## 2.1 usesrmainbl模块的静态结构和动态行为

### 2.1.1 usermainbl模块局部模块的职责

UserMainbl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。UserMainbl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是做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2.1.1-1所示.

### 2.1.2 usermainbl模块局部模块的接口规范

UsermainController的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UsermainController.login | 语法 | public UserVO login(long userID, String password); | |
| 前置条件 | Password符合输入规则 | |
| 后置条件 | 查找是否有相应的User，根据输入的密码返回登录验证的结果及相应的用户信息值对象 | |
| UsermainController.logout | 语法 | public ResultMessage logout(long userID); | |
| 前置条件 | 已经登录 | |
| 后置条件 | 返回操作结果，相应用户退出登录状态 | |
| 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 | 保存此客户的用户名和密码 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| Usermain.login | | | 用户登录 |
| Usermain.logout | | | 用户登出 |
| Usermain.signUp | | | 用户注册 |

UsermainBL的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Usermain.login | 语法 | public UserVO login(long userID, String password); | |
| 前置条件 | Password符合输入规则 | |
| 后置条件 | 查找是否有相应的User，根据输入的密码返回登录验证的结果及相应的用户信息值对象 | |
| Usermain.logout | 语法 | public ResultMessage logout(long userID); | |
| 前置条件 | 已经登录 | |
| 后置条件 | 返回操作结果，相应用户退出登录状态 | |
| 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 | 保存此客户的用户名和密码 | |
| 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.insert(UserPO user) | | | 在数据库中插入UserPO对象 |

### 2.1.3 usermainbl模块的行为

## 2.7 manageroombl模块的静态结构和动态行为

### 2.7.1 manageroombl模块局部模块的职责

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.

### 2.7.2 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 |

### 2.7.3 manageroombl模块的行为

## 2.8 strategybl模块的静态结构和动态行为

### 2.8.1 strategybl模块局部模块的职责

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.

### 2.8.2 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 |

### 2.8.3 strategybl模块的行为

## 2.9 managehotelbl模块的静态结构和动态行为

### 2.9.1 managehotelbl模块局部模块的职责

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.

### 2.9.2 managehotelbl模块局部模块的接口规范

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 |

### 2.9.3 managehotelbl模块的行为