**进销存系统**

**体系结构设计文档**

**V 1.1 正式版**

**南京大学软件学院**

**吮指黄金柠檬脆皮鸡工作组**

黄涵倩严顺宽王宁(79)金翠

**2014年11月10日**

**目录**

更新历史………………………………………………………………………………3

[1、引言](#_引言)……………………………………………………………………………3

1.1编制目的……………………………………………………………………3

1.2词汇表………………………………………………………………………3

1.3参考目录……………………………………………………………………3

[2、产品概述](#_2、产品概述)…………………………………………………………………………3

[3、逻辑视角](#_3、逻辑视角)…………………………………………………………………………3

[4、组合视角](#_4、组合视角)…………………………………………………………………………5

4.1开发包图……………………………………………………………………5

4.2运行时进程…………………………………………………………………7

4.3物理部署……………………………………………………………………7

[5、接口视角](#_5、接口视角)…………………………………………………………………………8

5.1模块的职责…………………………………………………………………8

5.2用户界面层的分解…………………………………………………………10

5.2.1用户界面层模块的职责……………………………………………11

5.2.2用户界面层模块的接口规范………………………………………11

5.2.3用户界面层模块设计原理…………………………………………12

5.3业务逻辑层的分解…………………………………………………………12

5.3.1业务逻辑层模块的职责……………………………………………12

5.3.2业务逻辑层模块的接口规范………………………………………12

5.4数据层的分解………………………………………………………………19

5.4.1数据层模块的职责…………………………………………………21

5.4.2数据层模块的接口规范……………………………………………22

5.5客户端启动模块的分解……………………………………………………27

5.5.1客户端启动模块的职责……………………………………………28

5.5.2客户端启动模块的接口规范………………………………………28

5.5.3客户端启动模块设计原理…………………………………………28

5.6服务器端启动模块的分解………………………………………………29

5.6.1服务器端启动模块的职责………………………………………29

5.6.2服务器端启动模块的接口规范……………………………………29

5.6.3服务器端启动模块设计原理………………………………………29

5.7客户端网络模块的分解……………………………………………………29

5.8服务器端网络模块的分解…………………………………………………30

[6、信息视角](#信息视角)…………………………………………………………………………31

6.1数据持久化对象…………………………………………………………31

6.2序列化文件表……………………………………………………………………33

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 黄涵倩严顺宽王宁(79)金翠 | 2014/10/15 | 最初草稿 | V0.1草稿版 |
| 黄涵倩严顺宽王宁(79)金翠 | 2014/10/17 | 修改了ReceiptBL.View，ReceiptDataService.Show, Promotionbl.Show,PromotionDataService.Show接口语法的返回类型 | V1.0 |
| 金翠 | 2014/11/10 | 据批改意见，详细补充并明确业务层个模块职责 | V1.1正式版 |
| 严顺宽 | 2014/12/21 | 完善库存部分的信息 | V1.2 |

# 引言

* 1. **编制目的**

本报告详细完成对进销存系统的概要设计，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 |
| 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高校选课系统软件需求规格说明。

# 2、产品概述

参考进销存系统用例文档和高校选课系统软件需求规格说明中对产品的概括描述。

# 3、逻辑视角

在进销存系统中，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）能够很好地展示整个高层抽象。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1，2所示。



**图1 参照体系结构风格的包图表达逻辑视角**



**图2进销存系统最终的软件体系结构设计逻辑方案**

# 4、组合视角

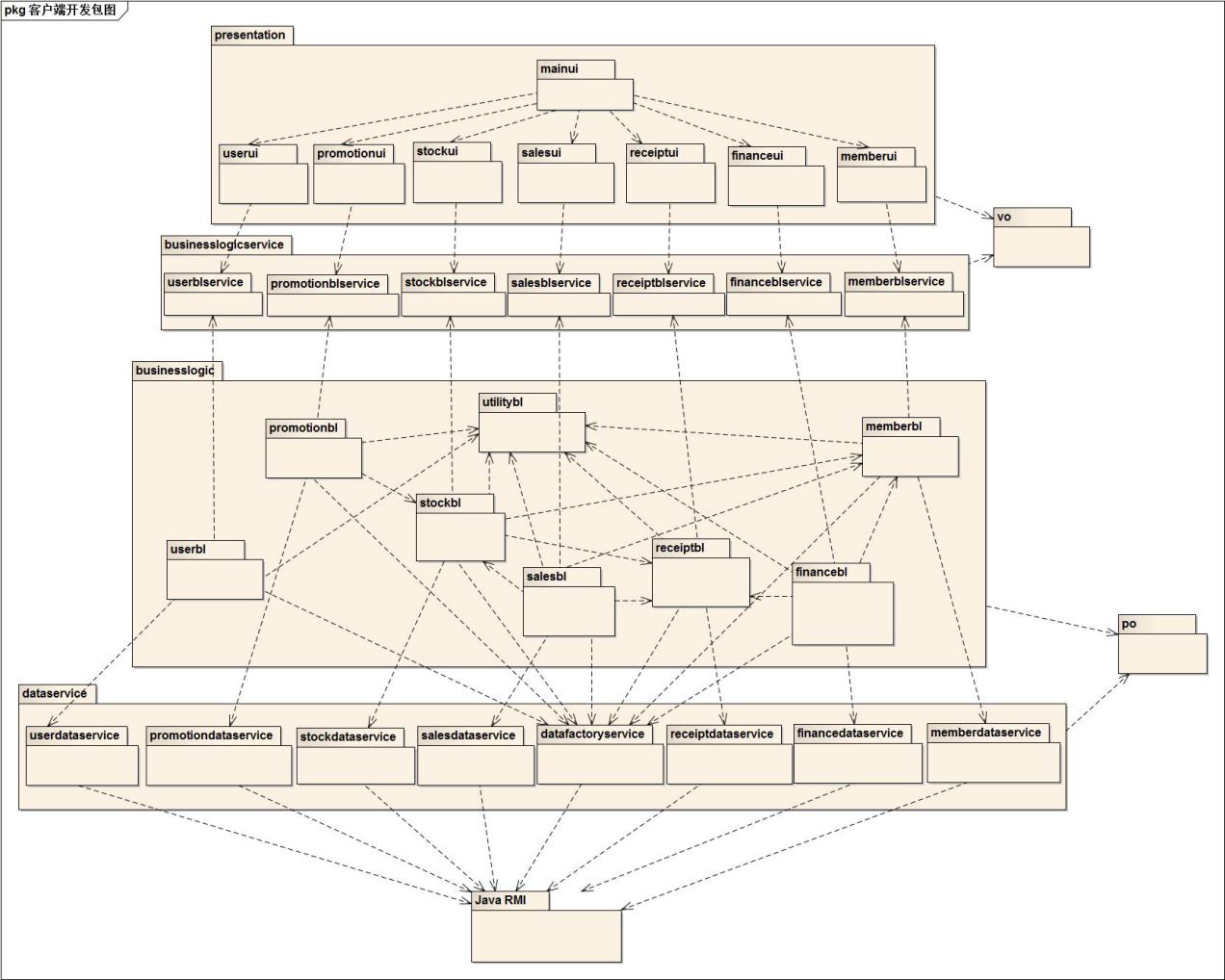
**4.1 开发包图**

进销存系统的最终开发包设计如表1所示。

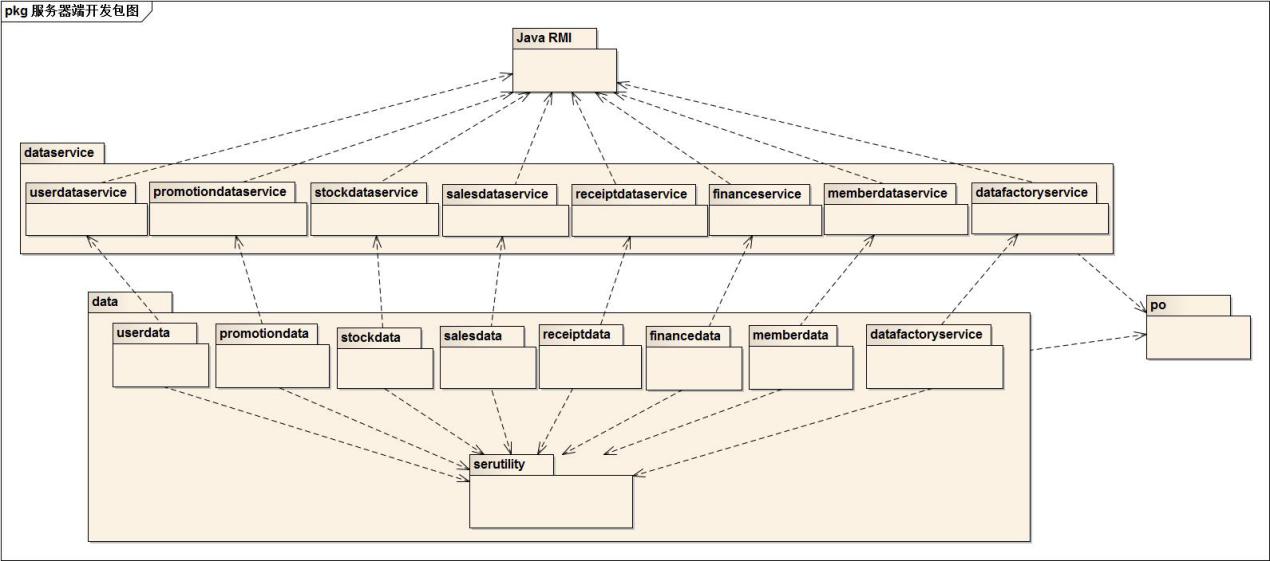
**表1进销存系统的最终开发包设计**

|  |  |
| --- | --- |
| **开发（物理）包** | **依赖的其他开发包** |
| **mainui** | **salesui,stockui,financeui,receiptui,promotionui,memberui,userui,vo** |
| **salesui** | **salesblservice，界面类库包,vo** |
| **salesblservice** |  |
| **salesbl** | **salesblservice,salesdataservice,po,stockbl,** receiptbl**,promotionbl,memberbl** |
| **salesdataservice** | **java rmi,po** |
| **salesdata** | **serutility,po,salesdataservice** |
| **stockui** | **stockblservice,界面类库包** |
| **stockblservice** |  |
| **stockbl** | **stockblservice,stockdataservice,po,** receiptbl**,memberbl** |
| **stockdataservice** | **java rmi,po** |
| **stockdata** | **java rmi,po,serutility** |
| **financeui** | **financeblservice,界面类库包** |
| **financeblservice** |  |
| **financebl** | **financeblservice,financedataservice,po,memberbl,** receiptbl |
| **financedataservice** | **java rmi,po** |
| **financedata** | **java rmi,po,serutility** |
| **receiptui** | **receiptblservice,界面类库包** |
| **receiptblservice** |  |
| **receiptbl** | **receiptblservice, receiptdataservice** |
| **receiptdataservice** | **java rmi,po** |
| **receiptdata** | **java rmi,po,serutility** |
| **Promotionui** | **Promotionblservice, 界面类库包** |
| **Promotionblservice** |  |
| **Promotionbl** | **Promotionblservice,promotiondataservice,po ,stockbl** |
| **Promotiondataservice** | **java rmi,po** |
| **Promotiondata** | **java rmi,po,serutility** |
| **Memberui** | **memberblservice,界面类库包** |
| **Memberblservice** |  |
| **Memberbl** | **Memberblservice,memberdataservice,vo** |
| **Memberdataservice** | **java rmi,po** |
| **memberdata** | **java rmi,po,serutility** |
| **userui** | **userblservice,界面类库包** |
| **userblservice** |  |
| **userbl** | **userinterface,userdataclient,userpo** |
| **userdataservice** | **java rmi,po** |
| **userdata** | **java rmi,po,serutility** |
| **vo** |  |
| **po** |  |
| **utilitybl** |  |
| **界面类库包** |  |
| **java rmi** |  |
| **serutility** | **io类库包** |

进销存系统客户端开发包图如图3所示，服务器端开发包图如图4所示。



**图3客户器端开发包图**



**图4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。进程图如图5所示。



**图5 进程图**

**4.3 物理部署**

进销存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 模块的职责**

客户端模块和服务器模块视图分别如图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 |

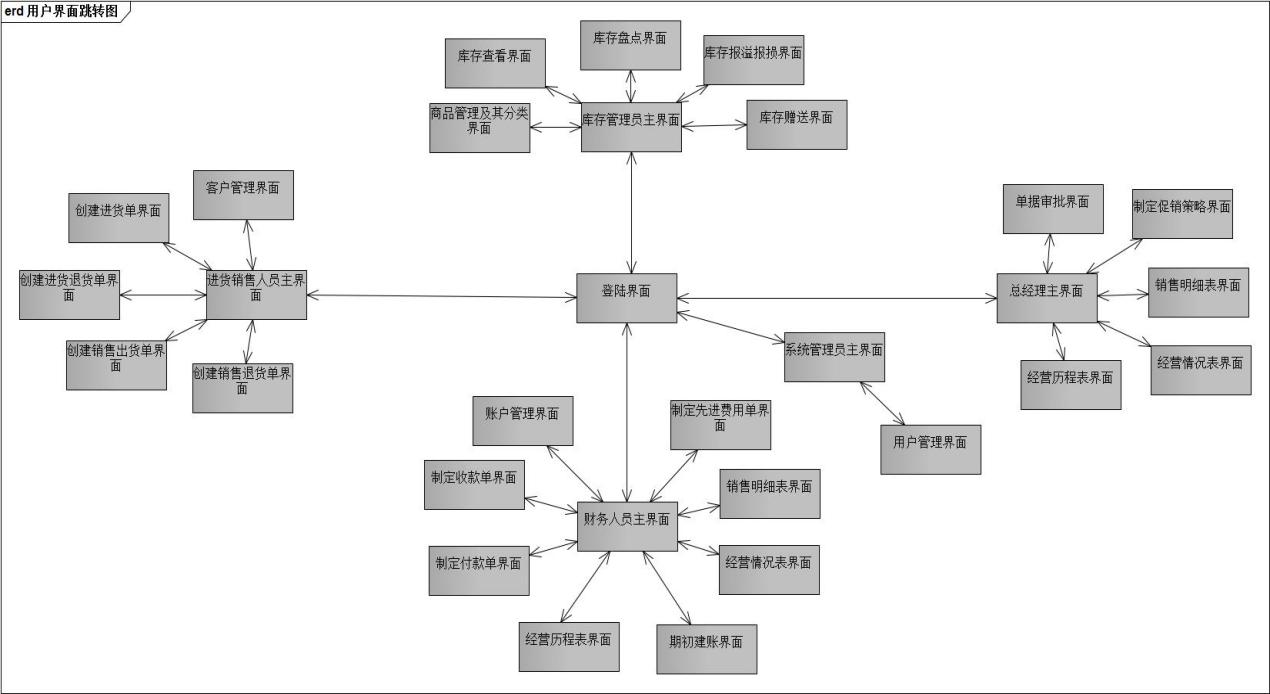
每一层只是使用下方直接接触的层。层与层之间仅仅是通过接口的调用来完成的。层之间调用的接口如表4所示。

**表4 层之间调用的接口**

|  |  |  |
| --- | --- | --- |
| **接口** | **服务调用方** | **服务提供方** |
| UserBLService  PromotionBLService  SalesBLService  StockBLService  FinanceBLService  ReceiptBLService  MemberBLService | 客户端展示层 | 客户端业务逻辑层 |
| UserDataService  PromotionDataService  SalesDataService  StockDataService  FinanceDataService  ReceiptDataService  MemberDataService SerFactory | 客户端业务逻辑层 | 服务器端数据层 |

**5.2 用户界面层的分解**

根据需求，系统存在27个用户界面：登录界面、库存管理员主界面、财务人员主界面、进货销售人员主界面、总经理主界面、系统管理员主界面、库存盘点界面、库存查看界面、库存报溢报损界面、库存赠送界面、商品管理及其分类界面、客户管理界面、制定进货单界面、制定退货单界面、制定销售出货单界面、制定销售退货单界面、账户管界面、制定收款单界面、制定付款单界面、制定现金费用单界面、经营里程表界面、销售明细表界面、经营情况表界面、期初建账界面、单据审批界面、制定促销策略界面、用户管理界面。界面跳转如图9所示。



**图9 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10所示。



**图10 用户界面类**

**5.2.1用户界面模块的职责**

如表5所示为用户界面层模块的职责

**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跳转 |

**5.2.2用户界面模块的接口规范**

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**

|  |  |  |
| --- | --- | --- |
| 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7所示。

**表7用户界面层模块需要的服务接口**

|  |  |
| --- | --- |
| **服务名** | **服务** |
| businesslogicservice.UserBLService | 登录界面的业务逻辑（身份验证）接口  用户的增删改查的业务逻辑接口 |
| businesslogicservice.ReceiptBLService | 查看界面和单据审批的业务逻辑接口 |
| businesslogicservice.SalesBLService | 进货单，退货单制定的业务逻辑接口  销售出货单，销售退货单制定的业务逻辑接口  查看界面的业务逻辑接口 |
| businesslogicservice.StockBLService | 商品分类管理，商品管理的业务逻辑接口  库存管理,库存赠送的业务逻辑接口 |
| businesslogicservice.FinanceBLService | 账户的增删改查的业务逻辑接口  收付款单，现金费用单制定的业务逻辑接口  期初建账的业务逻辑接口  查看界面的业务逻辑接口 |
| businesslogicservice.PromotionBLService | 促销策略制定和修改的业务逻辑接口 |
| businesslogicservice.MemberBLService | 客户的增删改查的业务逻辑接口 |

**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，userbl模块负责处理登录界面的业务逻辑，promotionbl负责处理选择与促销策略有关的界面的业务逻辑。业务逻

辑层的设计如图11所示。



**图11 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** |
| **memberbl** | 负责实现对应于会员增删改查所需要的服务 |
| **salesbl** | 负责实现所有关于进货，进货退货，销售，销售退货，查看销售明细界面所需要的服务 |
| **userbl** | 负责实现对应于用户登录，管理员增删改查用户界面所需要的服务 |
| **stockbl** | 负责实现所有关于商品及其分类管理,库存报溢,报损，报警及库存赠送界面所需要的服务 |
| **financebl** | 负责实现所有对应于创建收付款单，现金费用单，账户管理，期初建账，查看经营情况表所需要的服务 |
| **Receiptbl** | 负责实现所有单据的审批，修改，红冲，查看经营历程的各项单据业务所需的服务 |
| **promotionbl** | 负责实现对应于促销策略制定，修改，发布，查看界面所需的服务 |

**5.3.2业务逻辑层模块的接口规范**

**表9 userbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| User.login | 语法 | public int login(String ID,String password) | |
| 前置条件 | password符合输入规则 | |
| 后置条件 | 查找是否存在相同的User，根据输入的password返回登录验证的结果 | |
| User.addUser | 语法 | public int addUser(UserVO vo) | |
| 前置条件 | User经过验证有添加新用户的权利，输入了有效的新用户信息 | |
| 后置条件 | 向系统中输入新增的用户账户信息 | |
| User.deleteUser | 语法 | public int deleteUser(UserVO vo) | |
| 前置条件 | User经过验证有删除用户的权利，输入了有效的待删除用户信息 | |
| 后置条件 | 系统查找对应的用户账户信息，并删除该用户对象 | |
| User.modifyUser | 语法 | public int modifyUser(UserVO vo) | |
| 前置条件 | User经过验证有修改用户的权利，输入了有效的用户修改信息 | |
| 后置条件 | 系统根据输入的用户信息和用户编号更新用户信息 | |
| User.showUser | 语法 | public UserPO showUser(String UserID) | |
| 前置条件 | User经过验证有查看所有用户的权利，输入了有效的用户ID信息 | |
| 后置条件 | 根据输入的ID查找用户，并返回用户信息 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| FileFactory.getUserData | | | 得到User文件的引用 |
| UserDataService.add(UserPO po) | | | 在文件记录中插入UserPO对象 |
| UserDataService.check(UserPO po) | | | 得到根据输入的ID和password验证的结果 |
| UserDataService.delete(UserPO po) | | | 在文件中删除一个po对象 |
| userDataService.modify(UserPO po) | | | 在文件中更新一个po对象 |
| Info.showUserInfo(long ID) | | | 得到根据输入的ID返回对应的po对象 |

**表10 stockbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| Stock.addGoodsClass | 语法 | public int addGoodsClass (GoodsClassVO vo) | |
| 前置条件 | User经过验证有添加新商品分类的权利，输入了有效的新分类信息 | |
| 后置条件 | 向系统中输入新增的商品分类信息 | |
| Stock.deleteGoodsClass | 语法 | public int deleteGoodsClass (GoodsClassVO vo) | |
| 前置条件 | User经过验证有删除分类的权利，输入了有效的待删除的信息 | |
| 后置条件 | 系统查找对应的商品分类信息，并删除该商品分类对象 | |
| Stock.modifyGoodsClass | 语法 | public int modifyGoodsClass (GoodsClassVO vo) | |
| 前置条件 | User经过验证有修改商品分类的权利，输入了有效的商品分类信息 | |
| 后置条件 | 系统根据输入的分类名称和商品上级分类更新商品分类信息 | |
| Stock.show | 语法 | public ArrayList<GoodsClassVO> show() | |
| 前置条件 | User经过验证有查看商品分类的权利 | |
| 后置条件 | 系统返回所有商品分类信息 | |
| Stock.showGoodsClassInfo | 语法 | public GoodsClassVO showGoodsClassInfo(String name) | |
| 前置条件 | User经过验证有查看某种商品分类的权利 | |
| 后置条件 | 系统根据输入的商品分类的名字，返回该商品分类信息 | |
| Stock.addGoods | 语法 | public int addGoods (GoodsVO vo) | |
| 前置条件 | User经过验证有添加新商品的权利，输入了有效的新商品信息 | |
| 后置条件 | 向系统中输入新增的商品信息 | |
| Stock.deleteGoods | 语法 | public int deleteGoods (GoodsVO vo) | |
| 前置条件 | User经过验证有删除商品信息的权利，输入了有效的待删除商品信息 | |
| 后置条件 | 系统查找对应的商品信息，并删除该商品对象 | |
| Stock.modifyGoods | 语法 | public int modifyGoods (GoodsVO vo) | |
| 前置条件 | User经过验证有修改商品属性的权利，输入了有效的新商品属性 | |
| 后置条件 | 系统根据输入的商品的各个属性更新商品信息 | |
| Stock.findGoods | 语法 | public ArrayList<GoodsPO> findGoods (String message) | |
| 前置条件 | User经过验证有查找商品的权利，输入了有效的商品关键字信息 | |
| 后置条件 | 系统根据输入的商品的相关属性显示查询到的商品信息 | |
| Stock.findByID | 语法 | public GoodsVO findByID(String id) throws RemoteException | |
| 前置条件 | User经过验证有查找商品的权利，输入有效商品id | |
| 后置条件 | 系统根据用户输入的商品id，返回该id的商品信息 | |
| Stock.showGoods | 语法 | public ArrayList<GoodsVO> showGoods() | |
| 前置条件 | User经过验证有获得所有商品信息的权利，想获取所有商品信息 | |
| 后置条件 | 系统返回库存所有记录商品的详细信息 | |
| Stock.showGoodsByClass | 语法 | public ArrayList<GoodsVO> showGoodsByClass(String className) | |
| 前置条件 | User经过验证有获得所有商品信息的权利，输入了有效的商品分类名称 | |
| 后置条件 | 系统根据用户输入的分类名称，返回该名称下的所有商品信息 | |
| Stock.getAllGoodsName | 语法 | public String[] getAllGoodsName() | |
| 前置条件 | User经过验证有获得所有商品信息的权利，想获取所有商品名称 | |
| 后置条件 | 系统返回库存所有记录商品的名称 | |
| Stock.getMaxID | 语法 | public String getMaxID() | |
| 前置条件 | 新增商品时，需要对其进行编号，获取当前最大编号 | |
| 后置条件 | 系统返回当前记录中的商品最大编号 | |
| Stock.showStock | 语法 | public ArrayList<GoodsPO> showStock (String beginDate,String endDate) | |
| 前置条件 | User经过验证有查看库存的权利，输入了有效的时间段 | |
| 后置条件 | 系统根据输入的时间段显示查询到的该时间段内的库存商品信息 | |
| Stock.checkStock | 语法 | public ArrayList<GoodsPO>checkStock () | |
| 前置条件 | User经过验证有盘点库存的权利 | |
| 后置条件 | 系统显示当天的库存快照 | |
| Stock.add addStockOverOrLow | 语法 | public int addStockOverOrLow (StockOverOrLowVO vo) | |
| 前置条件 | 无 | |
| 后置条件 | 系统创建并存储库存报溢（损）单，返回操作成功与否 | |
| Stock.addStockError | 语法 | public int addStockError(StockErrorVO vo) | |
| 前置条件 | 库存商品数量低于警戒数量 | |
| 后置条件 | 系统生成库存报警单提示库存人员 | |
| Stock.showStockOverReceipt | 语法 | public ArrayList<StockOverOrLowVO> showStockOverReceipt() | |
| 前置条件 | 操作人员经过验证并想查看所有库存报溢单的信息 | |
| 后置条件 | 系统返回所有库存报溢单信息 | |
| Stock.showStockLowReceipt | 语法 | public ArrayList<StockOverOrLowVO> showStockLowReceipt() | |
| 前置条件 | 操作人员经过验证并想查看所有库存报溢单的信息 | |
| 后置条件 | 系统返回所有库存报损单的信息 | |
| Stock.getGoodsOverIncome | 语法 | public double getGoodsOverIncome(String beginDate, String endDate) | |
| 前置条件 | 操作人员经过验证并想获取某段时间段内的库存报溢收入，输入了有效的时间段 | |
| 后置条件 | 系统返回当前时间段内的库存报溢收入值 | |
| Stock.getGoodsLowCost | 语法 | public double getGoodsLowCost(String beginDate, String endDate) | |
| 前置条件 | 操作人员经过验证并想获取某段时间段内的库存报损支出，输入了有效的时间段 | |
| 后置条件 | 系统返回当前时间段内的库存报损支出值 | |
| Stock.isEnough | 语法 | public boolean isEnough(String ID, int num) | |
| 前置条件 | 操作人员经过验证且需要检查某种商品是否满足特定数量，输入了有效的商品ID和数量 | |
| 后置条件 | 系统检查改商品是否满足数量并返回Boolean值结果 | |
| Stock.setPrimeCostIncome | 语法 | public double setPrimeCostIncome(ArrayList<CommodityVO> commodityList) | |
| 前置条件 | 操作人员经过验证且有获得商品调价收入的权利，输入了有效的商品列表 | |
| 后置条件 | 系统根据输入的商品列表，返回这些商品总的调价收入 | |
| Stock.getGiftCost | 语法 | public double getGiftCost(String beginDate, String endDate) | |
| 前置条件 | 操作人员经过验证且想获得某段时间段内的库存赠送支出的权利，输入了有效的时间段 | |
| 后置条件 | 系统返回该时间段的库存赠送支出值 | |
| Stock.stockNumCheck | 语法 | public boolean stockNumCheck(String goodsID) | |
| 前置条件 | 操作人员经过验证且想检查当前某种商品是否已经库存报警，输入了有效的商品id | |
| 后置条件 | 系统检查当前库存是否报警 | |
| Stock.addGift | 语法 | public int addGift (GiftVO vo) | |
| 前置条件 | User经过验证有新建库存赠送单的权利，输入了有效的赠品信息，客户尚未因满足促销条件而获得库存赠送 | |
| 后置条件 | 系统创建并存储库存赠送单，并返回操作成功与否 | |
| **需要的服务（需接口）** | | | |
| 服务名 | | | 服务 |
| FileFactory.getStockData | | | 得到Stock文件的引用 |
| StockDataService.addGoodsClass (GoodsClassPO po) | | | 向外部文件中插入一个GoodsClassPo对象 |
| StockDataService.deleteGoodsClass (GoodsClassPO po) | | | 从文件中删除一个GoodsClassPO对象 |
| StockDataService.modifyGoodsClass (GoodsClassPO po) | | | 修改文件中对应的GoodsClassPo对象 |
| StockDataService.show() | | | 显示所有商品分类信息 |
| StockDataService.getMaxID() | | | 获得当前商品分类的最大编号 |
| StockDataService.showGoodsClassInfo(String name) | | | 显示某种商品分类的所有信息 |
| StockDataService.addGoods(GoodsPO po) | | | 向外部文件插入一个GoodsPo对象 |
| StockDataService. deleteGoods (GoodsPO po) | | | 从文件中删除一个GoodsPo对象 |
| StockDataService.modifyGoods (GoodsPO po) | | | 修改文件中对应的GoodsPo对象 |
| StockDataService.findGoods(String message) | | | 根据message返回文件中响应的商品的详细信息 |
| StockDataService.findByID(String id) | | | 根据商品编号返回该商品信息 |
| StockDataService.getMaxID() | | | 返回当前商品编号的最大值 |
| StockDataService.showGoods | | | 显示所有商品信息 |
| StockDataService.addStockError(StockErrorPO po) | | | 向文件中插入一个库存报警单对象 |
| StockDataService.getStockOverOrLowPO() | | | 返回所有库存报溢报损单 |
| StockDataService.getStockErrorPO() | | | 返回所有库存报警单 |
| StockDataService.getMaxID() | | | 获得当前记录的最大库存报溢（损）单的编号 |
| StockDataService.getErrorMaxID() | | | 获得当前记录的最大库存报警单编号 |
| StockDataService.findByID(String id) | | | 根据ID返回该ID的库存报溢（损）单 |
| StockDataService.excute(StockOverOrLowPO po) | | | 执行当前库存报溢（损）单 |
| StockDataService.addStockOverOrLow（StockOverOrLowPO po） | | | 向外部文件中插入一个StockOverOrLowPo对象 |
| StockDataService.addGift(GiftPO po) | | | 向外部文件中插入一个GiftPo对象 |
| StockDataService.dealGift(GiftPO po) | | | 处理库存赠送单 |
| StockDataService.getGiftList() | | | 返回所有的库存赠送单信息 |
| StockDataService.getGiftList(String beginDate, String endDate) | | | 返回某段时间内的库存赠送单的信息 |
| StockDataService.getMaxID() | | | 获得当前记录的库存赠送单的最大编号 |
| StockDataService.findByID(String id) | | | 根据ID返回该ID的库存赠送单 |
| StockDataService.modify(GiftPO po) | | | 修改当前库存赠送单的信息 |

**表11 memberbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| member. addMember | 语法 | public int addMember(MemberVO vo) | |
| 前置条件 | 需要添加客户时 | |
| 后置条件 | 向系统中添加新客户信息 | |
| member. deleteMember | 语法 | public int deleteMember(MemberVO vo) | |
| 前置条件 | 需要删除会员时 | |
| 后置条件 | 系统查找对应的客户信息，并删除MemberPO | |
| member. modifyMember | 语法 | public int modifyMember(MemberVO vo) | |
| 前置条件 | 需要修改会员信息时 | |
| 后置条件 | 系统查找对应客户信息，并修改 | |
| member. findMember | 语法 | public MemberVOfindMember(String message) | |
| 前置条件 | 需要查找客户时 | |
| 后置条件 | 根据输入的关键字查找客户，并返回客户信息 | |
| **需要的服务（需接口）** | | | |
| **服务名** | | | **服务** |
| FileFactory.getMemberData | | | 得到Member文件的引用 |
| memberDataService.add(MemberPO po) | | | 插入一个MemberPO对象 |
| memberDataService.delete(MemberPO po) | | | 删除一个MemberPO对象 |
| memberDataService.modify(MemberPO po) | | | 修改一个MemberPO对象 |
| memberDataService.find(String message) | | | 根据message返回对应MemberPo对象 |

**表12 salesbl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| sales.Purchase | 语法 | public int purchase (PurchaseVO vo); | |
| 前置条件 | 使用创建进货单功能时 | |
| 后置条件 | 系统中存储进货单信息 | |
| sales.PurchaseReturn | 语法 | public int purchaseReturn (PurchaseReturnVO vo); | |
| 前置条件 | 使用创建进货退货单功能时 | |
| 后置条件 | 系统中存储进货退货单信息 | |
| sales.Sale | 语法 | public int sale (SaleVO vo); | |
| 前置条件 | 使用创建销售单功能时 | |
| 后置条件 | 系统中存储销售单信息 | |
| sales.SaleReturn | 语法 | public int saleReturn (SaleReturnVO vo); | |
| 前置条件 | 使用创建销售退货单功能时 | |
| 后置条件 | 系统中存储销售退货单信息 | |
| sales.showSaleList | 语法 | public SaleListVO showSaleList() | |
| 前置条件 | 使用查看销售明细表功能时 | |
| 后置条件 | 系统返回销售明细表 | |
| **需要的服务（需接口）** | | | |
| **服务名** | | | **服务** |
| FileFactory.getSalesData | | | 得到Sales文件的引用 |
| salesDataService.getSaleList() | | | 返回一个SaleListPO对象 |
| salesDataService.purchase(PurchasePOpo) | | | 插入一个PurchasePO对象 |
| salesDataService.purchaseReturn(PurchaseReturnPOpo) | | | 插入一个PurchaseReturnPo对象 |
| salesDataService.Sale(SalePOpo) | | | 插入一个SalePO对象 |
| salesDataService.SaleReturn(SaleReturnPOpo) | | | 插入一个SaleReturnPO对象 |

**表13 financebl模块的接口规范**

|  |  |  |  |
| --- | --- | --- | --- |
| **提供的服务（供接口）** | | | |
| Finance.addAccount | 语法 | public intaddAccount(AccountVO vo) | |
| 前置条件 | User经验证有增加账户的权利，用户输入有效的账户信息 | |
| 后置条件 | 向账户列表中添加相应的账户 | |
| Finance.deleteAccount | 语法 | public int deleteAccount(AccountVO vo) | |
| 前置条件 | User经验证有删除账户的权利，用户输入有效的账户信息 | |
| 后置条件 | 向账户列表中删除相应的账户 | |
| Finance.modifyAccount | 语法 | public int modifyAccount(AccountVO vo) | |
| 前置条件 | User经验证有修改账户属性的权利，用户输入有效的账户信息 | |
| 后置条件 | 向账户列表中修改相应的账户 | |
| Finance.findAccount | 语法 | public AccountVO findAccount(AccountVO vo) | |
| 前置条件 | User经验证有查询账户的权利，用户输入查询对象 | |
| 后置条件 | 查找是否存在相同的账户，返回对应的账户信息 | |
| Finance.createCollection | 语法 | public int createCollection(CollectionVO vo) | |
| 前置条件 | User经验证有制定收款单的权利，用户输入有效的收款单信息 | |
| 后置条件 | 向系统输入新的收款单信息 | |
| Finance.createPayment | 语法 | public int createPayment(PaymentVO vo) | |
| 前置条件 | User经验证有制定付款单的权利，用户输入有效的付款单信息 | |
| 后置条件 | 向系统输入新的付款单信息 | |
| Finance.createCashlist | 语法 | public int createCashlist(CashlistVO vo) | |
| 前置条件 | User经验证有制定现金费用单的权利，用户输入有效的现金费用单信息 | |
| 后置条件 | 向系统输入新的现金费用单信息 | |
| Finance. viewBSL | 语法 | public BSLVO viewBSL() | |
| 前置条件 | User经验证有查看经营情况表的权利 | |
| 后置条件 | 系统显示一段时间收支信息 | |
| Finance.createRedExtrusion | 语法 | public int createRedExtrusion(ReceiptVo vo) | |
| 前置条件 | User经验证有红冲权利，用户需要红冲单据 | |
| 后置条件 | 系统生成一张红冲单,并在单据列表中显示 | |
| Finance.createRedExtrusionAndCopy | 语法 | public int createRedExtrusionAndCopy(ReceiptVo vo) | |
| 前置条件 | User经验证有红冲并复制的权利，用户需要红冲单据并复制 | |
| 后置条件 | 系统生成一张红冲单和一张以之前单子为模板的草稿单，并在单据列表中显示 | |
| Finance.initialInfo | 语法 | public int initialStock(BeginInfoVo vo) | |
| 前置条件 | User经验证有期初建账权利，用户需要期初建账 | |
| 后置条件 | 系统初始化一套账 | |
| Finance.showInfo | 语法 | public BeginInfoVO view(BeginInfoVO vo) | |
| 前置条件 | User经验证有查看套账权利，用户需要查看套账 | |
| 后置条件 | 系统显示套账信息 | |
| **需要的服务（需接口）** | | | |
| **服务名** | | | **服务** |
| FileFactory.getAccountData | | | 得到Account文件的引用 |
| FinanceDataService.findAccount(String name) | | | 根据输入的账户名查找单一持久化对象AccountPO |
| FinanceDataService.addAccount(AccountPO po) | | | 插入单一持久化对象AccountPO |
| FinanceDataService.deleteAccount(AccountPO po) | | | 删除单一持久化对象AccountPO |
| FinanceDataService.modifyAccount(AccountPO po) | | | 修改单一持久化对象AccountPO |
| FinanceDataService.createCollection(CollectionPO po) | | | 插入单一持久化对象CollectionPO |
| FinanceDataService.createPayment(PaymentPO po) | | | 插入单一持久化对象PaymentPO |
| FinanceDataService.createCashlist(CashlistPO po) | | | 插入单一持久化对象CashlistPO |
| Finance.view() | | | 返回一个BSLPO对象 |
| FinanceDataService.initInfo(BeginInfoVO vo) | | | 在文件中添加一套账的初始化信息BeginInfoPO |
| FinanceDaraService.getInfo(BeginInfoVO vo) | | | 返回一个BeginInfoPO对象 |

**表14 receiptbl模块的接口规范**

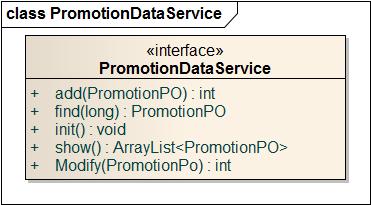
|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Receipt.Add | 语法 | Public int Add(ReceiptVo vo) |
| 前置条件 | User经验证，可以创建单据 |
| 后置条件 | 向待审批单据列表库中添加该单据 |
| Receipt.Modify | 语法 | Public int Modify(String id) |
| 前置条件 | User选中了有效单据 |
| 后置条件 | 更新该单据信息 |
| Receipt.Batch | 语法 | Public int Batch(String[] id) |
| 前置条件 | 当前有待审批单据，且user已选中单据 |
| 后置条件 | 更新该单据组信息 |
| Receipt.Send | 语法 | Public void Send(ReceiptVo vo) |
| 前置条件 | 该业务单据创建成功 |
| 后置条件 | 持久化更新涉及的领域对象的数据 |
| Receipt.Reply | 语法 | Public void Reply(String id,) |
| 前置条件 | 该单据审批完成 |
| 后置条件 | 该单据制单人收到审批结果信息 |
| Receipt.View | 语法 | Public ArrayList<ReceiptVO> view() |
| 前置条件 | 已有单据记录 |
| 后置条件 | 返回所有单据信息 |
| Receipt.refresh | 语法 | Public ArrayList<ReceiptVO> refresh() |
| 前置条件 | 已有单据记录 |
| 后置条件 | 返回所有待审批单据信息 |
| Receipt.approve | 语法 | Public int approve(String id) |
|  | 前置条件 | 有单据提交待审批 |
|  | 后置条件 | 更新该单据信息 |
| 需要的服务(需接口) | | |
| ReceiptDataService.Add(ReceiptPo po) | | 插入单一持久化对象 |
| ReceiptDataService.Modify(ReceiptPo po) | | 更新单一持久化对象 |
| ReceiptDataService.Modify(ReceiptPo[] po) | | 更新单一持久化对象组 |
| ReceiptDataService.find(Type type) | | 根据类型type查找ReceiptPo对象 |
| ReceiptDataService.find(Date date) | | 根据日期date查找ReceiptPo对象 |
| ReceiptDataService.Show() | | 得到所得的ReceiptPo对象 |
| FileFactory.getReceiptData | | 得到Receipt文件的引用 |

**表15 promotionbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| Promotion.Add | 语法 | Public int Add(PromotionVo vo) |
| 前置条件 | 启动一个策略制定会合 |
| 后置条件 | 保存该促销策略信息 |
| Promotion.Modify | 语法 | Public int Modify(PromotionVo vo) |
| 前置条件 | 系统中有可修改促销策略 |
| 后置条件 | 更新该促销策略信息 |
| Promotion.Match | 语法 | Public PromotionVo Match(SaleVO vo) |
| 前置条件 | 有销售发生 |
| 后置条件 | 返回符合的促销策略信息 |
| Promotion.Present | 语法 | Public GiftVo Present(PromotionVo vo) |
| 前置条件 | 有符合促销策略的销售发生 |
| 后置条件 | 返回库存赠送但信息 |
| Promotion.Show | 语法 | Public ArrayList<PromotionVO> Show() |
| 前置条件 | 存在创建成功的促销策略 |
| 后置条件 | 持久化更新涉及的领域对象的数据 |
| 需要的服务(需接口) | | |
| PromotionDataService.Add(PromotionPo po) | | 插入单一持久化对象 |
| PromotionDataService.Modify(PromotionPo po) | | 修改一个PromotionPo对象 |
| PromotionDataService.Show () | | 得到所有的PromotionPo对象 |
| PromotionDataService.find(String id) | | 根据ID查找单一持久化对象 |
| FileFactory.getPromotionData | | 得到Promotion文件的引用 |

**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。这些服务由DataService接口提供。在本系统中我们采用序列化文件的形式保存持久化数据。具体如图12—17所示。



**图12 数据层模块（PromotionDataService）的描述**



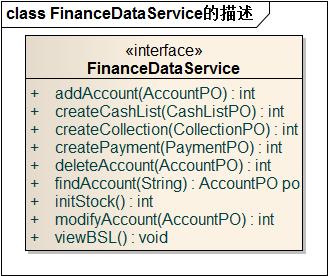
**图13 数据层模块（StockDataService）的描述**



**图14 数据层模块（SalesDataService）的描述**



**图15 数据层模块（MemberDataService）的描述**



**图16 数据层模块（FinanceDataService）的描述**



**图17 数据层模块（ReceiptDataService）的描述**

**5.4.1数据层模块的职责**

数据层模块的职责如表16－22所示。

**表16 数据层模块（UserDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| UserDataService | 持久化外部文件的接口，提供集体载入、集体保存、增、删、改、查服务 |

**表17 数据层模块（StockDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| StockDataService | 持久化外部文件的接口，提供集体载入、集体保存、增、删、改、查服务服务 |

**表18 数据层模块（SalesDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| salesDataService | 持久化外部文件的接口，提供集体载入、集体保存、增、删、改、查服务 |

**表19数据层模块（MemberDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| memberDataService | 持久化外部文件的接口，提供集体载入、集体保存、增、删、改、查服务 |

**表20数据层模块（FinanceDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| FinanceDataService | 持久化外部文件的接口，提供集体载入、集体保存、增、删、改、查服务 |

**表21 数据层模块（ReceiptDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| ReceiptDataService | 持久化外部文件的接口，提供集体载入、集体保存、增、删、改、查服务 |

**表22 数据层模块（PromotionDataService）的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| PromotionDataService | 持久化外部文件的接口，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**

数据库模块的接口规范如表23－29所示

**表23 数据层模块（UserDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserDataService.add | 语法 | public int add(UserPO po) |
| 前置条件 | 同样ID的po在文件中不存在 |
| 后置条件 | 在外部文件中增加一个po记录 |
| UserDataService.check | 语法 | public boolean check(UserPO po) |
| 前置条件 | 无 |
| 后置条件 | 返回用户名和密码是否匹配 |
| UserDataService.delete | 语法 | public int delete(UserPO po) |
| 前置条件 | 在外部文件中存在某一个ID 的po需要删除 |
| 后置条件 | 删除一个po对象 |
| UserDataService.modify | 语法 | public int modify(UserPO po) |
| 前置条件 | 在外部文件中存在某一个ID 的po需要更新 |
| 后置条件 | 更新一个po对象 |
| UserDataService.showUserInfo | 语法 | public UserPO showUserInfo (String ID) |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的UserPO的结果 |

**表24数据层模块（StockDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| StockDataService. addGoodsClass | 语法 | public int addGoodsClass (GoodsClassPO po) |
| 前置条件 | 需要新建一个商品分类 |
| 后置条件 | 向外部文件中添加一个GoodsClassPo |
| StockDataService. deleteGoodsClass | 语法 | public int deleteGoodsClass (GoodsClassPO po) |
| 前置条件 | 外部一个商品分类需要删除 |
| 后置条件 | 删除文件中的一个GoodsClasspO |
| StockDataService. modifyGoodsClass | 语法 | public int modifyGoodsClass (GoodsClassPO po); |
| 前置条件 | 外部一个商品分类需要修改 |
| 后置条件 | 修改文件中一个GoodsClassPo |
| StockDataService. show | 语法 | public ArrayList<GoodsClassPO> show() |
| 前置条件 | 需要显示所有商品分类信息 |
| 后置条件 | 返回记录的所有商品分类信息 |
| StockDataService. getMaxID | 语法 | public String getMaxID() |
| 前置条件 | 需要得到当前记录的最大的分类编号 |
| 后置条件 | 返回当前记录的最大的分类编号 |
| StockDataService.showGoodsClassInfo | 语法 | public GoodsClassPO showGoodsClassInfo(String name) |
| 前置条件 | 需要显示某个名字的商品分类PO |
| 后置条件 | 返回该商品分类的详细信息 |
| StockDataService. addGoods | 语法 | public int addGoods(GoodsPO po) |
| 前置条件 | 需要新增一个商品 |
| 后置条件 | 向外部文件中添加一个GoodsPO对象 |
| StockDataService. deleteGoods | 语法 | public int deleteGoods (GoodsPO po) |
| 前置条件 | 外部一个商品需要被删除 |
| 后置条件 | 删除文件中的一个GoodsPO |
| StockDataService. modifyGoods | 语法 | public int modifyGoods (GoodsPO po) |
| 前置条件 | 外部一个商品需要修改 |
| 后置条件 | 修改文件中一个GoodsPO |
| StockDataService. findGoods | 语法 | public ArrayList<GoodsPO>findGoods (String message) |
| 前置条件 | 需要根据关键字来查找商品 |
| 后置条件 | 返回对应的GoodsPO对象 |
| StockDataService.findByID | 语法 | public GoodsPO findByID(String id) |
| 前置条件 | 外部清求显示某个ID的商品 |
| 后置条件 | 返回该ID的商品PO |
| StockDataService.getMaxID | 语法 | public String getMaxID() |
| 前置条件 | 需要获得当前记录的商品最大编号 |
| 后置条件 | 返回当前记录的商品最大编号 |
| StockDataService.showGoods | 语法 | public ArrayList<GoodsPO> showGoods() |
| 前置条件 | 需要显示所有商品信息 |
| 后置条件 | 返回记录的所有商品信息 |
| StockDataService. showStock | 语法 | public ArrayList<GoodsPO>showStock (String beginDate,String endDate) |
| 前置条件 | 需要查看一个时间段内的库存商品信息 |
| 后置条件 | 返回该时间段内的库存商品信息 |
| StockDataService. checkStock | 语法 | public ArrayList<GoodsPO>checkStock(SalesPO po) |
| 前置条件 | 需要盘点当天的库存 |
| 后置条件 | 返回当天的库存快照 |
| StockDataService. addStockOverOrLow | 语法 | public int addStockOverOrLow(StockOverOrLowPO po) |
| 前置条件 | 无 |
| 后置条件 | 向外部文件中添加一个StockOverOrLowPO |
| StockDataService.getStockOverOrLowPO | 语法 | public ArrayList<StockOverOrLowPO> getStockOverOrLowPO() |
| 前置条件 | 需要获得所有库存报溢报损单 |
| 后置条件 | 返回当前记录的所有库存报溢报损单 |
| StockDataService.getStockErrorPO | 语法 | public ArrayList<StockErrorPO> getStockErrorPO() |
| 前置条件 | 需要得到所有库存报警单 |
| 后置条件 | 返回当前记录的所有库存报警单 |
| StockDataService.getMaxID | 语法 | public String getMaxID() |
| 前置条件 | 需要得到当前记录的库存报溢报损单的最大编号 |
| 后置条件 | 返回当前记录的最大编号 |
| StockDataService.getMaxErrorID | 语法 | public String getErrorMaxID() |
| 前置条件 | 需要得到当前记录的最大报警单编号 |
| 后置条件 | 返回当前记录的最大报警单编号 |
| StockDataService.findByID | 语法 | public StockOverOrLowPO findByID(String id) |
| 前置条件 | 需要显示某个ID的库存报溢报损单 |
| 后置条件 | 返回该ID的单据的详细信息 |
| StockDataService.excute | 语法 | public int excute(StockOverOrLowPO po) |
| 前置条件 | 执行库存报溢报损单 |
| 后置条件 | 系统修改单据状态为执行完毕 |
| StockDataService. addGift | 语法 | public int addGift(GiftPO po) |
| 前置条件 | 需要进行一次库存赠送 |
| 后置条件 | 向外部文件中添加一个GiftPO |
| StockDataService. addStockError | 语法 | public int addStockError (StockErrorPO po) |
| 前置条件 | 无 |
| 后置条件 | 向外部文件中写入一个StockErrorPO |
| StockDataService.dealGift | 语法 | public int dealGift(GiftPO po) |
| 前置条件 | 需要处理一个库存赠送单 |
| 后置条件 | 系统修改库存赠送单状态为已执行 |
| StockDataService.getGiftList | 语法 | public ArrayList<GiftPO> getGiftList() |
| 前置条件 | 需要得到所有库存赠送单 |
| 后置条件 | 返回所有库存赠送单信息 |
| StockDataService.getGiftList | 语法 | public ArrayList<GiftPO> getGiftList(String beginDate, String endDate) |
| 前置条件 | 需要得到某个时间段的库存赠送单 |
| 后置条件 | 返回该时间段的库存赠送单的信息 |
| StockDataService.getMaxID | 语法 | public String getMaxID() |
| 前置条件 | 需要得到当前记录的库存赠送的最大编号 |
| 后置条件 | 返回当前记录的库存赠送单的最大编号 |
| StockDataService.findByID | 语法 | public GiftPO findByID(String id) |
| 前置条件 | 需要显示某个ID的库存赠送单 |
| 后置条件 | 返回该编号的库存赠送单的PO |
| StockDataService.modify | 语法 | public int modify(GiftPO po) |
| 前置条件 | 需要修改某个库存赠送单 |
| 后置条件 | 系统修改这个库存赠送单并存储 |

**表25数据层模块（SalesDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| salesDataService. getSaleList | 语法 | public saleListPO getSaleList() |
| 前置条件 | 请求查看销售明细表 |
| 后置条件 | 返回一个销售明细表PO |
| salesDataService. Purchase | 语法 | public int purchase(PurchasePO po) |
| 前置条件 | 符合进货条件 |
| 后置条件 | 存储一个PurchasePO对象 |
| salesDataService. PurchaseReturn | 语法 | public int purchaseReturn(PurchaseReturnPO po) |
| 前置条件 | 符合退货条件 |
| 后置条件 | 存储一个PurchaseReturnPO 对象 |
| salesDataService. Sale | 语法 | public int sale(SalePO po) |
| 前置条件 | 符合销售条件 |
| 后置条件 | 存储一个SalePO对象 |
| salesDataService. SaleReturn | 语法 | public int saleReturn(SaleReturnPO po) |
| 前置条件 | 符合退货条件 |
| 后置条件 | 存储一个SaleReturnPO对象 |

**表26 数据层模块（MemberDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| memberDataService.add | 语法 | public int add(MemberPO po) |
| 前置条件 | member不重名 |
| 后置条件 | 存储一个memberPO对象 |
| memberDataService.delete | 语法 | public int delete(MemberPO po) |
| 前置条件 | 该member不存在应收应付 |
| 后置条件 | 删除一个memberPO对象 |
| memberDataService.modify | 语法 | public int modify(MemberPO po) |
| 前置条件 | member存在 |
| 后置条件 | 修改一个memberPO对象信息 |
| memberDataService. find | 语法 | public MemberPO find(String message) |
| 前置条件 | 用户输入关键词 |
| 后置条件 | 反馈出memberPO信息 |

**表27 数据层模块（FiananceDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| FinanceDataService.findAccount | 语法 | public AccountPO findAccount(Account PO) throws RemoteException; |
| 前置条件 | 同样名字的AccountPO在Account中存在 |
| 后置条件 | 按账户名查找并返回相应的AccountPO记录 |
| FinanceDataService.addAccount | 语法 | public int addAccount(AccountPO po) throws RemoteException; |
| 前置条件 | 同样名字的AccountPO在Account中不存在 |
| 后置条件 | 在Account中增加一个AccountPO记录 |
| FinanceDataService.deleteAccount | 语法 | public int deleteAccount(AccountPO po) throws RemoteException; |
| 前置条件 | 在Account中存在某个AccountPO需要删除 |
| 后置条件 | 删除一个AccountPO记录 |
| FinanceDataService.modifyAccount | 语法 | public int modifyAccount(AccountPO po) throws RemoteException; |
| 前置条件 | 在Account中存在某个的AccountPO需要修改 |
| 后置条件 | 修改一个AccountPO记录 |
| FinanceDataService.createCollection | 语法 | public int createCollection (CollectionPO po) throws RemoteException; |
| 前置条件 | 符合收款条件 |
| 后置条件 | 存储一张收款单CollectionPO |
| FinanceDataService.createPayment | 语法 | public int createPayment(PaymentPO po) throws RemoteException |
| 前置条件 | 符合收款条件 |
| 后置条件 | 存储一张付款单PaymentPO |
| FinanceDataService.createCashlist | 语法 | public int createCashlist (CashlistPO po) throws RemoteException |
| 前置条件 | 符合创建现金费用单条件 |
| 后置条件 | 存储一张现金费用单CashlistPO |
| FinanceDataService.viewBSL | 语法 | Public BSLPO viewBSL() throws RemoteException |
| 前置条件 | 符合查看经营情况表条件 |
|  | 返回一个BSLPO记录 |
| FinanceDataService.initInfo | 语法 | public int initInfo(BeginInfoVO vo) throws RemoteException); |
| 前置条件 | 无 |
| 后置条件 | 存储期初信息 |
| FinanceDataService.getInfo | 语法 | Public BeginInfoPO getInfo(BeginInfoVO vo) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回一个BeginInfoPO记录 |

**表28 数据层模块（ReceiptDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| ReceiptDataService.Add | 语法 | Public int Add(ReceiptPo po) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 在数据库中增加一个po记录 |
| ReceiptDataService.find | 语法 | Public ReceiptPO find(String id)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按ID进行查找返回相应的ReceiptPo()结果 |
| ReceiptDataService.modify | 语法 | Public int Modify(ReceiptPo po)throws RemoteException |
| 前置条件 | 该单据id存在 |
| 后置条件 | 修改一个po |
| ReceiptDataService.show | 语法 | Public ArrayList<ReceiptPO> show()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的po |
| ReceiptDataService.init | 语法 | Public void init()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 初始持久化数据库的使用 |

**表29 数据层模块（PromotionDataService）的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务(供接口) | | |
| PromotionDataService.Add | 语法 | Public int Add(PromotionPo po) throws RemoteException; |
| 前置条件 | 无 |
| 后置条件 | 在数据库中增加一个po记录 |
| PromotionDataService.find | 语法 | Public PromotionPO find(String 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.Modify | 语法 | public int Modify(PromotionPo po) throws RemoteException; |
| 前置条件 | 存在该id的PromotionPO对象 |
| 后置条件 | 更新PromotionPo |
| PromotionDataService.show | 语法 | Public ArrayList<PromotionPO> show()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回所有的po |
| PromotionDataService.init | 语法 | Public void init()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 初始持久化数据库的使用 |

**5.5客户端启动模块的分解**

客户端启动模块逻辑上是一个横跨客户端各层的模块，为客户端各层提供初始化启动服务。客户端启动模块主要包含主程序类ClientStarter。主程序类是整个程序的入口，负责整个程序的展示层初始化、业务逻辑层初始化和网络初始化。其结构如图18所示。



**图18客户端启动模块结构图**

**5.5.1 客户端启动模块的职责**

如表30所示为客户端启动模块的职责。

**表30 客户端启动模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| ClientStarter | 程序入口。负责展示层初始化、业务逻辑层初始化和网络初始化。 |

**5.5.2客户端启动模块的接口规范**

客户端启动模块的接口规范如表31所示。

**表31 客户端启动模块的接口规范**

|  |  |  |
| --- | --- | --- |
| ClientStarter | 语法 | main（args:String[]） |
| 前置条件 | 无 |
| 后置条件 | 根据配置初始化好相应的网络连接并进行展示层和业务逻辑层初始化。 |

客户端启动模块需要的服务接口如表32所示。

**表32 客户端启动模块的服务接口**

|  |  |
| --- | --- |
| **服务名** | **服务** |
| java.rmi.LocateRegistry.createRegistry() | 启动RMI注册服务 |
| java.rmi.Naming.rebind() | 注册RMI服务 |

**5.5.3客户端启动模块设计原理**

启动模块是一个独立于分层体系之外的模块，可以直接访问各层的接口，完成系统启动的需求。

**5.6服务器端启动模块的分解**

服务器端启动模块逻辑上是一个横跨服务器端各层的模块，为服务器端各层提供初始化启动服务。服务器端启动模块主要包含主程序类SeverStarter。主程序类是整个程序的入口，负责整个程序的网络初始化和外部文件初始化。其结构如图19所示。



**图19 服务器端启动模块结构图**

**5.6.1 服务器端启动模块的职责**

如表33所示为服务器端启动模块的职责。

**表33 服务器端启动模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| ServerStarter | 程序入口。负责网络初始化和外部文件初始化。 |

**5.6.2 服务器端启动模块的接口规范**

服务器端启动模块的接口规范如表34所示。

**表34服务器端启动模块的接口规范**

|  |  |  |
| --- | --- | --- |
| ServerStarter | 语法 | main（args:String[]） |
| 前置条件 | 无 |
| 后置条件 | 根据配置初始化好相应的网络连接和进行序列化文件初始化。 |

服务器端启动模块需要的服务接口如表35所示。

**表35服务器端启动模块的服务接口**

|  |  |
| --- | --- |
| **服务名** | **服务** |
| java.rmi.LocateRegistry.createRegistry() | 启动RMI注册服务 |
| java.rmi.Naming.rebind() | 注册RMI服务 |

**5.6.3服务器端启动模块设计原理**

启动模块是一个独立于分层体系之外的模块，可以直接访问各层的接口，完成系统启动的需求。

**5.7客户端网络模块的分解**

由于数据是存储在服务器上的，客户端访问数据必须通过网络通信层，具体是由客户端的业务逻辑层和服务器端的数据层通过网络通信层进行通信。如图19所示。

具体网络通信层的实现是通过远程方法调用RMI来实现的。



**图20客户端数据通信层模块的描述**

其实网络通信层的实际运作代码是由RMI API所完成的，在客户端我们只需要部署RMI服务的Stub文件和Java RMI类库即可。如表36所示。

**表36网络通信层模块的职责**

|  |  |
| --- | --- |
| **模块** | **职责** |
| IDataRemoteService\_Stub | 给业务逻辑层提供服务接口 |
| Java RMI 类库 | 实现RMI机制 |

**5.8服务器端网络模块的分解**

由于数据是存储在服务器上的，客户端访问数据必须通过网络通信层，具体是由客户端的业务逻辑层和服务器端的数据层通过网络通信层进行通信。

具体服务器端模块中只需要包含Java RMI类库即可。但是其数据层实现RMI的类需要符合以下条件：



**图21 服务器数据层实现RMI的类的描述**

DatabaseService 是被业务逻辑层调用的接口。为了实现RMI， DatabaseService需要继承Remote接口，而数据层中的实现这个接口的类SerService都需要继承UnicastRemoteObject类。

以下是服务器网络模块的RMI详细的规范：

**表37服务器网络模块的RMI接口规范**

|  |  |
| --- | --- |
| **类/接口** | **规范** |
| 远程服务接口 | extends Remote，方法 throws RemoteException |
| 远程服务的实现类 | extendsUnicastRemoteObject |

**表38服务器网络模块的RMI接口返回值的规范**

|  |  |
| --- | --- |
| **类/接口** | **规范** |
| 作为远程引用传递 | 该类需要extends Remote，方法 throws RemoteException |
| 通过序列化文件进行复制 | Implements SerFile |

**6、信息视角**

**6.1 数据持久化对象**

系统的PO类就是对应的相应的实体类，在此只做简单的介绍。

* UserPO类包含用户的用户名、密码属性。
* SalesPO类保存销售和进货时的数据的类，包含单据编号，名称，型号，供应商，仓库，操作员，业务员，商品，价格，数量，备注属性。
* StockPO类保存商品分类，商品，和库存的各项信息，包括商品分类编号，商品编号，名称，型号，数量，进价，零售价，库存均价，批次，批号，出厂日期，审批状态属性。
* FinancePO类保存账户，收付款，期初建账时的数据的类，包含账户名称，金额，单据编号，客户，操作员，备注，条目名，单据类型，业务员，仓库，时间区间，商品名称，类型，型号，进价，售价属性。
* ReceiptPO类包含单据的各项信息，包含单据编号，客户，操作员，备注，单据类型，审批状态，创建时间，业务员，仓库属性。
* PromotionPO类保存促销策略的各项信息，包含促销起始日，促销结束日，礼品编号，礼品数量，商品ID，代金券金额，代金券编号，审批状态属性。
* MemberPO类保存客户的信息，包含客户编号，分类，级别，姓名，电话，地址，邮编，电子邮箱，应收额度，应收，应付，默认业务员。

持久化用户对象UserPO的定义如图21所示。

public class UserPO{

String id;

String name;

String password;

UserRole role;

public UserPO(String I, String n, String p,UserRole r){

id = i;

name = n;

password = p;

role = r;

}

public String getName(){

return name;

}

public String getID(){

return id;

}

public String getPassword(){

return password;

}

Public UserRole getRole(){

Return role;

}

}

**图持久化用户对象UserPO的定义**

**图22 持久化用户对象UserPO的定义**

**6.2序列化文件列表**

数据库中包含User.ser、Stock.ser、Member.ser、Sales.ser、Promotion.ser、Receipt.ser、Finance.ser