

体系结构描述文档

灯具照明行业进销存管理系统



2014-10-14

小百合工作室

更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人员 | 日期 | 变更原因 | 版本号 |
| 陈云龙 | 2014-10-14 | 最初草稿 | V1.0 |
| 陈云龙 | 2014-10-15 | 部分文档 | V1.1 |
| 曾婧 | 2014-10-19 | 添加开发包图 | V1.2 |
| 景诗维  陈云龙 | 2014-10-22 | 添加用户界面层分解、数据层分解和信息视角 | V1.3 |
| 曾婧 | 2014-10-24 | 添加了业务逻辑层的分解 | V1.4 |
| 刘磊 | 2014-10-25 | 添加了模块的职责 | V1.5 |
|  |  |  |  |
|  |  |  |  |

目录

[1. 引言 1](#_Toc401802261)

[1.1. 编制目的 1](#_Toc401802262)

[2. 产品描述 1](#_Toc401802263)

[3. 逻辑视角 1](#_Toc401802264)

[4. 组合视角 2](#_Toc401802265)

[4.1. 开发包图 2](#_Toc401802266)

[4.2. 运行时进程 6](#_Toc401802267)

[4.3. 物理部署 7](#_Toc401802268)

[5. 接口视角 7](#_Toc401802269)

[5.1. 模块的职责 7](#_Toc401802270)

[5.2. 用户界面层的分解 7](#_Toc401802271)

[5.2.1. 用户界面层模块的职责 8](#_Toc401802272)

[5.2.2. 用户界面模块的接口规范 8](#_Toc401802273)

[5.2.3. 用户界面模块设计原理 9](#_Toc401802274)

[5.3. 业务逻辑层的分解 9](#_Toc401802275)

[5.4. 数据层的分解 9](#_Toc401802276)

[5.4.1. 数据层模块的职责 9](#_Toc401802277)

[5.4.2. 数据层模块的接口规范 9](#_Toc401802278)

[6. 信息视角 13](#_Toc401802279)

[6.1. 数据持久化对象 13](#_Toc401802280)

[6.2. 文件格式 14](#_Toc401802281)

# 引言

## 编制目的

本报告详细完成对灯具照明行业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编写，是了解系统的导航。

# 产品描述

参考灯具照明行业进销存系统用例文档和灯具照明行业进销存系统软件规格说明中对产品的概括描述。

# 逻辑视角

灯具照明行业进销存系统中，选择了分层体系结构的风格，将系统分为3层（展示层、业务逻辑层、数据层）能够很好的示意整个高层抽象。展示层包括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和图二所示。

图 1参照体系结构风格的暴徒表达逻辑视角

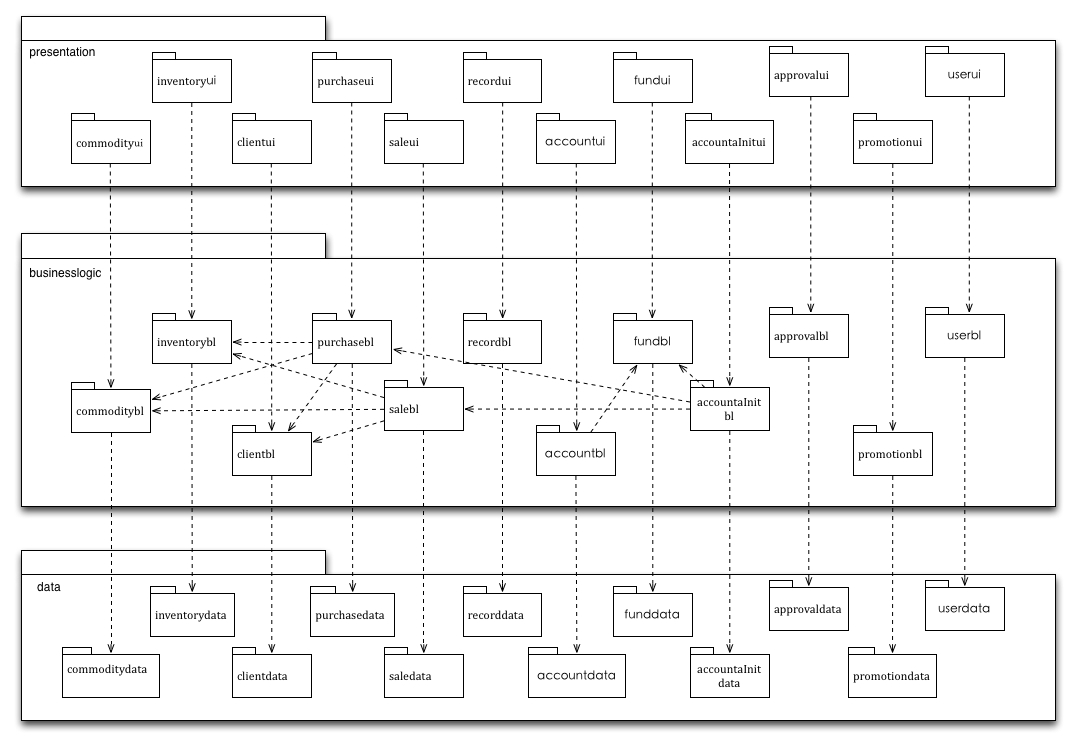


图 2 软件体系结构逻辑设计方案

# 组合视角

## 开发包图

灯具照明行业进销存系统的最终开发包设计如表1所示

**表1 灯具照明行业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 | commodityui, inventoryui, clientui, purchaseui, saleui, recordui, accountui, approvalui, promotionui, accountainitui, userui, fundui |
| commodityui | commodityblservice, 界面类库包 |
| commodityblservice |  |
| commoditybl | commodityblservice, po |
| commoditydataservice | Java RMI, po |
| commoditydata | commoditydataservice, databaseutility |
| inventoryui | inventoryblservice, 界面类库包 |
| inventoryblservice |  |
| inventorybl | inventoryblservice, po |
| inventorydataservice | Java RMI, po |
| inventorydata | inventorydataservice, databaseutility |
| clientui | clientblservice, 界面类库包 |
| clientblservice |  |
| clientbl | clientblservice, po |
| clientdataservice | Java RMI, po |
| clientdata | clientdataservice, databaseutility |
| purchaseui | purchaseblservice, 界面类库包 |
| purchaseblservice |  |
| purchasebl | purchaseblservice,inventorybl,commoditybl, clientbl, po |
| purchasedataservice | Java RMI, po |
| purchasedata | purchasedataservice, databaseutility |
| saleui | saleblservice, 界面类库包 |
| saleblservice |  |
| salebl | saleblservice, inventorybl, commoditybl, clientbl, po |
| saledataservice | Java RMI, po |
| saledata | saledataservice, databaseutility |
| recordui | recordblservice, 界面类库包 |
| recordblservice |  |
| recordbl | recordblservice, po |
| recorddataservice | Java RMI, po |
| recorddata | recorddataservice, databaseutility |
| accountui | accountblservice, 界面类库包 |
| accountblservice |  |
| accountbl | accountblservice, fundbl, po |
| accountdataservice | Java RMI, po |
| accountdata | accountdataservice, databaseutility |
| fundui | fundblservice, 界面类库包 |
| fundblservice |  |
| fundbl | fundblservice, po |
| funddataservice | Java RMI, po |
| funddata | funddataservice, databaseutility |
| accountaintui | accountaintblservice, 界面类库包 |
| accountaintblservice |  |
| accountaintbl | accountaintblservice, fundbl, purchasebl, salebl, po |
| accountaintdataservice | Java RMI, po |
| accountaintdata | accountaintdataservice, databaseutility |
| approvalui | approvalblservice, 界面类库包 |
| approvalblservice |  |
| approvalbl | approvalblservice, po |
| approvaldataservice | Java RMI, po |
| approvaldata | approvaldataservice, databaseutility |
| promotionui | promotionblservice, 界面类库包 |
| promotionblservice |  |
| promotionbl | promotionblservice, po |
| promotiondataservice | Java RMI, po |
| promotiondata | promotiondataservice, databaseutility |
| userui | userblservice, 界面类库包 |
| userblservice |  |
| userbl | userblservice, po |
| userdataservice | Java RMI, po |
| userdata | userdataservice, databaseutility |
| vo |  |
| po |  |
| utilitybl |  |
| 界面类库包 |  |
| Java RMI |  |
| databaseutility |  |

****

图 3 客户端开发包图



图 4 服务器端开发包图

## 运行时进程

在灯具照明行业进销存系统中，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，服务器端进程是在服务器端机器上运行。



图 5 进程图

## 物理部署

灯具照明行业进销存系统中客户端构件是放在客户端机上，服务器端构件是放在服务器端机器上。在客户端节点上，还要部署RMIStub构件。由于Java RMI 构件属于JDK 1.8的一部分。所以，在系统JDK环境已经设置好的情况下，不需要独立部署。部署图如图6所示。



图 6 部署图

# 接口视角

## 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。

启动模块

启动模块

展示层

网络模块

业务逻辑层

数据层

网络模块

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 | 基于窗口的进销存管理系统客户端用户界面 |
| 业务逻辑层 | 对于用户界面的输入进行响应并进行业务处理逻辑 |
| 客户端网络模块 | 利用java RMI机制查找RMI服务 |

**表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 |
| commodityblservice  inventoryblservice  clientblservice  purchaseblservice  saleblservice  recordblservice  accountblservice  fundblservice  accountaintblservice  approvalblservice  promotionblservice  userblservice | 客户端展示层 | 客户端业务逻辑层 |
| commoditydataservice  inventorydataservice  clientdataservice  purchasedataservice  saledataservice  recorddataservice  accountdataservice  funddataservice  accountaintdataservice  approvaldataservice  promotiondataservice  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一层依赖了一个接口(需接口)，而下层实现这个接口(供接口)。saleblservice提供了Sales界面所需要的所有业务逻辑功能。saledataservice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。



Panel(sale)

sale

<<interface>>

saledataservice

+ find(String id):void

+ insert(PurchasePO po):void

+ delete(PurchasePO po):void

+ update(PurchasePO po)：void

+ init():void

saledataservice???

图9 销售用例层之间调用的接口

## 用户界面层的分解

系统存在30个用户界面,他们分别为登录界面、库存管理人员界面组（库存管理人员主界面、商品分类管理界面、商品管理界面、库存查看界面、商品详细信息界面、商品分类详细信息界面、库存赠送界面、警报类单据界面以及库存盘点界面）、进货销售人员界面组（进货销售人员主界面、制定进货单界面、制定进货退货单界面、制定销售单界面、制定销售退货单界面、客户管理界面以及历史单据查看界面）、财务人员界面组（财务人员主界面、制定现金单界面、制定收款单界面、制定付款单界面、销售明细表界面、经营历程表界面、经营情况表界面、账户管理界面、期初建账界面）、总经理界面组（总经理主界面、审批单据以及促销策略界面）以及管理员界面组（用户权限与信息管理界面）。其用户界面之间的跳转如下图所示：



其中所有用户界面通过同一个类处理，即主用户界面类（MainFrame）



### 用户界面层模块的职责

MainFrame主要用于负责界面的显示和界面之间的跳转工作。

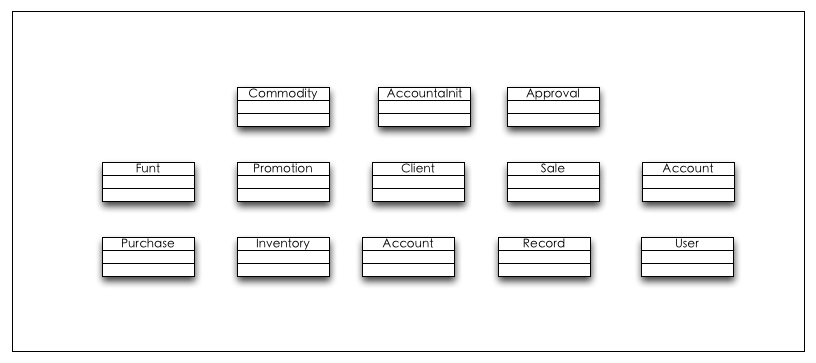
### 用户界面模块的接口规范

|  |  |  |
| --- | --- | --- |
| MainFrame | 语法 | init(args:String[]) |
| 前置条件 | 用户触发相应需要显示界面的条件 |
| 后置条件 | 显示Frame并且加载Panel |

### 用户界面模块设计原理

用户界面你用java的Swing和AWT库来实现。

## 业务逻辑层的分解



### 业务逻辑层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| commoditybl | 负责实现商品与商品管理界面所需要的服务 |
| inventorybl | 负责实现仓库管理界面所需要的服务 |
| clientbl | 负责实现客户管理界面所需要的服务 |
| purchasebl | 负责实现进货界面所需要的服务 |
| salebl | 负责实现销售界面所需要的服务 |
| recordbl | 负责实现查看表单界面所需要的服务 |
| accountbl | 负责实现公司账户界面所需要的服务 |
| funtbl | 负责实现资金管理界面所需要的服务 |
| accountainit | 负责实现建账界面所需要的服务 |
| approvalbl | 负责实现审批界面所需要的服务 |
| promotionbl | 负责实现销售策略界面所需要的服务 |
| userbl | 负责实现用户管理界面所需要的服务 |

### 业务逻辑模块的接口模范

**commoditybl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Commodity.addCommoSort | 语法 | public ResultMessage addCommoSort  (String sortName, CommoditySort parentSort) |
| 前置条件 | 分类名称输入合法，并且父类中没有添加商品 |
| 后置条件 | 添加新的商品分类，持久化更新涉及的对象的数据 |
| Commodity.deletCommoSort | 语法 | public ResultMessage deleteCommoSort(  String sortName) |
| 前置条件 | 分类名称输入合法 |
| 后置条件 | 删除商品分类，持久化更新涉及的对象的数据 |
| Commodity.updCommoSort | 语法 | public ResultMessage updCommoSort  (Sring sortName) |
| 前置条件 | 分类名称输入合法 |
| 后置条件 | 更改商品分类，持久化更新涉及的对象的数据 |
| Commodity.addCommo | 语法 | public ResultMessage addCommo(Sort sortName, String name, String type, int purPrice, int salePrice) |
| 前置条件 | 分类、商品名称、型号、进价、零售价输入合法 |
| 后置条件 | 将商品添加到商品分类中，持久化更新涉及的对象的数据 |
| Commodity.deletCommo | 语法 | public ResultMessage deletCommo  (String name, Boolean isOperated) |
| 前置条件 | 商品名称输入合法，商品没有被操作过 |
| 后置条件 | 删除该商品，持久化更新涉及的对象的数据 |
| Commodity.updCommo | 语法 | public ResultMessage updCommo  (String name, info CommoInfo) |
| 前置条件 | 商品名称输入合法，更改信息合法 |
| 后置条件 | 商品更新，持久化更新涉及的对象的数据 |
| Commodity.findCommo | 语法 | public ResultMessage findCommo  (String name, int id) |
| 前置条件 | 商品名称输入合法，id输入合法 |
| 后置条件 | 查找到该商品 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| CommodityDataService.  findCommoSort(int id) | 根据ID进行查找单一持久化对象 | |
| CommodityDataService.  findCommo (int id) | 根据ID进行查找单一持久化对象 | |
| CommodityDataService.  findCommos(Sting field, int id) | 根据字段名和值进行查找多个持久化对象 | |
| CommodityDataService.  add(CommodityPO po) | 添加单一化持久对象 | |
| CommodityDataService.  delete(CommodityPO po) | 删除单一化持久对象 | |
| CommodityDataService.  update(CommodityPO po) | 更改单一化持久对象 | |
| CommodityDataService.  getCommodityData | 得到Commodity数据的服务引用 | |

**inventorybl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Inventory.viewInventory | 语法 | public ResultMessage view  (Date beginDate, Date endDate) |
| 前置条件 | 日期的输入合法 |
| 后置条件 | 显示在此时间段内的出／入库数量／金额，销售／进货数量／金额，以及库存合计 |
| Inventory.checkRecode | 语法 | public ResultMessage checkRecord( ) |
| 前置条件 | 无 |
| 后置条件 | 显示截至盘点时间的库存快照 |
| Inventory.giftRecord | 语法 | public ResultMessage giftRecord  (Commodity commodity,  int commoNum, Client client) |
| 前置条件 | 商品、商品数量、客户的输入合法 |
| 后置条件 | 生成相应的库存赠送单 |
| Inventory.overflowRecord | 语法 | public ResultMessage overflowRecord  (Commodity commodity, int num) |
| 前置条件 | 完成库存的盘点，输入数据 |
| 后置条件 | 生成相应的库存报溢单 |
| Inventory.lossRecord | 语法 | public ResultMessage lossRecord  (Commodity commodity, int num) |
| 前置条件 | 完成库存的盘点，输入数据 |
| 后置条件 | 生成相应的库存报损单 |
| Inventory.alarmRecord | 语法 | public ResultMessage alarmRecord  (Commodity commodity, int num) |
| 前置条件 | 进行商品数量上的操作 |
| 后置条件 | 生成相应的库存报警单 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| InventoryDataService.  insert(InventoryBillPO po) | 生成单据，告知是否成功 | |

**clientbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Client.findClient | 语法 | public ResultMessage findClient(String keywords, long id) |
| 前置条件 | 关键字符合输入条件，id符合输入条件 |
| 后置条件 | 查找到目标客户 |
| Client.addClient | 语法 | public ResultMessage addClient(String kind, String name, String level, String phoneNum, String address, String post) |
| 前置条件 | 类型、名字、级别、电话、地址、邮编符合输入条件 |
| 后置条件 | 添加客户，持久化更新涉及的对象的数据 |
| Client.updClient | 语法 | public ResultMessage updClient(String kind, String name, String level, String phoneNum, String address, String post) |
| 前置条件 | 修改的属性符合输入条件 |
| 后置条件 | 客户属性被修改，持久化更新涉及的对象的数据 |
| Client.deletClient | 语法 | public ResultMessage deletClient(String name) |
| 前置条件 | 输入的客户名称符合输入条件 |
| 后置条件 | 删除该客户，持久化更新涉及的对象的数据 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| ClientDataService.  findClient(int id) | 根据ID进行查找单一持久化对象 | |
| ClientDataService.  findClients(String fields, int id) | 根据字段名和值进行查找多个持久化对象 | |
| ClientDataService.  add(ClientPO po) | 添加单一化持久对象 | |
| ClientDataService.  delet(ClientPO po) | 删除单一化持久对象 | |
| ClientDataService.  update(ClientPO po) | 更改单一化持久对象 | |
| ClientDataService.  getClientData | 得到Client数据的服务引用 | |

**purchasebl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Purchase.addPur | 语法 | public ResultMessage addPur  (Commodity commodity, int num, int price, String remark) |
| 前置条件 | 商品、数量、单价、备注符合输入规则 |
| 后置条件 | 制定出新的进货单 |
| Purchase.addPurBack | 语法 | public ResultMessage addPurBack  (Commodity commodity, int num, int price, String remark) |
| 前置条件 | 商品、数量、单价、备注符合输入规则 |
| 后置条件 | 制定出新的进货退货单 |
| Purchase.submit | 语法 | public ResultMessage submit() |
| 前置条件 | 单据已经被制定出 |
| 后置条件 | 单据被提交发送给经理进行审批 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| PurchaseDataService.  find(String id) | 根据ID进行查找单一持久化对象 | |
| PurchaseDataService.  insert(PurchasePO po) | 插入单一持久化对象 | |
| PurchaseDataService.  delete(PurchasePO po) | 删除单一持久化对象 | |
| PurchaseDataService.  update(PurchasePO po) | 更新单一持久化对象 | |
| PurchaseDataService.init | 清空单一持久化对象 | |

**salebl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Sale.addSale | 语法 | public ResultMessage addSale  (Commodity commodity, int num, int price, String remark) |
| 前置条件 | 商品、数量、单价、备注符合输入规则 |
| 后置条件 | 制定出新的销售单 |
| Sale.addSaleBack | 语法 | public ResultMessage addSaleBack  (Commodity commodity, int num, int price, String remark) |
| 前置条件 | 商品、数量、单价、备注符合输入规则 |
| 后置条件 | 制定出新的销售退货单 |
| Sale.submit | 语法 | public ResultMessage submit() |
| 前置条件 | 单据已经被制定出 |
| 后置条件 | 单据被提交发送给经理进行审批 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| SaleDataService.  find(String id) | 根据ID进行查找单一持久化对象 | |
| SaleDataService.  insert(PurchasePO po) | 插入单一持久化对象 | |
| SaleDataService.  delete(PurchasePO po) | 删除单一持久化对象 | |
| SaleDataService.  update(PurchasePO po) | 更新单一持久化对象 | |
| SaleDataService.init | 清空单一持久化对象 | |

**recordbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Record.saleDetail | 语法 | public ResultMessage saleDetail(Time begin, Time end, Commodity commodity, Client client, User user, Storage Storage) |
| 前置条件 | 时间区间、商品名、客户、业务员、仓库符合输入规则 |
| 后置条件 | 返回符合条件的销售明细表 |
| Record.businessPro | 语法 | public ResultMessage bussinessPro  (Time begin, Time end, String kind, Client client, User user, Storage storage) |
| 前置条件 | 时间区间、单据类型、商品、业务员、仓库符合输入规则 |
| 后置条件 | 返回符合条件的单据的历程表 |
| Record.red | 语法 | public ResultMessage red  (Record record) |
| 前置条件 | 有之前的单子做模板 |
| 后置条件 | 添加红冲 |
| Record.copyRed | 语法 | public ResultMessage copyRed(Record record) |
| 前置条件 | 有之前的单子做模板 |
| 后置条件 | 修改红冲 |
| Record.bussinessState | 语法 | public ResultMessage businessState  (Time begin, Time end, String kind) |
| 前置条件 | 时间区间、类型符合输入规则 |
| 后置条件 | 返回符合条件的经营情况表 |
| Record.earning | 语法 | public ResultMessage earning() |
| 前置条件 | 完成单据的类型选择 |
| 后置条件 | 返回计算得到的收入 |
| Record.expense | 语法 | public ResultMessage expense() |
| 前置条件 | 完成单据的类型选择 |
| 后置条件 | 返回计算得到的支出 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| RecordDataService.  insert(RecordPO po) | 插入单一持久化对象 | |
| RecordDataService.  delete(RecordPO po) | 删除单一持久化对象 | |
| RecordDataService.  update(RecordPO po) | 更新单一持久化对象 | |

**accountbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Account.findAccount | 语法 | public ResultMessage findAccount  (String keyword) |
| 前置条件 | keyword符合输入规则 |
| 后置条件 | 返回查找的账户 |
| Account.addAccount | 语法 | public ResultMessage addAccount(String name, int money) |
| 前置条件 | name和money符合输入规则 |
| 后置条件 | 增加新的账户，持久化更新涉及的对象的数据 |
| Account.deletAccount | 语法 | public ResultMessage deleteAccount  (String name) |
| 前置条件 | name符合输入规则 |
| 后置条件 | 删除账户，持久化更新涉及的对象的数据 |
| Account.updAccount | 语法 | public ResultMessage updAccount(String name) |
| 前置条件 | name符合输入规则 |
| 后置条件 | 更改账户，持久化更新涉及的对象的数据 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| AccountDataService.  find(String id) | 根据ID进行查找单一持久化对象 | |
| AccountDataService.  insert(AccountPO po) | 插入单一持久化对象 | |
| AccountDataService.  delete(AccountPO po) | 删除单一持久化对象 | |
| AccountDataService.  update(AccountPO po) | 更新单一持久化对象 | |
| AccountDataService.init() | 清空单一持久化对象 | |
| AccountDataService.getClientData | 得到Account数据的服务引用 | |

**funtbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Funt.addExpenses | 语法 | public ResultMessage addExpenses  (Client client, Account account, int money, String remark) |
| 前置条件 | 客户、账户、金额、备注符合输入规则 |
| 后置条件 | 建立收款单 |
| Funt.addPay | 语法 | public ResultMessage addPay(Client client, Account account, int money, String remark) |
| 前置条件 | 客户、账户、金额、备注符合输入规则 |
| 后置条件 | 建立付款单 |
| Funt.addCash | 语法 | public ResultMessage addCash(Account account, String name, int money, String remark) |
| 前置条件 | 账户、金额、条目名、备注符合输入规则 |
| 后置条件 | 建立现金费用单 |
| Funt.submit | 语法 | public ResultMessage submit() |
| 前置条件 | 单据已经被制定出 |
| 后置条件 | 单据被提交发送给经理进行审批 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| FuntDataService.  find(String id) | 根据查找单一持久化对象 | |
| FuntDataService.  insert(FuntPO po) | 插入单一持久化对象 | |
| FuntDataService.  delete(FuntPO po) | 删除单一持久化对象 | |
| FuntDataService.  update(FuntPO po) | 更新单一持久化对象 | |
| FuntDataService.init | 初始化单一持久化对象 | |

**accountaInit模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AccountaInit.addInit | 语法 | public ResultMessage addInit  (Commodity commodity, Client client, Account account) |
| 前置条件 | 商品、客户、银行账户符合输入规范 |
| 后置条件 | 建立期初账目 |
| AccountaInit.getEverage | 语法 | public ResultMessage getEverage  (int time, double purPrice, double salePrice) |
| 前置条件 | 已经添加了商品信息 |
| 后置条件 | 返回上年的平均进价和售价 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| AccountaInitDataService.  insert(AccountaInitPO po) | 插入单一持久化对象 | |
| AccountaInitDataService.  init | 初始化持久数据 | |

**approvalbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Apporval.checkRecord | 语法 | public ResultMessage checkRecord  (Record record) |
| 前置条件 | 有需要审核的单据 |
| 后置条件 | 查看单据内容 |
| Apporval.updRecord | 语法 | public ResultMessage updRecord() |
| 前置条件 | 已经查看到单据内容 |
| 后置条件 | 修改单据内容 |
| Apporval.submitRecord | 语法 | public ResultMessage submitRecord() |
| 前置条件 | 确认单据内容没有错误 |
| 后置条件 | 通过单据，持久化更新涉及的对象的数据 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| ApprovalDataService.  Insert(ApprovalPO po) | 插入单一持久化对象 | |

**promotionbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Promotion.promote | 语法 | public ResultMessage promote() |
| 前置条件 | 有一些基本数据，比如商品、销售数据 |
| 后置条件 | 得到销售策略 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| PromotionDataService.  find(String condition) | 根据需要制定的策略条件查找可以生成销售策略的持久化对象 | |

**userbl模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| User.login | 语法 | public ResultMessage login(String id, String password); |
| 前置条件 | password符合输入规则 |
| 后置条件 | 查找是否存在相应的User，根据输入的password返回登录验证的结果 |
| User.addUser | 语法 | public ResultMessage addUser(String id, String password, String position) |
| 前置条件 | 添加的User不重复 |
| 后置条件 | 添加User |
| User.deletUser | 语法 | public ResultMessage deleteUser(String id) |
| 前置条件 | id符合输入规则 |
| 后置条件 | 查找相应的User，根据输入的id删除该用户 |
| User.updateUser | 语法 | public ResultMessage updateUser(String id, String newPosition) |
| 前置条件 | id符合输入规则 |
| 后置条件 | 查找相应的User，根据输入的id，更改该用户的权限 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| DatabaseFacory.getUserDataBase | 得到User的服务的引用 | |
| UserDateService.find  (UserPO po) | 查找单一持久化对象 | |
| UserDataService.insert  (UserPO po) | 插入单一持久化对象 | |
| UserDataService.delete  (UserPO po) | 删除单一持久化对象 | |
| userDataService.update  (UserPO po) | 更新单一持久化对象 | |

## 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。持久化数据保存采用序列化文件

### 数据层模块的职责

数据层模块职责如下表所示

**数据层模块职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| InventoryDataService | 提供库存单据的提交服务（赠送单、报溢单、报损单、报警单） |
| PurchaseDataService | 提供进货(退货)数据集体载入、保存、增加、删除、查找 |
| RecordDataService | 提供销售明细表、经营历程表、经营情况表的查看 |
| FundDataService | 收款单、付款单、现金单的集体载入、保存、查看修改操作 |
| ApprovalDataService | 提供单据的集体载入、保存、查看修改操作 |
| UserDataService | 提供用户集体载入、保存、增删改查服务 |
| CommodityDataServer | 提供商品集体载入、保存、增删改查服务 |
| SaleDataService | 提供销（退货）售数据集体载入、保存、增加、删除、查找 |
| AccountDataService | 提供账户数据集体载入、保存、增删改查服务 |
| ClientDataService | 提供客户数据集体载入、保存、增删改查服务 |
| AccountaInitDataService | 提供期初建账数据集体载入、保存、增加、查看服务 |
| PromotionDataService | 提供数据的保存、修改操作，保存最近卖得比较少的商品。或者组合同一种分类下的商品，进行组合赠送什么的 |
| CommodityManageDataServer | 提供商品分类数据集体载入、保存、增删改查服务 |

### 数据层模块的接口规范

数据层模块的接口规范如下表

**数据层模块的接口规范**

InventoryDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| InventoryDataService.insert | 语法 | public ResultMessage insert(InventoryBillPO po) |
| 前置条件 | 无 |
| 后置条件 | 返回提交是否成功 |
| InventoryDataService.find | 语法 | public InventoryBillPO find(String id) |
| 前置条件 | 无 |
| 后置条件 | 查找返回相应的结果 |
| InventoryDataService.delete | 语法 | public ResultMessage delete (InventoryBillPO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 删除po |
| InventoryDataService.update | 语法 | public ResultMessage update (InventoryBillPO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| InventoryDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

PurchaseDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| PurchaseDataService.insert | 语法 | public ResultMessage insert(PurchasePO po) |
| 前置条件 | 同样的po在数据中不存在 |
| 后置条件 | 增加一条po数据 |
| PurchaseDataService.find | 语法 | public PurchasePO find(String id) |
| 前置条件 | 无 |
| 后置条件 | 查找返回相应的结果 |
| PurchaseDataService.delete | 语法 | public ResultMessage delete (String id) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 删除po |
| PurchaseDataService.update | 语法 | public ResultMessage update (PurchasePO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| PurchaseDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

ApprovalDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| ApprovalDataService.update | 语法 | public ResultMessage update(PersistentObject po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| ApprovalDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

UserDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserDataService.insert | 语法 | public ResultMessage insert(UserPO po) |
| 前置条件 | 同样的po在数据中不存在 |
| 后置条件 | 增加一条po数据 |
| UserDataService.find | 语法 | public UserPO find(String id) |
| 前置条件 | 无 |
| 后置条件 | 查找返回相应的结果 |
| UserDataService.delete | 语法 | public ResultMessage delete (String username) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 删除po |
| UserDataService.update | 语法 | public ResultMessage update (UserPO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| UserDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

CommodityDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CommodityDataService.insert | 语法 | public ResultMessage insert(CommodityPO po) |
| 前置条件 | 同样的po在数据中不存在 |
| 后置条件 | 增加一条po数据 |
| CommodityDataService.find | 语法 | public ResultMessage find(String id) |
| 前置条件 | 无 |
| 后置条件 | 查找返回相应的结果 |
| CommodityDataService.delete | 语法 | public ResultMessage delete (CommodityPO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 删除po |
| CommodityDataService.update | 语法 | public ResultMessage update (CommodityPO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| CommodityDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

SaleDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| SaleDataService.insert | 语法 | public ResultMessage insert(SalesPO po) |
| 前置条件 | 同样的po在数据中不存在 |
| 后置条件 | 增加一条po数据 |
| SaleDataService.find | 语法 | public SalesPO find(String id) |
| 前置条件 | 无 |
| 后置条件 | 查找返回相应的结果 |
| SaleDataService.delete | 语法 | public ResultMessage delete (String id) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 删除po |
| SaleDataService.update | 语法 | public ResultMessage update (SalesPO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| SaleDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

AccountDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AccountDataService.insert | 语法 | public ResultMessage insert(AccountPO po) |
| 前置条件 | 同样的po在数据中不存在 |
| 后置条件 | 增加一条po数据 |
| AccountDataService.find | 语法 | public AccountPO find(String name) |
| 前置条件 | 无 |
| 后置条件 | 查找返回相应的结果 |
| AccountDataService.delete | 语法 | public ResultMessage delete (String name) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 删除po |
| AccountDataService.update | 语法 | public ResultMessage update (AccountPO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| AccountDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

ClientDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| ClientDataService.insert | 语法 | public ResultMessage insert(ClientPO po) |
| 前置条件 | 同样的po在数据中不存在 |
| 后置条件 | 增加一条po数据 |
| ClientDataService.find | 语法 | public ClientPO find(String id) |
| 前置条件 | 无 |
| 后置条件 | 查找返回相应的结果 |
| ClientDataService.delete | 语法 | public ResultMessage delete (String id) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 删除po |
| ClientDataService.update | 语法 | public ResultMessage update (ClientPO po) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| ClientDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

AccountaInitDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AccountaInitDataService.insert | 语法 | public ResultMessage insert(AccountaInitPO po) |
| 前置条件 | 同样的po在数据中不存在 |
| 后置条件 | 增加一条po数据 |
| AccountaInitDataService.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |
| AccountaInitDataService.find | 语法 | public AccountaInitPOfind(String id) |
| 前置条件 | 无 |
| 后置条件 | 返回AccountaInitPO持久数据 |

CommodityManageDataServer

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CommodityManageDataServer.insert | 语法 | public ResultMessage insert  (CommodityManagePO po) |
| 前置条件 | 同样的po在数据中不存在 |
| 后置条件 | 增加一条po数据 |
| CommodityManageDataServer.find | 语法 | public CommodityManagePO find(String id) |
| 前置条件 | 无 |
| 后置条件 | 查找返回相应的结果 |
| CommodityManageDataServer.delete | 语法 | public ResultMessage delete  (String id) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 删除po |
| CommodityManageDataServer.update | 语法 | public ResultMessage update  (CommodityManagePOpo) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 更新po |
| CommodityManageDataServer.init | 语法 | public void init() |
| 前置条件 | 无 |
| 后置条件 | 初始化持久数据 |

RecordDataService

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| RecordDataService.getSaleDetail | 语法 | public ArrayList<PersistentObject> getSaleDetail  (String item) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 返回符合上述条件的商品销售记录，包含如下信息：时间（精确到天），商品名，型号，数量，单价，总额 |
| RecordDataService.getBusinessProcess | 语法 | public ArrayList<PersistentObject> getBusinessProcess  (String item) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 返回1. 销售类单据（销售出货单，销售退货单） 2. 进货类单据（进货单，进货退货单） 3. 财务类单据（付款单，收款单，现金费用单）4. 库存类单据（报溢单，报损单，赠送单）。 |
| RecordDataService.getBusinessCondition | 语法 | public ArrayList<PersistentObject> getBusinessCondition  (String item) |
| 前置条件 | 同样的po在数据中存在 |
| 后置条件 | 返回信息： 1. 收入类：销售收入、商品类收入（商品报溢收入 成本调价收入 进货退货差价 代金券与实际收款差额收入）。收入类显示折让后总收入，并显示折让了多少。2. 支出类：销售成本、商品类支出（商品报损 商品赠出）。支出类显示总支出。3. 利润：折让后总收入-总支出 |

# 信息视角

## 数据持久化对象

系统的PO类就是对应的相关的实体类，如下所示。

* PersistentObject 所有po类的父类，包括时间，编号
* UserPO包括用户名、密码、身份属性。
* ClientPO包括编号、分类（进货商、销售商）、级别（五级，一级普通用户，五级VIP客户）、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。
* CommodityPO包括编号、名称、型号、库存数量、进价、零售价、最近进价、最近

零售价。

* PurchasePO保存进货时候数据的类，包括时间、状态、单据编号（格式为：JHD-yyyyMMdd-xxxxx，后五位每天从1开始编号，所以一天最多可以生成99999条单子），供应商，仓库，操作员，入库商品列表，备注，总额合计。其中入库商品列表包含的信息有：商品编号，名称（从商品选择界面进行选择），型号，数量（手动输入），单价（默认为商品信息中的进价），金额，备注（手动输入）。没有下划线的部分是自动计算并填充进去的
* SalesPO就是保存销售时候数据的类，包括时间、状态、单据编号（XSD-yyyyMMdd-xxxxx），客户（仅显示销售商），业务员（和这个客户打交道的公司员工，可以设置一个客户的默认业务员），操作员（当前登录系统的用户），仓库，出货商品清单，折让前总额，折让，使用代金卷金额，折让后总额，备注。出货商品清单中要显示商品的编号，名称（从商品选择界面选择），型号，数量（手工输入），单价（默认为商品信息里的销售价，可修改），金额（自动生成），商品备注。
* AccountPO包括创建时间、账户名称和金额
* AccountProceedsBillPO 包括时间、状态，单据编号（SKD-yyyyMMdd-xxxxx），客户（同时包含供应商和销售商），操作员（当前登录用户），转账列表，总额汇总。转账列表中的一项包含：银行账户，转账金额，备注
* AccountPaymentBillPO 包括时间、状态、单据编号（XJFYD-yyyyMMdd-xxxxx）,操作员（当前登录用户），银行账户，条目清单，总额。条目清单中包括：条目名，金额，备注。
* InventoryBillPO 包括时间、单据类型、商品、商品数量。(赠送单、报溢单、报损单、报警单)
* AccountaInitPO 包括创建时间、添加商品信息（商品分类，某一商品的名称，类别，型号，进价/售价(默认为上年平均)，最近进价和最近售价留空），客户信息（客户分类，某一个客户的名称，联系方式等，应收应付(之前遗留)），银行账户信息（名称，余额）。
* CommodityManagePO编号、名称、子节点、父节点

## 文件格式

统一用\*.ser格式，比如client.ser