酒店预订系统

软件体系结构设计文档

V3.0

南京大学软件学院

刘宇翔 刘伟 刘宗侃 刘铮

2016.10.11

V3.0

变更记录

修改人员	日期	变更原因	版本号
刘铮	2016.10.9	软件体系结构设计初步模型	v1.0
刘铮	2016.10.11	整体性细节修改	v3.0

目录

1. 引言

1.1. 编制目的

本报告详细完成对酒店预订系统的概要设计，达到指导详细设计和开发的目的，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
HRS	酒店预订系统	无

1.3. 参考资料

1) 酒店预订系统用例文档 v3.0

2) 酒店预订系统软件需求规格说明文档 v3.0

2. 产品描述

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

3. 逻辑视角

酒店预订系统中，选择了分层体系结构风格，将系统分为 3 层（展示层、业务逻辑层、数据层）能够很好地示意整个高层抽象。

展示层包含 GUI 页面的实现，业务逻辑层包含业务逻辑处理的实现，

数据层负责数据的持久化和访问。分层体系结构的逻辑视角和逻辑设计方案如图 3-1 和图 3-2 所示。

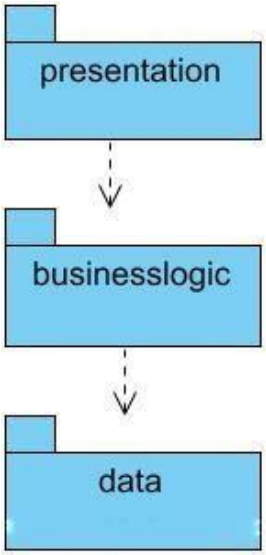

图 3-1 参照体系结构风格的包图表达逻辑视角

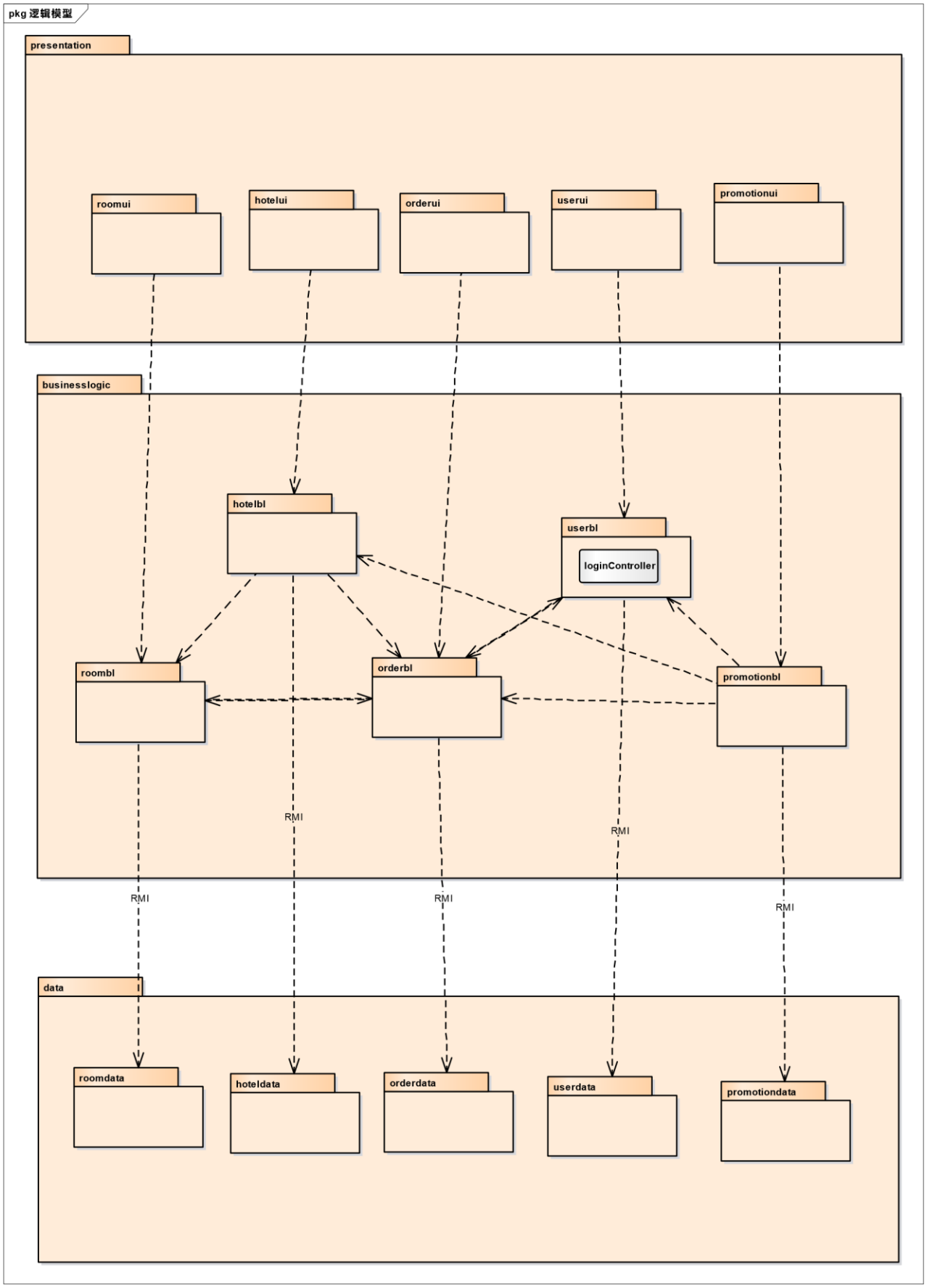
pkg 逻辑模型
presentation
roomui
hotelui
orderui
userui
promotionui
businesslogic
hotelbl
userbl
loginController
roombl
orderbl
promotionbl
RMI
RMI
RMI
RMI
RMI
data
roomdata
hoteldata
orderdata
userdata
promotiondata

图 3-2 软件体系结构逻辑设计方案

4. 组合视角

4.1. 开发包图

酒店预订系统的最终开发包设计如表 4-1-1 所示。

表 4-1-1 酒店预订系统的最终开发包设计

开发物理包	依赖的其他开发包
mainui	roomui,hotelui,orderui,userui,promotionui,vo
roomui	界面类库包, roomblservice,vo
roomblservice	
rooombl	roomblservice,roomdataservice,orderbl,po
roomdataservice	Java RMI,po
roomdata	Java RMI,po,databaseutility
hotelui	界面类库包, hotelblservice,vo
hotelblservice	
hotelbl	hotelblservice,hoteldataservice,,roombl,orderbl,po
hoteldataservice	Java RMI,po
hoteldata	Java RMI,po,databaseutility
orderui	界面类库包, orderblservice,vo
orderblservice	
orderbl	orderblservice,orderdataservice,userbl,promotionbl, po
orderdataservice	Java RMI,po
orderdata	Java RMI,po,databaseutility
userui	界面类库包, userblservice,vo
userblservice	

userbl	userblservice,userdataservice,po
userdataservice	Java RMI,po
userdata	Java RMI,po,databaseutility
promotionui	界面类库包, promotionblservice,vo

promotionblservice	
promotionbl	promotionblservice,promotiondataservice,orderbl,userbl,hotelbl,po
promotiondataservice	Java RMI,po
promotiondata	Java RMI,po,databaseutility
vo	
po	
utility	
界面类库包	
Java RMI	
databaseutility	JDBC

酒店预订系统客户端开发包图如图 4-1-1 所示，服务器端开发包图如图 4-1-2 所示。

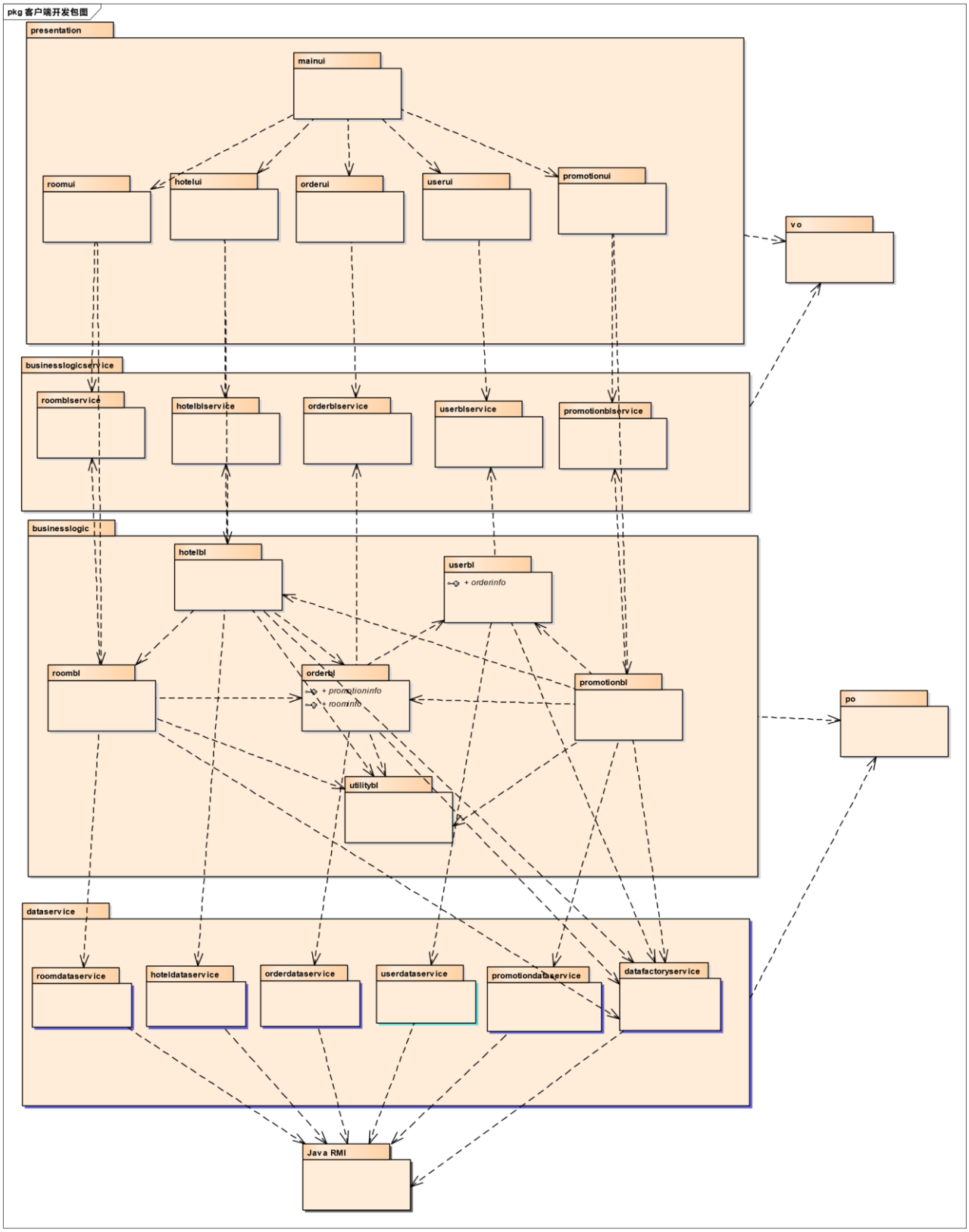

图 4-1-1 酒店预订系统客户端开发包图

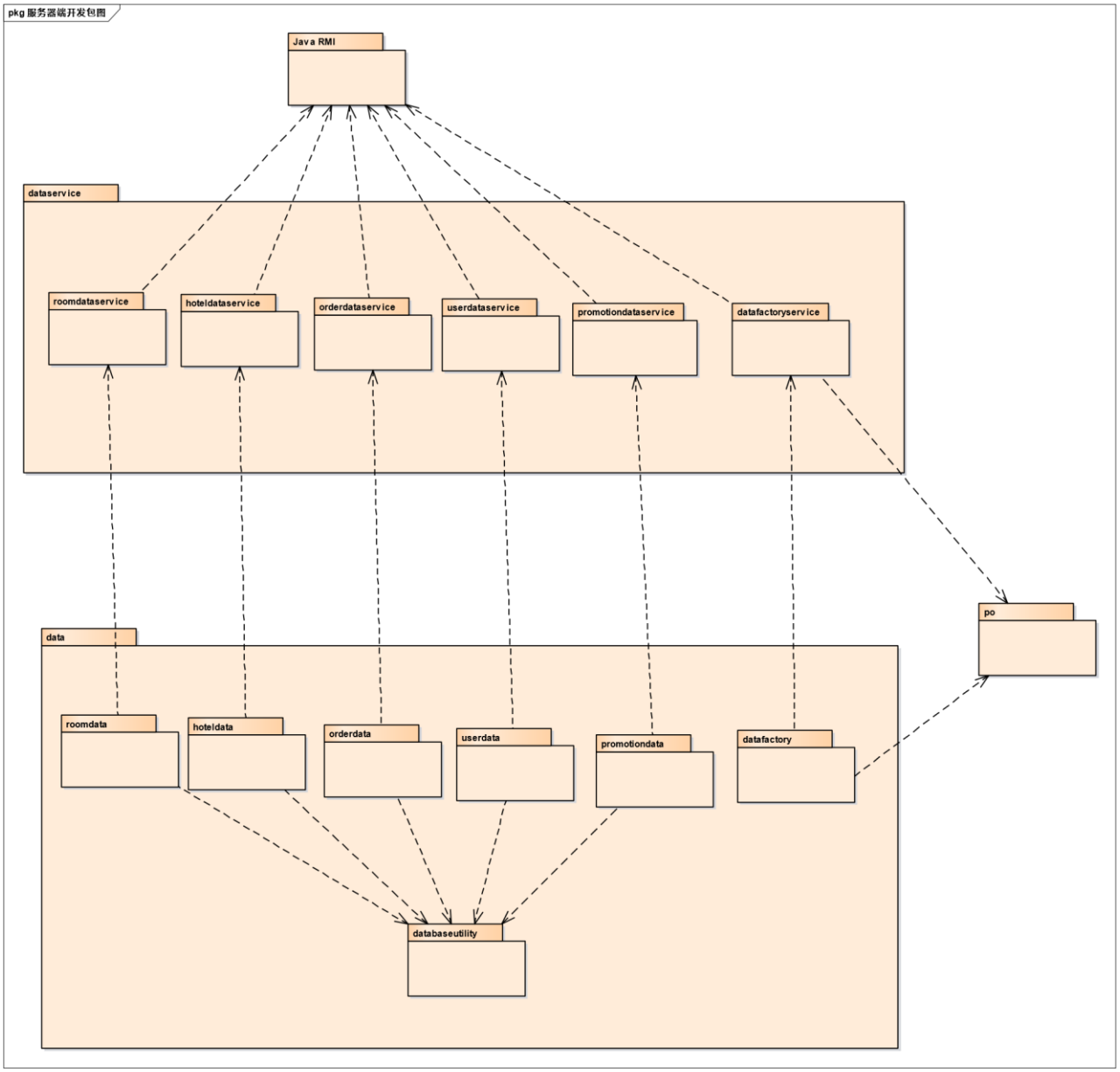

图 4-1-2 酒店预订系统服务器端开发包图

4.2. 运行时进程

在酒店预订系统中，会有多个客户端进程和一个服务器端进程，其进程图如图 4-2-1 所示。结合部署图，客户端进程是在客户端机器上运行，服务器端进程在服务器端机器上运行。

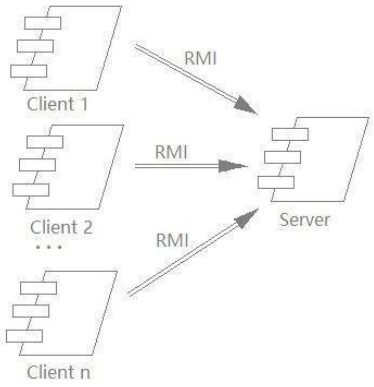

图 4-2-1 进程图

4.3. 物理部署

酒店预订系统中客户端构件是放在客户端机器上，服务器端构件是放在服务器端机器上。在客户端节点上，还要部署 RMIStub 构件。由于 Java RMI 构件属于 JDK 1.8 的一部分。所以，在系统 JDK 环境已经设置好的情况下，不需要再独立部署。部署图如图 4-3-1 所示。

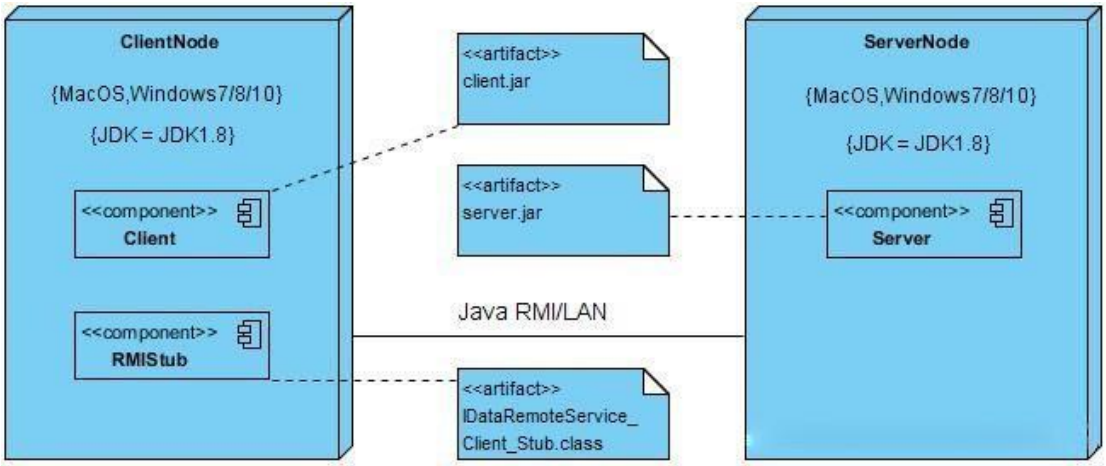

图 4-3-1 部署图

5. 接口视角

5.1. 模块的职责

客户端模块和服务器端模块视图分别如图 5-1-1 和图 5-1-2 所示。

客户端各层和服务端各层的职责分别入表 5-1-1 和表 5-1-2 所示。

图 5-1-1 客户端模块视图

图 5-1-2 服务器端模块视图

表 5-1-1 客户端各层的职责

层	职责
启动模块	负责初始化网络通信机制，启动用户界面
用户界面层	基于窗口的酒店预订系统客户端用户界面
业务逻辑层	对于用户界面的输入进行响应并进行业务处理逻辑

客户端网络模块	利用 Java RMI 机制查找 RMI 服务

表 5-1-2 服务器端各层的职责

层	职责
启动模块	负责初始化网络通信机制，启动用户界面
数据层	负责数据的持久化及数据访问接口
服务器端网络模块	利用 Java RMI 机制开启 RMI 服务，注册 RMI 服务

每一层只是使用下方直接接触的层。层与层之间仅仅是通过接口的调用来完成的。层之间调用的接口如表 5-1-3 所示。

表 5-1-3 层之间调用的接口

接口	服务调用方	服务提供方
RoomBLService HotelBLService OrderBLService UserBLService PromotionBLService	客户端展示层	客户端业务逻辑层

RoomDataService HotelDataService OrderDataService UserDataService PromotionDataService DatabaseFactory	客户端业务逻辑层	服务端数据层

5.2. 用户界面层的分解

根据需求，系统存在 26 个用户界面：酒店工作人员主界面，客房信息管理界面，登录界面，房间信息更新界面，客房信息录入界面，酒店基本信息界面，订单界面，营销策略制定界面，客户主界面，会员注册界面，网站管理人员主界面，用户管理界面，添加酒店界面，历史酒店浏览界面，个人基本信息界面，订单管理界面，订单评价界面，信用记录界面，酒店搜索界面，酒店浏览界面，订单生成界面，网站营销人员主界面，制定营销策略界面，制定会员等级及折扣界面，异常订单申诉处理界面，信用值管理界面。界面跳转如图 5-2-1 所示。

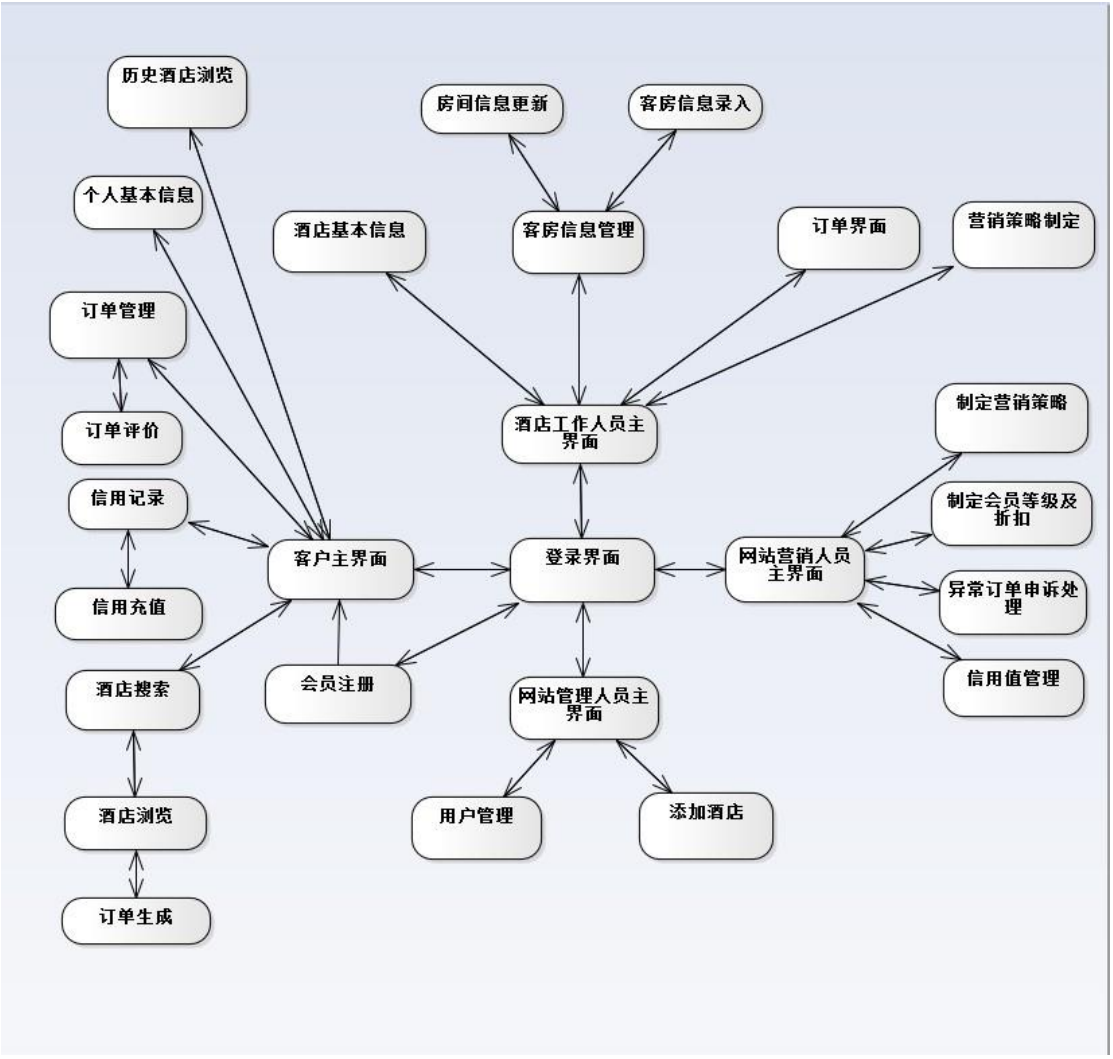

图 5-2-1 用户界面跳转

服务器端和客户端的用户界面设计接口是一致的，只是具体的页面不一样。

用户界面类如图 5-2-2 所示。

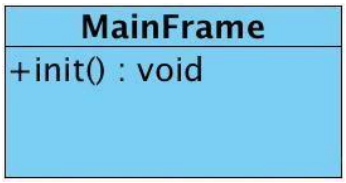

图 5-2-2 用户界面类

5.2.1. 用户界面层模块的职责

如表 5-2-1 所示为用户界面层模块的职责。

表 5-2-1 用户界面层模块的职责

模块	职责
MainFrame	界面 Frame，负责界面的显示和界面的跳转

5.2.2. 用户界面层模块的接口规范

用户界面层模块的接口规范如表 5-2-2-1 所示。 表

5-2-2-1 用户界面层模块的接口规范

MainFrame	语法	init(args:String[])
	前置条件	无
	后置条件	显示 Frame 以及 LoginPanel

用户界面层需要的服务接口如表 5-2-2-2 所示。

表 5-2-2-2 用户界面层模块需要的服务接口

服务名	服务
UserBLService.login(String id, String password)	登录界面的业务逻辑接口
*BLService	每个界面都有一个相应的业务逻辑接口

5.2.3. 用户界面模块设计原理

用户界面利用 Java 的 Swing 和 AWT 库来实现。

5.3. 业务逻辑层的分解

业务逻辑层包括多个针对界面的业务逻辑处理对象。业务逻辑层的设计如图

5-3-1 所示。

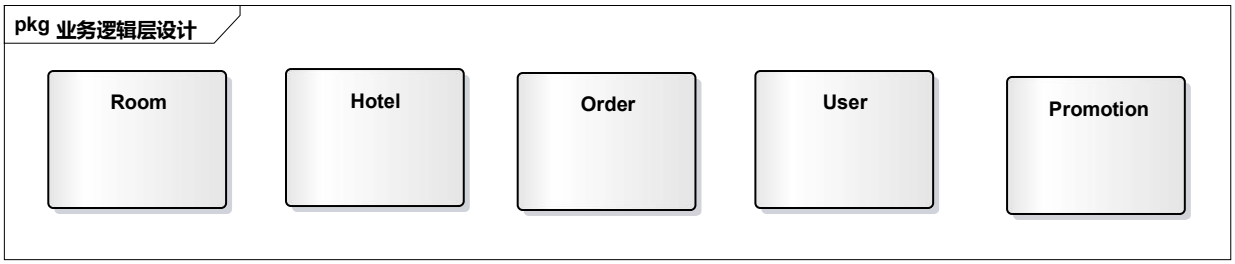

图 5-3-1 业务逻辑层的设计

5.3.1. 业务逻辑层模块的职责 业务逻辑层模块的职责如表

5-3-1-1 所示。

表 5-3-1-1 业务逻辑层模块的职责

模块	职责
hotelbl	负责实现酒店界面所需要的服务
roombl	负责实现查看成本收益表和经营情况表界面所需要的服务
promotionbl	负责实现营销策略界面所需要的服务
orderbl	负责实现订单界面所需要的服务
userbl	负责实现对应与登录界面所需要的服务

5.3.2. 业务逻辑层模块的接口规范

表 5-3-2-1 hotelbl 模块的接口规范

提供的服务（供接口）		
Hotel.messagelook	语法	public ResultMessage messagelook(HotelVO hvo)

	前置条件	用户已登录成功
	后置条件	查找地址和商圈内的酒店，返回范围内的酒店信息
Hotel.messagesearch	语法	public ArrayList<HotelVO> messagesearch(MessageInput in)
	前置条件	用户已登录成功
	后置条件	查找符合条件的酒店并返回酒店列表
Hotel.historylook	语法	public ArrayList<HotelVO> historylook(String id)
	前置条件	用户已登录成功
	后置条件	查找用户的历史订单的酒店并返回历史酒店列表
Hotel.messagemaintain	语法	public ResultMessage messagemaintain(MessageInput in)
	前置条件	酒店工作人员已登陆
	后置条件	系统修改酒店的基本信息

Hotel.accountadmin	语法	public ResultMessage accountadmin(MessageInput in)
	前置条件	网站管理人员已登录
	后置条件	系统增加酒店帐号和名称以及该酒店工作人员的帐号
Hotel.setscore	语法	public boolean setscore(int score)
	前置条件	用户对酒店作出评分
	后置条件	系统增加评分记录，更新评分的均值
Hotel.setcomment	语法	public boolean setcomment(String comment)
	前置条件	用户对酒店作出评价
	后置条件	系统增加酒店的评价记录
Hotel.getroominfo	语法	public boolean getroominfo(HotelVO vo)
	前置条件	用户要查看酒店详细信息
	后置条件	系统显示剩余房间

Hotel.pricesort	语法	public HotelVO pricesort(ArrayList<Hotel> ah)
	前置条件	客户要求酒店按价格高低排序
	后置条件	显示价格由低到高的酒店列表
Hotel.starsort	语法	public HotelVO starsort(ArrayList<Hotel> ah)
	前置条件	客户要求酒店按星级排序
	后置条件	显示按照星级排序的酒店列表
Hotel.scoresort	语法	public HotelVO scoresort(ArrayList<Hotel> ah)
	前置条件	客户要求酒店按平均评分排序
	后置条件	显示平均评分由高到低的酒店列表
Hotel.gethistoryorder	语法	public OrderVO gethistoryorder(HotelVO vo)
	前置条件	客户要求查看酒店细节

	后置条件	系统显示此客户在该酒店的历史订单
Hotel.gethistoryhotel	语法	public ArrayList<HotelVO> gethistoryhotel()
	前置条件	客户要求查看预定过的酒店
	后置条件	系统显示此客户的历史预定酒店

需要的服务（需接口）	
服务名	服务
DatabaseFactroy.getHotelDatabase	得到 Hotel 数据库的服务的引用
HotelDataService.find(String location)	在数据区寻找符合在区域中的酒店
HotelDataService.search(MessageInput in)	在数据区寻找符合条件的酒店
HotelDataService.getHistory(HotelPO po)	在数据区寻找该酒店的历史订单
HotelDataService.insert(HotelPO po)	插入单一持久化对象
HotelDataService.delete(HotelPO po)	删除单一持久化对象
HotelDataService.update(MessageInput in)	更新单一持久化对象
Order.gethistory(userVO vo)	得到该客户订过的酒店

Order.findByType(hotelVO vo)	根据选中的历史酒店得到订单

表 5-3-2-2 roombl 模块的接口规范

提供的服务（供接口）		
Room.messageupdate	语法	public ResultMessage messageupdate(MessageInput in)
	前置条件	酒店工作人员已登录成功
	后置条件	系统修改房间信息
Room.messageadd	语法	public ResultMessage messageadd(MessageInput in)
	前置条件	酒店工作人员已登录成功
	后置条件	系统修改酒店含有的房间信息
Room.getorderinfo	语法	public RoomVO messageadd(OrderVO vo)
	前置条件	无
	后置条件	得到订单上的客户的个人信息和入住退房时间

Room.getRoom	语法	public ArrayList<RoomVO> getRoom()
	前置条件	无
	后置条件	系统返回可入住房间
Room.getPrice	语法	public long getPrice(String roomType)
	前置条件	无
	后置条件	系统返回该类型房间价格

需要的服务（需接口）	
服务名	服务
DatabaseFactroy.getRoomDatabase	得到 Room 数据库的服务的引用
RoomDataService.insert(RoomPO po)	插入单一持久化对象
RoomDataService.update(RoomPO po)	更新单一持久化对象
RoomDataService.delete(RoomPO po)	删除单一持久化对象
Order.getRoomInfo(OrderVO vo)	得到该订单关于房间的信息
RoomDataService.changestate(String state)	在数据库更新房间状态

表 5-3-2-3 promotionbl 模块的接口规范

提供的服务（供接口）		
Promotion.madebyhotel	语法	public ResultMessage madebyhotel (PromotionVO vo)
	前置条件	酒店工作人员已登录成功
	后置条件	系统录入此酒店营销策略，并发布
Promotion.madebyweb	语法	public ResultMessage madebyweb (PromotionVO vo)
	前置条件	网站营销人员已登录成功
	后置条件	系统录入此网站营销策略，并发布
Promotion.memberlevelmade	语法	public ResultMessage memberlevelmade(PromotionVO vo)
	前置条件	网站营销人员已登录成功
	后置条件	系统记录会员等级标准
Promotion.cancel	语法	public boolean cancel(Promotion promotion)

	前置条件	无
	后置条件	在系统中取消此订单
Promotion.getPromotion	语法	public ArrayList<PromotionVO vo> getPromotion(String userID)
	前置条件	无
	后置条件	系统根据客户返回能应用的营销策略

需要的服务（需接口）	
服务名	服务
DatabaseFactroy.getPromotionDatabase	得到 Promotion 数据库的服务的引用
PromotionDataService.insert(PromotionPO po)	插入单一持久化对象
PromotionDataService.delete (PromotionPO po)	删除单一持久化对象
PromotionDataService.memberlevelinsert(MessageInput in)	在数据区添加会员等级制度
PromotionDataService.memberlevelupdate(MessageInput in)	在数据区更新会员等级制度

表 5-3-2-4 orderbl 模块的接口规范

提供的服务（供接口）		
Order.show	语法	public ArrayList<OrderVO> show()
	前置条件	打开订单的界面
	后置条件	系统返回所有订单信息
Order.getRoomInfo	语法	public ResultMessage getRoomInfo (OrderVO vo)
	前置条件	订单状态发生变化
	后置条件	系统返回该订单关于房间的信息
Order.cancel	语法	public void cancel()
	前置条件	无
	后置条件	关闭订单界面
Order.gethistory	语法	public ArrayList<Hotel> gethistory(userVO vo)

	前置条件	无
	后置条件	系统返回该用户订过的酒店

Order.findByHotel	语法	public ArrayList<OrderVO> findByType(hotelVO vo)
	前置条件	客户选择历史酒店
	后置条件	系统根据选中的历史酒店返回订单
Order.findByType	语法	public ArrayList<OrderVO> findByType(String type)
	前置条件	选择某种类型
	后置条件	系统根据选中的类型查找订单，并返回订单
Order.showDetail	语法	public OrderVO showDetail(String orderID)
	前置条件	选择某个订单
	后置条件	系统根据选中订单的订单号查找订单详情，并返回

Order.cancelOrder	语法	public void cancelOrder(String orderID,Time currentTime)
	前置条件	未执行正常订单
	后置条件	系统将订单类型变为已撤销，保存撤销时间
Order.deduct	语法	public void duduct(OrderVO order)
	前置条件	撤销订单时间距最晚订单执行时间小于六小时
	后置条件	系统扣除信用值，信用值为订单价值的二分之一，更新会员等级
Order.whetherDeduct	语法	public Boolean whetherDeduct(Time currentTime,String orderID)
	前置条件	订单被撤销
	后置条件	系统返回判断结果

Order.makeOrder	语法	public void makeOrder(Time currentTime,Time in,Time out,Time ddl,RoomType roomType,int num,int numOfPerson,boolean haveChild)
	前置条件	订单成功提交
	后置条件	系统生成订单数据
Order.whetherMake	语法	public boolean whetherMake(String uerID)
	前置条件	订单被提交

	后置条件	系统返回判断结果
Order.done	语法	public void done(String orderID,String userID)
	前置条件	客户按时入住
	后置条件	系统改变订单类型为已执行，为客户增加信用值，更新会员等级
Order.abnormalOrder	语法	public void abnormalOrder(String orderID,String userID)

	前置条件	客户未按时入住
	后置条件	系统改变订单类型为异常，为客户减去信用值，更新会员等级
Order.delayIn	语法	public void delayIn(String orderID,String userID)
	前置条件	客户延迟入住
	后置条件	系统改变订单类型为已执行，为客户恢复信用值，更新会员等级
Order.endExecute	语法	public void endExecute()
	前置条件	订单执行完成
	后置条件	系统结束订单执行任务，持久化更新涉及的领域对象的数据

Order.comment	语法	public void comment(String comment,OrderVO order)
	前置条件	订单已执行
	后置条件	系统更新订单信息，并显示评价

Order.findByID	语法	public OrderVO findByID(String ID)
	前置条件	ID 输入正确
	后置条件	系统根据 ID 查找，并返回订单
Order.regain()	语法	public void regain(OrderVO vo,Choice choice)
	前置条件	申诉合理
	后置条件	系统恢复客户信用值，更新会员等级
Order.getPrice()	语法	public long getPrice(OrderVO vo,String userID)
	前置条件	订单生成
	后置条件	系统计算订单价值并返回
Order.payment()	语法	public ResultMessage payment(OrderVO vo)
	前置条件	客户未支付订单

	后置条件	系统完成支付任务

需要的服务（需接口）	
服务名	服务
DatabaseFactory.getOrderDatabase	得到 Order 数据库的服务的引用
OrderDataService.findByType(String type)	根据类型查找单一持久化对象
OrderDataService.findByID(String ID)	根据 ID 查找单一持久化对象
OrderDataService.insert(OrderPO po)	插入单一持久化对象
OrderDataService.update(OrderPO po)	更新单一持久化对象
OrderDataService.delete(OrderPO po)	删除单一持久化对象
User.updateCredit(UserVO vo,long credit)	更新客户信用值
User.updateLevel(UserVO vo)	更新客户会员等级
User.updateCreditRecord(UserVO vo)	更新客户信用记录
Room.getRoom()	得到可入住房间

Room.getPrice(String roomType)	得到特定类型房间的价格
Promotion.getPrmotion(user ID)	得到促销策略

表 5-3-2-5 userbl 模块的接口规范

供的服务（供接口）		
User.findByID	语法	public ResultMessage findByID(String userID)
	前置条件	输入的 ID 有效
	后置条件	系统根据客户 ID 查找客户信息并返回
User.update	语法	pulic ResultMessage update(userVO vo)
	前置条件	选择的客户符合要求
	后置条件	系统持久化更新该客户信息
User.add	语法	public ResultMessage add(userVO vo)
	前置条件	输入的客户信息符合要求

	后置条件	系统持久化增加客户信息

User.delete	语法	public ResultMessage delete(userVO vo)
	前置条件	选择的客户符合要求
	后置条件	系统持久化删除该客户信息
User.showCredit	语法	public long showCredit(String userID)
	前置条件	输入的 ID 有效
	后置条件	系统根据客户 ID 查找其信用值并返回
User.updateCredit	语法	public ReusltMessage updateCredit(userVO vo,long credit)
	前置条件	客户充入一定金额
	后置条件	系统持久化更新该客户信用值
User.upadteLevel	语法	public ResultMessage updateLevel(userVO vo)

	前置条件	客户信用值改变
	后置条件	系统持久化更新该客户会员等级

User.updateCreditRecord	语法	public ResultMessage updateCreditRecord(userVO vo)
	前置条件	客户信用值改变
	后置条件	客户持久化更新该客户信用记录
User.register	语法	public ResultMessage register(userVO vo)
	前置条件	客户未注册
	后置条件	系统持久化注册该客户
User.maintainPersonalInfo	语法	public ResultMessage maintainPeersonalInfo(userVO vo)
	前置条件	客户已登录
	后置条件	系统持久化维护该客户信息

User.showCreditRecord	语法	public CreditRecordVO showCreditRecord(userVO vo)
	前置条件	无
	后置条件	系统查找该用户信用记录并返回

User.topUp	语法	public ResultMessage topUp(long money)
	前置条件	输入金额有效
	后置条件	系统增加该客户信用值
User.login	语法	public UserType login(String ID, String password)throws RemoteException
	前置条件	账号密码符合输入规则
	后置条件	系统返回登录类型
User.logout	语法	public ResultMessage logout(String ID)

	前置条件	客户已登录
	后置条件	无

需要的服务（需接口）	
服务名	服务
DatabaseFactory.getUserDatabase	得到 User 数据库的服务的引用
UserDataService.findByID(String ID)	根据 ID 查找客户信息
UserDataService.insert(userPO po)	插入单一持久化对象
UserDataService.update(userPO po)	更新单一持久化对象
UserDataService.delete(userPO po)	删除单一持久化对象

5.4. 数据层的分解

数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增、删、

改、查。各业务逻辑需要的服务由各 DataService 接口提供。持久化数据的保存采用数据库进行保存。

以 HotelDataService 为例，数据层模块的描述具体如图 5-4-1 所示。

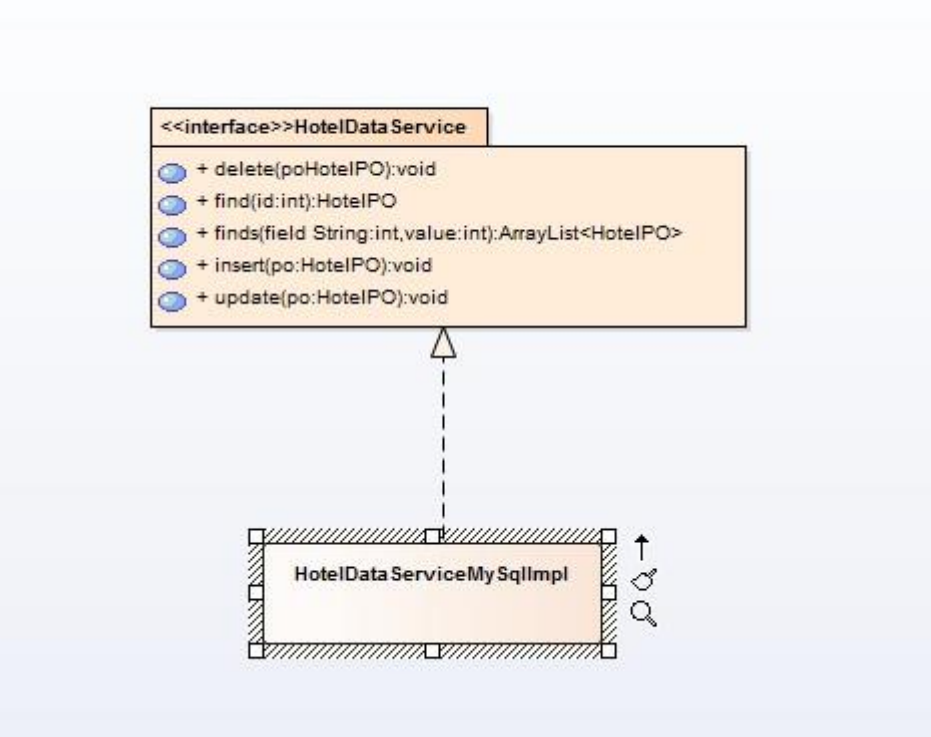

图 5-4-1 HotelDataService 数据层模块

5.4.1. 数据层模块的职责

数据层模块的职责如表 5-4-1 所示。

表 5-4-1 数据层模块的职责

模块	职责
hotelDataService	基于数据库的持久化数据的接口，提供 HotelPO 的集体载入、集体保存、增、删、改、查服务
roomDataService	基于数据库的持久化数据的接口，提供 RoomPO 的集体载入、集体保存、增、删、改、查服务
orderDataService	基于数据库的持久化数据的接口，提供 OrderPO 的集体载入、集体保存、增、删、改、查服务
userDataService	基于数据库的持久化数据的接口，提供 UserPO 的集体载入、集体保存、增、查服务
promotionDataService	基于数据库的持久化数据的接口，提供 PromotionPO 的集体载入、集体保存、增、查服务

5.4.2. 数据层模块的接口规范

数据层模块的接口规范如表 5-4-2-1 ~ 5-4-2-5 所示。表 5-4-2-1 数据层模块(HotelDataService)的接口规范

提供的服务（供接口）		
HotelDataService.find	语法	Public HotelPO find(long id) throws RemoteException
	前置条件	无
	后置条件	按 ID 进行查找返回相应的 HotelPO 结果
HotelDataService.insert	语法	public void insert(HotelPO po) throws RemoteException
	前置条件	同样 ID 的 po 在 Mapper 中不存在
	后置条件	在数据库中增加一个 po 记录
HotelDataService.delete	语法	public void delete(HotelPO po) throws RemoteException
	前置条件	在数据库中存在同样 ID 的 po
	后置条件	删除一个 po
HotelDataService.update	语法	public void update(HotelPO po) throws RemoteException
	前置条件	在数据库中存在相同 ID 的 po
	后置条件	更新一个 po

HotelDataService.init	语法	public void init(HotelPO po) throws RemoteException
	前置条件	无
	后置条件	初始化持久化数据库
HotelDataService.finish	语法	public void finish(HotelPO po) throws RemoteException
	前置条件	无
	后置条件	结束持久化数据库的使用

表 5-4-2-2 数据层模块(RoomDataService)的接口规范

提供的服务（供接口）		
RoomDataService.find	语法	Public RoomPO find(long id) throws RemoteException
	前置条件	无
	后置条件	按 ID 进行查找返回相应的 RoomPO 结果
RoomDataService.insert	语法	public void insert(RoomPO po) throws RemoteException
	前置条件	同样 ID 的 po 在 Mapper 中不存在

	后置条件	在数据库中增加一个 po 记录
RoomDataService.delete	语法	public void delete(RoomPO po) throws RemoteException
	前置条件	在数据库中存在同样 ID 的 po
	后置条件	删除一个 po
RoomDataService.update	语法	public void update(RoomPO po) throws RemoteException
	前置条件	在数据库中存在相同 ID 的 po
	后置条件	更新一个 po
RoomDataService.init	语法	public void init(RoomPO po) throws RemoteException
	前置条件	无
	后置条件	初始化持久化数据库
RoomDataService.finish	语法	public void finish(RoomPO po) throws RemoteException
	前置条件	无
	后置条件	结束持久化数据库的使用

表 5-4-2-3 数据层模块(OrderDataService)的接口规范

提供的服务（供接口）

OrderDataService.find	语法	Public OrderPO find(long id) throws RemoteException
	前置条件	无
	后置条件	按 ID 进行查找返回相应的 OrderPO 结果
OrderDataService.insert	语法	public void insert(OrderPO po) throws RemoteException
	前置条件	同样 ID 的 po 在 Mapper 中不存在
	后置条件	在数据库中增加一个 po 记录
OrderDataService.delete	语法	public void delete(OrderPO po) throws RemoteException
	前置条件	在数据库中存在同样 ID 的 po
	后置条件	删除一个 po
OrderDataService.update	语法	public void update(OrderPO po) throws RemoteException
	前置条件	在数据库中存在相同 ID 的 po
	后置条件	更新一个 po
OrderDataService.init	语法	public void init(OrderPO po) throws RemoteException
	前置条件	无

	后置条件	初始化持久化数据库
OrderDataService.finish	语法	public void finish(OrderPO po) throws RemoteException
	前置条件	无
	后置条件	结束持久化数据库的使用

表 5-4-2-4 数据层模块(UserService)的接口规范

提供的服务（供接口）		
UserDataService.find	语法	Public UserPO find(long id) throws RemoteException
	前置条件	无
	后置条件	按 ID 进行查找返回相应的 OrderPO 结果
UserDataService.insert	语法	public void insert(UserPO po) throws RemoteException
	前置条件	同样 ID 的 po 在 Mapper 中不存在
	后置条件	在数据库中增加一个 po 记录
UserDataService.delete	语法	public void delete(UserPO po) throws RemoteException

	前置条件	在数据库中存在同样 ID 的 po
	后置条件	删除一个 po
UserDataService.update	语法	public void update(UserPO po) throws RemoteException
	前置条件	在数据库中存在相同 ID 的 po
	后置条件	更新一个 po
UserDataService.init	语法	public void init(UserPO po) throws RemoteException
	前置条件	无
	后置条件	初始化持久化数据库
UserDataService.finish	语法	public void finish(UserPO po) throws RemoteException
	前置条件	无
	后置条件	结束持久化数据库的使用

表 5-4-2-5 数据层模块(PromotionDataService)的接口规范

提供的服务（供接口）		
PromotionDataService.find	语法	Public PromotionPO find(long id) throws RemoteException

	前置条件	无
	后置条件	按 ID 进行查找返回相应的 PromotionPO 结果
PromotionDataService.insert	语法	public void insert(PromotionPO po) throws RemoteException
	前置条件	同样 ID 的 po 在 Mapper 中不存在
	后置条件	在数据库中增加一个 po 记录
PromotionDataService.delete	语法	public void delete(PromotionPO po) throws RemoteException
	前置条件	在数据库中存在同样 ID 的 po
	后置条件	删除一个 po
PromotionDataService.update	语法	public void update(PromotionPO po) throws RemoteException
	前置条件	在数据库中存在相同 ID 的 po

	后置条件	更新一个 po
PromotionDataService.init	语法	public void init(PromotionPO po) throws RemoteException
	前置条件	无
	后置条件	初始化持久化数据库
PromotionDataService.finish	语法	public void finish(PromotionPO po) throws RemoteException
	前置条件	无
	后置条件	结束持久化数据库的使用

6. 信息视角

6.1. 数据持久化对象

系统的 PO 类是对应的相关的实体类。系统主要的 PO 类如下所示：

- HotelPO 类：酒店 PO 类，包含酒店地址，酒店商圈，酒店星级，酒店评分，酒店简介，设施服务，酒店名称，酒店预订状态，酒店联系方式，酒店账号

- RoomPO 类：房间 PO 类，包含房间号，房间状态，房间类型，房间数量，房间价格

- PromotionPO 类：促销策略 PO 类，包含促销策略名称和策略编号，活动开始和结束时间，策略商圈，会员适用等级，会员适用类型，最低订购数量，折扣额度

- OrderPO 类：订单 PO 类，包含订单号，订单状态，人数，有无儿童，订单价值，订购房间类型，订购数量，预定入住时间，预定退房时间，最晚执行时间，评价，评分

- UserPO 类：客户 PO 类，包含客户名称，客户账号，客户联系方式，会员等级，会员种类，生日，企业名称

- CreditRecordPO 类：信用记录类，包含记录时间，动作（订单执行，订单异常，订单撤销，充值），信用度变化，当前信用值

持久化对象如 HotelPO 的定义如图 6-1-1 所示。

```
public class HotelPO implements Serializable {

    private String hotelAddress;
    private String hotelDistrict;
    private String hotelStar;
    private String hotelProfile;
    private String hotelService;
    private String hotelName;
    private String hotelPhone;
    private String hotelReservation;
    private String hotelAccout;

    public OrderPO (String hAddress,String hDistrict,String hStar,String
hProfile,String hService,String hName,String hPhone,String
hReservation,String hAccout) {

     hotelAddress=hAddress;
    hotelDistrict=hDistrict;
    hotelStar=hStar;
    hotelProfile=hProfile;
    hotelService=hService;
    hotelName=hName;
    hotelPhone=hPhone;
    hotelReservation=HReservation;
    hotelAccout=hAccout;

    }

 public String gethotelAddress {    return
hotelAddress;
    }
```

```
    public void sethotelAddress(String s) {
        hotelAddress=s;
    }
    public String gethotelDistrict {
    return hotelDistrict;
    }
    public void sethotelDistrict (String s) {
        hotelDistrict=s;
    }
 public String gethotelStar {    return
hotelStar;
    }
    public void sethotelStar(String s){
    hotelStar=s;
    }
 public String gethotelProfile {    return
hotelProfile;
    }
    public void sethotelProfile(String s){
        hotelProfile=s;
    }
 public String gethotelService{    return
hotelService;
    }
    public String sethotelService(String s){
        hotelService=s;
    }
    public String gethotelName {
    return hotelName;
    }
```

```
    public String sethotelName(String s){
        hotelName=s;
    }
    public      String      gethotelPhone
    {  return hotelPhone;
    }
    public String sethotelPhone(String s){
        hotelPhone=s;
    }
    public      String      gethotelReservation
    {  return hotelReservation;
    }
    public String sethotelReservation(String s){
        hotelReservation=s;
    }
    public      String      gethotelAccount
    {  return hotelAccount;
    }
    public String sethotelAccount(String s){
        hotelAccount=s;
    }

}
```

图 6-1-1 持久化酒店对象 HotelPO 的定义

6.2. 数据库持久化

采用数据库存储，将数据存储在各个 PO 对应的表里。