# 酒店管理系统体系结构设计文档

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14日

## 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 全体人员 | 2016-10-11 | 最初草稿（创建文档模板） | V0.1 |

## 1、引言

### 1.1 编制目的

本报表详细完成对酒店管理系统的概要设计，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

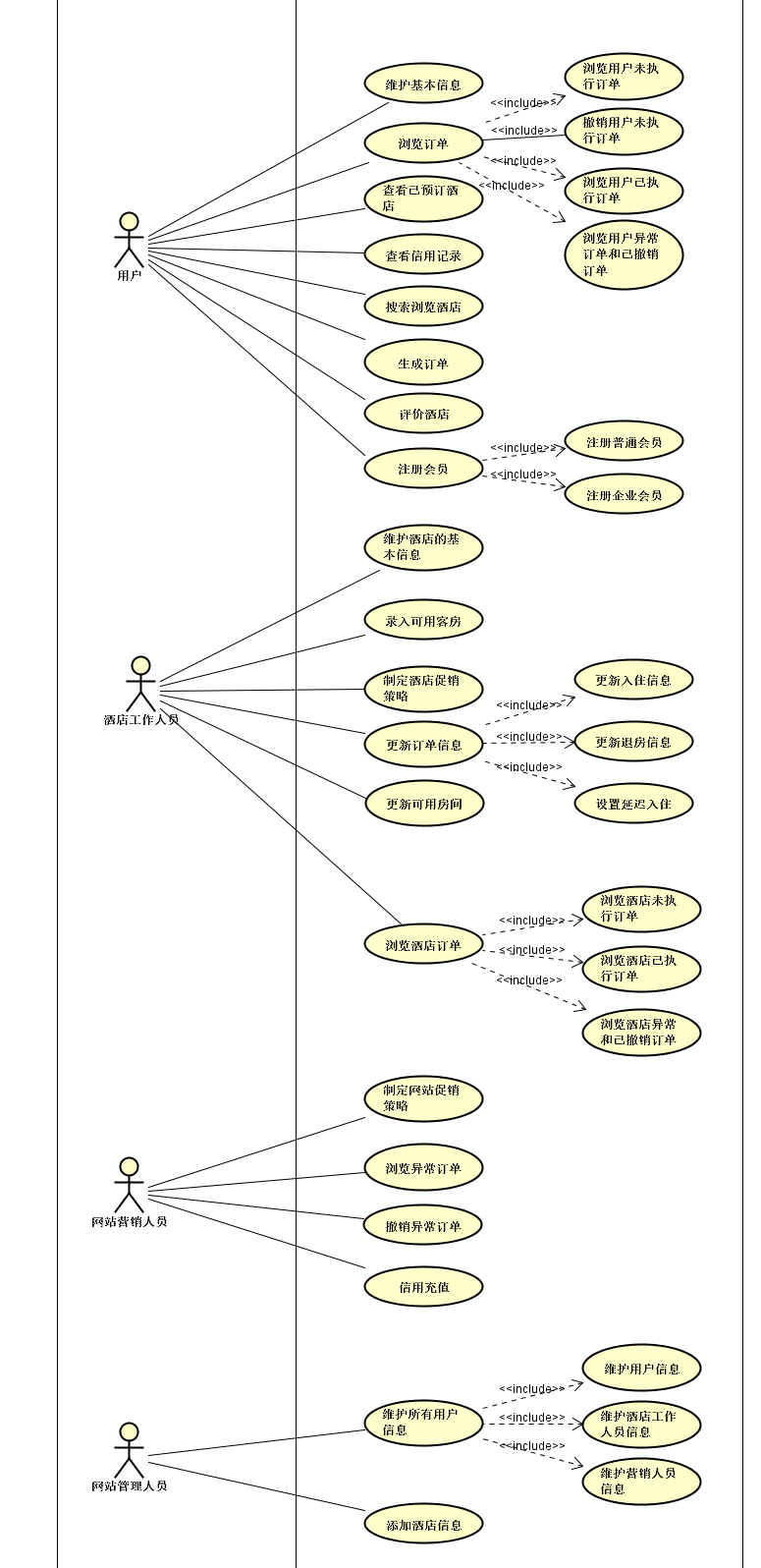
|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| UI | 表示某展示层 |  |
| BL | 表示某逻辑层 |  |
| Data | 表示某数据层 |  |
| RMI | 表示远程方法调用 |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

### 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

## 2、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。酒店管理系系统主要是应用于在线的互联网酒店预定系统，主要功能见用例图如下。



## 3、逻辑视角

酒店管理系统中，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1和图2所示。



图一 参照体系结构风格的包图表达逻辑视角

图二 软件体系结构逻辑设计方案

## 4、组合视角

[软工2 166]与抽象的逻辑设计相比，实现物理设计要考虑更多的实现细节，这些细节有：

1. presentation层与logic层被置于客户端，data层被置于服务器端，那么logic层的开发包不可能依赖于data层的开发包。使用RMI技术，将data层开发包分解为置于客户端的dataservice接口包和置于服务器端的data层开发包。这样一来，logic层开发包依赖于dataservice包，dataservice包和data层的开发包都依赖于RMI类库包。
2. 所有的data层开发包都需要进行数据持久化（例如读写数据库、读写文件等），所以它们会有一些重复代码，可以将重复代码独立为新的开发包，然后所有的data层开发包都依赖于databasetility。datavaseutility会依赖于Spring JDBC类库包。
3. 所有的presentation层开发包都需要使用图形类型建立界面，都要依赖于图形界面类库包。
4. 此外，presentation层实现时，由mainui包负责整个页面之间的跳转逻辑。其他各包负责各自页面自身的功能。
5. 在分层风格的典型设计中，不希望高层直接依赖于低层，而是为低层建立接口包，实现依赖倒置原则，所以应该调整为：各presentation层开发包（调用）依赖于logic层接口包blservice包，logic层开发包也依赖于（实现了）data层接口包dataservice包。
6. 在分层风格的典型设计中，presentation层与logic层之间、logic层与data层之间可能会传递复杂数据对象，那么相邻两层都需要使用数据对象声明，所以需要将数据对象声明独立为开发包（VO包和PO包）。
7. 使用依赖倒置原则消除包的循环依赖现象，将循环依赖变为单向依赖：

8）在logic层中，初始化和业务逻辑层上下文的工作被分配到utility包中。

经过细节改进，最终建立酒店管理系统开发包设计如表4.1-1，其局部包图如图4.1-1和4.1-2所示：

### 4.1 开发包图

表4.1-1 酒店管理系统的最终开发包设计

|  |  |
| --- | --- |
| 开发（物理）包 | 依赖的其他开发包 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| VO |  |
| PO |  |
| databaseutility | Spring JDBC |
| 界面类库包 |  |
| JAVA RMI | java.swing,java.awt,java.2D,java.3D |

图4.1-1 酒店管理系统客户端开发包图

图4.1-2 酒店管理系统服务器端开发包图

### 4.2 运行时进程

在酒店管理系统中，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所示。结合部署图，客户端进程实在客户端机器上运行，服务器端进程是在服务器端机器上运行。

 图4.2 进程图

### 4.3 物理部署

酒店管理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JavaRMI构件属于JDK6.0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所示。

4.3 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。客户端各层和服务器端各层的职责分别如表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 | 基于窗口的进销存系统客户端用户界面。 |
| 业务逻辑层 | 对于用户界面的输入响应和业务处理逻辑。 |
| 客户端网络模块 | 利用Java RMI机制查找RMI服务,检测网络连接状态，进行断线重连 |

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 |
|  | 客户端展示层 | 客户端业务逻辑层 |
|  | 客户端业务逻辑层 | 服务器端数据层 |

借用用户管理用例来说明层之间的调用，如图5.1-3所示。每一层之间都是由上层依赖了一个接口（需接口），而下层实现这个接口（供接口）。UserBLService提供了User界面所需要的所有业务逻辑功能UserDataService提供了对数据库的增、删、改、查等操作。这样的实现就大大降低了层与层之间的耦合。



图5.1-3 用户管理用例层之间调用的接口

### 5.2 用户界面层的分解

根据需求，系统存在29个用户界面：。

界面跳转如图5.2所示。

图5.2 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5.2.1-1所示为用户界面层模块的职责。

表5.2.1-1 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

|  |  |  |
| --- | --- | --- |
| Main\_ui | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |
|  | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 |  |

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 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| businesslogicservice.LoginBLService | 登陆界面的业务逻辑接口 |
| businesslogicservice.\*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。例如，User对象负责处理登陆界面的业务逻辑；User对象负责处理管理用户的业务逻辑。业务逻辑层的设计如图5.3-1所示。

图5.3-1 业务逻辑层

#### 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 |
| UserService | 负责用户注册和所有用户的登录；负责用户信息管理 |
| OrderService | 负责订单的浏览、下单、撤销、评价，酒店的更新入住、退房、延迟入住信息、浏览订单，网站促销人员的浏览、撤销异常订单 |
| HotelService | 负责搜索浏览酒店，酒店基本信息管理；网站管理人员添加酒店 |
| RoomService | 负责录入可用客房和可用客房管理 |
| StaffService | 负责网站管理人员管理用户、酒店工作人员、网站营销人员信息 |
| CreditService | 负责用户查看信用记录，网站营销人员信用充值 |
| HotelDiscount | 负责酒店工作人员制定酒店促销策略 |
| OfflineRecord | 负责酒店工作人员更新线下入住信息 |
| WebDiscount | 负责网站营销人员制定促销策略 |

#### 5.3.2 业务逻辑层模块的接口规范

表5.3.2-1 UserService模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | 返回该用户的详细信息 | |
| 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 | 返回注册的结果信息 | |
| 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 | 返回更新的结果信息 | |
| UserService.login | 语法 | public ResultMessage 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 | 返回登录的结果信息 | |
| 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 |
| StaffService.login(String username,String password) | | | 职员的登录服务 |
|  | | |  |

表5.3.2-2模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
|  | | |  |

表5.3.2-3模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
|  | 语法 |  | |
| 前置条件 |  | |
| 后置条件 |  | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |
|  | | |  |

### 5.4 数据层的分解

数据层主要给业务逻辑层提供数据防伪服务，包括对于持久化数据的增、删、改、查。User业务逻辑需要的服务由UserDataService接口提供。由于持久化数据的保存可能存在多种形式：Txt文件、序列化文件、数据库等，所示抽象了数据服务。数据层模块的具体描述如图5.4所示。



图5.4 数据层模块的描述

#### 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 |
| DataService | 持久化数据库的接口，提供集体载入，集体保存、增、删、改、查服务。 |
| DataServiceTxtImpl | 基于TXT文件的持久化数据库的接口，提供集体载入、集体保存、增、删、改、查服务 |
| DataServiceSpringImpl | 基于数据库的持久化数据的接口，提供集体载入、集体保存、增、删、改、查服务 |

#### 5.4.2 数据层模块的接口规范

表5.4.2-1数据层模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
|  | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |

## 6、信息视角

### 6.1 数据持久化对象

系统的PO类就是对应的相关的实体类，在此只做简单的介绍，如表6.1所示：

表6.1 PO类简单介绍

|  |  |
| --- | --- |
| 类名 | 包含的属性 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

持久化用户对象如UserPO的定义如图6.1，更多定义见原型代码

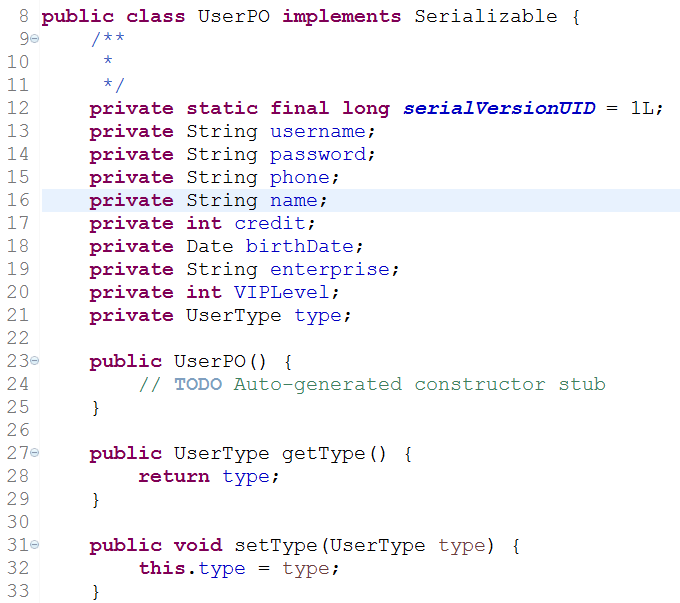


图6.1 持久化用户对象UserPO的定义

### 6.2 数据库表

|  |  |  |
| --- | --- | --- |
| 表名 | 内容 | 字段设计 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |