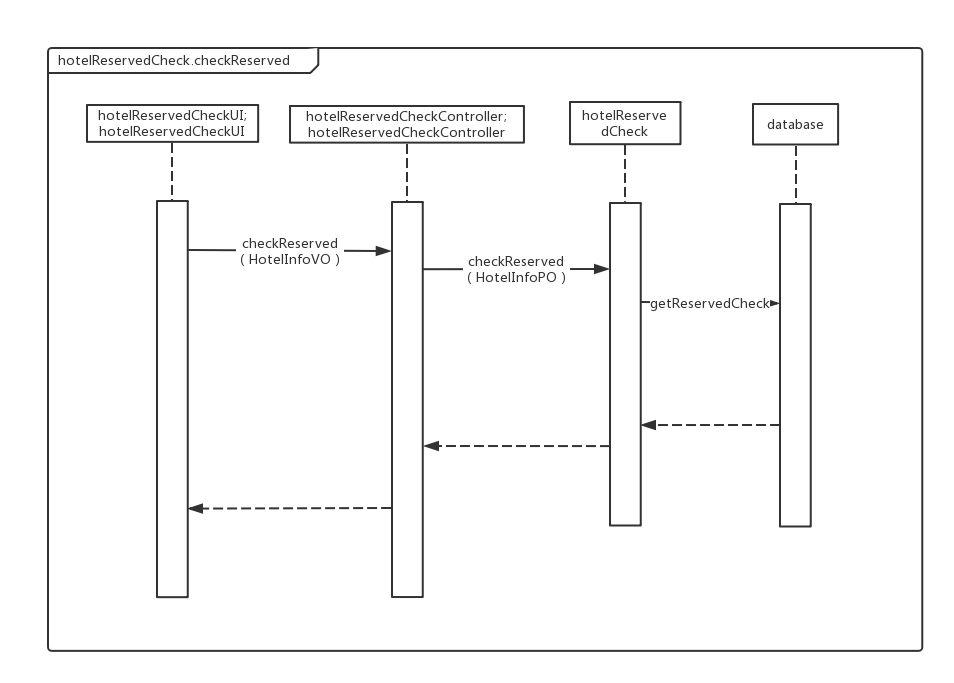
4、业务逻辑层的动态模型

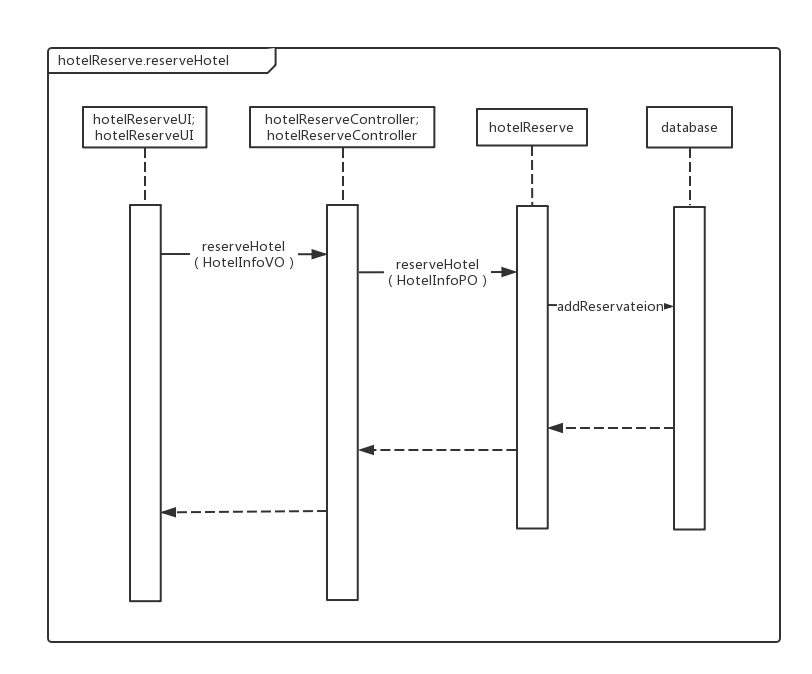
在酒店信息管理中，业务分为搜索酒店、查看酒店、预定酒店、评价酒店、维护酒店基本信息、添加客房，其顺序图如下图所示：

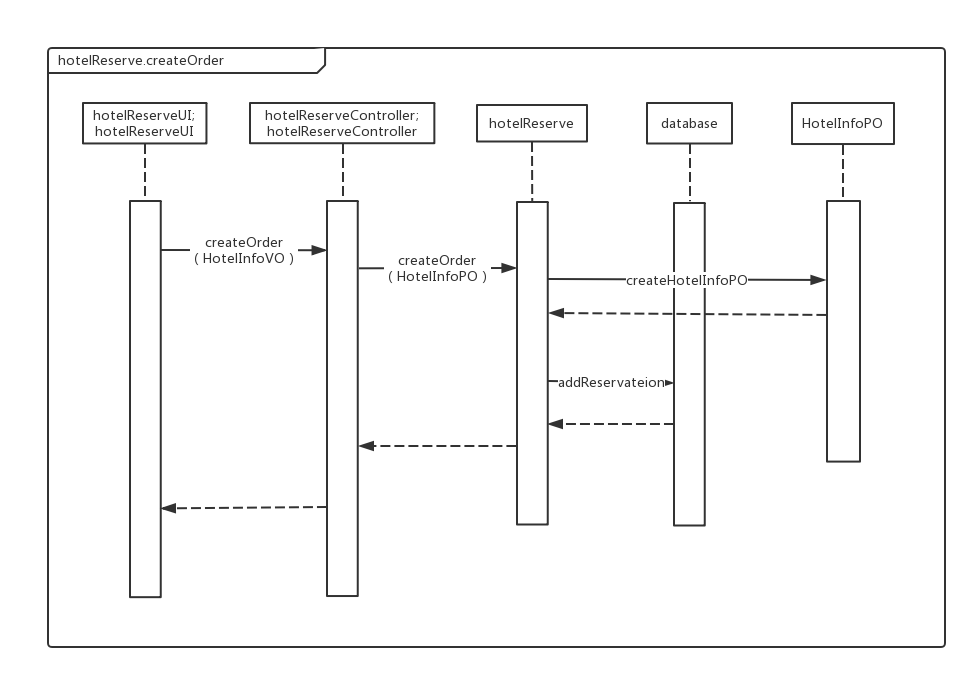


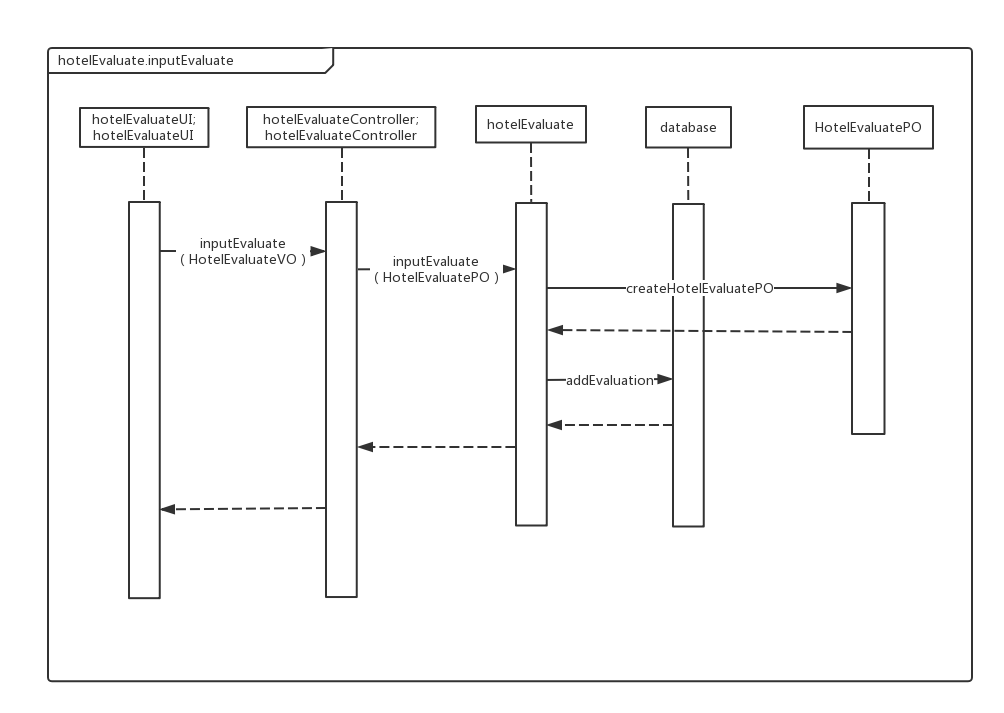


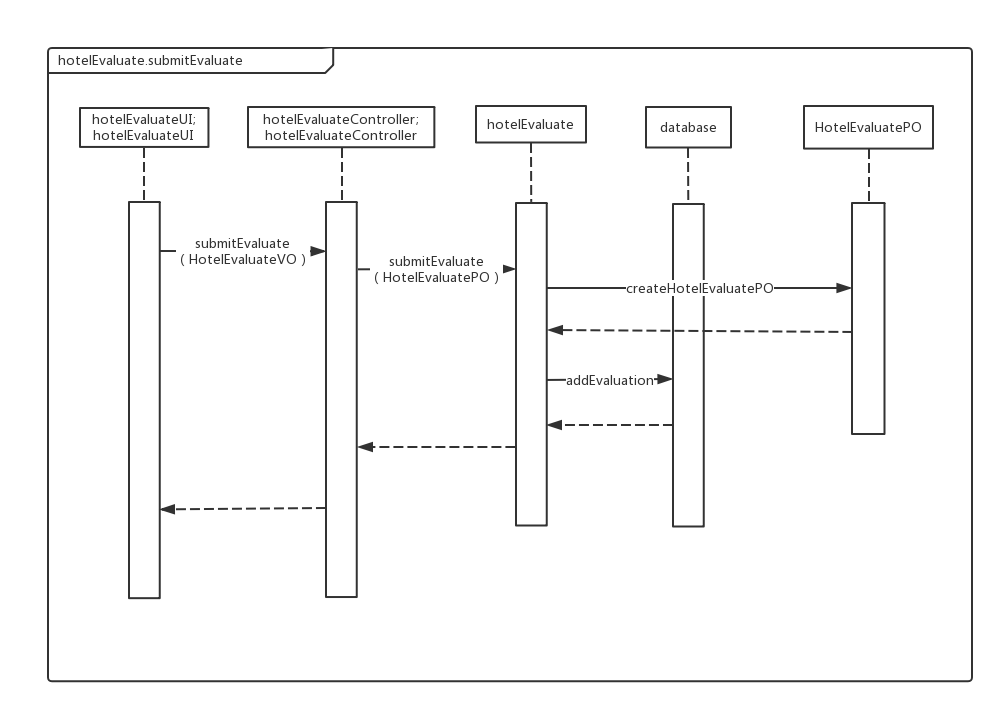


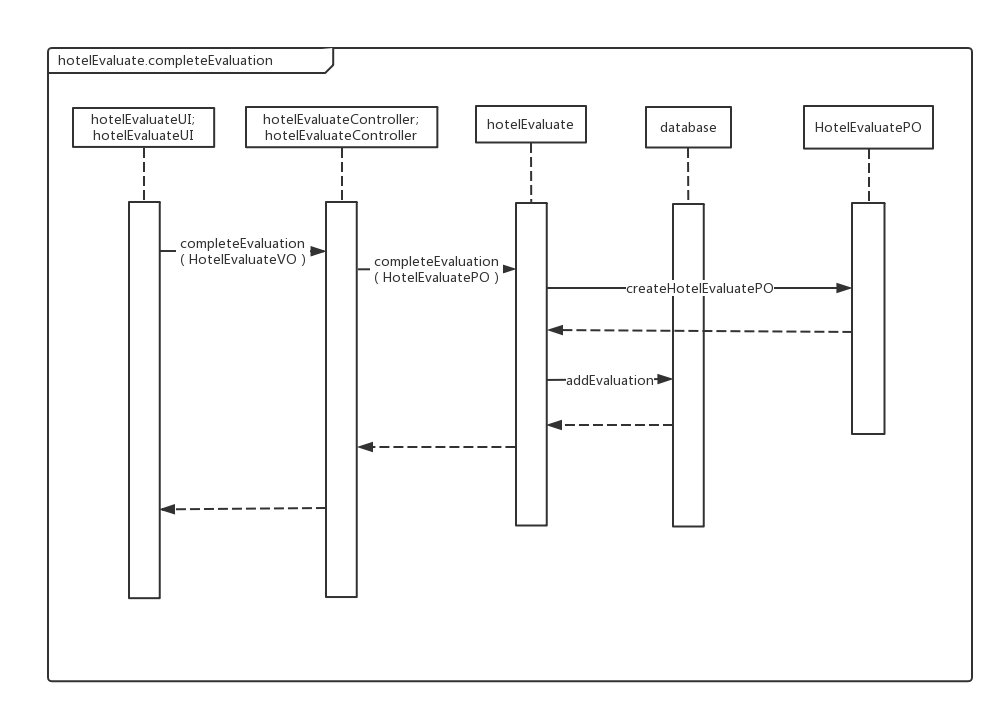


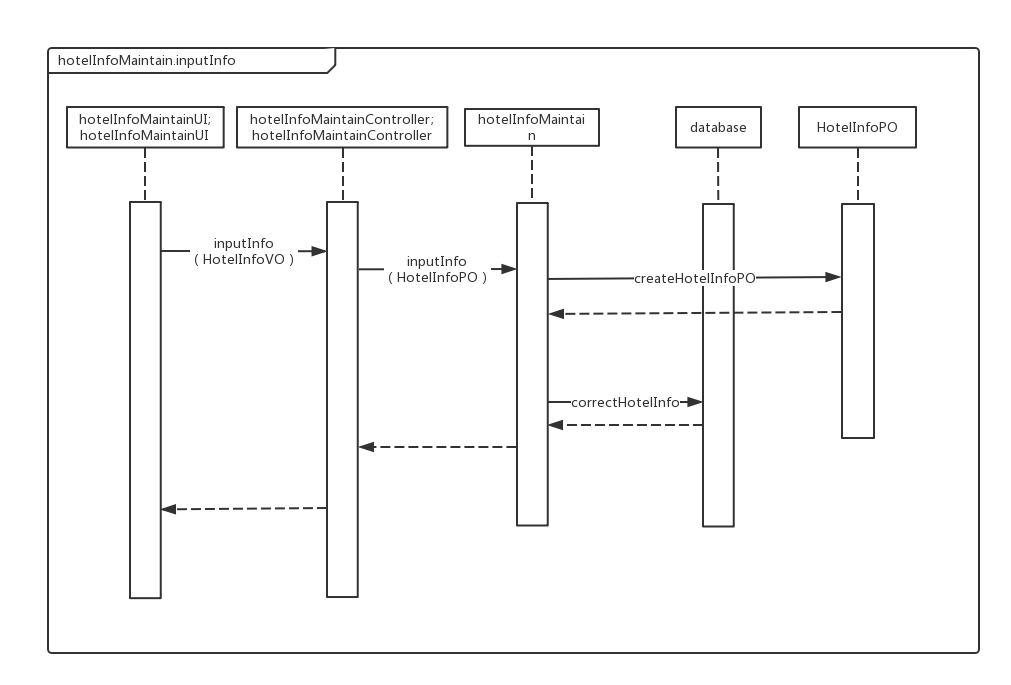


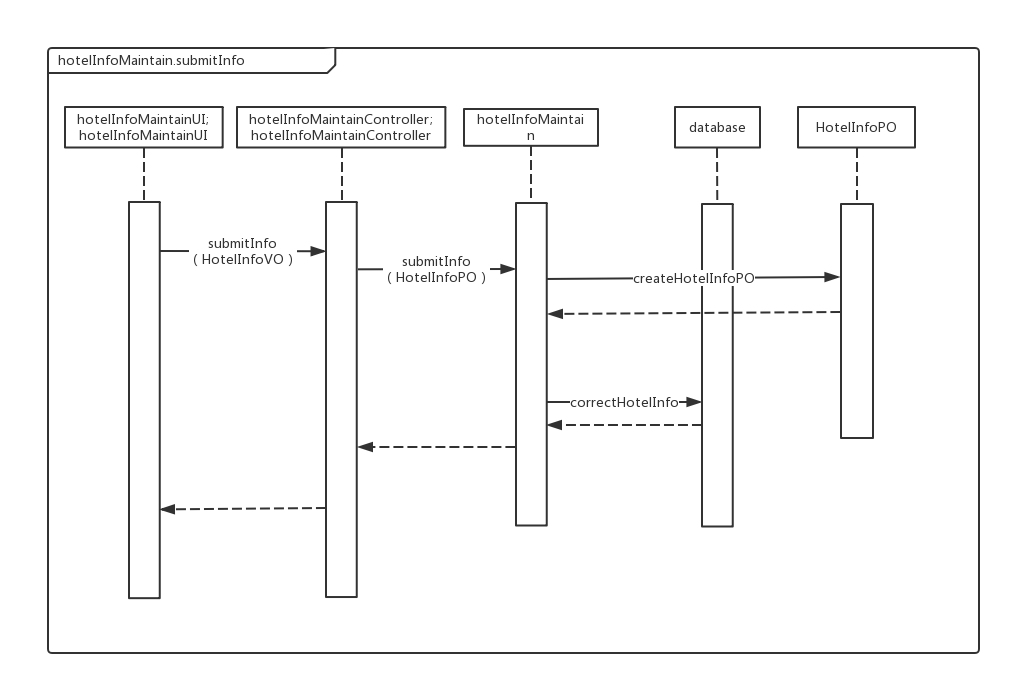


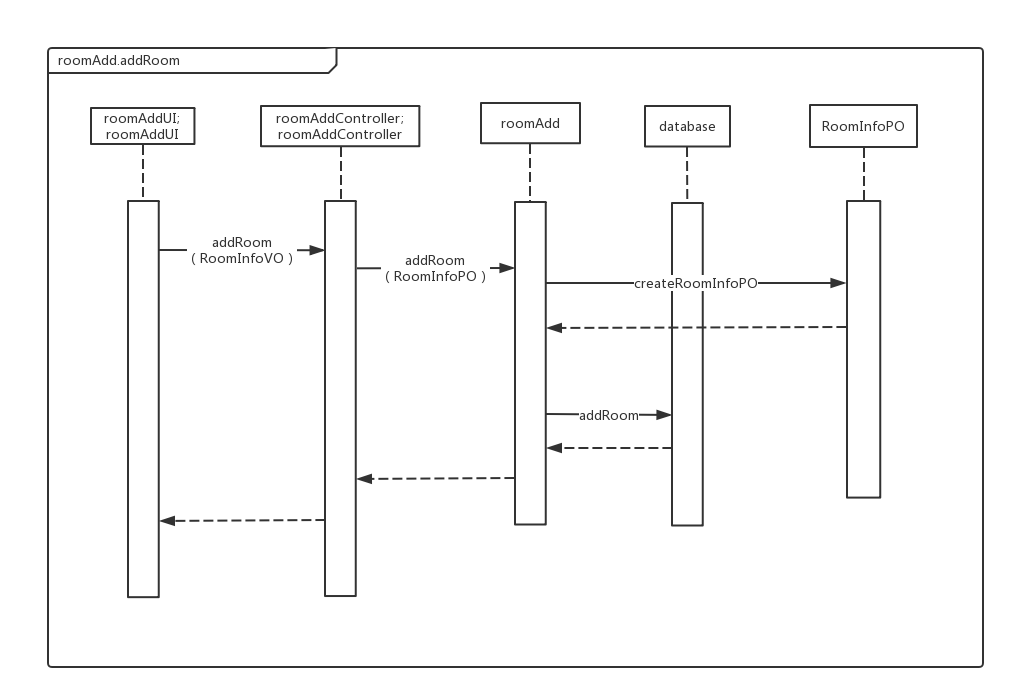


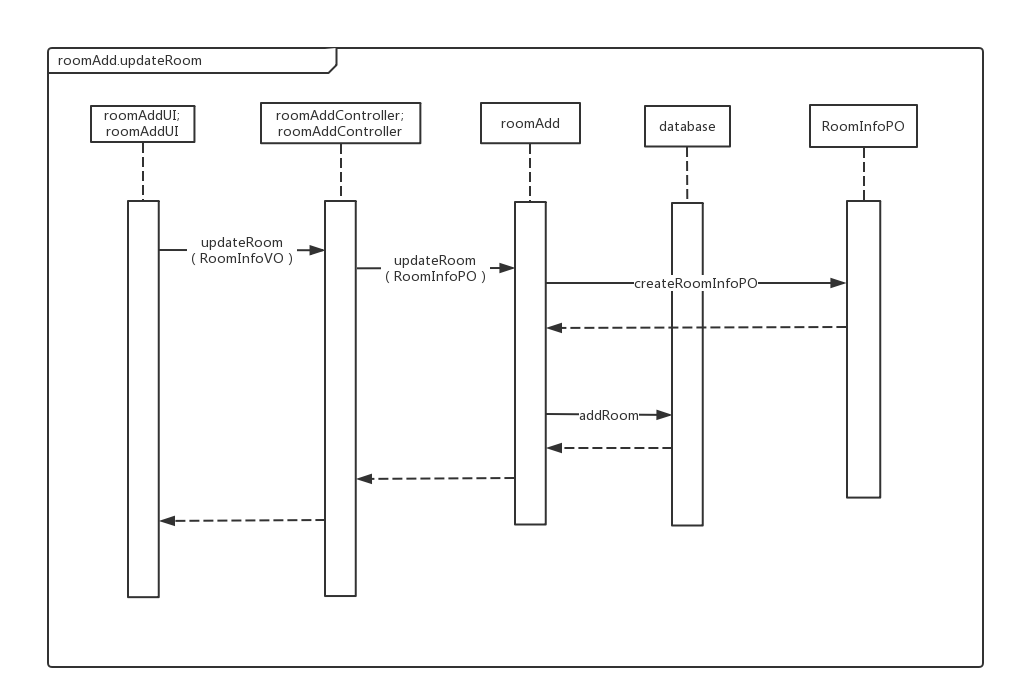




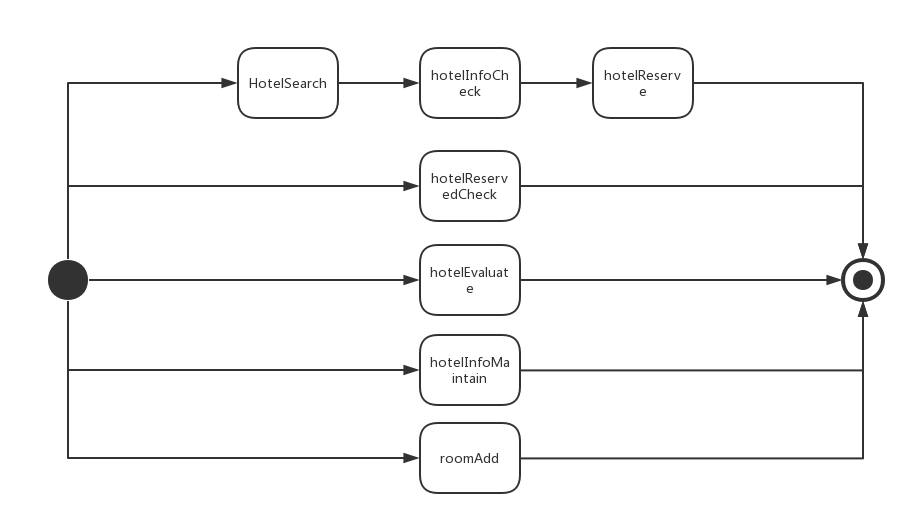








下图描述了hotel对象的生存期间的状态序列、引起转移的事件，以及因状态转移而伴随的动作。



5.业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### Vip模块

1、模块概述

Vip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。

Vip模块的职责和接口参见软件系统结构描述文档内对该模块的描述。

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。在展示层和业务逻辑层之间添加VipBLService接口。在业务逻辑层和数据层之间添加VipDataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加VipController，这样VipController会将对会员信息管理的业务逻辑处理委托给Vip对象。VipPO是作为账户信息的持久性对象被添加到设计模型中去的。

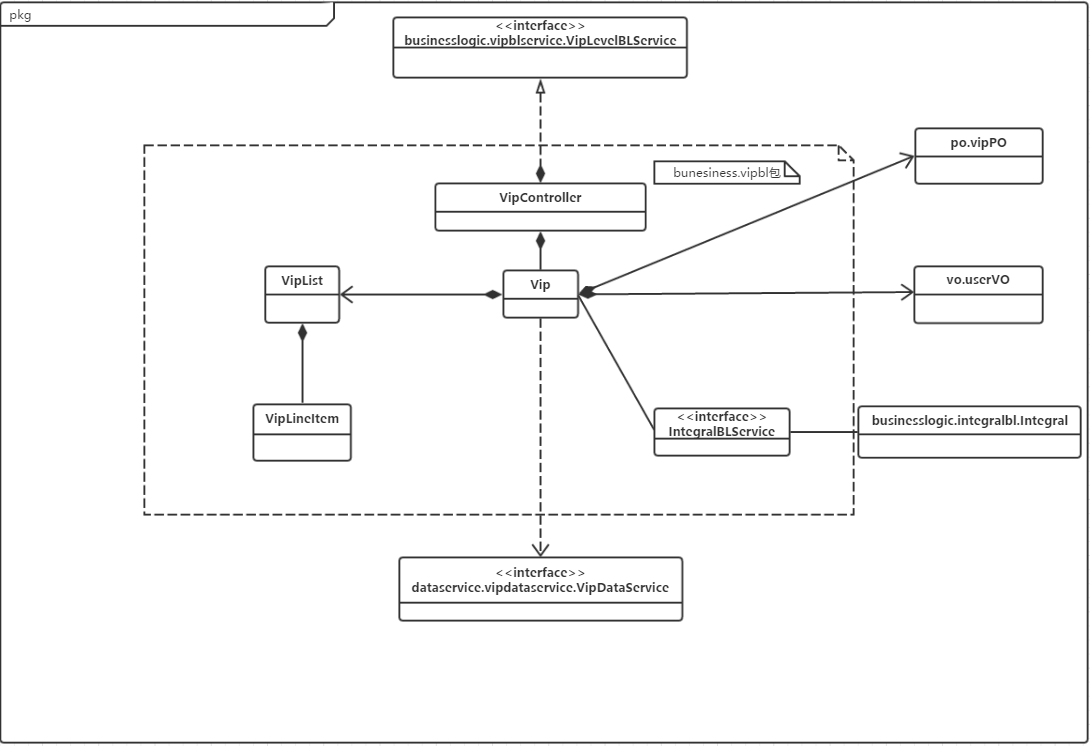


表4.2.4.1 vipbl模块各个类的设计

vipbl模块各个类的职责如表4.2.4.2所示。

|  |  |
| --- | --- |
| 模块 | 职责 |
| VipControl | 负责实现会员部分所需要的服务 |
| Vip | 会员的领域模型对象，拥有会员等级划分的积分要求信息，可以解决用户当前信用值对应的会员等级问题 |
| VipList | 负责保存会员等级对应信用值的相关信息 |
| VipLineItem | 会员等级对应信用值的条目信息 |

3、模块内部类的接口规范

**VipControl类的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| VipControl.searchLevelNeed | 语法 | public ArrayList<vipVO> searchLevelNeed() |
| 前置条件 | 已经创建一个vip领域对象，且输入格式符合规范 |
| 后置条件 | 调用vip类的searchLevelNeed方法，系统自动返回返回各个会员等级需要的信用值 |
| VipControl.searchLevel | 语法 | public int searchLevel(userVO user) |
| 前置条件 | 已经创建一个vip领域对象，且输入格式符合规范 |
| 后置条件 | 调用vip类的searchLevel方法，系统自动返回当前用户对应的会员等级 |
| VipControl.changeLevelNeed | 语法 | public void changeLevelNeed(vipVO vip) |
| 前置条件 | 已经创建一个vip领域对象，且输入格式符合规范 |
| 后置条件 | 调用vip类的changeLevelNeed方法，系统自动修改会员等级要求，并且持久化更新vipVO |
| VipControl. addLevel | 语法 | public void addLevel (vipVO vip) |
| 前置条件 | 已经创建一个vip领域对象，且输入格式符合规范 |
| 后置条件 | 调用vip类的addLevel方法，系统自动添加一个会员等级，并且持久化更新vipVO |
| VipControl. subLevel | 语法 | public void subLevel (vipVO vip) |
| 前置条件 | 已经创建一个vip领域对象，且输入格式符合规范 |
| 后置条件 | 调用vip类的subLevel方法，系统自动删除一个会员等级，并且持久化更新vipVO |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| Vip.searchLevelNeed | 获得会员等级和信用值的对应关系 | |
| Vip.searchLevel | 获得对应的会员等级 | |
| Vip.changeLevelNeed | 修改会员等级和信用值至今的对应关系 | |
| Vip.addLevel | 增加一个会员等级 | |
| Vip.subLevel | 删除一个会员等级 | |

**Vipbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Vip.searchLevelNeed | 语法 | public ArrayList<vipVO> searchLevelNeed() |
| 前置条件 | 用户要求查看会员等级需要的信用值 |
| 后置条件 | 系统返回各个会员等级需要的信用值 |
| Vip.searchLevel | 语法 | public int searchLevel(userVO user) |
| 前置条件 | 用户获得当前积分对应的会员等级 |
| 后置条件 | 系统根据用户的信息，反馈当前用户的会员等级 |
| Vip.changeLevelNeed | 语法 | public void changeLevelNeed(vipVO vip) |
| 前置条件 | 用户修改相应会员等级需要的信用值 |
| 后置条件 | 系统将当前等级需要的信用值修改为integral |
| Vip.addLevel | 语法 | public void addLevel (vipVO vip) |
| 前置条件 | 用户增加相应会员等级需要的信用值 |
| 后置条件 | 系统将当前增加会员等级 |
| Vip.subLevel | 语法 | public void subLevel (vipVO vip) |
| 前置条件 | 用户删除相应会员等级需要的信用值 |
| 后置条件 | 系统删除一个会员等级 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| 服务名 | 服务 |
| Integral.getIntegral | 获取对应客户的信用值 |
| VipDataService.insert | 添加一个持久化的会员信息 |
| VipDataService.delete | 删除一个持久化的会员信息 |
| VipDataService.update | 更新一个持久化的会员信息 |
| VipDataService.findByLevel | 按照等级找到一个持久化的会员信息 |
| VipDataService.Show | 展示所有的持久化的会员信息 |

4.业务逻辑层的动态模型

在会员业务中，分为会员等级需要的信用值查看，用户当前会员等级，修改会员等级需要的信用值功能。图4.2.4.3表示了得到会员等级和信用值对应关系的相关对象之间的协作。

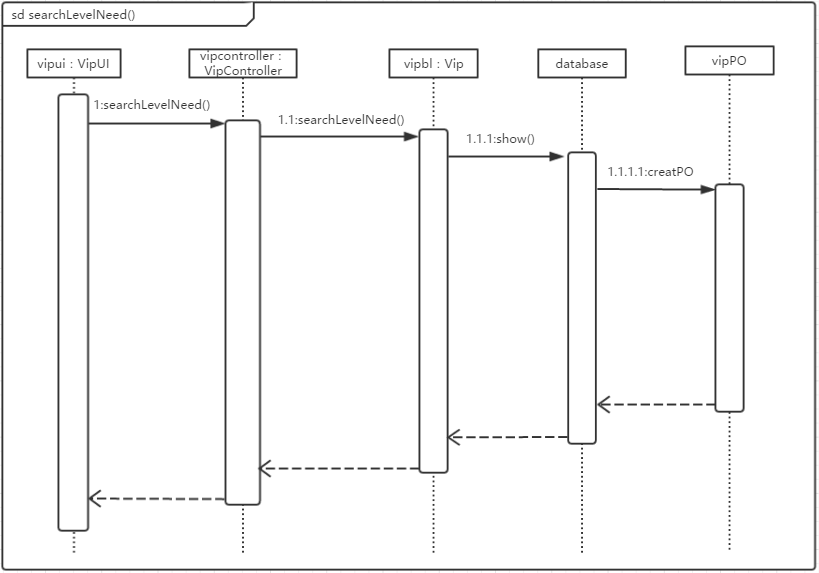


图4.2.4.3查询会员等级和信用值对应关系

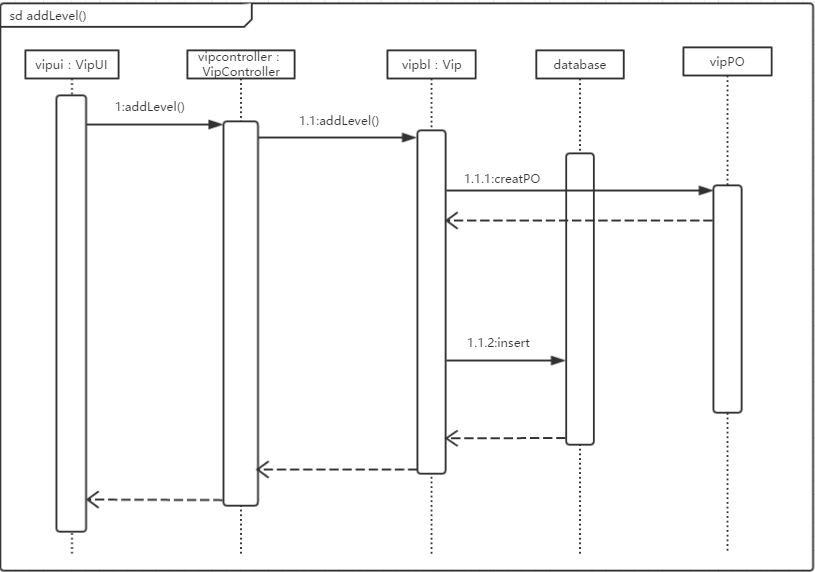
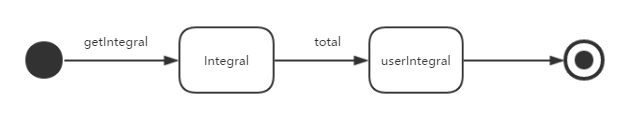


图4.2.4.3增加一个会员等级

下图所示的状态图描述了Vip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。



5.业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### Integral模块

1、模块概述

Integral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。

Integral模块的职责和接口参见软件系统结构描述文档内对该模块的描述。

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。在展示层和业务逻辑层之间添加IntegralBLService接口。在业务逻辑层和数据层之间添加IntegralDataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加IntegralController，这样IntegralController会将对信用值操作的业务逻辑处理委托给Integral对象。IntegralPO是作为信用值的持久性对象被添加到设计模型中去的。

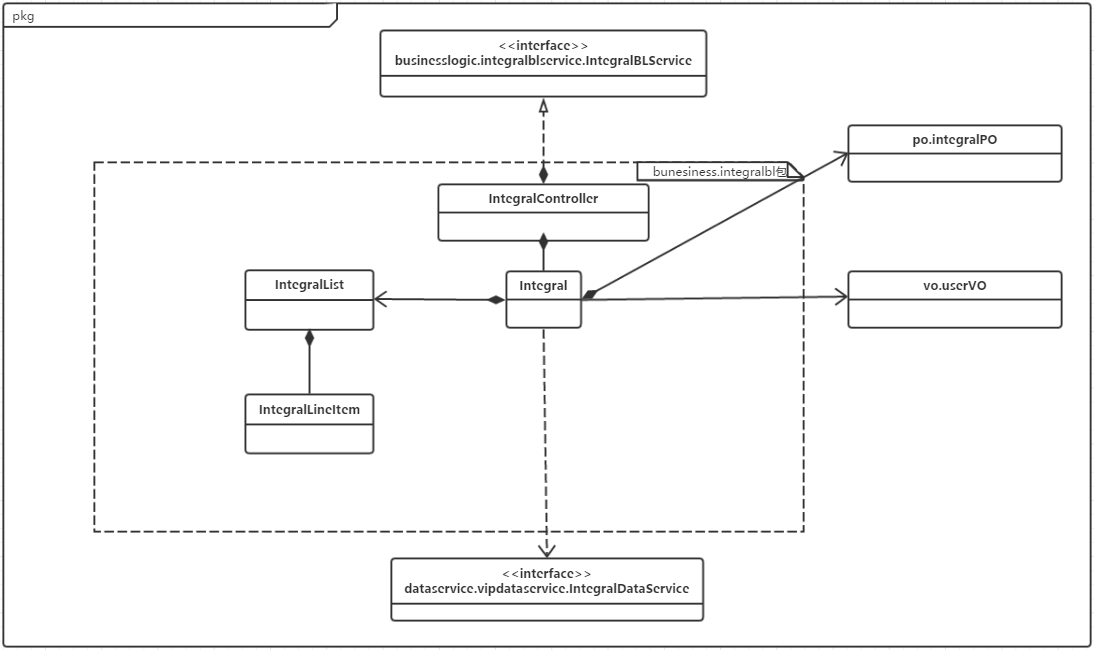


表4.2.5.1 integralbl模块各个类的设计

integralbl模块各个类的职责如表4.2.5.2所示。

|  |  |
| --- | --- |
| 模块 | 职责 |
| IntegralControl | 负责实现信用值部分所需要的服务 |
| Integral | 信用值的领域模型对象，拥有对应用户的信用值信息，可以解决用户信用值变化，修改问题 |
| IntegralList | 负责保存信用值的相关信息 |
| IntegralLineItem | 信用值的条目信息 |

3、模块内部类的接口规范

**IntegralControl类的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| IntegralControl.getIntegral | 语法 | public IntegralVO getIntegral(userVO client) |
| 前置条件 | 已经创建一个integral领域对象，且输入格式符合规范 |
| 后置条件 | 调用integral类的getIntegral方法，系统返回当前用户信用值 |
| IntegralControl.addIntegral | 语法 | public void addIntegral(userVO client，int value) |
| 前置条件 | 已经创建一个integral领域对象，且输入格式符合规范 |
| 后置条件 | 调用integral类的addIntegral方法，系统将要增加的信用值增加到该用户身上，并且持久化更新integralVO |
| IntegralControl.subIntegral | 语法 | public void subIntegral(userVO client，int value) |
| 前置条件 | 已经创建一个integral领域对象，且输入格式符合规范 |
| 后置条件 | 调用integral类的subIntegral方法，系统将要减少的信用值从该用户减少，并且持久化更新integralVO |
| IntegralControl.changeIntegral | 语法 | public void changeIntegral(userVO client，int value) |
| 前置条件 | 已经创建一个integral领域对象，且输入格式符合规范 |
| 后置条件 | 调用integral类的changeIntegral方法，系统该用户信用值修改为规定数值，并且持久化更新integralVO |
| 需要的接口（需接口） | | |
| 服务名 | 服务 | |
| Integral.getIntegral(userVO client) | 得到信用值单一化持久对象 | |
| Integral.addIntegral(userVO client，int value) | 增加信用值数额 | |
| Integral.subIntegral(userVO client，int value) | 减少信用值数额 | |
| Integral.changeIntegral(userVO client，int value) | 修改信用值数额 | |

**Integralbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Integral.getIntegral | 语法 | public IntegralVO getIntegral(userVO client) |
| 前置条件 | 用户要获得当前的信用值 |
| 后置条件 | 系统获取当前用户的信用值，返回回来 |
| Integral.addIntegral | 语法 | public void addIntegral(userVO client，int value) |
| 前置条件 | 用户或系统要对当前客户的信用值进行增加 |
| 后置条件 | 系统将增加的数值增加到当前客户的信用值上 |
| Integral.subIntegral | 语法 | public void subIntegral(userVO client，int value) |
| 前置条件 | 用户或系统要对当前客户的信用值进行减少 |
| 后置条件 | 系统将增加的数值从当前客户的信用值减少 |
| Integral.changeIntegral | 语法 | public void changeIntegral(userVO client，int value) |
| 前置条件 | 用户或系统要对当前客户的信用值进行修改 |
| 后置条件 | 系统将用户信用值修改为当前数值 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| IntegralDataService.insert | 增加一个持久化的信用值 |
| IntegralDataService.delete | 删除一个持久化的信用值 |
| IntegralDataService.update | 更新一个持久化的信用值 |
| IntegralDataService.findByName | 根据名字找到一个持久化的信用值 |
| IntegralDataService.Show | 展示所有的持久化的信用值 |

4.业务逻辑层的动态模型

在信用值业务中，分为信用值修改，信用值查看，信用值增加功能。图4.2.5.3表示了信用值修改的相关对象之间的协作。

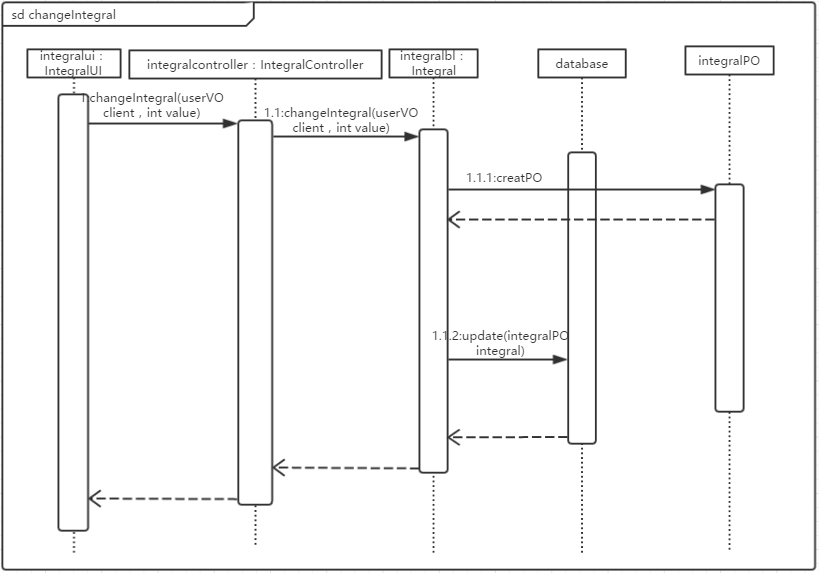
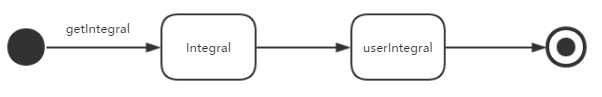


图4.2.5.3信用值修改

下图所示的状态图描述了Integral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。



5.业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### Promotionbl模块

1、模块概述

Promotion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。

Promotion模块的职责和接口参见软件系统结构描述文档内对该模块的描述。

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。在展示层和业务逻辑层之间添加PromotionBLService接口。在业务逻辑层和数据层之间添加PromotionDataService接口。为了隔离业务逻辑职责和逻辑控制职责，添加PromotionHotelController，PromotionWebController，PromotionValueController，这样PromotionHotelController会将对信用值操作的业务逻辑处理委托给PromotionHotel对象，PromotionWebController会将对信用值操作的业务逻辑处理委托给PromotionWeb对象，PromotionValueController会将对信用值操作的业务逻辑处理委托给PromotionValue对象。PromotionPO是作为促销策略的持久性对象被添加到设计模型中去的。

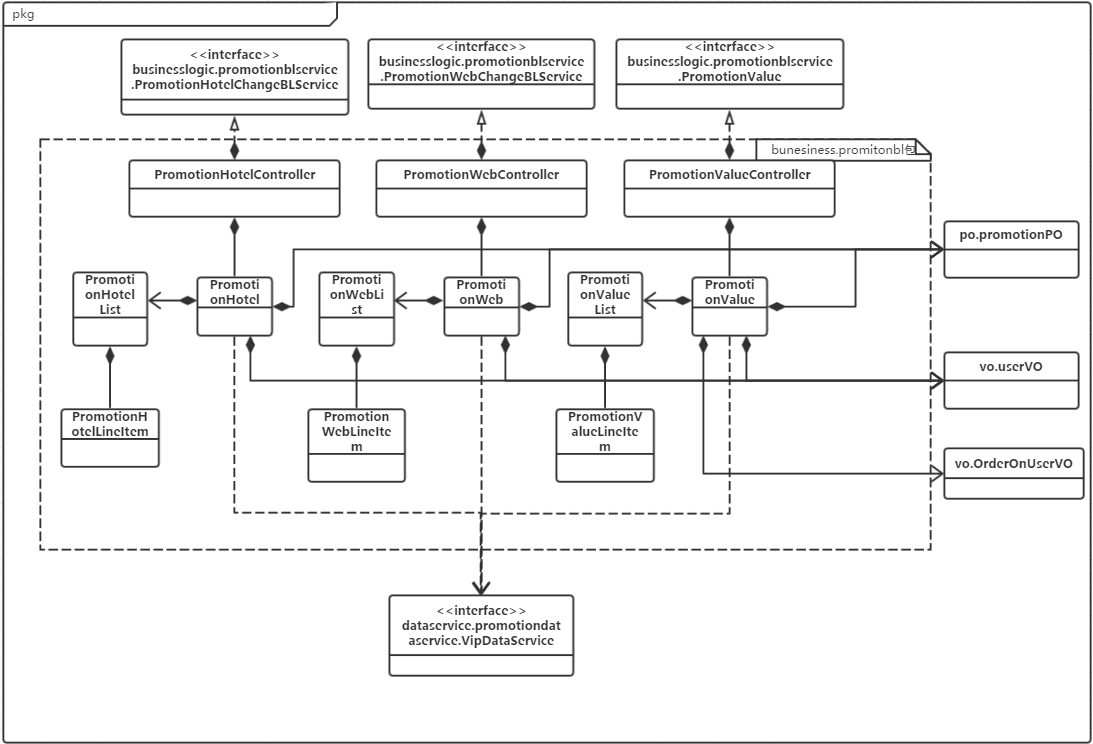


表4.2.6.1 promotionbl模块各个类的设计

promotionbl模块各个类的职责如表4.2.6.2所示。

|  |  |
| --- | --- |
| 模块 | 职责 |
| promotionHotelControl | 负责实现酒店促销策略部分需要的服务 |
| PromotionHotel | 酒店促销策略的领域模型对象，拥有策略适用用户种类，促销策略降价比例，促销策略有效时间等信息，可以解决酒店促销策略的修改，增加，删除问题。 |
| PromotionHotelList | 负责保存酒店促销策略的相关信息 |
| PromotionHotelLineItem | 酒店促销策略的条目信息 |
| promotionWebControl | 负责实现网站促销策略部分需要的服务 |
| PromotionWeb | 网站促销策略的领域模型对象，拥有策略适用用户种类，促销策略降价比例，促销策略有效时间等信息，可以解决网站促销策略的修改，增加，删除问题。 |
| PromotionWebList | 负责保存网站促销策略的相关信息 |
| PromotionWebLineItem | 网站促销策略的条目信息 |
| promotionValueControl | 负责实现订单经过促销策略之后的订单价值部分所需要的服务 |
| PromotionValue | 订单价值计算的领域模型对象，拥有各种促销策略适用对象信息，能够解决促销策略匹配问题 |
| PromotionValueList | 负责保存订单价值计算的相关信息 |
| PromotionValueLineItem | 订单价值计算的条目信息 |

3、模块内部类的接口规范

**promotionHotelControl类的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionHoelControl.changeBirthCut | 语法 | public void changeBirthCut(promotionPO po) |
| 前置条件 | 已经创建一个PromotionHoel领域对象，且输入格式符合规范 |
| 后置条件 | 调用PromotionHoel类的changeBirthCut方法，系统将生日促销策略进行修改，并且持久化更新promotionPO |
| PromotionHoelControl.changeOverCut | 语法 | public void changeOverCut(promotionPO po) |
| 前置条件 | 已经创建一个PromotionHoel领域对象，且输入格式符合规范 |
| 后置条件 | 调用PromotionHoel类的changeOverCut方法，系统将三间以上促销策略进行修改，并且持久化更新promotionPO |
| PromotionHoelControl.changeJoinCut | 语法 | public void changeJoinCut (promotionPO po) |
| 前置条件 | 已经创建一个PromotionHoel领域对象，且输入格式符合规范 |
| 后置条件 | 调用PromotionHoel类的changeJoinCut方法，系统将合作企业促销策略进行修改，并且持久化更新promotionPO |
| PromotionHoelControl.changeHotelCustomCut | 语法 | public void changeHotelCustomCut (promotionPO po) |
| 前置条件 | 已经创建一个PromotionHoel领域对象，且输入格式符合规范 |
| 后置条件 | 调用PromotionHoel类的changeHotelCustomCut方法，系统将自定义促销策略进行修改，并且持久化更新promotionPO |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| PromotionHoel.changeBirthCut | 修改会员生日优惠政策 | |
| PromotionHoel.changeOverCut | 修改三间以上优惠政策 | |
| PromotionHoel.changeJoinCut | 修改合作企业优惠政策 | |
| PromotionHoel.changeHotelCustomCut | 修改酒店自定义优惠政策 | |

**promotionWebControl类的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionWebControl.changeLevelCut | 语法 | public void changeLevelCut(promotionPO po) |
| 前置条件 | 已经创建一个PromotionWeb领域对象，且输入格式符合规范 |
| 后置条件 | 调用PromotionWeb类的changeLevelCut方法，系统将会员等级优惠策略进行修改，并且持久化更新promotionPO |
| PromotionWebControl.changeCircleCut | 语法 | public void changeCircleCut(promotionPO po) |
| 前置条件 | 已经创建一个PromotionWeb领域对象，且输入格式符合规范 |
| 后置条件 | 调用PromotionWeb类的changeCircleCut方法，系统将特定商圈优惠策略进行修改，并且持久化更新promotionPO |
| PromotionWebControl.changeWebCustomCut | 语法 | public void changeWebCustomCut(promotionPO po) |
| 前置条件 | 已经创建一个PromotionWeb领域对象，且输入格式符合规范 |
| 后置条件 | 调用PromotionWeb类的changeWebCustomCut方法，系统将自定义优惠策略进行修改，并且持久化更新promotionPO |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| PromotionWeb.changeLevelCut | 修改会员等级优惠策略 | |
| PromotionWeb.changeCircleCut | 修改会员特定商圈优惠策略 | |
| PromotionWeb.changeWebCustomCut | 修改网站自定义优惠策略 | |

**promotionValueControl类的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionValueControl.getValue | 语法 | public promotionVO getValue(userVO user, OrderOnUserVO orderVO) |
| 前置条件 | 已经创建一个PromotionValue领域对象，且输入格式符合规范 |
| 后置条件 | 调用PromotionValue类的getValue方法，系统自动计算订单价值，并且持久化更新promotionPO |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| PromotionValue.getValue | 获得该订单经过打折之后的信息 | |

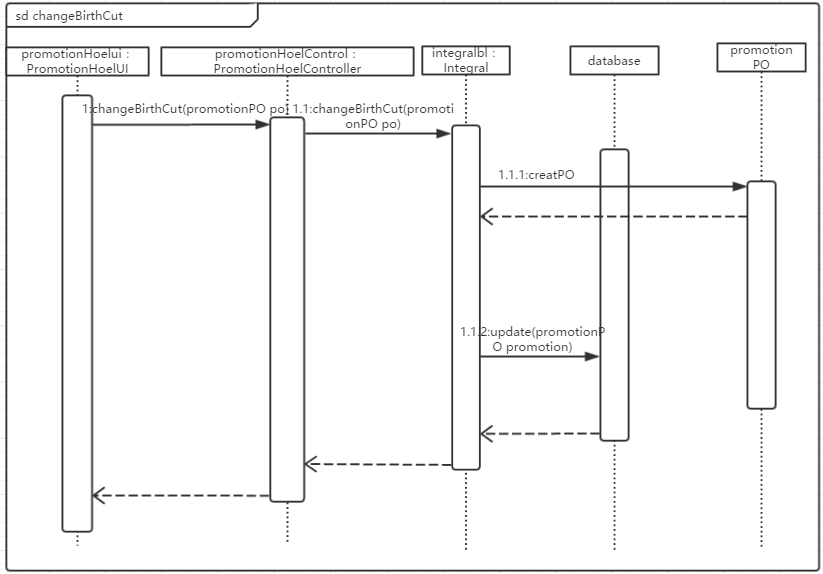
**promotionbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PromotionHoel.changeBirthCut | 语法 | public void changeBirthCut(promotionPO po) |
| 前置条件 | 酒店工作人员要修改酒店会员生日促销策略 |
| 后置条件 | 系统将酒店会员生日促销策略降价比例修改为ratio |
| PromotionHoel.changeOverCut | 语法 | public void changeOverCut(promotionPO po) |
| 前置条件 | 酒店工作人员要修改酒店三间以上优惠策略 |
| 后置条件 | 系统将酒店三间以上促销策略降价比例修改为ratio |
| PromotionHoel.changeJoinCut | 语法 | public void changeJoinCut (promotionPO po) |
| 前置条件 | 酒店工作人员要修改酒店合作企业客户优惠策略 |
| 后置条件 | 系统将酒店合作企业促销策略降价比例修改为ratio |
| PromotionHoel.changeHotelCustomCut | 语法 | public void changeHotelCustomCut (promotionPO po) |
| 前置条件 | 酒店工作人员要修改酒店自定义优惠策略 |
| 后置条件 | 系统将酒店自定义优惠策略修改为当前设定 |
| PromotionWeb.changeLevelCut | 语法 | public void changeLevelCut(promotionPO po) |
| 前置条件 | 网站营销人员修改网站会员等级优惠 |
| 后置条件 | 系统将网站会员等级优惠修改为ratio |
| PromotionWeb.changeCircleCut | 语法 | public void changeCircleCut(promotionPO po) |
| 前置条件 | 网站营销人员修改网站vip商圈专属折扣优惠 |
| 后置条件 | 系统将网站vip商圈专属优惠折扣修改为ratio |
| PromotionWeb.changeWebCustomCut | 语法 | public void changeWebCustomCut(promotionPO po) |
| 前置条件 | 网站营销人员修改网站自定义促销策略 |
| 后置条件 | 系统将网站自定义促销策略修改为当前设定 |
| PromotionValue.getValue | 语法 | public promotionVO getValue(userVO user, OrderOnUserVO orderVO) |
| 前置条件 | 订单要利用促销策略计算订单价值 |
| 后置条件 | 系统返回订单经过促销策略打折后的价值 |

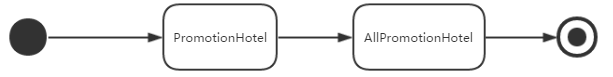
|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| PromotionDataService.insert | 怎加一个促销策略 |
| PromotionDataService.delete | 删除一个促销策略 |
| PromotionDataService.update | 更新一个促销策略 |
| PromotionDataService.findByName | 根据促销策略种类找到一个促销策略 |
| PromotionDataService.Show | 展示所有的促销策略 |

4.业务逻辑层的动态模型

在促销策略业务中，分为酒店促销策略修改，网站促销策略修改，订单价值计算功能。图4.2.6.3表示了酒店促销会员生日策略修改的相关对象之间的协作。



下图所示的状态图描述了PromotionHotel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。



5.业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

## 数据层的分解

数据层模块的分解详见需求规格说明文档中的相关描述

### OrderDataService模块

1、模块概述

OrderDataService模块负责订单的相关信息，供读取、保存、修改。

模块内部接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| OrderOnUserDataService .insert | 语法 | public void insert(OrderOnUserPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中不存在 |
| 后置条件 | 新增一个po |
| OrderOnUserDataService .delete | 语法 | public void delete(OrderOnUserPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 删除一个po |
| OrderOnUserDataService.update | 语法 | public void update(OrderOnUserPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 更新一个po |
| OrderOnUserDataService.findByName | 语法 | public OrderOnUserPO  findByName(String name) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按名字进行查找返回相应的OrderOnUserPO结果 |
| OrderOnUserDataService.findByCondition | 语法 | public ArrayList<OrderOnUserPO>  findByCondition(String condition) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按条件进行查找返回相应的OrderOnUserPO结果 |
| OrderOnUserDataService.show | 语法 | public  ArrayList<OrderOnUserPO>  show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的客户订单po |
| OrderOnHotelDataService.insert | 语法 | public void insert(OrderOnHotelPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中不存在 |
| 后置条件 | 新增一个po |
| OrderOnHotelDataService.delete | 语法 | public void delete(OrderOnHotelPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 删除一个po |
| OrderOnHotelDataService.update | 语法 | public void update(OrderOnHotelPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 更新一个po |
| OrderOnHotelDataService.find | 语法 | public OrderOnHotelPO  findByName(String searchInfo) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 进行查找返回相应的OrderOnHotelPO结果 |
| OrderOnHotelDataService.show | 语法 | public  ArrayList<OrderOnHotelPO>  show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的酒店订单po |
| OrderOnWebDataService.insert | 语法 | public void insert(OrderOnWebPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中不存在 |
| 后置条件 | 新增一个po |
| OrderOnWebDataService.delete | 语法 | public void delete(OrderOnWebPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 删除一个po |
| OrderOnWebDataService.update | 语法 | public void update(OrderOnWebPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 更新一个po |
| OrderOnWebDataService.findByName | 语法 | public OrderOnWebPO  findByName(String name) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按名字进行查找返回相应的OrderOnWebPO结果 |
| OrderOnWebDataService.findByCondition | 语法 | public ArrayList<OrderOnWebPO>  findByCondition(String condition) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按条件进行查找返回相应的OrderOnWebPO结果 |
| OrderOnWebDataService.show | 语法 | public  ArrayList<OrderOnWebPO>  show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的异常订单po |

### UserDataService模块

1. 模块概述

UserDataService建立逻辑和数据库之间的联系，获取和更改数据信息。

1. 模块内部接口规范

serdataservice模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的接口（供接口） | | |
| Userdataservice.GetLoginInfo | 语法 | Public String GetLoginInfo(UserManagementPo po); |
| 前置条件 | 数据库中有输入ID的登录信息。 |
| 后置条件 | 返回对应账号的密码。 |
| Userdataservice.AddUser | 语法 | Public String AddUser(UserManagementPo po); |
| 前置条件 | 无。 |
| 后置条件 | 返回新注册用户的userID。 |
| Userdataservice.GetUserBaseInfo | 语法 | Public UserVo GetUserBaseInfo(UserManagementPo po); |
| 前置条件 | 输入的用户ID在数据库中已有注册。 |
| 后置条件 | 返回对应用户的个人信息。 |
| Userdataservice.SetUserBaseInfo | 语法 | Public boolean SetUserBaseInfo(UserManagementPo po); |
| 前置条件 | 输入的用户ID存在且VO中信息完整。 |
| 后置条件 | 修改用户信息数据。 |
| Userdataservice.AddMembers | 语法 | Public boolean AddMembers(CustomerManagementDataServicePo po); |
| 前置条件 | 输入的userID在数据库中存在。 |
| 后置条件 | 将相应用户信息中的是否为会员一栏修改为是。 |
| Userdataservice.GetCustomerOrders | 语法 | Public ArrayList<OrderVo> GetCustomerOrders(CustomerManagementDataServicePo po); |
| 前置条件 | 输入的userID存在。 |
| 后置条件 | 返回相应用户ID对应的订单列表。 |
| Userdataservice.UpDateCustomerOrders | 语法 | Public boolean UpdateCustomerOrders(CustomerManagementDataServicePo po); |
| 前置条件 | 输入的userID存在且OrderID对应的订单状态为可修改。 |
| 后置条件 | 修改相应订单状态。 |
| Userdataservice.GetCustomerrHotel | 语法 | Public ArrayList<HotelVo> GetCustomerHotel(CustomerManagementDataServicePo po); |
| 前置条件 | 输入的userID存在。 |
| 后置条件 | 返回对应用户的酒店信息数组。 |
| Userdataservice.SetCustomerHotel | 语法 | Public boolean SetCustomerHotel (CustomerManagementDataServicePo po); |
| 前置条件 | 输入的userID和hotelID存在。 |
| 后置条件 | 系统更新相应用户的酒店信息。 |
| Userdataservice.SetCustomerCredit | 语法 | Public boolean SetCustomerCredit(CustomerManagementDataServicePo po); |
| 前置条件 | 输入的userID存在且输入的信用度在合理范围内。 |
| 后置条件 | 更新相应用户的信用度信息和信用记录。 |

### HotelDataService模块

（1）模块概述

HotelDataService模块负责保存酒店的信息，进行增删改查的操作。

（2）模块内部接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelInfoDataService.Insert | 语法 | public void insert(HotelInfoPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中不存在 |
| 后置条件 | 新增一个po |
| HotelInfoDataService.Delete | 语法 | public void delete(HotelInfoPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 删除一个po |
| HotelInfoDataService.Update | 语法 | public void update(HotelInfoPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 更新一个po |
| HotelInfoDataService.FindByName | 语法 | public HotelInfoPO  findByName(String name) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按名字进行查找返回相应的HotelInfoPO结果 |
| HotelInfoDataService.FindByCondition | 语法 | public HotelInfoPO  findByCondition(String condition) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按条件进行查找返回相应的HotelInfoPO结果 |
| HotelInfoDataService.Show | 语法 | public  ArrayList<HotelInfoPO>  show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的库存po |
| HotelEvaluateDataService.Insert | 语法 | public void insert(HotelEvaluatePO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中不存在 |
| 后置条件 | 新增一个po |
| HotelEvaluateDataService.Delete | 语法 | public void delete(HotelEvaluatePO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 删除一个po |
| HotelEvaluateDataService.Find | 语法 | public HotelEvaluatePO  findByName(String searchInfo) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 进行查找返回相应的HotelEvaluatePO结果 |
| HotelEvaluateDataService.Show | 语法 | public  ArrayList<HotelEvaluatePO>  show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的库存po |
| HotelEvaluateDataService.Insert | 语法 | public void insert(RoomInfoPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中不存在 |
| 后置条件 | 新增一个po |
| HotelEvaluateDataService.Delete | 语法 | public void delete(RoomInfoPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 删除一个po |
| HotelEvaluateDataService.Update | 语法 | public void update(RoomInfoPO po) throws RemoteException |
| 前置条件 | 相同的po在数据文件中已存在 |
| 后置条件 | 更新一个po |
| HotelEvaluateDataService.FindByName | 语法 | public RoomInfoPO  findByName(String name) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按名字进行查找返回相应的RoomInfoPO结果 |
| HotelEvaluateDataService.FindByCondition | 语法 | public RoomInfoPO  findByCondition(String condition) throws  RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按条件进行查找返回相应的RoomInfoPO结果 |
| HotelEvaluateDataService.Show | 语法 | public  ArrayList<RoomInfoPO>  show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的库存po |

### VipDataService模块

（1）模块概述

VipDataService模块负责保存会员的相关信息，供读取、保存、修改。

（2）模块内部接口规范

|  |  |  |
| --- | --- | --- |
| 需要的服务（需接口） | | |
| VipDataService.insert | 语法 | public void insert(vipPO vip) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中不存在 |
| 后置条件 | 在数据库中增加一个 po 记录 |
| VipDataService.delete | 语法 | public void delete(vipPO vip) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中存在 |
| 后置条件 | 在数据库中删除一个 po 记录 |
| VipDataService.update | 语法 | public void update(vipPO vip) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中存在 |
| 后置条件 | 在数据库中更新一个 po 记录 |
| VipDataService.findByLevel | 语法 | public vipPO findByLevel(int level) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按等级返回相应的vipPO结果 |
| VipDataService.Show | 语法 | public ArrayList<vipPO> show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的库存 PO |

### IntegralDataService模块

（1）模块概述

IntegralDataService模块负责保存信用值的相关信息，供读取、保存、修改。

（2）模块内部接口规范

|  |  |  |
| --- | --- | --- |
| 需要的服务（需接口） | | |
| IntegralDataService.insert | 语法 | public void insert(integralPO integral) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中不存在 |
| 后置条件 | 在数据库中增加一个 po 记录 |
| IntegralDataService.delete | 语法 | public void delete(integralPO integral) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中存在 |
| 后置条件 | 在数据库中删除一个 po 记录 |
| IntegralDataService.update | 语法 | public void update(integralPO integral) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中存在 |
| 后置条件 | 在数据库中更新一个 po 记录 |
| IntegralDataService.findByName | 语法 | public integralPO findByLevel(String name) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按等级返回相应的integralPO结果 |
| IntegralDataService.Show | 语法 | public ArrayList<integralPO> show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的库存 PO |

### PromotionDataService模块

（1）模块概述

PromotionDataService模块负责保存促销策略的相关信息，供读取、保存、修改。

（2）模块内部接口规范

|  |  |  |
| --- | --- | --- |
| 需要的服务（需接口） | | |
| PromotionDataService.insert | 语法 | public void insert(promotionPO promotion) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中不存在 |
| 后置条件 | 在数据库中增加一个 po 记录 |
| PromotionDataService.delete | 语法 | public void delete(promotionPO promotion) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中存在 |
| 后置条件 | 在数据库中删除一个 po 记录 |
| PromotionDataService.update | 语法 | public void update(promotionPO promotion) throws RemoteException |
| 前置条件 | 同样 ID 的 po 在 Mapper 中存在 |
| 后置条件 | 在数据库中更新一个 po 记录 |
| PromotionDataService.findByName | 语法 | public promotionPO findByLevel(String name) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按等级返回相应的promotionPO结果 |
| PromotionDataService.Show | 语法 | public ArrayList<promotionPO> show() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的库存 PO |

# 依赖视角

下图分别为客户端和服务器端各自的包之间的联系：

