进销存系统（JXCS）

软件体系结构描述文档

**By：Room404**

**队长：金鑫**

**队员：周宇宸 梁思宇 李斐**

目录

[1.引言 1](#_Toc402121574)

[1.1编制目的 2](#_Toc402121575)

[1.2词汇表 2](#_Toc402121576)

[1.3参考资料 2](#_Toc402121577)

[2.产品概述 2](#_Toc402121578)

[3.逻辑视角 2](#_Toc402121579)

[4.组合视角 4](#_Toc402121580)

[4.1开发包图 4](#_Toc402121581)

[4.2运行时进程 7](#_Toc402121582)

[5.接口视角 8](#_Toc402121583)

[5.1模块的指责 8](#_Toc402121584)

[5.2 用户界面层的分解 10](#_Toc402121585)

[5.2.1用户界面层模块的职责 12](#_Toc402121586)

[5.2.2用户界面层模块的接口规范 12](#_Toc402121587)

[5.2.3 用户界面模块设计原理 13](#_Toc402121588)

[5.3业务逻辑层的分解 13](#_Toc402121589)

[5.3.1业务逻辑层模块的职责 13](#_Toc402121590)

[5.3.2业务逻辑层模块的接口规范 14](#_Toc402121591)

[5.4数据层的分解 30](#_Toc402121592)

[5.4.1数据层模块的职责 36](#_Toc402121593)

[5.4.2数据层模块的接口规范 37](#_Toc402121594)

[6.信息视角 44](#_Toc402121595)

[6.1 数据持久化对象 44](#_Toc402121596)

[6.2 Txt持久化格式 46](#_Toc402121597)

# 1.引言

## 1.1编制目的

本报告详细完成对某灯具公司进销存系统的概要设计，达到知道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 | 词汇定义 | 备注 |
| JXCS | 灯具公司进销存系统 | 无 |
| bl | 业务逻辑（business logic） | 无 |

## 1.3参考资料

[IEEE1471-2000];

进销存系统用例文档和进销存系统软件需求规格说明。

# 2.产品概述

参考进销存系统管理系统用例文档和进销存系统软件需求规格说明中对产品的概括描述。

# 3.逻辑视角

进销存系统中，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。

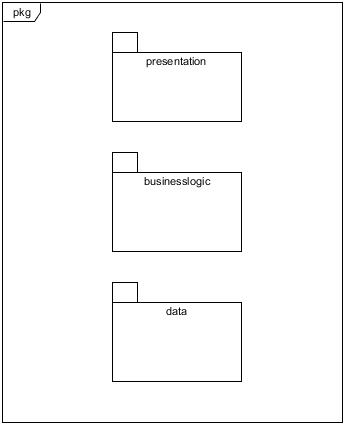


图1 参照体系结构风格的包图表达逻辑视角

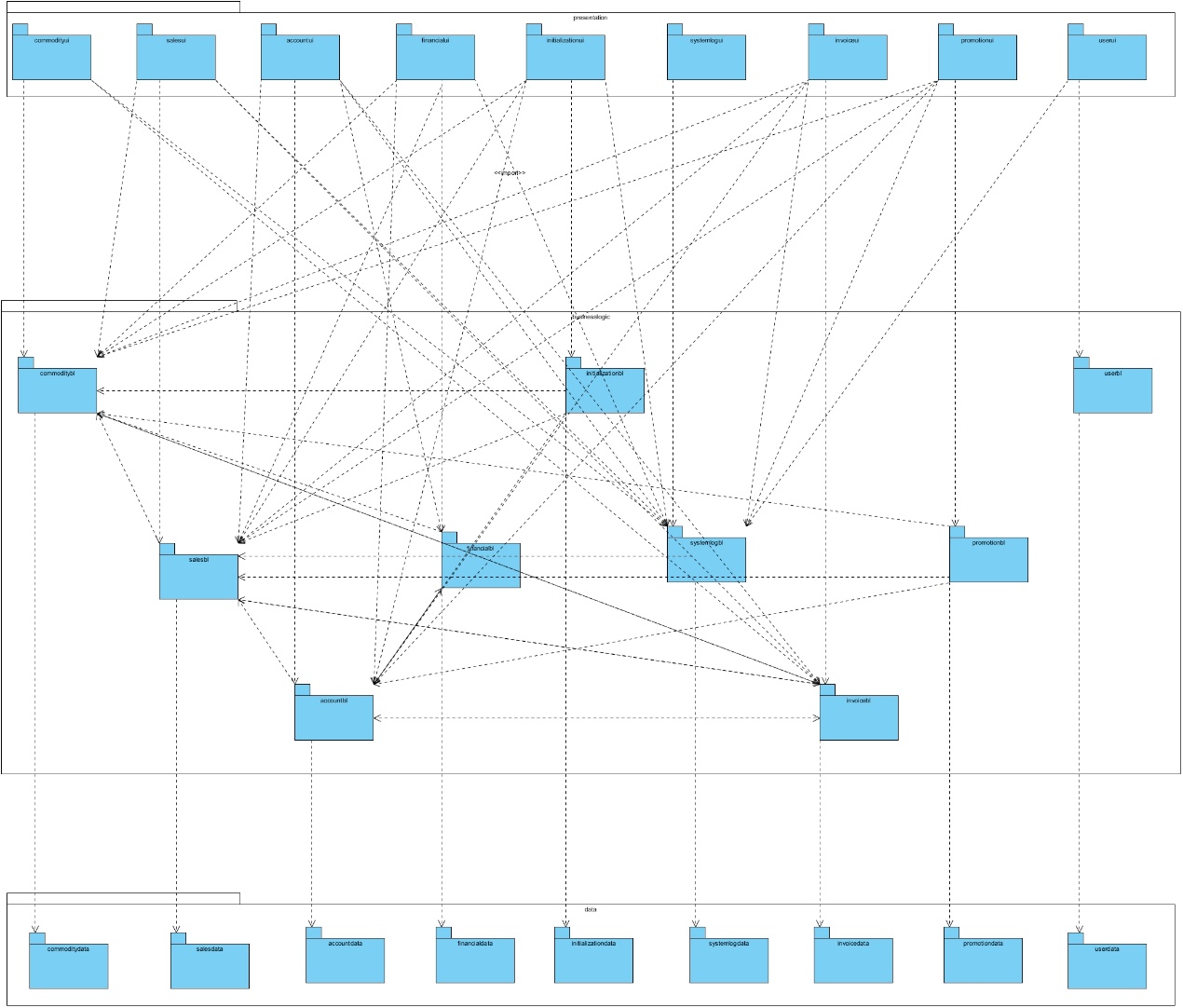


图2 软件体系结构逻辑设计方案

# 4.组合视角

## 4.1开发包图

进销存系统的最终开发包设计如表1所示。

表1 进销存系统的最终开发包设计

|  |  |
| --- | --- |
| **功能** | **对应逻辑包** |
| Mainui | Commodityui, Salesui, Accountui,Managerui, Adminui |
| Commodityui | Commodityblservice,界面类库包,vo |
| Commodityblservice |  |
| Commoditybl | Commodityblservice, Commoditydataservice, po, Invoicebl, Systemlogbl |
| Commoditydataservice | JavaRMI, po |
| Commoditydata | JavaRMI, po, databaseutility |
| Salesui | Salesblservice,界面类库包,vo |
| Salesblservice |  |
| Salesbl | Salesblservice, Salesdataservice, po, Commoditybl, Invoicebl, Systemlogbl |
| Salesdataservice | JavaRMI, po |
| Salesdata | JavaRMI, po, databaseutility |
| Accountui | Accountblservice,界面类库包,vo |
| Accountblservice |  |
| Accountbl | Accountblservice, Accountdataservice, po, Financialbl, Invoicebl, Systemlogbl, Salesbl |
| Accountdataservice | JavaRMI, po |
| Accountdata | JavaRMI, po, databaseutility |
| Financialui | Financialblservice,界面类库包,vo |
| Financialblservice |  |
| Financialbl | Financialblservice, Financialdataservice, po, Accountbl, Systemlogbl, Commoditybl, Salesbl, Salesbl |
| Financialdataservice | JavaRMI, po |
| Financialdata | JavaRMI, po, databaseutility |
| Initializationui | Initializationblservice,界面类库包,vo |
| Initializationblservice |  |
| Initializationbl | Initializationblservice, Initializationdataservice, po, Commoditybl, Accountbl, Systemlogbl, Salesbl |
| Initializationdataservice | JavaRMI, po |
| Initializationdata | JavaRMI, po, databaseutility |
| Systemlogui | Systemlogblservice,界面类库包,vo |
| Systemlogblservice |  |
| Systemlogbl | Systemlogblservice, Systemlogdataservice, po |
| Systemlogdataservice | JavaRMI, po |
| Systemlogdata | JavaRMI, po, databaseutility |
| Invoiceui | Invoiceblservice,界面类库包,vo |
| Invoiceblservice |  |
| Invoicebl | Invoiceblservice, Invoicedataservice, po, Commoditybl, Salesbl, Accountbl, Systemlogbl |
| Invoicedataservice | JavaRMI, po |
| Invoicedata | JavaRMI, po, databaseutility |
| Promotionui | Promotionblservice,界面类库包,vo |
| Promotionblservice |  |
| Promotionbl | Promotionblservice, Promotiondataservice, po, Commoditybl, Salesbl, Systemlogbl, commodityblService |
| Promotiondataservice | JavaRMI, po |
| Promotiondata | JavaRMI, po, databaseutility |
| Userui | Userblservice,界面类库包,vo |
| Userblservice |  |
| Userbl | Userblservice, Userdataservice, po, Systemlogbl |
| Userdataservice | JavaRMI, po |
| Userdata | JavaRMI, po, databaseutility |
| Vo |  |
| Po |  |
| Utilitybl |  |
| 界面类库包 |  |
| Java RMI |  |
| Databaseutility | JDBC |

进销存系统客户端开发包如图3所示，服务器端开发包图如图4所示。

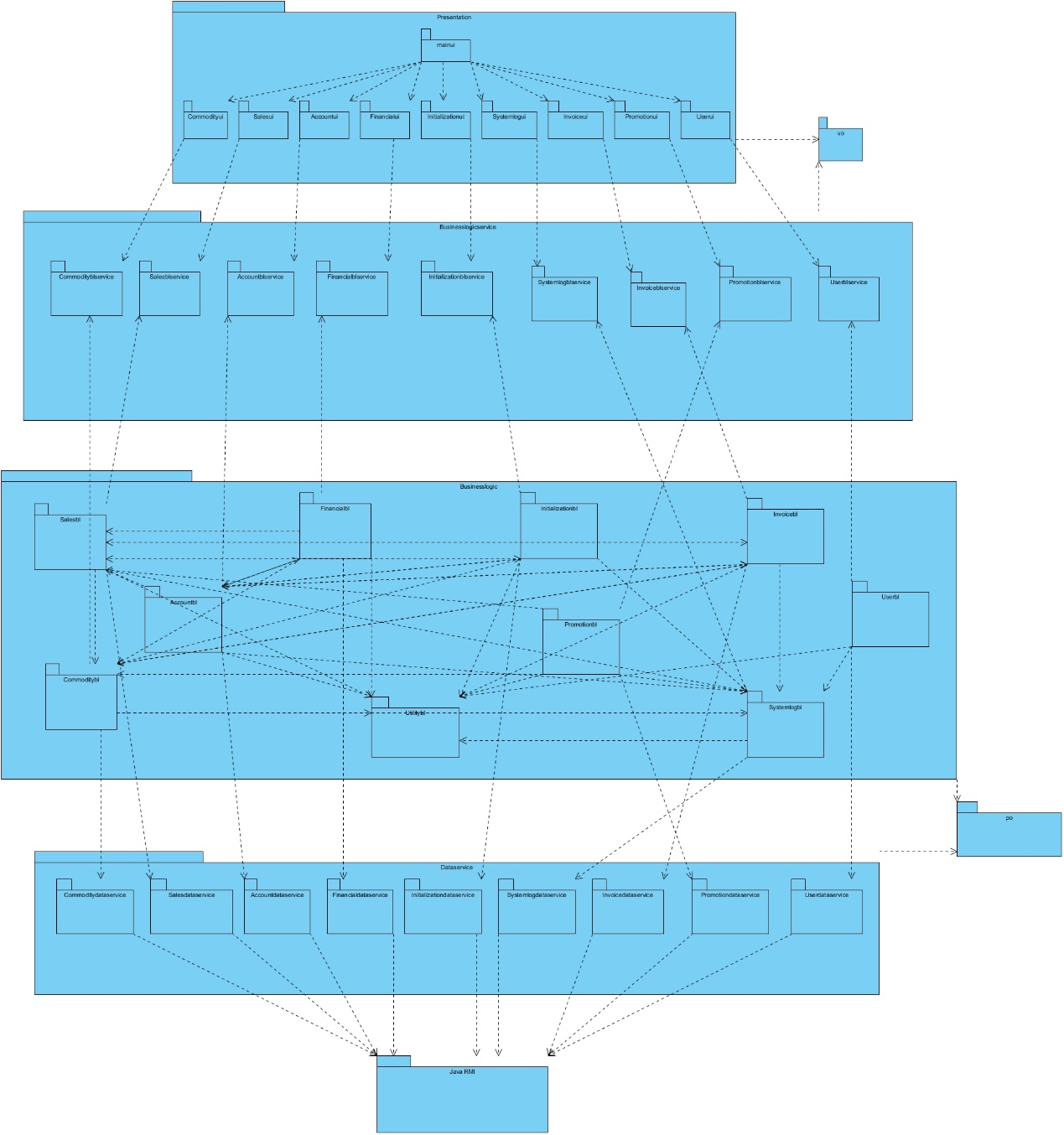


图3 进销存系统客户端开发包图

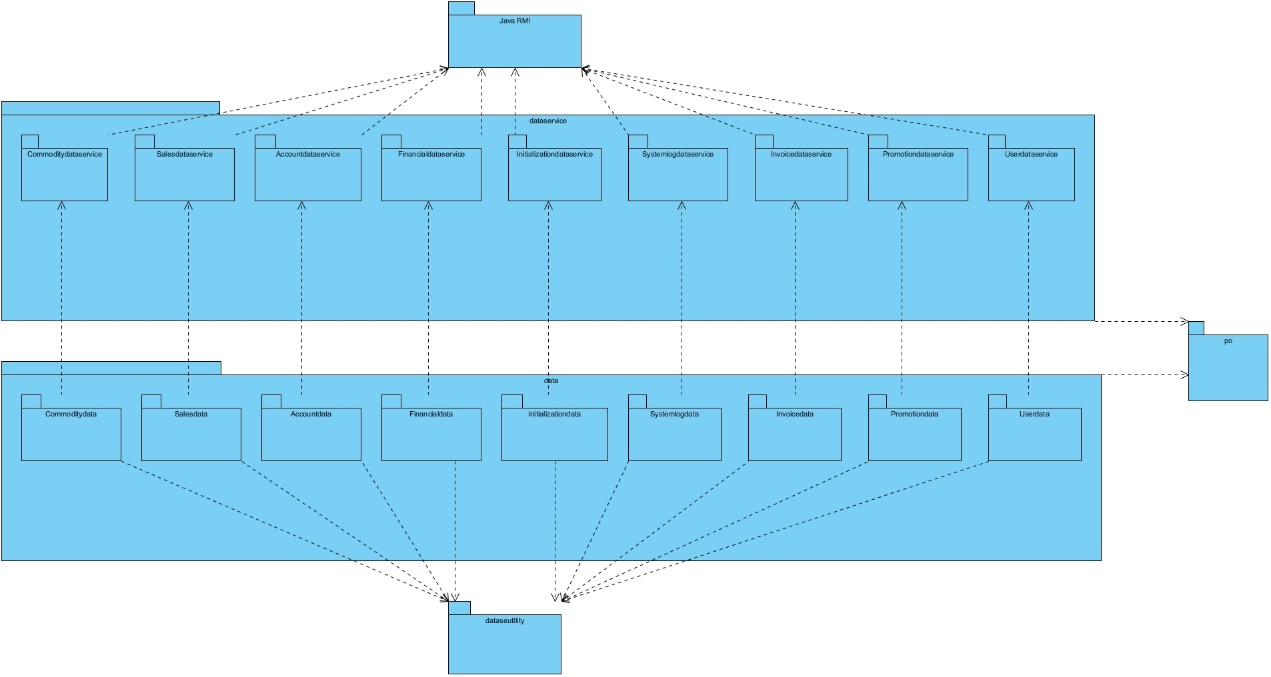


图4 进销存系统服务器端开发包图

## 4.2运行时进程

在进销存系统中，会有多个客户端进程和一个服务器端进程，其进程图如图5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。

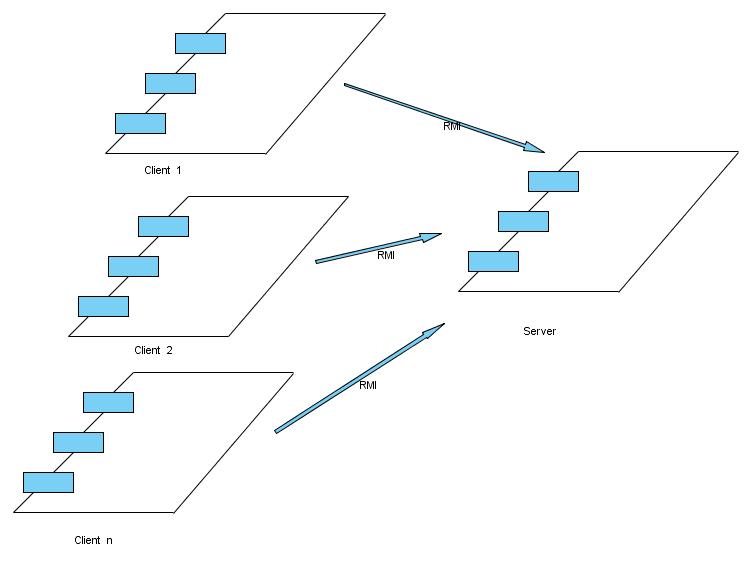


图5 进程图

**4.3物理部署**

进销存系统中客户端构建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6所示。

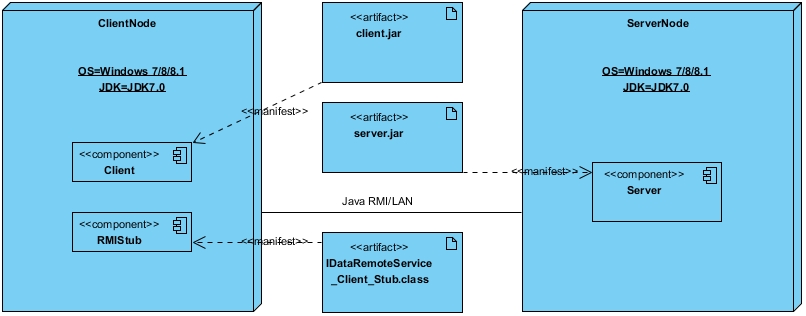


图6 部署图

# 5.接口视角

## 5.1模块的指责

客户端模块和服务器端模块试图分别如图7和图8所示。客户端各层和服务器端各层的职责分别如表2和表3所示。



图7 客户端模块视图



图8 服务器端模块视图

表2 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进行相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3 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 |
| 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4所示。

|  |  |  |
| --- | --- | --- |
| 接口 | 服务调用方 | 服务提供方 |
| CommodityBlService  InvoidceBlService  SalesBlService  AccountBlService  CustomerBlService  FinancialBlService  SystemlogBlService  InitializationBlService  PromotionBlService  UserBlService | 客户端展示层 | 客户端业务逻辑层 |
| CommodityDataService  SalesDataService  AccountDataService  FinancialDataService  InitializationDataService  PromotionDataService  InvoiceDataServoice  SystemlogDataService  UserDataService | 客户端业务逻辑层 | 服务器端数据层 |

借用销售用例来说明层之间的调用，如图9所示。每一层之间都是由上层依赖了一个借口（需接口），而下层实现这个接口（供接口）SalesBLService提供了Sales界面所需要的所有业务逻辑功能。SalesDataService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。

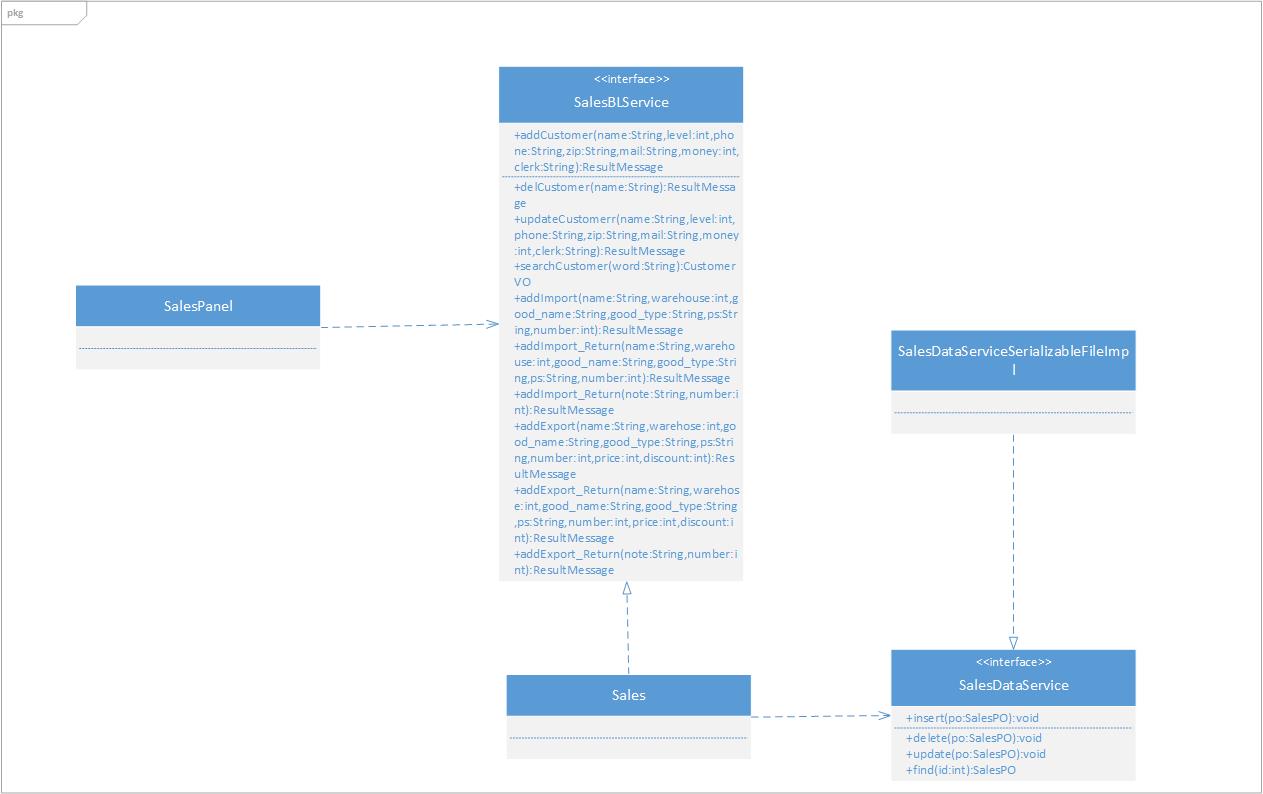


图9 销售用例层之间调用的接口

## 5.2 用户界面层的分解

根据需求，系统存在 个用户界面。界面跳转如图10所示。

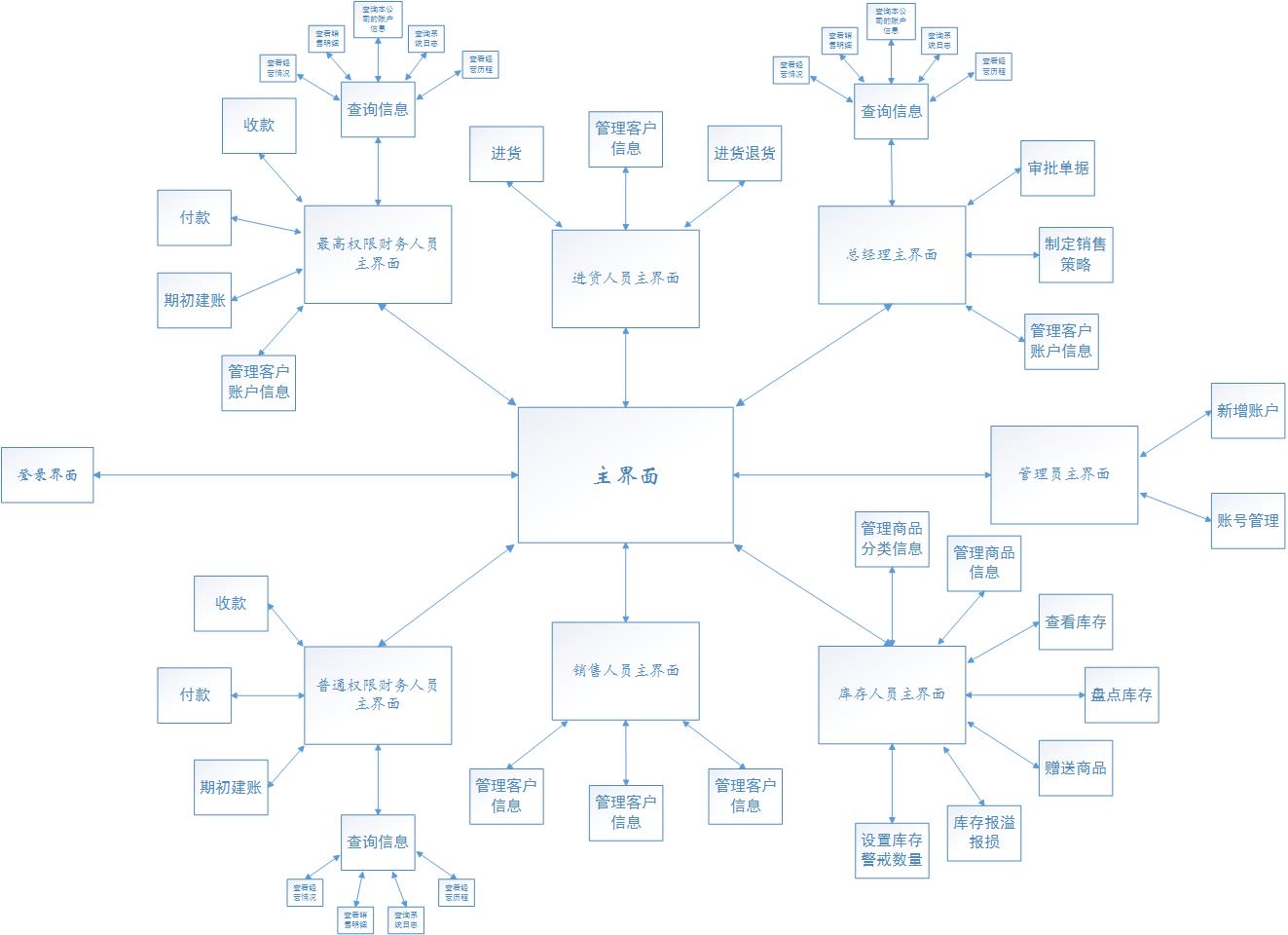


图10 用户界面跳转

服务器端和客户端的用户设计接口是一致的，只是具体的界面不一样。用户界面类如图11所示。



图11 用户界面类

### 5.2.1用户界面层模块的职责

如表5所示为用户界面层的职责

表5 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6所示。

表6 用户界面层模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| MainFrame | 语法 | Init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示Frame,welcomePanel以及loginPanel |

用户界面层需要的服务接口如表7所示。

表7 用户界面层模块需要的服务接口

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| ShowPanel | 语法 | Show(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示Frame与Panel |
| HidePanel | 语法 | Hide(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 隐藏Frame与Panel |
| 需要的服务 | | |
| 服务名 | 服务 | |
| Businesslogicservice.LoginBLService | 登陆界面的业务逻辑接口 | |
| Businesslogicservice.CommodityBLService | 库存界面的业务逻辑接口 | |
| Businesslogcservice.SalesBLService | 销售界面的业务逻辑接口 | |
| Businesslogcservice.AccountBLService | 账户管理界面的业务逻辑接口 | |
| Businesslogcservice.FinancialBLService | 销售表单管理界面的业务逻辑接口 | |
| Businesslogcservice.InitializationBLService | 期初建账界面的业务逻辑接口 | |
| Businesslogcservice.SystemlogBLService | 系统日志界面的业务逻辑接口 | |
| Businesslogcservice.InvoiceBLService | 单据审批界面的业务逻辑接口 | |
| Businesslogcservice.PromotionBLService | 促销界面的业务逻辑接口 | |
| Businesslogcservice.UserBLService | 用户管理界面的业务逻辑接口 | |

### 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；Sales对象负责销售界面的业务逻辑。业务逻辑层的设计如图12所示。

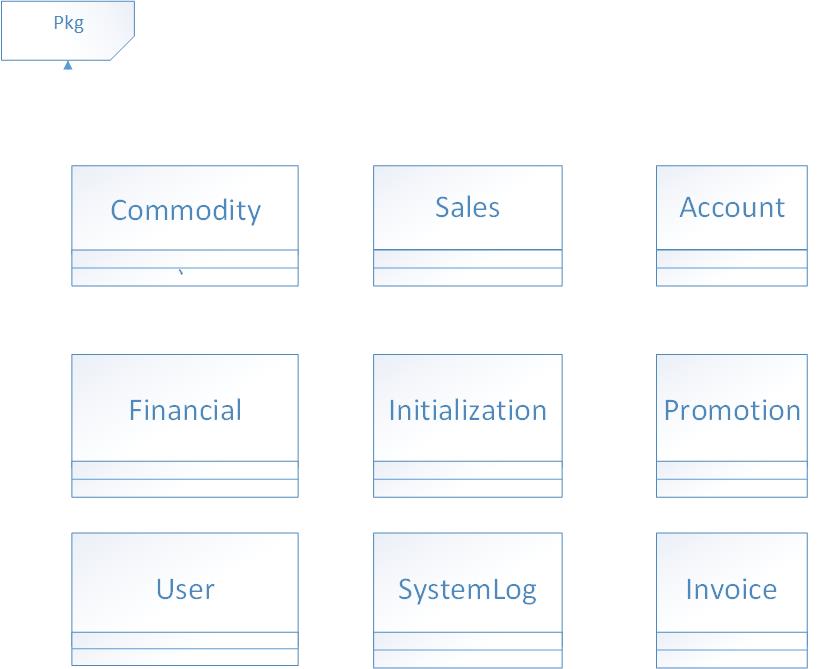


图12 业务逻辑层的设计

### 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8所示。

表8 业务逻辑层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| Commoditybl | 负责实现库存界面所需要的服务 |
| Salesbl | 负责实现销售界面所需要的服务 |
| Accountbl | 负责实现账户界面所需要的服务 |
| Financialbl | 负责实现财务界面所需要的服务 |
| Initializationbl | 负责实现建账初期界面所需要的服务 |
| Promotionbl | 负责实现促销策略界面所需要的服务 |
| Userbl | 负责实现与登陆界面所需要的服务 |
| SystemLogbl | 负责实现系统日志界面所需要的服务 |
| Invoicebl | 负责实现单据审批界面所需要的服务 |

### 5.3.2业务逻辑层模块的接口规范

接口规范如下图所示：

Commoditybl的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| Commodity.addCommodity | 语法 | Public int addCommodity\_up  (CommodityVOvo1,SortVO vo2); |
| 前置条件 | 系统启动添加商品，且输入合法。 |
| 后置条件 | 添加商品，返回是否成功，返回0为成功，否则为失败。 |
| Commodity.delCommodity | 语法 | Public int delCommodity\_up  (CommodityVO vo); |
| 前置条件 | 系统启动删除商品，且输入合法。 |
| 后置条件 | 删除商品，返回是否成功，返回0为成功，否则为失败。 |
| Commodity.updateCommodity | 语法 | public int updateCommodity\_up  (CommodityVO vo1,CommodityVO vo2);//更新商品信息，用vo2的信息替换vo1  public int updateCommodity\_up  (CommodityVO vo1,SortVO vo2);//修改商品所属的分类信息 |
| 前置条件 | 系统启动修改商品信息，且输入合法。 |
| 后置条件 | 更新商品，返回是否更新成功，返回0为成功，否则为失败。 |
| Commodity.searchCommodity | 语法 | //模糊查找商品  Public ArrayList<CommodityVO> searchFuzzyCommodity\_up(String word);  //精确查找商品  Public CommodityVO  searchAccurateCommodity\_up(String name,String type); |
| 前置条件 | 无 |
| 后置条件 | 返回搜索结果，模糊查找返回一个CommodityVO类型的arrayList，精确查找返回查找的商品的CommodityVO，若未查找到相关信息，返回空的arrayList或commodityVO |
| Commodity.addSort | 语法 | public int addSort\_up(SortVO vo1, SortVO vo2);//vo1为子类，vo2为父类 |
| 前置条件 | 系统启动添加分类，并且输入合法 |
| 后置条件 | 添加分类，返回是否添加成功，返回0为成功，否则为失败。 |
| Commodity.delSort | 语法 | public int delSort\_up(SortVO vo); |
| 前置条件 | 系统启动删除分类，且输入合法 |
| 后置条件 | 删除分类，返回是否删除成功，返回0为成功，否则为失败。 |
| Commodity.updateSort | 语法 | //修改分类信息，用vo2的信息替代vo1  Public int updateSort\_up\_Inf(SortVO vo1, SortVO vo2);  //将分类移至另一分类下，将vo1的信息移到vo2下  public int  updateSort\_up\_Mov(SortVO vo1,SortVO vo2); |
| 前置条件 | 系统启动更新分类信息，且输入合法 |
| 后置条件 | 修改分类，返回是否修改成功，返回0为成功，否则为失败 |
| Commodity.Examine | 语法 | public ArrayList<ExamineVO> Examine\_up(String time1, String time2); |
| 前置条件 | 系统启动库存查看，且输入的时间格式正确 |
| 后置条件 | 返回查看结果 |
| Commodity.addGift | 语法 | Public ResultMessage addGift(String name, String type) |
| 前置条件 | 无 |
| 后置条件 | 增加赠品，返回是否成功 |
| Commodity.delGift | 语法 | Public ResultMessage delGift(String name, String type) |
| 前置条件 | 无 |
| 后置条件 | 删除赠品，返回是否成功 |
| Commodity.patch | 语法 | Public ResultMessage patch(String name, String type, int number) |
| 前置条件 | 输入的number为数字 |
| 后置条件 | 生成报溢报损单，返回是否成功 |
| Commodity.warn | 语法 | Public ResultMessage warn(String name, String type, int number) |
| 前置条件 | 输入的number为数字 |
| 后置条件 | 返回是否设置成功 |
| 需要的服务 | | |
| 服务名 | 服务 | |
| CommodityDataService.findGood(String name, String type) | 根据name和type查找精确商品对象 | |
| CommodityDataService.addGood(CommodityPO po1, SortPO po2) | 在分类中添加商品对象 | |
| CommodityDataService.delGood(CommodityPO po) | 删除商品对象 | |
| ComodityDataService.getGoodNote(SortPO po1) | 根据商品所属分类获得商品对象的编号 | |
| CommodityDataService.updateGood(CommodityPO po1, CommodityPO po2) | 更新商品对象信息 | |
| CommodityDataService.updateGood(CommodityPO po1,SortPO po2) | 更新商品对象所属分类 | |
| CommodityDataService.addSort(SortPO po1，SortPO po2) | 添加分类对象 | |
| CommodityDataService.delSort(SortPO po) | 删除分类对象 | |
| CommodityDataService.updateSort(SortPO po1, SortPO po2) | 更新分类对象信息 | |
| CommodityDataService.updateSort\_MOV(SortPO po1,SortPO po2) | 更新分类对象所属分类 | |
| CommodityDataService.getSortNote(SortPO po1) | 根据分类所属父类获得分类编号 | |
| CommodityDataService.addSendGift(SendGiftPO po) | 添加赠品对象 | |
| CommodityDataService.findSendGift(SendGiftPO po) | 查找赠品对象 | |
| CommodityDataService.findSendGift(String note) |  | |
| CommodityDataService.delSendGift(SendGiftPO po) | 删除赠品对象 | |
| CommodityDataService.delSendGift(String note) |  | |
| CommodityDataService.getSendGiftNote() | 获得赠送商品的编号 | |
| CommodityDataService.getAllSendGift() | 获得当前所有赠送商品的信息 | |
| CommodityDataService.getAll() | 获得所有商品对象 | |
| CommodityDataService.getAllSort() | 获得所有商品分类对象 | |
| CommodityDataService.addWarn(WarnPO po) | 添加库存报警单对象 | |
| CommodityDataService.addPatch(PatchPO po) | 添加报溢报损单对象 | |
| CommodityDataService.delPatch(String note) | 根据报溢报损单编号查询到相应单据并删除报溢报损单对象 | |
| CommodityDataService.getPatch(String note) | 根据报溢报损单编号查询到单一报溢报损单对象 | |
| CommodityDataService.getAllPatch() | 获得当前所有报溢报损单对象 | |
| CommodityDataService.getPatchNote() | 获得报溢报损单编号 | |
| CommodityDataService.passPatch(PatchPO po) | 系统根据审批通过的报溢报损单信息修改库存信息 | |
| CommodityDataService.refusePatch(PatchPO po) | 系统根据审批拒绝的报溢报损单信息修改单据信息 | |
| CommodityDataService.passSendGift(SendGiftPO po) | 系统根据审批通过的赠送单信息修改库存信息 | |
| CommodityDataService.refuseSendGift(SendGift po) | 系统根据审批拒绝的报溢报损单信息修改单据信息 | |
| CommodityDataService.passImport(Import po) | 系统根据审批通过的进货单信息修改库存信息 | |
| CommodityDataService.passImport\_Return(Import\_Return po) | 系统根据审批通过的今后退货单修改库存信息 | |
| CommodityDateService.passExport\_Return(Export\_Return po) | 系统根据审批通过的销售退货单修改库存信息 | |
| CommodityDataService.passExport(Export po) | 系统根据审批通过的销售单修改库存信息 | |
| CommodityDataService.clear() | 清空所有对象 | |

Salesbl的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| Sales.addCustomer | 语法 | Public int addCustomer\_up(Customer customerVO) |
| 前置条件 | 系统启动添加客户，且输入合法。 |
| 后置条件 | 返回提示是否成功添加客户，返回为成功，否则为失败。 |
| Sales.delCustomer | 语法 | Public int delCustomer\_up(CustomerVO customerVO) |
| 前置条件 | 系统启动删除客户，切输入合法 |
| 后置条件 | 返回提示是否成功删除客户，返回0为成功，否则为失败。 |
| Sales.updateCustomer | 语法 | Public int updateCustomer\_up(CustomerVO vo1,CustomerVO vo2) |
| 前置条件 | 有客户的信息发生改变，需要更新,系统启动更新客户信息，且输入合法 |
| 后置条件 | 返回提示是否成功更新客户信息，返回0为成功，否则为失败。 |
| Sales.searchCustomer | 语法 | //模糊查找客户  Public ArrayList<CustomerVO> searchFuzzyCustomer\_up(String word)  //根据客户姓名精确查找客户  Public CustomerVO searchExactCustomer\_up(String name) |
| 前置条件 | 系统启动查找客户 |
| 后置条件 | 返回搜索结果，模糊查找返回一个CustomerVO类型的arrayList，精确查找返回查找的客户的CustomerVO，若未查找到相关信息，返回空的arrayList或customerVO |
| Sales.addImport | 语法 | Public int addImport\_up  (ImportMenuVO importMenuVO) |
| 前置条件 | 商品需要进货时，系统启动添加进货单 |
| 后置条件 | 系统返回添加进货单是否成功，成功返回0交由总经理审批，否则为失败。 |
| Sales.addImport\_Return | 语法 | Public int addImport\_Return\_up  (ImportMenuVO importMenuVO) |
| 前置条件 | 当已经进货某商品但是要进行退货时，系统启动进货退货，且输入合法 |
| 后置条件 | 系统返回添加进货退货单是否成功，成功返回0交由总经理审批，否则为失败。 |
| Sales.addExport | 语法 | Public int addExport\_up  (ExportMenuVO exportMenuVO) |
| 前置条件 | 发生销售时,系统启动添加销售单，且输入合法 |
| 后置条件 | 系统返回添加销售单是否成功，成功返回0交由总经理审批，否则为失败。 |
| Sales.addExport\_Return | 语法 | public int addExport\_Return\_up  (ExportMenuVO exportMenuVO); |
| 前置条件 | 当已经销售某商品后，但是发生销售退货时，系统启动添加销售退货单，且输入合法 |
| 后置条件 | 系统返回添加销售退货单是否成功，成功返回0交由总经理审批，否则为失败。 |
| Sales.getImport\_Note | 语法 | public String getImportNote\_up(); |
| 前置条件 | 当发生进货时，系统启动添加进货单 |
| 后置条件 | 系统返回要新添加的进货单的编号。 |
| Sales.getImport\_Return\_Note | 语法 | public String  getImport\_ReturnNote\_up(); |
| 前置条件 | 当发生进货退货时，系统启动添加进货退货单 |
| 后置条件 | 系统返回要新添加的进货退货单的编号。 |
| Sales.getExport\_Return\_Note | 语法 | public String  getExport\_ReturnNote\_up(); |
| 前置条件 | 当发生销售退货时，系统启动添加销售退货单 |
| 后置条件 | 系统返回要新添加的销售退货单的编号。 |
| Sales.getExport\_Note | 语法 | public String getExportNote\_up(); |
| 前置条件 | 当发生销售时，系统启动添加销售单 |
| 后置条件 | 系统返回要新添加的销售单的编号。 |
| Sales.searchNote | 语法 | // 根据编号查找进货单  public ImportMenuVO searchImportNote\_up(String note);  // 根据编号查找销售单  public ExportMenuVO searchExportNote\_up(String note);  // 根据编号查找进货退货单  public ImportMenuVO searchImport\_ReturnNote\_up(String note);    // 根据编号查找销售退货单  public ExportMenuVO searchExport\_ReturnNote\_up(String note); |
| 前置条件 | 需要通过单据编号查询单据具体信息 |
| 后置条件 | 系统根据编号查询到单据具体信息，并返回单据具体信息 |
| Sales.getCustomer | 语法 | // 获得所有供货商信息  public ArrayList<CustomerVO> getAllImportCustomer\_up();  // 获得所有销售客户信息  public ArrayList<CustomerVO> getAllExportCustomer\_up(); |
| 前置条件 | 无 |
| 后置条件 | 系统返回当前所有供货商或销售商的信息 |
| Sales.getReturnMaxNumber | 语法 | // 根据进货编号查找进货可退货最大数量  public int getImport\_ReturnMaxNumber\_up(String old\_note);  // 根据销售编号查找销售可退货最大数量  public int getExport\_ReturnMaxNumber\_up(String old\_note); |
| 前置条件 | 发生销售退货，或进货退货时 |
| 后置条件 | 系统根据编号查找并返回可以退货的最大数量 |
| Sales.getAllCommodity | 语法 | public ArrayList<CommodityVO> getAllCommodity\_up(); |
| 前置条件 | 发生销售，销售退货，进货，进货退货时 |
| 后置条件 | 系统返回当前所有商品的商品信息 |
| Sales.getCommodity | 语法 | public CommodityVO getCommodity\_up(String name, String type); |
| 前置条件 | 发生销售，销售退货，进货，进货退货时 |
| 后置条件 | 系统根据商品名称和型号查找并返回当前所有商品的商品信息 |
| Sales.getClerkDiscount | 语法 | public int getClerkDiscount\_up(); |
| 前置条件 | 销售人员或销售经理给客户折扣时 |
| 后置条件 | 系统返回当前操作员能修改的最大折扣 |
| Sales.getDiscount | 语法 | public int getDiscount\_up(int money, int level); |
| 前置条件 | 发生销售，且总经理制订了与本销售过程相关的折扣促销，且该销售符合该折扣促销策略。 |
| 后置条件 | 系统根据客户等级和金额查找并返回折扣促销金额金额 |
| Sales.getVoucher | 语法 | public int getVoucher\_up(int money, int level); |
| 前置条件 | 发生销售，且总经理制订了与本销售过程相关的代金券促销，且该销售符合代金券促销策略 |
| 后置条件 | 系统根据客户等级和销售金额查找并返回代金券促销策略金额 |
| Sales.getProGift | 语法 | public CommodityVO getProGift\_up(int money, int level); |
| 前置条件 | 发生销售，且总经理制订了与本销售过程相关的赠品促销，且该销售符合赠品促销策略 |
| 后置条件 | 系统根据客户等级和销售金额查找并返回赠品信息 |
| Sales.getCustomerNote | 语法 | Public String getCustomerNote\_up() |
| 前置条件 | 添加客户时，需要系统自动分配编号 |
| 后置条件 | 系统返回分配的编号 |
| Sales.addDraftBill | 语法 | //添加草稿进货单  Public int addDraftImport\_up (ImportMenuVO importMenuVO)  //添加草稿进货退货单  Public int addDraftImport\_Return\_up (ImportMenuVO importMenuVO)  //添加草稿销售单  Public int addDraftExport\_up (ExportMenuVO exportMenuVO)  //添加草稿销售退货单  Public int addDraftExport\_Return\_up (ExportMenuVO exportMenuVO) |
| 前置条件 | 系统需要添加草稿状态的进货单/销售单/进货退货单/销售退货单 |
| 后置条件 | 系统返回是否添加成功，0代表成功，否则代表不成功 |
| Sales.getDraftBill | 语法 | //获得所有的草稿进货单  Public ArrayList<ImportMenuVO> getAllDraftImport\_up()  //获得所有的草稿进货退货单  public ArrayList<ImportMenuVO> getAllDraftImport\_Return\_up();  // 获得所有的草稿销售单  public ArrayList<ExportMenuVO> getAllDraftExport\_up ();  // 获得所有的草稿销售退货单  public ArrayList<ExportMenuVO> getAllDraftExport\_Return\_up(); |
| 前置条件 | 销售过程中，销售人员要修改草稿单 |
| 后置条件 | 系统返回全部草稿单据 |
| Sales.searchDraftBill | 语法 | // 根据编号查找草稿进货单  public ImportMenuVO searchDraftImport\_up(String note);  // 根据编号查找草稿进货退货单  public ImportMenuVO searchDraftImport\_Return\_up(String note);  // 根据编号查找草稿销售退货单  public ExportMenuVO searchDraftExport\_Return\_up(String note);  // 根据编号查找草稿销售单  public ExportMenuVO searchDraftExport\_up(String note); |
| 前置条件 | 销售人员根据编号查找草稿单 |
| 后置条件 | 系统根据单据编号返回相应的草稿单据 |
| 需要的服务 | | |
| SalesDataService.addCustomer(CustomerPO po) | 添加客户CustomerPO对象 | |
| SalesDataService.delCustomer(CustomerPO po) | 删除客户CustomerPO对象 | |
| SalesDataService.updateCustomer(CustomerPO po1, CustomerPO po2) | 更新客户CustomerPO对象信息 | |
| SalesDataService.findCustomer(String name) | 根据name查找客户CustomerPO对象 | |
| SalesDataService.getCustomerNote() | 获得要新增客户编号 | |
| SalesDataService.addImport(ImportPO po) | 添加进货单ImportPO对象 | |
| SalesDataService.addImport\_Return(Import\_ReturnPO po) | 添加进货退货单Import\_ReturnPO对象 | |
| SalesDataService.addExport(ExportPO po) | 添加销售单ExportPO对象 | |
| SalesDataService.addExport\_Return(Export\_ReturnPO po) | 添加销售退货单Export\_ReturnPO对象 | |
| SalesDataService.getAllCustomer() | 获得所有的客户CustomerPO对象 | |
| SalesDataService.getAllImport() | 获得所有的进货单ImportPO对象 | |
| SalesDataService.getAllImport\_Return() | 获得所有的进货退货单Import\_ReturnPO对象 | |
| SalesDataService.getAllExport() | 获得所有的销售单ExportPO对象 | |
| SalesDataService.getAllExport\_Return() | 获得所有的销售退货单Export\_ReturnPO对象 | |
| SalesDataService.getImportNote() | 获得要新增的进货单的单据编号 | |
| SalesDataService.getImport\_ReturnNote() | 获得要新增的进货退货单的单据编号 | |
| SalesDataService.getExportNote() | 获得要新增的销售单的单据编号 | |
| SalesDataService.getExport\_ReturnNote() | 获得要新增的销售退货单的单据编号 | |
| SalesDataService.getImport(String note) | 根据单据编号进行查找单一的进货单对象 | |
| SalesDataService.getExport(String note) | 根据单据编号进行查找单一的销售单对象 | |
| SalesDataService.getImport\_Return(String note) | 根据单据编号进行查找单一的进货退货单对象 | |
| SalesDataService.getExport\_Return(String note) | 根据单据编号进行查找单一的销售退货单对象 | |
| SaleDataService.getAllImportCustomer() | 获得所有进货商 | |
| SaleDataService.getAllExportCustomer() | 获得所有销售商 | |
| SaleDataService.addDraftImport(ImportPO po) | 添加草稿进货单 | |
| SaleDataService .addDraftImport\_Return(Import\_ReturnPO po) | 添加草稿进货退货单 | |
| SaleDataService. addDraftExport(ExportPO po) | 添加草稿销售单 | |
| SaleDataService. addDraftExport\_Return(Export\_ReturnPO po) | 添加草稿销售退货单 | |
| SaleDataService.delDraftImport(String note) | 删除草稿进货单 | |
| SaleDataService.delDraftImport\_Return(String note) | 删除草稿进货退货单 | |
| SaleDataService.delDraftExport(String note) | 删除草稿销售单 | |
| SaleDataService.delDraftExport\_Reutrn(String note) | 删除草稿销售退货单 | |
| SaleDataService. getAllDraftImport() | 获得草稿进货单 | |
| SaleDataService.getAllDraftImport\_Return() | 获得草稿进货退货单 | |
| SaleDataService. getAllDraftExport() | 获得草稿销售单 | |
| SaleDataService.getAllDraftExport\_Return() | 获得草稿销售退货单 | |
| SaleDataService.getDraftImport(String note) | 根据单据编号查找草稿进货单 | |
| SaleDataService.getDraftImport\_Return(String note) | 根据单据编号查找草稿进货退货单 | |
| SaleDataService.getDraftExport(String note) | 根据单据编号查找草稿销售单 | |
| SaleDataService.getDraftExport\_Return(String note) | 根据单据编号查找草稿销售退货单 | |
| SaleDataService.getImport\_ReturnMaxNumber(String old\_note) | 根据单据编号获得最大进货退货数量 | |
| SalesDataService.getExport\_ReturnMaxNumber(String old\_note) | 根据单据编号获得最大销售退货数量 | |
| SalesDataService.clear() | 清除所有对象 | |

Accountbl的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| Account.checkAccount() | 语法 | Public AccountVO checkAccount\_up() |
| 前置条件 | 管理人员为总经理或财务经理 |
| 后置条件 | 返回账户信息 |
| Account.addAccount | 语法 | Public int addAccount\_up(AccountVO vo) |
| 前置条件 | 需要添加账户时，管理人员为总经理或财务经理 |
| 后置条件 | 返回是否添加成功，返回0为成功，否则为失败 |
| Account.delAccount | 语法 | Public int delAccount\_up(AccountVO vo) |
| 前置条件 | 需要删除账户时，管理人员为总经理或财务经理 |
| 后置条件 | 返回是否删除账户成功，返回0为成功，否则为失败 |
| Account.updateAccount | 语法 | Public int updateAccount\_up(AccountVO vo1,AccountVO vo2) |
| 前置条件 | 需要更新账户信息时，管理人员为总经理或财务经理 |
| 后置条件 | 完成账户信息修改，返回是否成功修改,返回1为成功，否则为失败。 |
| Account.searchAccount | 语法 | //模糊查找账户  public ArrayList<AccountVO> searchFuzzyAccount\_up(String word);    //精确查找账户  public AccountVO searchAccurateAccount\_up(String name); |
| 前置条件 | 操作员是总经理或财务经理 |
| 后置条件 | 返回搜索结果，模糊查找返回一个AccountVO类型的arrayList，精确查找返回查找的账户的VO，若未查找到相关信息，返回空的arrayList或AccountVO |
| Account.addReceipt | 语法 | public int addReceipt\_up(GetVO vo); |
| 前置条件 | 收到货款时，系统启动添加收款单，且输入的信息合法 |
| 后置条件 | 完成收款单，系统返回是否成功，成功返回0并交由总经理审批，否则失败。 |
| Account.addPayment | 语法 | public int addPayment\_up(PayVO vo); |
| 前置条件 | 完成付款后，系统启动添加付款单，且输入的信息合法 |
| 后置条件 | 完成付款单，系统返回是否成功，成功返回0并交由总经理审批，否则失败。 |
| Account.getAllReceipt | 语法 | Public ArrayList<GetVO> getAllReceipt\_up(); |
| 前置条件 | 无 |
| 后置条件 | 系统返回当前所有收款单 |
| Account.getAllPayment | 语法 | public ArrayList<PayVO> getAllPayment\_up(); |
| 前置条件 | 无 |
| 后置条件 | 系统返回当前所有的付款单 |
| Account.searchReceipt | 语法 | public GetVO searchReceipt\_up(String note); |
| 前置条件 | 无 |
| 后置条件 | 系统根据收款单单据编号查找并返回收款单具体信息 |
| Account.searchPayment | 语法 | public PayVO searchPayment\_up(String note); |
| 前置条件 | 无 |
| 后置条件 | 系统根据付款单单据编号查找并返回付款单具体信息 |
| Account.getReceiptNote | 语法 | public String getReceiptNote\_up(); |
| 前置条件 | 需要添加收款单时，系统启动添加收款单 |
| 后置条件 | 系统返回要添加的新的收款单的编号 |
| Account.getPaymentNote | 语法 | public String getPaymentNote\_up(); |
| 前置条件 | 需要添加付款单时，系统启动添加付款单 |
| 后置条件 | 系统返回要添加的新的付款单的编号 |
| Account.getOperator | 语法 | public String getOperator\_up(); |
| 前置条件 | 无 |
| 后置条件 | 系统返回当前登录人员姓名 |
| Account.getAllCustomer | 语法 | Public ArrayList<CustomerVO> getAllCustomer\_up(); |
| 前置条件 | 无 |
| 后置条件 | 系统返回当前所有客户的信息 |
| Account.getAllAccount | 语法 | public ArrayList<AccountVO> getAllAccount\_up(); |
| 前置条件 | 操作员为总经理或者财务经理 |
| 后置条件 | 系统返回当前所有账户的信息 |
| 需要的服务 | | |
| AccountDataService.getAccount() | | 获得当前账户信息 |
| AccountDataService.addAccount(AccountPO po) | | 添加账户AccountPO对象 |
| AccountDataService.delAccount(AccountPO po) | | 删除账户AccountPO对象 |
| AccountDataService.updateAccount(AccountPO po1, AccountPO po2) | | 更新账户AccountPO对象 |
| AccountDataService.addMoney(AccountPO po,double money) | | 增加账户AccountPO对象po中的余额 |
| AccountDataService.delMoney(AccountPO po,double money) | | 减少账户AccountPO对象po中的余额 |
| AccountDataService.findAccount(String name) | | 根据name查找账户对象AccountPO po |
| AccountDataService.getAllAccount() | | 获得所有的账户对象AccountPO po |
| AccountDataService.addReceipt(ReceiptPO po) | | 添加收款单ReceiptPO对象ReceiptPO po |
| AccountDataService.addPayment(PaymentPO po) | | 添加付款单PaymentPO对象PaymentPO po |
| AccountDataService.findReceipt(String note) | | 根据收款单编号查找获得单一ReceiptPO po对象 |
| AccountDataService.findPayment(String note) | | 根据付款单编号查找获得单一PaymentPO po对象 |
| AccountDataService.getAllReceipt() | | 获得所有的收款单对象ReceiptPO po |
| AccountDataService.getAllPayment() | | 获得所有的付款单对象PaymentPO po |
| AccountDataService.getReceiptNote() | | 获得要新增的收款单的单据编号 |
| AccountDataService.getPaymentNote() | | 获得要新增的付款单的单据编号 |
| AccountDataService. passReceipt(ReceiptPO po) | | 新增审批通过的收款单对象ReceiptPO po |
| AccountDataService.passPayment(PaymentPO po) | | 新增审批通过的付款单对象PaymentPO po |
| AccountDataService.refuseReceipt(ReceiptPO po) | | 新增审批拒绝的首款单对象ReceiptPO po |
| AccountDAtaService.refusePayment(PaymentPO po) | | 新增审批拒绝的付款单对象PaymentPO po |
| AccountDataService.clear | | 清除所有对象 |

Financialbl的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| Financial.saleList | 语法 | public ArrayList<SalesDetailVO> saleList\_up(String time1, String time2, String good\_name,String good\_type, String customer\_name, String clerk, String warehouse); |
| 前置条件 | 操作员是经过验证的财务人员或总经理，输入信息合法 |
| 后置条件 | 查询销售明细，返回符合条件的销售明细表 |
| Financial.allBill | 语法 | public ArrayList<AllBillVO> allBill\_up  (String time1, String time2, String note\_type,String customer\_name, String clerk, String warehouse); |
| 前置条件 | 操作员是经过验证的财务人员，输入信息合法 |
| 后置条件 | 查询经营历程表，返回符合条件的经营历程表。 |
| Financial.operatingCondition | 语法 | public ArrayList<ConditionVO> operatingCondition\_up(String start\_time, String end\_time); |
| 前置条件 | 操作员是经过验证的财务人员，输入信息合法 |
| 后置条件 | 查看经营情况表，返回符合条件的经营情况表 |
| Financial.getOperator | 语法 | public String getOperator\_up(); |
| 前置条件 | 无 |
| 后置条件 | 返回当前登陆人员的姓名 |
| 需要的服务 | | |
| FinancialDataService.getSaleList() | 获得所有销售明细对象SalesListPO | |
| FinancialDataService.getAllBill() | 获得所有经营历程对象AllBillPO | |
| FinancialDataService.getOperatingCondition() | 获得所有经营情况对象OperatingConditionPO | |
| FinancialDataService.addSaleList(SaleListPO po) | 添加销售明细的单据对象SaleListPO po | |
| FinancialDataService.addAllBill(AllBillPO po) | 添加经营历程的单据对象AllBillPO po | |
| FinancialDataService.addOperatingCondition(OperatingConditionPO po) | 添加经营情况的单据对象OperatingConditionPO po | |
| FinancialDataService.clear() | 清空所有对象 | |

Initializationbl的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| Initialization.newSystem | 语法 | Public ResultMessage newSystem() |
| 前置条件 | 操作人员输入完成期初建账的信息，包括商品，客户，账户 |
| 后置条件 | 旧系统内容将被删除，新系统替换旧系统 |
| Initialization.addCommodity | 语法 | Public ResultMessage addCommodity(String name, String type, int in\_price, int out\_price) |
| 前置条件 | 操作人员为财务人员 |
| 后置条件 | 记录此商品信息，等待替换库存内商品 |
| Initialization.addCustomer | 语法 | Public ResultMessage addCustomer(String name, String phone) |
| 前置条件 | 操作人员为财务人员 |
| 后置条件 | 记录此客户信息，等待替换客户信息 |
| Initialization.addAccount | 语法 | Public ResultMessage addAccount(String name, int money) |
| 前置条件 | 操作人员为财务人员 |
| 后置条件 | 记录此账户信息，等待替换账户信息 |
| Initialization.showInformation | 语法 | Public ResultMessage showInformation() |
| 前置条件 | 操作人员为财务人员 |
| 后置条件 | 返回期初信息 |
| 需要的服务 | | |
| InitializationDataService.getInformation | 查看InitializationPO对象的信息 | |

Promotionbl的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务 | | | |
| Promotion.makeDiscount | 语法 | | public int makeDiscount\_up(DiscountVO vo); |
| 前置条件 | | 操作用户为总经理,系统启动新增折扣促销策略 |
| 后置条件 | | 制作好折价优惠之后，立即更新至销售 |
| Promotion.makeVoucher | 语法 | | public int makeVoucher\_up(VoucherVO vo); |
| 前置条件 | | 操作用户为总经理,系统启动新增代金券促销策略 |
| 后置条件 | | 制作好代金券优惠之后，立即更新至销售 |
| Promotion.makeGift | 语法 | | public int makeGift\_up(ProGiftVO vo); |
| 前置条件 | | 操作用户为总经理，系统启动新增赠品促销策略 |
| 后置条件 | | 制作好赠品优惠之后，立即更新至销售 |
| Promotion.delDiscount | 语法 | | public int delDiscount\_up(int level); |
| 前置条件 | | 操作用户为总经理,系统启动删除折扣促销策略 |
| 后置条件 | | 系统删除折扣优惠策略，立即更新至销售 |
| Promotion.delVoucher | 语法 | | public int delVoucher\_up(int level); |
| 前置条件 | | 操作用户为总经理,系统启动删除代金券促销策略 |
| 后置条件 | | 系统删除代金券优惠，立即更新至销售 |
| Promotion.delGift | 语法 | | public int delGift\_up(int level); |
| 前置条件 | | 操作用户为总经理，系统启动删除赠品促销策略 |
| 后置条件 | | 系统删除赠品优惠，立即更新至销售 |
| Promotion.getDiscount | 语法 | | Public ArrayList<DiscountVO> getDiscount\_up(); |
| 前置条件 | | 操作用户为总经理 |
| 后置条件 | | 返回所有折扣促销策略信息 |
| Promotion.getProGift | 语法 | | public ArrayList<ProGiftVO> getProGift\_up(); |
| 前置条件 | | 操作用户为总经理 |
| 后置条件 | | 返回所有赠品促销信息 |
| Promotion.getVoucher | 语法 | | public ArrayList<VoucherVO> getVoucher\_up(); |
| 前置条件 | | 操作用户为总经理 |
| 后置条件 | | 返回所有代金券促销信息 |
| 需要的服务 | | | |
| PromotionDataService.addDiscount(DiscountPO po) | | 添加折价促销DiscountPO方案对象 | |
| PromotionDataService.addGift(ProGiftPO po) | | 添加赠品促销ProGiftPO方案对象 | |
| PromotionDataService.addVoucher(VoucherPO po) | | 添加代金券促销VoucherPO方案对象 | |
| PromotionDataService.delDiscount(int level) | | 根据用户等级查找到折扣促销策略DiscountPO方案对象后删除 | |
| PromotionDataService.delGift(int level) | | 根据用户等级查找到赠品促销策略  ProGiftPO方案对象后删除 | |
| PromationDataService.delVoucher(int level) | | 根据用户等级查找到代金券促销策略  VoucherPO方案对象后删除 | |
| PromotionDataService.getDiscount(int level) | | 根据用户等级获得折扣促销策略DiscountPO | |
| PromotionDataService.getGift(int level) | | 根据用户等级获得赠品促销策略ProGiftPO | |
| PromotionDataService.getVoucher(int level) | | 根据用户等级获得代金券促销策略VoucherPO | |
| PromotionDataservice.showDiscount() | | 获得当前折价促销方案对象 | |
| PromotionDataService.showProGift() | | 获得当前赠品促销方案对象 | |
| PromotionDataService.showVoucher() | | 获得当前代金券促销方案对象 | |
| PromotionDataService.clear() | | 清空所有对象 | |

Invoicebl的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务 | | | |
| Invoice.show | 语法 | | public ArrayList<InvoiceVO> show\_up(); |
| 前置条件 | | 操作人员为总经理 |
| 后置条件 | | 返回所有待审批单据信息 |
| Invoice.show\_pass | 语法 | | public ArrayList<InvoiceVO> show\_pass(); |
| 前置条件 | | 操作人员为总经理 |
| 后置条件 | | 返回所有审批通过的单据信息 |
| Invoice.show\_refuse | 语法 | | Public ArrayList<InvoiceVO>  show\_refuse(); |
| 前置条件 | | 操作人员为总经理 |
| 后置条件 | | 返回所有审批拒绝单据的信息 |
| Invoice.pass | 语法 | | public int pass\_up(InvoiceVO vo); |
| 前置条件 | | 操作人员为总经理，库存管理人员，销售人员、财务人员填写好单据 |
| 后置条件 | | 总经理审批通过单据 |
| Invoice.refuse | 语法 | | public int refuse\_up(String note); |
| 前置条件 | | 操作人员为总经理 |
| 后置条件 | | 系统通过单据编号查找到具体单据信息，总经理审批拒绝单据 |
| Invoice.search | 语法 | | public InvoiceVO searchNote\_up(String note); |
| 前置条件 | | 无 |
| 后置条件 | | 系统通过单据编号查找并返回单据具体信息 |
| 需要的服务 | | | |
| InvoiceDataServoice.getInvoice(String note) | | 根据单据编号查找并获得一个单据InvoicePO对象 | |
| InvoiceDataService.addInvoice(InvoicePO po) | | 添加一个待审批单据InvoicePO对象 | |
| InvoiceDataService.passInvoice(InvoicePO po) | | 审批通过一个单据InvoicePO对象 | |
| InvoiceDataService.refuseInvoice(InvoicePO po) | | 审批拒绝一个单据InvoicePO对象 | |
| InvoiceDataService.getAllInvoice() | | 获得所有单据InvoicePO对象 | |
| InvoiceDataService.getInvNote() | | 获得一个审批单据的编号 | |
| InvoiceDataService.clear() | | 清空所有单据 | |

Systemlog的接口规格

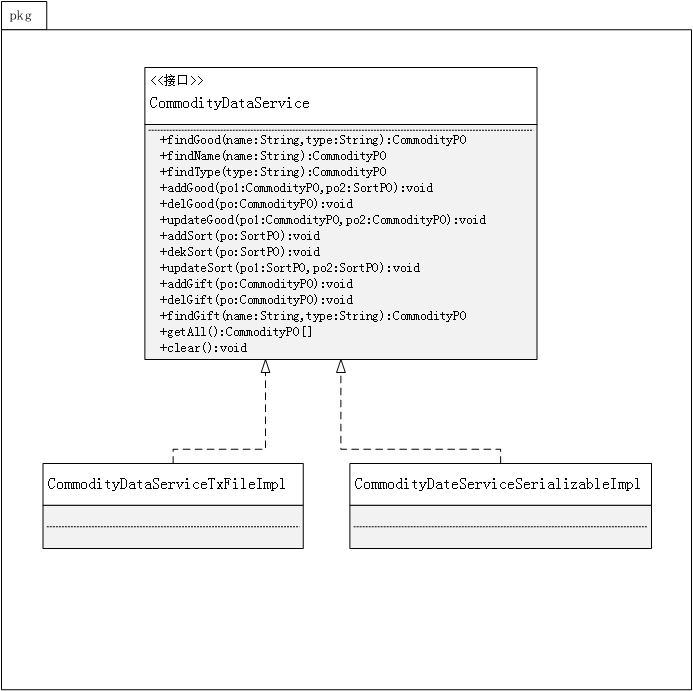
|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| Systemlog.show | 语法 | Public ResultMessage show() |
| 前置条件 | 操作人员为财务人员或总经理 |
| 后置条件 | 显示日志内容，返回是否成功 |
| Systemlog.add | 语法 | Public ResultMessage add(String word) |
| 前置条件 | 其他人员完成操作后，发送操作记录并存储 |
| 后置条件 | 添加日志内容，返回是否成功 |
| 需要的服务 | | |
| SystemlogDataService.get() | 获得日志SystemlogPO对象 | |
| SystemlogDataService.add(SystemlogPO po) | 添加日志SystemlogPO对象 | |
| SystemlogDataService.clear() | 清空日志对象 | |

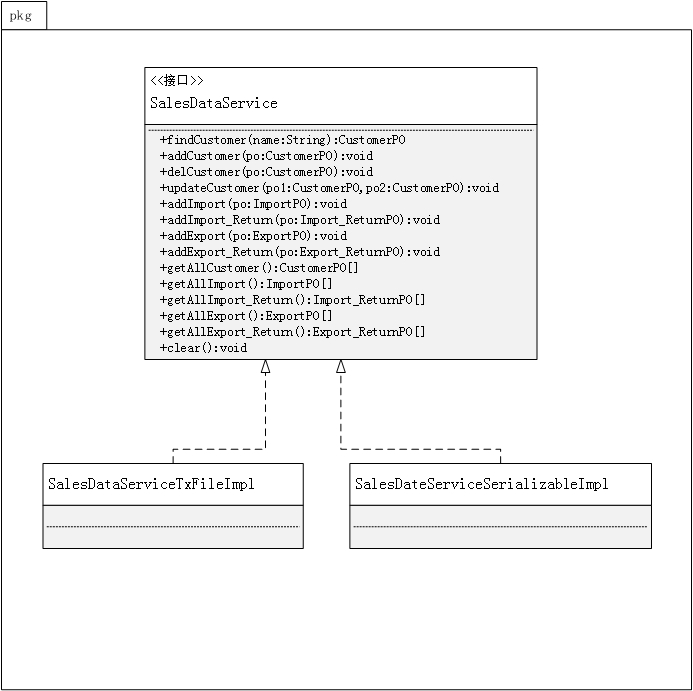
Userbl的接口规范

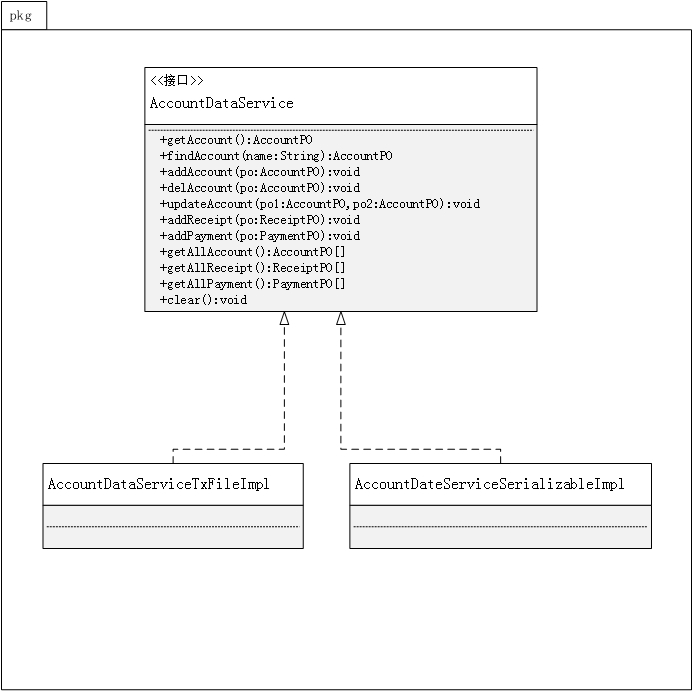
|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| User.addUser | 语法 | public int addUser\_up(UserVO vo); |
| 前置条件 | 管理员身份登陆系统 |
| 后置条件 | 完成新账号的添加，可以开始使用 |
| User.delUser | 语法 | public int delUser\_up(UserVO vo); |
| 前置条件 | 管理员身份登陆系统 |
| 后置条件 | 删除账号信息，注销用户 |
| User.show | 语法 | public ArrayList<UserVO> show\_up(); |
| 前置条件 | 管理员身份登陆系统 |
| 后置条件 | 返回显示当前所有的身份信息 |
| User.login | 语法 | public int login\_up(String name, String password); |
| 前置条件 | 有用户请求登陆 |
| 后置条件 | 验证账号与密码，返回是否成功 |
| User.getUserNote | 语法 | public String getUserNote(); |
| 前置条件 | 管理员添加新用户 |
| 后置条件 | 系统返回新用户的编号 |
| User.searchUser | 语法 | public ArrayList<UserVO> searchUser\_up(String name); |
| 前置条件 | 无 |
| 后置条件 | 系统根据用户名称查找并返回用户信息 |
| 需要的服务 | | |
| UserDataService.addUser(UserPO po) | 添加一个UserPO对象 | |
| UserDataService.delUser(String name) | 根据姓名查找到一个UserPO对象并删除 | |
| UserDataService.getUser(String name) | 根据名字查找到UserPO对象 | |
| UserDataService.show() | 获得所有UserPO对象 | |
| UserDataService.clear() | 清空所有对象 | |

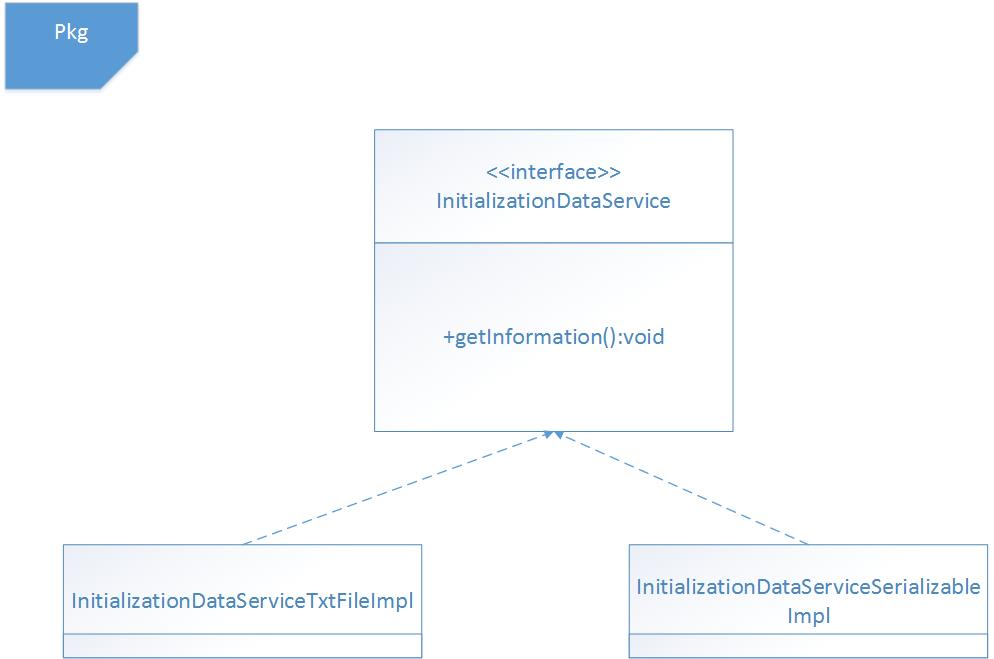
## 5.4数据层的分解

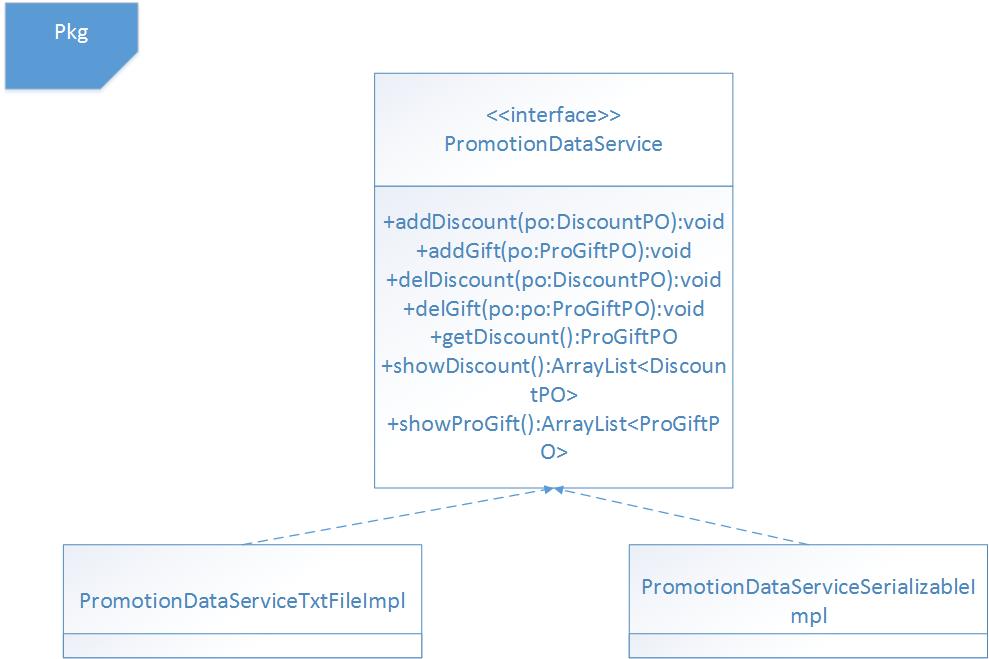
数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增删改查。Sales业务逻辑需要的服务由SalesDataService接口提供。由于持久化数据的保存可能存在多种形式：Txt文件，序列化文件，数据库等，所示抽象了数据服务。数据层模块的描述具体如下图所示。

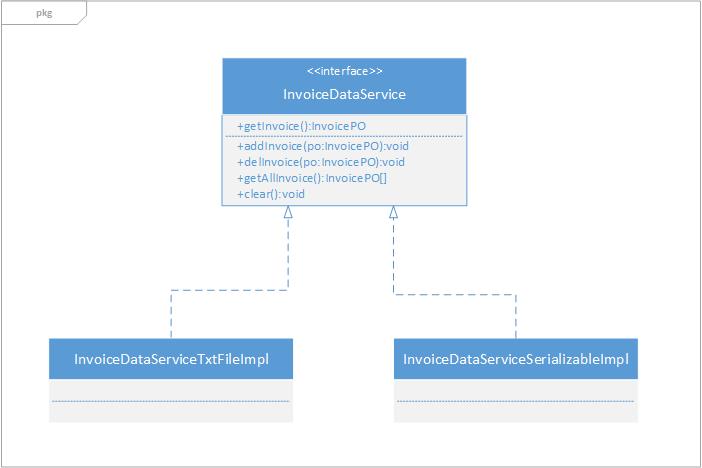


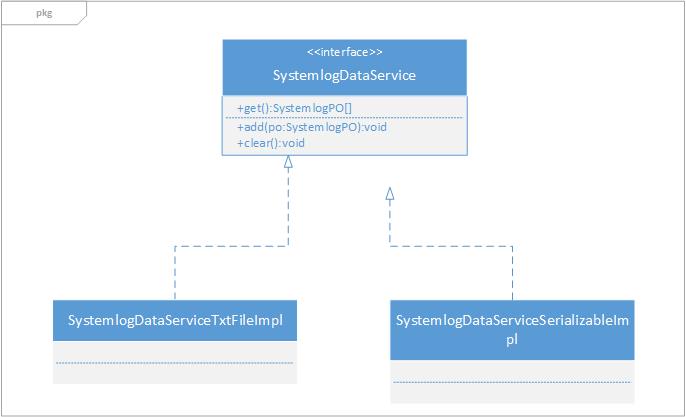


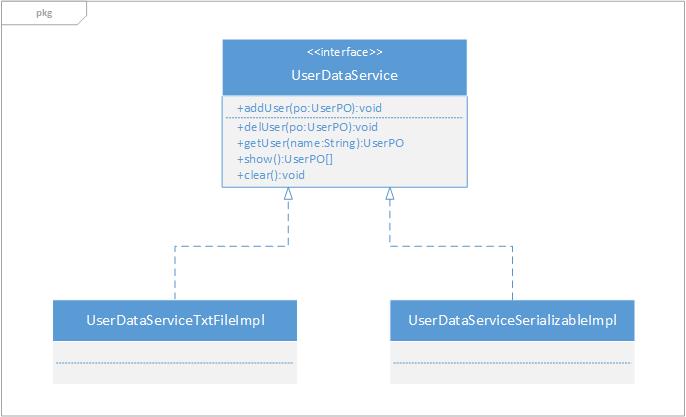












### 5.4.1数据层模块的职责

数据称模块的职责如表11所示。

表11 数据层模块的职责

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| CommodityDataServiceMySerializableFileImpl | 基于序列化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 |

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| SalesDataServiceMySerializableFileImpl | 基于序列化的持久化数a据库的接口，提供集体载入、集体保存、增、删、改、查服务 |

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| AccountDataServiceMySerializableFileImpl | 基于序列化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 |

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| FinancialDataServiceMySerializableFileImpl | 基于序列化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 |

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| InitializationDataServiceMySerializableFileImpl | 基于序列化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 |

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| SystemlogDataServiceMySerializableFileImpl | 基于序列化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 |

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| InvoiceDataServiceMySerializableFileImpl | 基于序列化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 |

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| PromotionDataServiceMySerializableFileImpl | 基于序列化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 |

|  |  |
| --- | --- |
| 模 块 | 职 责 |
| UserDataServiceMySerializableFileImpl | 基于序列化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 |

### 5.4.2数据层模块的接口规范

数据称模块的接口规范如表12所示

表12 数据层模块的接口规范

CommodityData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| CommodityDataService.findGood | 语法 | Public CommodityPO findGood(String name, String type) |
| 前置条件 | 无 |
| 后置条件 | 按名称和型号查找并返回相应的CommodityPO结果 |
| CommodityDataService.addGood | 语法 | Public void addGood(CommodityPO po1, SortPO po2) |
| 前置条件 | 序列化文件中不存在相同的CommodityPO对象，且存在一个SortPO对象可以添加CommodityPO对象 |
| 后置条件 | 添加一个CommodityPO对象 |
| CommodityDataService.delGood | 语法 | Public void delGood(CommodityPO po) |
| 前置条件 | 序列化文件中存在相同的CommodityPO对象，此商品的库存数量为0 |
| 后置条件 | 删除此CommodityPO对象 |
| CommodityDataService.updateGood | 语法 | //更新商品信息  Public void updateGood(CommodityPO po1, CommodityPO po2)  //更新商品所属分类  public boolean updateGood(CommodityPO po1,SortPO po2) throws RemoteException; |
| 前置条件 | 一个CommodityPO对象需要更新 |
| 后置条件 | 更新CommodityPO信息 |
| CommodityDataService.getGoodNote | 语法 | public String getGoodNote(SortPO po1) |
| 前置条件 | 需要添加商品时 |
| 后置条件 | 获得商品编号 |
| CommodityDataService.addSort | 语法 | public boolean addSort(SortPO po1,SortPO po2) |
| 前置条件 | 序列化文件中不存在相同名称的SortPOpo1，且存在SortPO对象po2 |
| 后置条件 | 添加一个SortPO对象 |
| CommodityDataService.delSort(SortPO po) | 语法 | public boolean delSort(SortPO po) |
| 前置条件 | 序列化文件中存在一个SortPO对象需要删除 |
| 后置条件 | 删除这个SortPO对象 |
| CommodityDataService.updateSort | 语法 | //改变分类所属分类  public boolean updateSort\_Mov(SortPO po1,SortPO po2)  //改变分类具体信息  public boolean updateSort(SortPO po1,SortPO po2) |
| 前置条件 | 序列化文件中存在一个SortPO对象需要更新 |
| 后置条件 | 更新这个SortPO对象信息 |
| CommodityDataService.getSortNote | 语法 | public String getSortNote(SortPO po1) |
| 前置条件 | 需要添加一个SortPO对象时 |
| 后置条件 | 获得一个分类的编号 |
| CommodityDataService.getTempSort | 语法 | public SortPO getTempSort() |
| 前置条件 |  |
| 后置条件 |  |
| CommodityDataService.addGift | 语法 | public boolean addSendGift(SendGiftPO po) |
| 前置条件 | 序列化文件不存在相同名称与型号的GiftPO，数量小于库存内相同名称与型号CommodityPO对象的数量 |
| 后置条件 | 序列化文件添加GiftPO对象，库存中相同名称与型号的CommodityPO对象数量减少 |
| CommodityDataService.findGift | 语法 | public SendGiftPO findSendGift(String note) public SendGiftPO findSendGift(SendGiftPO po) |
| 前置条件 | 无 |
| 后置条件 | 根据名字和型号查找并返回SendGiftPO |
| CommodityDataService.delGift | 语法 | public boolean delSendGift(SendGiftPO po)  public boolean delSendGift(String note) |
| 前置条件 | 序列化文件中存在相同名称与型号的GiftPO对象 |
| 后置条件 | 删除GiftPO对象，库存中相同名称与型号的CommodityPO数量增加 |
| CommodityDataService.getSendGiftNote | 语法 | **public** String getSendGiftNote() |
| 前置条件 | 无 |
| 后置条件 | 获得一个赠品的编号 |
| CommodityDataService.getAllSendGift | 语法 | public ArrayList<SendGiftPO> getAllSendGift() |
| 前置条件 | 无 |
| 后置条件 | 获得所有赠品SendGiftPO对象 |
| CommodityDataService.getAll | 语法 | public ArrayList<CommodityPO> getAll() |
| 前置条件 | 无 |
| 后置条件 | 获得所有的CommodityPO对象 |
| CommodityDataService.getAllSort | 语法 | public ArrayList<SortPO> getAllSort() |
| 前置条件 | 无 |
| 后置条件 | 获得所有SortPO对象 |
| CommodityDataService.addWarn | 语法 | public boolean addWarn(WarnPO po) |
| 前置条件 |  |
| 后置条件 |  |
| CommodityDateService.addPatch | 语法 | public boolean addPatch(PatchPO po) |
| 前置条件 | 库存存在报溢报损 |
| 后置条件 | 序列化文件中添加一个PatchPO对象 |
| CommodityDataService.delPatch | 语法 | public boolean delPatch(String note) |
| 前置条件 | 存在一个报溢报损单PatchPO对象需要删除 |
| 后置条件 | 查找并删除该报溢报损单 |
| CommodityDataService.getPatch | 语法 | public PatchPO getPatch(String note) |
| 前置条件 | 无 |
| 后置条件 | 根据单据编号获得一个报溢报损单PatchPO对象 |
| CommodityDataService.getAllPatch | 语法 | public ArrayList<PatchPO> getAllPatch() |
| 前置条件 | 无 |
| 后置条件 | 获得所有PatchPO对象 |
| CommodityDataService.getPatchNote | 语法 | public String getPatchNote() |
| 前置条件 | 添加报溢报损单PatchPO 对象时 |
| 后置条件 | 获得一个报溢报损单编号 |
| CommofityDataService.passPatch | 语法 | public boolean passPatch(PatchPO po) |
| 前置条件 | 存在一个PatchPO对象审批通过 |
| 后置条件 | 根据审批通过的PatchPO对象修改库存信息 |
| CommunityDataService.refusePatch | 语法 | public boolean refusePatch(String note) |
| 前置条件 | 存在一个PatchPO对象审批被拒绝 |
| 后置条件 | 根据审批拒绝的PatchPO对象修改库存信息 |
| CommodityDataService.passSendGift | 语法 | public boolean passSendGift(SendGiftPO po) |
| 前置条件 | 存在一个SendGiftPO对象审批通过 |
| 后置条件 | 根据审批通过的SendGiftPO对象修改库存信息 |
| CommodiyDataService.refuseSendGift | 语法 | public boolean refuseSendGift(String note) |
| 前置条件 | 存在一个SendGiftPO对象审批被拒绝 |
| 后置条件 | 根据审批拒绝的SendGiftPO对象修改库存信息 |
| CommodityDataService.passImport | 语法 | public boolean passImport(ImportPO po) |
| 前置条件 | 存在一个ImportPO对象审批通过 |
| 后置条件 | 根据审批通过的ImportPO对象修改库存信息 |
| CommodityDataService.passImport\_Return | 语法 | public boolean passImport\_Return(Import\_ReturnPO po) |
| 前置条件 | 存在一个Import\_ReturnPO对象审批通过 |
| 后置条件 | 根据审批通过的Import\_ReturnPO对象修改库存信息 |
| CommodityDataService.passExport\_Return | 语法 | public boolean passExport\_Return(Export\_ReturnPO po) |
| 前置条件 | 存在一个Export\_ReturnPO对象审批通过 |
| 后置条件 | 根据审批通过的Export\_ReturnPO对象修改库存信息 |
| CommodityDataService.passExport | 语法 | public boolean passExport(ExportPO po) |
| 前置条件 | 存在一个ExportPO对象审批通过 |
| 后置条件 | 根据审批通过的ExportPO对象修改库存信息 |
| CommodityDataService.addDraftPatch | 语法 |  |
| 前置条件 |  |
| 后置条件 |  |
| CommodityDataService.clear() | 语法 | Public void clear() |
| 前置条件 | 进行期初建账之前 |
| 后置条件 | 删除所有CommodityPO与SortPO对象 |

SalesData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| SalesDataService.addCustomer(CustomerPO po) | 语法 | Public void addCustomer(CustomerPO po) |
| 前置条件 | 序列化文件中不存在相同名称的CustomerPO对象 |
| 后置条件 | 序列化文件添加CustomerPO对象 |
| SalesDataService.delCustomer(CustomerPO po) | 语法 | Public void delCustomer(CustomerPO po) |
| 前置条件 | 序列化文件中存在相同名称的CustomerPO对象，此客户的应付或应收为0 |
| 后置条件 | 序列化文件删除Customer对象 |
| SalesDataService.updateCustomer(CustomerPO po1, CustomerPO po2) | 语法 | Public void updateCustomer(CustomerPO po1, CustomerPO po2) |
| 前置条件 | 序列化文件中存在相同名称的CustomerPO，输入的两个CustomerPO对象名称不一致 |
| 后置条件 | 序列化文件修改Customer对象 |
| SalesDataService.findCustomer(String name) | 语法 | Public CustomerPO findCustomer(String name) |
| 前置条件 | 无 |
| 后置条件 | 根据名称查找并返回CustomerPO对象 |
| SalesDataService.addImport(ImportPO po) | 语法 | Public void addImport(ImportPO po) |
| 前置条件 | 进货单得到总经理审批通过 |
| 后置条件 | 序列化文件中添加ImportPO对象，完成进货单 |
| SalesDataService.addImport\_Return(Import\_ReturnPO po) | 语法 | Public void addImport\_Return(Import\_ReturnPO po) |
| 前置条件 | 进货退货单得到总经理审批通过，库存中相同名称与型号的CommodityPOd对象数量大于退货数量 |
| 后置条件 | 序列化文件中添加Import\_ReturnPO对象，完成进货的退货 |
| SalesDataService.addExport(ExportPO po) | 语法 | Public void addExport(ExportPO po) |
| 前置条件 | 销售单得到总经理审批通过，库存中相同名称与型号的CommodityPO对象数量大于销售数量 |
| 后置条件 | 序列化文件中添加ExportPO对象，完成销售 |
| SalesDataService.addExport\_Return(Export\_ReturnPO po) | 语法 | Public void addExport\_Return(Export\_ReturnPO po) |
| 前置条件 | 销售退货单得到总经理审批通过 |
| 后置条件 | 序列化文件中添加Export\_ReturnPO对象，完成销售的退货 |
| SalesDataService.getAllCustomer() | 语法 | Public ArrayList<CustomerPO> getAllCustomer() |
| 前置条件 | 无 |
| 后置条件 | 返回所有CustomerPO对象 |
| SalesDataService.getAllImport() | 语法 | Public ImportPO[] getAllImport() |
| 前置条件 | 无 |
| 后置条件 | 返回包含所有Import对象的数组 |
| SalesDataService.getAllImport\_Return() | 语法 | Public Import\_ReturnPO[] getAllImport\_Return() |
| 前置条件 | 无 |
| 后置条件 | 返回包含所有Import\_ReturnPO对象的数组 |
| SalesDataService.getAllExport() | 语法 | Public ExportPO[] getAllExport() |
| 前置条件 | 无 |
| 后置条件 | 返回包含所有ExportPO对象的数组 |
| SalesDataService.getAllExport\_Return() | 语法 | Public Export\_ReturnPO[] getAllExport\_Return() |
| 前置条件 | 无 |
| 后置条件 | 返回包含所有Export\_ReturnPO对象的数组 |
| SalesDataService.clear() | 语法 | Public void clear() |
| 前置条件 | 进行期初建账之前 |
| 后置条件 | 清空所有的CustomerPO，ImportPO，Import\_ReturnPO，ExportPO，Export\_ReturnPO对象 |
| SaleDataService.addDraftImport(ImportPO po) | 语法 | public boolean addDraftImport(ImportPO po) |
| 前置条件 | 销售人员选择添加草稿进货单 |
| 后置条件 | 序列化文件中添加草稿进货单 |
| SaleDataService.addDraftImport\_Return(Import\_ReturnPO po) | 语法 | public boolean addDraftImport\_Return(Import\_ReturnPO po) |
| 前置条件 | 销售人员选择添加草稿进货退货单 |
| 后置条件 | 序列化文件中添加草稿进货退货单 |
| SaleDataService.addDraftExport(ExportPO po) | 语法 | public boolean addDraftExport(ExportPO po) |
| 前置条件 | 销售人员选择添加草稿销售单 |
| 后置条件 | 序列化文件中添加草稿销售单 |
| SaleDataService.addDraftExport\_Return(Export\_ReturnPO po) | 语法 | public boolean addDraftExport\_Return(Export\_ReturnPO po) |
| 前置条件 | 销售人员选择添加草稿销售退货单 |
| 后置条件 | 序列化文件中添加草稿销售退货单 |
| SaleDataService.delDraftImport(String note) | 语法 | public boolean delDraftImport(String note) |
| 前置条件 | 销售人员选择删除草稿进货单 |
| 后置条件 | 序列化文件中删除草稿进货单 |
| SaleDataService.delDraftImport\_Return(String note) | 语法 | public boolean delDraftImport\_Return(String note) |
| 前置条件 | 销售人员选择删除草稿进货退货单 |
| 后置条件 | 序列化文件中删除草稿进货退货单 |
| SaleDataService.delDraftExport(String note) | 语法 | public boolean .delDraftExport(String note) |
| 前置条件 | 销售人员选择删除草稿销售单 |
| 后置条件 | 序列化文件中删除草稿销售单 |
| SaleDataService.delDraftExport\_Return(String note) | 语法 | public boolean delDraftExport\_Reutrn(String note) |
| 前置条件 | 销售人员选择删除草稿销售退货单 |
| 后置条件 | 序列化文件中删除草稿销售退货单 |
| SaleDataService.getAllDraftImport() | 语法 | public ArrayList<ImportPO> getAllDraftImport() |
| 前置条件 | 无 |
| 后置条件 | 获得所有的草稿进货单 |
| SaleDataService.getAllDraftImport\_Return() | 语法 | Public ArrayList<Import\_ReturnPO> getAllDraftImport\_Return() |
| 前置条件 | 无 |
| 后置条件 | 获得所有的草稿进货退货单 |
| SaleDataService.getAllDraftExport() | 语法 | public ArrayList<ExportPO> getAllDraftExport() |
| 前置条件 | 无 |
| 后置条件 | 获得所有的草稿销售单 |
| SaleDataService. getAllDraftExport\_Return() | 语法 | Public ArrayList<Export\_ReturnPO> getAllDraftExport\_Return() |
| 前置条件 | 无 |
| 后置条件 | 获得所有的草稿销售退货单 |
| SaleDataService.getDraftImport(String note) | 语法 | public ImportPO getDraftImport(String note) |
| 前置条件 | 销售人员根据编号查找草稿进货单 |
| 后置条件 | 根据编号返回草稿进货单 |
| SaleDataService.getDraftImport\_Return(String note) | 语法 | public Import\_ReturnPO getDraftImport\_Return(String note) |
| 前置条件 | 销售人员根据编号查找草稿进货退货单 |
| 后置条件 | 根据编号返回草稿进货退货单 |
| SaleDataService.getDraftExport(String note) | 语法 | public ExportPO getDraftExport(String note) |
| 前置条件 | 销售人员根据编号查找草稿销售单 |
| 后置条件 | 根据编号返回草稿销售单 |
| SaleDataService.getDraftExport\_Return(String note) | 语法 | public Export\_ReturnPO getDraftExport\_Return(String note) |
| 前置条件 | 销售人员根据编号查找草稿销售退货单 |
| 后置条件 | 根据编号返回草稿销售退货单 |
| SaleDataService.getImport | 语法 | public ImportPO getImport(String note) |
| 前置条件 | 销售人员根据编号查找进货单 |
| 后置条件 | 根据编号返回进货单 |
| SaleDataService.getImport\_Return | 语法 | Public Import\_ReturnPO getImport\_Return(String note) |
| 前置条件 | 销售人员根据编号查找进货退货单 |
| 后置条件 | 根据编号返回进货退货单 |
| SaleDataService. getExport | 语法 | public ExportPO getExport(String note) |
| 前置条件 | 销售人员根据编号查找销售单 |
| 后置条件 | 根据编号返回销售单 |
| SaleDataService.getExport\_Return | 语法 | public Export\_ReturnPO getExport\_Return(String note) |
| 前置条件 | 销售人员根据编号查找销售退货单 |
| 后置条件 | 根据编号返回销售退货单 |
| SaleDataService. getImportNote | 语法 | public String getImportNote() |
| 前置条件 | 销售人员请求添加进货单时 |
| 后置条件 | 系统返回进货单编号 |
| SaleDataService.getImport\_ReturnNote | 语法 | public String getImport\_ReturnNote() |
| 前置条件 | 销售人员请求添加进货退货单时 |
| 后置条件 | 系统返回进货退货单编号 |
| SaleDataService. getExportNote | 语法 | public String getExportNote() |
| 前置条件 | 销售人员请求添加销售单时 |
| 后置条件 | 系统返回销售单编号 |
| SaleDataService.getExport\_ReturnNote | 语法 | public String getExport\_ReturnNote() |
| 前置条件 | 销售人员请求添加销售退货单时 |
| 后置条件 | 系统返回销售退货单编号 |
| SaleDataService.getImport\_ReturnMaxNumber | 语法 | public int getImport\_ReturnMaxNumber(String old\_note) |
| 前置条件 | 销售人员进行进货退货时，请求将输入的退货数量与最大可退货数量进行比较 |
| 后置条件 | 返回可最大进货退货数量 |
| SaleDataService.getExport\_ReturnMaxNumber | 语法 | public int getExport\_ReturnMaxNumber(String old\_note) |
| 前置条件 | 销售人员进行销售退货时，请求将输入的退货数量与最大可退货数量进行比较 |
| 后置条件 | 返回可最大销售退货数量 |
| SalesDataService.clear | 语法 | public boolean clear() |
| 前置条件 | 销售人员选择清空所有对象 |
| 后置条件 | 序列化文件中清空所有对象 |

AccountData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| AccountDataService.getAccount() | 语法 | Public AccountPO getAccount() |
| 前置条件 | 无 |
| 后置条件 | 返回当前账户AccountPO对象 |
| AccountDataService.addAccount(AccountPO po) | 语法 | Public void addAccount(AccountPO po) |
| 前置条件 | 数据库中不存在相同名称的AccountPO对象 |
| 后置条件 | 数据库添加AccountPO对象 |
| AccountDataService.delAccount(AccountPO po) | 语法 | Public void delAccount(AccountPO po) |
| 前置条件 | 数据库中存在相同名称的AccountPO对象，此对象中应付和应收为0 |
| 后置条件 | 数据库删除AccountPO对象 |
| AccountDataService.updateAccount(AccountPO po1, AccountPO po2) | 语法 | Public void updateAccount(AccountPO po1, AccountPO po2) |
| 前置条件 | 数据库中存在相同名称的AccountPO对象，输入的两个AccountPO对象的名称相同 |
| 后置条件 | 数据库更新AccountPO对象的信息 |
| AccountDataService.findAccount(String name) | 语法 | Public AccountPO findAccount(String name) |
| 前置条件 | 无 |
| 后置条件 | 根据名称查找并返回AccountPO对象 |
| AccountDataService.getAllAccount() | 语法 | Public AccountPO[] getAllAccount() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有AccountPO对象的数组 |
| AccountDataService.addReceipt(ReceiptPO po) | 语法 | Public void addReceipt(ReceiptPO po) |
| 前置条件 | 收款单得到总经理审批通过 |
| 后置条件 | 数据库添加ReceiptPO对象 |
| AccountDataService.addPayment(PaymentPO po) | 语法 | Public void addPayment(PaymentPO po) |
| 前置条件 | 付款单得到总经理审批通过 |
| 后置条件 | 数据库添加PaymentPO对象 |
| AccountDataService.getAllReceipt() | 语法 | Public ReceiptPO[] getAllReceipt() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有ReceiptPO对象的数组 |
| AccountDataService.getAllPayment() | 语法 | Public PaymentPO[] getAllPayment() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有PaymentPO对象的数组 |
| AccountDataService.clear | 语法 | Public void clear() |
| 前置条件 | 进行期初建账之前 |
| 后置条件 | 清除所有AccountPO，ReceiptPO，PaymentPO对象 |

FinancialData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| FinancialDataService.getSaleList() | 语法 | Public SaleListPO[] getSaleList() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有SaleListPO对象的数组 |
| FinancialDataService.getAllBill() | 语法 | Public AllBillPO[] getAllBill() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有AllBillPO对象的数组 |
| FinancialDataService.getOperatingCondition() | 语法 | Public OperatingConditionPO[] getOperatingCondition() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有OperatingConditionPO对象的数组 |
| FinancialDataService.addSaleList(SaleListPO po) | 语法 | Public void addSaleList(SaleListPO po) |
| 前置条件 | 完成一个进货或销售单据的审批 |
| 后置条件 | 数据库添加一个SaleListPO对象 |
| FinancialDataService.addAllBill(AllBillPO po) | 语法 | Public void addAllBill(AllBillPO po) |
| 前置条件 | 完成任何单据的审批 |
| 后置条件 | 数据库添加一个AllBillPO对象 |
| FinancialDataService.addOperatingCondition(OperatingConditionPO po) | 语法 | Public void addOperationCondition(OperatingConditionPO po) |
| 前置条件 | 完成收付款单据的审批 |
| 后置条件 | 数据库添加一个OperatingConditionPO对象 |
| FinancialDataService.clear() | 语法 | Public void clear() |
| 前置条件 | 进行期初建账之前 |
| 后置条件 | 清楚所有SaleListPO，AllBillPO，OperatingConditionPO对象 |

InitializationData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| InitializationDataService.getInformation | 语法 | Public InitializationPO getInformation() |
| 前置条件 | 无 |
| 后置条件 | 返回InitializationPO对象 |

PromotionData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| PromotionDataService.addDiscount(DiscountPO po) | 语法 | Public void addDiscount(DiscountPO po) |
| 前置条件 | 无 |
| 后置条件 | 添加或替换当前的DiscountPO对象 |
| PromotionDataService.addGift(ProGiftPO po) | 语法 | Public void addGift(ProGiftPO po) |
| 前置条件 | 无 |
| 后置条件 | 添加或替换当前的ProGiftPO对象 |
| PromotionDataService.addVoucher(ProGiftPO po) | 语法 | Public void addVoucher(VoucherPO po) |
| 前置条件 | 无 |
| 后置条件 | 添加或替换当前的ProGiftPO对象 |
| PromotionDataService.delDiscount(int level) | 语法 | Public void delDiscount(int level) |
| 前置条件 | 当前存在处于有效期的DiscountPO对象 |
| 后置条件 | 清除DiscountPO对象 |
| PromotionDataService.delGift(int level) | 语法 | Public void delGift(int level) |
| 前置条件 | 当前存在处于有效期的ProGiftPO对象 |
| 后置条件 | 清除ProGiftPO对象 |
| PromotionDataService.delVoucher(int level) | 语法 | Public void delVoucher(int level) |
| 前置条件 | 当前存在处于有效期的VoucherPO对象 |
| 后置条件 | 清除VoucherPO对象 |
| PromotionDataservice.getDiscount(int level) | 语法 | Public DiscountPO getDiscount(int level) |
| 前置条件 | 无 |
| 后置条件 | 根据客户登记返回对应的DiscountPO对象 |
| PromotionDataService.getGift(int level) | 语法 | Public ProGiftPO getGift(int level) |
| 前置条件 | 无 |
| 后置条件 | 根据客户登记返回对应的ProGiftPO对象 |
| PromotionDataService.geVoucher(int level) | 语法 | Public VoucherPO getGift(int level) |
| 前置条件 | 无 |
| 后置条件 | 根据客户等级返回相应的VoucherPO对象 |
| PromotionDataService.showDiscount() | 语法 | Public ArrayList<DiscountPO> showDiscount() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有的DiscountPO对象的arrayList |
| PromotionDataService.showProGift() | 语法 | Public ArrayList<ProgiftPO> showProGift() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有的ProGiftPO对象的arraylist |
| PromotionDataService.showVoucher() | 语法 | Public ArrayList<VouvherPO> showProGift() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有的VoucherPO对象的arraylist |
| PromotionDataService.clear() | 语法 | Public boolean clear() |
| 前置条件 | 无 |
| 后置条件 | 清空所有促销策略，并返回是否成功清空 |

InvoiceData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| InvoiceDataServoice.getInvoice() | 语法 | Public InvoicePO getInvoice() |
| 前置条件 | 无 |
| 后置条件 | 返回最新的待审批的InvoicePO对象 |
| InvoiceDataServoice.addInvoice(InvoicePO po) | 语法 | Public void addInvoice(InvoicePO po) |
| 前置条件 | 无 |
| 后置条件 | 添加InvoicePO对象 |
| InvoiceDataServoice.delInvoice(InvoicePO po) | 语法 | Public void delInvoice(InvoicePO po) |
| 前置条件 | 存在相同编号的InvoicePO对象 |
| 后置条件 | 删除InvoicePO对象 |
| InvoiceDataService.getAllInvoice() | 语法 | Public InvoicePO[] getAllInvoice() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有InvoicePO对象的数组 |
| InvoiceDataService.clear() | 语法 | Public void clear() |
| 前置条件 | 无 |
| 后置条件 | 清除所有InvoicePO对象 |

SystemlogData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| SystemlogDataService.get() | 语法 | Public SystemlogPO[] get() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有SystemlogPO对象的数组 |
| SystemlogDataService.add(String word) | 语法 | Public void add(SystemlogPO po) |
| 前置条件 | 无 |
| 后置条件 | 添加SystemlogPO对象 |
| SystemlogDataService.clear() | 语法 | Public void clear() |
| 前置条件 | 无 |
| 后置条件 | 清除所有SystemlogPO对象 |

UserData的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务 | | |
| UserDataService.addUser(UserPO po) | 语法 | Public void addUser(UserPO po) |
| 前置条件 | 不存在相同名称的UserPO对象 |
| 后置条件 | 添加UserPO对象 |
| UserDataService.delUser(UserPO po) | 语法 | Public void delUser(UserPO po) |
| 前置条件 | 存在相同名称的UserPO对象 |
| 后置条件 | 删除UserPO对象，若用户正在登陆，则强制退出程序 |
| UserDataService.getUser(String name) | 语法 | Public UserPO getUser(String name) |
| 前置条件 | 无 |
| 后置条件 | 根据名称查找返回UserPO对象 |
| UserDataService.show() | 语法 | Public UserPO[] show() |
| 前置条件 | 无 |
| 后置条件 | 返回一个包含所有UserPO对象的数组 |
| UserDataService.clear() | 语法 | Public void clear() |
| 前置条件 | 无 |
| 后置条件 | 清除所有UserPO对象 |

# 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。

CustomerPO类包括客户的名称、客户等级、客户邮箱、邮编、地址、电话。

UserPO类包含软件使用者的用户名、密码属性

CommodityPO类是包含商品的价格、数量和名字属性

SalesListPO类是保存销售时的数据的类，包含时间，商品属性，客户属性，业务员，仓库，价格，数量，总价

GiftPromotionPO包含商品名称、商品类型、商品ID、促销起始日、促销结束日、礼品编号、礼品数量、客户等级、赠品起送金额，赠品截止金额。

DiscountPO类包括起始时间、结束时间、折价起始金额、折价金额、促销截止金额、客户等级

VoucherPO类包括起始时间、结束时间、代金券起始金额、代金券金额、促销截止金额、客户等级

AccountPO类包括客户名称、应收金额、应付金额、已收金额、已付金额属性

InitialiazationPO包括商品信息、客户信息和账户信息

SaleListPO包括时间、商品名称、商品型号、客户、业务员、仓库、数量、单价、总额等属性

AllBillPO包括时间、单据类型、客户、业务员、仓库、单据信息等属性

OperatingConditionPO包括时间、单据类型和单据信息属性

InvoicePO包括单据编号、单据类型

SystemLogPO包括时间、系统日志

持久化用户对象UserPO的定义如图14所示

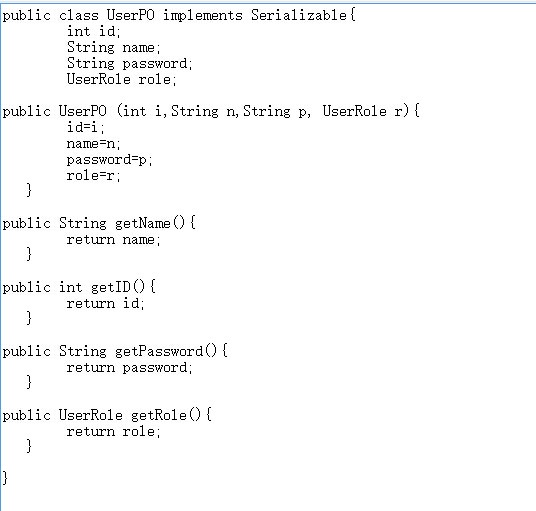


图14 持久化用户对象UserPO的定义

## 6.2 Txt持久化格式

Txt数据保持格式以Commodity.txt为例。每行分别对应货号、商品名称、价格、数量。中间用“：”隔开。如下所示：

123：杯子：10:32

456：桌子：20:22