# 酒店预定系统详细设计文档

学 院：南京大学软件学院

成 员：宋欣建 叶涵 姚政 黄程尔

完成日期：2016年10月29日

## 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 宋欣建 | 2016-10-25 | 最初草稿（创建文档模板） | V0.1 |

## 一、引言

### 1.1 编制目的

本报告详细完成对酒店预定系统的详细设计，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

## 二、中层设计之逻辑层设计

### 2.1 User模块的静态结构与动态行为

#### 2.1.1 局部模块的职责

（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的非功能需求。User模块的职责及接口参见软件系统结构描述文档表5.3.2.1

（2）整体结构

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模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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UserService, UserDAO 两个接口。UserPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.1.1-1 User模块设计

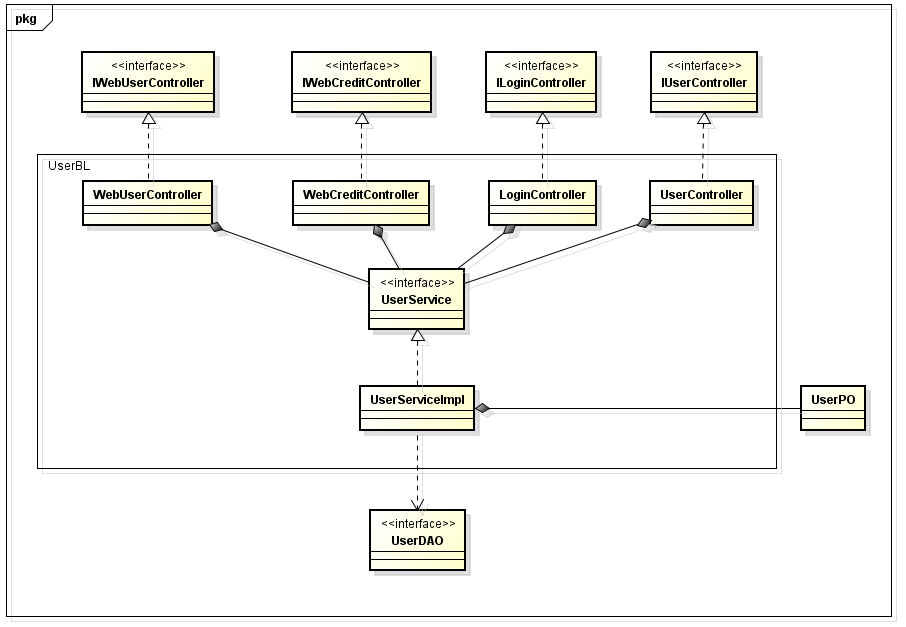


表2.1.1-2 User模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserController | 负责实现对应个人信息界面所需的方法 |
| LoginController | 负责实现对应登录注册界面所需的方法 |
| WebMarketController | 负责实现对应网站信用充值界面所需的方法 |
| WebStaffController | 负责实现对应网站用户信息界面所需的方法 |
| UserServiceImpl | 负责处理有关用户的功能 |

#### 2.1.2 局部模块的接口规范

表2.1.2 -1 UserService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserService.findByUsername | 语法 | public UserVO findByUsername(String username) | |
| 前置条件 | 已知用户名，需要获得该用户的详细信息 | |
| 后置条件 | 无 | |
| UserService.register | 语法 | public ResultMessage register(UserVO) | |
| 前置条件 | 已知用户信息，需要将其注册 | |
| 后置条件 | 无 | |
| UserService.update | 语法 | public ResultMessage update(UserVO) | |
| 前置条件 | 已知该用户修改后的信息，需要更新该用户信息 | |
| 后置条件 | 更新数据库中用户的信息 | |
| UserService.login | 语法 | public UserVO login(String username,String password) | |
| 前置条件 | 已知该用户的用户名和密码，需要将其登录 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserDAO.findByUsername(String username) | | | 查找单一持久化序列 |
| UserDAO.add(UserPO) | | | 添加单一持久化序列 |
| UserDAO.update(UserPO) | | | 更新单一持久化序列 |

#### 2.1.3 局部模块的行为

图2.1.3-1 User模块时序图1

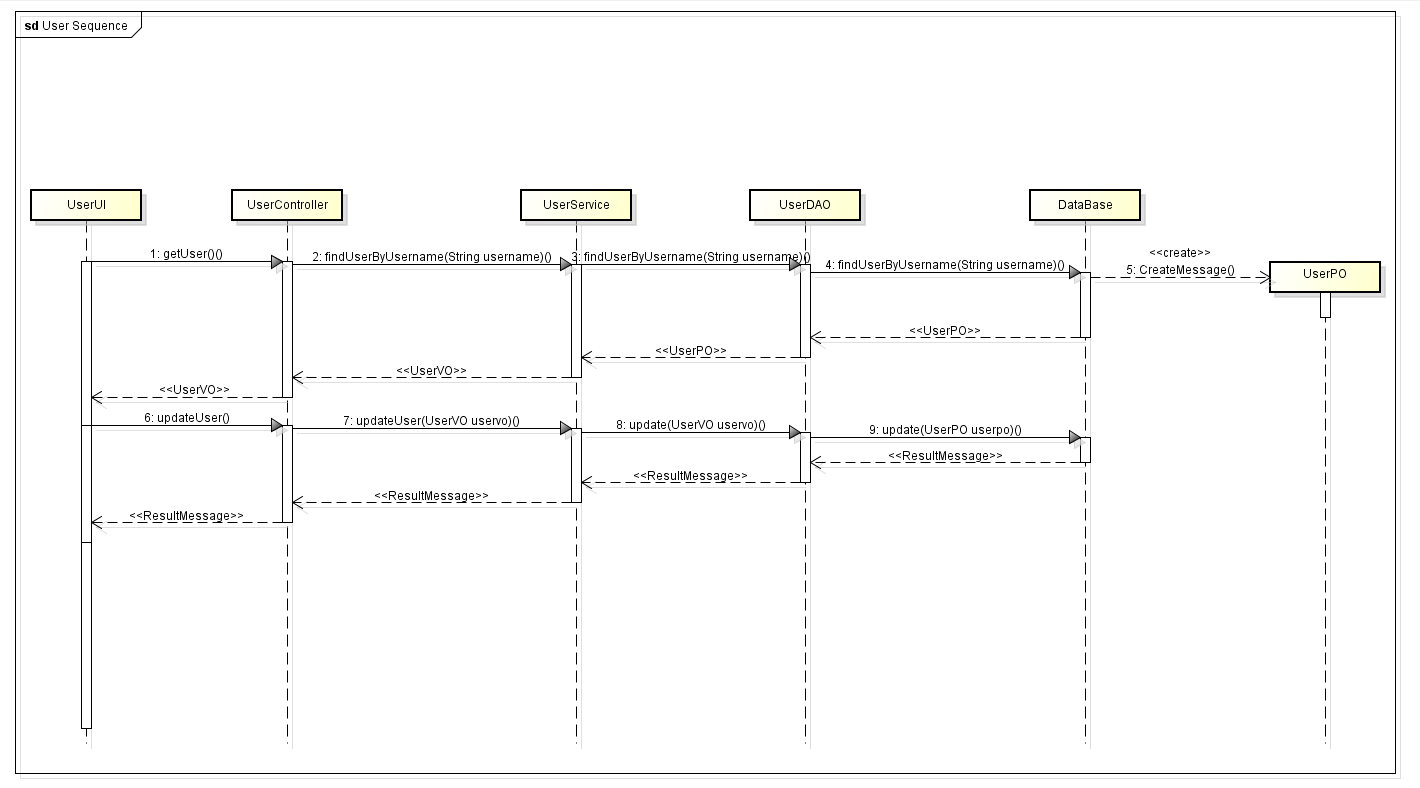


图2.1.3-1 表明了酒店预定系统中，当用户查询和修改用户信息时，用户业务逻辑处理的相关对象之间的协作。

图2.1.3-2 User模块时序图2

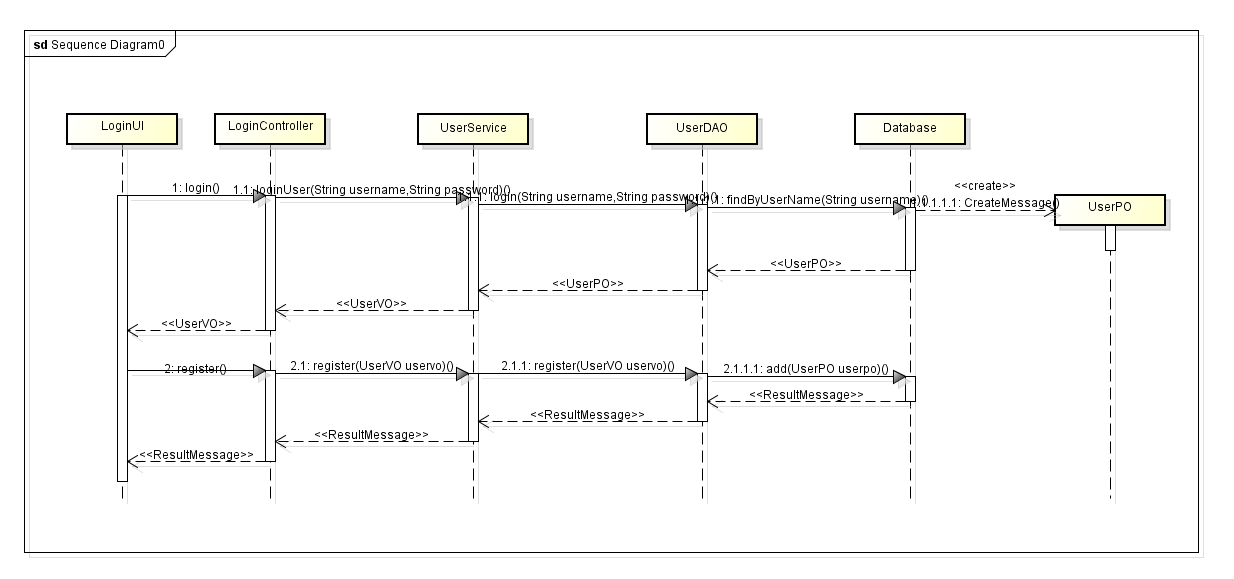
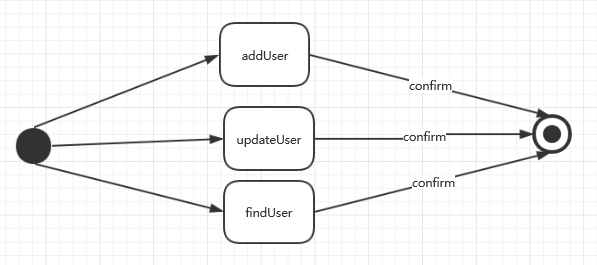


图2.1.3-2 表明了酒店预定系统中，当用户登录注册时，用户业务逻辑处理的相关对象之间的协作。

图2.1.3-2 User模块状态图



### 2.2 Order模块的静态结构与动态行为

#### 2.2.1 OrderSearchService

##### 2.2.1.1 局部模块的职责

（1）模块概述

OrderSearchService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。OrderSearchService模块的职责及接口参见软件系统结构描述文档表5.3.2-2

（2）整体结构

OrderSearchService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。OrderSearchService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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OrderSearchService, OrderDAO两个接口。OrderPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.2.1.1-1 OrderSearchService模块设计

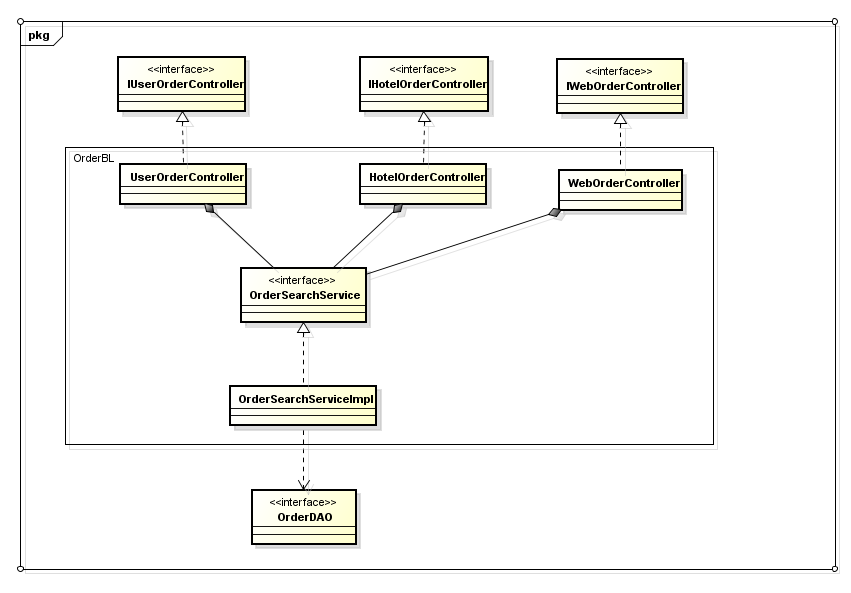


表2.2.1.1-2 OrderSearchServic模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserOrderController | 负责实现对应用户订单信息界面所需的方法 |
| HotelOrderController | 负责实现对应酒店订单信息界面所需的方法 |
| WebOrderController | 负责实现对应网站订单信息界面所需的方法 |
| OrderSearchServiceImpl | 负责处理有关订单搜索的功能 |

##### 2.2.1.2 局部模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| OrderSearchService.findByID | 语法 | public OrderVO findByID(int id) | |
| 前置条件 | 已知该订单的ID，需要得到该订单的详细信息 | |
| 后置条件 | 返回该订单的详细信息 | |
| OrderSearchService.findByUsernameAndType | 语法 | public List<OrderVO>  findByUsernameAndType(String username,OrderStatus) | |
| 前置条件 | 已知订单的所属用户和订单类型，需要获得符合条件的订单列表 | |
| 后置条件 | 无 | |
| OrderSearchService. findByUsername | 语法 | public List<OrderVO> findByUsername(String username) | |
| 前置条件 | 已知订单的所属用户，需要获得该用户的所有订单 | |
| 后置条件 | 无 | |
| OrderSearchService.findByHotelAndUsername | 语法 | public List<OrderVO> findByHotelAndUsername(int hotelID,String username) | |
| 前置条件 | 已知订单的所属酒店和所属用户，需要获得所有订单 | |
| 后置条件 | 无 | |
| OrderSearchService. findByOrderType | 语法 | public List<OrderVO> findByOrderType(OrderStatus) | |
| 前置条件 | 已知订单的类型，需要获得所有该类型的订单 | |
| 后置条件 | 无 | |
| OrderSearchService. findByHotelAndTime | 语法 | public List<OrderVO> findByHotelAndTime(int hotelID,Date begin,Date end) | |
| 前置条件 | 已知酒店和时间，需要得到符合该条件的所有订单 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderDAO. findByID(int ID) | | | 查找单一持久化序列 |
| OrderDAO. findByUsernameAndType(String username,OrderStatus) | | | 查找单一持久化列表 |
| OrderDAO. findByUsername(String username) | | | 查找单一持久化列表 |
| OrderDAO. findByHotelAndUsername(int hotelID,String username | | | 查找单一持久化列表 |
| OrderDAO.findByOrderType(OrderStatus) | | | 查找单一持久化列表 |
| OrderDAO. findByHotelAndTime(int hotelID,Date begin,Date end); | | | 查找单一持久化列表 |

##### 2.2.1.3 局部模块的行为

图2.2.1.3-1 OrderSearchService模块时序图1

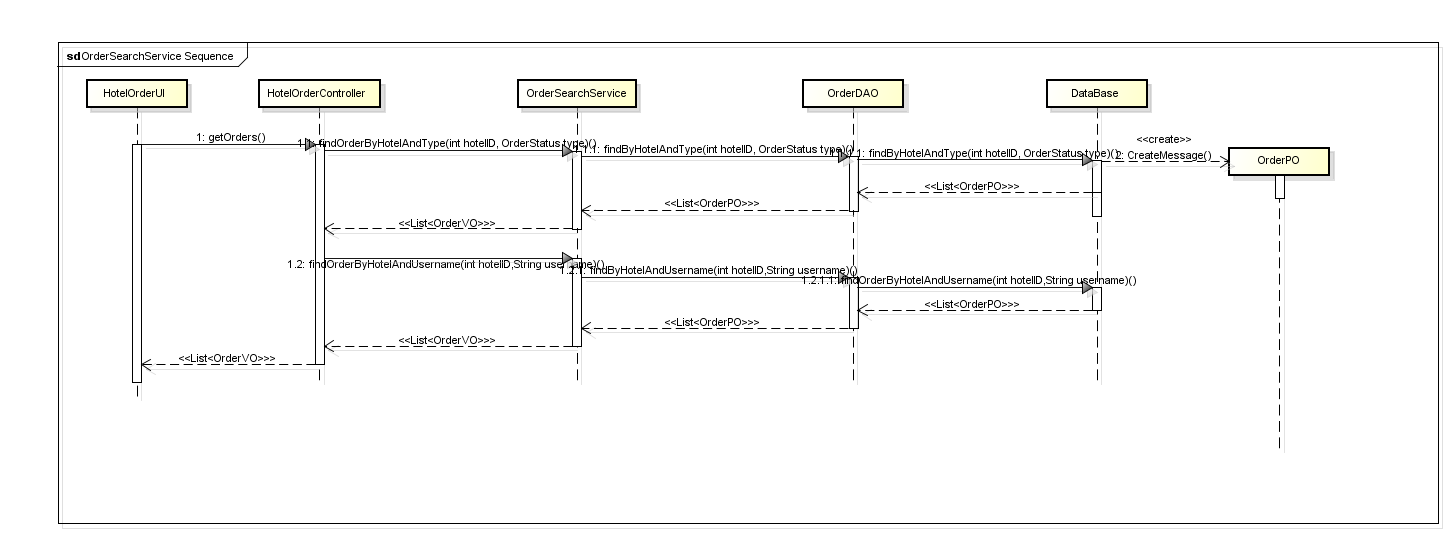


图2.2.1.3-1 表明了酒店预定系统中，当酒店管理人员查询订单信息时，订单业务逻辑处理的相关对象之间的协作。

图2.2.1.3-2 OrderSearchService模块时序图2

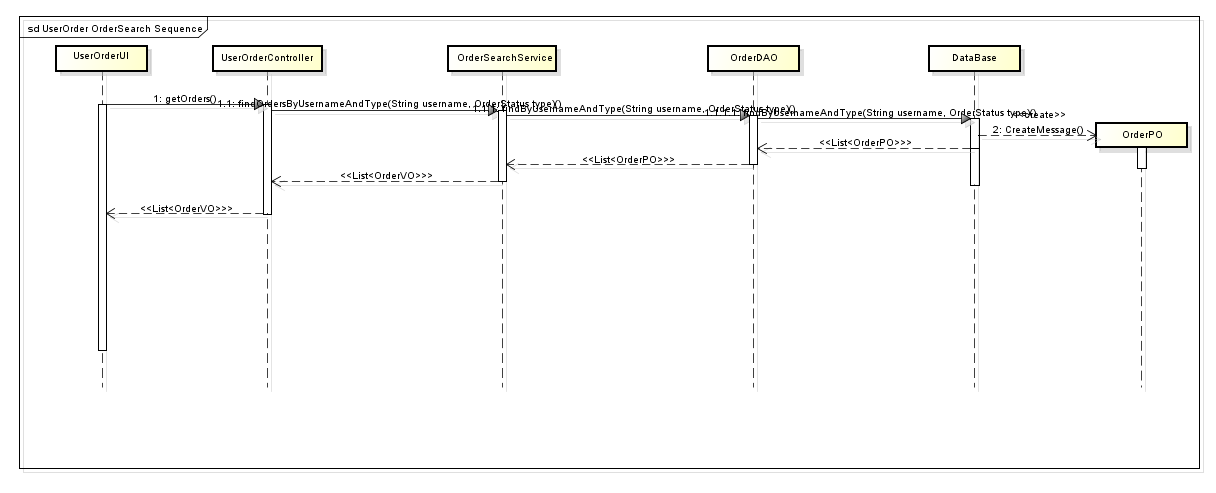


图2.2.1.3-2 表明了酒店预定系统中，当用户查询订单信息时，订单业务逻辑处理的相关对象之间的协作。

图2.2.1.3-3 OrderSearchService模块时序图2

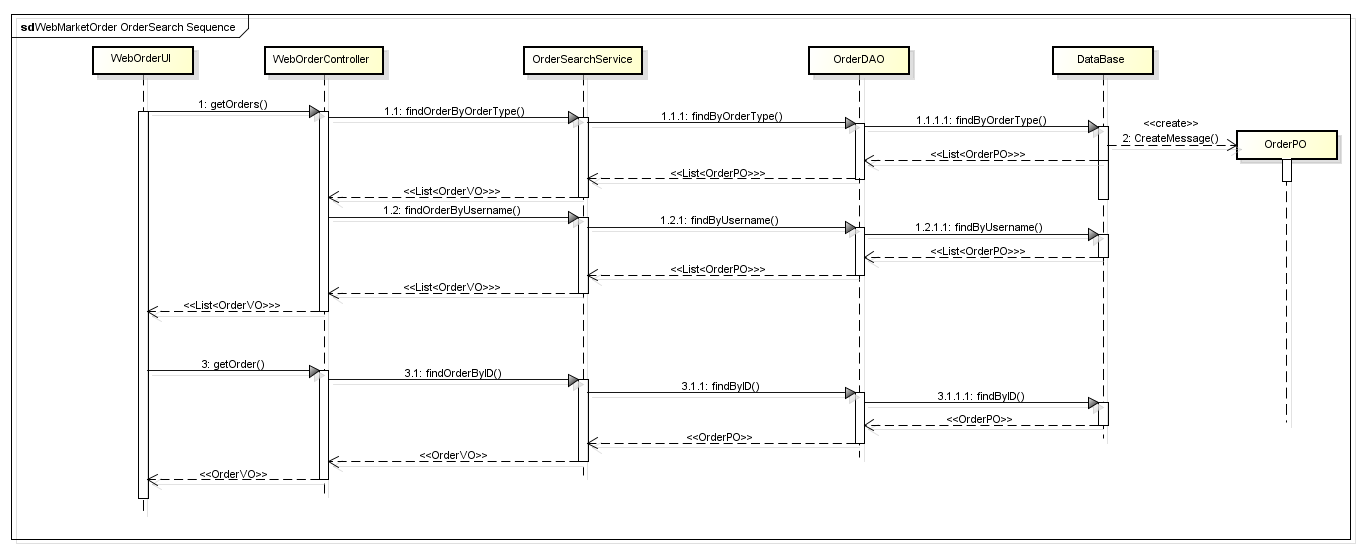
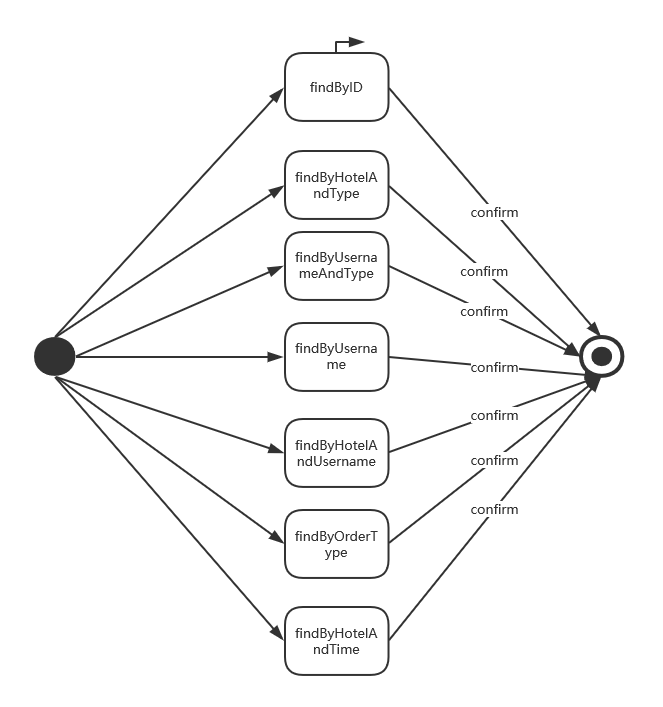


图2.2.1.3-3 表明了酒店预定系统中，当网站营销人员查询订单信息时，订单业务逻辑处理的相关对象之间的协作。

图2.2.1.3-4 OrderSearchService模块状态图



#### 2.2.2 OrderService

##### 2.2.2.1 局部模块的职责

（1）模块概述

OrderService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。OrderService模块的职责及接口参见软件系统结构描述文档表5.3.2-2

（2）整体结构

OrderService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。OrderService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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OrderService, OrderDAO两个接口。OrderPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.2.2.1-1OrderService模块设计

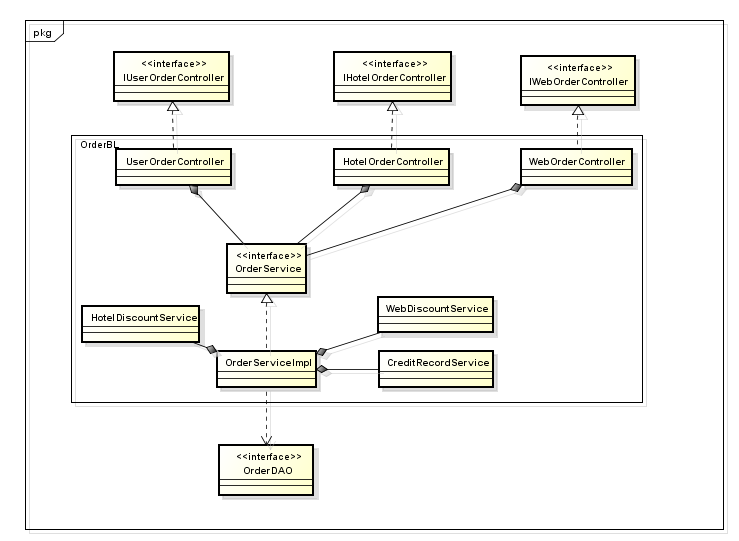


表2.2.2.1-2 OrderService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserOrderController | 负责实现对应用户订单信息界面所需的方法 |
| HotelOrderController | 负责实现对应酒店管理界面所需的方法 |
| WebOrderController | 负责实现对应网站管理界面所需的方法 |
| OrderServiceImpl | 负责处理有关订单的功能 |

##### 2.2.2.2 局部模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| OrderService.  placeOrder | 语法 | public OrderVO placeOrder(OrderVO) | |
| 前置条件 | 已知该订单部分信息，需要将订单信息进行完善 | |
| 后置条件 | 更新订单对象的信息 | |
| OrderService.add | 语法 | public ResultMessage add(OrderVO) | |
| 前置条件 | 已知该订单信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中添加该订单信息 | |
| OrderService.checkin | 语法 | public ResultMessage checkin(OrderVO ) | |
| 前置条件 | 已知订单信息，需要将该订单状态改为已入住 | |
| 后置条件 | 在数据库中修改该订单状态 | |
| OrderService.checkout | 语法 | public ResultMessage checkout(OrderVO ordervo) | |
| 前置条件 | 已知订单信息，需要将该订单的状态改为已退房 | |
| 后置条件 | 在数据库中修改该订单状态 | |
| OrderService.revoke | 语法 | public ResultMessage revoke(OrderVO ) | |
| 前置条件 | 已知订单信息，需要将该订单的状态改为已撤销 | |
| 后置条件 | 在数据库中修改该订单状态 | |
| OrderService.remark | 语法 | public ResultMessage remark(OrderVO ) | |
| 前置条件 | 已知订单信息，需要增加该订单的评价信息 | |
| 后置条件 | 在数据库中修改该订单状态 | |
| OrderService.delayCheckin | 语法 | public ResultMessage delayCheckin(OrderVO ) | |
| 前置条件 | 已知订单信息，需要将该订单的状态改为已入住 | |
| 后置条件 | 在数据库中修改该订单状态 | |
| 需要的服务（需接口） | | | |
| OrderDAO. add(OrderPO) | | | 添加单一持久化列表 |
| OrderDAO. update(OrderPO) | | | 更新单一持久化序列 |
| HotelDiscountService.findAllByHotelID(int hotelID) | | | 返回该酒店的所有优惠策略 |
| CreditRecordService. add(CreditRecordVO) | | | 添加当前用户的一条信用记录 |
| WebDiscountService.findAll() | | | 返回网站的所有优惠策略 |

##### 2.2.2.3 局部模块的行为

图2.2.2.3-1 OrderSearchService模块时序图1

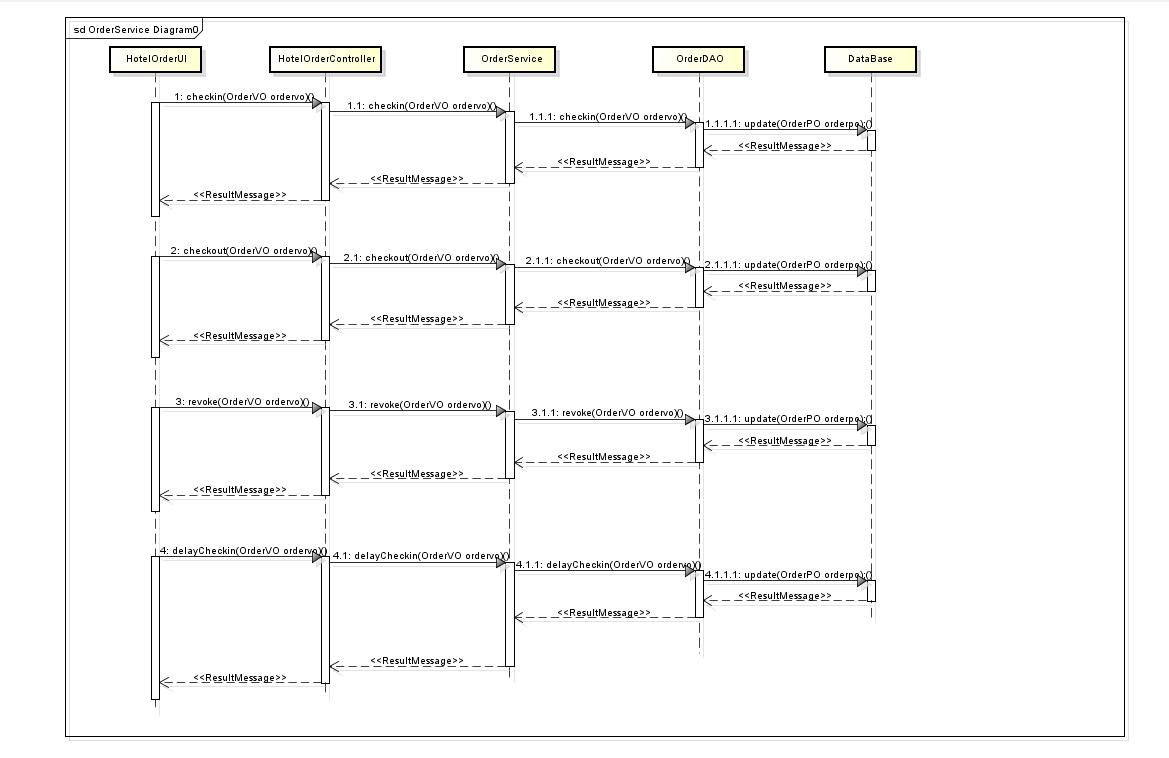


图2.2.2.3-1 表明了酒店预定系统中，当酒店管理人员进行订单管理时，订单业务逻辑处理的相关对象之间的协作。

图2.2.2.3-2 OrderSearchService模块时序图2

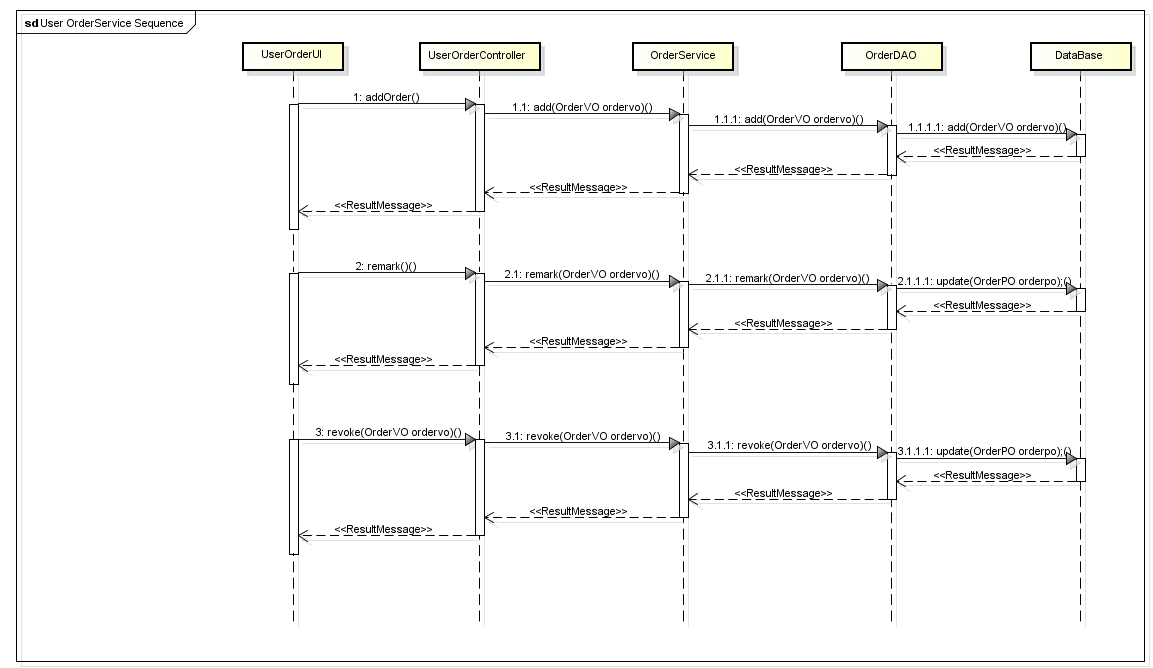


图2.2.2.3-2 表明了酒店预定系统中，当用户进行订单管理时，订单业务逻辑处理的相关对象之间的协作。

图2.2.2.3-3 OrderSearchService模块时序图3

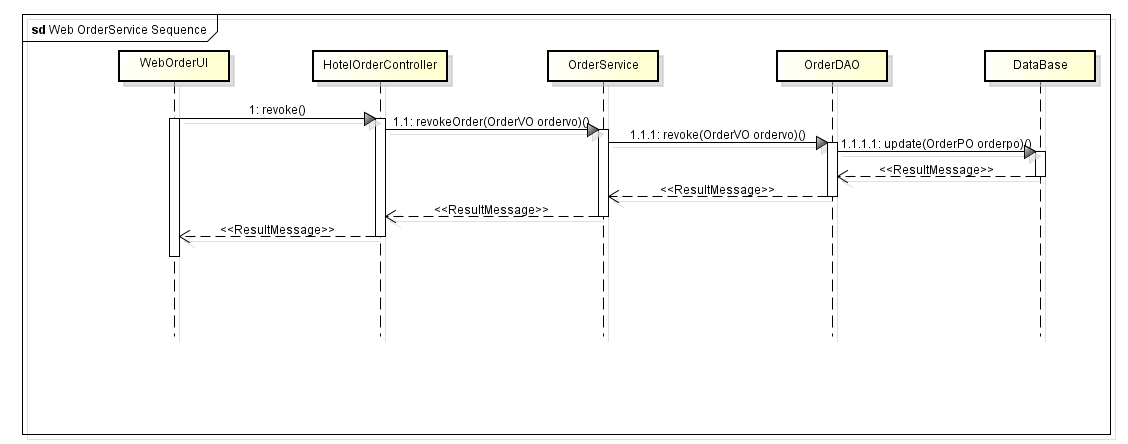
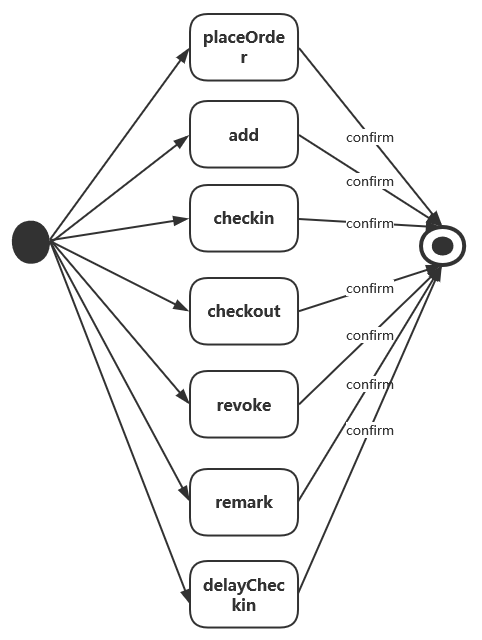


图2.2.2.3-3 表明了酒店预定系统中，当网站营销人员进行订单管理时，订单业务逻辑处理的相关对象之间的协作。

图2.2.1.3-4 OrdeService模块状态图



### 2.3 Hotel模块的静态结构与动态行为

#### 2.3.1 HotelService

##### 2.3.1.1 局部模块的职责

（1）模块概述

HotelService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。HotelService模块的职责及接口参见软件系统结构描述文档表5.3.2-3

（2）整体结构

HotelService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。HotelService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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HotelService, HotelDAO两个接口。HotelPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.3.1.1-1 HotelService模块设计

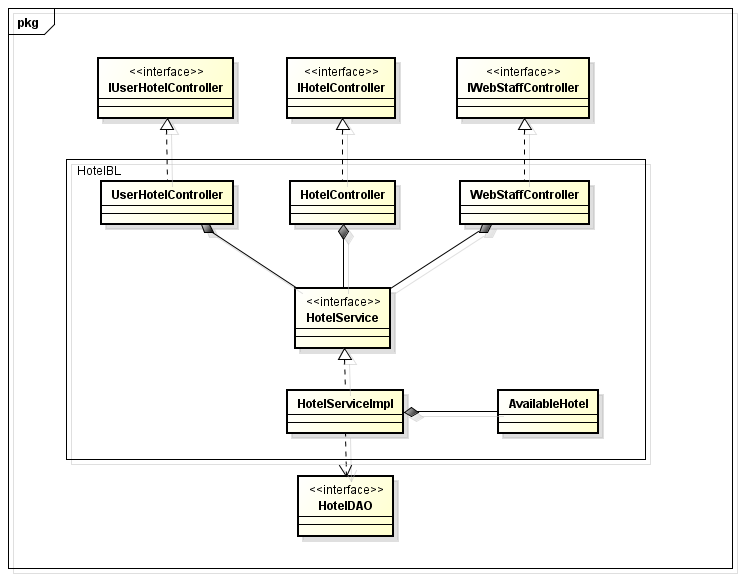


表2.3.1.1-2 HotelService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserHotelController | 负责实现对应用户酒店信息界面所需的方法 |
| HotelController | 负责实现对应酒店管理界面所需的方法 |
| WebStaffController | 负责实现对应网站管理人员酒店添加界面所需的方法 |
| HotelServiceImpl | 负责处理有关酒店的功能 |

##### 2.3.1.2 局部模块的接口规范

表2.3.1.2 -1 HotelService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelService. findByID | 语法 | public HotelVO findByID(int hotelID) | |
| 前置条件 | 已知该酒店的编号，需要该酒店的详细信息 | |
| 后置条件 | 无 | |
| HotelService.update | 语法 | public ResultMessage update(HotelVO) | |
| 前置条件 | 已知该酒店的更新后的信息，需要更新该酒店信息 | |
| 后置条件 | 在数据库中更新该酒店信息 | |
| HotelService.add | 语法 | public ResultMessage add(HotelVO) | |
| 前置条件 | 已知该酒店的信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中增加该酒店信息 | |
| HotelService.findOrderedHotelAndOrder | 语法 | public Map<HotelVO,List<OrderVO>>>  findOrderedHotelAndOrder(String username) | |
| 前置条件 | 已知用户名，需要用户曾预订的酒店列表以及每个酒店对应的订单信息 | |
| 后置条件 | 返回酒店列表 | |
| HotelService. find | 语法 | public Map<HotelVO,List<RoomVO>>>  find (int loc,int circle,Date begin,Date end) | |
| 前置条件 | 已知该酒店的地址、商圈、订单时间，得到可用房间 | |
| 后置条件 | 无 | |
| HotelService. filter | 语法 | public Map<HotelVO,List<RoomVO>>  filter(List<FilterCondition>) | |
| 前置条件 | 已知酒店列表，需要按照给定的过滤条件进行筛选 | |
| 后置条件 | 无 | |
| HotelService. order | 语法 | public Map<HotelVO,List<RoomVO>>  order(OrderRule rule,boolean isDecrease); | |
| 前置条件 | 已知酒店列表，需要按照给定条件进行排序 | |
| 后置条件 | 返回排好序的酒店列表 | |
| HotelService. getRoomDetail | 语法 | public List<RoomVO>  getRoomDetail(,int hotelID) | |
| 前置条件 | 已知酒店编号，需要取出该酒店的对应的房间信息 | |
| 后置条件 | 无 | |
| LocationService.findAll | 语法 | public List<LocationVO> findAll() | |
| 前置条件 | 需要所有的地点信息 | |
| 后置条件 | 无 | |
| CommCircleService.findByLoc | 语法 | public List<CommercialCircleVO> findByLocation(int locID) | |
| 前置条件 | 已知地点，需要该地点的所有商圈 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelDAO.findByID(int hotelID) | | | 查找单一持久化序列 |
| HotelDAO.update(HotelPO) | | | 更新单一持久化序列 |
| HotelDAO.add(HotelPO) | | | 增加单一持久化序列 |
| HotelDAO. find (int loc,int circle,Date begin,Date end) | | | 查找单一持久化列表 |
| LocationDAO. findAll() | | | 查找单一持久化列表 |
| CommCircleDAO. findByLoc(int locID) | | | 查找单一持久化列表 |
| OrderService.findByUsername(String username) | | | 返回该用户的所有订单列表 |
| OrderService. findByHotelAndUsername(int hotelID,String username) | | | 返回该酒店的该用户的订单记录列表 |
| RoomService. findAvailableByHotelID (int hotelID) | | | 返回该酒店的所有可用房间信息 |

##### 2.3.1.3 局部模块的行为

图2.3.1.3-1 HotelService模块时序图1

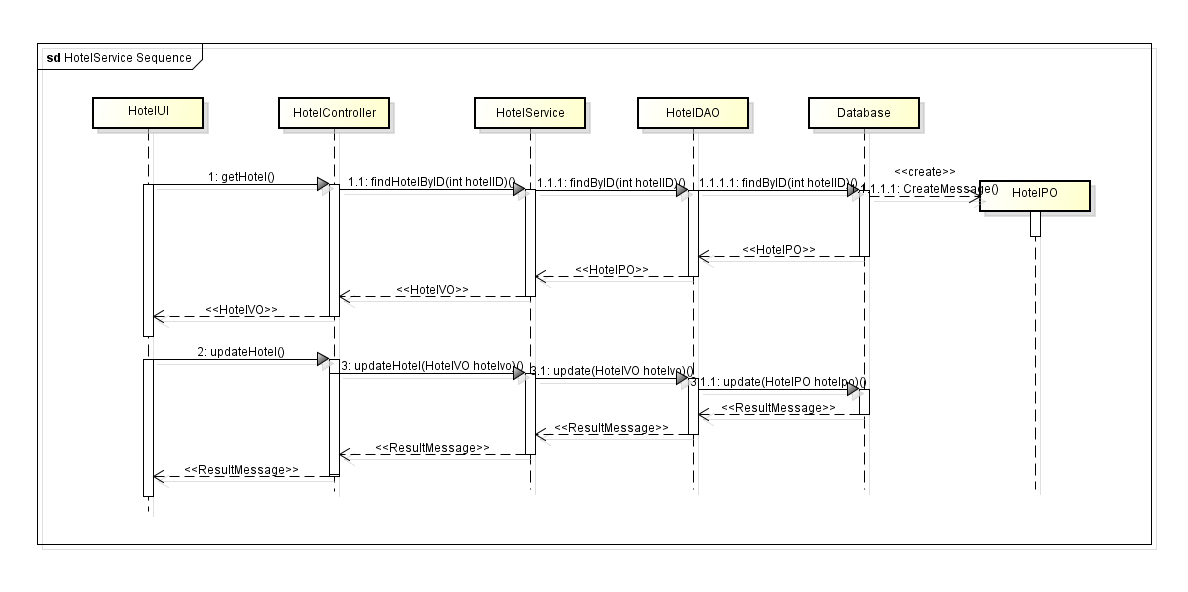


图2.1.3-1 表明了酒店预定系统中，当酒店管理人员查询和修改酒店信息时，酒店业务逻辑处理的相关对象之间的协作。

图2.3.1.3-2 HotelService模块时序图2

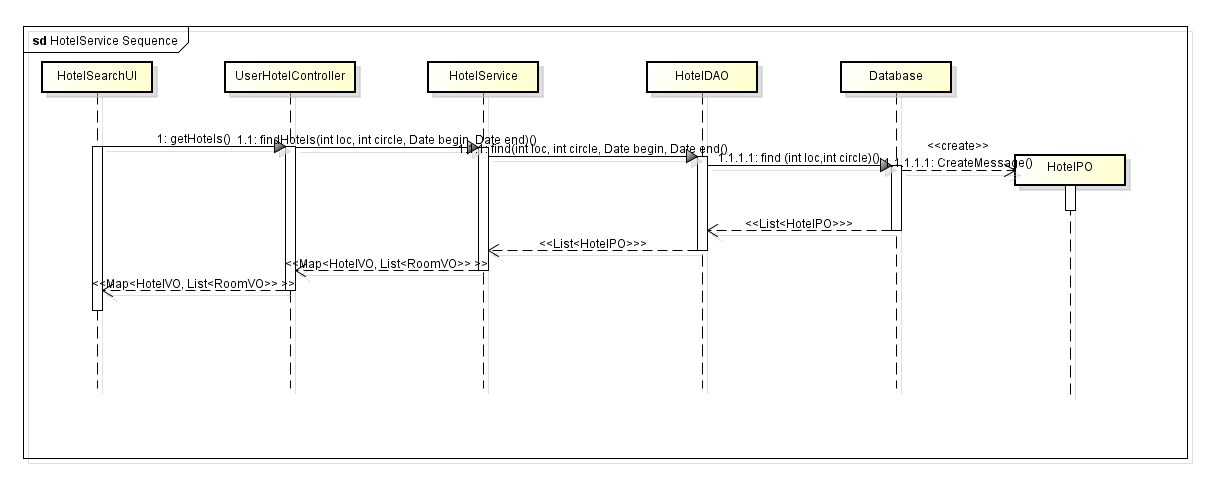
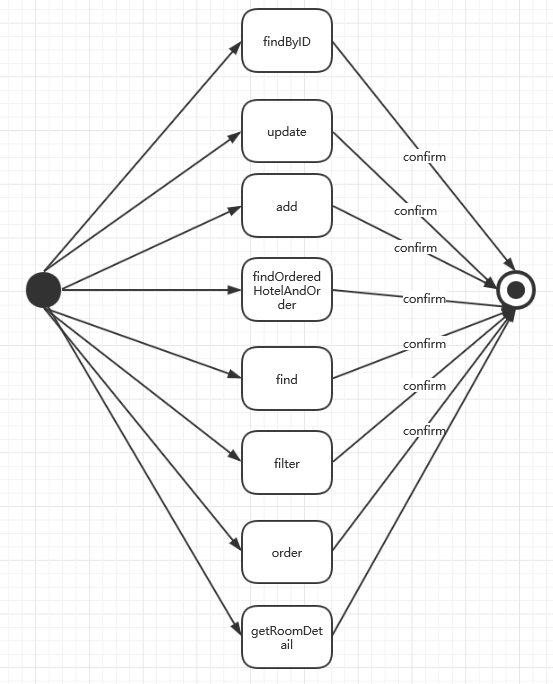


图2.3.1.3-2 表明了酒店预定系统中，当用户查询酒店信息时，酒店业务逻辑处理的相关对象之间的协作。

图2.3.1.3-3 HotelService模块状态图



#### 2.3.2 LocationService

##### 2.3.2.1 局部模块的职责

（1）模块概述

LocationService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。LocationService模块的职责及接口参见软件系统结构描述文档表5.3.2-3

（2）整体结构

LocationService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。LocationService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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LocationService, LocationDAO两个接口。LocationPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.3.1.1-1 LocationService模块设计

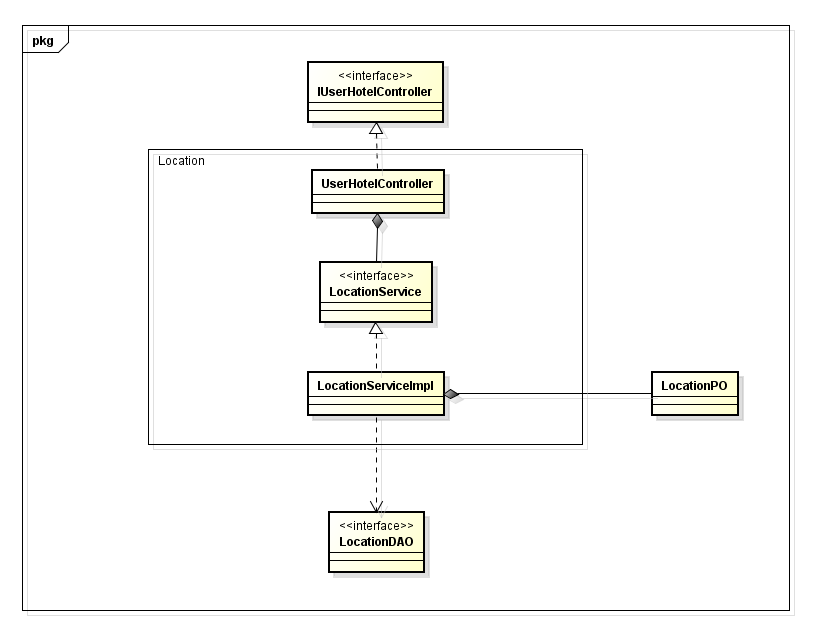


表2.3.1.1-2 LocationService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserHotelController | 负责实现对应用户酒店信息界面所需的方法 |
| LocationServiceImpl | 负责处理有关城市的功能 |

##### 2.3.2.2 局部模块的接口规范

表2.3.2.2 -1 LocationService的接口规范

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 提供的服务（供接口） | | | | |
| LocationService.findAll | 语法 | public List<LocationVO> findAll() | | |
| 前置条件 | 需要所有的地点信息 | | |
| 后置条件 | 无 | | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| LocationDAO. findAll() | | | 查找单一持久化列表 |

##### 2.3.2.3 局部模块的行为

图2.3.2.3-1 LocationService模块时序图

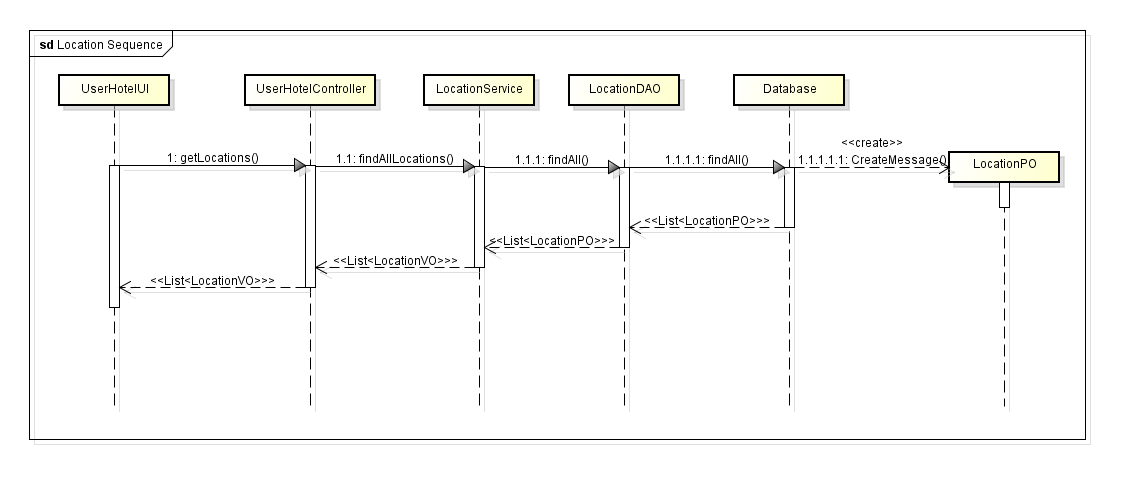
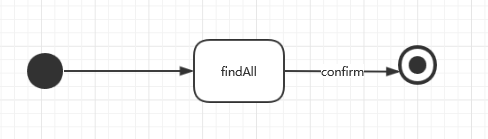


图2.3.2.3-1 表明了酒店预定系统中，当用户查询酒店信息时，城市业务逻辑处理的相关对象之间的协作。

图2.3.2.3-2 LocationService模块状态图



#### 2.3.3 CommCircleService

##### 2.3.3.1 局部模块的职责

（1）模块概述

CommCircleService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。CommCircleService模块的职责及接口参见软件系统结构描述文档表5.3.2-3

（2）整体结构

CommCircleService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。CommCircleService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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CommCircleService, CommCircleDAO两个接口。CommercialCirclePO是做为管理信息的持久化对象被添加到设计模型中的。

图2.3.3.1-1 CommCircleService模块设计

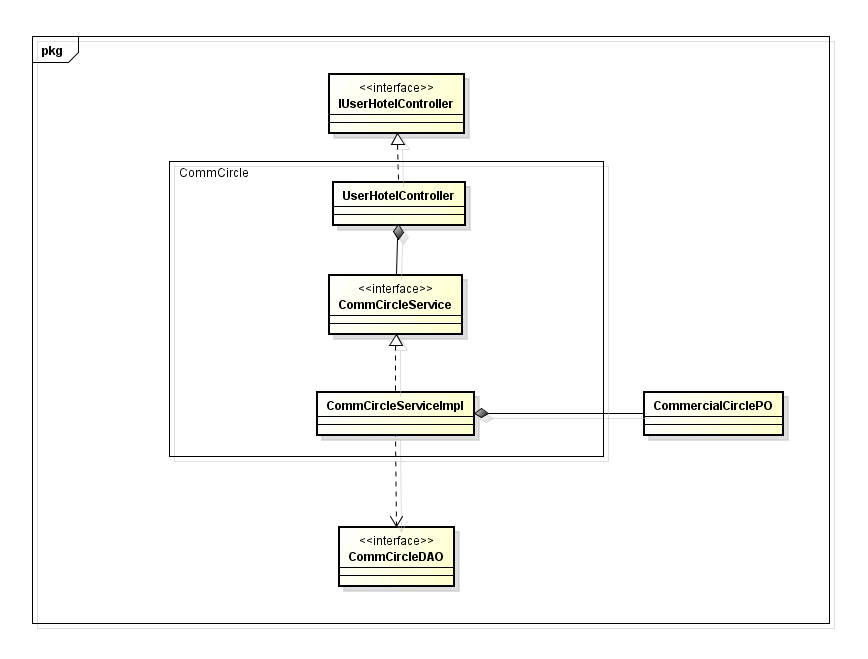


表2.3.1.1-2 CommCircleService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| UserHotelController | 负责实现对应用户酒店信息界面所需的方法 |
| CommCircleServiceImpl | 负责处理有关商圈的功能 |

##### 2.3.3.2 局部模块的接口规范

表2.3.3.2 -1 CommCircleService的接口规范

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 提供的服务（供接口） | | | | |
| CommCircleService.findByLoc | 语法 | public List<CommercialCircleVO> findByLocation(int locID) | | |
| 前置条件 | 已知地点，需要该地点的所有商圈 | | |
| 后置条件 | 无 | | |
| 需要的服务（需接口） | | | |
| 服务名 | | | | 服务 |
| CommCircleDAO. findByLoc(int locID) | | | | 查找单一持久化列表 |

##### 2.3.2.3 局部模块的行为

图2.3.2.3-1 CommCircleService模块时序图

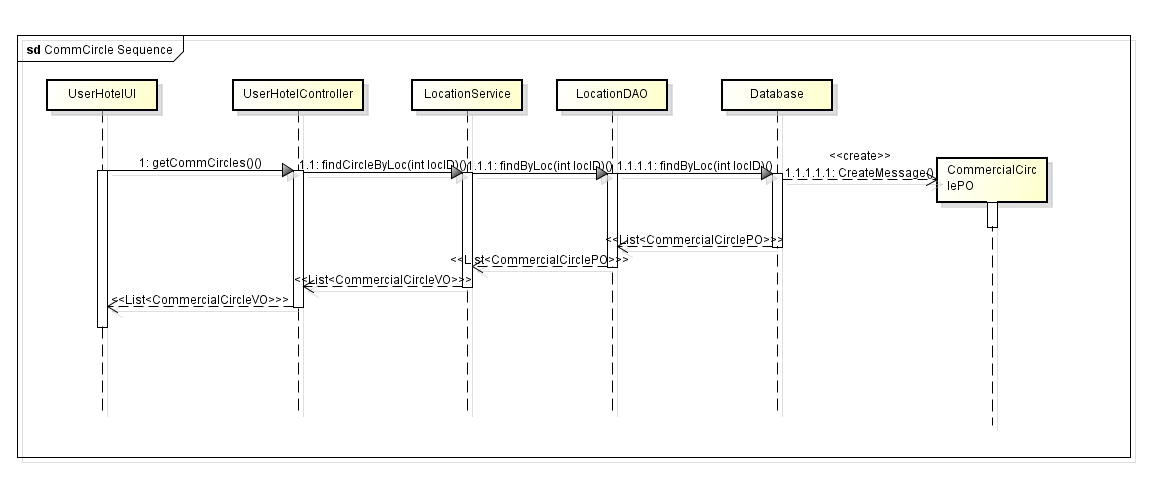
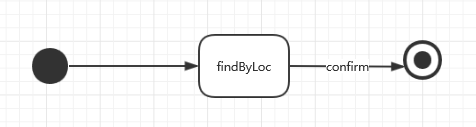


图2.3.2.3-1 表明了酒店预定系统中，当用户查询酒店信息时，商圈业务逻辑处理的相关对象之间的协作。

图2.3.2.3-2 CommCircleService模块状态图



### 2.4 Room模块的静态结构与动态行为

#### 2.4.1 局部模块的职责

（1）模块概述

Room模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。Room模块的职责及接口参见软件系统结构描述文档表5.3.2.4

（2）整体结构

Room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。Room模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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RoomService, RoomDAO 两个接口。RoomPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.4.1-1 Room模块设计

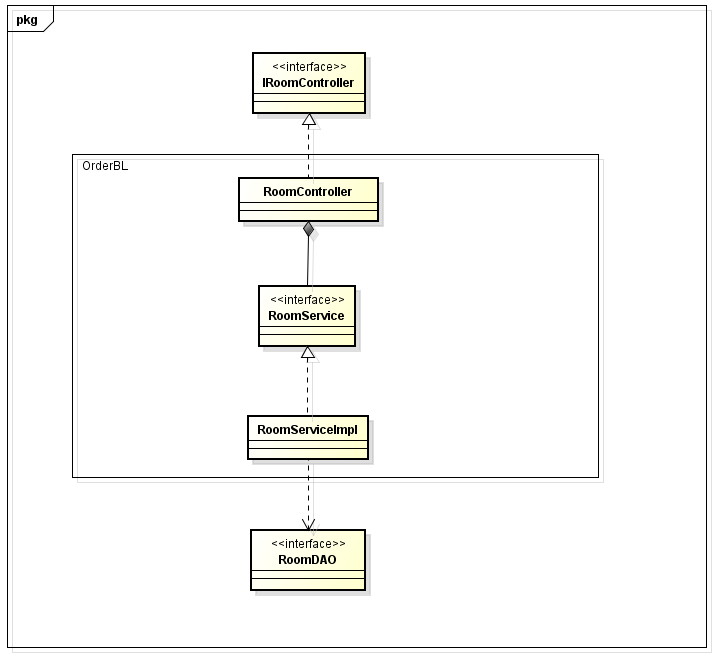


表2.4.1-2 Room模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| RoomController | 负责实现对应个人信息界面所需的方法 |
| RoomServiceImpl | 负责处理有关房间信息的功能 |

#### 2.4.2 局部模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| RoomService. findByHotelID | 语法 | public List<RoomVO> findByHotelID (int hotelID) | |
| 前置条件 | 已知酒店，需要房间详细信息列表 | |
| 后置条件 | 无 | |
| RoomService.findAvailableByHotelID | 语法 | public List<RoomVO> findAvailableByHotelID(int hotelID,Date begin,Date end) | |
| 前置条件 | 已知酒店和时间，需要在该时间段内有效的房间信息 | |
| 后置条件 | 无 | |
| RoomService.findNotAddedRoomType | 语法 | List<RoomType> findNotAddedRoomType(int hotelID) | |
| 前置条件 | 已知酒店编号，需要该酒店尚未添加过的房间类型列表 | |
| 后置条件 | 无 | |
| RoomService.update | 语法 | public ResultMessage update(RoomVO) | |
| 前置条件 | 已知房间更新后的信息，需要更新数据库中的信息 | |
| 后置条件 | 在数据库中更新房间信息 | |
| RoomService.add | 语法 | public ResultMessage add(RoomVO) | |
| 前置条件 | 已知房间信息，需要将其添加到数据库中 | |
| 后置条件 | 在数据库中保存新添加的房间信息 | |
| RoomService. findAvailableRoomNum | 语法 | public int findAvailableRoomNum(int hotelID,RoomType type,Date begin,Date end) | |
| 前置条件 | 已知酒店、房间类型和时间，需要可用房间数量 | |
| 后置条件 | 返回可用房间数量 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| RoomDAO. findByHotelAndType(int hotelID,RoomType) | | | 查找单一持久化序列 |
| RoomDAO. findByHotelID(int hotelID) | | | 查找单一持久化列表 |
| RoomDAO. add(RoomPO) | | | 增加单一持久化序列 |
| RoomDAO. update(RoomPO) | | | 更新单一持久化序列 |
| RoomDAO. findAvailableRoomNum (int hotelID,RoomType type,Date begin,Date end) | | | 返回在当前酒店和房间类型下一段时间内可用房间数量 |

#### 2.4.3 局部模块的行为

图2.4.3-1 RoomService模块时序图1

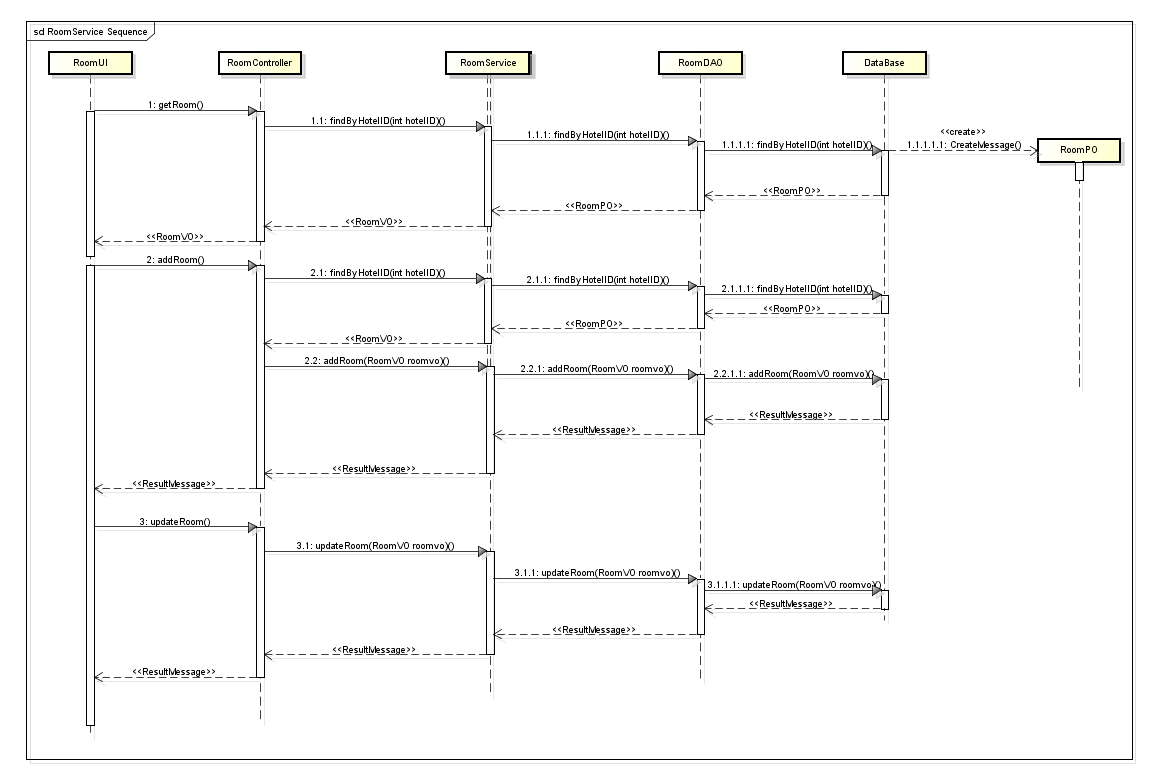
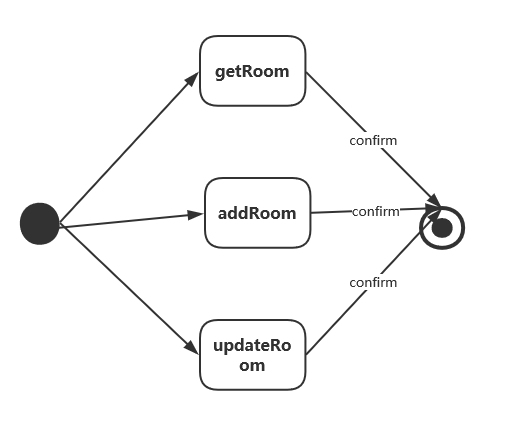


图24. 3-1 表明了酒店预定系统中，当酒店管理人员查询，修改，增加可用房间信息时时，订单业务逻辑处理的相关对象之间的协作。

图2.4.3-2 OrderSearchService模块状态图



### 2.5 Promotion模块的静态结构与动态行为

#### 2.5.1 HotelDiscountService

#### 2.5.1.1 局部模块的职责

（1）模块概述

HotelDiscountService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。HotelDiscountService模块的职责及接口参见软件系统结构描述文档表5.3.2-7

（2）整体结构

HotelDiscountService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。HotelDiscountService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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HotelDiscountService, HotelDiscountDAO 两个接口。HotelDiscountPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.5.1.1-1 HotelDiscountService模块设计

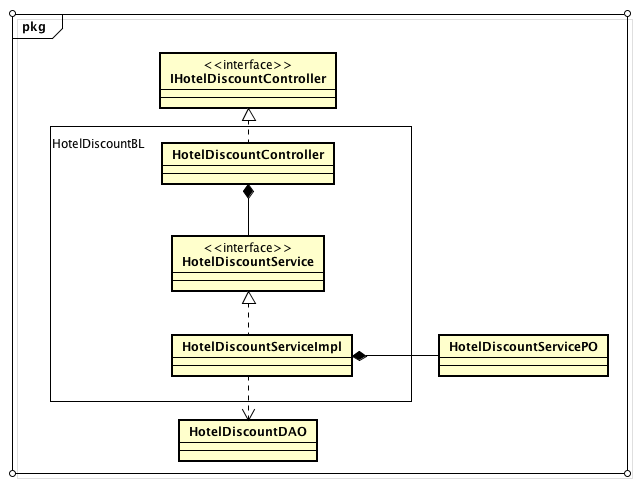


表2.5.1.1-2 HotelDiscountService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| HotelDiscountController | 负责实现对应酒店折扣界面所需的方法 |
| HotelDiscountServiceImpl | 负责处理有关酒店折扣的功能 |

#### 2.5.1.2 局部模块的接口规范

表2.5.1.2 -1 HotelDiscountService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelDiscountService. add | 语法 | public ResultMessage add (HotelDiscountVO) | |
| 前置条件 | 已知酒店促销策略信息，需要将其添加到数据库 | |
| 后置条件 | 返回添加的结果信息 | |
| HotelDiscountService.update | 语法 | public ResultMessage update(HotelDiscountVO) | |
| 前置条件 | 已知更新后的酒店促销策略信息，需要更新数据库 | |
| 后置条件 | 返回更新的结果信息 | |
| HotelDiscountService.delete | 语法 | public ResultMessage delete(int id) | |
| 前置条件 | 已知酒店优惠策略的编号，需要将其删除 | |
| 后置条件 | 返回删除的结果信息 | |
| HotelDiscountService.findAllByHotelID | 语法 | public List<HotelDiscountVO> findAllByHotelID(int hotelID) | |
| 前置条件 | 已知酒店编号，需要该酒店所有的优惠策略 | |
| 后置条件 | 返回该酒店的所有优惠策略 | |
| HotelDiscountService.findNotAddedType | 语法 | List<HotelDiscountType> findNotAddedType(int hotelID); | |
| 前置条件 | 已知酒店编号，需要返回该酒店尚未添加过的酒店促销策略类型 | |
| 后置条件 | 无 | |
| HotelDiscountService.createAllStrategies | 语法 | public List<HotelDiscount> createAllStrategies(int hotelID); | |
| 前置条件 | 已知酒店编号，需要返回该酒店的所有策略领域对象 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelDiscountDAO.add (HotelDiscountPO) | | | 添加单一持久化序列 |
| HotelDiscountDAO.update(HotelDiscountPO) | | | 更新单一持久化序列 |
| HotelDiscount.delete(int id) | | | 删除单一持久化序列 |
| HotelDiscount.findAllByHotelID(int hotelID); | | | 查询单一持久化列表 |

#### 2.5.1.3 局部模块的行为

图2.5.1.3-1 HotelDiscountService模块时序图

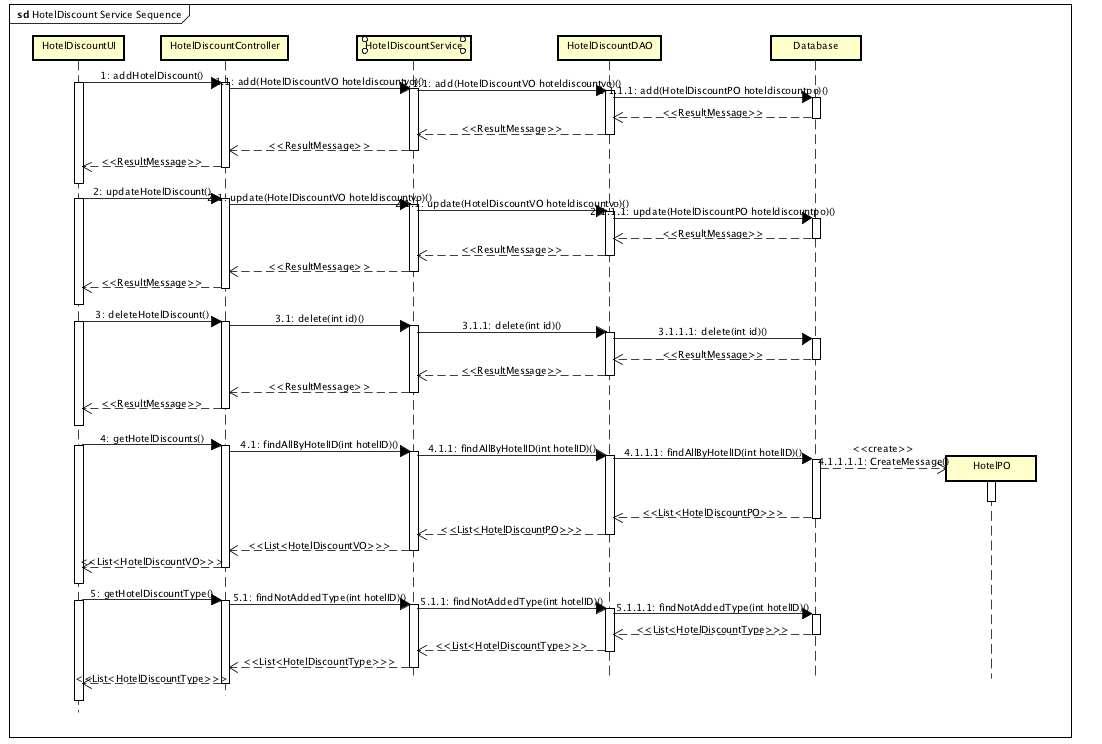
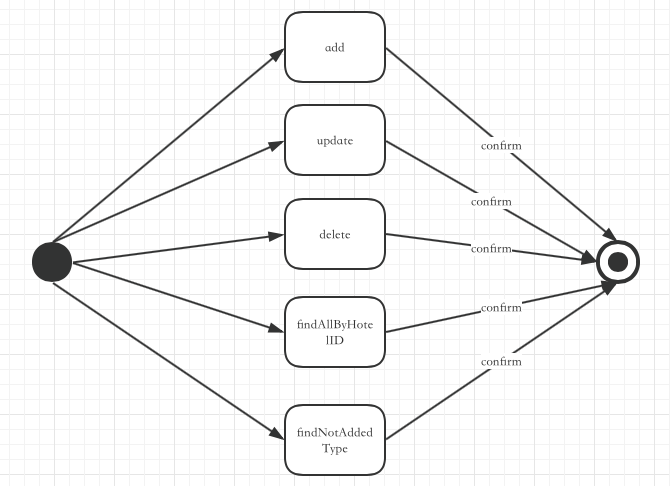


图2.5.1.3-1 表明了酒店预定系统中，当酒店工作人员添加、更新、删除、查看酒店折扣时，酒店折扣业务逻辑处理的相关对象之间的协作。

图2.5.1.3-2 HotelDiscountService模块状态图



#### 2.5.2 WebDiscountService

#### 2.5.2.1 局部模块的职责

（1）模块概述

WebDiscountService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。WebDiscountService模块的职责及接口参见软件系统结构描述文档表5.3.2-9

（2）整体结构

WebDiscountService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。WebDiscountService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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WebDiscountService, WebDiscountDAO 两个接口。WebDiscountPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.5.2.1-1 WebDiscountService模块设计

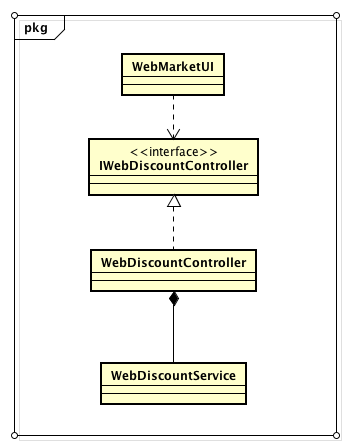


表2.5.2.1-2 WebDiscountService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| WebDiscountController | 负责实现对应网站折扣界面所需的方法 |
| WebDiscountServiceImpl | 负责处理有关网站折扣的功能 |

#### 2.5.2.2 局部模块的接口规范

表2.5.2.2 -1 WebDiscountService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| WebDiscountService.findAll | 语法 | public List<WebDiscountVO> findAll() | |
| 前置条件 | 无，需要网站所有促销策略信息 | |
| 后置条件 | 返回网站所有促销策略信息 | |
| WebDiscountService.add | 语法 | public ResultMessage add (WebDiscountVO) | |
| 前置条件 | 已知网站促销策略信息，需要将其添加到数据库 | |
| 后置条件 | 返回添加的结果信息 | |
| WebDiscountService. update | 语法 | public ResultMessage update(WebDiscountVO) | |
| 前置条件 | 已知更新后的网站促销策略信息，需要将数据库中的信息进行更新 | |
| 后置条件 | 返回更新的结果信息 | |
| WebDiscount.delete | 语法 | public ResultMessage delete(int id) | |
| 前置条件 | 已知网站促销策略编号，需要将其从数据库中删除 | |
| 后置条件 | 返回删除的结果信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| WebDiscountDAO. add (WebDiscountPO) | | | 添加单一持久化序列 |
| WebDiscountDAO.update | | | 更新单一持久化序列 |
| WebDiscount. delete | | | 删除单一持久化序列 |
| WebDiscount.findAll | | | 查找单一持久化列表 |

#### 2.5.2.3 局部模块的行为

图2.5.2.3-1 WebDiscountService模块时序图

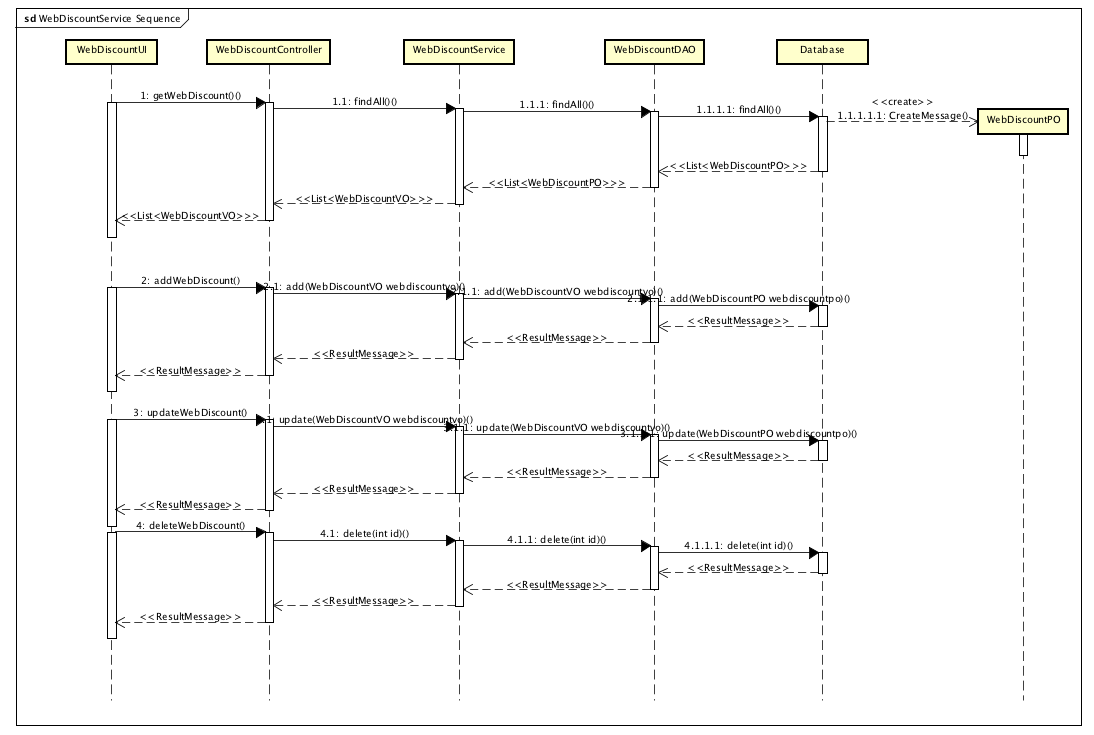
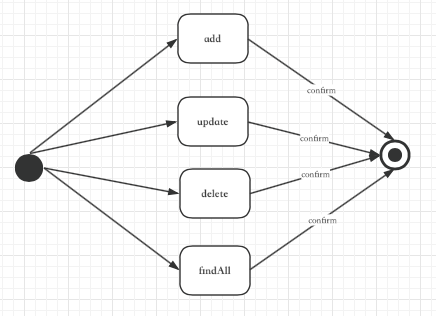


图2.5.2.3-1 表明了酒店预定系统中，当网站营销人员查看、添加、更新、删除网站折扣时，网站折扣业务逻辑处理的相关对象之间的协作。

图2.5.2.3-2 WebDiscountService模块状态图



#### 2.5.2 EnterpriseService

#### 2.5.2.1 局部模块的职责

（1）模块概述

EnterpriseService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。EnterpriseService模块的职责及接口参见软件系统结构描述文档表5.3.2-7

（2）整体结构

EnterpriseService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。EnterpriseService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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EnterpriseService, EnterpriseDAO 两个接口。EnterprisePO是做为管理信息的持久化对象被添加到设计模型中的。

图2.5.3.1-1 EnterpriseService模块设计

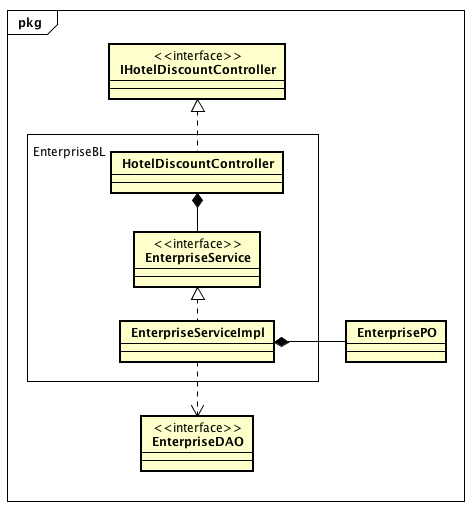


表2.5.3.1-2 EnterpriseService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| HotelDiscountController | 负责实现对应酒店促销策略界面，与合作企业有关的方法 |
| EnterpriseServiceImpl | 负责处理有关合作企业的功能 |

#### 2.5.3.2 局部模块的接口规范

表2.5.3.2 -1 EnterpriseService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| EnterpriseService.getAllEnterprises | 语法 | public List<EnterpriseVO> getAllEnterprises() | |
| 前置条件 | 无，需要所有的企业信息 | |
| 后置条件 | 无 | |
| EnterpriseService.add | 语法 | public ResultMessage add(EnterpriseVO ); | |
| 前置条件 | 已知企业信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中保存新添加的酒店信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| EnterpriseDAO.findAll(); | | | 返回企业信息列表 |
| EnterpriseDAO.add(EnterprisePO ) | | | 新增单一持久化序列 |

#### 2.5.3.3 局部模块的行为

图2.5.3.3-1 EnterpriseService模块时序图

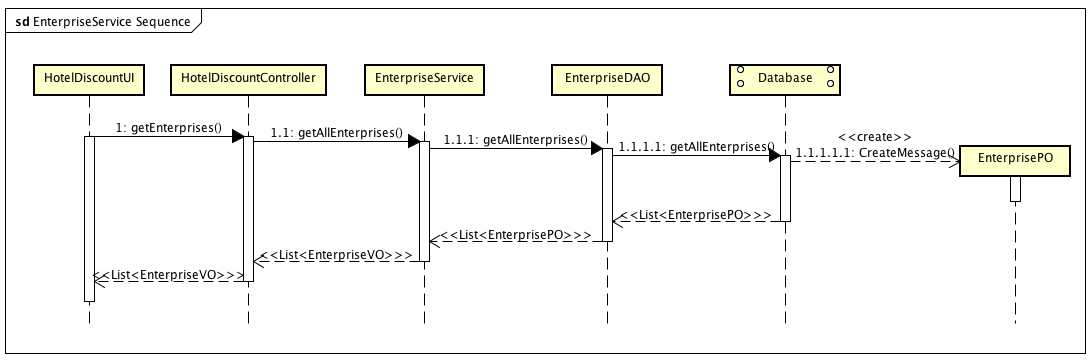
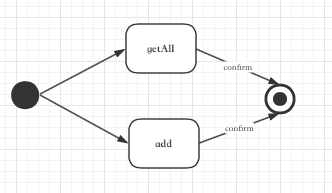


图2.5.3.3-1 表明了酒店预定系统中，当酒店工作人员查看合作企业列表时，合作企业业务逻辑处理的相关对象之间的协作。

图2.5.3.3-2 EnterpriseService模块状态图



### 2.6 CreditRecord模块的静态结构与动态行为

#### 2.6.1 局部模块的职责

（1）模块概述

CreditRecord模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。CreditRecord模块的职责及接口参见软件系统结构描述文档表5.3.2-6

（2）整体结构

CreditRecord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。CreditRecord模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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CreditRecordService, CreditRecordDAO 两个接口。CreditRecordPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.6.1-1 CreditRecord模块设计

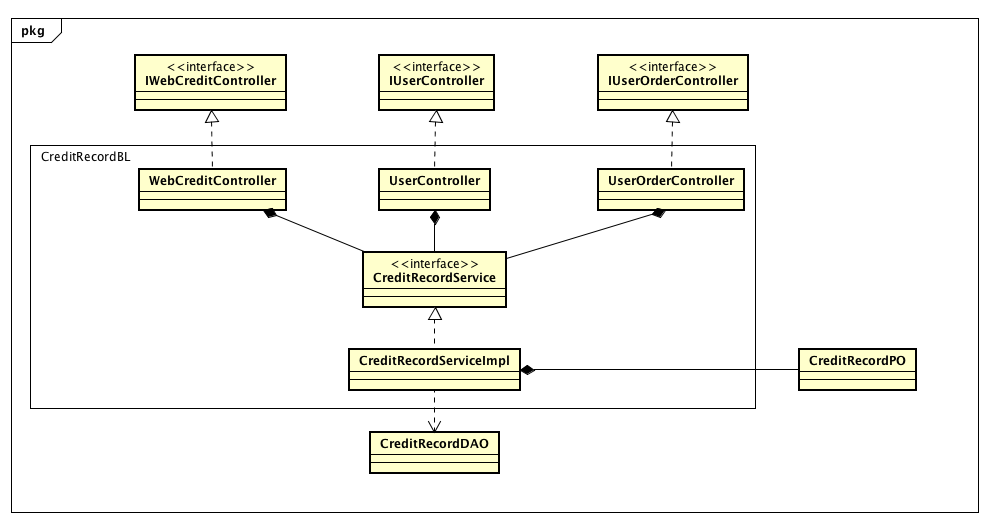


表2.6.1-2 CreditRecord模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| WebCreditController | 负责实现对应信用充值界面所需的方法 |
| UserController | 负责实现对应个人信息界面所需的方法 |
| UserOrderController | 负责实现对应订单信息界面所需的方法 |
| CreditRecordServiceImpl | 负责处理有关信用记录的功能 |

#### 2.6.2 局部模块的接口规范

表2.6.2-1CreditRecordService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| CreditRecordService. findByUsername | 语法 | public List<CreditRecordVO> findByUsername(String username) | |
| 前置条件 | 已知用户名，需要该用户的所有信用记录 | |
| 后置条件 | 返回该用户的所有信用记录 | |
| CreditRecordService.add | 语法 | public ResultMessage add(CreditRecordVO) | |
| 前置条件 | 已知信用记录的信息，需要将其添加到数据库 | |
| 后置条件 | 返回添加的结果信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| CreditRecordDAO. findByUsername(String username) | | | 查找单一持久化列表 |
| CreditRecordDAO. add(CreditRecordPO) | | | 增加单一持久化序列 |
| UserService. findByUsername(String username) | | | 查找用户的原始信用值 |
| UserService. update(UserVO) | | | 更新用户的信用值信息 |

#### 2.6.3 局部模块的行为

图2.6.3-1 CreditRecord模块时序图1

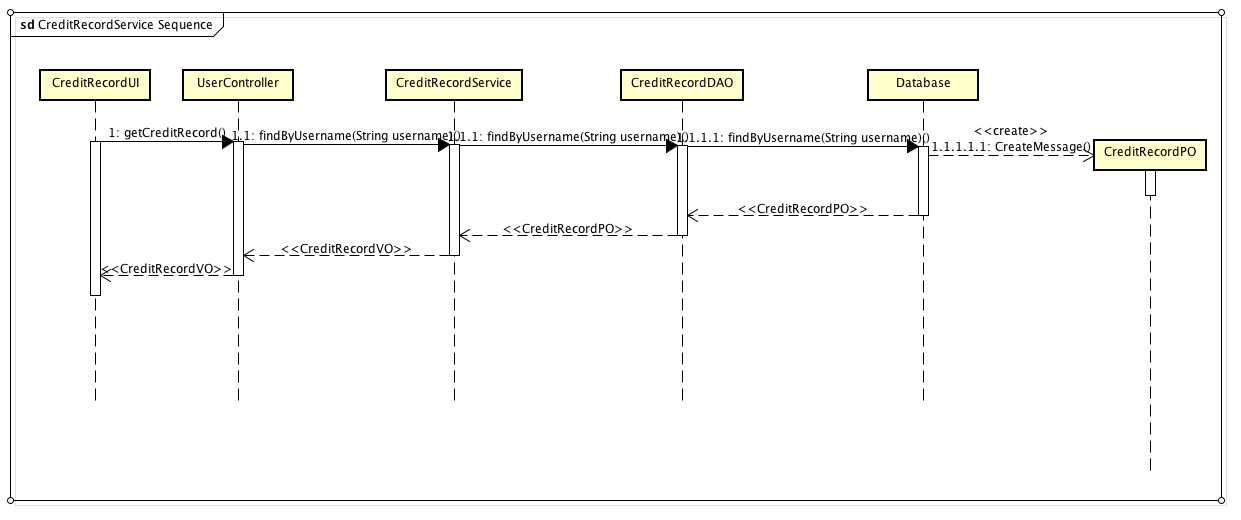


图2.6.3-1 表明了酒店预定系统中，当用户查询信用记录时，信用记录业务逻辑处理的相关对象之间的协作。

图2.6.3-2 CreditRecord模块时序图2

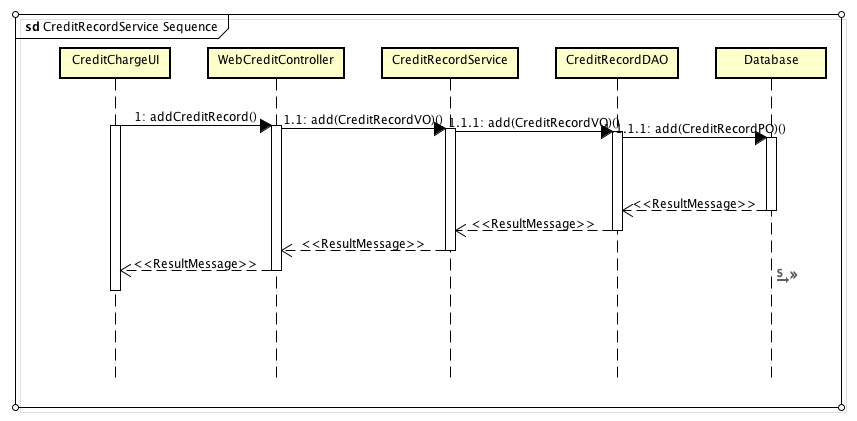
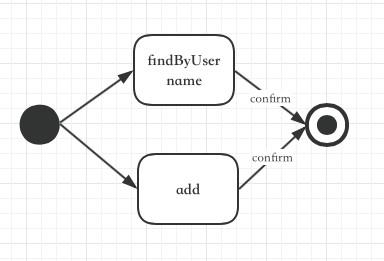


图2.6.3-2 表明了酒店预定系统中，当网站营销人员为用户信用充值时，信用记录业务逻辑处理的相关对象之间的协作。

图2.6.3-3 CreditRecord模块状态图



**2.7 OfflineRecord模块的静态结构与动态行为**

**2.7.1 OfflineRecordService**

**2.7.1.1 局部模块的职责**

（1）模块概述

OfflineRecordService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。OfflineRecordService模块的职责及接口参见软件系统结构描述文档表5.3.2-8

（2）整体结构

OfflineRecordService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。OfflineRecordService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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OfflineRecordService, OfflineRecordDAO两个接口。OfflineRecordPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.7.1.1-1 OfflineRecordService模块设计

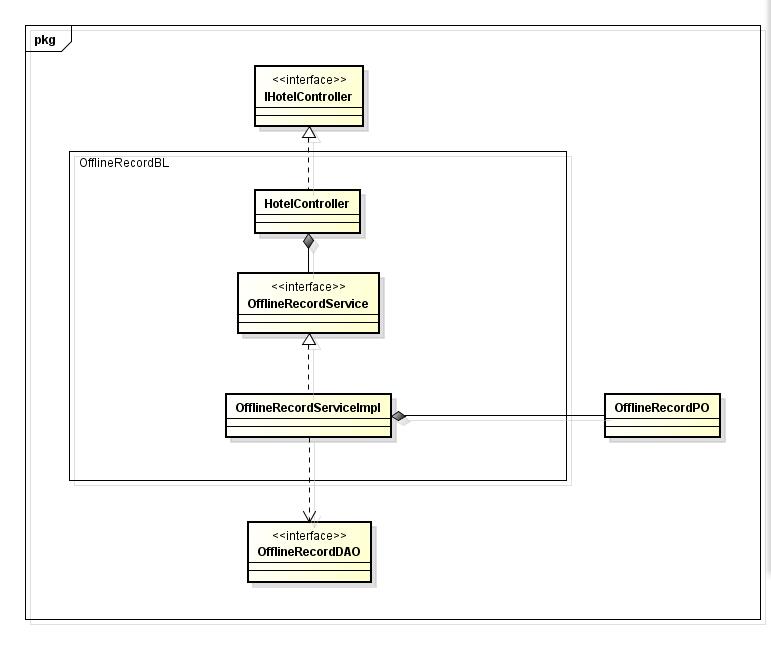


表2.3.1.1-2 OfflineRecordService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| HotelController | 负责实现对应酒店线下记录界面所需的方法 |
| OfflineRecordServiceImpl | 负责处理有关线下记录的功能 |

**2.7.1.2 局部模块的接口规范**

表2.7.1.2 -1 OfflineRecordService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| OfflineRecordService. findByID | 语法 | public OfflineRecordVO findByID(int id) | |
| 前置条件 | 已知线下记录编号，需要线下记录的信息 | |
| 后置条件 | 返回该编号对应的线下记录信息 | |
| OfflineRecordService. add | 语法 | public ResultMessage add(OfflineRecordVO) | |
| 前置条件 | 已知线下记录信息，需要将其添加到数据库 | |
| 后置条件 | 返回添加的结果信息 | |
| OfflineRecordService.checkin | 语法 | public ResultMessage checkin(OfflineRecordVO) | |
| 前置条件 | 已知线下记录信息，需要将该记录状态改为已入住 | |
| 后置条件 | 在数据库中更新线下记录信息 | |
| OffLineRecordService.checkout | 语法 | public ResultMessage checkout(OfflineRecordVO) | |
| 前置条件 | 已知线下记录信息，需要将该记录状态改为已退房 | |
| 后置条件 | 在数据库中更新线下记录信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OfflineRecordDAO. findByID(int id) | | | 查找单一持久化序列 |
| OfflineRecordDAO.add(OfflineRecordPO) | | | 添加单一持久化序列 |
| OfflineRecordDAO.update(OfflineRecordPO) | | | 更新单一持久化序列 |

**2.7.1.3 局部模块的行为**

图2.7.1.3-1 OfflineRecordService模块时序图1

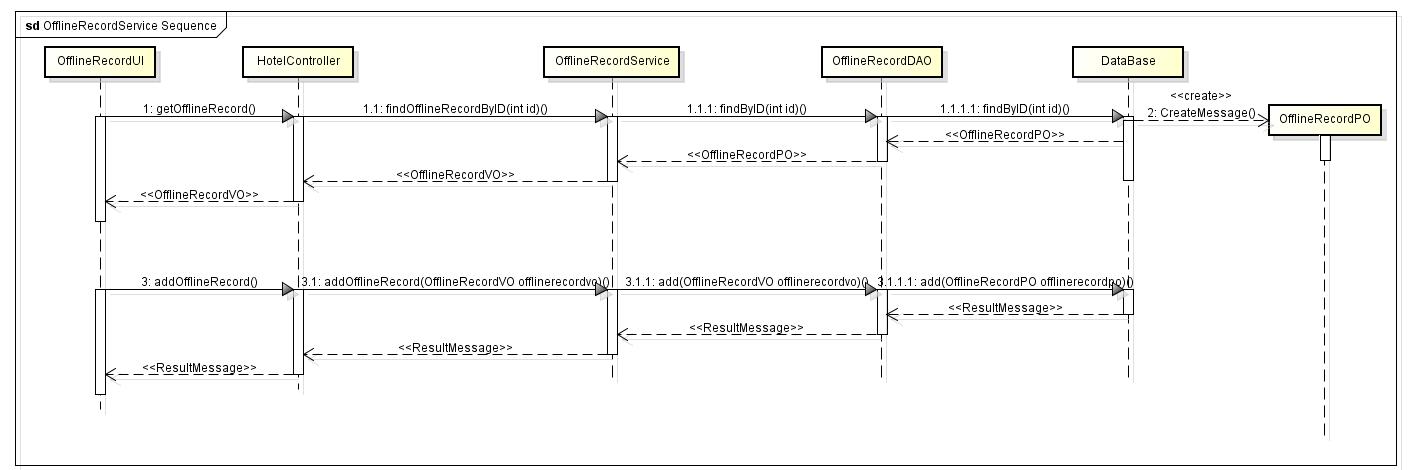


图2.7.1.3-1 表明了酒店预订系统中，当酒店工作人员查询和添加线下记录时，线下记录业务逻辑处理的相关对象之间的协作。

图2.7.1.3-2 OfflineRecordService模块时序图2

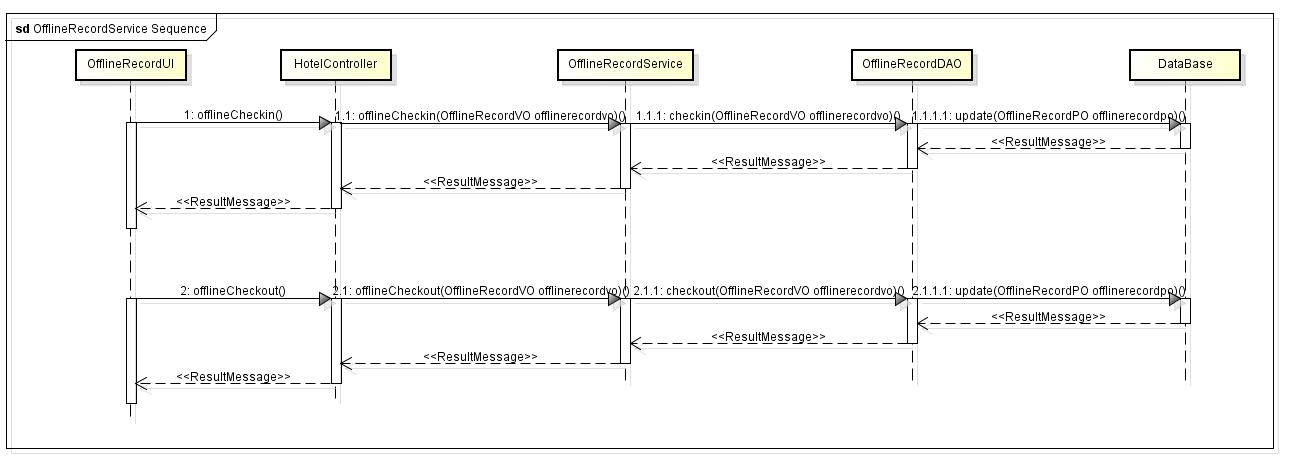


图2.7.1.3-2 表明了酒店预订系统中，当酒店工作人员修改线下记录时，线下记录业务逻辑处理的相关对象之间的协作。

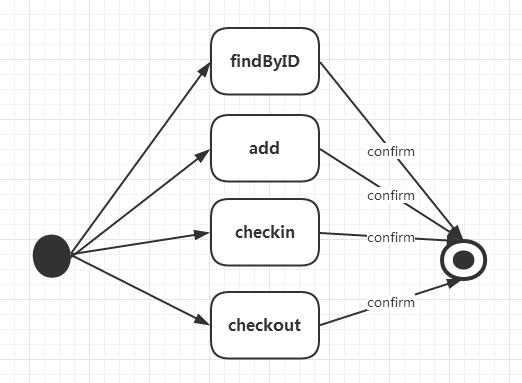


图2.7.1.3-3 OfflineRecordService模块状态图

**2.8 Staff模块的静态结构与动态行为**

**2.8.1 StaffService**

**2.8.1.1 局部模块的职责**

（1）模块概述

StaffService模块承担的需求参见需求规格说明文档功能需求及相关的非功能需求。StaffService模块的职责及接口参见软件系统结构描述文档表5.3.2-5

（2）整体结构

StaffService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。StaffService模块的职责及接口参见软件体系结构描述文档。根据体系结构的设计，采用分层风格，将系统分为展示层，业务逻辑层，数据层。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StaffService, StaffDAO两个接口。StaffPO是做为管理信息的持久化对象被添加到设计模型中的。

图2.8.1.1-1 StaffService模块设计

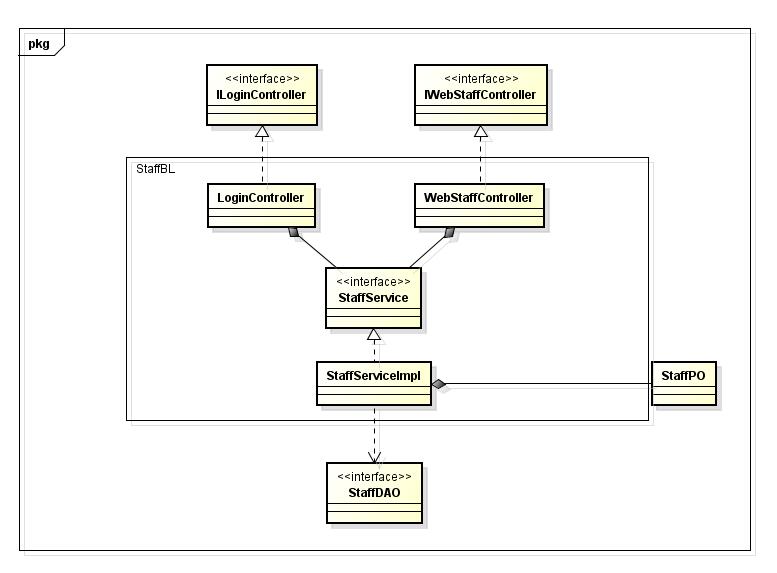


表2.8.1.1-2 StaffService模块各个类的职责

|  |  |
| --- | --- |
| 类 | 职责 |
| LoginController | 负责实现对应登录注册界面所需的方法 |
| WebStaffController | 负责实现对应酒店工作人员、网站营销人员信息管理界面 |
| StaffServiceImpl | 负责处理有关酒店工作人员、网站管理人员管理用户、网站营销人员的功能 |

**2.7.1.2 局部模块的接口规范**

表2.8.1.2 -1 StaffService的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| StaffService.login | 语法 | public StaffVO login(String username,String password) | |
| 前置条件 | 已知用户名和密码，需要进行登录 | |
| 后置条件 | 无 | |
| StaffService.update | 语法 | public ResultMessage update(StaffVO) | |
| 前置条件 | 已知更新后的职员信息，需要更新数据库中的信息 | |
| 后置条件 | 在数据库中更新职员信息 | |
| StaffService.add | 语法 | public ResultMessage add(StaffVO) | |
| 前置条件 | 已知职员信息，需要将其添加到数据库中 | |
| 后置条件 | 在数据库中保存新添加的职员信息 | |
| StaffService.findByUsername | 语法 | public StaffVO findByUsername(String username) | |
| 前置条件 | 已知用户名，需要返回对应的职员信息 | |
| 后置条件 | 无 | |
| StaffService.findByHotelName | 语法 | public StaffVO findByHotelName(String hotelName) | |
| 前置条件 | 已知酒店名称，需要返回对应的职员信息（如果是酒店工作人员） | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| StaffDAO.update(StaffPO) | | | 更新单一持久化序列 |
| StaffDAO. add(StaffPO) | | | 添加单一持久化序列 |
| StaffDAO. findByUsername(String username) | | | 查找单一持久化序列 |
| StaffDAO.findByHotelName(String hotelName); | | | 查找单一持久化序列 |

**2.8.1.3 局部模块的行为**

图2.8.1.3-1 StaffService模块时序图1

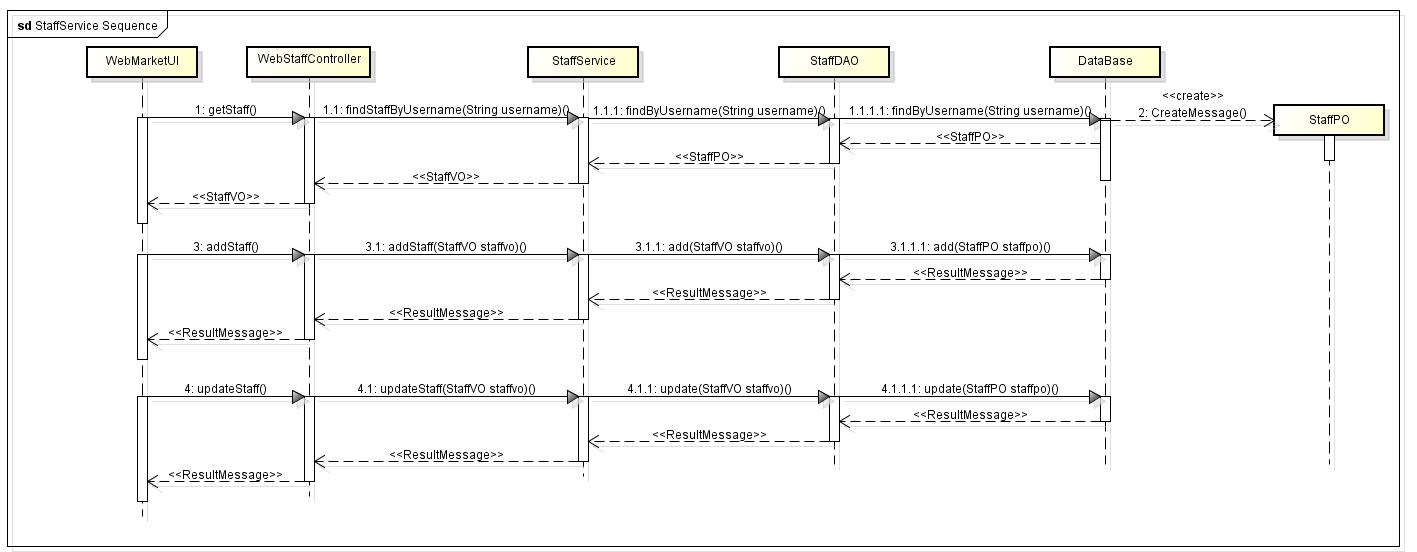


图2.8.1.3-1 表明了酒店预订系统中，当网站管理人员查询、修改和增加网站营销人员信息时，职员业务逻辑处理的相关对象之间的协作。

图2.8.1.3-2 StaffService模块时序图2

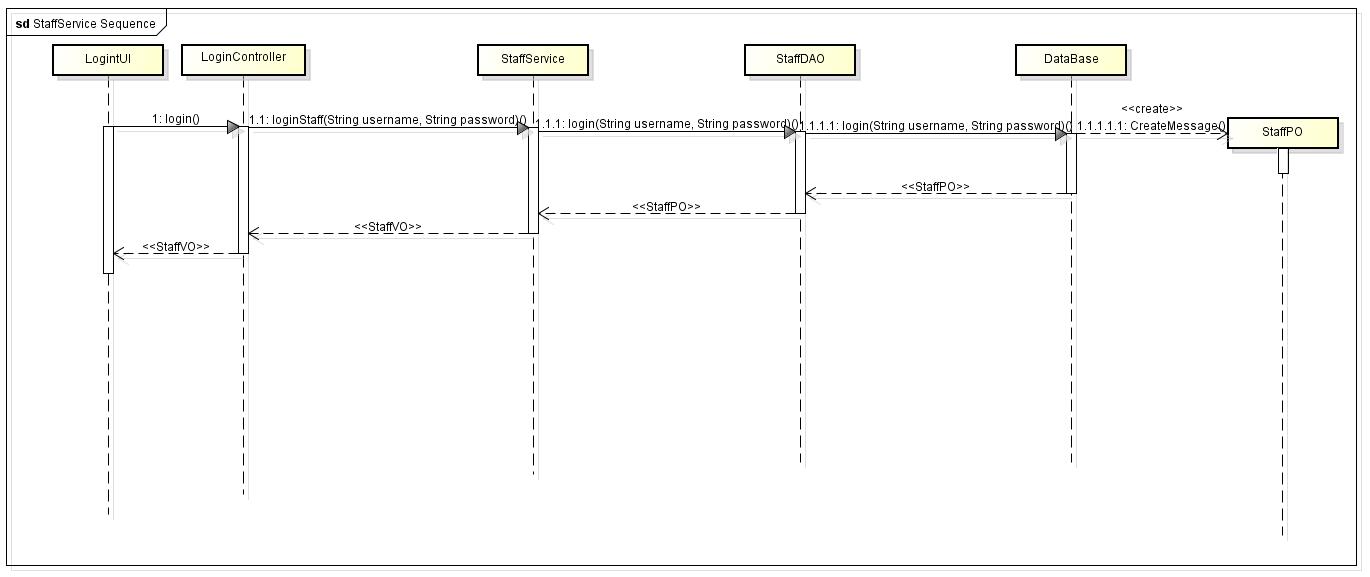


图2.8.1.3-2 表明了酒店预订系统中，当职员登录时，职员业务逻辑处理的相关对象之间的协作。

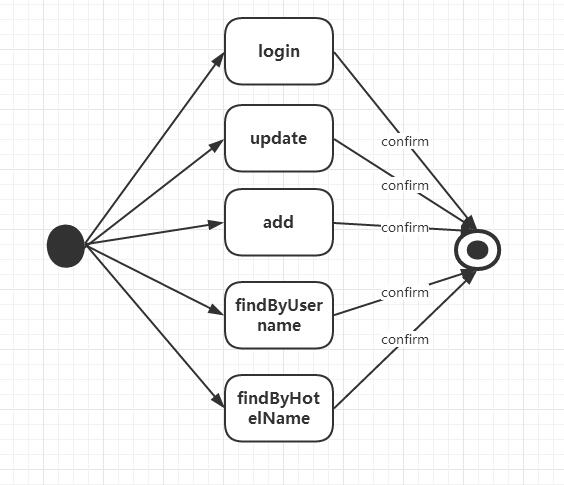


图2.8.1.3-3 StaffService模块状态图

## 三、中层设计之控制器设计

下面来解释控制器Controller引入的原因：

为了避免按照用例分解导致的逻辑重复冗余问题，按照业务逻辑进行模块分解，这就导致了逻辑层与展示层并非是一一对应的，往往单一展示层模块会依赖于多个逻辑层模块。依据最小接口原则，且为了保证展示层能方便地调用逻辑层，整合展示层和逻辑层之间复杂的依赖关系，引入了Controller这一概念。展示层中每一界面与Controller一一对应，而Controller则持有Service，包含了对应界面所依赖的最小接口。为了保证接口隔离，设置Controller的接口IController，展示层仅依赖于IController接口而非Controller实现。

### 3.1 LoginUI包所对应的Controller

#### 3.1.1 LoginController

图3.1.1-1 LoginController 类图

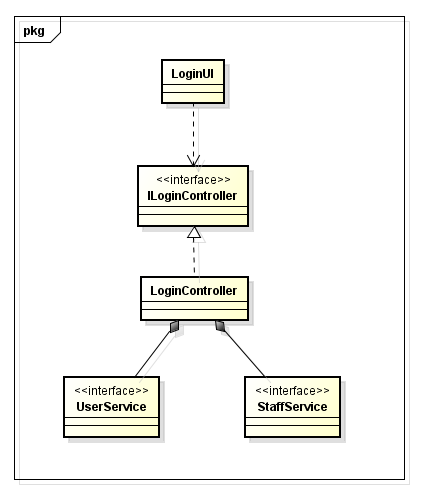


表3.1.1 -2 LoginController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| LoginController. register | 语法 | public ResultMessage register(UserVO uservo) | |
| 前置条件 | 已知用户信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中保存新增的用户信息 | |
| LoginController.loginUser | 语法 | public UserVO loginUser(String username, String password) | |
| 前置条件 | 已知用户名和密码，需要验证该用户的身份 | |
| 后置条件 | 无 | |
| LoginController.loginStaff | 语法 | public StaffVO loginStaff(String username, String password) | |
| 前置条件 | 已知用户名和密码，需要验证该职员的身份 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserService.login(String username,String  password) | | | 查找单一持久化序列 |
| UserService.register(UserVO) | | | 添加单一持久化序列 |
| StaffService.login(String username,String  password) | | | 查找单一持久化序列 |

### 3.2 UserUI包所对应的Controller

#### 3.2.1 UserController

图3.2.1-1 UserController 类图

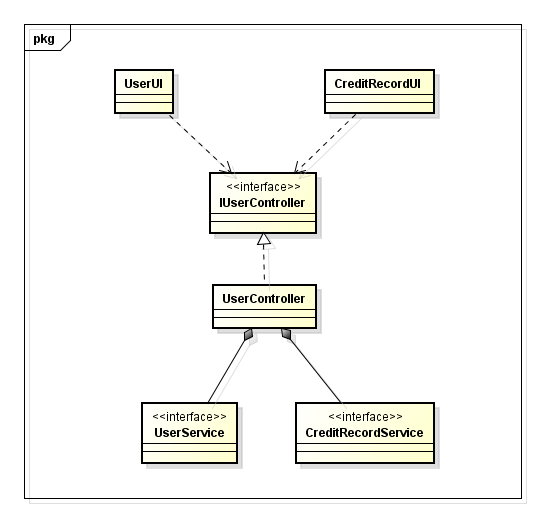


表3.2.1 -2 UserController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserController.updateUser | 语法 | public ResultMessage updateUser(UserVO uservo) | |
| 前置条件 | 已知更新后的信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中更新用户信息 | |
| UserController.findUserByUsername | 语法 | public UserVO  findUserByUsername(String username) | |
| 前置条件 | 已知用户名，需要返回该用户的详细信息 | |
| 后置条件 | 无 | |
| UserController.findCreditRecordByUsername | 语法 | public List<CreditRecordVO>  findCreditRecordByUsername(String username) | |
| 前置条件 | 已知用户名，需要该用户所有的信用记录 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserService.update(UserVO) | | | 更新单一持久化序列 |
| UserService.findByUsername(String username) | | | 查找单一持久化序列 |
| CreditRecordService.findByUsername(String username) | | | 查找单一持久化列表 |

#### 3.2.2 UserOrderController

图3.2.2-1 UserOrderController 类图

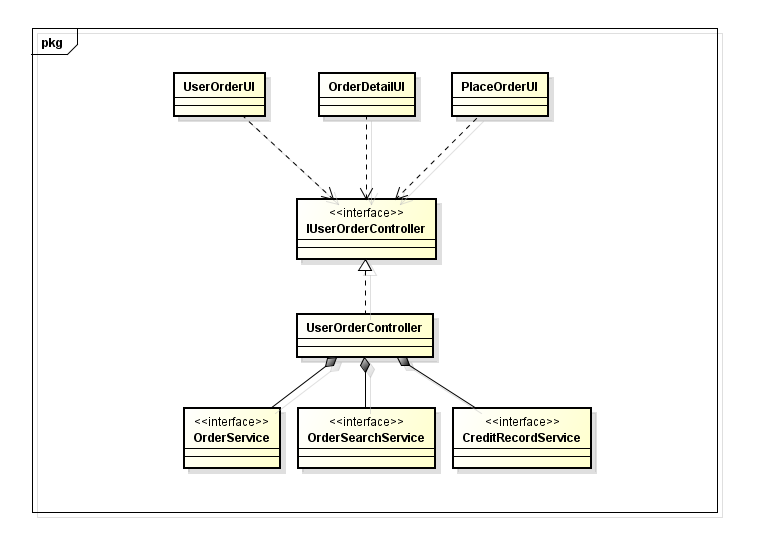


表3.2.2 -2 UserOrderController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserOrderController.findOrdersByUsernameAndType | 语法 | public List<OrderVO>  findOrdersByUsernameAndType(String username, OrderStatus type) | |
| 前置条件 | 已知用户名和订单类型，需要所有满足条件的订单 | |
| 后置条件 | 无 | |
| UserOrderController. placeOrder | 语法 | public OrderVO placeOrder(OrderVO  ordervo) | |
| 前置条件 | 已知订单信息，需要对订单进行处理，为下单做准备 | |
| 后置条件 | 无 | |
| UserOrderController addOrder | 语法 | public ResultMessage addOrder(OrderVO ordervo) | |
| 前置条件 | 已知订单信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中保存新添加的订单 | |
| UserOrderController.remark | 语法 | public ResultMessage remark(OrderVO  ordervo) | |
| 前置条件 | 已知订单信息，需要添加其评论信息 | |
| 后置条件 | 在数据库中保存新的订单信息 | |
| UserOrderController.revoke | 语法 | public ResultMessage revoke(OrderVO ordervo) | |
| 前置条件 | 已知订单信息，需要将订单状态改为已撤销 | |
| 后置条件 | 在数据库中保存新的订单信息 | |
| UserOrderController.addCreditRecord | 语法 | public ResultMessage  addCreditRecord(CreditRecordVO creditrecordvo); | |
| 前置条件 | 已知信用记录信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中保存新添加的信用记录信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderService. placeOrder(OrderVO ordervo) | | | 处理订单信息 |
| OrderService. add(OrderVO ordervo) | | | 添加单一持久化序列 |
| OrderService. revoke(OrderVO ordervo) | | | 撤销订单 |
| OrderService. remark(OrderVO ordervo) | | | 评价订单 |
| OrderSearchService. findByUsernameAndType(String username,  OrderStatus type) | | | 查找单一持久化列表 |
| CreditRecordService. add(CreditRecordVO  creditrecordvo) | | | 添加单一持久化序列 |

#### 3.2.3 UserHotelController

图3.2.3-1 UserHotelController 类图

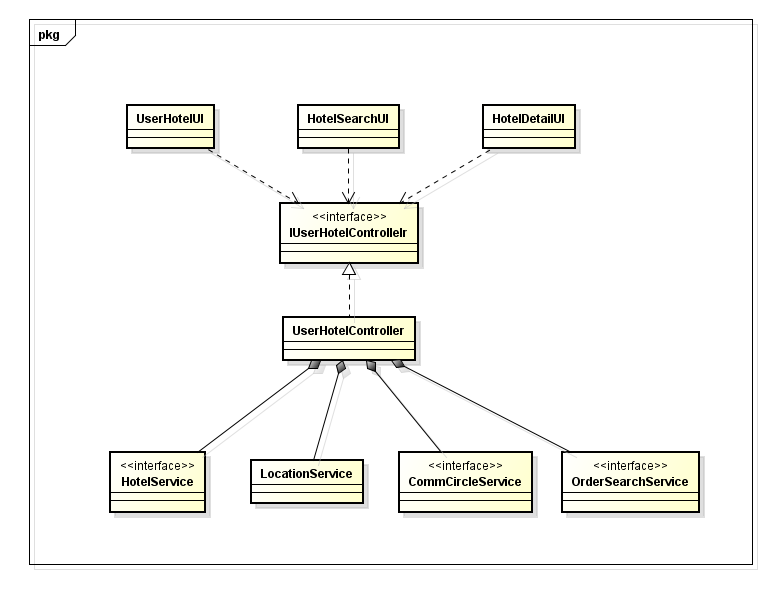


表3.2.3 -2 UserHotelController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserHotelController.findOrderByHotelAndUsername | 语法 | public List<OrderVO>  findOrderByHotelAndUsername(int hotelID, String username) | |
| 前置条件 | 已知酒店编号和用户名，需要返回满足条件的订单列表 | |
| 后置条件 | 无 | |
| UserHotelController. getRoomDetail | 语法 | public List<RoomVO> getRoomDetail(int  hotelID) | |
| 前置条件 | 已知酒店编号，需要返回满足条件的房间列表 | |
| 后置条件 | 无 | |
| UserHotelController. findOrderedHotelAndOrder | 语法 | public Map<HotelVO, List<OrderVO>> findOrderedHotelAndOrder(String username) | |
| 前置条件 | 已知用户名，需要找到该用户曾预定的酒店和订单信息 | |
| 后置条件 | 无 | |
| UserHotelController.findHotels | 语法 | public Map<HotelVO, List<RoomVO>> findHotels(int loc, int circle, Date begin, Date end) | |
| 前置条件 | 已知酒店、商圈和时间，需要返回满足条件的酒店和房间列表 | |
| 后置条件 | 无 | |
| UserHotelController.filterHotels | 语法 | public Map<HotelVO, List<RoomVO>> filterHotels(List<FilterCondition> conditions) | |
| 前置条件 | 已知过滤条件列表，需要返回过滤后的酒店房间列表 | |
| 后置条件 | 无 | |
| UserHotelController. orderHotels | 语法 | public Map<HotelVO, List<RoomVO>> orderHotels(OrderRule rule, boolean  isDecrease) | |
| 前置条件 | 已知排序规则和升降序，需要排序后的房间酒店列表 | |
| 后置条件 | 无 | |
| UserHotelController. findAllLocations | 语法 | public List<LocationVO>  findAllLocations() | |
| 前置条件 | 需要所有城市列表 | |
| 后置条件 | 无 | |
| UserHotelController. findCircleByLoc | 语法 | public List<CommercialCircleVO>  findCircleByLoc(int locID) | |
| 前置条件 | 已知城市编号，需要当前城市的所有商圈 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelSearchService. findByHotelAndUsername(int hotelID, String username) | | | 查找单一持久化列表 |
| HotelService. getRoomDetail(int hotelID) | | | 查找单一持久化列表 |
| HotelService.findOrderedHotelAndOrder(String username) | | | 查找单一持久化列表 |
| HotelService. find(int loc, int circle, Date begin, Date end) | | | 查找单一持久化列表 |
| HotelService. filter(List<FilterCondition>  conditions) | | | 过滤订单 |
| HotelService. order(OrderRule rule, boolean  isDecrease) | | | 排序订单 |
| LocationService. findAll() | | | 查找单一持久化列表 |
| CommCircleService. findByLoc(int locID) | | | 查找单一持久化列表 |

### 3.3 HotelUI包所对应的Controller

#### 3.3.1 HotelController

图3.3.1-1 HotelController 类图

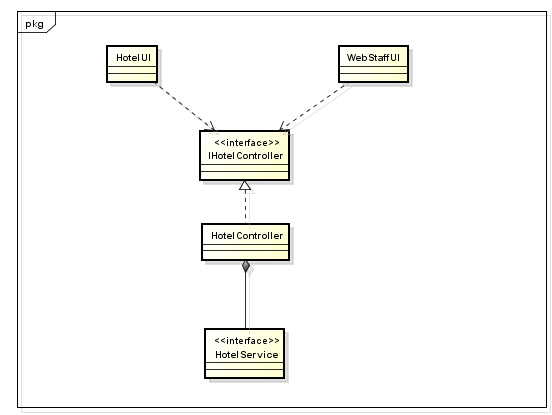


表3.3.1 -2 HotelController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelController.findHotelByID | 语法 | public HotelVO. findHotelByID(int HotelID) | |
| 前置条件 | 已知酒店ID，需要返回该酒店的详细信息 | |
| 后置条件 | 无 | |
| HotelController.updateHotel | 语法 | public ResultMessage updateHotel(Hotel VO hotelvo) | |
| 前置条件 | 已知更新后的信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中更新酒店信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelService.update(HotelVO) | | | 更新单一持久化序列 |
| HotelService.findByID(int HotelID) | | | 查找单一持久化序列 |

#### 3.3.2 HotelOrderController

图3.3.2-1 HotelOrderController 类图

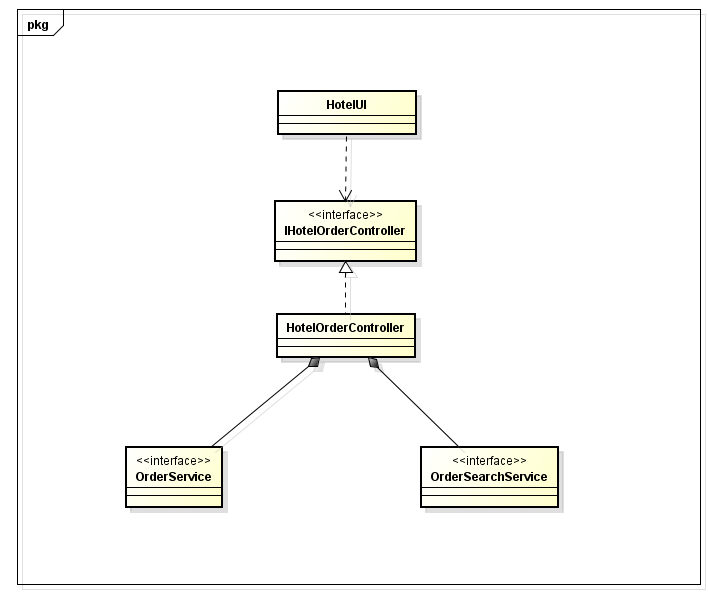


表3.3.2 -2 HotelOrderController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelOrderController.findOrderByHotelAndType | 语法 | Public List<OrderVO> findOrderByHotel-AndType(int HotelID,OrderStatus type) | |
| 前置条件 | 已知酒店ID和订单状态，需要返回该酒店特定状态的所有订单的详细信息 | |
| 后置条件 | 无 | |
| HotelOrderController.findOrderByHotelAndUsername | 语法 | Public List<OrderVO> findOrderByHotel-AndUsername(String username,int HotelID) | |
| 前置条件 | 已知酒店ID和订单所有者用户名，需要返回该用户在此酒店的所有订单的详细信息 | |
| 后置条件 | 无 | |
| HotelOrderController.findOrderByID | 语法 | Public OrderVO findOrderByID(int id) | |
| 前置条件 | 已知订单ID，需要返回该订单的详细信息 | |
| 后置条件 | 无 | |
| HotelOrderController.checkin | 语法 | Public ResultMessage checkout(OrderVO ordervo) | |
| 前置条件 | 已知订单信息，需要进行入住登记 | |
| 后置条件 | 在数据库中更新订单信息 | |
| HotelOrderController.revoke | 语法 | Public ResultMessage revoke(OrderVO or-dervo) | |
| 前置条件 | 已知订单信息，需要进行订单撤销 | |
| 后置条件 | 在数据库中删除订单信息 | |
| HotelOrderController.delayCheckin | 语法 | Public ResultMessage delayCheckin(Order-VO ordervo) | |
| 前置条件 | 已知订单信息，需要进行订单延迟入住 | |
| 后置条件 | 在数据库中更新订单信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderService.checkout(OrderVO ordervo) | | | 更新单一持久化序列 |
| OrderService.checkin(OrderVO ordervo) | | | 更新单一持久化序列 |
| OrderService.revoke (OrderVO ordervo) | | | 更新单一持久化序列 |
| OrderService.delayCheckin(OrderVO ordervo) | | | 更新单一持久化序列 |
| OrderSearchService.findByID(int OrderID) | | | 查找单一持久化序列 |
| OrderSearchService.findByHotelAndType(int HotelID,OrderStatus type) | | | 查找单一持久化序列 |
| OrderSearchService. findByHotelAndUsername (String username,int HotelID) | | | 查找单一持久化序列 |

#### 3.3.3 HotelDiscountController

图3.3.3-1 HotelDiscountController 类图

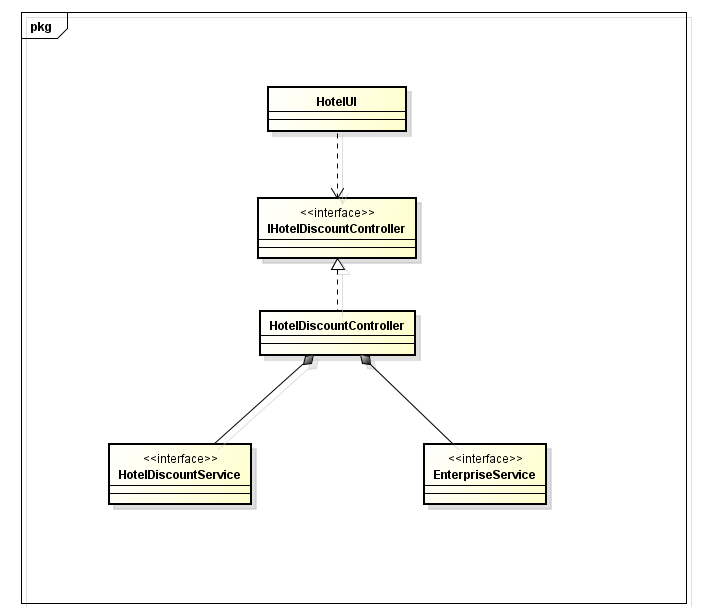


表3.3.3 -2 HotelDiscountController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| HotelDiscountController.findAllByHotelID | 语法 | Public List<HotelDiscountVO> findAllBy-HotelID(int HotelID) | |
| 前置条件 | 已知酒店ID，需要返回该酒店所有优惠策略的详细信息 | |
| 后置条件 | 无 | |
| HotelDiscountController.findNotAddedType | 语法 | Public List<HotelDiscountType> findNotA-ddedType (int HotelID) | |
| 前置条件 | 已知酒店ID，需要返回该酒店尚未制定的优惠策略类型 | |
| 后置条件 | 无 | |
| HotelDiscountController.getAllEnterprises | 语法 | Public List<EnterpriseVO> getAllEnterprises () | |
| 前置条件 | 返回所有企业信息 | |
| 后置条件 | 无 | |
| HotelDiscountController.delete | 语法 | Public ResultMessage delete(int id) | |
| 前置条件 | 已知优惠策略id，需要删除该优惠策略 | |
| 后置条件 | 在数据库中更新优惠策略信息 | |
| HotelDiscountController.add | 语法 | Public ResultMessage add(HotelDiscount-VO hoteldiscountvo) | |
| 前置条件 | 已知优惠策略信息，需要进行优惠策略的添加 | |
| 后置条件 | 在数据库中添加优惠策略信息 | |
| HotelDiscountController.update | 语法 | Public ResultMessage update(HotelDisc-ountVO hoteldiscountvo) | |
| 前置条件 | 已知优惠策略信息，需要修改该优惠策略 | |
| 后置条件 | 在数据库中更新优惠策略信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelDiscountService. add(HotelDiscountVO hot-eldiscountvo) | | | 更新单一持久化序列 |
| HotelDiscountService. update(HotelDiscountVO hoteldiscountvo) | | | 更新单一持久化序列 |
| HotelDiscountService. delete(int id) | | | 更新单一持久化序列 |
| HotelDiscountService. findAllByHotelID(int Hote lID) | | | 查找单一持久化序列 |
| HotelDiscountService. findNotAddedType (int Ho telID) | | | 查找单一持久化序列 |
| EnterpriseService. getAllEnterprises() | | | 查找单一持久化序列 |

### 3.4 WebMarketUI包所对应的Controller

#### 3.4.1 WebDiscountController

图3.4.1-1 WebDiscountController 类图

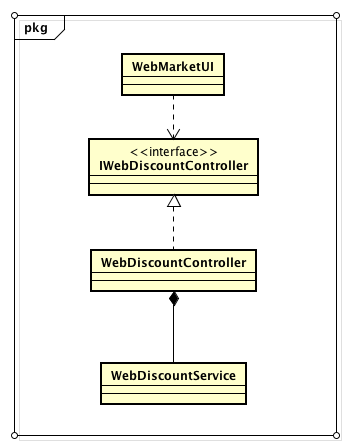


表3.4.1-2 WebDiscountController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| WebDiscountController.findAll | 语法 | Public List<WebDiscountVO> findAll() | |
| 前置条件 | 需要返回所有WebDiscount列表 | |
| 后置条件 | 无 | |
| WebDiscountController.add | 语法 | Public ResultMessage add (WebDiscountVO webdiscountvo) | |
| 前置条件 | 已知网站折扣，需要将其添加到数据库中 | |
| 后置条件 | 在数据库中保存新增的网站折扣信息 | |
| WebDiscountController.update | 语法 | Public ResultMessage update (WebDiscountVO webdiscountvo) | |
| 前置条件 | 已知网站折扣，需要将其在数据库中更新 | |
| 后置条件 | 在数据库中更新网站折扣信息 | |
| WebDiscountController.delete | 语法 | Public ResultMessage delete(int id) | |
| 前置条件 | 已知网站折扣ID，需要将其删除 | |
| 后置条件 | 在数据库中删除此网站折扣 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| WebDiscountService.findAll() | | | 查找单一持久化序列 |
| WebDiscountService.add(WebDiscountVO webdiscountvo) | | | 添加单一持久化序列 |
| WebDiscountService.update(WebDiscountVO webdiscountvo) | | | 更新单一持久化序列 |
| WebDiscountService.delete() | | | 删除单一持久化序列 |

#### 3.4.2 WebOrderController

图3.4.2-1 WebOrderController 类图

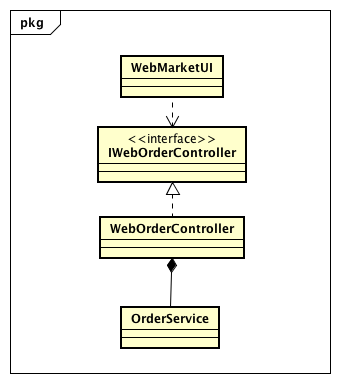


表3.4.2 -2 WebOrderController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| WebOrderController.  findOrderByOrderType | 语法 | Public List<OrderVO> findOrderByOrderType(OrderStatus status) | |
| 前置条件 | 已知订单状态，需要返回符合条件的订单列表 | |
| 后置条件 | 无 | |
| WebOrderController.  findOrderByID | 语法 | Public OrderVO findOrderByID(int ID) | |
| 前置条件 | 已知订单ID,需要返回该订单信息 | |
| 后置条件 | 无 | |
| WebOrderController.  revokeOrder | 语法 | Public ResultMessage revokeOrder(OrderVO ordervo) | |
| 前置条件 | 已知订单记录，需要撤销该订单 | |
| 后置条件 | 在数据库中删除此订单记录 | |
| WebOrderController.  findOrderByUsername | 语法 | Public List<OrderVO> findOrderByUsername(String username) | |
| 前置条件 | 已知username，需要返回对应的订单列表 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| OrderService.revoke(OrderVO ordervo) | | | 删除单一持久化序列 |
| OrderService.findByID(int ID) | | | 查找单一持久化序列 |
| OrderService.findByUsername(String username) | | | 查找单一持久化序列 |
| OrderService.findByOrderType(OrderStatus status) | | | 查找单一持久化序列 |

#### 3.4.3 WebCreditController

图3.4.3-1 WebCreditController 类图

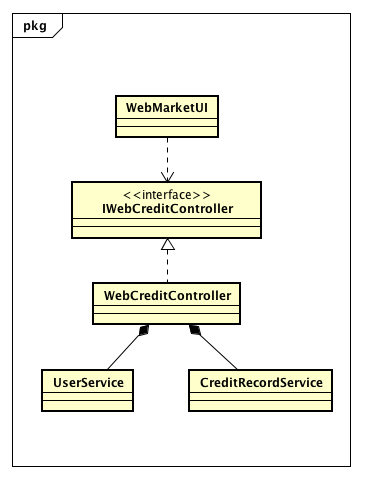


表3.4.3 -2 WebCreditController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| WebCreditController.  findUserByUsername | 语法 | Public userVO findUserByUsername(String username) | |
| 前置条件 | 已知username，需要返回相关用户信息 | |
| 后置条件 | 无 | |
| WebCreditController.  addCreditRecord | 语法 | Public ResultMessage addCreditRecord(CreditRecordVO creditrecordvo) | |
| 前置条件 | 已知信用记录，需要将其添加到数据库 | |
| 后置条件 | 在数据库中保存新增的信用记录 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| CreditRecordService.findByUsername(String username) | | | 查找单一持久化序列 |
| CreditRecordService.add(CreditRecordVO creditrecordvo) | | | 添加单一持久化序列 |
| UserService.findByUsername(String username) | | | 查找单一持久化序列 |

**3.5 WebStaffUI包所对应的Controller**

**3.5.1 WebUserController**

图3.5.1-1 WebUserController 类图

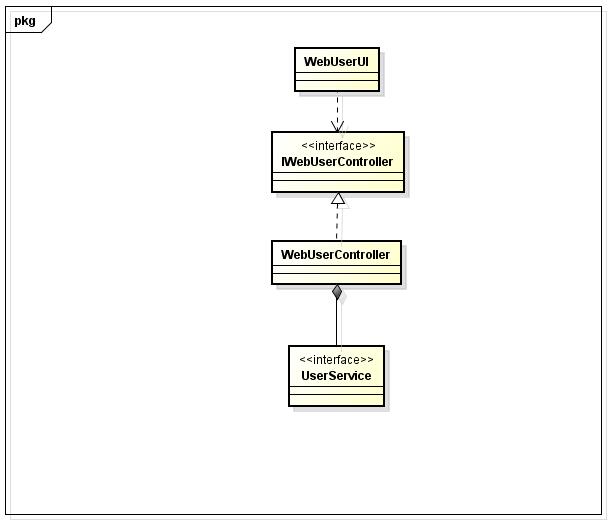


表3.5.1 -2 WebUserController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| WebUserController.findUserByUsername | 语法 | public UserVO  findUserByUsername(String username) | |
| 前置条件 | 已知用户名，需要返回该用户的信息 | |
| 后置条件 | 无 | |
| WebUserController.updateUser | 语法 | public ResultMessage updateUser(UserVO uservo) | |
| 前置条件 | 已知更新后的信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中更新用户信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| UserService.update(UserVO) | | | 更新单一持久化序列 |
| UserService.findByUsername(String username) | | | 查找单一持久化序列 |

**3.5.2 WebStaffController**

图3.5.2-1 WebStaffController 类图

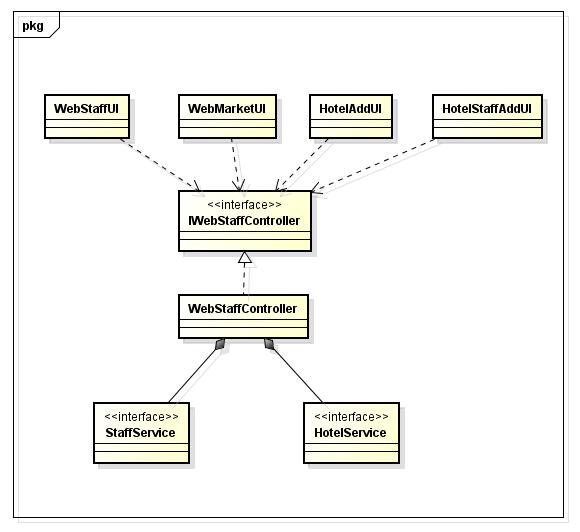


表3.5.2 -2 WebStaffController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| WebStaffController.addHotel | 语法 | Public ResultMessage addHotel(HotelVO hotelvo) | |
| 前置条件 | 已知酒店信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中更新酒店信息 | |
| WebStaffController.addStaff | 语法 | public ResultMessage addStaff(StaffVO staffvo) | |
| 前置条件 | 已知职员信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中更新职员信息 | |
| WebStaffController.updateStaff | 语法 | Public ResultMessage updateStaff(StaffVO staffvo) | |
| 前置条件 | 已知更新后的信息，需要将其添加到数据库 | |
| 后置条件 | 在数据库中更新职员信息 | |
| WebStaffController.findStaffByUsername | 语法 | Public StaffVO findStaffByUsername(String username) | |
| 前置条件 | 已知职员的用户名，需要返回该职员的信息 | |
| 后置条件 | 无 | |
| WebStaffController.findStaffByHotelName | 语法 | Public StaffVO findStaffByHotelName(String hetelName) | |
| 前置条件 | 已知酒店工作人员的酒店名称，需要返回该酒店工作人员的信息 | |
| 后置条件 | 无 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| HotelService.add(HotelVO hotelvo) | | | 添加单一持久化序列 |
| StaffService.add(StaffVO staffvo) | | | 添加单一持久化序列 |
| StaffService.update(StaffVO staffvo) | | | 更新单一持久化序列 |
| StaffService.findByUsername(String username) | | | 查找单一持久化序列 |
| StaffService.findByHotelName(String hotelName) | | | 查找单一持久化序列 |