**步客酒店预订系统**

**详细设计描述文档**

学 院：南京大学软件学院

成 员：王友运卞纯源王雪 赵德宇

完成日期：2016年12月25日

目录

[更新历史 3](#_Toc470288469)

[引言 4](#_Toc470288470)

[1.1编制目的 4](#_Toc470288471)

[1.2词汇表 4](#_Toc470288472)

[1.3参考资料 4](#_Toc470288473)

[1. 产品概述 4](#_Toc470288474)

[2. 体系结构设计概述 5](#_Toc470288475)

[3. 结构视角 5](#_Toc470288476)

[4.1业务逻辑层的分解 5](#_Toc470288477)

[4.1.1．Creditbl模块 5](#_Toc470288478)

[4.1.2．Userbl模块 10](#_Toc470288479)

[4.1.3．accountbl模块 23](#_Toc470288480)

[4.1.4．roombl模块 28](#_Toc470288481)

[4.1.5．Hotelbl模块 34](#_Toc470288482)

[4.1.6．orderbl模块 45](#_Toc470288483)

[4.1.7．promotionbl模块 59](#_Toc470288484)

[4.2. 界面展示层的分解 67](#_Toc470288485)

[4.2.1界面层跳转 67](#_Toc470288486)

[4.2.2下面给出几个人员的重要跳转以作示例 69](#_Toc470288487)

[5.依赖视角 72](#_Toc470288488)

[图5-1客户端开发包 72](#_Toc470288489)

[图5-2 服务器端开发包 73](#_Toc470288490)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 王友运 | 2016-10-24 | 创建文档模板 | V1.0 |
| 全体成员 | 2016-10-28 | 完成文档初稿 | V1.1 |
| 卞纯源 | 2016-11-5 | 修改user的调用 | V1.2 |
| 王友运 | 2016-11-13 | 修改hotel包的类间协作图 | V1.3 |
| 王友运 | 2016-12-15 | 修改包的划分，删除evaluation和 ordermanagement | V2.0 |
| 王友运 | 2016-12-18 | 修改order包和hotel包的模块类设计图 | V2.1 |
| 赵德宇 | 2016-12-22 | 添加界面层的详细设计描述 | V3.0 |

# 引言

## 1.1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。

## 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

参考酒店预订系统用例文档和酒店预订系统软件需求规格说明文档中对产品的概括描述。

# 体系结构设计概述

参考酒店预订系统概要设计文档中对体系结构设计的概述

# 结构视角

本系统采用委托式控制风格。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。

## 4.1业务逻辑层的分解

### 4.1.1．Creditbl模块

**（1）模块概述**

Creditbl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

Creditbl模块的职责及接口参见软件体系结构描述文档。

**（2）整体结构**

根据体系结构的设计，采用分层风格，将系分为展示层，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，在ui和logic层之间添加了CreditServiceCreditChangeService,两个接口，在logic和data之间添加了CreditDao接口。

Credit类提供获取信用和vip信用信息的方法。因为信用记录改变等需要被order调用，所以建立CreditChange类，处理历史信用记录，提供修改信用和获取信用记录的方法。CreditPO是做为管理信息的持久化对象被添加到设计模型中的。在添加处理信用记录，更改信用址的同时，有可能会对用户的信用等级产生改变（保存在user里），所以credit模块需要user的接口：UpdateClientVip。

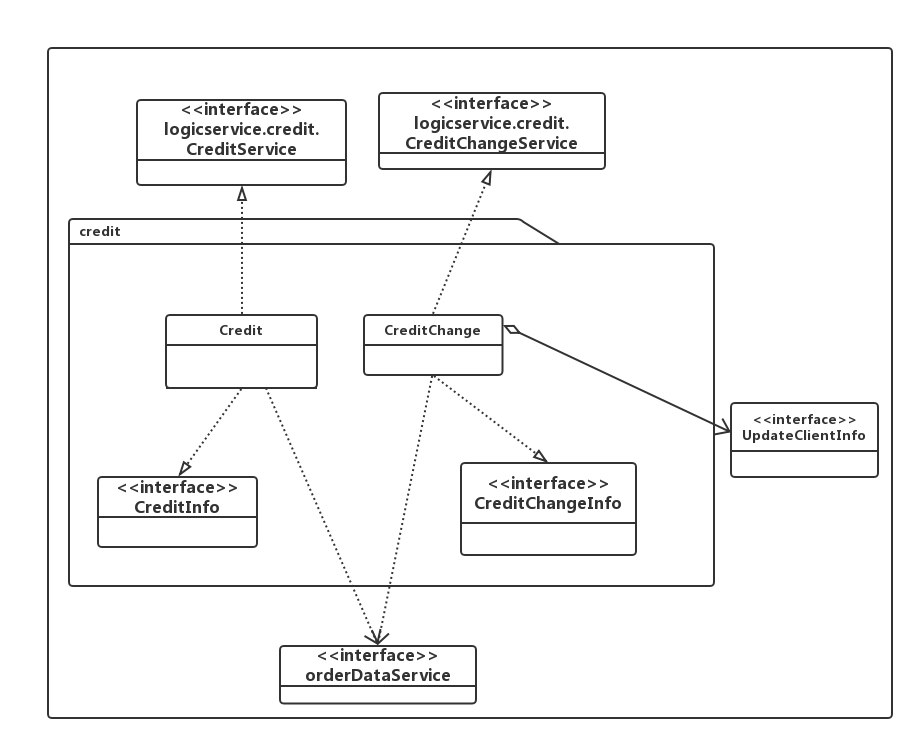
****

图4.1.1-1credit模块类的设计

**（3）模块内部类的接口规范**

表4.1.1-1credit的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Credit.getCredit | 语法 | publicint getCredit (String user\_id); | |
| 前置条件 | 已知客户id，需要获得客户的信用值 | |
| 后置条件 | 返回信用值 | |
| Credit.setVipCredit | 语法 | public ResultMessage setVIPCredit(, int credit\_num); | |
| 前置条件 | 网站营销人员已经登录 | |
| 后置条件 | 返回成功与否 | |
| Credit.getVIPCredit | 语法 | public int getVIPCredit ( ) | |
| 前置条件 | 需要获得升级所需的信用值 | |
| 后置条件 | 返回信用值 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Data层 | | |  |
| CreditDao. getCredit（String userID） | | | 获得某一用户的信用值 |
| CreditDao.setVIPCredit(int level, int credit\_num) | | | 设置vip升级所需要的信用值 |
| CreditDao.getVIPCredit(int level) | | | 获得升级指定VIP所需要的信用值 |
|  | | |  |

表4.1.1-2CreditHistory的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CreditHistory.changeCredit | 语法 | Public ResultMessagechangeCredit(CreditChangeVO vo) | |
| 前置条件 | 用户执行订单或者撤销订单或者充值，或者网站营销人员撤销异常订单 | |
| 后置条件 | 系统更新用户的信用记录和信用值 | |
| CreditChange.getCreditHistory | 语法 | publicArrayList<CreditChangeVO>getCreditHistory(String userID); | |
| 前置条件 | 用户登陆，需要查看信用记录 | |
| 后置条件 | 数据从数据层调用至界面 | |
| CreditChange.rechargeCredit | 语法 | Public ResultMessage rechargeCredit(CreditChangeVO vo) | |
| 前置条件 | 用户向网站营销人员进行线下充值，网站营销人员调用该方法给用户充值 | |
| 后置条件 | 调用同类中的changeCredit方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| User.UpdateClientVip.updateClientVip(String userID, int level) | | | （当信用值的改变导致用户vip等级的改变时）改变用户的vip等级 |
| Data层 | | |  |
| CreditDao.changeCredit(CreditChangePO po) | | | 向数据层添加信用记录 |
| CreditDao.getCreditHistory(String userID) | | | 根据ID获取用户的所有信用记录 |
|  | | |  |

**（4）动态模型**

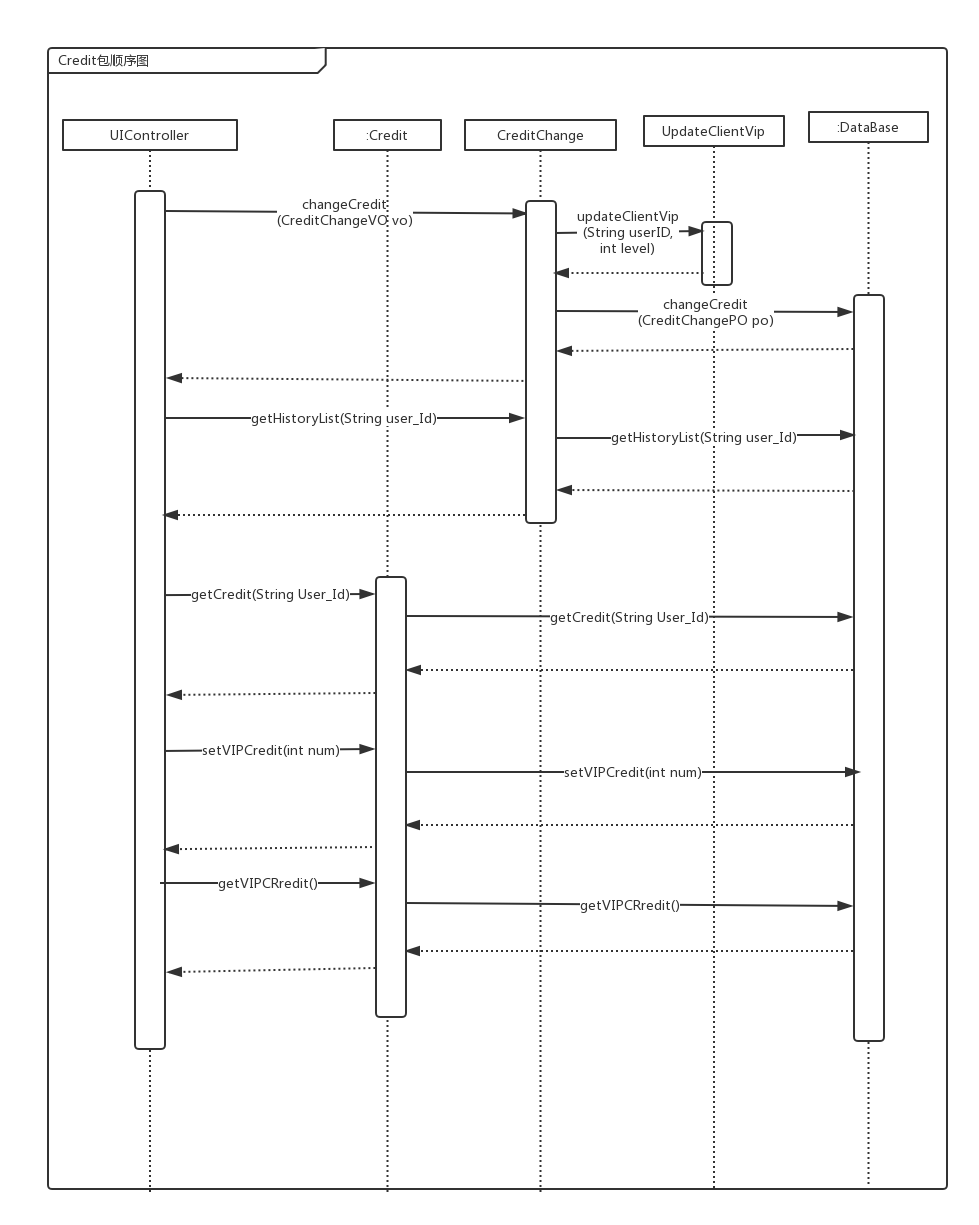
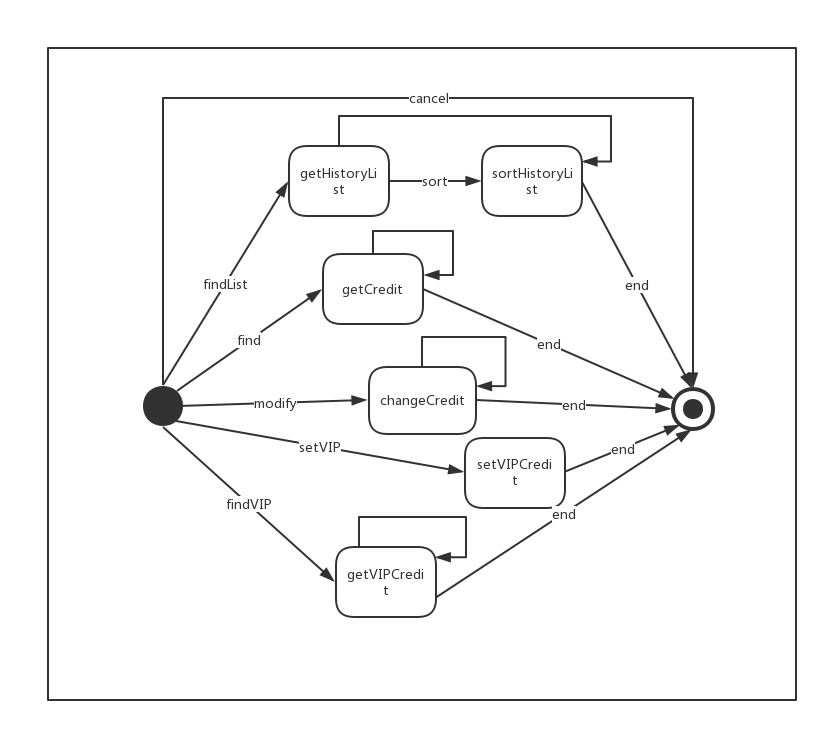
****

图4.1.1-2credit顺序图

图4.1.1-3 信用信息管理状态图

### 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。

**（2）整体结构**

根据体系结构的设计，采用分层风格，将系分为展示层，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，同时分离四种用户的职责，在ui和logic之间添加了ClientService, HotelManagerService, WebBusinessService, WebManagerService四个接口, 在logic和data之间添加了ClientDao, HotelManagerDao, WebBusinessDao, WebManagerDao四个接口。Client类是为了处理客户信息被添加在模型中。HotelManager类是为了处理酒店工作人员信息被添加在模型中。WebBusiness类是为了处理网站营销人员信息被添加在模型中。WebManager类是为了实现对所有用户信息的添加和更改被添加在模型中，其中他关于各种用户信息管理功能的实现是调用所对应用户的类的方法来实现的。ClientPO是做为管理信息的持久化对象被添加到设计模型中的。

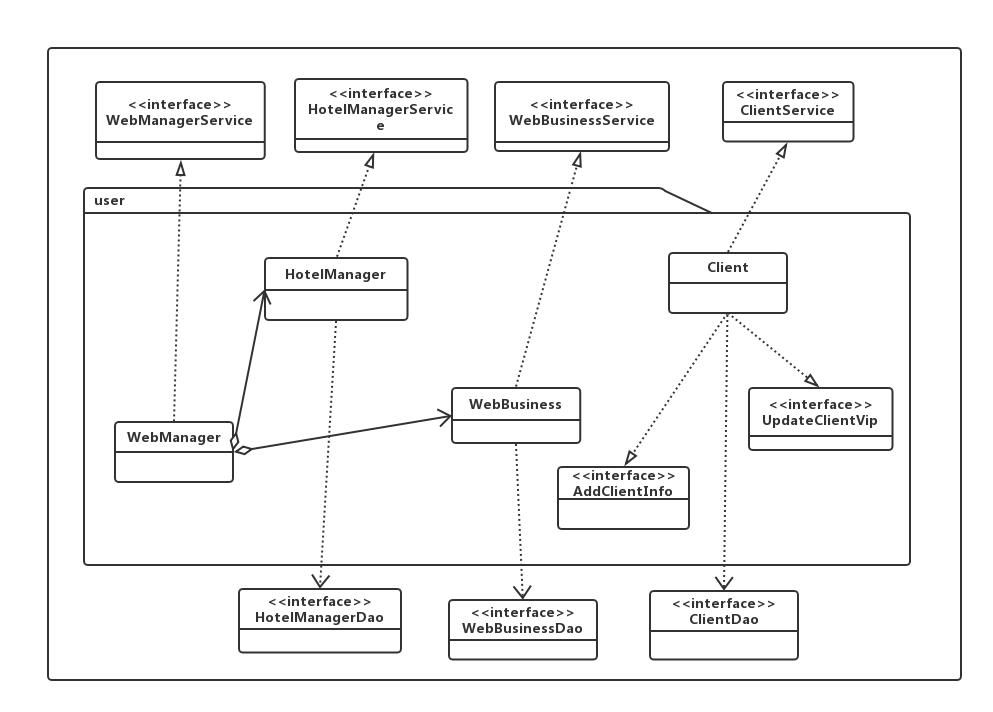
****

图4.1.2-1user模块类的设计

**（3）模块内部类的接口规范**

表4.1.2-1Client的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Client.getClientInfo | 语法 | Public ClientVO getClientInfo(StringuserID); | |
| 前置条件 | 已获得客户ID，需要查看基本信息 | |
| 后置条件 | 返回对应的用户基本信息 | |
| Clilent.updateClientInfo | 语法 | Public ResultMessage updateClientInfo(ClientVO vo); | |
| 前置条件 | 已知用户ID，需要修改对应的基本信息 | |
| 后置条件 | 更新用户数据并且返回是否修改成功 | |
| Client.registerVIP | 语法 | Public ResultMessage registerVIP(VIPVO vo); | |
| 前置条件 | 已获得用户ID和注册会员信息 | |
| 后置条件 | 更新用户vip信息并且返回注册会员是否成功 | |
| Client.getVIPInfo | 语法 | Public VIPVO getVIPInfo(String userID); | |
| 前置条件 | 客户已注册为会员，已获得客户ID，需要查看注册会员信息 | |
| 后置条件 | 查看数据并且返回用户注册会员信息 | |
| Client.updateClientVip | 语法 | Public booleanupdateClientVip (String userID, int level); | |
| 前置条件 | 用户信用值改变并且达到了vip升级或者降级的标准 | |
| 后置条件 | 更新数据并且返回成功与否 | |
| Client.isVip | 语法 | Public boolean isVip(String userID) | |
| 前置条件 | 用户已经登录，需要查看自己是否是vip | |
| 后置条件 | 查看数据并且返回是与否 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| ClientDao.updateClientInfo(ClientPO user) | | | 修改用户信息 |
| Data层 | | |  |
| ClientDao.getClientInfo(Stirng userID) | | | 根据ID获取用户的信息 |
| ClientDao.addClient(ClientPO po) | | | 新增用户信息 |
|  | | |  |

表4.1.2-2HotelManager的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| HotelManager.addHotelManager | 语法 | Public ResultMessage addHotelManager(HotelManagerVO hotelManager); |
| 前置条件 | 已添加酒店，需要添加对应的酒店工作人员信息 |
| 后置条件 | 更新数据并且返回是否添加成功 |
| HotelManager.getHotelManagerInfo | 语法 | Public HotelManagerVO getHotelManagerInfo(String hotelID); |
| 前置条件 | 需要获得酒店工作人员信息 |
| 后置条件 | 查找数据返回酒店工作人员信息 |
| HotelManager.updateHotelManagerInfo | 语法 | publicResultMessageupdateHotelManagerInfo (HotelManagerVO hotelManagerInfo) |
| 前置条件 | 已知酒店ID，需要更改的酒店工作人员信息 |
| 后置条件 | 更新数据并返回是否修改成功 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Data层 | |  |
| HotelManagerDao.getHotelManagerInfo(Stirng hotel\_ID) | | 根据ID查找酒店管理人员的信息 |
| HotelManagerDao.addHotelManager(HotelManagerPO hotelManager) | | 新增单一持久化对象：酒店管理人员 |
| HotelManagerDao.updateHotelManagerInfo(HotelManagerPO hotelManager) | | 修改单一持久化对象：酒店管理人员的信息 |
|  | |  |

表4.1.2-3WebBusiness的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| WebBusiness.addWebBusiness | 语法 | Public ResultMessageaddWebBusiness(WebBusinessVO webBusinessInfo); |
| 前置条件 | 需要添加网站营销人员 |
| 后置条件 | 返回是否添加成功 |
| WebBusiness.getWebBusinessInfo | 语法 | publicboolean getWebBusinessInfo(String webBusiness\_ID); |
| 前置条件 | 需要获得网站营销人员信息 |
| 后置条件 | 返回网站营销人员信息 |
| WebBusiness.updateWebBusinessInfo | 语法 | Public ResultMessageupdateWebBusinessInfo(WebBusinessVO ,webBusinessInfo); |
| 前置条件 | 已知网站营销人员ID，需要修改网站营销人员信息 |
| 后置条件 | 返回是否修改成功 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| Data层 | |  |
| WebBusinessDao.getWebBusinessInfo(Stirng user\_ID) | | 根据ID查找网站管理人员的信息 |
| WebBusinessDao.addWebBusiness(WebBusinessPO webBusiness) | | 新增网站管理人员 |
| WebBusinessDao.updateWebBusinessInfo(WebBusinessPO webBusiness) | | 修改网站管理人员的信息 |

表4.1.2-4WebManager的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| WebManager.addHotelManager | 语法 | Public ResultMessage addHotelManager (HotelManagerVO hotelManager); |
| 前置条件 | 已添加酒店，需要添加对应的酒店工作人员信息 |
| 后置条件 | 调用HotelManger类的addHotelManager方法 |
| WebManager.getHotelManagerInfo | 语法 | Public HotelManagerVO getHotelManagerInfo(String hotelID); |
| 前置条件 | 需要获得酒店工作人员信息 |
| 后置条件 | 调用HotelManger类的getHotelManagerInfo方法 |
| WebManager.updateHotelManagerInfo | 语法 | publicResultMessage updateHotelManagerInfo (HotelManagerVO hotelManagerInfo) |
| 前置条件 | 已知酒店ID，需要更改的酒店工作人员信息 |
| 后置条件 | 调用HotelManger类的updateHotelManagerInfo方法 |
| WebManager.addWebBusiness | 语法 | Public ResultMessage addWebBusiness(WebBusinessVO webBusinessInfo); |
| 前置条件 | 需要添加网站营销人员 |
| 后置条件 | 调用WebBusiness类的addWebBusiness方法 |
| WebManager.getWebBusinessInfo | 语法 | publicboolean getWebBusinessInfo(String webBusiness\_ID); |
| 前置条件 | 需要获得网站营销人员信息 |
| 后置条件 | 调用WebBusiness类的addWebBusiness方法 |
| WebManager.updateWebBusinessInfo | 语法 | Public ResultMessagegetWebBusinessInfo(WebBusinessVO ,webBusinessInfo); |
| 前置条件 | 已知网站营销人员ID，需要修改网站营销人员信息 |
| 后置条件 | 调用WebBusiness类的updateWebBusinessInfo方法 |
| 需要的服务（需接口） | | |
| 服务名 | | 服务 |
| WebBusiness.addWebBusiness | | 添加网站营销人员 |
| WebBusiness.getWebBusinessInfo | | 获取网站营销人员的信息 |
| WebBusiness.updateWebBusinessInfo | | 修改网站营销人员的信息 |
| HotelManager.addHotelManager | | 添加酒店管理人员 |
| HotelManager.getHotelManagerInfo | | 获取酒店管理人员的信息 |
| HotelManager. updateHotelManagerInfo | | 修改酒店管理人员的信息 |

**（4）动态模型**

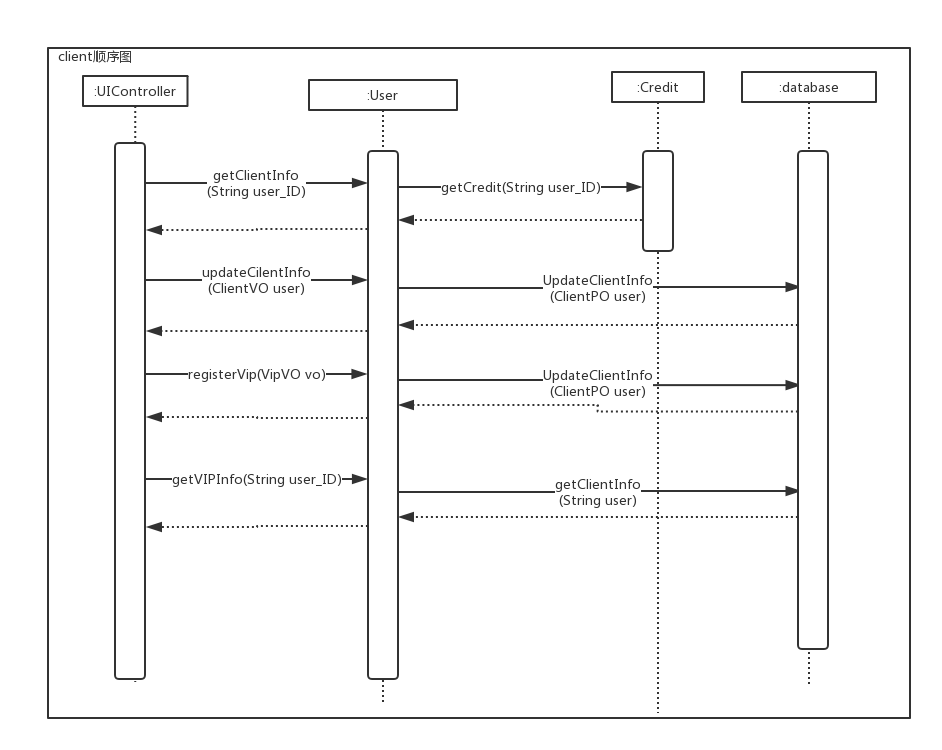


图4.1.2-2 client顺序图

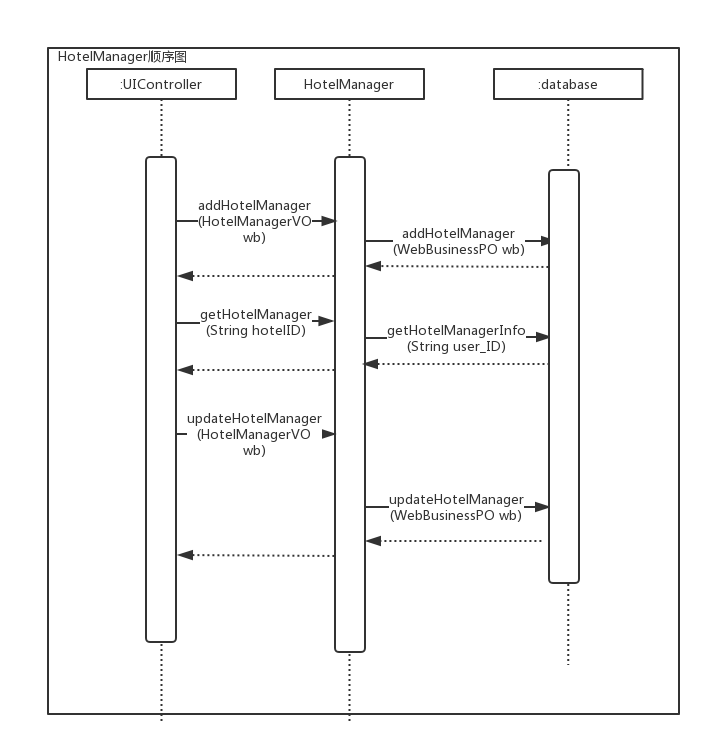


图4.1.2-3 hotelmanager顺序图

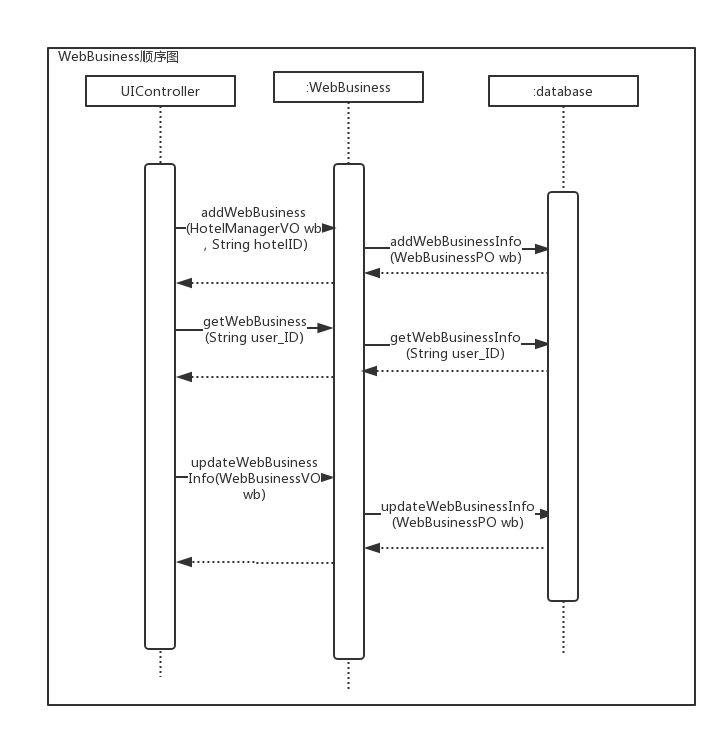


图4.1.2-4 网站营销人员顺序图

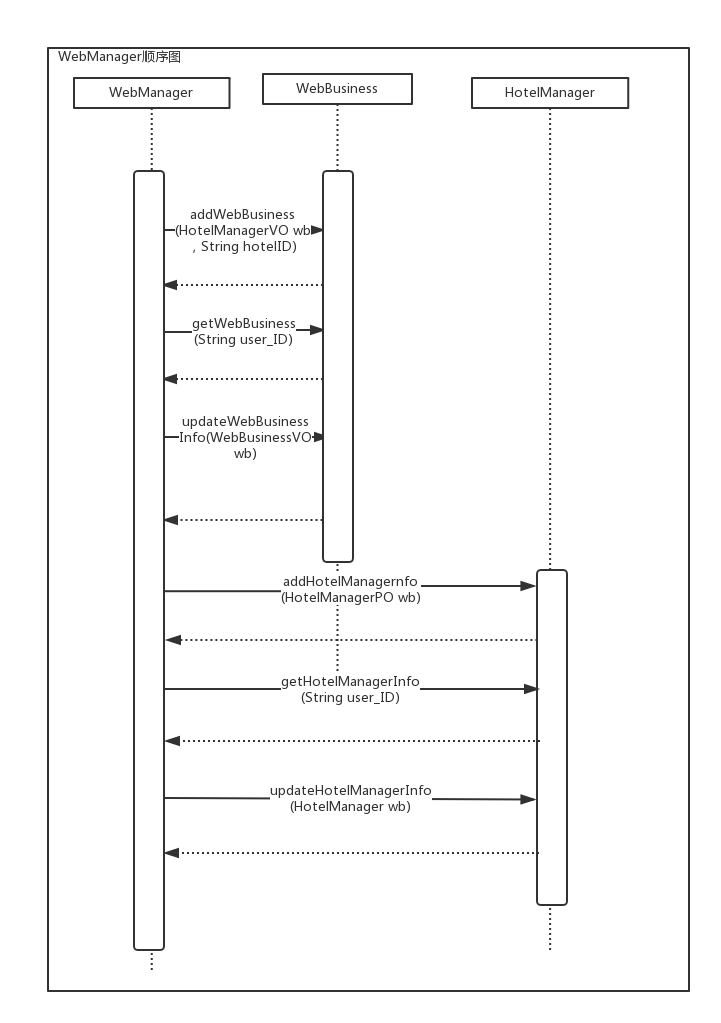
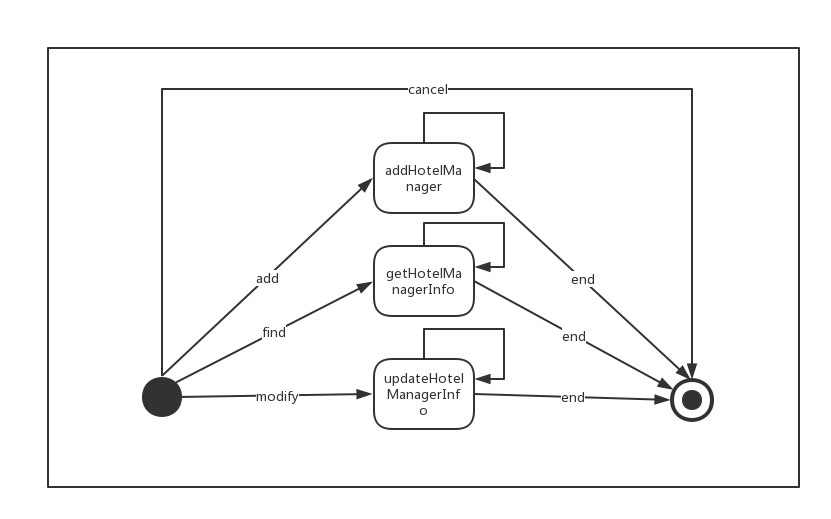


图4.1.2-5 网站管理人员顺序图

图4.1.2-5 酒店工作人员信息管理状态图

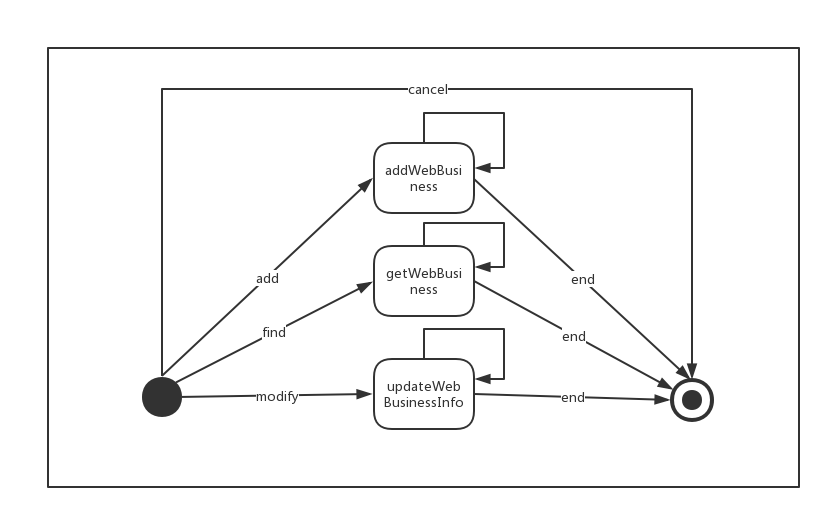


图4.1.2-6 网站管理人员信息管理状态图

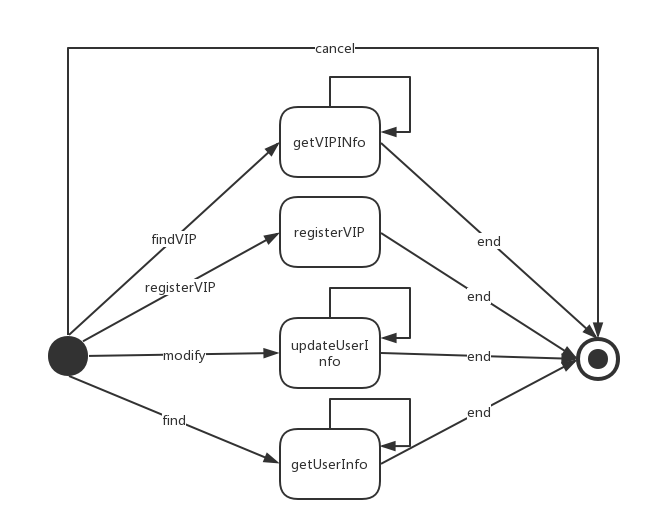


图4.1.2-7 客户信息管理状态图

### 4.1.3．accountbl模块

**（1）模块概述、**

accountbl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

accountbl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AccounService,AccountDao两个接口。在用户注册的的同时，需要初始化用户的基本信息，添加在数据中，所以需要调用user包的AddClientInfo接口。

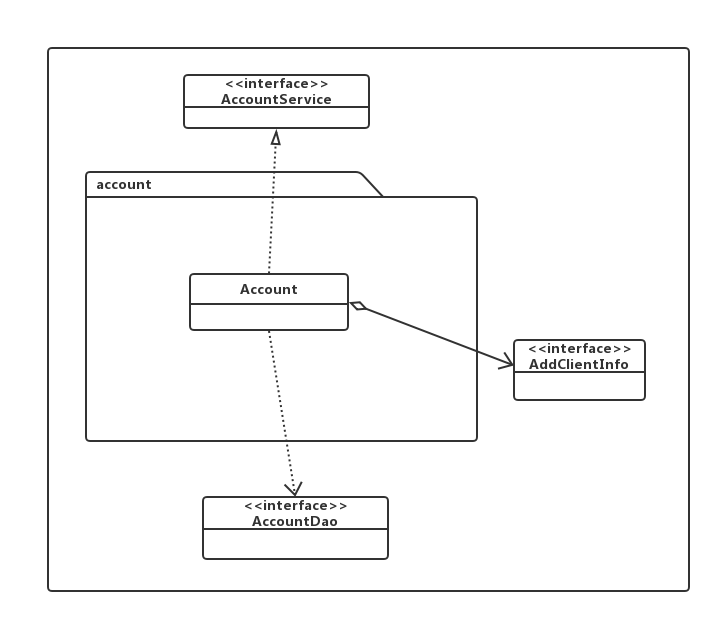
****

图4.1.6-1 account模块类的设计

**（3）模块内部类的接口规范**

表4.1.3-1Account的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Account.register | 语法 | publicboolean register(AccountVO account); | |
| 前置条件 | 用户没有账号 | |
| 后置条件 | 返回注册结果，增加一条帐号记录 | |
| Account.login | 语法 | Public Boolean loginr(AccountVO account); | |
| 前置条件 | 用户输入用户名和密码，希望登录 | |
| 后置条件 | 返回登录成功，更改帐号在线状态 | |
| Account.logout | 语法 | Public Boolean logout (String order\_id); | |
| 前置条件 | 用户处于登录状态，希望退出登录 | |
| 后置条件 | 返回登出结果，更改帐号在线状态 | |
| Account.modifyPassword | 语法 | Public boolean modifyPassword(AccountVO account); | |
| 前置条件 | 已获得需要修改帐号信息 | |
| 后置条件 | 返回修改结果，更新帐号信息 | |
| Account.getIdentity | 语法 | Publicstring getIdentity (String userID); | |
| 前置条件 | 已知帐号信息 | |
| 后置条件 | 查询数据返回帐号身份 | |
| Account.userIDExist | 语法 | Public boolean userIDExist (String userID); | |
| 前置条件 | 用户注册，确定用户名是否合法 | |
| 后置条件 | 查询数据返回帐号是否存在 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Data层 | | |  |
| AccountDao.addAccount | | | 添加新注册的账户 |
| AccountDao.updateAccount | | | 更新帐户信息 |
| AccountDao.getAccountInfo | | | 获取账户信息 |
| AccountDao.userIDExists | | | 判断用户名是否存在 |
| AccountDao.hasLogin | | | 判断用户是否处于登录状态，不能在不同地方同时登录一个账号 |
| AccountDao.setLogin | | | 登录后设置登录状态 |
| AccountDao.setLogout | | | 登出后设置登出状态 |
|  | | |  |

**（4）动态模型**

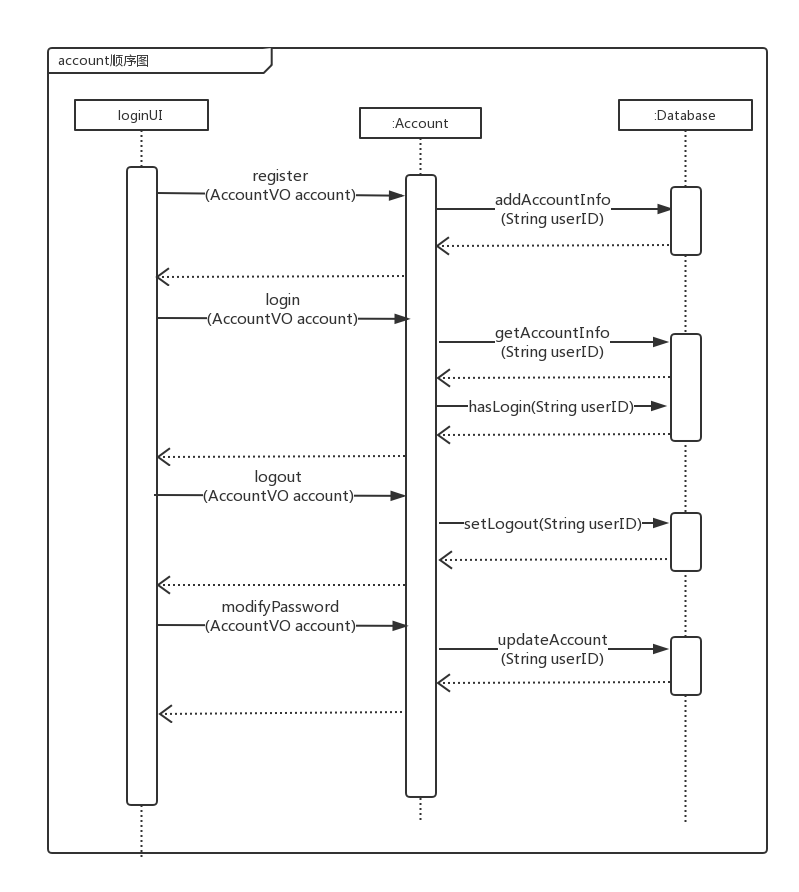
****

图4.1.3-2account模块的顺序图

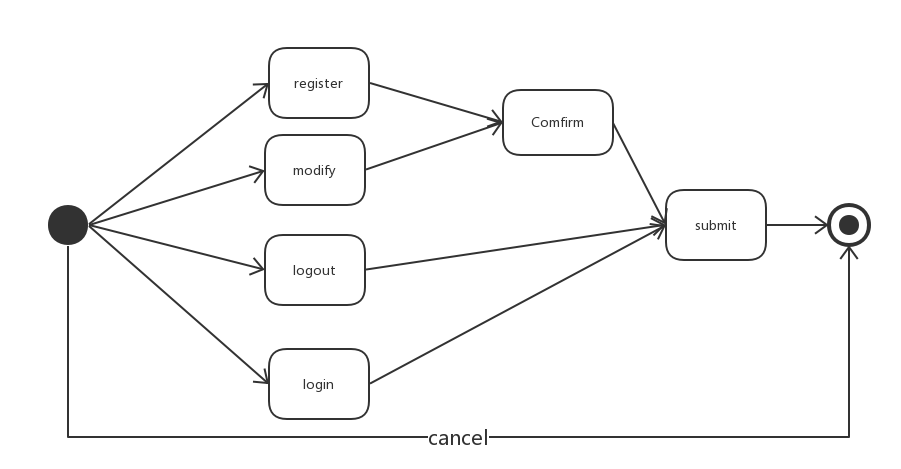


图4.1.3-4Account状态图

### 4.1.4．roombl模块

**（1）模块概述、**

roombl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

roombl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，在ui和logic添加了RoomService接口，在logic和data层添加了RoomDao接口。 Room处理管理房间信息查看，管理，更新。UpdateRoom类是为了方便order模块在订单执行时更新房间调用而存在的一个衍生类。

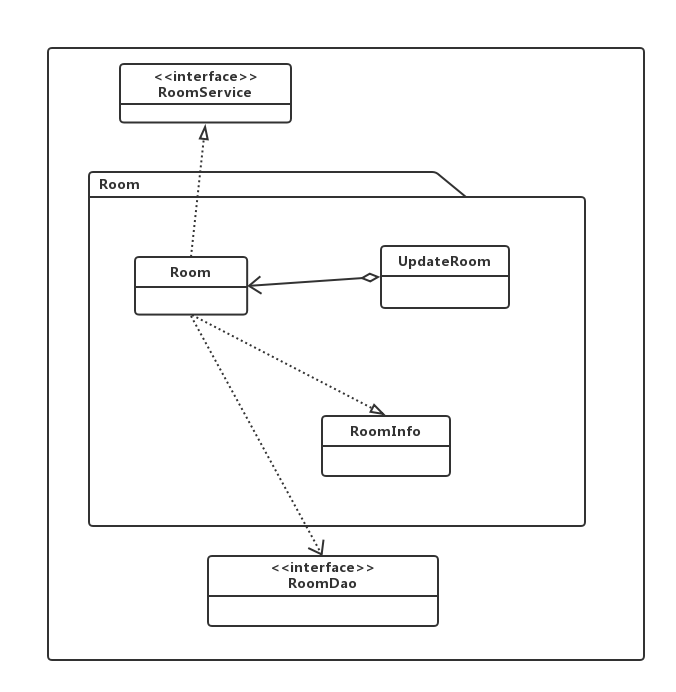
****

图4.1.4-1 room模块类的设计

**（3）模块内部类的接口规范**

表4.1.4-1room的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Room. addRoomInfo | 语法 | Public ResultMessage addRoomInfo(RoomVO room); | |
| 前置条件 | 已获取要增加的房间信息 | |
| 后置条件 | 返回增加结果，增加一个房间信息 | |
| Room. updateRoomInfo | 语法 | Public ResultMessage updateRoomInfo(RoomVO room); | |
| 前置条件 | 已获取要更新的房间信息 | |
| 后置条件 | 返回更新成功与否，更改房间信息 | |
| Room. getRoomInfo | 语法 | PublicRoomVO getRoomInfo(String hotelID, RoomType roomType); | |
| 前置条件 | 酒店管理人员已经登录，想要获取酒店的房间信息 | |
| 后置条件 | 返回房间信息 | |
| Room. getRoomList | 语法 | publicArrayList<RoomVo room>getRoomList(String hotelID); | |
| 前置条件 | 已知酒店id | |
| 后置条件 | 查找信息并返回改酒店的所有房间信息 | |
| Room. getRemainingRoomNums | 语法 | Public int getRemainingRoomNums(String hotelID , RoomType roomType); | |
| 前置条件 | 酒店管理人员已经登录，想要获得所有对应类型的房间的空房数量 | |
| 后置条件 | 时间设置为当天，调用getSpecificTimeRemainingRoomNums方法 | |
| Room. getSpecificTimeRemainingRoomNums | 语法 | Public int getSpecificTimeRemainingRoomNums (String hotelID , RoomType roomType， String time); | |
| 前置条件 | 酒店管理人员已经登录，想要获得所有对应类型的房间特定日期的空房数量 | |
| 后置条件 | 查找信息并返回改酒店对应类型房间特定日期的剩余数量 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Data层 | | |  |
| RoomDao.addRoom | | | 添加新录入的房间 |
| RoomDao.updateRoom | | | 更新房间信息 |
| RoomDao.getRoomInfo | | | 获取房间信息 |
| RoomDao.getHotelRooms | | | 获取指定酒店的所有房间信息 |

表4.1.4-2UpdateRoom的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UpdateRoom.UpdateRoomInSpecificTime | 语法 | Public ResultMessage UpdateRoomInSpecificTime (String hotelID, RoomType roomType, int changeNum, String time); | |
| 前置条件 | 订单执行成功或者退房成功 | |
| 后置条件 | 更改房间数量信息并返回成功与否 | |
| UpdateRoom.UpdateRoom | 语法 | Public ResultMessage UpdateRoom (String hotelID, RoomType roomType, int changeNum); | |
| 前置条件 | 订单执行成功或者退房成功 | |
| 后置条件 | 时间设置为当天，调用UpdateRoomInSpecificTime方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Room. getRoomInfo | | | 获取酒店指定类型房间信息 |
| Room. updateRoom | | | 更新指定酒店指定类型房间信息 |

**（4）动态模型**

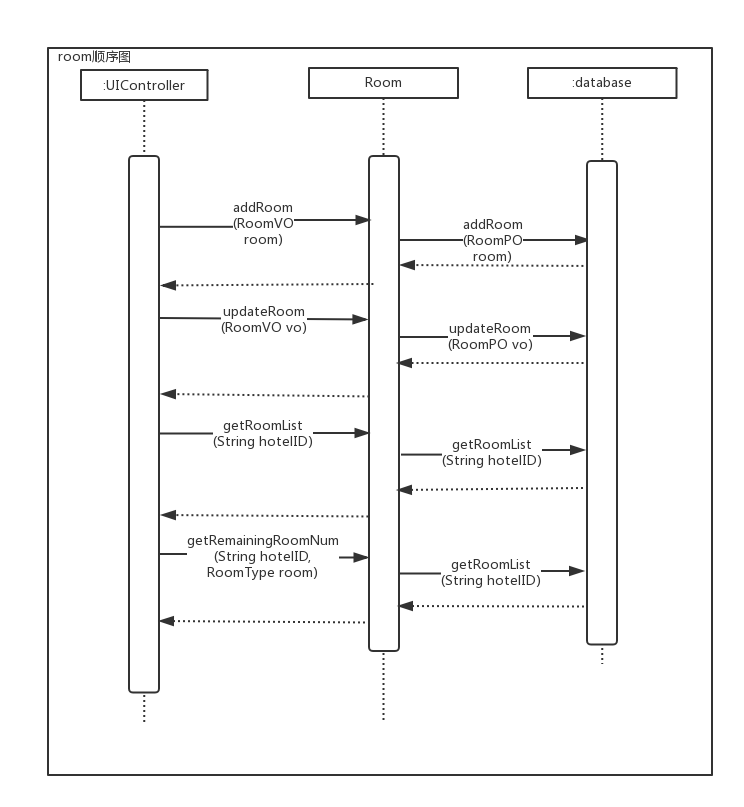
****

图4.1.4-2 room的顺序图

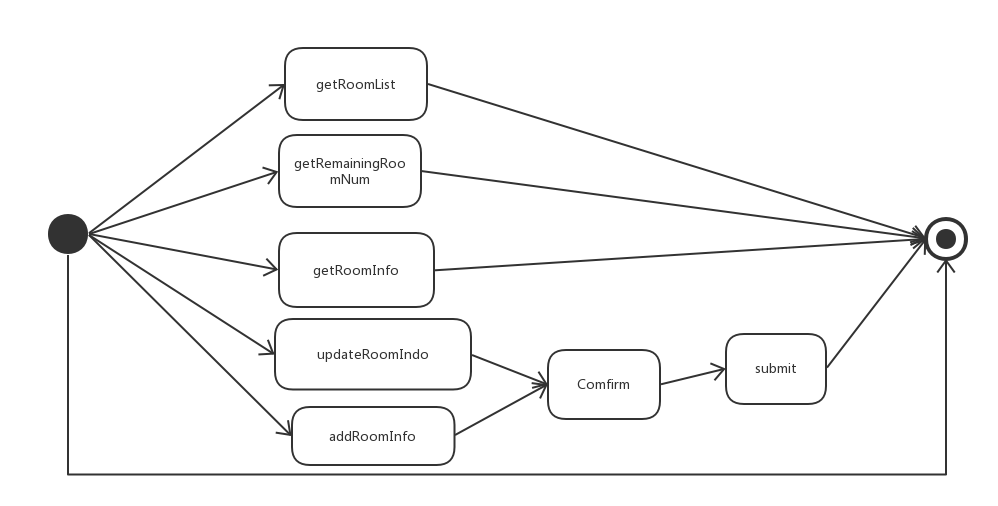


图4.1.4-3 room的状态图

### 4.1.5．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非功能需求

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，在ui和logic添加了CheckHotelService, SearchHotelService,UpdateHotelService三个接口，在data和logic添加了HotelDao接口。CheckHotelService处理酒店信息和评价信息的查看，SearchHotelService处理酒店列表的排序、搜索和筛选，UpdateHotelService 处理酒店信息的添加和更新。在处理酒店列表排序时，采用策略模式，不同排序方式的类均实现HotelSort接口，实现排序的逻辑方法。SearchHotel类中持有HotelSort对象，调用对应具体实现类的方法。不同HotelSort的实现类的对象的创建利用工厂模式，提高可修改性。 同时排序升序和降序，利用UpOrDownSort接口，利用 Up和Down两个实现类来确定。HotelPO是做为管理酒店信息的持久化对象被添加到设计模型中的。

在获取酒店评价信息时，需要用到获取该酒店所有订单信息的评价信息需要用到order包OrderHotelInfo。酒店信息管理需要管理酒店房间，需要用到room包RoomInfo接口。

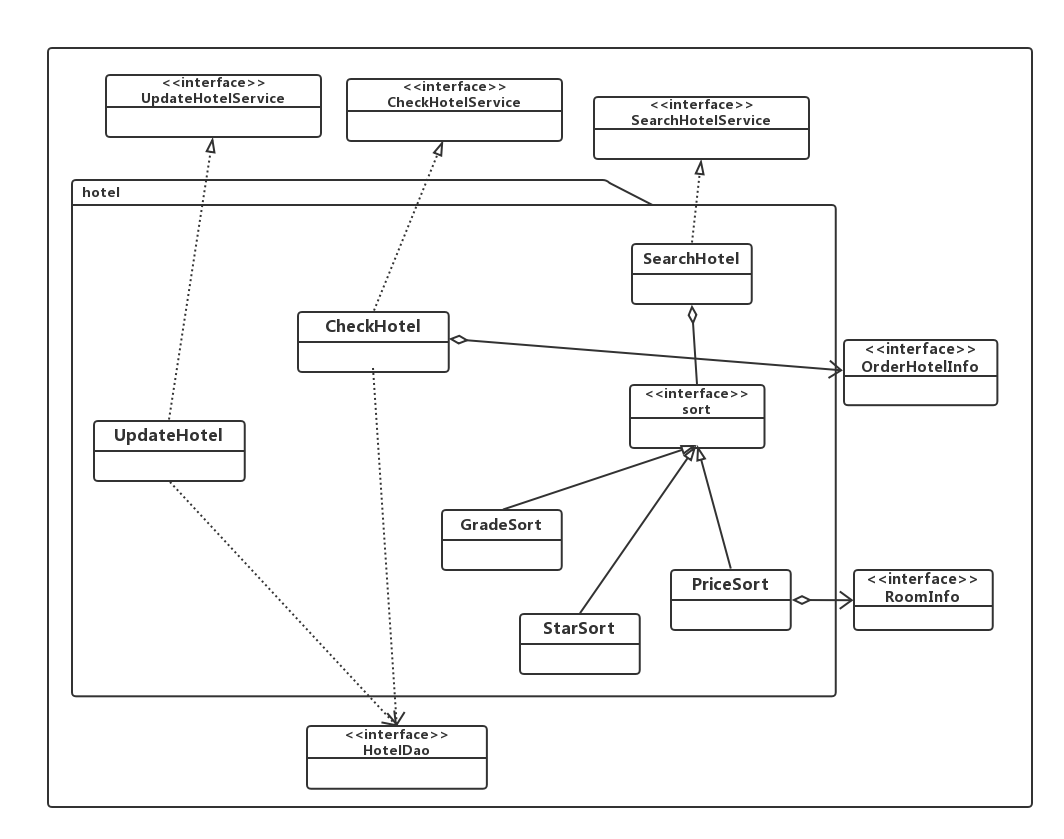
****

图4.1.5-1 hotel模块类的设计

**（3）模块内部类的接口规范**

表4.1.5-1SearchHotel的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| SearchHotel.getTradingArea | 语法 | publicArrayList<String> getTradingArea (String city); | |
| 前置条件 | 已获取城市名，需要获取城市对应的商圈列表 | |
| 后置条件 | 查看数据并返回商圈列表 | |
| SearchHotel.getInitialHotelList | 语法 | publicArrayList<HotelInfoVO>getInitialHotelList (String location,String tradingArea ); | |
| 前置条件 | 已知地址，商圈 | |
| 后置条件 | 返回该地址该商圈的酒店列表 | |
| SearchHotel.getSortedList | 语法 | publicArrayList<HotelInfoVO>getSortedList (HotelSearchCondition condition, ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知排序条件 | |
| 后置条件 | 返回符合条件的酒店信息 | |
| SearchHotel.getBookedHotelList | 语法 | PublicArrayList<HotelInfoVO>getBookedHotelList (String user\_id); | |
| 前置条件 | 用户已经登录，已知用户id | |
| 后置条件 | 返回预定过的酒店列表 | |
| SearchHotel.search | 语法 | PublicHotelInfoVO search(HotelSearchVO search); | |
| 前置条件 | 已知搜索条件 | |
| 后置条件 | 返回符合条件的酒店信息 | |
| SearchHotel.getCities | 语法 | publicArrayList<String>getCities (); | |
| 前置条件 | 用户已经登录想要搜索酒店 | |
| 后置条件 | 返回所有城市列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderHotelInfo.getHotelEvalutions | | | 获取酒店评价信息 |
| OrderHotelInfo.getBookedHotelList | | | 获取酒店评价信息 |
| PriceSort.getSortedList | | | 按价钱排序 |
| StarSort.getSortedList | | | 按星级排序 |
| GradeSortSort.getSortedList | | | 按评分排序 |
| Data层 | | |  |
| HotelDao.searchHotelList | | | 获取酒店列表 |
| HotelDao.getHotelInfoByHotelID | | | 获取酒店详细 |
| HotelDao.addHotel | | | 添加酒店信息 |
| HotelDao.updateHotel | | | 更改酒店信息 |
| HotelDao.getCities | | | 获取所有城市 |
| HotelDao.getTradingAreas | | | 获取指定城市的商圈列表 |

表4.1.5-2PriceSort的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| PriceSort.getSortedList | 语法 | publicArrayList<HotelInfoVO>getSortedList (ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合价格排序条件的列表 | |
| PriceSort.getSpecificSectionHotelList | 语法 | publicArrayList<HotelInfoVO>getSortedList (double startPrice, double endPrice, ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合价格区间条件的酒店列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UpOrDownSort | | | 升序或降序 |
| RoomInfo.getHotelLowestPrice | | | 获取酒店房间的最低价格 |

表4.1.5-3StarSort的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| StarSort.getSortedList | 语法 | publicArrayList<HotelInfoVO>getSortedList (ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合星级排序条件的列表 | |
| StarSort.getSpecificSectionHotelList | 语法 | publicArrayList<HotelInfoVO>getSortedList (double start, double end, ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合星级区间条件的酒店列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UpOrDownSort | | | 升序或降序 |

表4.1.5-4 GradeSort的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| GradeSort.getSortedList | 语法 | publicArrayList<HotelInfoVO>getSortedList (ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合评分排序条件的列表 | |
| GradeSort.getSpecificSectionHotelList | 语法 | publicArrayList<HotelInfoVO>getSortedList (double start, double end, ArrayList<HotelVO> hotels); | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合评分区间条件的酒店列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UpOrDownSort | | | 升序或降序 |

表4.1.5-5 UpOrDownSort的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UpOrDownSort.compare | 语法 | public boolean compare(double a, double b) | |
| 前置条件 | 已知初始酒店列表 | |
| 后置条件 | 返回符合星级排序条件的列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
|  | | |  |

表4.1.5-6CheckHotel的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CheckHotel.getHotelInfo | 语法 | PublicHotelVO getHotelInfo (String hotelID); | |
| 前置条件 | 已知酒店名 | |
| 后置条件 | 查看数据返回该酒店信息 | |
| CheckHotel.getHotelTradingArea | 语法 | Public StringgetHotelTradingArea(String hotel\_id);; | |
| 前置条件 | 促销策略需要获取酒店的商圈信息 | |
| 后置条件 | 返回该酒店的商圈信息 | |
| CheckHotel.getHotelEvalutaions | 语法 | PublicArrayList<EvalutionVO>getHotelEvalutaions (Hotel hotel); | |
| 前置条件 | 客户查看酒店评价信息 | |
| 后置条件 | 返回酒店的评价信息 | |
| CheckHotel.getRoomPrice | 语法 | Public double getRoomPrice (String hotelID ,RoomType roomType); | |
| 前置条件 | 生成订单时需要确定订单的原始价格 | |
| 后置条件 | 调用room类的getRoomPrice的方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| RoomInfo.getRoomPrice | | | 获得指定酒店指定房间类型的价格 |
| OrderHotelInfo.getHotelEvalutions | | | 获取酒店评价信息 |
| OrderHotelInfo.getBookHotelList | | | 获取制定客户预订过的酒店的id |
| Data层 | | |  |
| HotelDao.getHotelByHotelId | | | 获取酒店详细 |

表4.1.5-7 UpdateHotel的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UpdateHotel.updateHotelInfo | 语法 | publicResultMessage updateHotelInfo (HotelInfoVO hotel); | |
| 前置条件 | 酒店信息需要更改 | |
| 后置条件 | 更新数据库该酒店信息 | |
| UpdateHotel.addHotel | 语法 | publicResultMessage addHotel (HotelInfoVO hotel); | |
| 前置条件 | 网站管理人员输入添加的酒店信息 | |
| 后置条件 | 更新数据并返回成功与否 | |
| UpdateHotel.hotelIDExist | 语法 | public boolean hotelIDExist (String hotelID); | |
| 前置条件 | 添加酒店信息判断酒店id是否存在 | |
| 后置条件 | 查看数据并返回是与否 | |
| UpdateHotel.hotelHasManager | 语法 | public boolean hotelHasManager (String hotelID); | |
| 前置条件 | 添加酒店管理人员判断酒店管理人员是否存在 | |
| 后置条件 | 查看数据并返回是与否 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Data | | |  |
| HotelDao.getHotelByHotelId | | | 获取酒店信息 |
| HotelDao.updateHotel | | | 更新酒店信息 |
| HotelDao.addHotel | | | 添加酒店信息 |

**（4）动态模型**

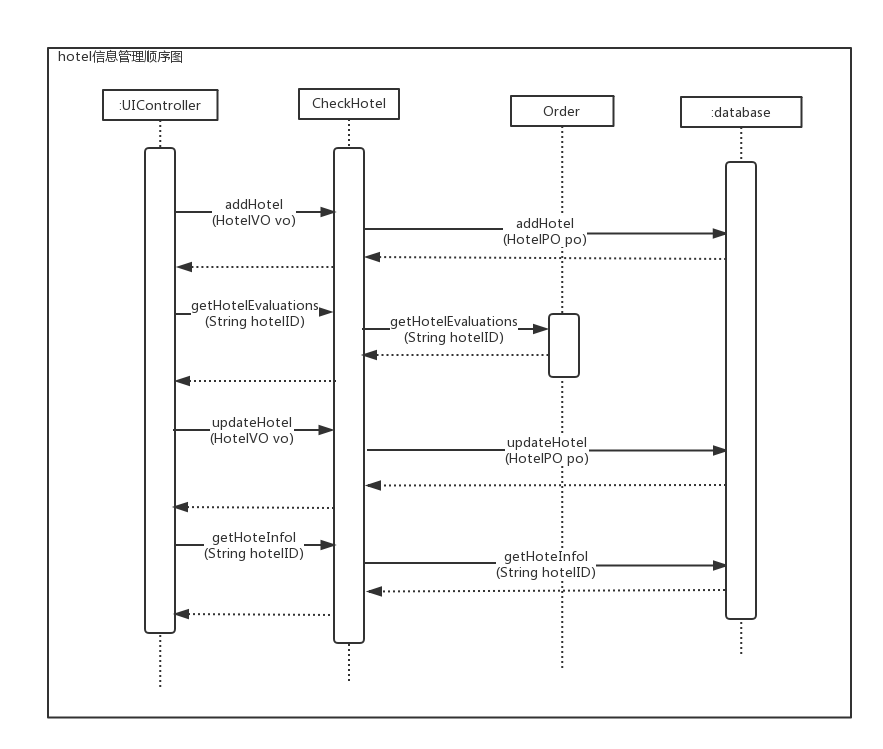
****

图4.1.5-2酒店信息顺序图

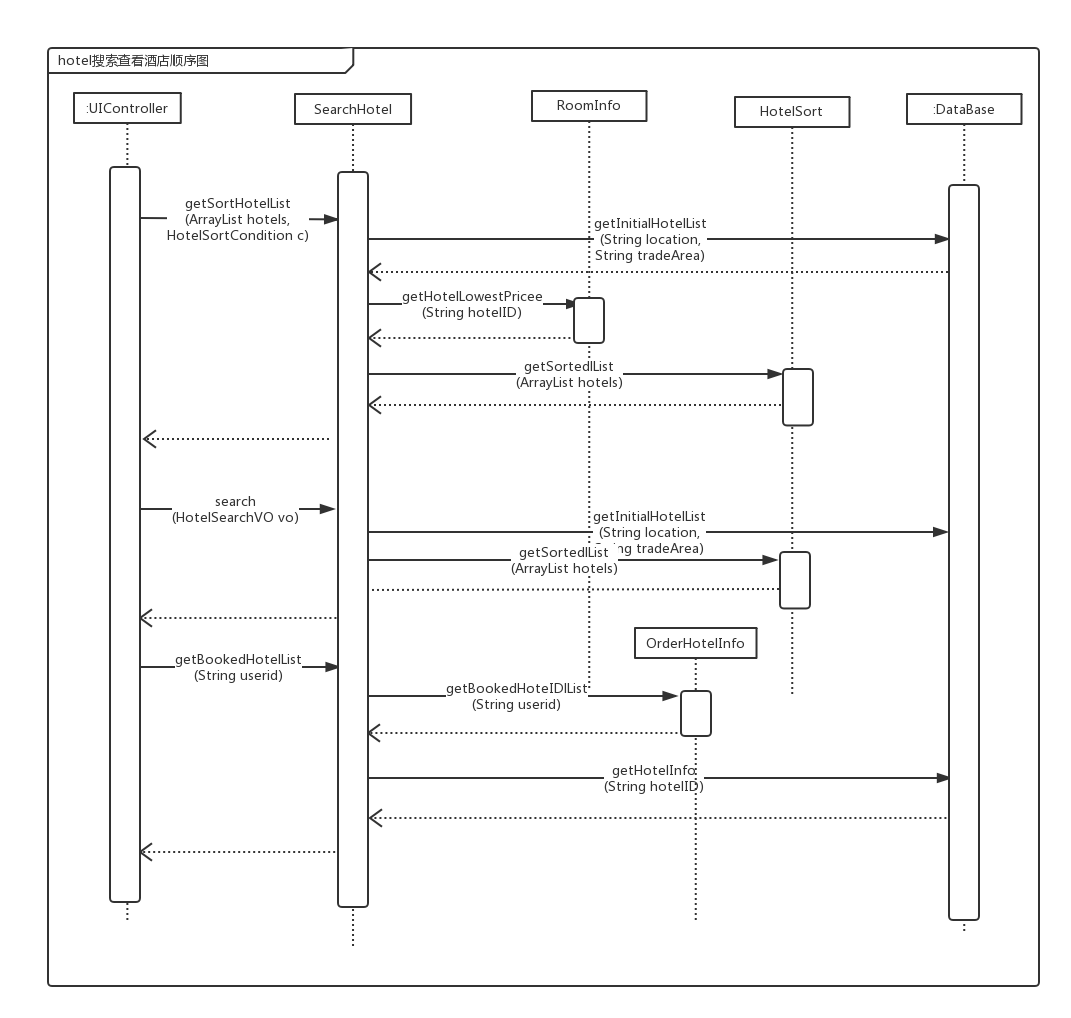


图4.1.5-3搜索酒店顺序图

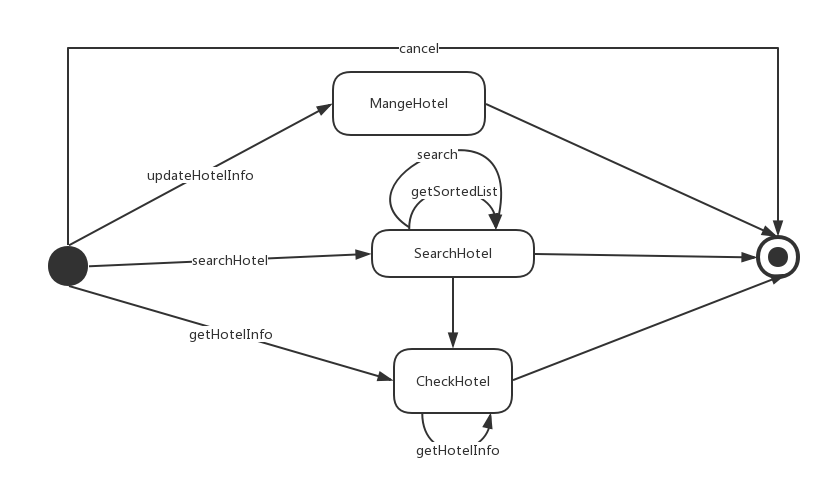


图4.1.5-4 hotel状态图

### 4.1.6．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非功能需求

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，在ui和logic添加了CreateOrderService,ExecuteOrderService ManageOrderService OrderListService ExecuteOrderService五个接口，在data和logic添加了OrderDao,OrderListDao两个个接口。OrderList类是为了筛选订单列表被添加在模型中，采用策略模式，每种筛选方法都是OrderListFilter的一个实现类，OrderList持有OrderListFilter的对象，筛选时调用其方法。OrderListFilter的实现类对象的创建利用工厂模式。OrderPO是做为管理信息的持久化对象被添加到设计模型中的。

在操作执行order的同时，需要更新房间，用到room包的RoomInfo接口，需要更新信用记录，用到credit包的CreditChangeInfo接口。在生成订单的时候， 需要计算促销信息，用到promotion包的CalculatePromotionInfo接口。

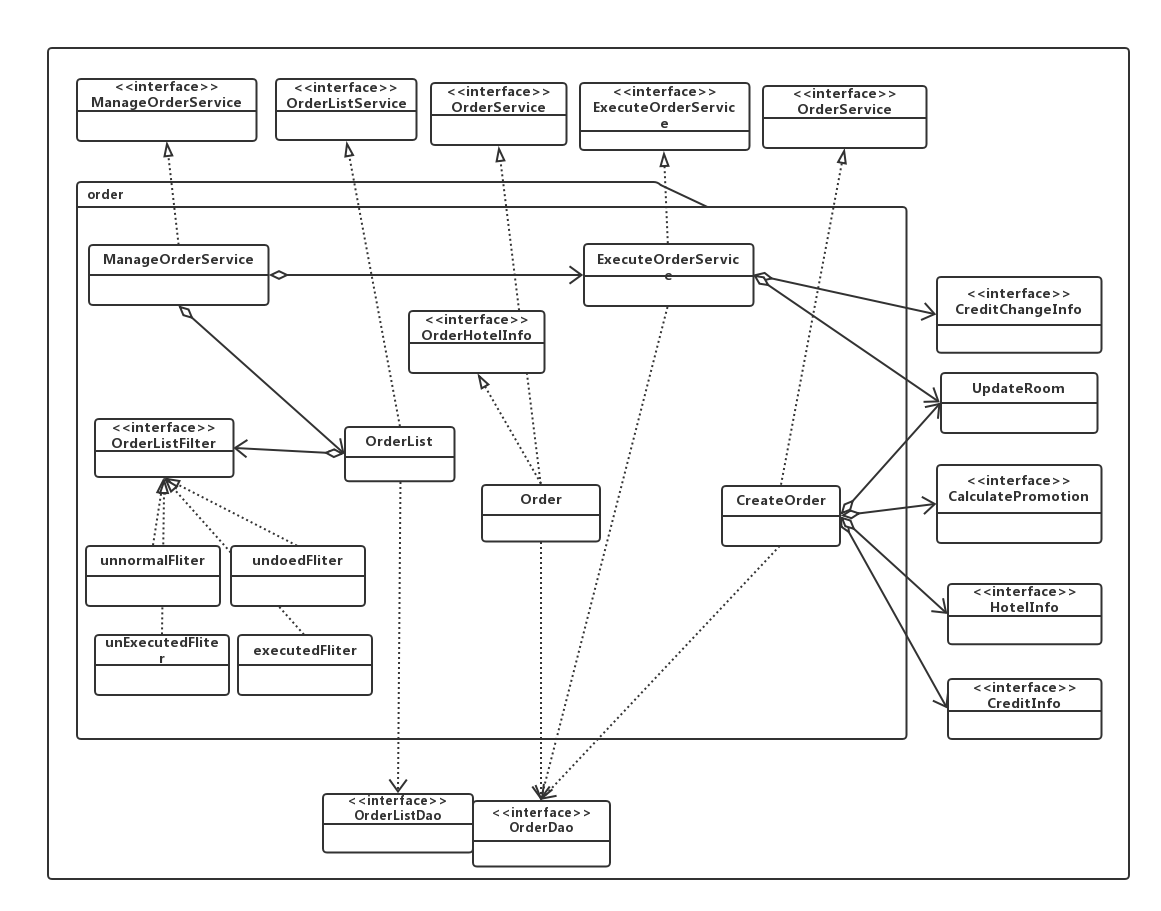
****

图4.1.6-1 order模块类的设计

**（3）模块内部类的接口规范**

表4.1.6-1 order的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Order.getOrderInfo | 语法 | public OrderVO getOrderInfo(String orderID); | |
| 前置条件 | 系统已创建订单 | |
| 后置条件 | 查询订单并返回订单信息 | |
| Order.getEvaluationInfo | 语法 | public EvaluationVO getEvaluationInfo(String orderID); | |
| 前置条件 | 客户对订单进行评价 | |
| 后置条件 | 查询评价并返回评价信息 | |
| Order.evaluate | 语法 | public ResultMessage evaluate(EvaluationVO evaluation); | |
| 前置条件 | 系统已创建订单 | |
| 后置条件 | 用户得到评价成功与否的结果 | |
| Order.getHotelEvaluations | 语法 | public ArrayList<EvaluationVO> getHotelEvaluations(String hotelID) | |
| 前置条件 | 酒店存在被评价记录 | |
| 后置条件 | 用户请求查看酒店评价 | |
| Order.getBookedHotelList | 语法 | public ArrayList<String> getBookedHotelList(String userID); | |
| 前置条件 | 被预定过酒店存在被评价记录 | |
| 后置条件 | 用户查询预定过酒店的评价记录 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Data层 | | |  |
| OrderDao.addEvaluation | | | 添加订单评价 |
| OrderDao.getOrderByOrderID | | | 获取订单信息 |
| OrderDao.getEvalutionByOrderID | | | 获取订单评价 |
| OrderDao.getBookedHotelList | | | 获取指定用户预订过的酒店id |

表4.1.6-2orderList的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| OrderList.filterUserOrderList | 语法 | public ArrayList<OrderVO> filterUserOrderList(String userID, OrderListCondition condition); | |
| 前置条件 | 用户有预定酒店的记录 | |
| 后置条件 | 网站营销人员查询今日订单 | |
| Order. filterHotelOrderList | 语法 | public ArrayList<OrderVO> filterHotelOrderList(String hotelID, OrderListCondition condition); | |
| 前置条件 | 用户有预定酒店的记录 | |
| 后置条件 | 酒店工作人员查询今日订单 | |
| Order. filterSpecificUserHotelOrderList | 语法 | public ArrayList<OrderVO> filterSpecificUserHotelOrderList(String userID, String hotelID); | |
| 前置条件 | 用户有预定酒店的历史纪录 | |
| 后置条件 | 用户查询历史订单记录 | |
| Order. filterWebDailyOrderList | 语法 | public ArrayList<OrderVO> filterWebDailyOrderList(OrderListCondition condition) | |
| 前置条件 | 用户预定酒店 | |
| 后置条件 | 网站营销人员查询当日订单 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Data | | |  |
| OrderListDao.getOrderListByUserID | | | 根据用户id获取订单 |
| OrderListDao.getOrderListByHotelID | | | 根据酒店id获取订单 |
| OrderListDao.getAllDailyOrders | | | 获取当天全部订单 |

表4.1.6-3 createOrder的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CreateOrder.createOrder | 语法 | PublicOrderVOcreateOrder(OrderVO order); | |
| 前置条件 | 用户已注册 | |
| 后置条件 | 用户下订单 | |
| Order. judgeCreditCanCreateOrder | 语法 | public boolean judgeCreditCanCreateOrder(String userID) | |
| 前置条件 | 用户已注册 | |
| 后置条件 | 用户下订单 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| CreditChange.changeCredit | | | 更新用户信用值和信用记录 |
| CalculatePromtionInfo.calculatePromotion | | | 生成订单时计算优惠信息 |
| UpdateRoom.updateRoom | | | 更新房间信息 |
| HotelInfo.getRoomPrice | | | 获取房间信息 |
| Data | | |  |
| OrderDao.getOrderNum | | | 获取订单次序，用于生成订单id |
| OrderDao.addOrder | | | 添加订单 |

表4.1.6-4ExecuteOrder的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ExecuteOrder. hasCheckOut | 语法 | public ResultMessage hasCheckOut(String orderID) | |
| 前置条件 | 用户退房 | |
| 后置条件 | 酒店管理人员设置退房信息 | |
| ExecuteOrder. checkOut | 语法 | public ResultMessage checkOut(String orderID) | |
| 前置条件 | 用户退房 | |
| 后置条件 | 酒店管理人员更新退房信息 | |
| ExecuteOrder. normalExecute | 语法 | public ResultMessage normalExecute(String orderID, String[] roomIDs); | |
| 前置条件 | 用户入住 | |
| 后置条件 | 酒店管理人员执行订单 | |
| ExecuteOrder. supplyOrder | 语法 | public ResultMessage supplyOrder(String orderID, String[] roomIDs); | |
| 前置条件 | 用户超时未入住 | |
| 后置条件 | 酒店管理人员补登记订单 | |
| ExecuteOrder. undoAbnormalOrder | 语法 | public ResultMessage undoAbnormalOrder(String orderID, boolean recoverAllDeletedCredit) | |
| 前置条件 | 用户超时未执行订单 | |
| 后置条件 | 网站营销人员撤销异常订单 | |
| ExecuteOrder. undoUnexecutedOrder | 语法 | public ResultMessage undoUnexecutedOrder(String orderID); | |
| 前置条件 | 用户已下订单，未执行 | |
| 后置条件 | 用户撤销订单 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| CreditChange.changeCredit | | | 更新用户信用值和信用记录 |
| UpdateRoom.updateRoom | | | 更新房间信息 |
| Data | | |  |
| OrderDao.getOrderByOrderID | | | 获取订单次序，用于生成订单id |
| OrderDao.updateOrder | | | 添加订单 |

表4.1.6-5 ManageOrder的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ManageOrder. undoAbnormalOrder | 语法 | public ResultMessage undoAbnormalOrder(String orderID, boolean recoverAllDeletedCredit) | |
| 前置条件 | 网站营销人员撤销异常订单 | |
| 后置条件 | 调用ExecuteOrder. undoAbnormalOrder方法 | |
| ManageOrder. getWebDailyUnexecutedOrderList | 语法 | public ArrayList<OrderVO> getWebDailyUnexecutedOrderList() | |
| 前置条件 | 网站营销人员浏览今日未执行订单 | |
| 后置条件 | 调用Order. filterWebDailyOrderList方法 | |
| ManageOrder.getWebDailyAbnormalOrderList | 语法 | public ArrayList<OrderVO>getWebDailyAbnormalOrderList() | |
| 前置条件 | 网站营销人员浏览今日异常订单 | |
| 后置条件 | 调用Order. filterWebDailyOrderList方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| ExecuteOrder.undoAbnormalOrder | | | 撤销异常订单 |
| OrderList.filterWebDailyOrderList | | | 筛选当天全部订单 |

表4.1.6-6 OrderListFilter的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| OrderListFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 需要筛选指定条件的订单 | |
| 后置条件 | 返回订单列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |

表4.1.6-7 AbnormalFilter的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| AbnormalFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 有全部订单列表，需要筛选异常的订单 | |
| 后置条件 | 返回异常订单列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |

表4.1.6-7 ExecutedFilter的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ExecutedFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 有全部订单列表，需要筛选已执行的订单 | |
| 后置条件 | 返回已执行订单列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |

表4.1.6-7 UnexecutedFilter的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UnexecutedFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 有全部订单列表，需要筛选未执行的订单 | |
| 后置条件 | 返回未执行订单列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |

表4.1.6-7 UndoedFilter的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UndoedFilter.filterList | 语法 | Public ArrayList<OrderPO> filterList(ArrayList<OrderPO> orders) | |
| 前置条件 | 有全部订单列表，需要筛选撤销未执行的订单 | |
| 后置条件 | 返回撤销未执行订单列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |

**（4）动态模型**

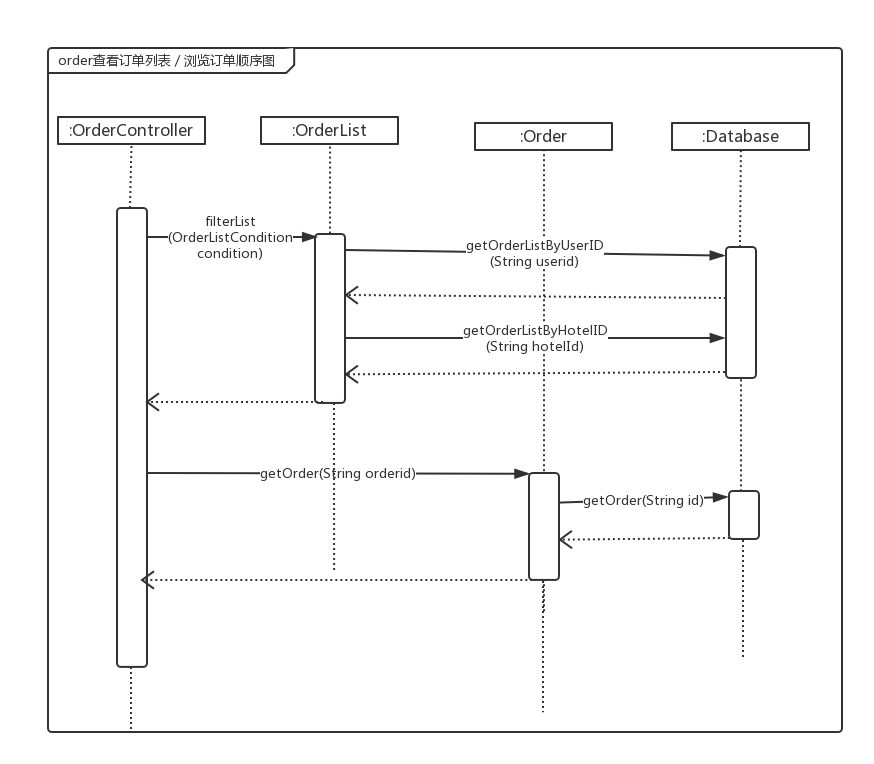
****

图4.1.6-2 浏览查看订单顺序图

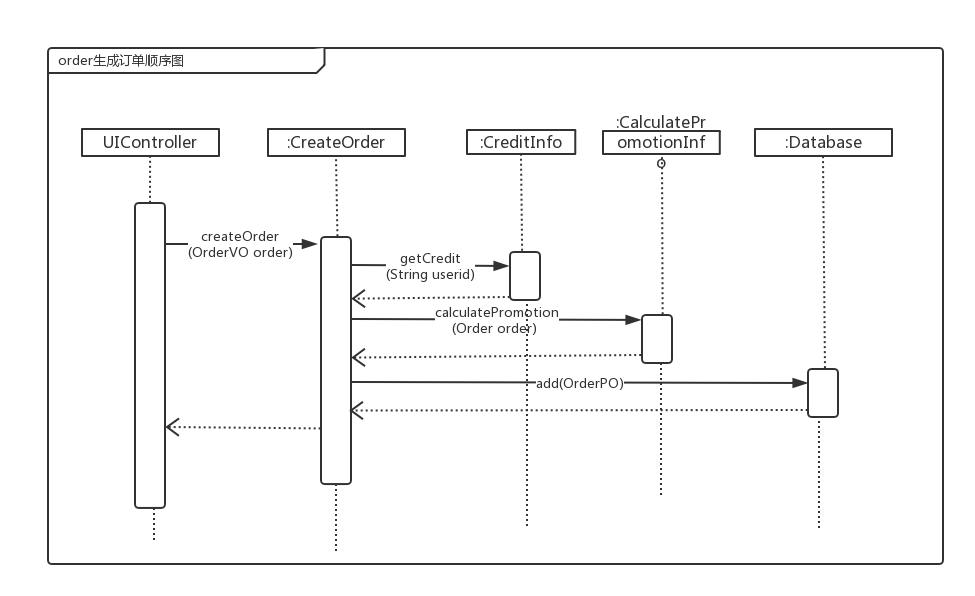
****

图4.1.6-3生成订单顺序图

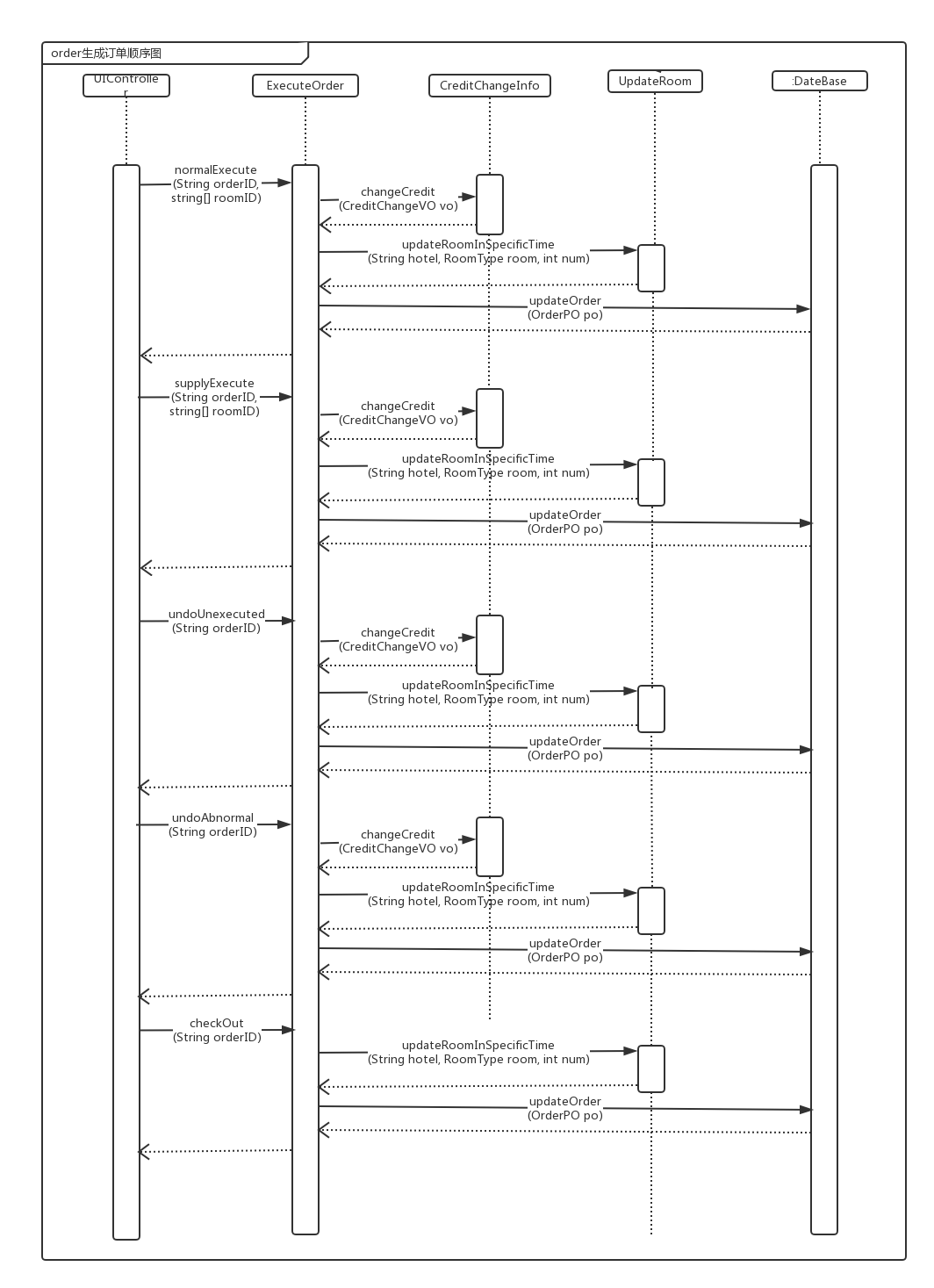


图4.1.6-4执行订单顺序图

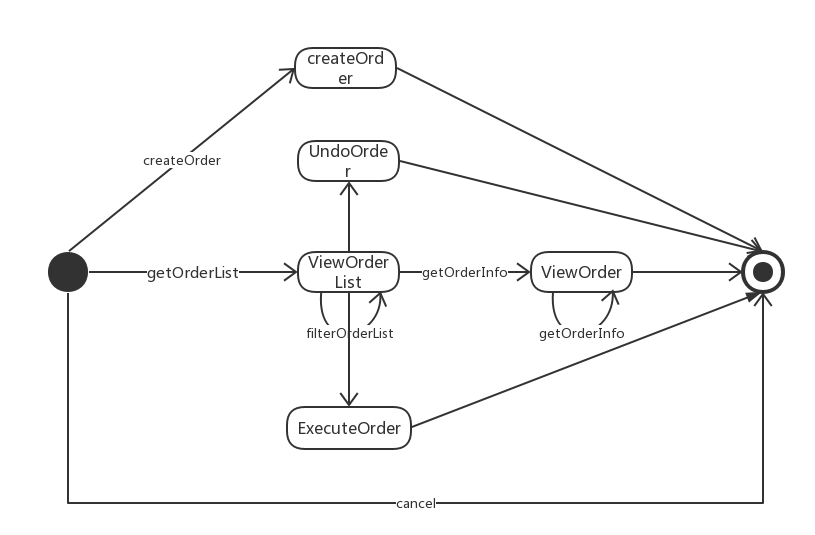


图4.1.6-5order状态图

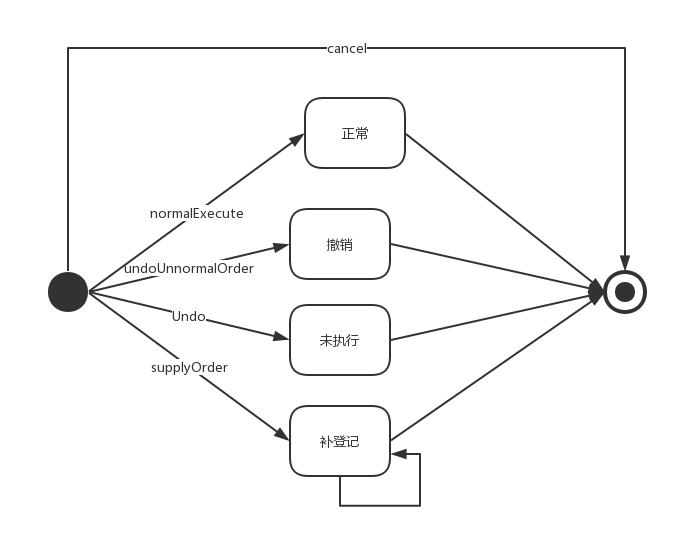


图4.1.6-5 order执行状态图

### 4.1.7．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非功能需求。

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，将系分为展示层，业务逻辑层，数据层。在ui和logic之间添加了ManagePromotionService接口，在logic和data之间添加了PromotionDao接口。CalculatePromotion类在订单生成时计算促销策略，采用策略模式，每种promotion的计算隐藏在promotion接口的实现类中，promotion接口的实现类的创建由工厂模式实现，方便以后添加新的促销策略。ManagePromotion处理促销策略的增删改查。

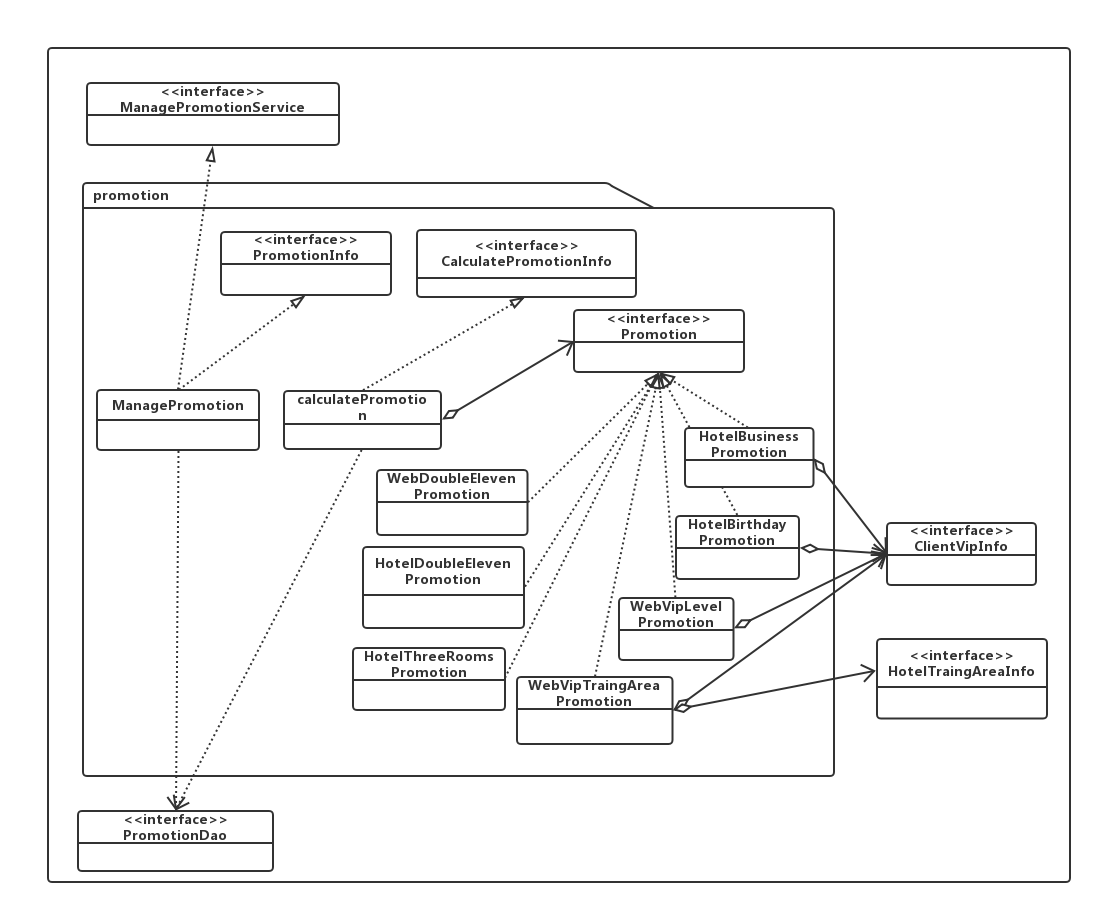


图4.1.7-1promotion模块类的设计

**（3）模块内部类的接口规范**

表4.1.7-1 ManagePromotion的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CalculatePromotion.addPromotion | 语法 | | public ResultMessage addPromotion(PromotionVO vo) |
| 前置条件 | | 网站营销人员和酒店工作人员添加促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion.getPromotion | 语法 | | public PromotionVO getPromotion(String promotionID) |
| 前置条件 | | 网站营销人员查看促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion. getHotelPromotions | 语法 | | public ArrayList<PromotionVO> getHotelPromotions(String hotelID, PromotionType promotionType) |
| 前置条件 | | 酒店工作人员查看酒店促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion. getWebPromotions | 语法 | | public ArrayList<PromotionVO> getWebPromotions(PromotionType promotionType) |
| 前置条件 | | 网站营销人员查看网站促销策略 |
| 后置条件 | | 无 |
| CalculatePromotion. deletePromotion | 语法 | | public ResultMessage deletePromotion(String promotionID) |
| 前置条件 | | 网站营销人员和酒店工作人员删除促销策略 |
| 后置条件 | | 无 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| Data层 | |  | |
| PromotionDao.getPromotionNum | | 获取promotion的次序 | |
| PromotionDao.addPromotion | | 添加promotion | |
| PromotionDao.getHotelPromotions | | 获取指定酒店的promotion列表 | |
| PromotionDao.getWebPromtions | | 获取网站promotions | |
| PromotionDao.deletePromotion | | 删除指定促销策略 | |

表4.1.7-2CalculatePromotion的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CalculatePromotion. calculatePromotion | 语法 | | public OrderVO calculatePromotion(OrderVO order) |
| 前置条件 | | 用户下订单 |
| 后置条件 | | 系统生成订单 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| HotelBirthdayPromotion.calculate | | 计算酒店会员生日折扣 | |
| HotelBusinessPromotino. calculate | | 计算酒店会员合作企业折扣 | |
| HotelDoubleElevenPromotion. calculate | | 计算酒店双11折扣 | |
| HotelThreeRoomsPromotion. calculate | | 计算酒店预订3间房间及以上折扣 | |
| WebDoubleElevenPromotion. calculate | | 计算网站双11折扣 | |
| WebVipLevelPromtion. calculate | | 计算网站会员等级折扣 | |
| WebVipTradingAreaPromotion. calculate | | 计算网站不同会员等级不同商圈不同折扣的优惠 | |
| Data层 | |  | |
| PromotionDao.getWebPromotions | | 获取网站所有促销信息 | |
| PromotionDao.getHotelPromotions | | 获取指定酒店促销促销信息 | |

表4.1.7-3 Promotion的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Promotion.judgePromotion | 语法 | | PublicBoolean judgePromotion(OrderVO orderVo); |
| 前置条件 | | 生成订单，需要计算促销 |
| 后置条件 | | 返回是否满足该促销的使用条件 |
| Promotion.calculate | 语法 | | Public doublecalculate\_Promotionid(Order order); |
| 前置条件 | | 生成订单，需要计算促销 |
| 后置条件 | | 计算订单促销策略后返回新的订单信息 |
| Promotion. changeToVO | 语法 | | public PromotionVO changeToVO() |
| 前置条件 | | 查看促销策略 |
| 后置条件 | | 生成促销策略内容返回新的策略信息 |
| 服务名 | | 服务 | |
| HotelBirthdayPromotion.judgePromtion | | 判断订单是否满足酒店会员生日折扣 | |
| HotelBirthdayPromotion.calculate | | 计算酒店会员生日折扣 | |
| HotelBusinessPromotino. judgePromtion | | 判断订单是否满足酒店会员合作企业折扣 | |
| HotelBusinessPromotino. calculate | | 计算酒店会员合作企业折扣 | |
| HotelDoubleElevenPromotion. judgePromtion | | 判断订单是否满足酒店双11折扣 | |
| HotelDoubleElevenPromotion. calculate | | 计算酒店双11折扣 | |
| HotelThreeRoomsPromotion. judgePromtion | | 判断订单是否满足酒店预订3间房间及以上折扣 | |
| HotelThreeRoomsPromotion. calculate | | 计算酒店预订3间房间及以上折扣 | |
| WebDoubleElevenPromotion. judgePromtion | | 判断订单是否满足网站双11折扣 | |
| WebDoubleElevenPromotion. calculate | | 计算网站双11折扣 | |
| WebVipLevelPromtion. judgePromtion | | 判断订单是否满足网站会员等级折扣 | |
| WebVipLevelPromtion. calculate | | 计算网站会员等级折扣 | |
| WebVipTradingAreaPromotion. judgePromtion | | 判断订单是否满足网站不同会员等级不同商圈不同折扣的优惠 | |
| WebVipTradingAreaPromotion. calculate | | 计算网站不同会员等级不同商圈不同折扣的优惠 | |
|  | |  | |

**（4）动态模型**

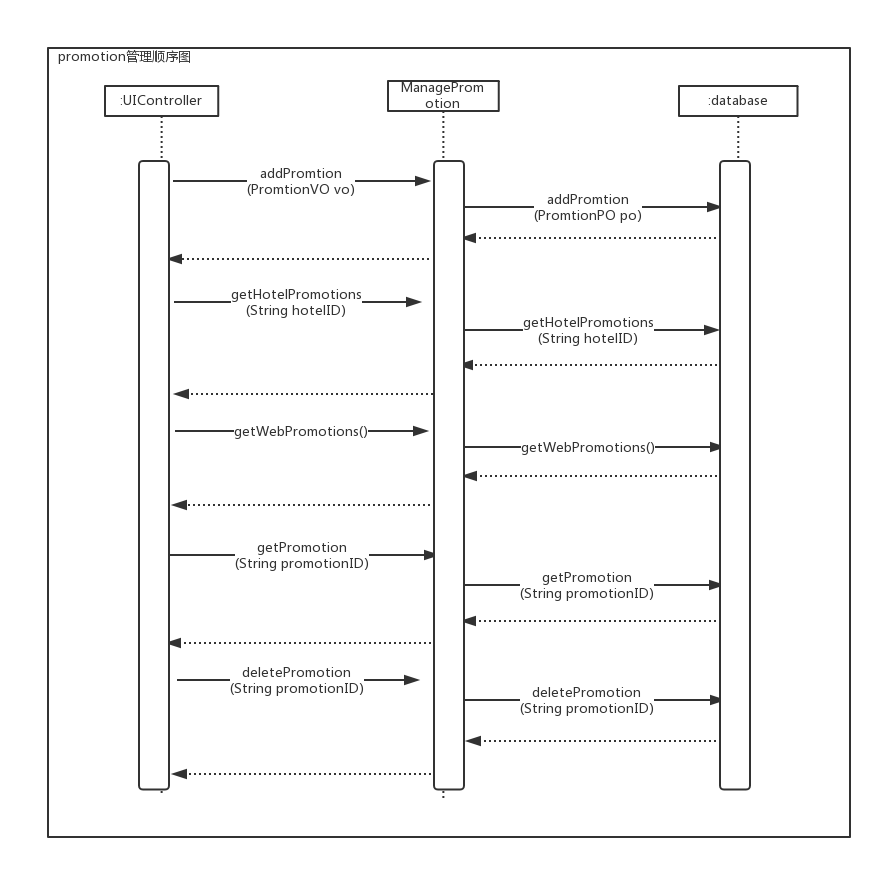


图4.1.9-2设置、增删改查网站、酒店营销策略的顺序图

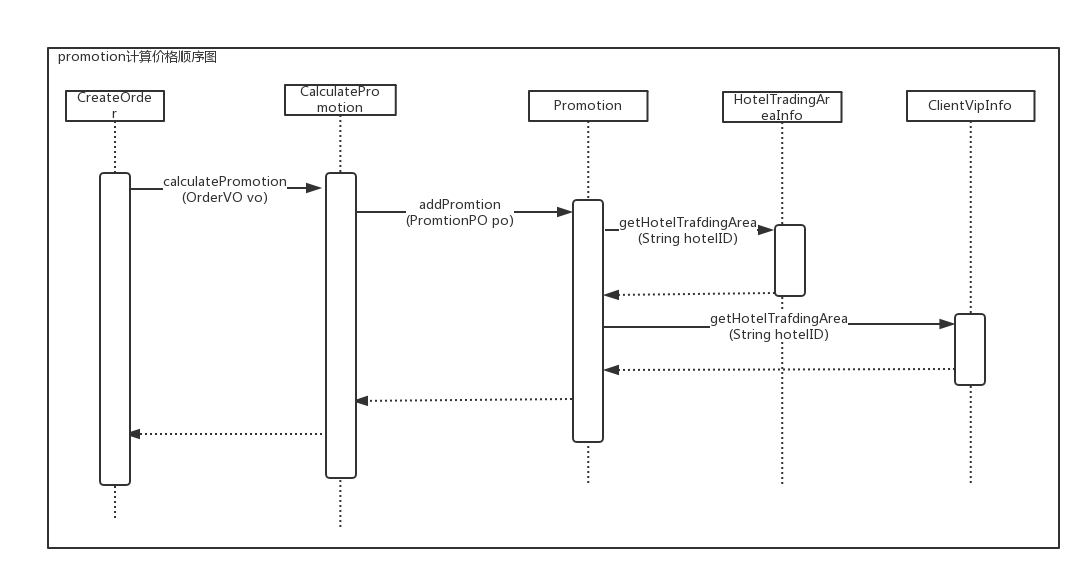


图4.1.9-3计算执行促销策略之后的订单价格的顺序图

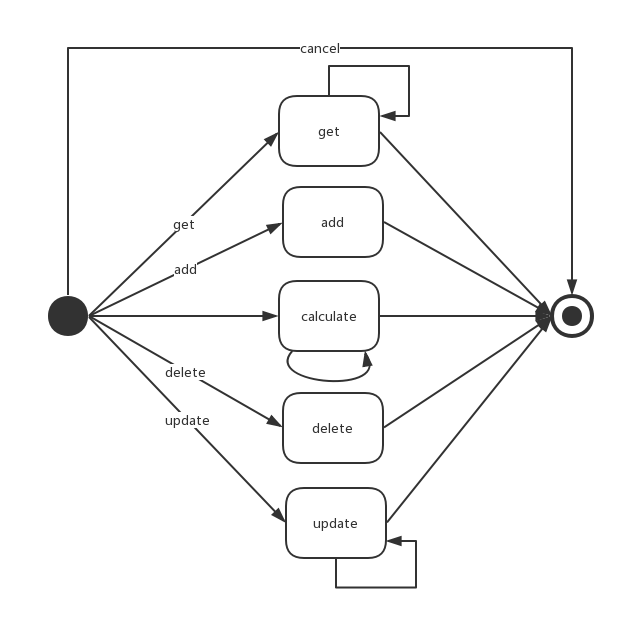


表4.1.9-4设置、返回增删改查酒店、网站营销策略的状态图

## 4.2. 界面展示层的分解

### 4.2.1界面层跳转

界面展示层主要分为控制跳转，传输、显示信息的controller类和布置组件以及监听的界面类。根据作业需求，界面类分为5个用户，游客、客户、酒店工作人员、网站营销人员、网站管理人员。对与每个用户有一个左侧的导航栏，并有对应的LeftController来控制点击左侧导航栏时，右侧界面的刷新，避免了真个界面的重复刷新。同时有继承左侧LeftController的不同子Controller来控制右侧界面单独的界面跳转和信息显示，同时将重要信息封装在父类Controller里避免了信息重复初始化。与逻辑层的交互也由所有controller类实现。

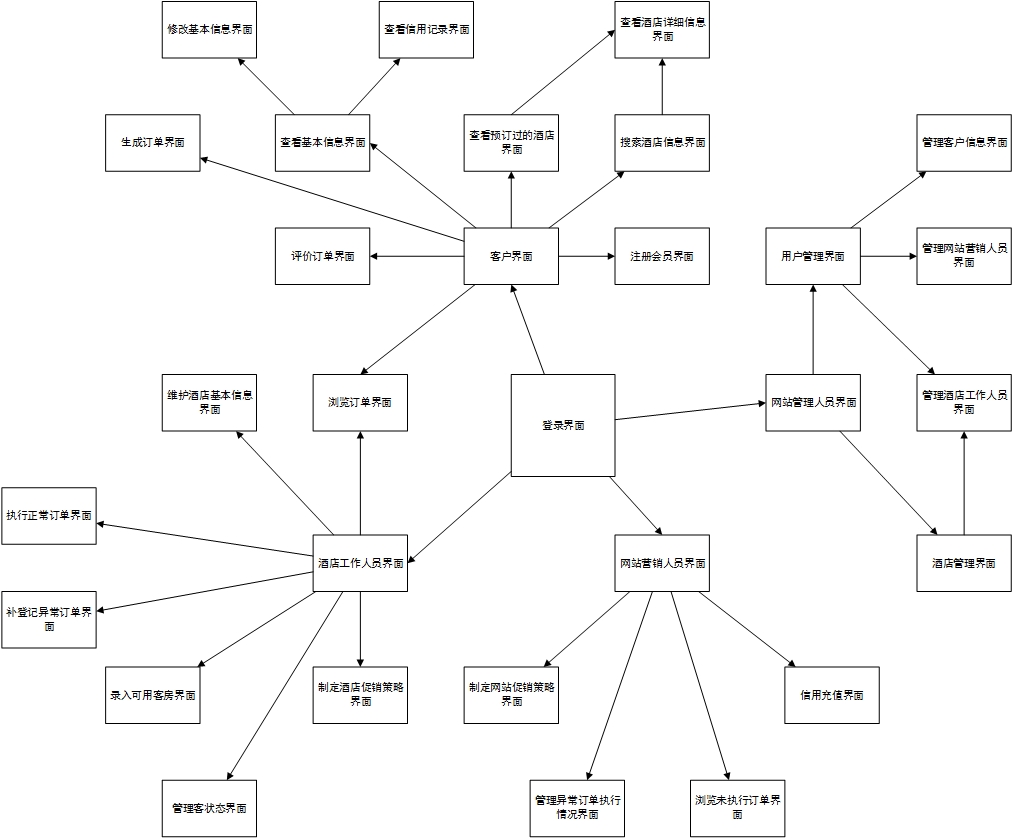


图4.2.1-1界面跳转示意图

### 4.2.2下面给出几个人员的重要跳转以作示例

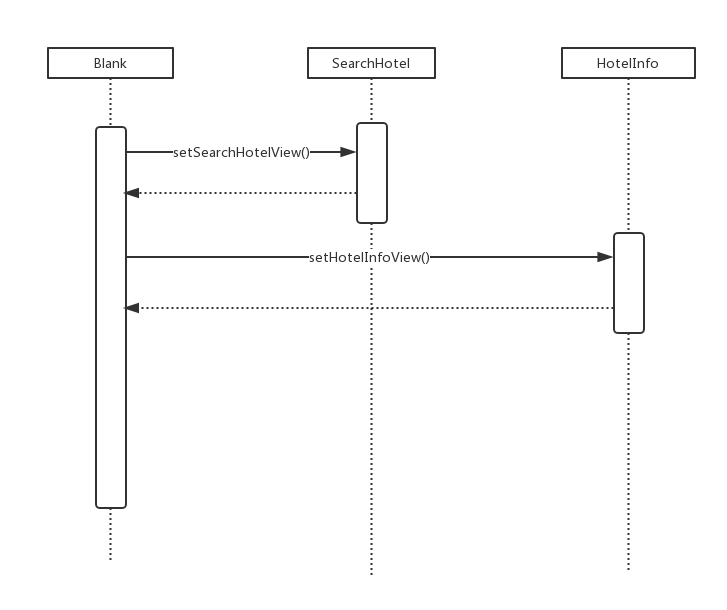


图4.2.2-1游客的界面层跳转示例

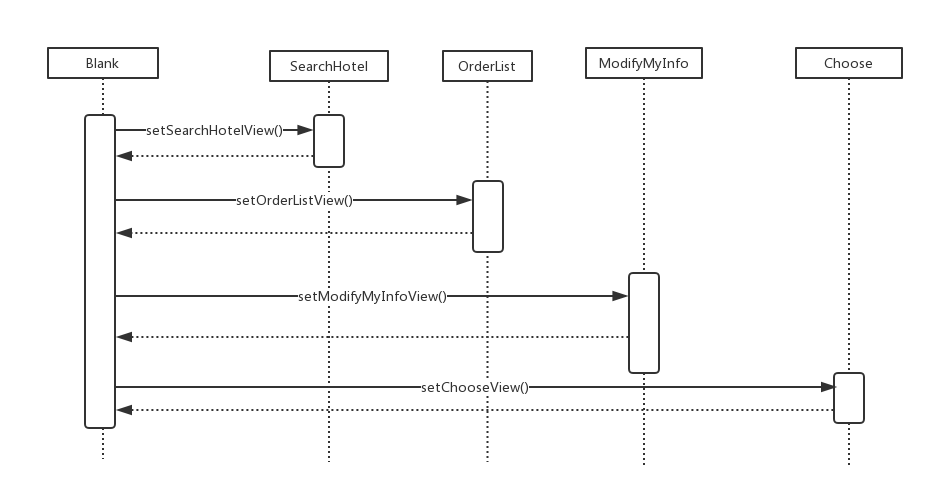


图4.2.2-2客户的界面层跳转示例

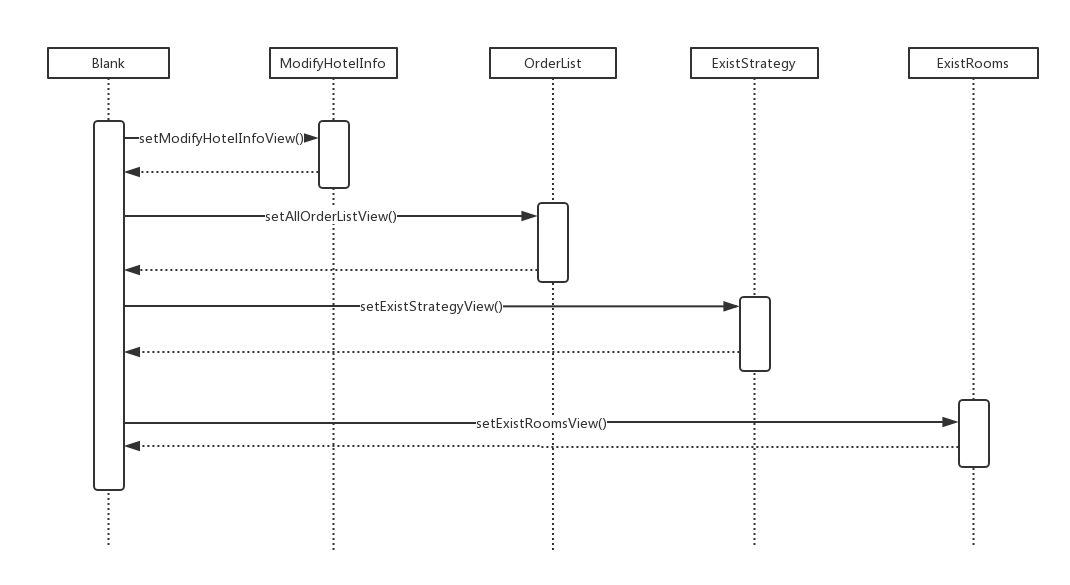


图4.2.2-3酒店工作人员的界面层跳转示例

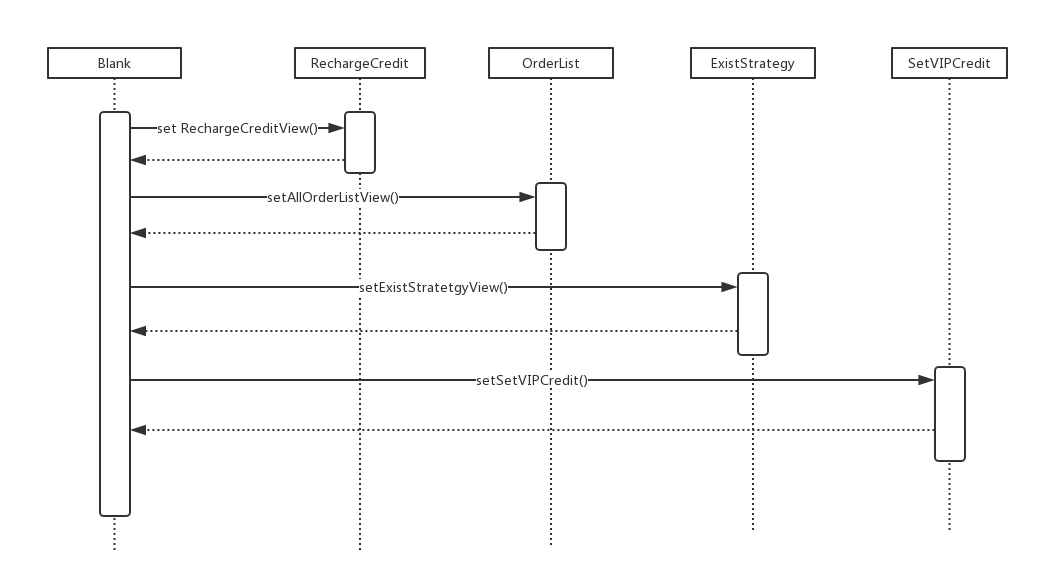


图4.2.2-4网站营销人员的界面层示例

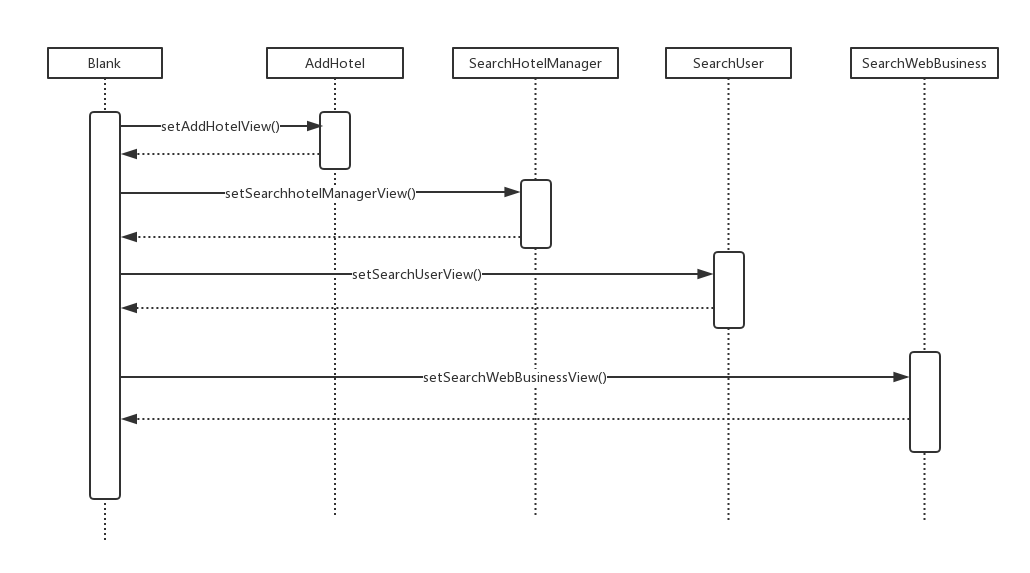
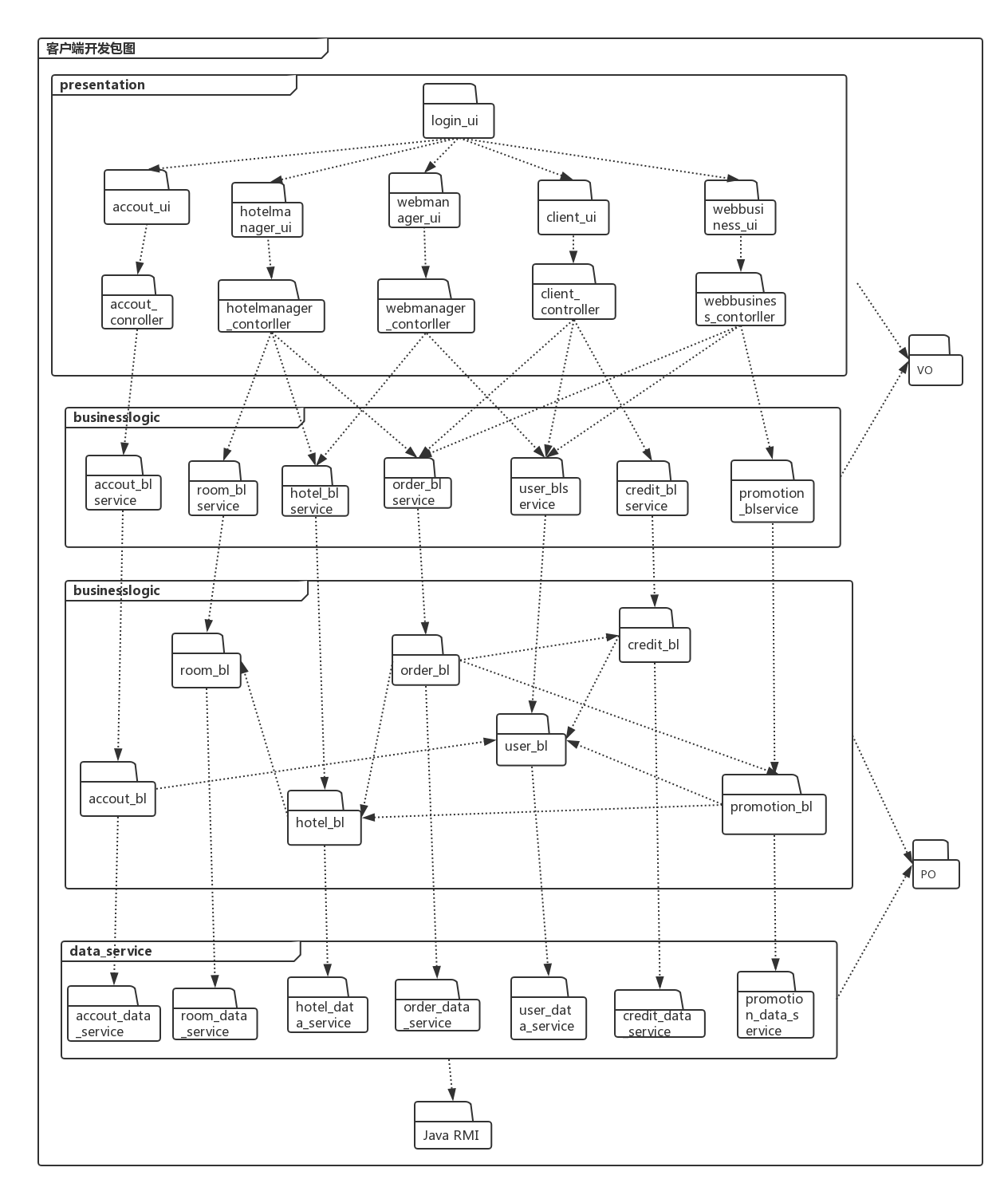


图4.2.2-5网站管理人员的界面层跳转示例

# 5.依赖视角



### 图5-1客户端开发包



### 图5-2 服务器端开发包