|  |
| --- |
| **快递公司管理系统** |
| 软件体系结构描述文档 |
|  |
| 本报告详细完成对快递公司管理系统的概要设计，达到直到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。 |
|  |
| **王新宇 孙康 宋子微 孙超** |
| **2015/10/18** |
|  |

1. **引言**

**1.1编制目的**

本报告详细完成对快递公司管理系统的概要设计，达到直到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。

**1.2参考资料**

·项目实践v4

·快递公司管理系统用例文档

·快递公司管理系统软件规格说明文档

**2. 产品概述**

快递公司管理系统是为×××物流公司开发的业务系统，开发的目标是帮助该物流公司处理日常的重点业务，包括物流信息查询，装车管理，库存管理，员工信息管理，财务信息管理。

通过快递公司管理系统的应用，期望为×××物流公司提高员工工作效率，提高货运速度，降低货运成本，减少库存积压，减少车辆，人员闲置或过量运作，提高满意度并扩大顾客规模，提高订单量和利润。

**3. 逻辑视角**

快递公司管理系统中，选择了分层体系结构风格，将系统分为三层（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。

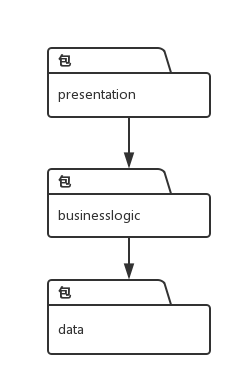


图1 参照体系结构风格的包图表达逻辑视角

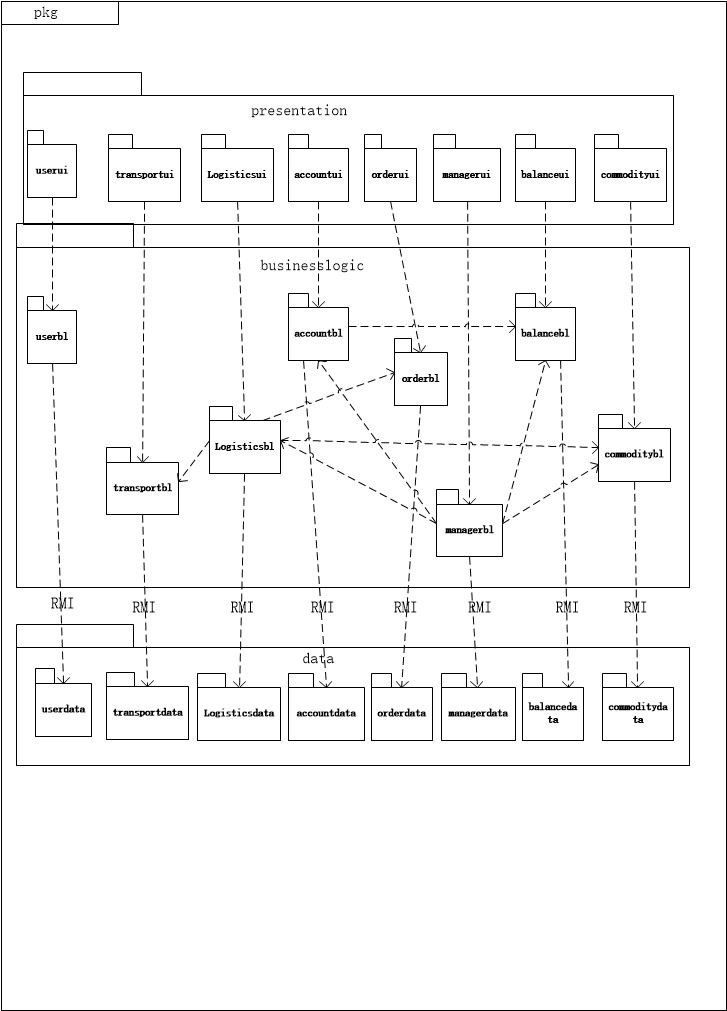


图2 软件体系结构逻辑设计方案

**4.组合视角**

**4.1开发包图**

快递公司管理系统的最终开发包设计如表1所示。

**表1 快递公司管理系统的最终开发包设计**

|  |  |
| --- | --- |
| **开发（物理）包** | **所依赖的其他开发包** |
| mainui | commodityui, transportui, logisticui,accountui, balanceui, userui, orderui, managerui,vo |
| commodityui | 界面类库包，commodityblservice |
| commodityblservice |  |
| commoditybl | commodityblservice, commoditydataservice,po, logisticbl |
| commoditydataservice | Java RMI,po |
| commoditydata | Java RMI,po,databaseutility |
| transportui | transportblservice,界面类库包 |
| transportblservice |  |
| transportbl | transportblservice, transportui,po |
| transportdataservice | Java RMI,po |
| transportdata | Java RMI,po,databaseutility |
| logisticui | Logisticblservice,界面类库包 |
| logisticblservice |  |
| logisticbl | Logisticblservice,logisticdataservice,po, transportbl(commodityInfo),orderbl |
| logisticdataservice | Java RMI,po |
| logisticdata | po,databaseutility, logisticdataservice |
| accountui | 界面类库包，accountblservice |
| accountblservice |  |
| accountbl | accountblservice, accountdataservice,po,balancebl |
| accountdataservice | Java RMI,po |
| accountdata | Java RMI,po,databaseutility |
| balanceui | balanceblservice,界面类库包 |
| balanceblservice |  |
| balancebl | balanceblservice, balancedataservice,po |
| balancedataservice | Java RMI,po |
| balancedata | po,databaseutility, balancedataservice |
| userui | userblservice,界面类库包 |
| userblservice |  |
| userbl | UserInterface,UserDataClient,UserPo |
| userdataservice | Java RMI,po |
| userdata | po,databaseutility,Java RMI |
| orderui | orderblservice,界面类库包 |
| orderblservice |  |
| orderbl | orderblservice, orderdataservice,po |
| orderdataservice | Java RMI,po |
| orderdata | po,databaseutility, orderdataservice |
| managerui | managerblservice,界面类库包 |
| managerblservice |  |
| managerbl | managerblservice, managerdataservice,balancebl,accountbl,po  commoditybl,logisticbl |
| managerdataservice | Java RMI,po |
| managerdata | po,databaseutility, managerdataservice |
| vo |  |
| po |  |
| ublilitybl |  |
| 界面类库包 |  |
| Java RMI |  |
| databaseutility | JDBC |

快递公司管理系统客户端开发包图如图3所示，服务器开发包图如图4所示。

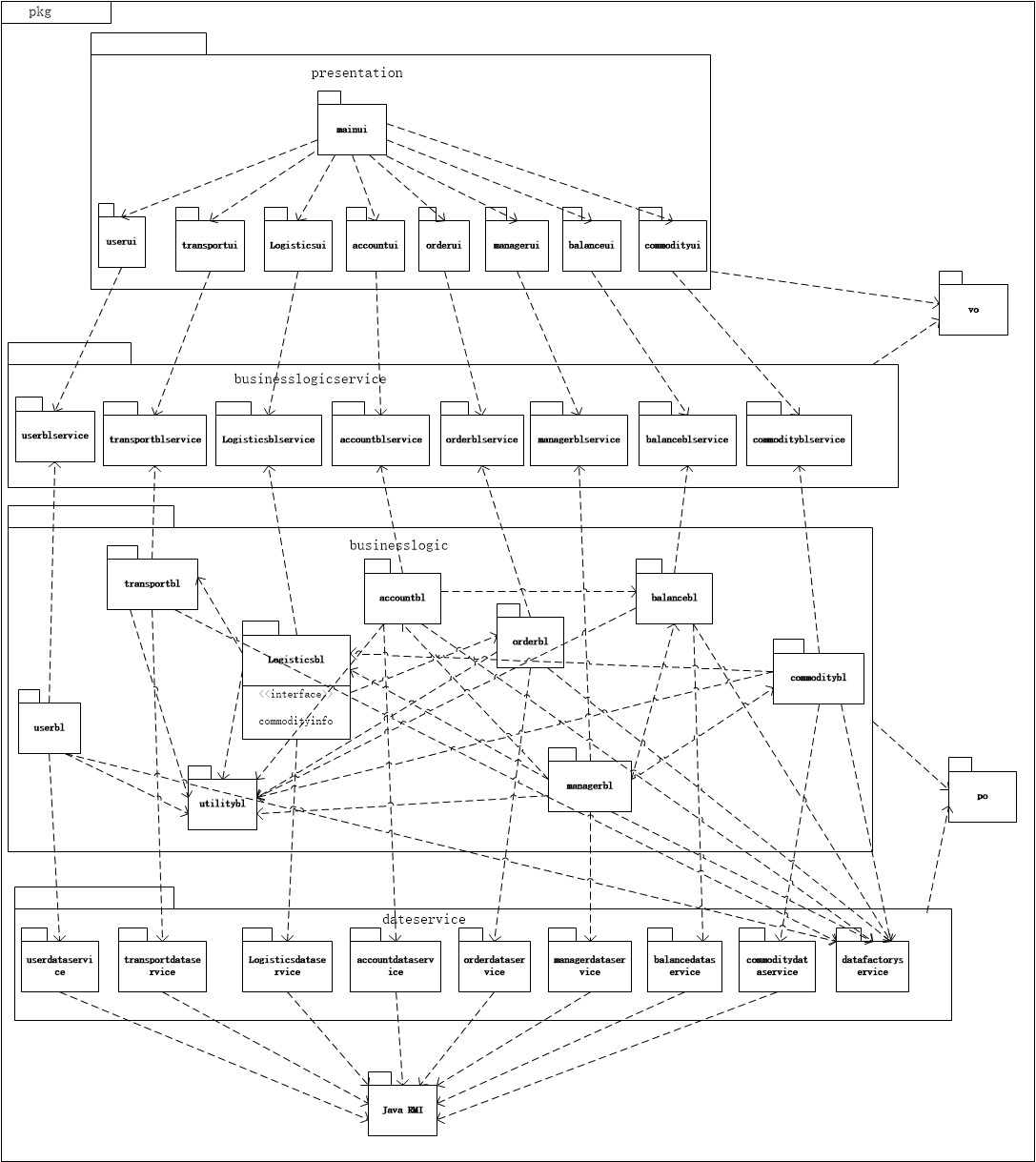


图3 快递公司管理系统客户端开发包图

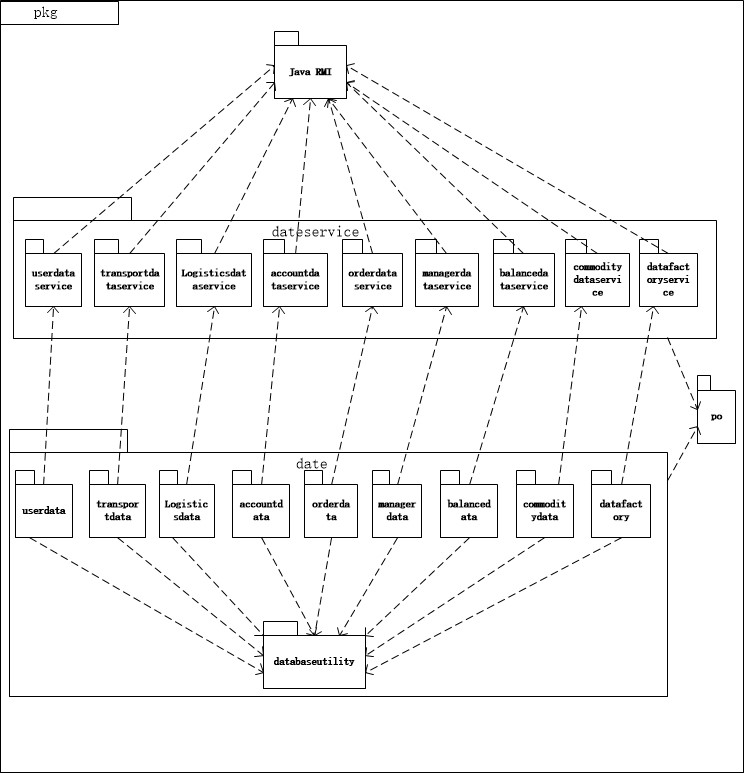
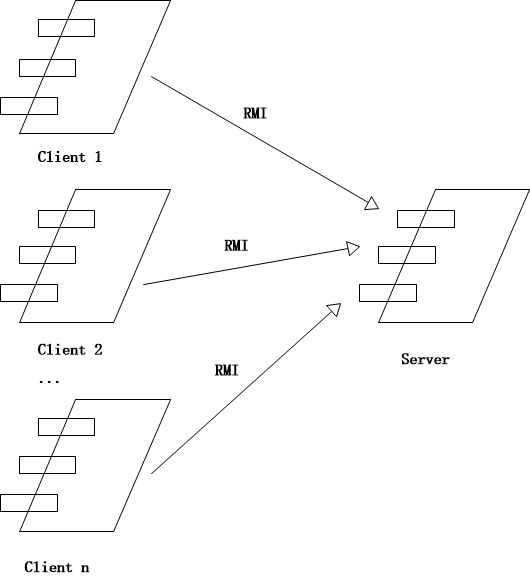


图4 快递公司管理系统服务器端开发包图

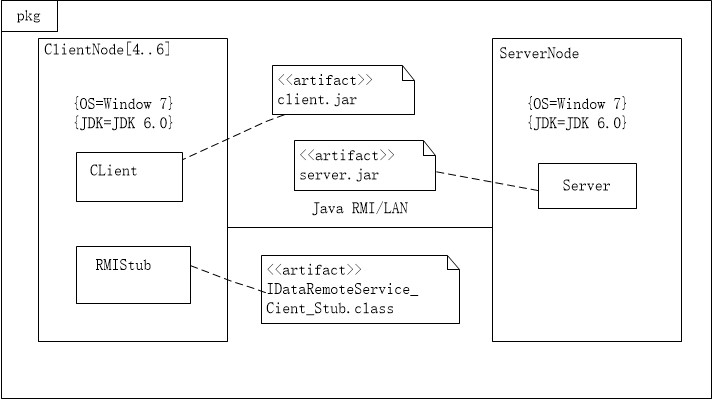
**4.2 运行时进程**

在快递管理系统中，会有多个客户端进程和一个服务器端进程，其进程图如下图所示。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。



**4.3 物理部署**

快递管理系统中客户端构件是放在客户端机器上，服务器端构件是放在服务器端机器上。在客户端节点上，还要部署RMIStub构件。由于Java RMI构件属于JDK 6.0 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。部署图如下图。



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。

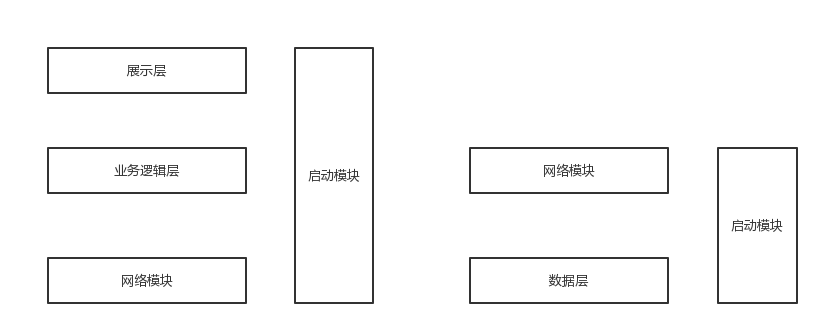


图7 客户端模块视图 图8 服务器模块视图

表2 客户端各层职责

|  |  |
| --- | --- |
| 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 | 基于窗口的快递公司客户端用户界面 |
| 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 |
| LoginBLService  TransportBLService  LogicsticsBLService  AccountBLService  OrderBLService  ManagerBLService  BalanceBLService  CommidityBLService | 客户端展示层 | 客户端业务逻辑层 |
| LoginDataService  TransportDataService  LogicsticsDataService  AccountDataService  OrderDataService  ManagerDataService  BalanceDataService  CommidityDataService  DatabaseFactory | 客户端业务逻辑层 | 服务器端数据层 |

借用订单的包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接口（需接口），而下层实现这个接口（供接口）。OrdersBLService提供了Orders界面所需要的所有业务逻辑功能。OrdersDataService提供了对数据库的操作。这样的实现就大大降低了层与层之间的耦合。

图9

5.2 用户界面层的分解

根据需求，系统存在38个用户界面：登录界面、寄件人主界面、快递员主界面、营业厅业务员主界面、中转中心业务员主界面、中转中心库存管理人员主界面、财务人员主界面、总经理主界面、管理员主界面、物流信息查询界面、订单处理界面、收件信息处理界面、车辆装车管理界面、接收货物界面、派件界面、记录收款信息界面、车辆信息管理界面、司机信息管理界面、管理到达信息界面、管理中转信息界面、管理装车信息界面、库存查看界面、库存盘点界面、入库管理界面、出库管理界面、库存分区调整界面、账户管理界面、结算管理界面、记录付款信息界面、成本管理界面、统计报表界面、期初建账界面、人员和机构管理界面、审批单据界面、查看统计分析界面、常量信息制定界面、薪水策略制定界面、用户管理界面 。界面跳转如图10所示。

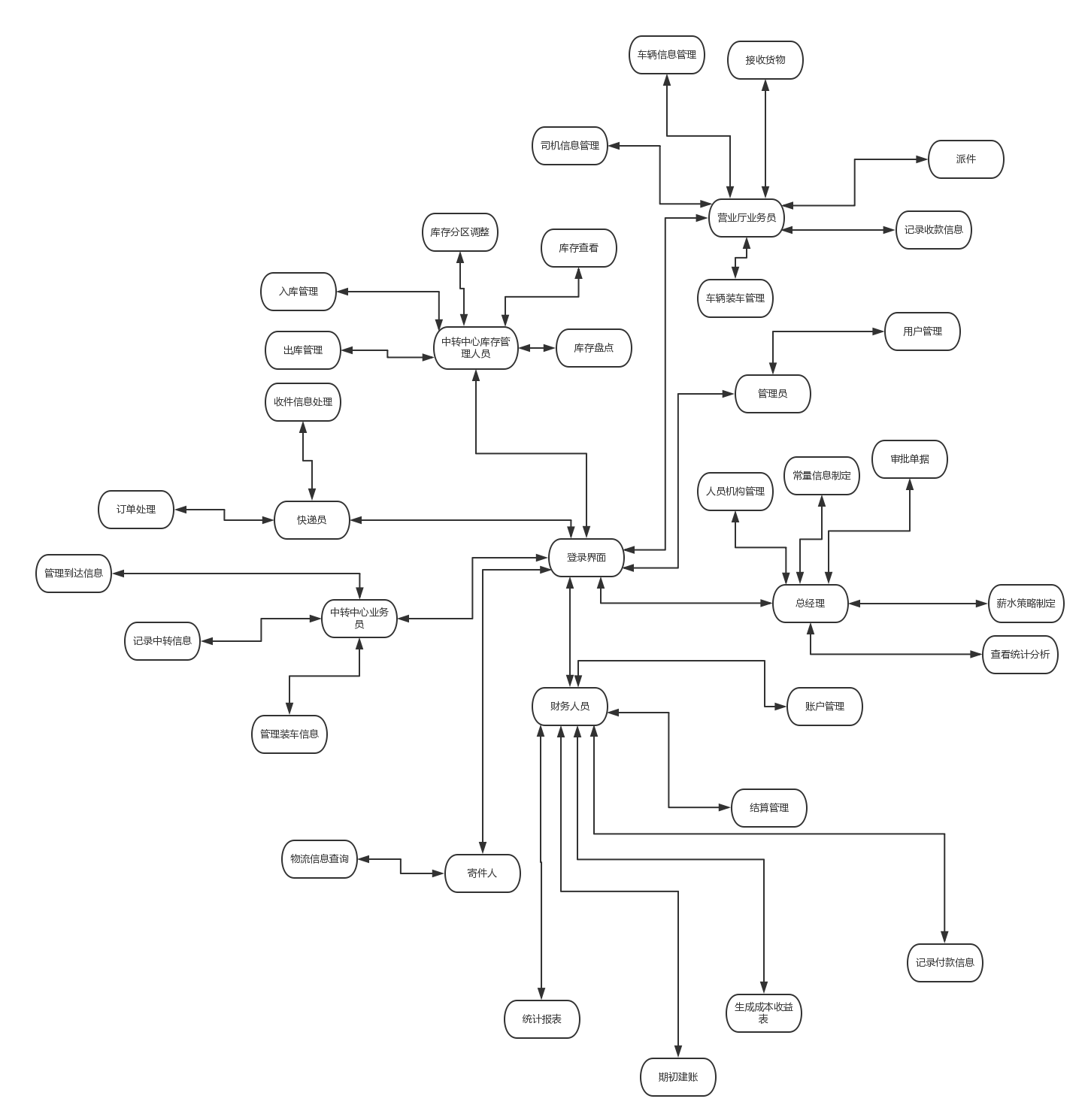
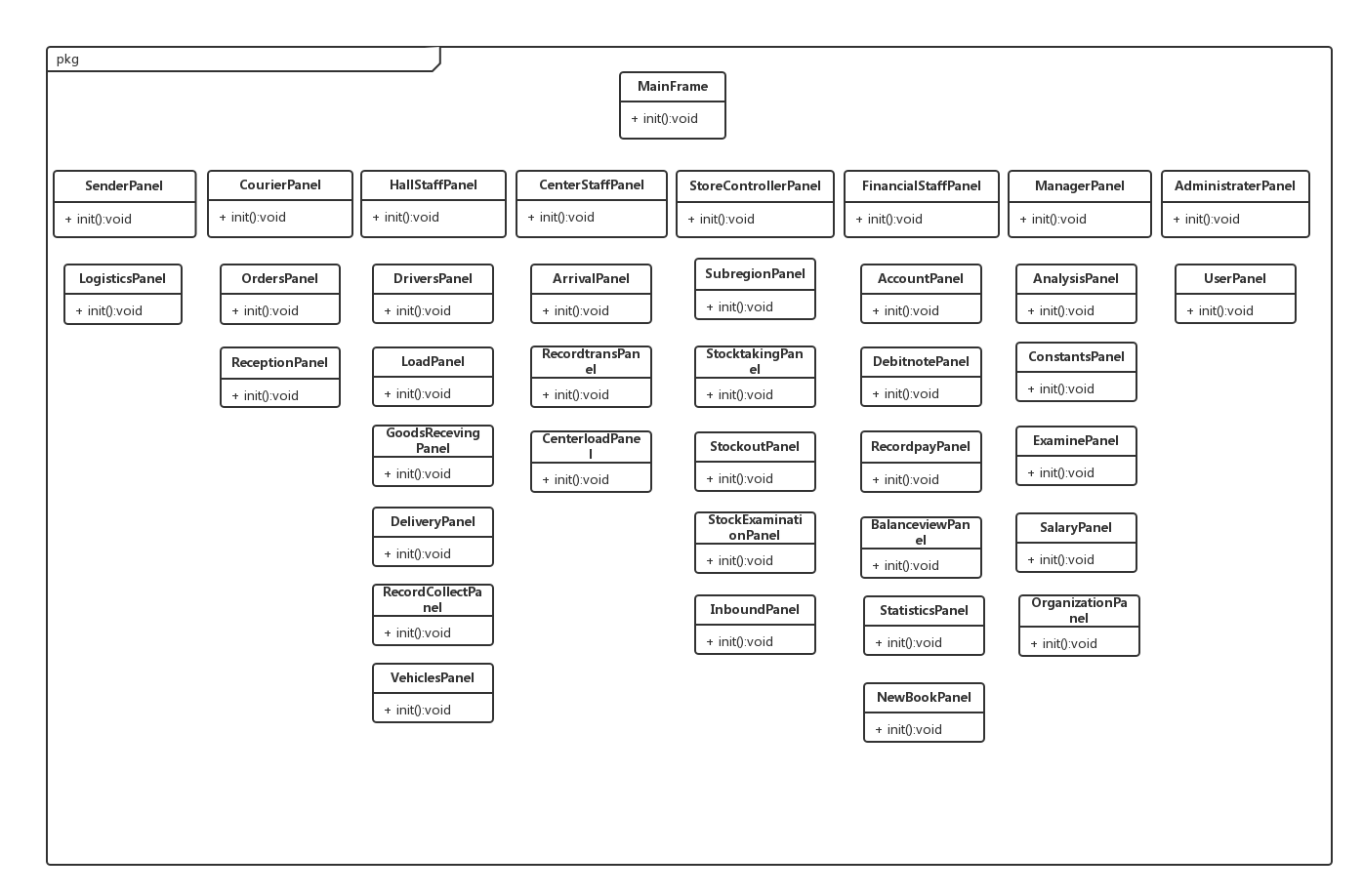


图10

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

 图11

5.2.1 用户界面层模块的职责

如表5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

|  |  |  |
| --- | --- | --- |
| 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.LoginBLService | 登录界面的业务逻辑接口 |
| Businesslogicservice.\*BLService | 每个界面都有一个响应的业务逻辑接口 |

5.2.3 用户界面模块设计原理

用户界面利用Java的Swing和AWT库来实现。

**5.3业务逻辑层的分解**

业务逻辑层包括多个针对界面的业务逻辑处理对象。

业务逻辑层的设计如图12所示。

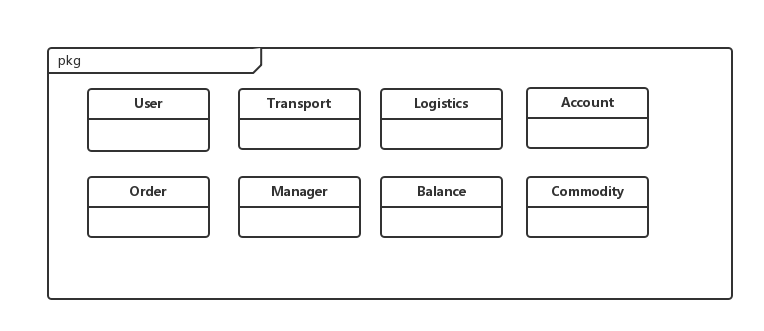
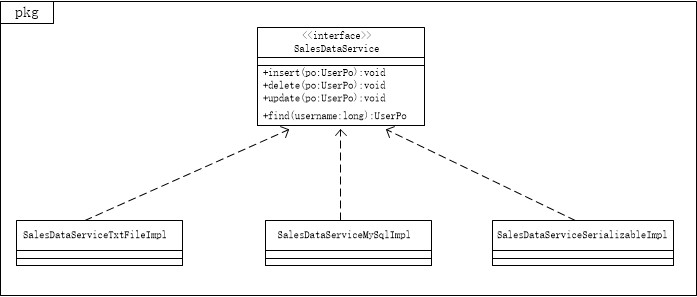
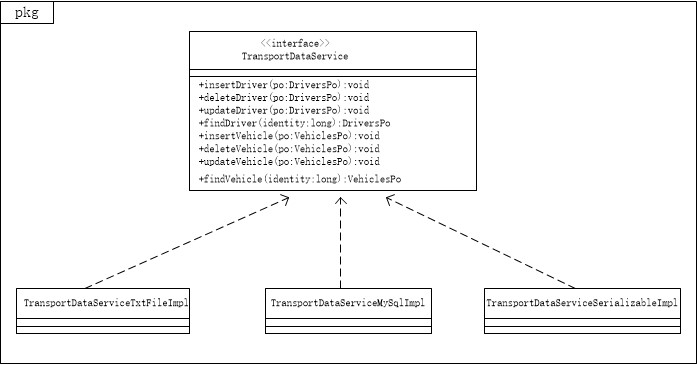


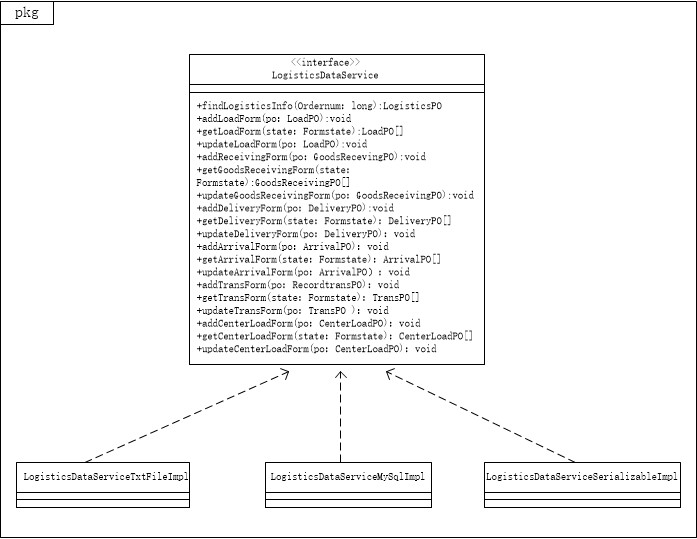
图12 业务逻辑层的设计

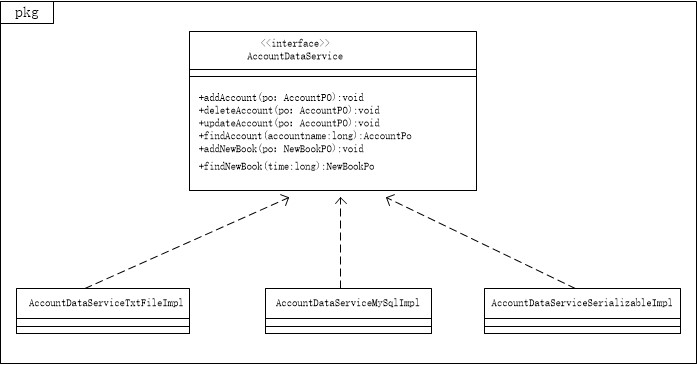
**5.4 数据层的分解**

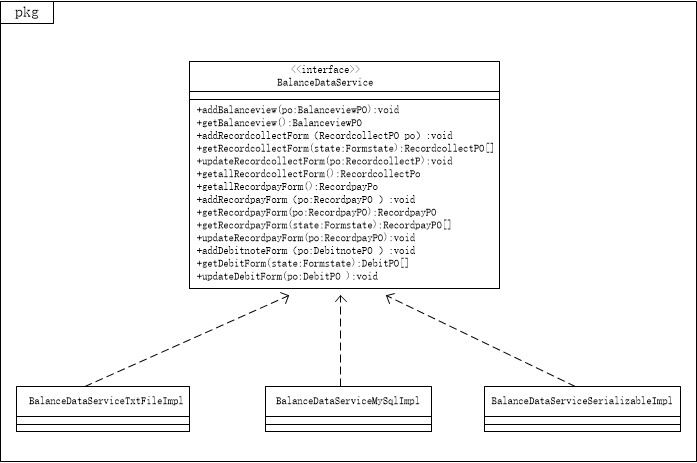
数据层主要给逻辑层提供数据访问服务，包括对于持久化数据的增，删，改，查。由于持久化数据的保存可能存在多种形式：Txt文件，序列化文件，数据库等，所示抽象了数据服务。数据层模块的具体描述如下图。

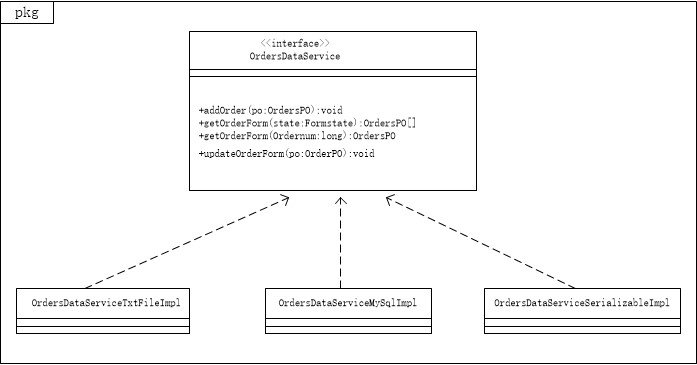


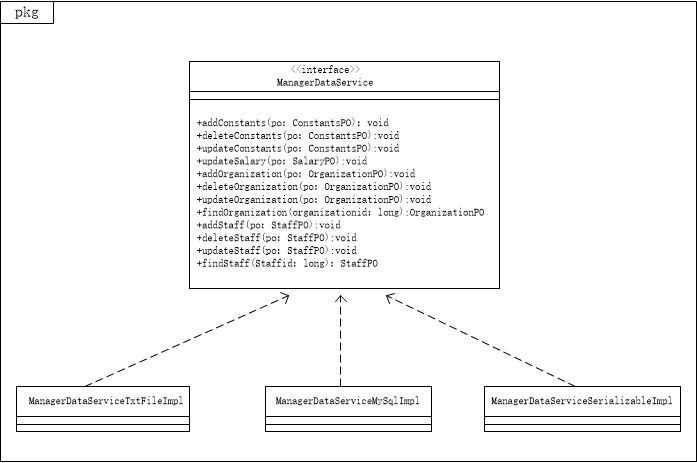


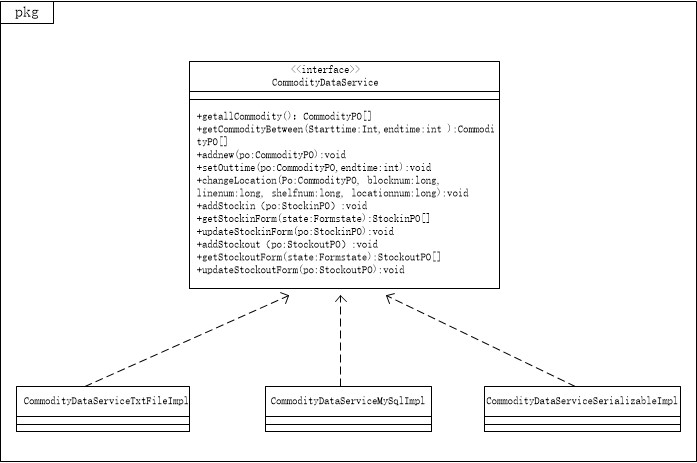












**5.4.1 数据层模块的职责**

|  |  |
| --- | --- |
| 模块 | 职责 |
| UserDataService | 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 |
| UserDataServiceTxtFileImpl | 基于Txt文件的持久化数据库接口，提供集体载入，集体保存，增，删，改，查服务 |
| UserDataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入，集体保存，增，删，改，查服务 |
| UserDataServiceMySqlImpl | 基于MySql数据库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 |
| TransportDataService | 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 |
| TransportDataServiceTxtFileImpl | 基于Txt文件的持久化数据库接口，提供集体载入，集体保存，增，删，改，查服务 |
| TransportDataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入，集体保存，增，删，改，查服务 |
| TransportDataServiceMySqlImpl | 基于MySql数据库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 |
| LogisticsDataService | 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 |
| LogisticsDataServiceTxtFileImpl | 基于Txt文件的持久化数据库接口，提供集体载入，集体保存，增，删，改，查服务 |
| LogisticsDataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入，集体保存，增，删，改，查服务 |
| LogisticsDataServiceMySqlImpl | 基于MySql数据库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 |
| AccountDataService | 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 |
| AccountDataServiceTxtFileImpl | 基于Txt文件的持久化数据库接口，提供集体载入，集体保存，增，删，改，查服务 |
| AccountDataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入，集体保存，增，删，改，查服务 |
| AccountDataServiceMySqlImpl | 基于MySql数据库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 |
| BalanceDataService | 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 |
| BalanceDataServiceTxtFileImpl | 基于Txt文件的持久化数据库接口，提供集体载入，集体保存，增，删，改，查服务 |
| BalanceDataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入，集体保存，增，删，改，查服务 |
| BalanceDataServiceMySqlImpl | 基于MySql数据库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 |
| OrdersDataService | 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 |
| OrdersDataServiceTxtFileImpl | 基于Txt文件的持久化数据库接口，提供集体载入，集体保存，增，删，改，查服务 |
| OrdersDataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入，集体保存，增，删，改，查服务 |
| OrdersDataServiceMySqlImpl | 基于MySql数据库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 |
| ManagerDataService | 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 |
| ManagerDataServiceTxtFileImpl | 基于Txt文件的持久化数据库接口，提供集体载入，集体保存，增，删，改，查服务 |
| ManagerDataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入，集体保存，增，删，改，查服务 |
| ManagerDataServiceMySqlImpl | 基于MySql数据库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 |
| CommodityDataService | 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 |
| CommodityDataServiceTxtFileImpl | 基于Txt文件的持久化数据库接口，提供集体载入，集体保存，增，删，改，查服务 |
| CommodityDataServiceSerializableFileImpl | 基于序列化文件的持久化数据库的接口，提供集体载入，集体保存，增，删，改，查服务 |
| CommodityDataServiceMySqlImpl | 基于MySql数据库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**

数据层模块的接口规范如表12所属。

**表12 TransportData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| TransportDataService.insertDriver | 语法 | public void insertDriver(DriversPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| TransportDataService.deleteDriver | 语法 | public void deleteDriver(DriversPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 删除一个po |
| TransportDataService.updateDriver | 语法 | public void updateDriver(DriversPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| TransportDataService.findDriver | 语法 | public DriversPO findDriver(long identity)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按identity进行查找返回相应的DriversPO结果 |
| TransportDataService.insertVehicle | 语法 | public void insertVehicle(VehiclesPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| TransportDataService.deleteVehicle | 语法 | public void deleteVehicle(VehiclesPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 删除一个po |
| TransportDataService.updateVehicle | 语法 | public void updateVehicle(VehiclesPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| TransportDataService.findVehicle | 语法 | public VehiclesPO findVehicle(long identity)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按identity进行查找返回相应的VehiclesPO结果 |

**LogisticsData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| LogisticsDataService. addLoadForm | 语法 | public void addLoadForm (LoadPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| LogisticsDataService.getLoadForm | 语法 | public LoadPO[] getLoadForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的LoadPO[]结果 |
| LogisticsDataService.updateLoadForm | 语法 | public void updateLoadForm(LoadPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| LogisticsDataService. findLogisticsInfo | 语法 | public LogisticsPO findLogisticsInfo (long ordernum)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按ordernum进行查找返回相应的LogisticsPO结果 |
| LogisticsDataService. addGoodsReceivingForm | 语法 | public void addGoodsReceivingForm (GoodsReceivingPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| LogisticsDataService.getGoodsReceivingForm | 语法 | public GoodsReceivingPO[] getGoodsReceivingForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的GoodsReceivingPO[]结果 |
| LogisticsDataService.updateGoodsReceivingForm | 语法 | public void updateGoodsReceivingForm (GoodsReceivingPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| LogisticsDataService. addGoodsReceivingForm | 语法 | public void addGoodsReceivingForm (GoodsReceivingPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| LogisticsDataService.getDeliveryForm | 语法 | public DeliveryPO[] getDeliveryForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的DeliveryPO[]结果 |
| LogisticsDataService.updateDeliveryForm | 语法 | public void updateDeliveryForm (DeliveryPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| LogisticsDataService. addDeliveryForm | 语法 | public void addDeliveryForm (DeliveryPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| LogisticsDataService.getArrivalForm | 语法 | public ArrivalPO[] getArrivalForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的ArrivalPO[]结果 |
| LogisticsDataService.updateArrivalForm | 语法 | public void updateArrivalForm (ArrivalPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| LogisticsDataService. addArrivalForm | 语法 | public void addArrivalForm (ArrivalPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| LogisticsDataService.getRecordtransForm | 语法 | public RecordtransPO[] getRecordtransForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的RecordtransPO[]结果 |
| LogisticsDataService.updateRecordtransForm | 语法 | public void updateRecordtransForm (RecordtransPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| LogisticsDataService. addRecordtransForm | 语法 | public void addRecordtransForm (RecordtransPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| LogisticsDataService.getCenterLoadForm | 语法 | public CenterLoadPO[] getCenterLoadForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的CenterLoadPO[]结果 |
| LogisticsDataService.updateCenterLoadForm | 语法 | public void updateCenterLoadForm (CenterLoadPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| LogisticsDataService. addCenterLoadForm | 语法 | public void addCenterLoadForm (CenterLoadPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |

**AccountData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| AccountService.addAccount | 语法 | public void addAccount(AccountPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| AccountService.deleteAccount | 语法 | public void deleteAccount(AccountPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 删除一个po |
| AccountService.updateAccount | 语法 | public void updateAccount(AccountPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| AccountService.findAccount | 语法 | public AccountsPO findAccount(long accountname)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按accountname进行查找返回相应的AccountPO结果 |
| AccountService.addNewBook | 语法 | public void addNewBook(NewBookPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| AccountService.findNewBook | 语法 | public NewBookPO[] findNewBook(long time)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按time进行查找返回相应的NewBookPO[]结果 |

**BalanceData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| BalanceDataService. addBalanceview | 语法 | public void addBalanceview (BalanceviewPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| BalanceDataService.getBalanceview | 语法 | public BalanceviewPO getBalanceview ()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 生成并返回的BalanceviewPO结果 |
| BalanceDataService. addRecordcollectForm | 语法 | public void addRecordcollectForm (RecordcollectPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| BalanceDataService.getRecordcollectForm | 语法 | public RecordcollectPO[] getRecordcollectForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的RecordcollectPO[]结果 |
| BalanceDataService.updateRecordcollectForm | 语法 | public void updateRecordcollectForm (RecordcollectPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| BalanceDataService.getRecordpayForm | 语法 | public RecordpayPO[] getRecordpayForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的RecordpayPO[]结果 |
| BalanceDataService.updateRecordpayForm | 语法 | public void updateRecordpayForm (RecordpayPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| BalanceDataService. addRecordpayForm | 语法 | public void addRecordpayForm (RecordpayPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| BalanceDataService.getDebitnoteForm | 语法 | public DebitnotePO[] getDebitnoteForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的DebitnotePO[]结果 |
| BalanceDataService.updateDebitnoteForm | 语法 | public void updateDebitnoteForm (DebitnotePO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| BalanceDataService. addDebitnoteForm | 语法 | public void addDebitnoteForm (DebitnotePO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| BalanceDataService. getallRecordcollectForm | 语法 | public RecordcollectPO[] getallRecordcollectForm ()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中的RecordcollectPO[]结果 |
| BalanceDataService. getallRecordpayForm | 语法 | public RecordpayPO[] getallRecordpayForm ()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中的RecordpayPO[]结果 |

**OrdersData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| OrdersDataService. addOrderForm | 语法 | public void addOrderForm (OrdersPO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| OrdersDataService.getOrderForm | 语法 | public OrdersPO getOrderForm (long ordernum)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按ordernum进行查找返回相应的OrdersPO结果 |
| OrdersDataService.getOrderForm | 语法 | public OrdersPO[] getOrderForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的OrdersPO[]结果 |
| OrdersDataService.updateOrderForm | 语法 | public void updateOrderForm (OrdersPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |

**ManagersData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| ManagersDataService. addConstants | 语法 | public void addConstants (ConstantsPO  po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| ManagersDataService. deleteConstants | 语法 | public void deleteConstants (ConstantsPO  po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 删除一个PO |
| ManagersDataService. updateConstants | 语法 | public void updateConstants (ConstantsPO  po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| ManagersDataService. updateSalary | 语法 | public void updateSalary (SalaryPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| ManagersDataService. addOrganization | 语法 | public void addOrganization (OrganizationPO  po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| ManagersDataService. deleteOrganization | 语法 | public void deleteOrganization (OrganizationPO  po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 删除一个PO |
| ManagersDataService. updateOrganization | 语法 | public void updateOrganization (OrganizationPO  po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| ManagerDataService.findOrganization | 语法 | public OrganizationPO findOrganization(long organizationid)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按organizationid进行查找返回相应的OrganizationPO结果 |
| ManagersDataService. addStaff | 语法 | public void addStaff (StaffPO  po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| ManagersDataService. deleteStaff | 语法 | public void deleteStaff (StaffPO  po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 删除一个PO |
| ManagersDataService. updateStaff | 语法 | public void updateStaff (StaffPO  po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| ManagerDataService.findStaff | 语法 | public StaffPO findStaff(long Staffid)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按Staffid进行查找返回相应的StaffPO结果 |

**CommodityData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| CommodityDataService. addStockinForm | 语法 | public void addStockinForm (StockinPO PO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| CommodityDataService.getStockinForm | 语法 | public StockinPO[] getStockinForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的StockinPO []结果 |
| CommodityDataService.updateStockinForm | 语法 | public void updateStockinForm (StockinPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| CommodityDataService. addStockoutForm | 语法 | public void addStockoutForm (StockoutPO PO po)throws RemoteException; |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| CommodityDataService.getStockoutForm | 语法 | public StockoutPO[] getStockoutForm (Formstate state)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按state进行查找返回相应的StockoutPO []结果 |
| CommodityDataService.updateStockoutForm | 语法 | public void updateStockoutForm (StockoutPO po)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| CommodityDataService.getallCommodity | 语法 | public CommodityPO[] getallCommodity()throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中的CommodityPO[] |
| CommodityDataService.getallCommodityBetween | 语法 | public CommodityPO[] getCommodityBetween(  long starttime,long endtime)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中所有入库时间在starttime和endtime之间的CommodityPO |
| CommodityDataService.addnew | 语法 | public void addnew(CommodityPO po) throws RemoteException |
| 前置条件 | 同样ID的po在Mapper中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| CommodityDataService.setOuttime | 语法 | public void addnew(setOuttime(CommodityPO  po,int endtime) throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |
| CommodityDataService.changeLocation | 语法 | public void changeLocation(CommodityPO  Po, long blocknum, long linenum, long shelfnum, long locationnum)throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的po |
| 后置条件 | 更新一个po |

**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。

UserPO类包含用户的用户名username、密码code、权限属性level、id。

LogisticsPO类包含快件的货运状态、历史轨迹属性。订单编号ordernum

OrdersPO类包含订单的寄件人姓名sendername、住址senderaddress、单位senderunit、电话senderphone、手机sendercellphone，收件人姓名receivername、住址receiveraddress、单位receiverunit、电话receiverphone、手机receivercellphone，原件数numbers、实际重量weight、体积volume、内件品名productname，尺寸productsize，包装费packagefee，费用合计totalfee，订单条形码号ordercode，快递类型ordertype、收件编号codeofreceiving、收件人receiver、收件日期属性receivingtime。

LoadPO类包含快件的装车日期loadtime、本营业厅编号hallcode、汽运编号motorcode、到达地destination、车辆代号vehiclecode、监装员supervisor、押运员supercargo、本次装箱所有订单条形码号allbarcode、运费fee、单据状态属性documentstate。

GoodsRecevingPO类包含快件的到达营业厅日期arrivaltime、中转单编号transcode、出发地departureplace、货物到达状态arrivalstate、单据状态属性documentstate。

DeliveryPO类包含快件的到达日期arrivaltime、托运订单条形码barcode、派件员信息delivorinfo、单据状态属性documentstate。

RecordcollectPO类包含收款日期collectiontime、收款金额collectionsum、收款营业厅业务员collectionman、对应的所有快递订单条形码号allordercode、单据状态属性documentstate。

VehiclesPO类包含车辆代号vehiclecode、车牌号vehiclenum、服役时间extendtime属性。

DriversPO类包含司机的编号drivercode、姓名drivername、出生日期birthtime、身份证号identifiercode、手机cellphone、性别drivergender、行驶证期限属性timelimit。

ArrivalPO类包含中转中心编号centercode、到达日期arrivaltime、中转单编号transcode、出发地departure、货物到达状态arrivalstate、单据状态属性documentstate。

RecordtransPO类包含装车日期loadtime、本中转中心中转单编号transcode，航班号flightcode、出发地departure、到达地destinaton、货柜号countercode、监装员supervisor、本次装箱所有托运单号allcode、运费fee、单据状态属性documentstate。

CenterloadPO类装车日期loadtime、本中转中心汽运编号motorcode、到达地destination、车辆代号vehiclecode、监装员supervisor、押运员supercargo、本次装箱所有订单条形码号allbarcode、运费fee、单据状态属性documentstate。

CommodityPO类包含快件的编号ordercode、入库日期intime、区号blocknum、排号linenum、架号shelfnum、位号locationnum、出库日期outtime属性。

StockinPO类包含快件的编号ordercode、入库日期intime、区号blocknum、排号linenum、架号shelfnum、位号locationnum、单据状态属性documentstate。

StockoutPO类包含快件的编号ordercode、出库日期outtime、目的地destination、装运形式loadform、中转单编号transcode或者汽运编号motorcode、单据状态属性documentstate。

AccountPO类包含账户的名称accountname、金额属性accountsum。

DebitnotePO类包含收款日期debittime、收款单位debitunit、收款人debitor、收款方debitside、收款金额sum、收款地点location、单据状态属性documentstate。

RecordpayPO类包含付款日期paytime、付款金额paysum、付款人payman、付款账号payaccount、租金rent、运费fee、人员工资salary、奖励reward、备注remark、单据状态属性documentstate。

BalanceviewPO类包含总收入income、总支出pay、总利润属性allbenefit。

NewbookPO类包含机构organization、人员staff、车辆vehicle、库存commodity、 银行账户名称accountname、金额属性accountsum。

OrganizationPO类包含机构名称name、机构类型type属性。

StaffPO类包含人员姓名name、职业job属性。

ConstantsPO类包含城市距离distance、价格price等属性。

SalaryPO类包含薪水策略属性。

持久化用户对象UserPO的定义如图14所示。

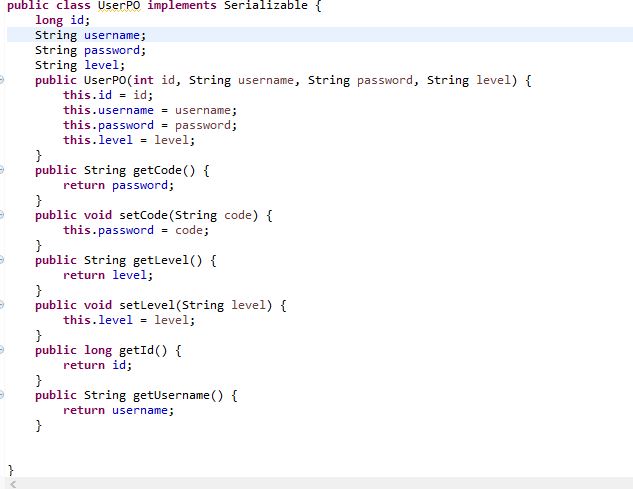


图14 持久化用户对象UserPO的定义

**6.2 Txt持久化对象**

Txt数据保持格式以DriversPO为例。每行分别对应编号、姓名、出生日期、身份证号、手机、性别、行驶证期限。中间用“:”隔开。如下所示：

025001002:张三:19800403:310228198004035430:13123456789:男:20160809

025002012:王三:19830813:310228198308134420:13123456889:男:20170516

**6.3数据库表**

数据库中包含UserPO表、LogisticsPO表、OrdersPO表、LoadPO表、GoodsRecevingPO表、DeliveryPO表、RecordcollectPO表、VehiclesPO表、DriversPO表、ArrivalPO表、RecordtransPO表、CenterloadPO表、CommodityPO表、StockinPO表、StockoutPO表、AccountPO表、DebitnotePO表、RecordpayPO表、StocktakingPO表、NewBookPO表、OrganizationPO表、StaffPO表、ConstantsPO表、SalaryPO表。

DeliveryPO类包含快件的到达日期arrivaltime、托运订单条形码barcode、派件员信息delivorinfo、单据状态属性documentstate。

RecordcollectPO类包含收款日期collectiontime、收款金额collectionsum、收款营业厅业务员collectionman、对应的所有快递订单条形码号allordercode、单据状态属性documentstate。

VehiclesPO类包含车辆代号、车牌号、服役时间、车辆图片属性。

DriversPO类包含司机的编号drivercode、姓名drivername、出生日期birthtime、身份证号identifiercode、手机cellphone、性别drivergender、行驶证期限属性timelimit。

ArrivalPO类包含中转中心编号centercode、到达日期arrivaltime、中转单编号transcode、出发地departure、货物到达状态arrivalstate、单据状态属性documentstate。

RecordtransPO类包含装车日期loadtime、本中转中心中转单编号transcode，航班号flightcode、出发地departure、到达地destinaiton、货柜号countercode、监装员supervisor、本次装箱所有托运单号allcode、运费fee、单据状态属性documentstate。

CenterloadPO类装车日期loadtime、本中转中心汽运编号motorcode、到达地destination、车辆代号vehiclecode、监装员supervisor、押运员supercargo、本次装箱所有订单条形码号allbarcode、运费fee、单据状态属性documentstate。

CommodityPO类包含快件的编号ordercode、入库日期intime、区号blocknum、排号linenum、架号shelfnum、位号locationnum、出库日期outtime、目的地destination、装运形式loadform、中转单编号transcode或者汽运编号motorcode、单据状态属性documentstate。

AccountPO类包含账户的名称accountname、金额属性accountsum。

DebitnotePO类包含收款日期debittime、收款单位debitunit、收款人debitor、收款方debitside、收款金额sum、收款地点location、单据状态属性documentstate。

RecordpayPO类包含付款日期paytime、付款金额paysum、付款人payman、付款账号payaccount、条目paylist、备注remark、单据状态属性documentstate。

//BalanceviewPO类包含总收入income、总支出pay、总利润属性allbenefit。

NewBookPO类包含机构organization、人员staff、车辆vehicle、库存comodity、 银行账户名称accountname、金额属性accountsum。

OrganizationPO类包含机构organization、人员信息属性staff()。

ConstantsPO类包含城市距离distance、价格price等属性。

SalaryPO类包含薪水策略属性。

持久化用户对象UserPO的定义如图14所示。

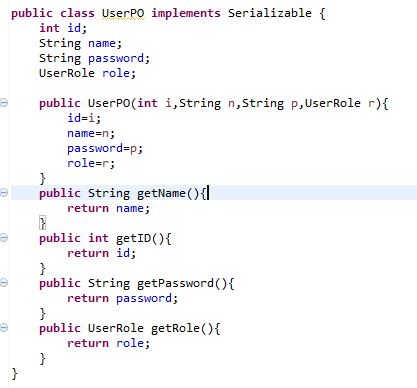


图14 持久化用户对象UserPO的定义

**6.2 Txt持久化对象**

Txt数据保持格式以DriversPO为例。每行分别对应编号、姓名、出生日期、身份证号、手机、性别、行驶证期限。中间用“:”隔开。如下所示：

025001002:张三:1980/4/3:310228198004035430:13123456789:男:2016\8\9

025002012:王三:1983/8/13:310228198308134420:13123456889:男:2017\5\6

**6.3数据库表**

数据库中包含UserPO表、LogisticsPO表、OrdersPO表、LoadPO表、GoodsRecevingPO表、DeliveryPO表、RecordcollectPO表、VehiclesPO表、DriversPO表、ArrivalPO表、RecordtransPO表、CenterloadPO表、CommodityPO表、AccountPO表、DebitnotePO表、RecordpayPO表、StocktakingPO表、NewBookPO表、OrganizationPO表、ConstantsPO表、SalaryPO表。