酒店管理系统

HMS (Hotel Management System)

详细设计文档

V2.1

钛4+小组 (Titanium Quartet)

蒋文荟 李佩瑶 李珍鸿 李一然

2016/10/29

目录

[更新历史 5](#_Toc465893040)

[一、引言 6](#_Toc465893041)

[1.1编制目的 6](#_Toc465893042)

[1.2词汇表 6](#_Toc465893043)

[二、产品概述 6](#_Toc465893044)

[三、体系结构设计概述 7](#_Toc465893045)

[四、结构视角 7](#_Toc465893046)

[4.1 usesrmainbl模块 7](#_Toc465893047)

[4.1.1 usermainbl模块概述 7](#_Toc465893048)

[4.1.2 usermainbl模块整体结构 7](#_Toc465893049)

[4.1.3 usermainbl模块内部类的接口规范 9](#_Toc465893050)

[4.1.4 业务逻辑层的动态模型 11](#_Toc465893051)

[4.2 personalbl模块 12](#_Toc465893052)

[4.2.1 personalbl模块概述 12](#_Toc465893053)

[4.2.2 personalbl模块整体结构 12](#_Toc465893054)

[4.2.3 personalbl模块内部类的接口规范 13](#_Toc465893055)

[4.2.4 业务逻辑层的动态模型 16](#_Toc465893056)

[4.3 createorderbl模块 19](#_Toc465893057)

[4.3.1 createorderbl模块概述 19](#_Toc465893058)

[4.3.2 createorder模块整体结构 19](#_Toc465893059)

[4.3.2 createorderbl模块内部类的接口规范 20](#_Toc465893060)

[4.3.4 业务逻辑层的动态模型 23](#_Toc465893061)

[4.4 hotelInfobl模块 24](#_Toc465893062)

[4.4.1 hotelInfobl模块概述 24](#_Toc465893063)

[4.4.2 hotelinfo模块整体结构 24](#_Toc465893064)

[4.4.3 hotelInfobl模块内部类的接口规范 25](#_Toc465893065)

[.4.4.4 业务逻辑层的动态模型 28](#_Toc465893066)

[4.5 searchhotelbl模块 29](#_Toc465893067)

[4.5.1 searchhotelbl模块概述 29](#_Toc465893068)

[4.5.2 searchhotelbl模块内部类的接口规范 30](#_Toc465893069)

[4.5.3 searchhotelbl模块的行为 32](#_Toc465893070)

[4.6 manageorderbl模块 33](#_Toc465893071)

[4.6.1 manageorderbl模块概述 33](#_Toc465893072)

[4.6.2 manageorderbl模块整体结构 34](#_Toc465893073)

[4.6.3 managorderbl模块内部类的接口规范 35](#_Toc465893074)

[4.6.4 业务逻辑层的动态模型 40](#_Toc465893075)

[4.7 manageroombl模块 41](#_Toc465893076)

[4.7.1 manageroombl模块概述 41](#_Toc465893077)

[4.7.2 manageroombl模块局部模块的接口规范 42](#_Toc465893078)

[4.7.3 manageroombl模块的行为 47](#_Toc465893079)

[4.8 strategybl模块 48](#_Toc465893080)

[4.8.1 strategybl模块概述 48](#_Toc465893081)

[4.8.2 strategybl模块局部模块的接口规范 49](#_Toc465893082)

[4.8.3 strategybl模块的行为 53](#_Toc465893083)

[4.9 managehotelbl模块 54](#_Toc465893084)

[4.9.1 managehotelbl模块概述 54](#_Toc465893085)

[4.9.2 managehotelbl模块内部类的接口规范 55](#_Toc465893086)

[4.9.3 managehotelbl模块的行为 58](#_Toc465893087)

[4.10 manageuserbl模块 59](#_Toc465893088)

[4.10.1 manageuserbl模块概述 59](#_Toc465893089)

[4.10.2 manageuserbl模块内部类的接口规范 60](#_Toc465893090)

[4.10.3 manageuserbl模块的行为 65](#_Toc465893091)

[五、依赖视角 66](#_Toc465893092)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 蒋文荟、李珍鸿、李佩瑶 | 2016/10/28 | 编写业务逻辑层各模块的设计类图、顺序图及状态图。 | V1.0 |
| 李珍鸿 | 2016/10/28 | 将各个模块合并为详细设计文档。 | V2.0 |
| 李一然 | 2016/10/29 | 增加文字描述，修正文档格式。 | V2.1 |

# 一、引言

## 1.1编制目的

本报告详细完成对酒店管理系统的详细设计，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

表1 详细设计文档词汇表

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| HMS | 酒店管理系统 |  |
| ui | 用户界面 |  |
| bl | 业务逻辑 |  |
| RMI | 远程方法调用 |  |
| PO | 持久化对象 |  |
| VO | 值对象 |  |

# 二、产品概述

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

# 三、体系结构设计概述

参考酒店管理系统体系结构设计文档对体系结构设计的概述。

# 四、结构视角

业务逻辑层包括多个针对界面的业务逻辑处理对象。业务逻辑层的开发包图如图12所示。

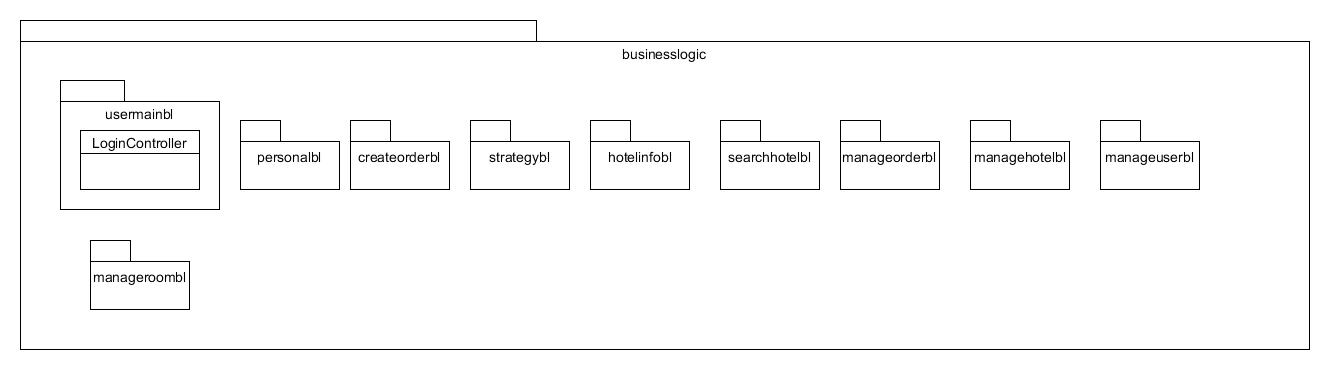


图1 业务逻辑层开发包图

## 4.1 usesrmainbl模块

### 4.1.1 usermainbl模块概述

usermainbl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。

usermainbl模块的职责及接口参加软件体系结构描述文档。

### 4.1.2 usermainbl模块整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了usermainblservice, userdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了UserMainController，这样UserMainController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给usermainbl 对象。UserPO是作为管理信息的持久化对象被添加到设计模型中的。

usermainbl模块的设计如图2所示:

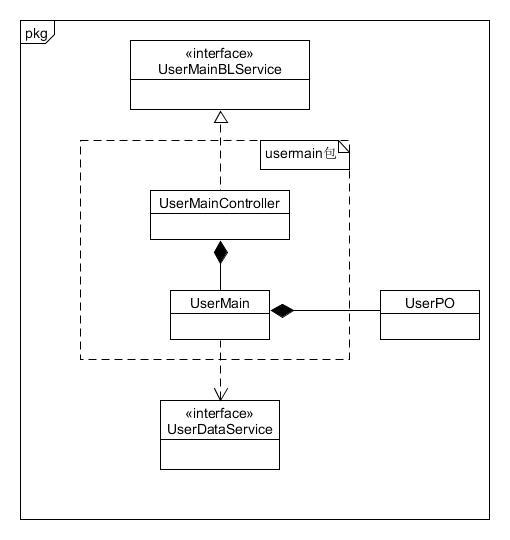


图2 usermainbl模块各个类的设计

usermainbl模块各个类的职责如表2所示：

表2 usermainbl 模块各个类的职责

|  |  |
| --- | --- |
| UserMainController | 负责用户的注册、登陆和登出 |
| UserMain | 帮助完成用户的注册、登陆和登出 |

### 4.1.3 usermainbl模块内部类的接口规范

表3 UsermainController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserMainController. login | 语法 | public UserVO login (String username, String password); | |
| 前置条件 | 用户名及密码符合要求。 | |
| 后置条件 | 根据用户名查找是否有相应的用户，再根据输入的密码返回登录验证的结果，若登录成功返回相应的用户值对象，否则返回null。 | |
| UserMainController. logout | 语法 | public ResultMessage logout (int userID); | |
| 前置条件 | 处于已登录状态。 | |
| 后置条件 | 返回操作结果，用户退出登录状态，客户端回到主登录界面。 | |
| UserMainController. isUnregistered | 语法 | public boolean isUnregistered (String username); | |
| 前置条件 | 正在注册新客户。 | |
| 后置条件 | 返回该用户名是否未被注册过。 | |
| UserMainController. signUp | 语法 | public ResultMessage signUp (String username, String password); | |
| 前置条件 | 输入的用户名和密码符合规则。 | |
| 后置条件 | 返回注册客户操作的结果，若注册成功切换至新客户的主界面。 | |
| UserMainController. currentUser | 语法 | public UserVO currentUser (); | |
| 前置条件 | 无。 | |
| 后置条件 | 返回当前用户相应的值对象，若当前客户端处于未登录状态则返回null。 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserMain. isUnregistered | | | 判断用户名是否已被使用 |
| UserMain. signUp | | | 用户注册 |
| UserMain. currentUser | | | 获得当前用户 |
| UserMain. login | | | 用户登陆 |
| UserMain. logout | | | 用户登出 |

表4 Usermainbl的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserMain. login | 语法 | public UserVO login (String username, String password); | |
| 前置条件 | 用户名及密码符合要求。 | |
| 后置条件 | 根据用户名查找是否有相应的用户，再根据输入的密码返回登录验证的结果，若登录成功返回相应的用户值对象，否则返回null。 | |
| UserMain. logout | 语法 | public ResultMessage logout (int userID); | |
| 前置条件 | 处于已登录状态。 | |
| 后置条件 | 返回操作结果，用户退出登录状态，客户端回到主登录界面。 | |
| UserMain. isUnregistered | 语法 | public boolean isUnregistered (String username); | |
| 前置条件 | 正在注册新客户。 | |
| 后置条件 | 返回该用户名是否未被注册过。 | |
| UserMain. signUp | 语法 | public ResultMessage signUp (String username, String password); | |
| 前置条件 | 输入的用户名和密码符合规则。 | |
| 后置条件 | 返回注册客户操作的结果，若注册成功切换至新客户的主界面。 | |
| UserMain. currentUser | 语法 | public UserVO currentUser (); | |
| 前置条件 | 无。 | |
| 后置条件 | 返回当前用户相应的值对象，若当前客户端处于未登录状态则返回null。 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| DatabaseFacory. getUserDatabase () | | | 得到User数据库服务的引用 |
| UserDataService. userExist (String username) | | | 是否存在该用户名对应的用户 |
| UserDataService. checkPassword (String username, String password) | | | 验证该用户密码是否正确 |
| UserDataService. insert (UserPO user) | | | 在数据库中插入UserPO对象 |

### 4.1.4 业务逻辑层的动态模型

图3表明了酒店管理系统中，客户端主界面中，用户进行注册及登录操作时用户管理业务逻辑模块中的相关对象的协作。

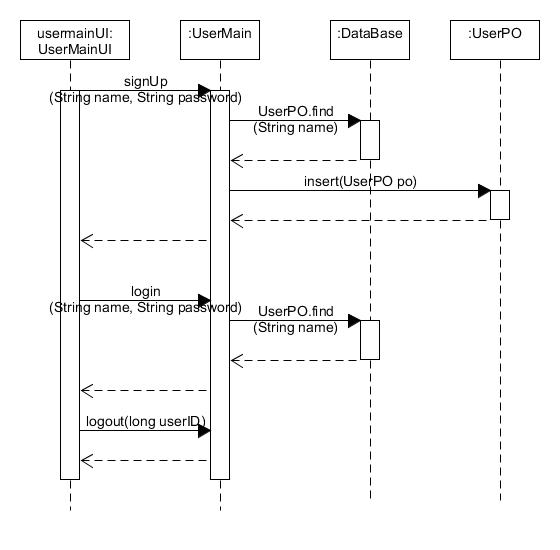


图3 注册及登录的顺序图

如图4所示的状态图表明了UserMain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。

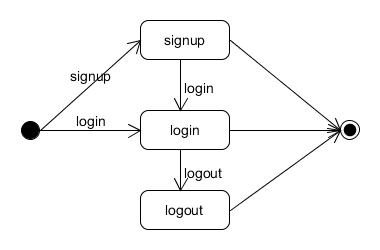


图4 UserMain对象状态图

## 4.2 personalbl模块

### 4.2.1 personalbl模块概述

personalbl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。

personalbl模块的职责及接口参见软件体系结构描述文档。

### 4.2.2 personalbl模块整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加personalblservice, personaldataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了PersonalController，这样PersonalController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给Personalbl 对象。PersonalPO是作为管理信息的持久化对象被添加到设计模型中的。

personalbl模块的设计如图5所示:

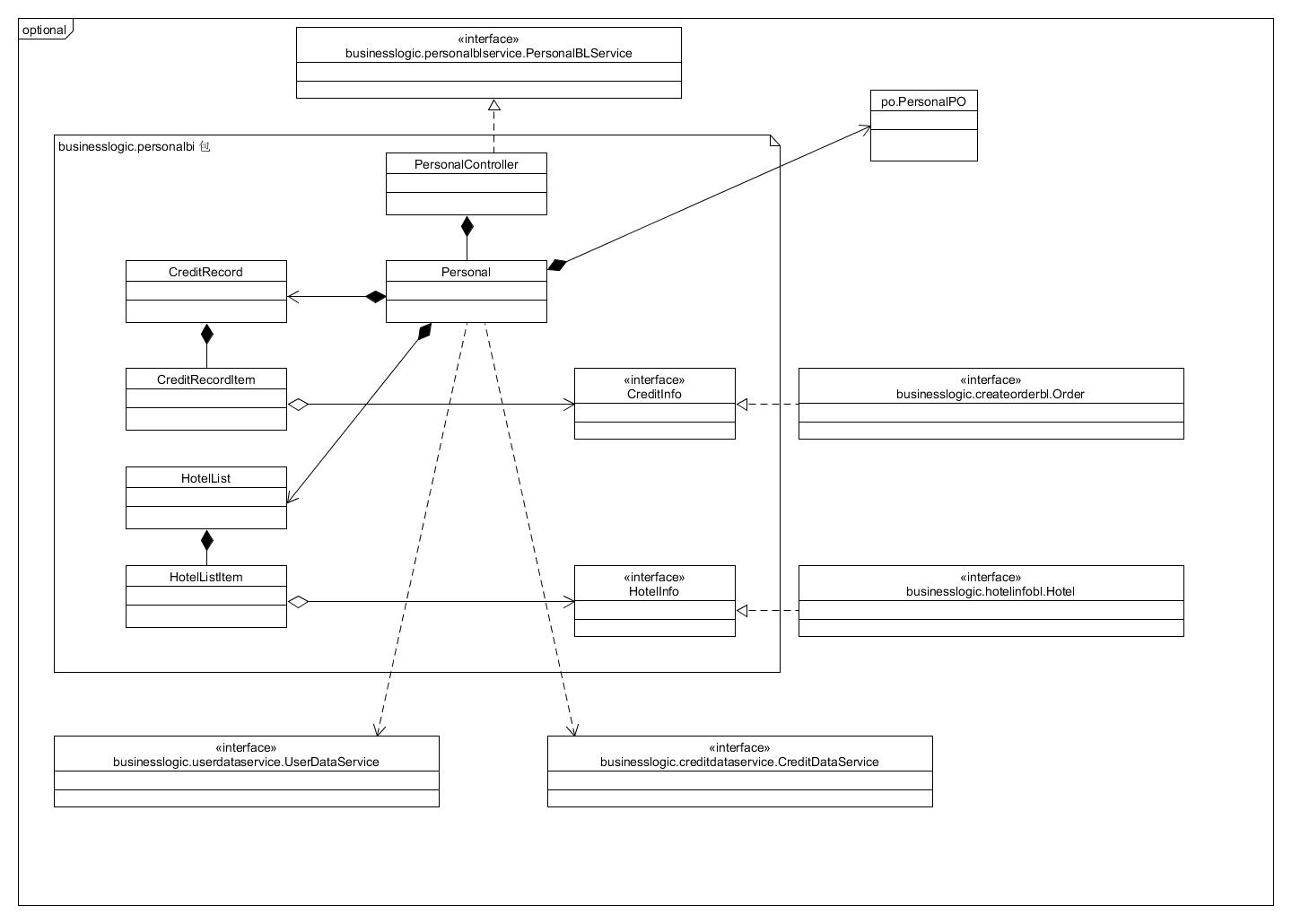


图5 personalbl模块各个类的设计

personalbl模块各个类的职责如表5所示：

表5 personalbl 模块各个类的职责

|  |  |
| --- | --- |
| PersonalController | 负责用户的注册、登陆和登出 |
| Personal | 帮助完成用户的注册、登陆和登出 |

### 4.2.3 personalbl模块内部类的接口规范

表6 PersonalController模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| PersonalController.getCreditRecord | 语法 | public List<CreditVO> getCreditRecord (long userID) | |
| 前置条件 | 已经创建Personal对象，并且输入符合规则 | |
| 后置条件 | 调用Personal领域对象的getCreditRecord | |
| PersonalController.memberSignIn | 语法 | public ResultMessage memberSignIn(PersonalVO personal) | |
| 前置条件 | 已经创建Personal领域对象 | |
| 后置条件 | 调用Personal领域对象的 getCreditRecord方法 | |
| PersonalController.manageInfo | 语法 | public ResultMessage manageInfo (PersonalVO personal) | |
| 前置条件 | 已经创建Personal领域对象 | |
| 后置条件 | 已经创建Personal 领域对象的manageInfo方法 | |
| PersonalController.getPersonal | 语法 | public PersonalVO getPersonal (long userID) | |
| 前置条件 | 已经创建Personal领域对象 | |
| 后置条件 | 调用Personal领域对象的 getPersonal方法 | |
| PersonalController.modifyPersonal | 语法 | public ResultMessage modifyPersonal (PersonalVO personal) | |
| 前置条件 | 已经创建Personal领域对象，并且输入合法 | |
| 后置条件 | 调用Personal领域对象的 modifyPersonal方法 | |
| PersonalController.getHotelList | 语法 | public List<HotelVO> getHotelList (long userID) | |
| 前置条件 | 已经创建Personal领域对象，并且输入合法 | |
| 后置条件 | 调用Personal 领域对象的getHotelList方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Personal.getCreditRecord | | | 得到当前客户的信用记录列表 |
| Personal.memberSignIn | | | 保存会员信息 |
| Personal.manageInfo | | | 保存修改后的用户信息 |
| Personal.getPersonal | | | 得到相应用户的信息 |
| Personal.modifyPersonal | | | 保存修改后的相应用户个人信息 |
| Personal.getHotelList | | | 得到当前客户预订过的酒店列表 |

表7 Personalbl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Personal.getCreditRecord | 语法 | public List<CreditVO> getCreditRecord (long userID) | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回当前客户的信用记录列表 | |
| Personal.memberSignIn | 语法 | public ResultMessage memberSignIn(PersonalVO personal) | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回操作结果，保存会员信息 | |
| Personal.manageInfo | 语法 | public ResultMessage manageInfo (PersonalVO personal) | |
| 前置条件 | 已经登录为网站管理人员 | |
| 后置条件 | 返回操作结果，若成功则系统保存修改后的用户信息 | |
| Personal.getPersonal | 语法 | public PersonalVO getPersonal (long userID) | |
| 前置条件 | 已经登录 | |
| 后置条件 | 返回相应用户的信息 | |
| Personal.modifyPersonal | 语法 | public ResultMessage modifyPersonal (PersonalVO personal) | |
| 前置条件 | 已经进入用户信息界面 | |
| 后置条件 | 返回操作结果，若成功则系统保存修改后的相应用户个人信息 | |
| Personal.getHotelList | 语法 | public List<HotelVO> getHotelList (long userID) | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回当前客户预订过的酒店列表 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserDataService.getUserInfo(long userID) | | | 根据用户编号进行查找多个持久化对象 |
| UserDataService.update(PersonalPO personal) | | | 更新单一持久化对象 |
| OrderDataService.getHotelList(long userID) | | | 根据用户编号进行查找多个持久化对象 |
| HotelInfoDataService.briefHotelInfo(long hotelID) | | | 根据酒店编号进行查找单个持久化对象 |
| CreditDataService.getRecord(long userID) | | | 根据用户编号进行查找多个持久化对象 |
| UserDataService.insert(PersonalPO personal) | | | 插入单一持久化对象 |
| DatabaseFactory.getPersonalDatabase | | | 得到Personal数据库的服务的引用 |
| DatabaseFactory.getCreditDatabase | | | 得到Credit数据库的服务的引用 |
| DatabaseFactory.getOrderDatabase | | | 得到Order数据库的服务的引用 |
| DatabaseFactory.getHotelDatabase | | | 得到Hotel数据库的服务的引用 |

### 4.2.4 业务逻辑层的动态模型

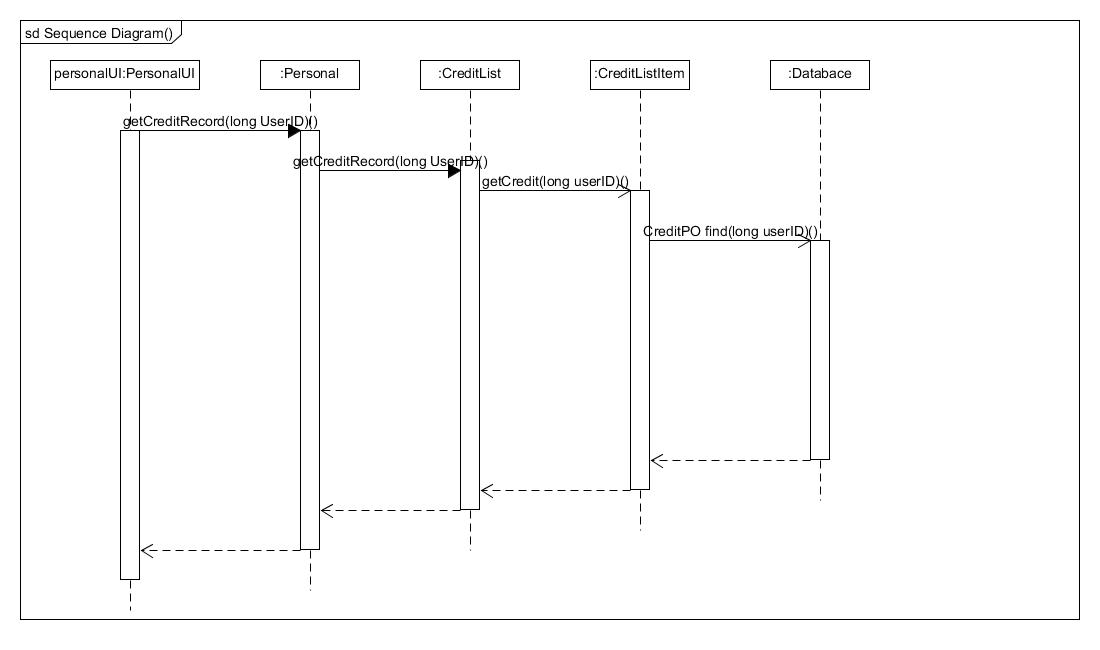


图6 查看信用记录的顺序图

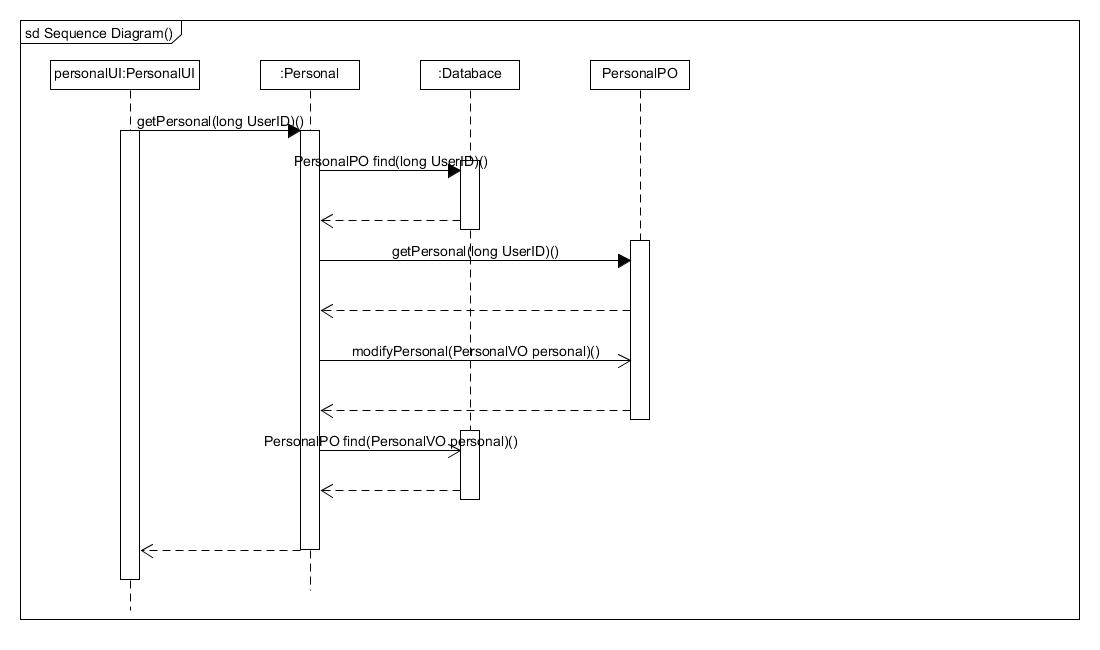


图7 查看用户信息顺序图

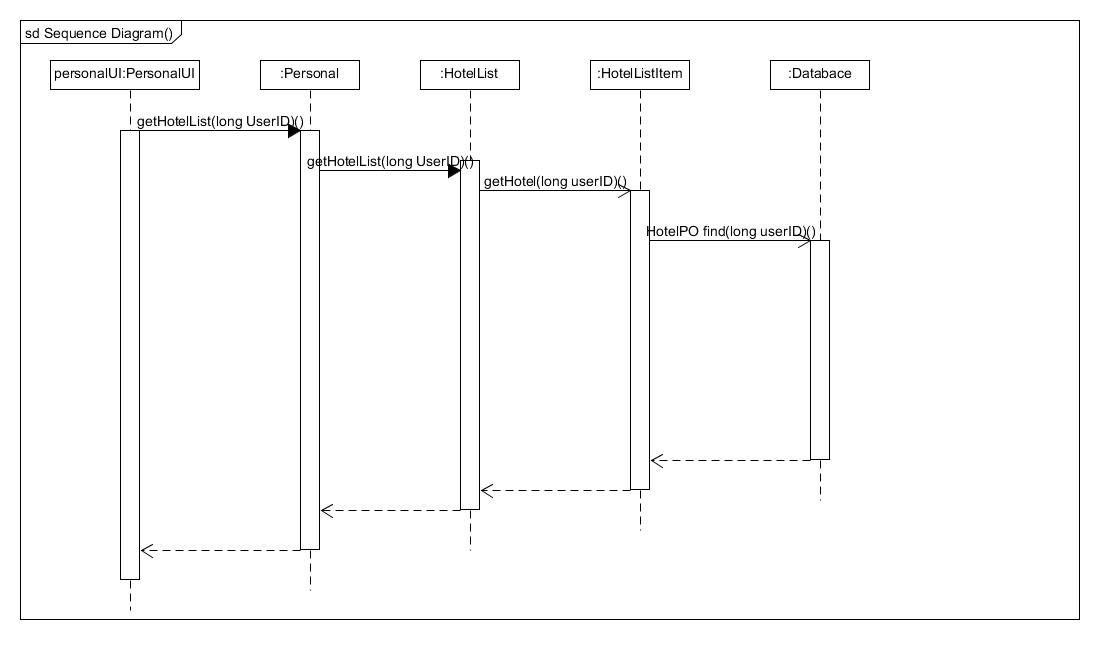


图8 查看预定过的酒店顺序图

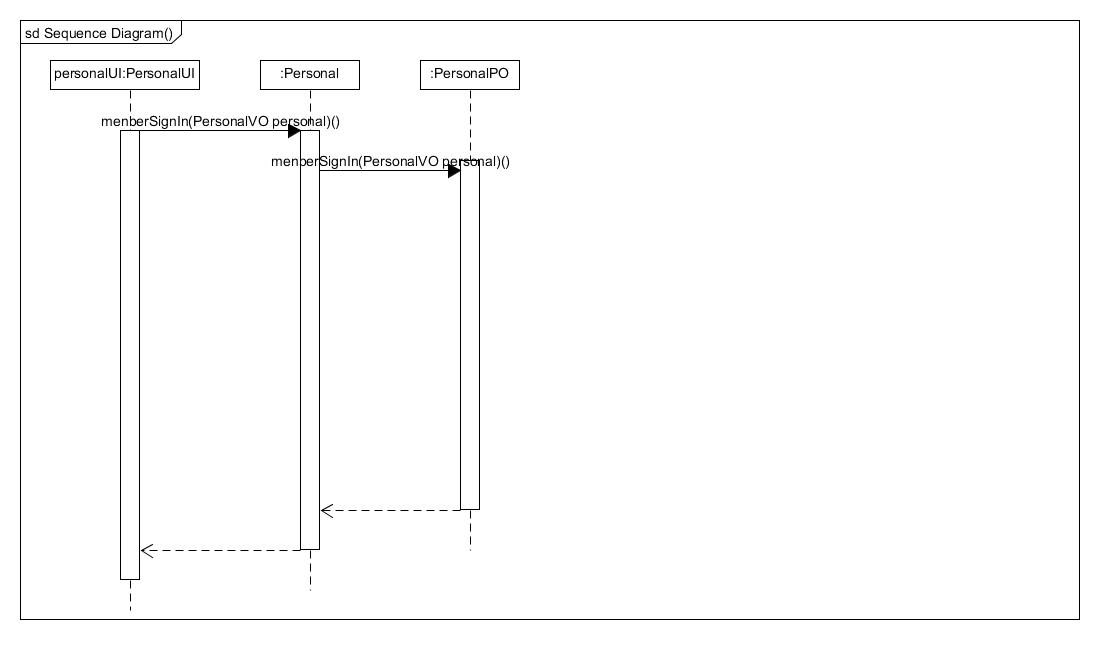


图9 注册会员顺序图

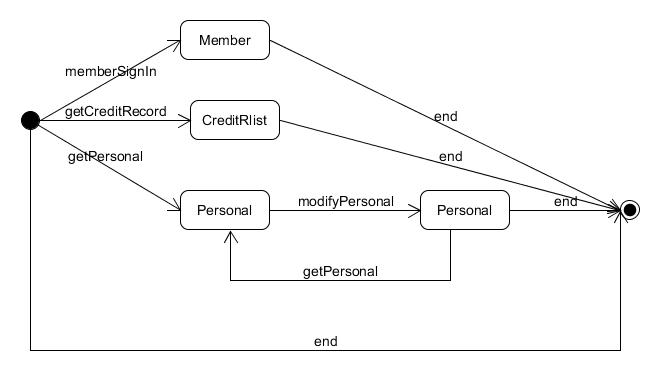


图10 Personal对象的状态图

## 4.3 createorderbl模块

### 4.3.1 createorderbl模块概述

createorderbl模块承担的需求参加需求规格说明文档功能需求及相关非功能需求。

createorderbl模块的职责及接口参加软件体系结构描述文档。

### 4.3.2 createorder模块整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了 createorderblservice, createorderdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了createorderController，这样createorderController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给createorderbl 对象。OrderPO是做为管理信息的持久化对象被添加到设计模型中的。

createorderbl模块的设计如图11所示:

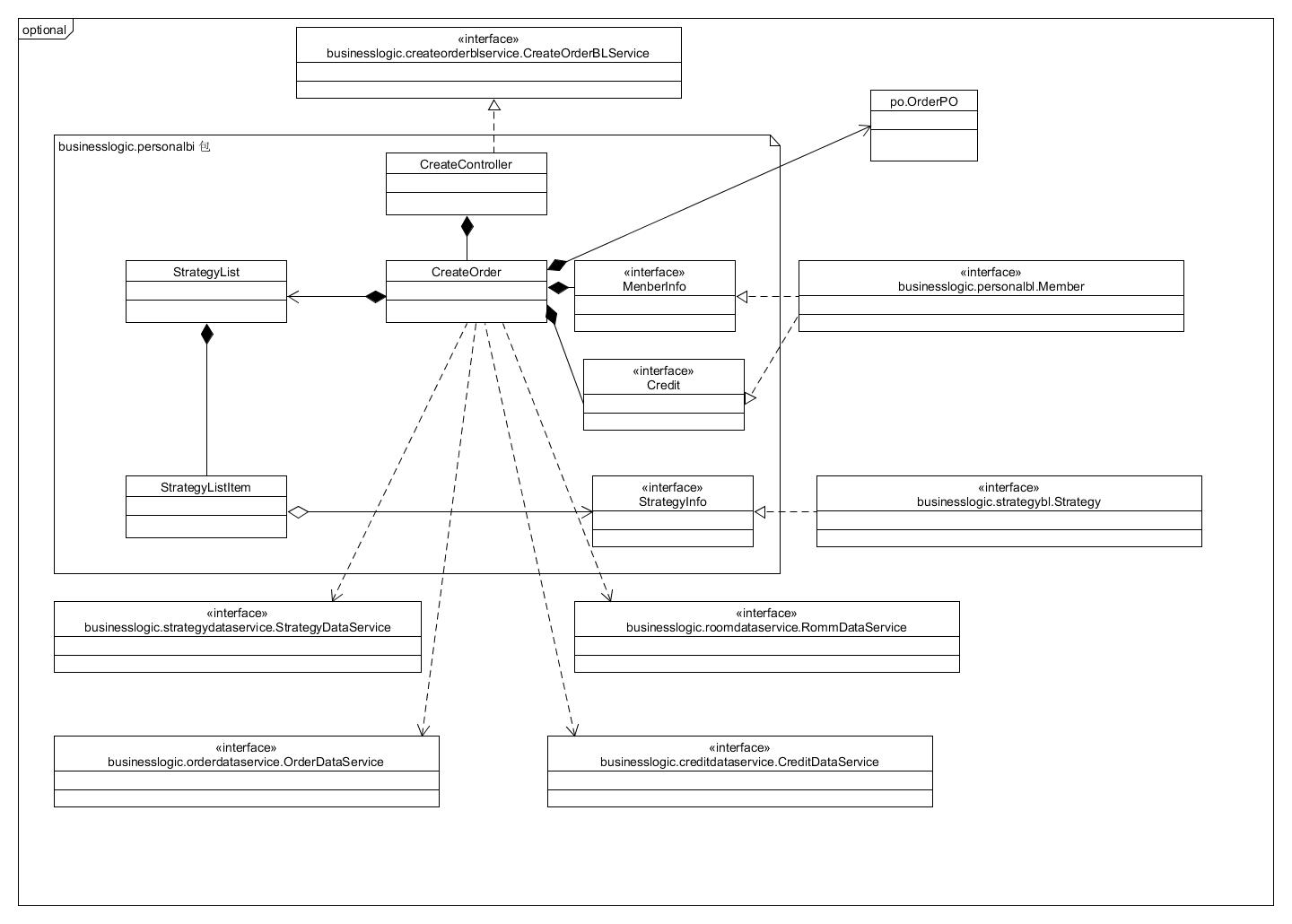


图11 createorderbl模块各个类的设计

createorderbl模块各个类的职责如表8所示：

表8 createorderbl 模块各个类的职责

|  |  |
| --- | --- |
| CreateOrderController | 负责用户的注册、登陆和登出 |
| CreateOrder | 帮助完成用户的注册、登陆和登出 |

### 4.3.2 createorderbl模块内部类的接口规范

表9 createorderController模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CreateOrderController.selectRoomType | 语法 | public ResultMessage selectRoomType (long hotelID, int roomTypeID); | |
| 前置条件 | 已经创建CreateOrder领域对象 | |
| 后置条件 | 调用CreteOrder领域对象selectRoomType方法 | |
| CreateOrderController.orderInfo | 语法 | public ResultMessage orderInfo (OrderInfoVO orderInfo); | |
| 前置条件 | 已经创建CreateOrder领域对象 | |
| 后置条件 | 调用CreateOrder 领域对象orderInfo方法 | |
| CreateOrderController.endCreateOrder | 语法 | public void endCreateOrder() | |
| 前置条件 | 已经创建CreateOrder领域对象 | |
| 后置条件 | 调用CreateOrder领域对象的 endCreateOrder方法 | |
| CreateOrderController.getOptimal | 语法 | public OrderStrategyVO getOptimal() | |
| 前置条件 | 已经创建CreateOrder领域对象 | |
| 后置条件 | 调用CreateOrder领域对象的 gerOptimal方法 | |
| CreateOrderController.otherStrategy | 语法 | public List<OrderStrategyVO> otherStrategy () | |
| 前置条件 | 已经创建CreateOrder领域对象 | |
| 后置条件 | 调用CreateOrder领域对象的otherStrate方法 | |
| CreateOrderController.selectStrategy | 语法 | public ResultMessage selectStrategy (long strategyID); | |
| 前置条件 | 已经创建CreateOrder领域对象，并且输入合法 | |
| 后置条件 | 调用CreateOrder领域对象的selectStrtegy | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderDataService.insert(OrderPO order) | | | 插入单一持久化对象 |
| HotelInfoDataService.getRoomType(long hotelID, int roomTypeID) | | | 根据酒店编号和客房类型编号进行查找单一持久化对象 |
| StrategyDataService.searchByHotel(long hotelID) | | | 根据酒店编号进行查找多个持久化对象 |
| UserDataService.getCreditBalance(long userID) | | | 根据用户编号进行查找单一持久化对象 |
| DatabaseFactory.getOrderDatabase | | | 得到Order的数据库的服务的引用 |
| DatabaseFactory.getPersonalDatabase | | | 得到Personal的数据库的服务的引用 |
| DatabaseFactory.getStrategyDatabase | | | 得到Strategy的数据库的服务的引用 |

表10 createorderbl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CreateOrder.selectRoomType | 语法 | public ResultMessage selectRoomType (long hotelID, int roomTypeID); | |
| 前置条件 | 已经登录为客户，已经选定订房起止日期 | |
| 后置条件 | 系统返回操作结果，若成功则系统保存相应的酒店编号及客房类型编号 | |
| CreateOrder.orderInfo | 语法 | public ResultMessage orderInfo (OrderInfoVO orderInfo); | |
| 前置条件 | 已经登录为客户，已经选定客房类型 | |
| 后置条件 | 系统返回操作结果，若成功则系统保存正在生成的订单的最晚订单执行时间、入住人姓名及酒店编号 | |
| CreateOrder.endCreateOrder | 语法 | public void endCreateOrder() | |
| 前置条件 | 已支付 | |
| 后置条件 | 系统处理订单信息，并改变相应客房状态 | |
| CreateOrder.getOptimal | 语法 | public OrderStrategyVO getOptimal() | |
| 前置条件 | 客户已完成订单信息填写 | |
| 后置条件 | 返回此订单的最优折扣策略信息及相应价格 | |
| CreateOrder.otherStrategy | 语法 | public List<OrderStrategyVO> otherStrategy () | |
| 前置条件 | 客户已完成订单信息填写 | |
| 后置条件 | 返回此订单可使用的策略信息及相应价格 | |
| CreateOrder.selectStrategy | 语法 | public ResultMessage selectStrategy (long strategyID); | |
| 前置条件 | 已完成订单信息填写 | |
| 后置条件 | 返回操作结果，若成功则系统保存此订单的信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderDataService.insert(OrderPO order) | | | 插入单一持久化对象 |
| HotelInfoDataService.getRoomType(long hotelID, int roomTypeID) | | | 根据酒店编号和客房类型编号进行查找单一持久化对象 |
| StrategyDataService.searchByHotel(long hotelID) | | | 根据酒店编号进行查找多个持久化对象 |
| UserDataService.getCreditBalance(long userID) | | | 根据用户编号进行查找单一持久化对象 |
| DatabaseFactory.getOrderDatabase | | | 得到Order的数据库的服务的引用 |
| DatabaseFactory.getPersonalDatabase | | | 得到Personal的数据库的服务的引用 |
| DatabaseFactory.getStrategyDatabase | | | 得到Strategy的数据库的服务的引用 |

### 4.3.4 业务逻辑层的动态模型

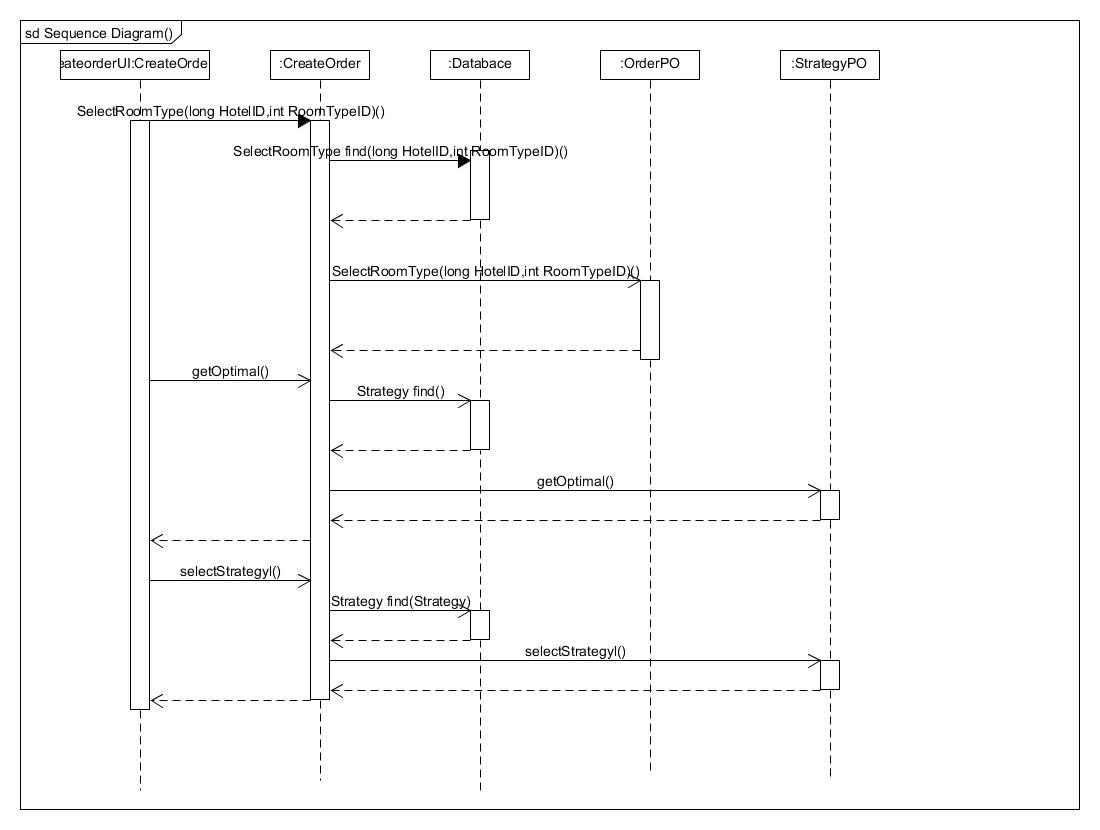


图12 生成订单的顺序图

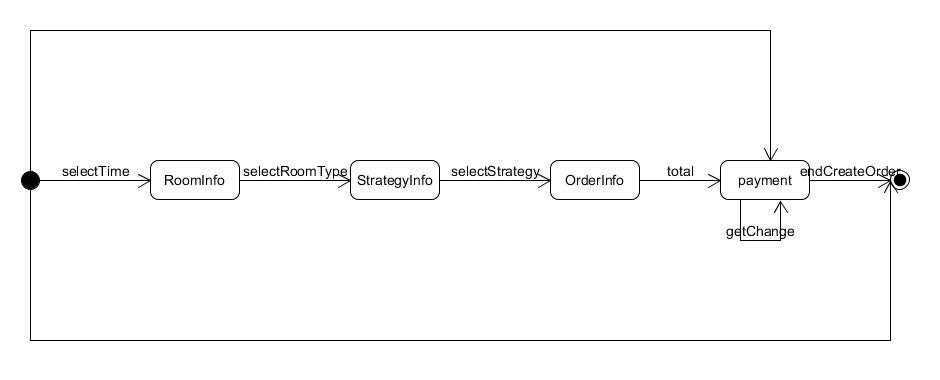


图13 CreateOrder对象的状态图

## 4.4 hotelInfobl模块

### 4.4.1 hotelInfobl模块概述

hotelInfobl模块承担的需求参加需求规格说明文档功能需求及相关非功能需求。

hotelInfobl模块的职责及接口参加软件体系结构描述文档。

### 4.4.2 hotelinfobl模块整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了 hotelInfoblservice, hotelInfodataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了HotelInfoController，这样HotelInfoController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给hotelInfobl 对象。HotelInfoPO是做为管理信息的持久化对象被添加到设计模型中的。

hotelInfolbl模块的设计如图14所示:

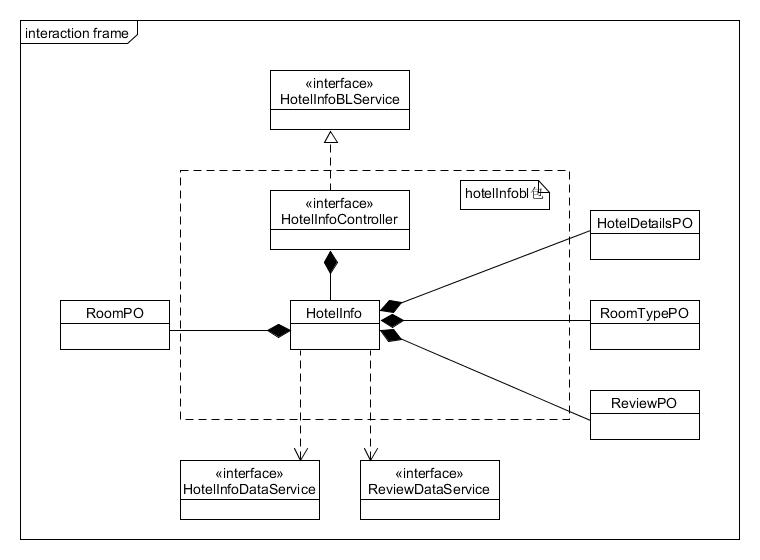


图14 manageorderbl模块各个类的设计

HotelInfobl模块各个类的职责如表11所示：

表11 HotelInfobl 模块各个类的职责

|  |  |
| --- | --- |
| HotelInfoController | 负责酒店的信息、房间的检索和修改 |
| HoteInfo | 帮助完成酒店的信息、房间的检索和修改 |

### 4.4.3 hotelInfobl模块内部类的接口规范

表12 hotelInfoController局部模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelInfoController. getHotelBrief | 语法 | public HotelBriefVO getHotelBrief (int hotelID); | |
| 前置条件 | 已经登录为客户且需要显示酒店简介。 | |
| 后置条件 | 返回该酒店的简要信息。 | |
| HotelInfoController. getHotelDetails | 语法 | public HotelDetailsVO getHotelDetails (int hotelID); | |
| 前置条件 | 已经登录为客户且需要显示酒店详情页面。 | |
| 后置条件 | 返回该酒店的详细信息。 | |
| HotelInfoController. availableRoomType | 语法 | public List<RoomTypeVO> availableRoomType (PreOrderVO preOrder); | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回满足该条件的所有可用房型 | |
| HotelInfoController. reviewHotel | 语法 | public ResultMessage reviewHotel (ReviewVO review) | |
| 前置条件 | 已经登录为客户且评论符合要求。 | |
| 后置条件 | 返回操作结果，若成功则系统保存相应的评价。 | |
| HotelInfoController. modifyHotelInfo | 语法 | public ResultMessage modifyHotelInfo (HotelInfoVO hotelInfo); | |
| 前置条件 | 已经登录为酒店管理人员，且对酒店信息做出了修改 | |
| 后置条件 | 返回操作结果，若成功则系统保存酒店信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelInfo. getHotelBrief | | | 得到酒店简要信息 |
| HotelInfo. getHotelDetails | | | 得到酒店详情 |
| HotelInfo. availableRoomType | | | 得到符合要求的可用房型 |
| HotelInfo. reviewHote | | | 评论酒店 |
| HotelInfo. modifyHotelInfo | | | 修改酒店信息 |

表13 hotelInfobl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelInfo. getHotelBrief | 语法 | public HotelBriefVO getHotelBrief (int hotelID); | |
| 前置条件 | 已经登录为客户且需要显示酒店简介。 | |
| 后置条件 | 返回该酒店的简要信息。 | |
| HotelInfo. getHotelDetails | 语法 | public HotelDetailsVO getHotelDetails (int hotelID); | |
| 前置条件 | 已经登录为客户且需要显示酒店详情页面。 | |
| 后置条件 | 返回该酒店的详细信息。 | |
| HotelInfo. availableRoomType | 语法 | public List<RoomTypeVO> availableRoomType (PreOrderVO preOrder); | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回满足该条件的所有可用房型 | |
| HotelInfo. reviewHotel | 语法 | public ResultMessage reviewHotel (ReviewVO review) | |
| 前置条件 | 已经登录为客户且评论符合要求。 | |
| 后置条件 | 返回操作结果，若成功则系统保存相应的评价。 | |
| HotelInfo. modifyHotelInfo | 语法 | public ResultMessage modifyHotelInfo (HotelInfoVO hotelInfo); | |
| 前置条件 | 已经登录为酒店管理人员，且对酒店信息做出了修改 | |
| 后置条件 | 返回操作结果，若成功则系统保存酒店信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserMain. currentUser () | | | 获取当前登录的用户 |
| ManageOrder. clientAtHotel (int hotelID) | | | 获取用户在该酒店的历史订单 |
| HotelInfoDataService. getHotelInfo (int hotelID) | | | 根据酒店编号进行查找单一持久化对象 |
| HotelInfoDataService. updateRate (int HotelID, double newRate) | | | 更新单一持久化对象 |
| HotelInfoDataService. getRoomTypes (int HotelID) | | | 根据酒店编号进行查找多个持久化对象 |
| HotelInfoDataService. update (HotelInfoPO hotelInfo) | | | 修改单一持久化对象 |
| ReviewDataService. searchByHotel (int hotelID) | | | 根据酒店编号进行查找多个持久化对象 |
| ReviewDataService. insert (ReviewPO review) | | | 插入单一持久化对象 |
| RoomDataService. availableRoomType (int hotelID, String startDate, String endDate, int numOfRoom) | | | 根据酒店编号、入住日期、离店日期及订房数量查找多个持久化对象 |
| OrderDataService. searchByClientHotel (int userID, int hotelID) | | | 根据用户编号、酒店编号查找多个持久化对象 |
| StrategyDataService. searchByHotel (long hotelID) | | | 根据酒店编号进行查找多个持久化对象 |
| DatabaseFactory. getOrderDatabase () | | | 得到Order的数据库的服务的引用 |
| DatabaseFactory. getRoomDatabase () | | | 得到Room的数据库的服务的引用 |
| DatabaseFactory. getReviewDatabase () | | | 得到Review的数据库的服务的引用 |
| DatabaseFactory. getHotelInfoDatabase () | | | 得到HotelInfo的数据库的服务的引用 |
| DatabaseFecory. getStrategyDatabase () | | | 得到Strategy的数据库的服务的引用 |

### .4.4.4 业务逻辑层的动态模型

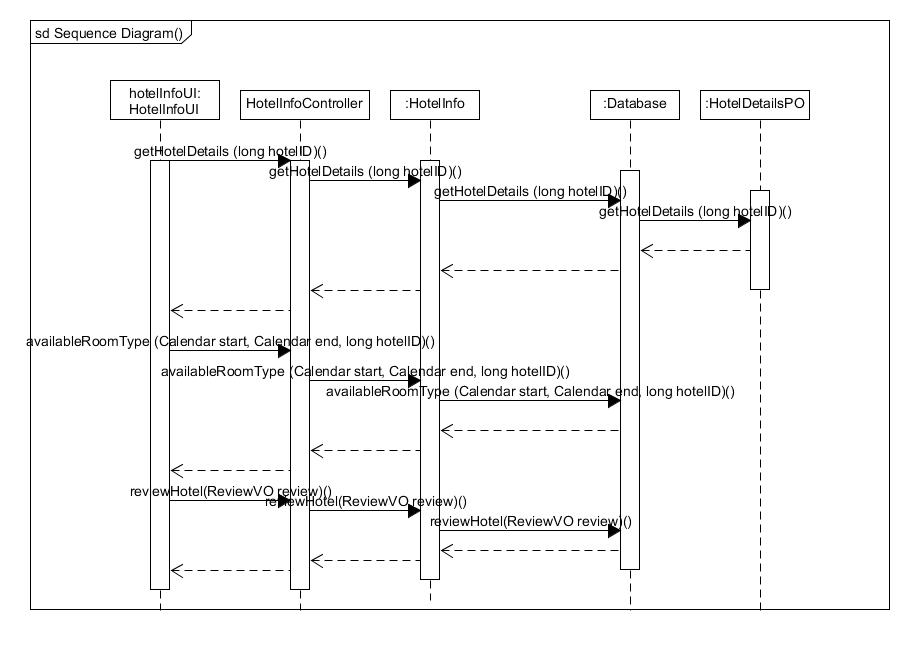


图15 查看酒店详情的顺序图

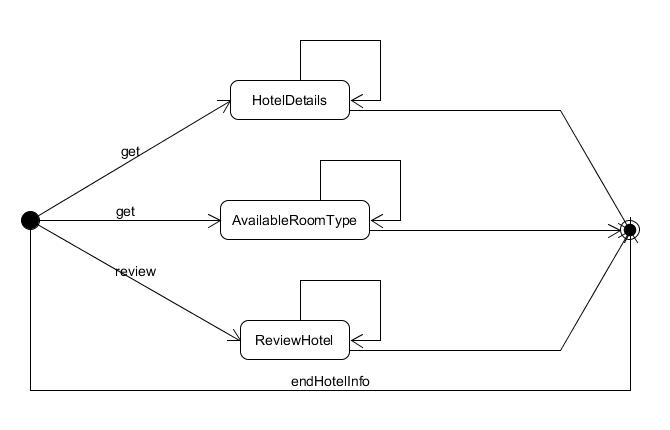


图16 HotelInfo类的状态图

## 4.5 searchhotelbl模块

### 4.5.1 searchhotelbl模块概述

searchhotelbl模块承担的需求参加需求规格说明文档功能需求及相关非功能需求。

searchhotelbl模块的职责及接口参加软件体系结构描述文档。

### 4.5.2 searchhotelbl模块整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了searchHotelblservice, searchHoteldataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了SearchHotelController，这样SearchHotelController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给searchHotelbl 对象。HotelPO是做为管理信息的持久化对象被添加到设计模型中的。

searchhotelbl模块的设计如图17所示:

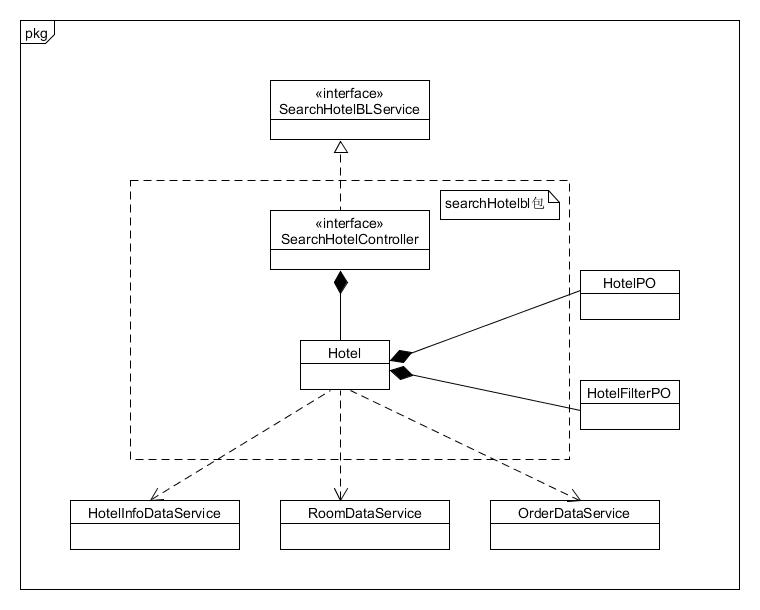


图17 searchhotelbl模块各个类的设计

searchhotelbl模块各个类的职责如表14所示：

表14 searchhotelbl 模块各个类的职责

|  |  |
| --- | --- |
| SearchHotelController | 负责酒店的搜索 |
| SearchHotel | 帮助完成酒店的搜索 |

### 4.5.3 searchhotelbl模块内部类的接口规范

表15 searchhotelbl局部模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| SearchHotelController. renewDistrict | 语法 | public List<DistrictVO> renewDistrict (); | |
| 前置条件 | 客户端刚刚启动且有网络连接。 | |
| 后置条件 | 返回商圈信息列表。 | |
| SearchHotelController. getHotelList | 语法 | public List<HotelBriefVO> getHotelList (FilterHotelVO filter, SortHotelVO sort, int rank1, int rank2); | |
| 前置条件 | 已经登录为客户，要显示查找到的酒店列表。 | |
| 后置条件 | 返回根据筛选条件、排序条件及起止序号生成的酒店简介列表。 | |
| SearchHotelController. recommend | 语法 | public List<HotelVO> recommend (); | |
| 前置条件 | 已经登录为客户且显示商圈的酒店列表 | |
| 后置条件 | 根据筛选条件查找符合要求的酒店列表并返回 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| SearchHotel. renewDistrict | | | 得到商圈信息列表 |
| SearchHotel. getHotelList | | | 得到符合的酒店列表 |
| SearchHotel. recommend | | | 得到推荐的酒店 |

表16 searchhotelbl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| SearchHotel. renewDistrict | 语法 | public List<DistrictVO> renewDistrict (); | |
| 前置条件 | 客户端刚刚启动且有网络连接。 | |
| 后置条件 | 返回商圈信息列表。 | |
| SearchHotel. getHotelList | 语法 | public List<HotelBriefVO> getHotelList (FilterHotelVO filter, SortHotelVO sort, int rank1, int rank2); | |
| 前置条件 | 已经登录为客户，要显示查找到的酒店列表。 | |
| 后置条件 | 返回根据筛选条件、排序条件及起止序号生成的酒店简介列表。 | |
| SearchHotel. recommend | 语法 | public List<HotelVO> recommend (); | |
| 前置条件 | 已经登录为客户且显示商圈的酒店列表 | |
| 后置条件 | 根据筛选条件查找符合要求的酒店列表并返回 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserMain. currentUser () | | | 获取当前登录的用户 |
| HotelInfo. availableRoomType (PreOrderVO preOrder) | | | 用酒店编号、入住日期、离店日期及订房数量查找客房类型 |
| HotelInfoDataService. getHotelList (int districtID); | | | 根据商圈编号进行查找多个持久化对象 |
| OrderDataService.hasBeenOrdered(long hotelID, long userID) | | | 根据酒店编号及订房客户的用户名进行查找多个持久化对象 |
| LocationDataService. getDistricts () | | | 查找多个持久化对象 |
| DatabaseFactory. getHotelInfo | | | 得到对HotelInfo数据库的服务的引用 |
| DatabaseFactory. getRoom | | | 得到对Room数据库的服务的引用 |
| DatabaseFactory. getOrder | | | 得到对Order数据库的服务的引用 |
| DatabaseFactory. getLocation | | | 得到对Location数据库的服务的引用 |

### 4.5.6 searchhotelbl模块的行为

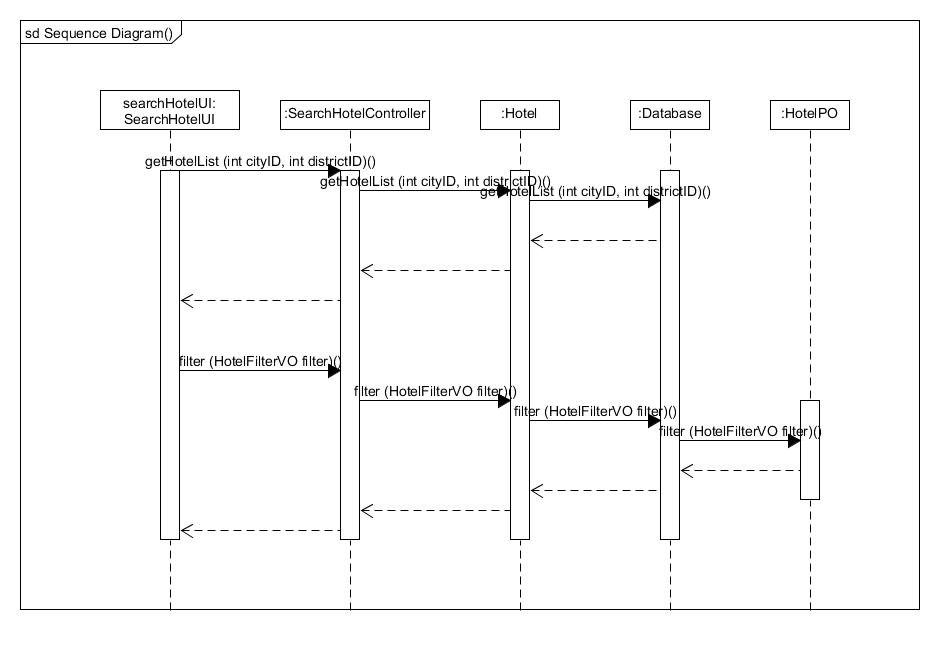


图18 搜索酒店的顺序图

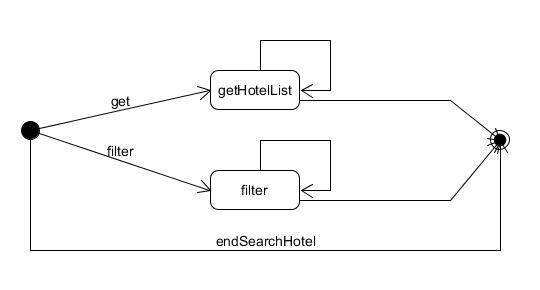


图19 SearchHotel类的状态图

## 4.6 manageorderbl模块

### 4.6.1 manageorderbl模块概述

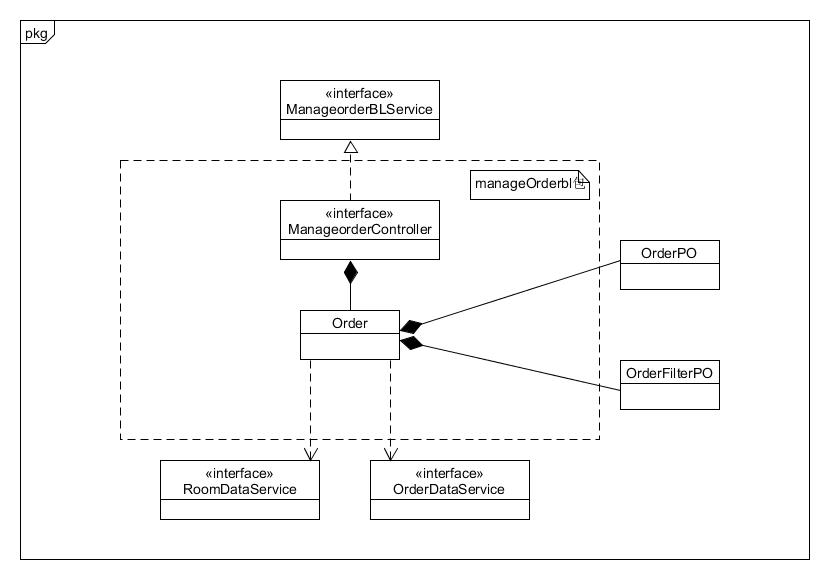
ManageOrderbl模块承担的需求参加需求规格说明文档功能需求及相关非功能需求。

ManageOrderbl模块的职责及接口参加软件体系结构描述文档。

### 4.6.2 manageorderbl模块整体结构

根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。每一层之间为了灵活性，添加了接口，以实现针对接口编程，隔离数据传输的职责，降低层与层之间耦合，添加了manageOrderservice, manageOrderdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了ManageOrderController，这样ManageOrderController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给manageOrderbl 对象。OrderPO是做为管理信息的持久化对象被添加到设计模型中的。

managehotelbl模块的设计如图2.9.1-1所示.



### 4.6.3 managorderbl模块内部类的接口规范

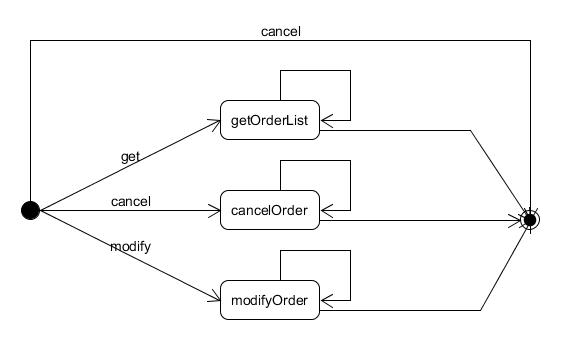
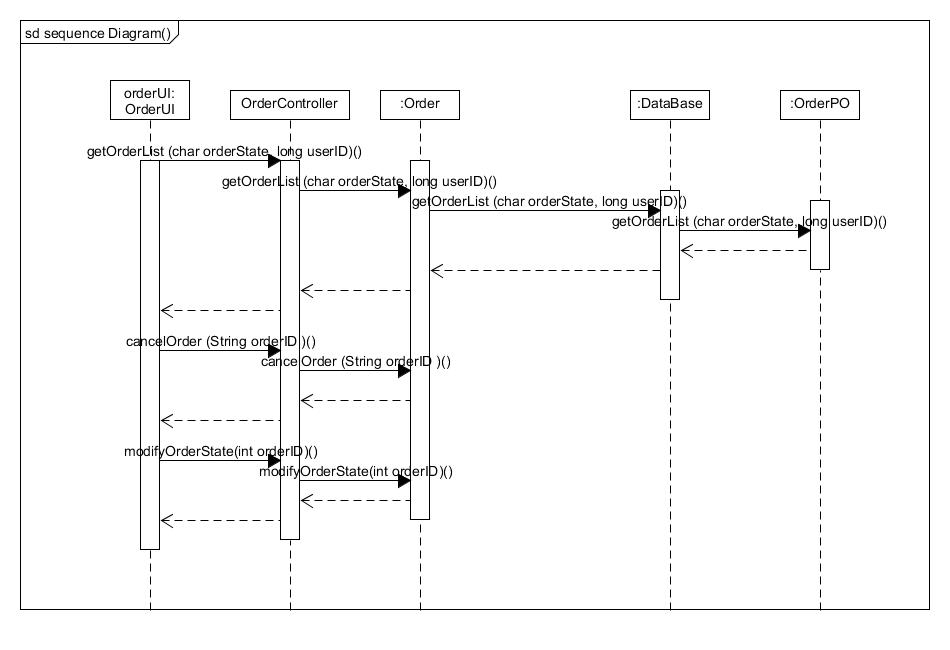
manageorderbl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ManageOrderController. clientOrderList | 语法 | public List<OrderVO> clientOrderList (char orderState, long userID); | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回客户该状态的订单列表 | |
| ManageOrderController. getOrderList | 语法 | public List<OrderVO> hotelOrderList (long hotelID, OrderFilterVO orderFilter, int page); | |
| 前置条件 | 已经登录为酒店工作人员或网站营销人员 | |
| 后置条件 | 返回全网站或酒店符合筛选条件的相应页码的订单列表 | |
| ManageOrderController. clientCancelOrder | 语法 | public ResultMessage clientCancelOrder (String orderID); | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回操作结果，若成功则系统撤销相应的订单 | |
| ManageOrderController. marketerCancelOrder | 语法 | public ResultMessage marketerCancelOrder (String orderID, CreditRestore restore); | |
| 前置条件 | 已经登录为网站营销人员 | |
| 后置条件 | 返回操作结果，若成功则系统撤销订单并相应地恢复信用值 | |
| ManageOrderController. getClientOrderList | 语法 | public List<OrderVO> getClientOrderList (long userID, long hotelID); | |
| 前置条件 | 已经登录为网站营销人员 | |
| 后置条件 | 返回该用户在该酒店的所有订单列表 | |
| ManageOrderController. checkIn | 语法 | public ResultMessage checkIn(String roomNum, Time leaveTime, String orderID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统修改相应订单及客房的状态 | |
| ManageOrderController. checkOut | 语法 | public ResultMessage checkOut(int orderID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统修改相应订单及客房的状态 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| ManageOrder. clientOrderList | | | 客户获得自己的订单列表 |
| ManageOrder. getOrderList | | | 酒店工作人员获得订单列表 |
| ManageOrder. clientCancelOrder | | | 客户撤销订单 |
| ManageOrder. marketCancelOrder | | | 网站营销人员撤销订单 |
| ManageOrder. getClientOrderList | | | 网站营销人员获得客户订单列表 |
| ManageOrder. checkIn | | | 酒店工作人员办理入住 |
| ManageOrder. checkOut | | | 酒店工作人员办理退房 |

manageorderbl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ManageOrder. clientOrderList | 语法 | public List<OrderVO> clientOrderList (char orderState, long userID); | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回客户该状态的订单列表 | |
| ManageOrder. getOrderList | 语法 | public List<OrderVO> hotelOrderList (long hotelID, OrderFilterVO orderFilter, int page); | |
| 前置条件 | 已经登录为酒店工作人员或网站营销人员 | |
| 后置条件 | 返回全网站或酒店符合筛选条件的相应页码的订单列表 | |
| ManageOrder. clientCancelOrder | 语法 | public ResultMessage clientCancelOrder (String orderID); | |
| 前置条件 | 已经登录为客户 | |
| 后置条件 | 返回操作结果，若成功则系统撤销相应的订单 | |
| ManageOrder. marketerCancelOrder | 语法 | public ResultMessage marketerCancelOrder (String orderID, CreditRestore restore); | |
| 前置条件 | 已经登录为网站营销人员 | |
| 后置条件 | 返回操作结果，若成功则系统撤销订单并相应地恢复信用值 | |
| ManageOrder. clientOrderList | 语法 | public List<OrderVO> clientOrderList (long userID, long hotelID); | |
| 前置条件 | 已经登录为网站营销人员 | |
| 后置条件 | 返回该用户在该酒店的所有订单列表 | |
| ManageOrder. checkIn | 语法 | public ResultMessage checkIn(String roomNum, Time leaveTime, String orderID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统修改相应订单及客房的状态 | |
| ManageOrder. checkOut | 语法 | public ResultMessage checkOut(int orderID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统修改相应订单及客房的状态 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderDataService. update(OrderPO order) | | | 更新单一持久化对象 |
| OrderDataService. getOrderPage(long hotelID, OrderFilterVO orderFilter, int page) | | | 根据筛选条件进行查找多个持久化对象 |
| RoomDataService. checkIn (int roomID) | | | 更新单一持久化对象 |
| RoomDataService. checkOut (int roomID) | | | 更新单一持久化对象 |
| DatabaseFactory. getOrder | | | 获得对Order数据库的服务的引用 |
| DatabaseFactory. getRoom | | | 获得对Room数据库的服务的引用 |
| DatabaseFactory. getCredit | | | 获得对Credit数据库服务的引用 |

### 4.6.4 业务逻辑层的动态模型

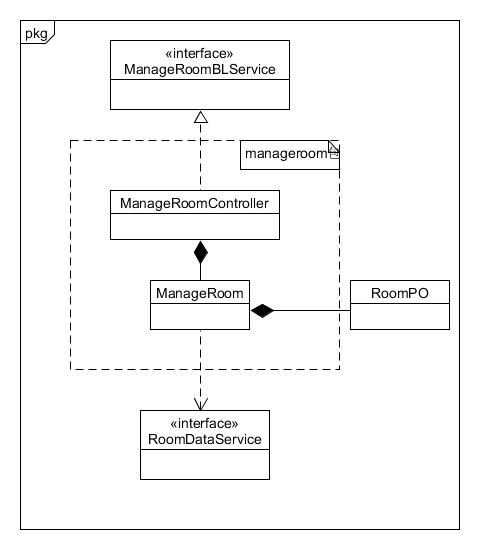


## 4.7 manageroombl模块

### 4.7.1 manageroombl模块概述

ManageRoombl模块承担的需求参加需求规格说明文档功能需求及相关非功能需求。ManageRoombl模块的职责及接口参加软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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manageroomblservice, roomdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了ManageRoomController，这样ManageRoomController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给manageroombl 对象。RoomPO是做为管理信息的持久化对象被添加到设计模型中的。

manageroombl模块的设计如图2.7.1-1所示.



### 4.7.2 manageroombl模块局部模块的接口规范

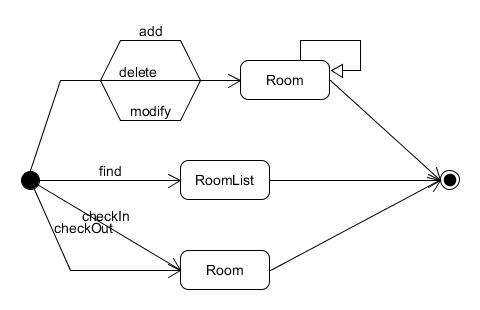
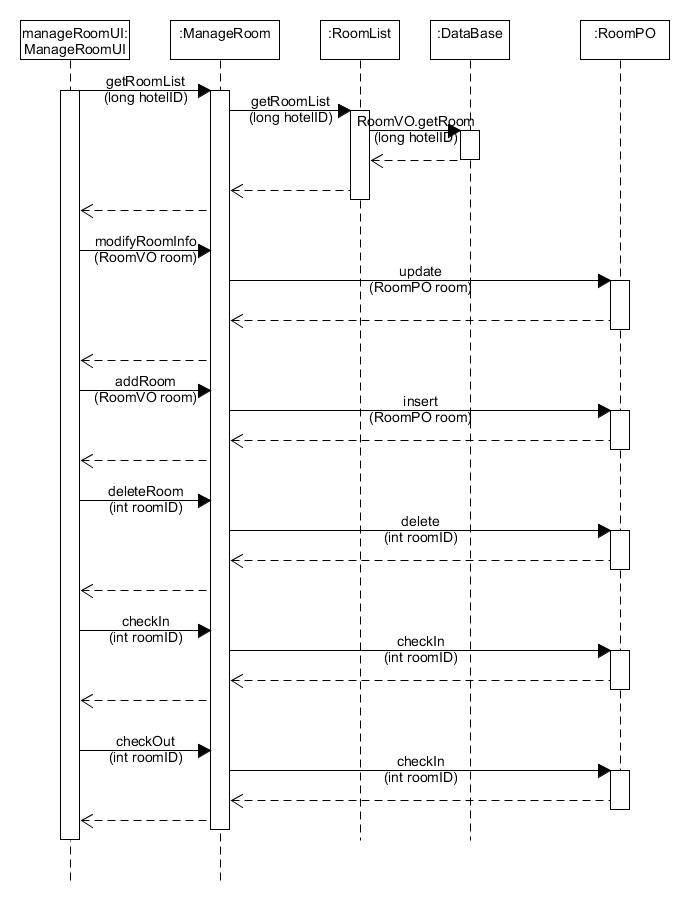
ManageRoomController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ManageRoomController. getRoomList | 语法 | public List<RoomVO> getRoomList (long hotelID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回相应酒店所有可用客房的列表 | |
| ManageRoomController. modifyRoomInfo | 语法 | public ResultMessage modifyRoomInfo (RoomVO room); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统保存修改后的相应房间信息 | |
| ManageRoomController. addRoom | 语法 | public ResultMessage addRoom (RoomVO room); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统添加相应的房间信息 | |
| ManageRoomController. deleteRoom | 语法 | public ResultMessage deleteRoom (int roomID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统删除相应的房间信息 | |
| ManageRoomController. checkIn | 语法 | public ResultMessage checkIn (int roomID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统将相应的客房设置为已入住状态 | |
| ManageRoomController. checkOut | 语法 | public ResultMessage checkOut (int roomID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统将相应的客房设置为空闲状态 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| ManageRoom. getRoomList | | | 查看酒店所有客房 |
| ManageRoom. modifyRoomInfo | | | 更新客房信息 |
| ManageRoom. addRoom | | | 添加可用客房 |
| ManageRoom. deleteRoom | | | 删除可用客房 |
| ManageRoom. checkIn | | | 修改客房状态为已入住 |
| ManageRoom. checkOut | | | 修改客房状态为空闲 |

ManageRoomBL的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ManageRoom. getRoomList | 语法 | public List<RoomVO> getRoomList (long hotelID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回相应酒店所有可用客房的列表 | |
| ManageRoom. modifyRoomInfo | 语法 | public ResultMessage modifyRoomInfo (RoomVO room); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统保存修改后的相应房间信息 | |
| ManageRoom. addRoom | 语法 | public ResultMessage addRoom (RoomVO room); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统添加相应的房间信息 | |
| ManageRoom. deleteRoom | 语法 | public ResultMessage deleteRoom (int roomID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统删除相应的房间信息 | |
| ManageRoom. checkIn | 语法 | public ResultMessage checkIn (int roomID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统将相应的客房设置为已入住状态 | |
| ManageRoom. checkOut | 语法 | public ResultMessage checkOut (int roomID); | |
| 前置条件 | 已经登录为酒店工作人员 | |
| 后置条件 | 返回操作结果，若成功则系统将相应的客房设置为空闲状态 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| DatabaseFactory. getRoom | | | 获得对Room数据库的服务的引用 |
| RoomDataService. update (RoomPO room) | | | 更新单一持久化对象 |
| RoomDataService. insert (RoomPO room) | | | 插入单一持久化对象 |
| RoomDataService. delete (int roomID) | | | 删除单一持久化对象 |
| RoomDataService. checkIn (int roomID) | | | 更新单一持久化对象 |
| RoomDataService. checkOut (int roomID) | | | 更新单一持久化对象 |

### 4.7.3 manageroombl模块的行为

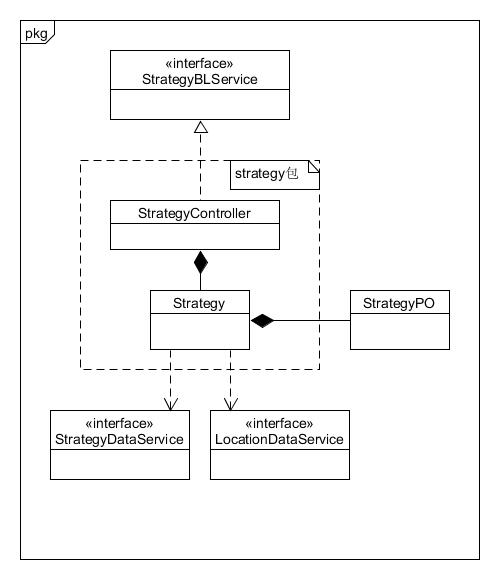


## 4.8 strategybl模块

### 4.8.1 strategybl模块概述

Strategybl模块承担的需求参加需求规格说明文档功能需求及相关非功能需求。Strategybl模块的职责及接口参加软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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strategyblservice, strategydataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了StrategyController，这样StrategyController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给strategybl 对象。StrategyPO是做为管理信息的持久化对象被添加到设计模型中的。

strategybl模块的设计如图2.8.1-1所示.



### 4.8.2 strategybl模块局部模块的接口规范

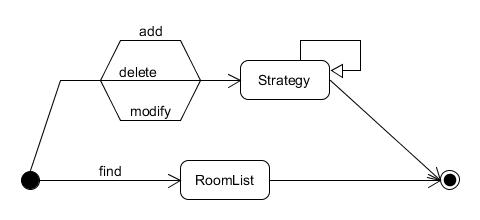
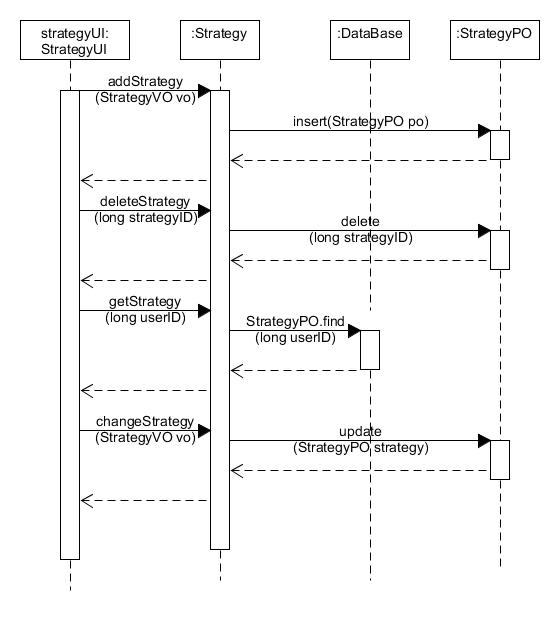
StrategyController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| StrategyController.addStrategy | 语法 | public ResultMessage addStrategy(StrategyVO); | |
| 前置条件 | 网站人员已经点击一类网站营销策略 | |
| 后置条件 | 返回操作结果，若成功则系统添加相应的营销策略 | |
| StrategyController.deleteStrategy | 语法 | public ResultMessage deleteStrategy (long strategyID) | |
| 前置条件 | 网站营销人员或者酒店管理人员已经登录 | |
| 后置条件 | 返回操作结果，若成功则系统删除编号对应的营销策略 | |
| StrategyController.getStrategy | 语法 | public List<StrategyVO> getStrategy(long userID) | |
| 前置条件 | 已登录为网站营销人员或酒店工作人员 | |
| 后置条件 | 根据用户编号返回相应营销策略列表 | |
| StrategyController.modifyStrategy | 语法 | public ResultMessage changeStrategy(StrategyVO); | |
| 前置条件 | 酒店工作人员或网站营销人员对当前已有的营销策略进行了修改 | |
| 后置条件 | 返回操作结果，若成功则系统保存修改后的相应营销策略 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Strategy.addStrategy | | | 添加促销策略 |
| Strategy.deleteStrategy | | | 删除促销策略 |
| Strategy.getStrategy | | | 得到用户适用的促销策略 |
| Strategy.modifyStrategy | | | 修改促销策略 |

StrategyBL的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Strategy.addStrategy | 语法 | public ResultMessage addStrategy(StrategyVO); | |
| 前置条件 | 网站人员已经点击一类网站营销策略 | |
| 后置条件 | 返回操作结果，若成功则系统添加相应的营销策略 | |
| Strategy.deleteStrategy | 语法 | public ResultMessage deleteStrategy (long strategyID) | |
| 前置条件 | 网站营销人员或者酒店管理人员已经登录 | |
| 后置条件 | 返回操作结果，若成功则系统删除编号对应的营销策略 | |
| Strategy.getStrategy | 语法 | public List<StrategyVO> getStrategy(long userID) | |
| 前置条件 | 已登录为网站营销人员或酒店工作人员 | |
| 后置条件 | 根据用户编号返回相应营销策略列表 | |
| Strategy.modifyStrategy | 语法 | public ResultMessage changeStrategy(StrategyVO); | |
| 前置条件 | 酒店工作人员或网站营销人员对当前已有的营销策略进行了修改 | |
| 后置条件 | 返回操作结果，若成功则系统保存修改后的相应营销策略 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| StrategyDataService.insert(StrategyPO strategy) | | | 插入单一持久化对象 |
| StrategyDataService.delete(long strategyID) | | | 删除单一持久化对象 |
| StrategyDataService.update(StrategyPO strategy) | | | 更新单一持久化对象 |
| DatabaseFactory.getStrategyDatabase | | | 得到strategy的数据库的服务的引用 |

### 4.8.3 strategybl模块的行为

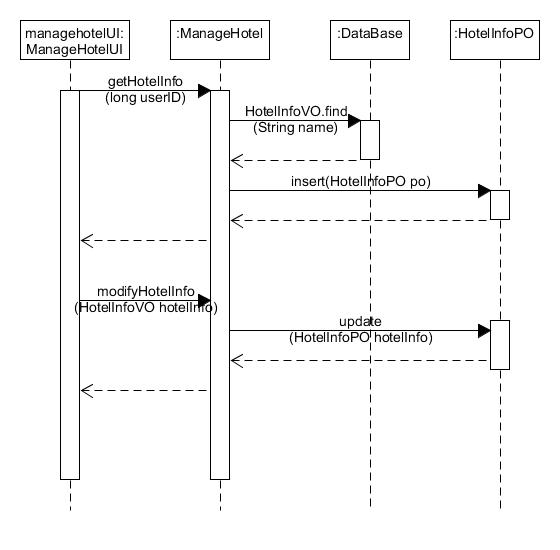


## 4.9 managehotelbl模块

### 4.9.1 managehotelbl模块概述

ManageHotelbl模块承担的需求参加需求规格说明文档功能需求及相关非功能需求。ManageHotelbl模块的职责及接口参加软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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managehotelblservice,hoteldataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了ManageHotelController，这样ManageHotelController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给managehotelbl 对象。HoteInfoPO是做为管理信息的持久化对象被添加到设计模型中的。

managehotelbl模块的设计如图2.9.1-1所示.



### 4.9.2 managehotelbl模块内部类的接口规范

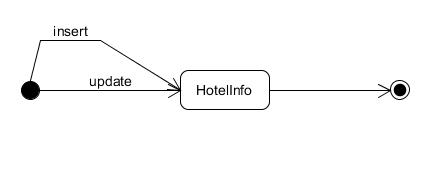
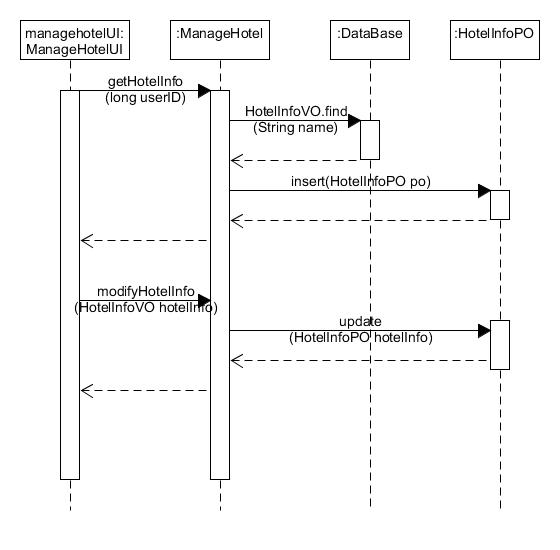
ManageHotelController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ManagehotelController. getHotelInfo | 语法 | public HotelInfoVO getHotelInfo(long userID); | |
| 前置条件 | 已经登录为酒店管理人员 | |
| 后置条件 | 返回酒店信息 | |
| ManagehotelController. modifyHotelInfo | 语法 | public ResultMessage modifyHotelInfo(HotelInfoVO hotelInfo); | |
| 前置条件 | 已经登录为酒店管理人员，且对酒店信息做出了修改 | |
| 后置条件 | 返回操作结果，若成功则系统保存酒店信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Managehotel. getHotelInfo | | | 得到酒店信息 |
| Managehotel. modifyHotelInfo | | | 修改酒店信息 |

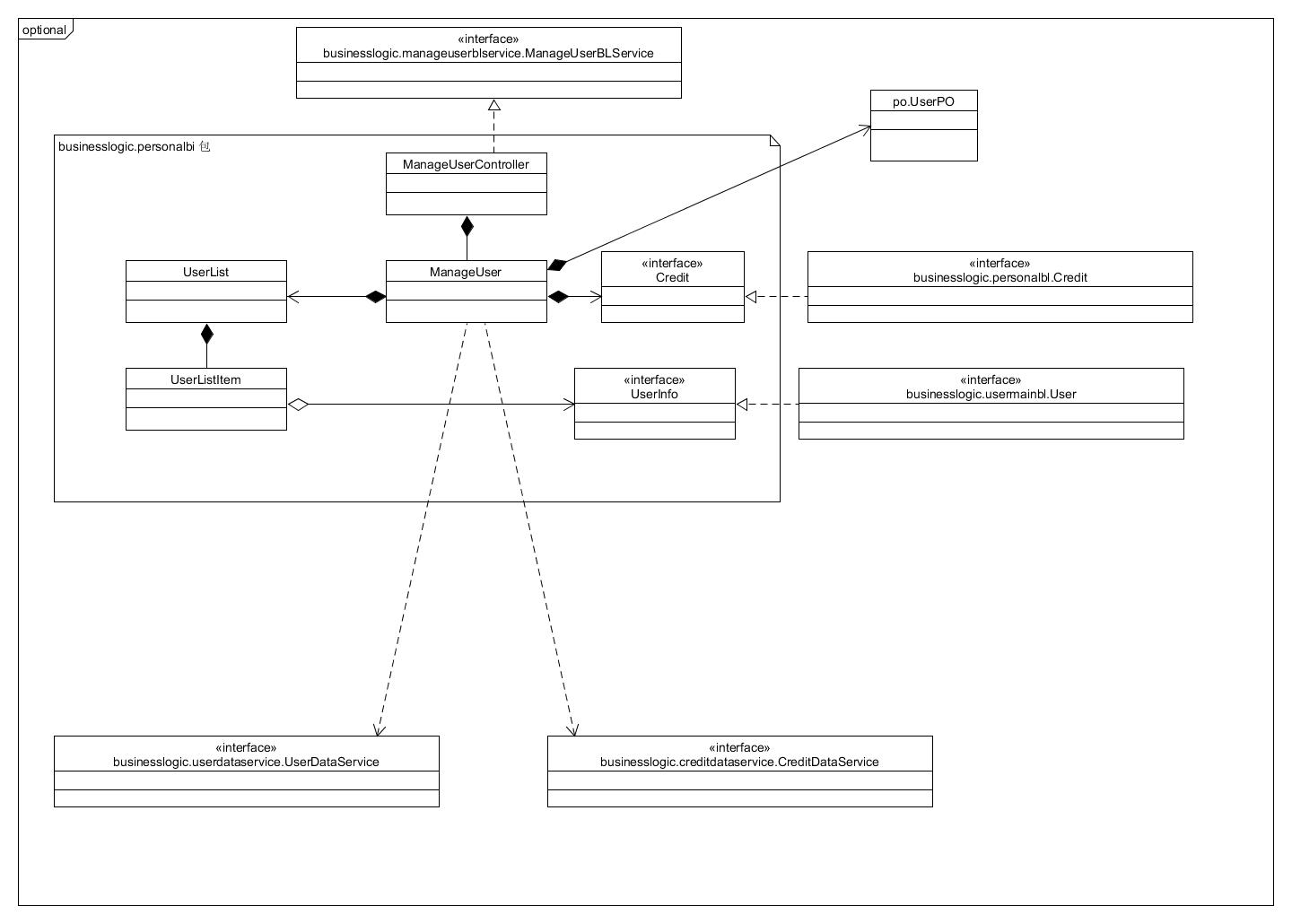
ManageHotelBL的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Managehotel. getHotelInfo | 语法 | public HotelInfoVO getHotelInfo(long userID); | |
| 前置条件 | 已经登录为酒店管理人员 | |
| 后置条件 | 返回酒店信息 | |
| Managehotel. modifyHotelInfo | 语法 | public ResultMessage modifyHotelInfo(HotelInfoVO hotelInfo); | |
| 前置条件 | 已经登录为酒店管理人员，且对酒店信息做出了修改 | |
| 后置条件 | 返回操作结果，若成功则系统保存酒店信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| DatabaseFactory. getHotelInfo | | | 获得对HotelInfo数据库的服务的引用 |
| DatabaseFactory. getUser | | | 获得对User数据库的服务的引用 |
| HotelInfoDataService.update(HotelInfoPO hotelInfo) | | | 修改单一持久化对象 |
| HotelInfoDataService.searchHotelDetails(long hotelID) | | | 根据酒店编号进行查找单一持久化对象 |

### 4.9.3 managehotelbl模块的行为



## 4.10 manageuserbl模块



### 4.10.1 manageuserbl模块概述

Manageuserbl模块承担的需求参加需求规格说明文档功能需求及相关非功能需求。Manageuserbl模块的职责及接口参加软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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manageuserblservice,userdataservice 两个接口。为了隔离业务逻辑职责和逻辑控制职责，我们添加了ManageuserController，这样ManageuserController 将会将用户管理相关的业务逻辑职责和逻辑控制委托给manageuserbl 对象。UserPO是做为管理信息的持久化对象被添加到设计模型中的。

manageuserbl模块的设计如图2.9.1-1所示.

### 4.10.2 manageuserbl模块内部类的接口规范

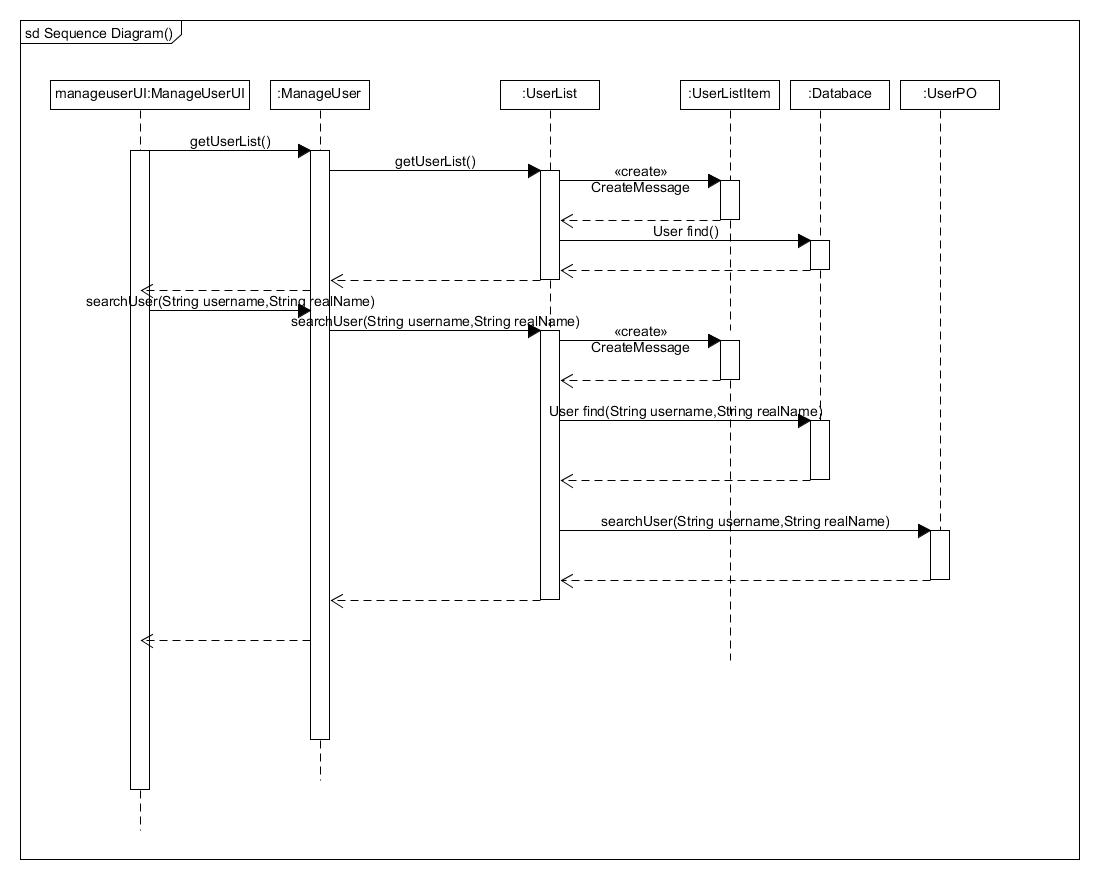
表18 manageuserControllerbl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Manageuser. SearchUserController | 语法 | public List<UserPO> searchUser(String username, String realName); | |
| 前置条件 | 已经创建Manageuser领域对象 | |
| 后置条件 | 调用Manageuser 领域对象SearchUser方法 | |
| Manageuser. searchHotelStaffController | 语法 | public List<HotelStaffPO> searchHotelStaff(int cityID, int districtID); | |
| 前置条件 | 已经创建Manageuser领域对象 | |
| 后置条件 | 调用Manageuser领域对象的 searchHotelStaffController方法 | |
| Manageuser. getUserController | 语法 | public UserPO getUser(long userID); | |
| 前置条件 | 已经创建Manageuser领域对象 | |
| 后置条件 | 调用Manageuser领域对象的 getUser方法 | |
| Manageuser. getUserListController | 语法 | public ArrayList<UserPO> getUserList(); | |
| 前置条件 | 已经创建Manageuser领域对象 | |
| 后置条件 | 调用Manageuser 领域对象getUserList方法 | |
| Manageuser. sortUserListVController | 语法 | public ArrayList<UserPO> sortUserList(String way); | |
| 前置条件 | 已经创建Manageuser领域对象 | |
| 后置条件 | 调用Manageuser 领域对象的sortUserList方法 | |
| Manageuser. creditController | 语法 | public ResultMessage credit(double money, long userID); | |
| 前置条件 | 已经创建Manageuser领域对象 | |
| 后置条件 | 已经Manageuser 领域对象creditController方法 | |
| Manageuser. addHotelController | 语法 | public ResultMessage addHotel(String hotlName); | |
| 前置条件 | 已经创建Manageuser领域对象 | |
| 后置条件 | 调用Manageuser 领域对象的addHotel方法 | |
| Manageuser. addHotelStaffController | 语法 | public ResultMessage add(String username, String password); | |
| 前置条件 | 已经创建Manageuser领域对象 | |
| 后置条件 | 调用Manageuser 领域对象的addHotelStaffController方法 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| DatabaseFactory. getUser | | | 获得对User数据库的服务的引用 |
| DatabaseFactory. getCredit | | | 获得对Credit数据库服务的引用 |
| HotelDataService. insert(HotelDetailsPO hotel) | | | 添加单一持久化对象 |
| UserDataService. insert(PersonalPO personal) | | | 插入单一持久化对象 |
| UserDataService. searchClient(String username, String realName) | | | 根据用户名及订房客户的真实姓名进行查找多个持久化对象 |
| UserDataService. getUserInfo(long userID) | | | 根据用户编号进行查找单一持久化对象 |
| UserDataService. update(PersonalPO personal) | | | 更新单一持久化对象 |

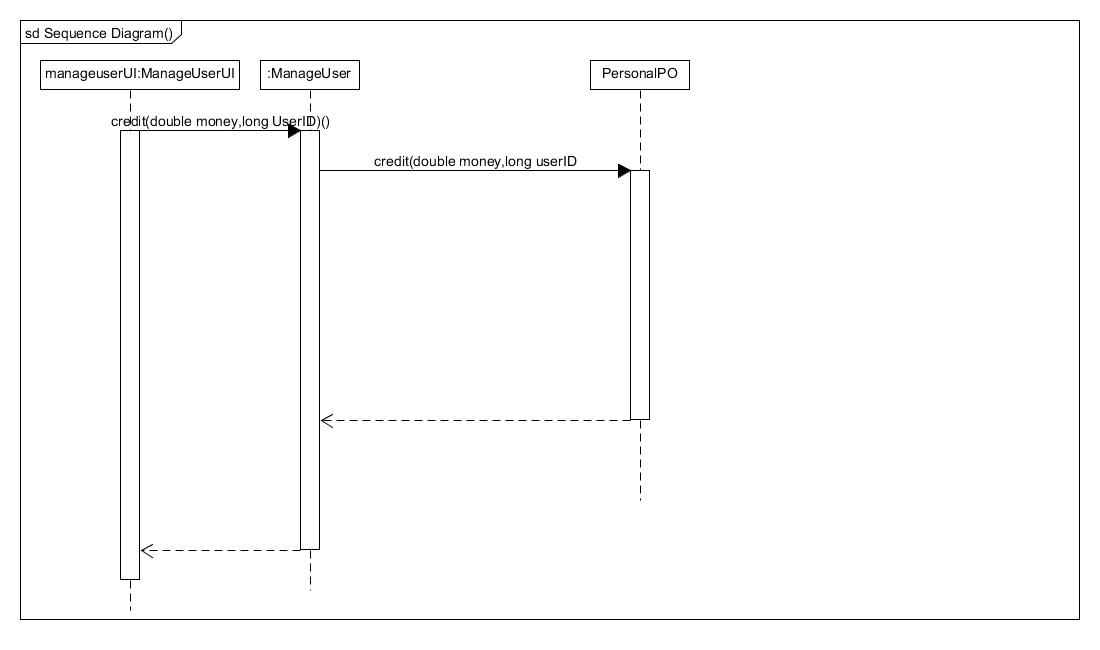
表18 manageuserbl模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Manageuser. searchUser | 语法 | public List<UserPO> searchUser(String username, String realName); | |
| 前置条件 | 已经登录为网站管理人员 | |
| 后置条件 | 返回用户信息列表 | |
| Manageuser. searchHotelStaff | 语法 | public List<HotelStaffPO> searchHotelStaff(int cityID, int districtID); | |
| 前置条件 | 已经登录为网站管理人员 | |
| 后置条件 | 返回符合要求的酒店工作人员信息列表 | |
| Manageuser. getUser | 语法 | public UserPO getUser(long userID); | |
| 前置条件 | 已经登录为网站管理人员 | |
| 后置条件 | 返回该用户的信息 | |
| Manageuser. getUserList | 语法 | public ArrayList<UserPO> getUserList(); | |
| 前置条件 | 无 | |
| 后置条件 | 返回客户名单列表 | |
| Manageuser. sortUserList | 语法 | public ArrayList<UserPO> sortUserList(String way); | |
| 前置条件 | 营销人员对用户排序，已知排序的方式 | |
| 后置条件 | 返回按照要求排序的客户名单列表 | |
| Manageuser. credit | 语法 | public ResultMessage credit(double money, long userID); | |
| 前置条件 | 已登录为网站营销人员 | |
| 后置条件 | 返回充值结果信息，若成功更新客户信用值 | |
| Manageuser. addHotel | 语法 | public ResultMessage addHotel(String hotlName); | |
| 前置条件 | 添加酒店，已知酒店名称 | |
| 后置条件 | 返回添加结果信息，若成功添加酒店数据 | |
| Manageuser. addHotelStaff | 语法 | public ResultMessage add(String username, String password); | |
| 前置条件 | 添加酒店工作人员，已知酒店编号 | |
| 后置条件 | 返回添加结果信息，若成功增加酒店工作人员 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| DatabaseFactory. getUser | | | 获得对User数据库的服务的引用 |
| DatabaseFactory. getCredit | | | 获得对Credit数据库服务的引用 |
| HotelDataService. insert(HotelDetailsPO hotel) | | | 添加单一持久化对象 |
| UserDataService. insert(PersonalPO personal) | | | 插入单一持久化对象 |
| UserDataService. searchClient(String username, String realName) | | | 根据用户名及订房客户的真实姓名进行查找多个持久化对象 |
| UserDataService. getUserInfo(long userID) | | | 根据用户编号进行查找单一持久化对象 |
| UserDataService. update(PersonalPO personal) | | | 更新单一持久化对象 |

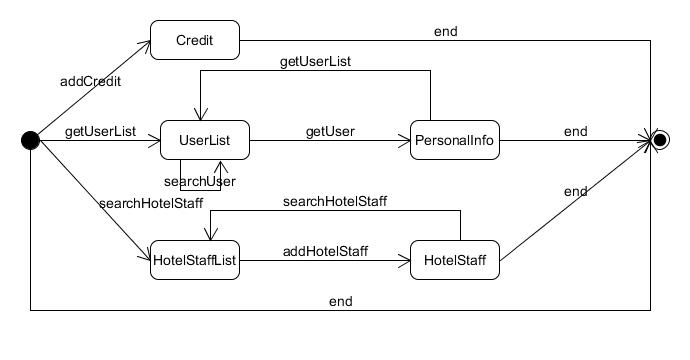
### 4.10.3 manageuserbl模块的行为



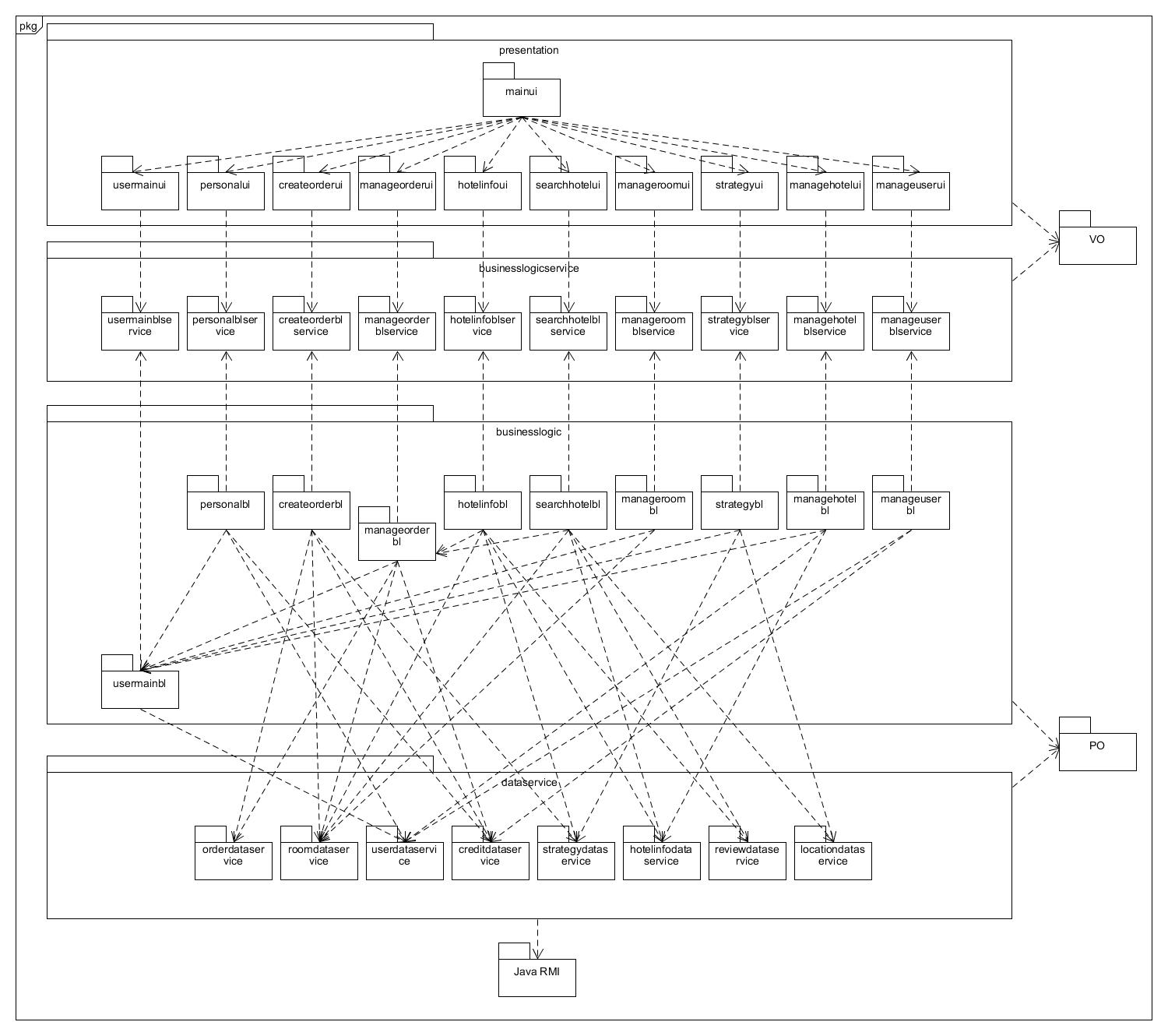
查找用户顺序图



信用充值顺序图



# 五、依赖视角



客户端包图



服务器包图