袁楚宏 袁阳阳 张鑫龙 周小帆

China Nanjing University

快递物流系统 软件体系结构描述文档

目录

[1. 引言 2](#_Toc433661539)

[1.1 编制目的 2](#_Toc433661540)

[1.2 词汇表 2](#_Toc433661541)

[1.3 参考资料 2](#_Toc433661542)

[2. 产品概述 2](#_Toc433661543)

[3. 逻辑视角 2](#_Toc433661544)

[4. 组合视角 4](#_Toc433661545)

[4.1 开发包图 4](#_Toc433661546)

[4.2 运行时进程 7](#_Toc433661547)

[4.3 物理部署 7](#_Toc433661548)

[5. 接口视角 8](#_Toc433661549)

[5.1 模块的职责 8](#_Toc433661550)

[5.2 用户界面层的分解 10](#_Toc433661551)

[5.2.1用户界面层模块的职责 11](#_Toc433661552)

[5.2.2 用户界面层的接口规范 13](#_Toc433661553)

[5.2.3 用户界面模块设计原理 21](#_Toc433661554)

[5.3 业务逻辑层的分解 21](#_Toc433661555)

[5.3.1 业务逻辑层模块的职责 21](#_Toc433661556)

[5.3.2 业务逻辑层模块的接口规范 21](#_Toc433661557)

[5.4 数据层的分解 33](#_Toc433661558)

[5.4.1 数据层模块的职责 33](#_Toc433661559)

[5.4.2 数据层模块的接口规范 34](#_Toc433661560)

[6. 信息视角 45](#_Toc433661561)

[6.1数据持久化对象 45](#_Toc433661562)

1. 引言
   1. 编制目的

本报告详细完成对快递物流系统的概要设计，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. 词汇表

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| ELS | 快递物流系统 |  |
| RMI | 远程接口调用 | 一般使用JavaRMI包 |

* 1. 参考资料

1. 快递物流系统用例文档
2. 快递物流系统需求规格文档
3. 产品概述

参考快递物流系统用例文档和快递物流系统需求规格说明中对产品的概括描述。

1. 逻辑视角

快递物流系统中，选择了分层体系结构风格，将系统分为三层（界面层，逻辑层和数据层）能够很好地展示整个高层抽象。界面层包含了GUI界面的实现，逻辑层负责业务逻辑的处理，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 参考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

1. 组合视角
   1. 开发包图

快递物流系统的最终开发包设计如表1所示。

表1 快递物流系统的最终开发包设计

|  |  |
| --- | --- |
| 开发（物理）包 | 依赖的其他开发包 |
| mainUI | userUI,transitInfoUI,orderNewUI,orderApproveUI,accountUI,cityUI,  stockUI,workOrgManUI,checkUI,VO |
| userUI | userLogicSer,界面类库包,VO |
| userLogicSer |  |
| userLogic | userLogicSer,userDataSer,PO,utilityLogic,dataFactorySer |
| userDataSer | JavaRMI,PO |
| userData | dataBaseUtility,PO,userDataSer |
| transitInfoUI | transitInfoLogicSer,界面类库包,VO |
| transitInfoLogicSer |  |
| transitInfoLogic | transitLogicSer,transitDataSer,PO,orderNewLogic,utilityLogic,dataFactroySer |
| transitInfoDataSer | JavaRMI,PO |
| transitInfoData | dataBaseUtility,PO,transitDataSer |
| orderNewUI | orderNewLogicSer,界面类库包,VO |
| orderNewSer |  |
| orderNewLogic | orderNewLogicSer,orderNewDataSer,cityLogic,checkLogic,PO,utilityLogic,dataFactroySer |
| orderNewDataSer | JavaRMI,PO |
| orderNewData | dataBaseUtility,PO, orderNewDataSer |
| orderApproveUI | orderApproveLogicSer,界面类库包,VO |
| orderApproveSer |  |
| orderApproveLogic | orderApproveLogicSer,orderApproveDataSer,PO,utilityLogic,dataFactroySer |
| orderApproveDataSer | JavaRMI,PO |
| orderApproveData | dataBaseUtility,PO, orderApproveDataSer |
| accountUI | accountLogicSer,界面类库包,VO |
| accountSer |  |
| accountLogic | accountLogicSer, accountDataSer,PO,utilityLogic,dataFactroySer |
| accountDataSer | JavaRMI,PO |
| accountData | dataBaseUtility,PO, accountDataSer |
| cityUI | cityLogicSer,界面类库包,VO |
| citySer |  |
| cityLogic | cityLogicSer, cityDataSer,PO,utilityLogic,dataFactroySer |
| cityDataSer | JavaRMI,PO |
| cityData | dataBaseUtility,PO, cityDataSer |
| workOrgManUI | workOrgManLogicSer,界面类库包,VO |
| workOrgManSer |  |
| workOrgManLogic | workOrgManLogicSer, workOrgManDataSer,PO,utilityLogic,dataFactroySer |
| workOrgManDataSer | JavaRMI,PO |
| workOrgManData | dataBaseUtility,PO, workOrgManDataSer |
| stockUI | stockLogicSer,界面类库包,VO |
| stockSer |  |
| stockLogic | stockLogicSer, stockDataSer,PO,utilityLogic,dataFactroySer |
| stockDataSer | JavaRMI,PO |
| stockData | dataBaseUtility,PO, stockDataSer |
| checkUI | checkLogicSer,界面类库包,VO |
| checkSer |  |
| checkLogic | checkLogicSer, checkDataSer,PO,utilityLogic,dataFactroySer |
| checkDataSer | JavaRMI,PO |
| checkData | dataBaseUtility,PO, checkDataSer |

快递物流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 快递物流系统客户端开发包图



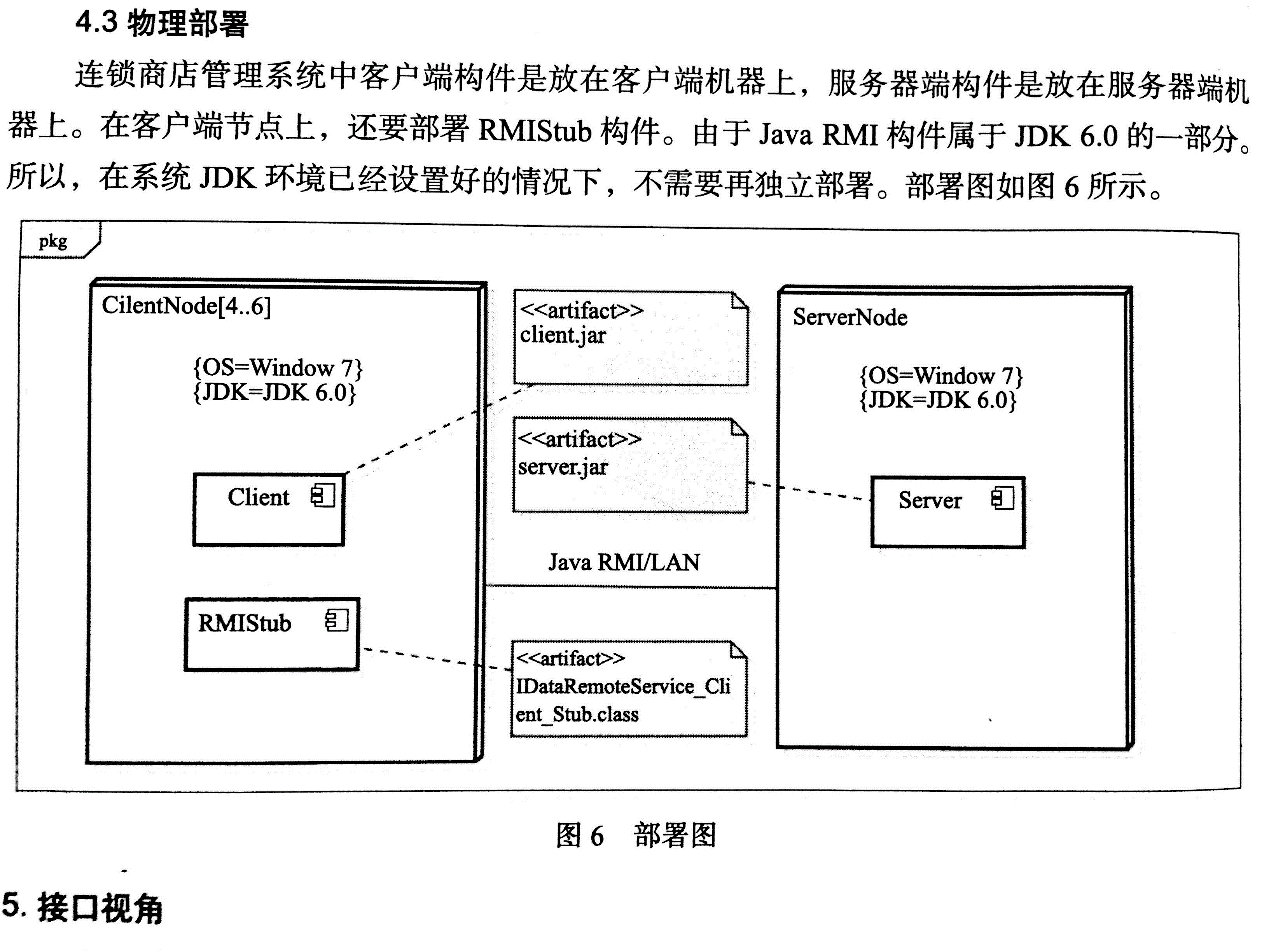
图4 快递物流系统服务器端开发包图

4.2 运行时进程

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



4.3 物理部署

快递物流管理系统中客户端构建是放在客户端机器上。在客户端节点上，还要部署RMIStub构件。由于Java RMI构件属于JDK7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。部署图如图所示。

1. 接口视角
   1. 模块的职责

客户端模块和服务器端模块分别如图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。

|  |
| --- |
| 展示层 |

|  |
| --- |
| 启动模块 |

|  |
| --- |
| 逻辑层 |

|  |
| --- |
| 网络模块 |

图7 客户端模块视图

|  |
| --- |
| 启动模块 |

|  |
| --- |
| 网络模块 |

|  |
| --- |
| 数据层 |

图8 服务端模块视图

**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 | 基于窗口的快递物流系统客户端用户界面 |
| 逻辑层 | 对于用户界面的输入进行响应并进行逻辑处理 |
| 客户端网络模块 | 利用Java RMI机制查找RMI服务 |

**表3 服务端各层的职责**

|  |  |
| --- | --- |
| 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 | 负责数据持久化及数据访问接口 |
| 服务端网络模块 | 利用Java RMI机制开启RMI服务，注册RMI服务 |

每一层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 |
| userLogicSer  transitInfoLogicSer  orderNewLogicSer  orderApproveLogicSer  accountLogicSer  cityLogicSer  workOrgManLogicSer  stockLogicSer  checkLogicSer | 客户端展示层 | 客户端业务逻辑层 |
| userDataSer  transitInfoDataSer  orderNewDataSer  orderApproveDataSer  accountDataSer  cityDataSer  workOrgManDataSer  stockDataSer  checkDataSer  DatabaseFactory | 客户端逻辑层 | 服务端数据层 |

借用订单信息输入用例来说明层之间的调用，如图9所示。每一层之间都是由上层依赖了一个接口（需接口），而下层实现这个接口（供接口）。orderNewLogicSer提供了orderNew界面所需要的所有业务逻辑功能。orderNewDataSer提供了对数据库的增、查等操作。这样的实现就大大降低了层与层之间的耦合。



图9 订单信息输入用例层之间调用的接口

* 1. 用户界面层的分解

根据需求，系统存在52个用户界面：登录界面，物流信息查询界面，快递员分界面，营业厅业务员分界面，中转中心业务员分界面，财务人员分界面，总

经理分界面，订单创建界面，订单查询界面，收件输入界面，营业中心装车单创建界面，营业中心接收单创建界面，派件单创建界面，收款单创建界面，车辆信息管理界面，车辆信息列表显示界面，车辆信息新增修改界面，司机信息管理界面，司机信息列表显示界面，司机信息新增修改界面，中转中心装运单创建界面，中转中心接收单创建界面，中转中心装车单创建界面，库存管理界面，库存报警界面，入库单创建界面，出库单创建界面，库存查看界面，库存盘点界面，库存分区调整界面，账户管理界面，账户信息查看界面，账户信息修改界面，收款单查看界面，付款单创建界面，成本收益表创建界面，经营情况表创建界面，期初建账界面，期初信息查询界面，薪水策略管理界面，薪水策略制定界面，薪水策略查看界面，机构管理界面，人员管理界面，城市管理界面，单据审批界面，统计分析查看界面，日志记录查询界面，系统用户管理界面，系统用户信息修改界面，系统用户信息查看界面。界面跳转如图所示。



服务器端无用户界面。用户界面类图如图所示。



图11 用户界面类

5.2.1用户界面层模块的职责

如表所示为用户界面层模块的职责。

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 |
| FunctionFrame | 功能Frame, 负责所有功能panel的显示 |
| HintFrame | 提示信息Frame, 负责所有提示信息的显示 |
| InfoFrame | 信息Frame,负责各种信息Panel（除提示信息外）的显示 |
| AccountManPanel | 账户管理界面，负责账户管理（增、删、改）操作 |
| CountInfoPanel | 期初信息显示界面，负责查看期初信息操作 |
| CountPanel | 账的管理界面，负责账的管理操作 |
| OrderApprovePanel | 订单审批界面，负责订单审批管理操作 |
| CityListPanel | 城市列表界面，负责处理城市列表管理操作 |
| ArrivePanel | 中转中心到达单生成界面，负责中转中心到达单生成操作 |
| CenterLoadingPanel | 中转中心装车单生成界面，负责中转中心装车单生成操作 |
| ExpressPanel | 快递单生成界面，负责订单信息输入操作 |
| HallLoadingPanel | 营业厅装车单生成界面，负责营业厅装车单生成操作 |
| IncomePanel | 收款单生成界面，负责收款单生成操作 |
| PaymentPanel | 付款单生成界面，负责付款单生成操作 |
| ReceivePanel | 收件单生成界面，负责收件信息输入操作 |
| RecipientPanel | 接收单生成界面，负责接收单生成操作 |
| SendPanel | 派件单生成界面，负责派件单生成操作 |
| StockInPanel | 库存入库单界面，负责处理入库单操作 |
| StockOutPanel | 库存出库单界面，负责处理出库单操作 |
| TransitPanel | 中转单生成界面，负责中转单生成操作 |
| StockCheckNowPanel | 库存查看界面，负责显示库存当前信息 |
| StockCheckPanel | 库存查看界面，负责显示库存信息 |
| StockWarningPanel | 库存警告界面，负责显示库存警告信息界面 |
| StockDividePanel | 库存分区界面，负责处理库存中架信息的管理操作 |
| TransitInfoInqPanel | 物流信息查询界面，负责物流信息查询操作 |
| LoginPanel | 登录界面，负责处理登录操作 |
| UserInfoPanel | 用户信息界面，负责显示用户信息 |
| UserListPanel | 用户列表界面，负责显示所有的用户 |
| CollectFormPanel | 成本收益表显示界面，负责成本收益表查看操作 |
| DiaryListPanel | 日志列表界面，负责日志查看操作 |
| IncomeListPanel | 收款信息列表显示界面，负责收款信息查看操作 |
| StatementSheetPanel | 经营情况表显示界面，负责经营情况表查看操作 |
| DriverInfoPanel | 司机信息管理界面，负责司机信息管理操作 |
| DriverListPanel | 司机信息列表显示界面，负责司机信息查看操作 |
| VanInfoPanel | 车辆信息管理界面，负责车辆信息管理操作 |
| VanListPanel | 车辆信息列表显示界面，负责司机信息查看操作 |
| OrgListPanel | 机构列表及信息管理界面，负责处理机构信息管理操作 |
| WageDraftPanel | 薪水制定界面，负责薪水制定操作 |
| WorkListPanel | 人员列表管理界面，负责人员列表管理操作 |
| WorkWageListPanel | 人员工资管理界面，负责人员工资管理操作 |

5.2.2 用户界面层的接口规范

UI模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| MainFrame | 语法 | | public MainFrame() |
| 前置条件 | | 系统启动 |
| 后置条件 | | 显示主界面 |
| HintFrame | 语法 | | public HintFrame() |
| 前置条件 | | 有需要显示的提示信息 |
| 后置条件 | | 显示相应的提示信息 |
| InfoFrame | 语法 | | public InfoFrame() |
| 前置条件 | | 用户查看各类信息 |
| 后置条件 | | 显示相应信息的Panel |
| FunctionFrame | 语法 | | public FunctionFrame() |
| 前置条件 | | 打开功能界面 |
| 后置条件 | | 显示功能界面及对应的功能Panel |
| \*Panel.add\* | 语法 | | public boolean add\*(\*VO vo) |
| 前置条件 | | 用户新增对象（包括车辆、司机、各种单据、账、账户、城市、机构、人员、用户等） |
| 后置条件 | | 显示编辑新增对象信息界面 |
| \*Panel.delete\* | 语法 | | public boolean delete\*(String id) |
| 前置条件 | | 用户删除对象（包括车辆、司机、各种单据、账、账户、城市、机构、人员、用户等） |
| 后置条件 | | 返回删除是否成功的提示信息界面 |
| \*Panel.modify\* | 语法 | | public boolean modify\*(\*VO vo) |
| 前置条件 | | 用户修改对象（包括车辆、司机、各种单据、账、账户、城市、机构、人员、用户等） |
| 后置条件 | | 显示编辑修改对象信息界面 |
| \*Panel.check\* | 语法 | | public \*VO check\*() |
| 前置条件 | | 用户查看对象（包括车辆、司机、各种单据、账、账户、用户等） |
| 后置条件 | | 显示要被查看的对象信息界面 |
| \*Panel.check\* | 语法 | | public ArrayList<\*VO> check\*() |
| 前置条件 | | 用户查看对象列表（包括车辆、司机、各种单据、账、账户、城市、机构、人员、用户、日志记录等） |
| 后置条件 | | 显示要被查看的对象列表界面 |
| \*Panel.confirm | 语法 | | public boolean modify\*(\*VO vo) |
| 前置条件 | | 用户确认操作（包括新增、修改等） |
| 后置条件 | | 返回操作是否成功的提示信息 |
| \*Panel.close | 语法 | | public boolean close() |
| 前置条件 | | 用户选择返回上一级或关闭当前界面 |
| 后置条件 | | 系统返回上一级 |
| 需要的接口（需接口） | | | |
| 服务名 | | 服务 | |
| \*LogicService | | 每个提示信息和信息查看都有一个相应的业务逻辑接口，相应增删改查的增删改查操作对应相应的业务逻辑接口，具体接口参见下面具体UI模块依赖的逻辑接口 | |

UserUI模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| UserInfoPanel | 语法 | | public UserInfoPanel (); |
| 前置条件 | | 用户请求打开用户信息界面 |
| 后置条件 | | 显示用户信息界面 |
| UserListPanel | 语法 | | public UserListPanel (); |
| 前置条件 | | 用户请求查看用户列表 |
| 后置条件 | | 显示用户列表 |
| LoginPanel | 语法 | | public LoginPanel() |
| 前置条件 | | 用户打开系统 |
| 后置条件 | | 显示登录界面 |
| 需要的接口（需接口） | | | |
| 服务名 | | 服务 | |
| UserLogicSer.LogicSer | | 负责提供登录的逻辑接口 | |
| LogicSer.login(String id,String password) | | 负责处理登录的接口 | |
| UserLogicSer.UserSer | | 负责提供用户管理的逻辑接口 | |
| UserSer.add(UserVO user) | | 负责新建用户的接口 | |
| UserSer.delete(String id) | | 负责删除用户的接口 | |
| UserSer.revise(UserVO user) | | 负责修改用户的接口 | |
| UserSer.check(String id) | | 负责查看用户的接口 | |
| UserSer.list() | | 负责获取用户列表的接口 | |

OrderNewUI模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| ExpressPanel | 语法 | | public ExpressPanel() |
| 前置条件 | | 用户选择生成快递单 |
| 后置条件 | | 显示订单信息输入界面 |
| ReceivePanel | 语法 | | public RecieivePanel() |
| 前置条件 | | 用户选择收件 |
| 后置条件 | | 显示收件信息输入界面 |
| PaymentPanel | 语法 | | public PaymentPanel() |
| 前置条件 | | 用户选择新建付款单 |
| 后置条件 | | 显示付款单生成界面 |
| HallLoadingPanel | 语法 | | public HallLoadingPanel() |
| 前置条件 | | 用户选择生成营业厅装车单 |
| 后置条件 | | 显示营业厅装车单生成界面 |
| IncomePanel | 语法 | | public IncomePanel() |
| 前置条件 | | 用户选择生成收款单 |
| 后置条件 | | 显示收款单生成界面 |
| RecipientPanel | 语法 | | public RecipientPanel() |
| 前置条件 | | 用户选择生成接收单 |
| 后置条件 | | 显示接收单生成界面 |
| SendPanel | 语法 | | public SendPanel() |
| 前置条件 | | 用户选择生成派件单 |
| 后置条件 | | 显示派件单生成界面 |
| ArrivePanel | 语法 | | public ArrivePanel() |
| 前置条件 | | 用户选择生成到达单 |
| 后置条件 | | 显示到达单生成界面 |
| CenterLoadingPanel | 语法 | | public CenterLoadingPanel() |
| 前置条件 | | 用户选择生成中转中心装车单 |
| 后置条件 | | 显示中转中心装车单生成界面 |
| TransitPanel | 语法 | | public TransitPanel() |
| 前置条件 | | 用户选择生成中转单 |
| 后置条件 | | 显示中转单生成界面 |
| StockInPanel | 语法 | | public StockInPanel() |
| 前置条件 | | 用户选择新建入库单 |
| 后置条件 | | 显示入库单界面 |
| StockOutPanel | 语法 | | public StockOutPanel() |
| 前置条件 | | 用户选择新建出库单 |
| 后置条件 | | 显示出库单界面 |
| 需要的接口（需接口） | | | |
| 服务名 | | 服务 | |
| OrderNewSer.Create | | 负责提供新建单据的逻辑接口 | |
| OrderNewSer.getCities | | 负责获取城市选项的逻辑接口 | |
| OrderNewSer.getHalls | | 负责获取营业厅选项的逻辑接口 | |
| OrderNewSer.getOrgs | | 负责获取到达地选项的逻辑接口 | |
| OrderNewSer.getVans | | 获取车辆代号选项的逻辑接口 | |
| OrderNewSer.getCouriers(String id) | | 获取快递员信息的逻辑接口 | |
| OrderNewSer.getExpress(String id) | | 获取订单信息的逻辑接口 | |

TransitInfoUI模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| TransitInfoInqPanel | 语法 | public TransitInfoInqPanel () |
| 前置条件 | 用户登录系统 |
| 后置条件 | 显示物流信息查询界面 |
| 需要的接口（需接口） | | |

|  |  |
| --- | --- |
| TransitInfoLogicSer.enterBarcodeCounrier(String  Barcode) | 查询订单信息界面的逻辑接口 |
| TransitInfoLogicSer.enterBarcodeCustomer(String  Barcode) | 查询物流轨迹和货运状态界面的逻辑接口 |

OrderApproveUI模块的接口规范

|  |
| --- |
| 提供的服务（供接口） |

|  |  |  |
| --- | --- | --- |
| OrderApprovePanel | 语法 | public void OrderApprovePanel () |
| 前置条件 | 用户选择单据审批功能管理 |
| 后置条件 | 显示订单审批界面 |

|  |
| --- |
| 需要的接口（需接口） |

|  |  |
| --- | --- |
| orderApproveLogicSer.askExamine() | 提供返回单据信息列表的逻辑接口 |
| orderApproveLogicSer. examine (PO po) | 更新订单审批状态的业务逻辑接口 |
| orderApproveLogicSer.getOrder(String id) | 提供返回订单信息的逻辑接口 |

AccountUI模块的接口规范

|  |
| --- |
| 提供的服务（供接口） |

|  |  |  |
| --- | --- | --- |
| AccountManPanel | 语法 | public AccountManPanel() |
| 前置条件 | 用户选择管理账户信息 |
| 后置条件 | 显示账户管理界面 |
| CountInfoPanel | 语法 | public CountInfoPanel() |
| 前置条件 | 用户选择查看期初信息 |
| 后置条件 | 显示期初信息界面 |
| CountPanel | 语法 | public CountPanel() |
| 前置条件 | 用户选择新建一套账 |
| 后置条件 | 显示新建账界面 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| 服务名 | 服务 |
| AccountLogicSer.newCount(countPO po) | 新建一套账的界面的逻辑接口 |
| AccountLogicSer.checkInitInfo() | 查看期初信息界面的逻辑接口 |
| AccountLogicSer. addAccount(accountPO po) | 新增账户的界面的逻辑接口 |
| AccountLogicSer.deleteAccount(String id) | 删除账户的界面的逻辑接口 |
| AccountLogicSer.reviseAccount(accountPO po) | 修改账户信息界面的逻辑接口 |
| AccountLogicSer.accountList() | 查看账户列表界面的逻辑接口 |
| AccountLogicSer.checkAccount() | 查看某一账户信息界面的逻辑接口 |
| AccountLogicSer.searchAccount(String keywords) | 检索账户界面的逻辑接口 |

CityUI模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CityListPanel | 语法 | public CityListPanel () |
| 前置条件 | 用户选择城市管理功能 |
| 后置条件 | 显示城市列表及城市信息管理界面 |

|  |
| --- |
| 需要的接口（需接口） |

|  |  |
| --- | --- |
| 服务名 | 服务 |

|  |  |
| --- | --- |
| CityLogicSer.addCity(String name) | 新增城市界面的逻辑接口 |
| CityLogicSer.reviseCity(CityVO city) | 修改城市信息界面的逻辑接口 |
| CityLogicSer.reviseCities (CitiesVO cities) | 修改城市间距离与价格信息界面的逻辑接口 |
| City.cityList | 负责获得城市列表 |

WorkOrgManUI模块的接口规范

|  |
| --- |
| 提供的服务（供接口） |

|  |  |  |
| --- | --- | --- |
| WageDraftPanel | 语法 | public WageDraftPanel () |
| 前置条件 | 用户选择薪水制定 |
| 后置条件 | 显示薪水制定界面 |
| WorkListPanel | 语法 | public WorkListPanel () |
| 前置条件 | 用户选择人员管理功能 |
| 后置条件 | 显示人员信息及人员列表界面 |
| WorkWageListPanel | 语法 | public WorkWageListPanel () |
| 前置条件 | 用户选薪水查看功能 |
| 后置条件 | 显示薪水列表及薪水信息界面 |
| DriverInfoPanel | 语法 | public DriverInfoPanel() |
| 前置条件 | 用户选择管理司机信息 |
| 后置条件 | 显示司机信息界面 |
| DriverListPanel | 语法 | public DriverListPanel() |
| 前置条件 | 用户选择查看司机信息 |
| 后置条件 | 显示司机信息列表界面 |
| VanInfoPanel | 语法 | public VanInfoPanel() |
| 前置条件 | 用户选择管理车辆信息 |
| 后置条件 | 显示车辆信息界面 |
| VanListPanel | 语法 | public VanListPanel() |
| 前置条件 | 用户选择查看车辆信息 |
| 后置条件 | 显示车辆信息列表界面 |
| OrgInfoPanel | 语法 | public OrgInfoPanel () |
| 前置条件 | 用户选择机构信息管理 |
| 后置条件 | 显示机构信息界面 |
| OrgListPanel | 语法 | public OrgListPanel () |
| 前置条件 | 用户选择机构列表管理 |
| 后置条件 | 显示机构信息管理界面 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| 服务名 | 服务 |
| WorkOrgManLSer.addOrg(OrgVO org) | 负责提供新建机构的逻辑接口 |
| WorkOrgManLSer.deleteOrg(String id) | 负责提供删除机构的逻辑接口 |
| WorkOrgManLSer.reviseOrg(OrgVO org) | 负责提供修改机构的逻辑接口 |
| WorkOrgManLSer. orgList | 负责提供显示机构列表的逻辑接口 |
| WorkOrgManLSer.checkOrg(String id) | 负责提供查看机构信息的逻辑接口 |
| WorkOrgManLSer.searchOrg(String keywords) | 负责提供搜索特定机构的逻辑接口 |
| WorkOrgManLSer.addWork（WorkVO work） | 负责提供新建人员的逻辑接口 |
| WorkOrgManLSer.deleteWork(String id) | 负责提供删除人员的逻辑接口 |
| WorkOrgManLSer.reviseWork(WorkVO work) | 负责提供修改人员的逻辑接口 |
| WorkOrgManLSer. workList | 负责提供显示人员列表的逻辑接口 |
| WorkOrgManLSer.checkWork(String id) | 负责提供查看人员信息的逻辑接口 |
| WorkOrgManLSer.searchWork(String keywords) | 负责提供搜索特定人员的逻辑接口 |
| WorkOrgManLSer.addVan(VanVO van） | 负责提供新建车辆的逻辑接口 |
| WorkOrgManLSer.deleteVan (String id) | 负责提供删除车辆的逻辑接口 |
| WorkOrgManLSer.reviseVan(VanVO driver) | 负责提供修改车辆的逻辑接口 |
| WorkOrgManLSer.searchVan(String keyword) | 负责提供关键词检索车辆的逻辑接口 |
| WorkOrgManLSer.addDriver(VanVO driver） | 负责提供新建司机的逻辑接口 |
| WorkOrgManLSer.deleteDriver (String id) | 负责提供删除司机的逻辑接口 |
| WorkOrgManLSer.reviseDriver (DriverVO driver) | 负责提供修改司机的逻辑接口 |
| WorkOrgManLSer. driverList | 负责提供显示司机列表的逻辑接口 |
| WorkOrgManLSer.checkDriver (String id) | 负责提供查看司机信息的逻辑接口 |
| WorkOrgManLSer.getWage(String id) | 负责薪水查看的逻辑接口 |
| WorkOrgManLSer.reviseWage(WageVO wage) | 负责薪水制定的逻辑接口 |

StockUI模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| StockWarningPanel | 语法 | | public StockWarningPanel() |
| 前置条件 | | 用户选择库存报警 |
| 后置条件 | | 显示库存报警界面 |
| StockCheckPanel | 语法 | | public StockCheckPanel() |
| 前置条件 | | 用户选择库存查看 |
| 后置条件 | | 显示库存查看界面 |
| StockCheckNowPanel | 语法 | | public StockNowPanel() |
| 前置条件 | | 用户选择库存盘点 |
| 后置条件 | | 显示当前库存情况 |
| StockDividePanel | 语法 | | public StockDividePanel() |
| 前置条件 | | 用户选择库存分区 |
| 后置条件 | | 显示库存分区界面和架列表 |
| 需要的接口（需接口） | | | |
| 服务名 | | 服务 | |
| StockLogicSer.WarnSer | | 负责提供库存报警服务的接口 | |
| WarnSer.setWarning(int w, Part part, String id) | | 负责设置警戒线 | |
| WarnSer.checkWarning(String id) | | 负责检测库存是否超出警戒线的接口 | |
| WarnSer.getWarning(String id) | | 负责获得警戒线 | |
| StockLogicSer.CheckStockInSer | | 负责提供查看入库服务 | |
| StockLogicSer.CheckStockOutSer | | 负责提供查看出库服务 | |
| CheckSer.checkStock(Date start, Date end, String id) | | 负责提供获得时间段中库存信息的接口 | |
| CheckSer.CheckNow | | 负责提供库存盘点服务 | |
| CheckNowSer.checkNow(String id) | | 负责提供获得当前库存情况的接口 | |
| CheckNowSer.setPoint(String id,Date date) | | 负责提供设置截止点的接口 | |
| CheckNowSer.outputExcel(String location, StockVO stock) | | 负责提供导出库存情况的接口 | |
| StockLogicSer.DivideSer | | 负责提供库存分区服务 | |
| StockDivideSer.searchShelf(String keyword,String id) | | 负责提供关键词检索架的接口 | |
| StockDivideSer.addShelf(ShelfVO shelf, String id) | | 负责提供添加架的接口 | |
| StockDivideSer.deleteShelf(String shelfId, String id) | | 负责提供删除架的接口 | |
| StockDivideSer.reviseShelf(ShelfVO shelf, String id) | | 负责提供修改架的接口 | |
| StockDivideSer.checkShelf(String shelfId, String id) | | 负责提供查看架的接口 | |

CheckUI模块的接口规范

|  |
| --- |
| 提供的服务（供接口） |

|  |  |  |
| --- | --- | --- |
| CollectFormPanel | 语法 | public CollectFormPanel() |
| 前置条件 | 用户选择查看成本收益表 |
| 后置条件 | 显示成本收益表界面 |
| IncomeListPanel | 语法 | public IncomeListPanel() |
| 前置条件 | 用户选择查看收款信息列表 |
| 后置条件 | 显示收款信息列表界面 |
| StatementSheetPanel | 语法 | public StatementSheetPanel() |
| 前置条件 | 用户选择查看经营情况表 |
| 后置条件 | 显示经营情况表界面 |
| StatementSheetPanel | 语法 | public StatementSheetPanel() |
| 前置条件 | 用户选择查看经营情况表 |
| 后置条件 | 显示经营情况表界面 |
| \*panel.outExcel | 语法 | public boolean outExcel() |
| 前置条件 | 用户选择将相关报表导出为Excel |
| 后置条件 | 显示选择导出位置的提示信息界面 |
| \*panel.confirmOutExcel | 语法 | public boolean confirmOutExcel() |
| 前置条件 | 用户确认导出位置 |
| 后置条件 | 显示导入是否成功的提示信息界面 |

|  |  |
| --- | --- |
| 需要的接口（需接口） | |
| 服务名 | 服务 |
| checkLogicSer. seeDiary (Date start, Date end) | 负责日志记录获得的逻辑接口 |
| checkLogicSer. seeStatementSheet (Date start, Date end) | 负责收款单记录查询的逻辑接口 |
| checkLogicSer. seeCollectRecord (Date date, String id) | 负责营业厅当日收款记录查询的逻辑接口 |
| checkLogicSer.getHall() | 负责营业厅的编号和名称查询的逻辑接口 |
| checkLogicSer. total (ArrayList<IncomeVO> income) | 负责合计的收款总额的逻辑接口 |
| Check.seeDiary | 负责查看相应的日志 |

UserPanel的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| AdministerPanel | 语法 | | public AdministerPanel (); |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示管理员界面 |
| BusinessHallPanel | 语法 | | public BusinessHallPanel () |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示营业厅业务员界面 |
| CourierPanel | 语法 | | public CourierPanel (); |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示快递员界面 |
| FinanceSuperPanel | 语法 | | public FinanceSuperPanel () |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示高级财务人员界面 |
| FinancePanel | 语法 | | public FinancePanel (); |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示财务人员界面 |
| ManagerPanel | 语法 | | public ManagerPanel () |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示总经理界面 |
| StockManagePanel | 语法 | | public StockManagePanel (); |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示库存管理人员界面 |
| TransitCenterPanel | 语法 | | public TransitCenterPanel () |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示中转中心业务员界面 |
| TransitInfoInqPanel | 语法 | | public TransitInfoInqPanel () |
| 前置条件 | | 用户登录系统 |
| 后置条件 | | 显示物流信息查询界面 |
| 需要的接口（需接口） | | | |
| UserLogicSer.LogicSer | | 负责提供登录的逻辑接口 | |

5.2.3 用户界面模块设计原理

用户界面利用Java的Swing和AWT库来实现。

5.3 业务逻辑层的分解

业务逻辑层包括多个针对界面的业务逻辑处理对象。例如，User对象负责处理用户管理和登录界面的业务逻辑，OrderNew对象负责新建单据界面的业务逻辑。业务逻辑层的设计如图12所示。



图12 业务逻辑层的设计

5.3.1 业务逻辑层模块的职责

业务逻辑层模块的职责如表所示。

|  |  |
| --- | --- |
| 模块 | 职责 |
| userLogic | 负责实现用户管理和登录界面所需要的服务 |
| transitInfoLogic | 负责实现物流信息查询界面所需要的服务 |
| orderNewLogic | 负责实现单据生成界面所需要的服务 |
| orderApproveLogic | 负责实现单据审批界面所需要的服务 |
| accountLogic | 负责实现账户管理界面所需要的服务 |
| cityLogic | 负责实现城市管理界面所需要的服务 |
| workOrgManLogic | 负责实现人员机构管理界面所需要的服务 |
| stockLogic | 负责实现库存管理界面所需要的服务 |
| checkLogic | 负责实现统计报表与查看日志界面所需要的服务 |

5.3.2 业务逻辑层模块的接口规范

各逻辑层模块接口规范如后表所示。

userLogic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 | id和password均输入 |
| 后置条件 | 查找是否存在相应的用户，根据输入的id和password返回结果和用户信息 |
| User.newUser | 语法 | public boolean newUser(UserVO user) |
| 前置条件 | 输入所有用户信息后，确认新建 |
| 后置条件 | 将新用户信息加入数据库，返回加入结果 |
| User.deleteUser | 语法 | public boolean deleteUser(String id) |
| 前置条件 | 选择了一个用户，选择删除 |
| 后置条件 | 在数据库中除去相应用户的信息，返回删除结果 |
| User.reviseUser | 语法 | public boolean reviseUser(UserVO user) |
| 前置条件 | 修改了用户的信息，确认 |
| 后置条件 | 在数据库中修改相应用户的信息，返回修改结果 |
| User.checkUser | 语法 | public ResultMessage checkUser(String id) |
| 前置条件 | 选择了一个用户，选择查看 |
| 后置条件 | 数据库中查找相应用户信息，返回信息和查看结果 |
| User.userList | 语法 | public ResultMessage userList() |
| 前置条件 | 用户选择查看用户列表 |
| 后置条件 | 返回数据库中所有用户的列表，包含用户的账号、姓名和所属机构 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| DatabaseFactory.getUserData() | 得到User数据库的服务的引用 | |
| UserDataSer.addUser(UserPO po) | 在数据库中添加UserPO对象 | |
| UserDataSer.deleteUser(String id) | 在数据库中删除id所指向的对象 | |
| UserDataSer.reviseUser(UserPO po) | 在数据库中修改UserPO中id指向的对象的信息 | |
| UserDataSer.findUser(String id) | 在数据库中找到相应用户的信息 | |
| UserDataSer.list() | 在数据库中获得用户列表 | |

transitInfo模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| TransitInfo.enterBarcodeCustom | 语法 | Public ResultMessage enterBarcodeCustom(String barcode); |
| 前置条件 | barcode符合输入规则 |
| 后置条件 | 查找是否存在相应的订单，据输入的barcode返回相应订单的物流信息和货运状态 |
| TransitInfo.enterBarcodeCourier | 语法 | Public ResultMessage enterBarcodeCourier(String barcode); |
| 前置条件 | barcode符合输入规则 |
| 后置条件 | 查找是否存在相应的订单，根据输入的barcode返回该订单所有信息 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| transitInfoDataSer.find(String  barcode) | 根据barcode进行查找订单对象 | |
| transitInfoDataSer.findTransit(String  barcode) | 根据barcode查找相应订单的物流信息和货运状态 | |

orderNewLogic模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderNew.Create | 语法 | public boolean Create(ArriveVO order) |
| 前置条件 | 到达单的各项信息已输入并确认 |
| 后置条件 | 将该新增到达单信息上传至数据库，返回上传是否成功 |
| OrderNew.getCities | 语法 | public ResultMessage getCities() |
| 前置条件 | 用户输入城市 |
| 后置条件 | 在数据库中查找所有城市，并返回城市名称列表 |
| OrderNew.Create | 语法 | public boolean Create(CenterLoadingVO order) |
| 前置条件 | 中转中心装车单的各项信息已输入并确认 |
| 后置条件 | 将该新增中转中心装车单信息上传至数据库，返回上传是否成功 |
| OrderNew.getHalls | 语法 | public ResultMessage getHalls() |
| 前置条件 | 用户输入到达地（营业厅） |
| 后置条件 | 在数据库中查找所有营业厅，并返回营业厅名称列表 |
| OrderNew.Create | 语法 | public boolean Create(ExpressVO order) |
| 前置条件 | 寄件单的各项信息已输入并确认 |
| 后置条件 | 将该寄件单信息上传至数据库，返回上传是否成功 |
| OrderNew.Create | 语法 | public boolean Create(HallLoadingVO order) |
| 前置条件 | 营业厅装车单的各项信息已输入并确认 |
| 后置条件 | 将该新增营业厅装车单上传至数据库，返回上传是否成功 |
| OrderNew.getOrgs | 语法 | public ResultMessage getOrgs() |
| 前置条件 | 用户输入到达地 |
| 后置条件 | 查找数据库中所有到达地，返回到达地名称列表 |
| OrderNew.getVans | 语法 | public ResultMessage getVan() |
| 前置条件 | 用户输入车辆代号 |
| 后置条件 | 查找数据库中所有车辆，返回车辆代号列表 |
| OrderNew.Create | 语法 | public boolean Create(IncomeVO order) |
| 前置条件 | 收款单的各项信息已输入并确认 |
| 后置条件 | 将该新增收款单信息上传至数据库，返回上传是否成功 |
| OrderNew.getCouriers | 语法 | public ResultMessage getCouriers(String id) |
| 前置条件 | 该id对应的营业厅在数据库中存在 |
| 后置条件 | 根据id在数据库中查找对应营业厅并返回该营业厅快递员姓名列表 |
| OrderNew.Create | 语法 | public boolean Create(PaymentVO order) |
| 前置条件 | 付款单的各项信息已输入并确认 |
| 后置条件 | 将该新增付款单信息上传至数据库，返回上传是否成功 |
| OrderNew.Create | 语法 | public boolean Create(ReceiveVO order) |
| 前置条件 | 收件单的各项信息已输入并确认 |
| 后置条件 | 将该新增收件信息上传至数据库，返回上传是否成功 |
| OrderNew.getExpress | 语法 | public ResultMessage getExpress(String id) |
| 前置条件 | 该id对应的订单信息在数据库中存在 |
| 后置条件 | 根据id在数据库中查找对应的订单信息并返回 |
| OrderNew.Create | 语法 | public boolean Create(RecipientVO order) |
| 前置条件 | 接收单的各项信息已输入并确认 |
| 后置条件 | 将该新增接收单信息上传至数据库，返回上传是否成功 |
| OrderNew.Create | 语法 | public boolean Create(SendOrderVO order) |
| 前置条件 | 派件单的各项信息已输入并确认 |
| 后置条件 | 将该新增派件信息上传至数据库，返回上传是否成功 |
| OrderNew.Create | 语法 | public boolean Create(StockInOrderVO order) |
| 前置条件 | 入库单的各项信息已输入并确认 |
| 后置条件 | 将该新增入库信息上传至数据库，返回上传是否成功 |
| OrderNew.Create | 语法 | public boolean Create(StockoutOrderVO order) |
| 前置条件 | 出库单的各项信息已输入并确认 |
| 后置条件 | 将该新增出库信息上传至数据库，返回上传是否成功 |
| OrderNew.Create | 语法 | public boolean Create(TransitOrderVO order) |
| 前置条件 | 中转单的各项信息已输入并确认 |
| 后置条件 | 将该新增中转单信息上传至数据库，返回上传是否成功 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| DatabaseFactory.getOrderData | 得到Order数据库的服务的引用 | |
| orderNewDataSer.create(CenterLoadingPO order) | 在数据库中添加CenterLoadingPO对象 | |
| orderNewDataSer. getHalls() | 在数据库中获取并返回营业厅信息 | |
| orderNewDataSer.create (ArrivePO order) | 在数据库中添加ArrivePO对象 | |
| orderNewDataSer. getCities() | 在数据库中获取并返回城市信息 | |
| orderNewDataSer.create(ExpressPO order) | 在数据库中添加ExpressPO对象 | |
| orderNewDataSer.create(HallLoadingPO order) | 在数据库中添加HallLoadingPO对象 | |
| orderNewDataSer. getOrgs() | 在数据库中获取并返回机构信息 | |
| orderNewDataSer. getVans() | 在数据库中获取并返回货车信息 | |
| orderNewDataSer.create(IncomePO order) | 在数据库中添加IncomePO对象 | |
| orderNewDataSer .getCouriers() | 在数据库中获取并返回快递员信息 | |
| orderNewDataSer.create(PaymentPO order) | 在数据库中添加PaymentPO对象 | |
| orderNewDataSer.create(ReceivePO order) | 在数据库中添加ReceivePO对象 | |
| orderNewDataSer.create(RecipientPO order) | 在数据库中添加RecipientPO对象 | |
| orderNewDataSer.create(SendPO order) | 在数据库中添加SendPO对象 | |
| orderNewDataSer.create(StockOutPO order) | 在数据库中添加StockOutPO对象 | |
| orderNewDataSer.create(StockInPO order) | 在数据库中添加StockInPO对象 | |
| orderNewDataSer.create(TransitPO order) | 在数据库中添加TransitPO对象 | |

workOrgManLogic模块的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| WorkOrgMan.addOrg | 语法 | | public boolean add(OrgVO org); |
| 前置条件 | | 用户输入了新机构的信息 |
| 后置条件 | | 将新增的机构信息加入数据库，返回加入结果 |
| WorkOrgMan.deleteOrg | 语法 | | public boolean deleteOrg(String id) |
| 前置条件 | | 用户选择一个机构进行删除 |
| 后置条件 | | 根据机构名在数据库中删除该账户信息，返回删除结果 |
| WorkOrgMan.reviseOrg | 语法 | | public boolean reviseOrg(OrgVO org) |
| 前置条件 | | 修改了机构信息并确认 |
| 后置条件 | | 更新数据库中该机构的信息，返回更新结果 |
| WorkOrgMan. orgList | 语法 | | public ResultMessage orgList() |
| 前置条件 | | 用户选择查看机构列表 |
| 后置条件 | | 返回数据库中所有机构的列表 |
| WorkOrgMan.checkOrg | 语法 | | public ResultMessage checkOrg(String id) |
| 前置条件 | | 用户选择一个机构进行查看 |
| 后置条件 | | 返回该机构信息 |
| WorkOrgMan.searchOrg | 语法 | | public ResultMessage searchOrg(String keywords) |
| 前置条件 | | 用户检索机构并输入关键字 |
| 后置条件 | | 根据该关键字查找符合条件的机构并返回 |
| WorkOrgMan.addWork | 语法 | | public ResultMessage addWork(WorkVO work); |
| 前置条件 | | 用户输入了新人员信息 |
| 后置条件 | | 将新增的人员信息加入数据库，返回加入结果 |
| WorkOrgMan.deleteWork | 语法 | | public ResultMessage deleteWork(String id) |
| 前置条件 | | 用户选择一个人员进行删除 |
| 后置条件 | | 根据人员名在数据库中删除该人员信息，返回删除结果 |
| WorkOrgMan.reviseWork | 语法 | | public ResultMessage reviseWork(WorkVO work) |
| 前置条件 | | 修改了人员信息并确认 |
| 后置条件 | | 更新数据库中该人员的信息，返回更新结果 |
| WorkOrgMan. workList | 语法 | | public ResultMessage workList() |
| 前置条件 | | 用户选择查看人员列表 |
| 后置条件 | | 返回数据库中所有人员的列表 |
| WorkOrgMan.checkWork | 语法 | | public ResultMessage checkWork(String id) |
| 前置条件 | | 用户选择一个人员进行查看 |
| 后置条件 | | 返回该人员信息 |
| WorkOrgMan.searchWork | 语法 | | public ResultMessage searchWork(String keywords) |
| 前置条件 | | 用户检索人员并输入关键字 |
| 后置条件 | | 根据该关键字查找符合条件的人员并返回 |
| WorkOrgMan.addDriver | 语法 | | public boolean addDriver(DriverVO driver) |
| 前置条件 | | 用户选择添加司机信息 |
| 后置条件 | | 在数据库中增加该司机信息，并返回添加结果 |
| WorkOrgMan.deleteDriver | 语法 | | public boolean deleteDriver(String id) |
| 前置条件 | | 该id对应的司机信息在数据库中存在 |
| 后置条件 | | 根据id在数据库中删除该司机信息，并返回删除结果 |
| WorkOrgMan.reviseDriver | 语法 | | public boolean reviseDriver(DriverVO driver) |
| 前置条件 | | 该司机在数据库中存在 |
| 后置条件 | | 在数据库中更新该司机信息，并返回更新结果 |
| WorkOrgMan.checkDriver | 语法 | | public ResultMessage checkDriver(String id) |
| 前置条件 | | 该id对应的司机信息在数据库中存在 |
| 后置条件 | | 根据id在数据库中查找该司机信息并返回 |
| WorkOrgMan.driverList | 语法 | | public ResultMessage driverList() |
| 前置条件 | | 用户查看司机信息列表 |
| 后置条件 | | 返回数据库中的司机信息列表 |
| WorkOrgMan.addVan | 语法 | | public boolean addVan (VanVO van) |
| 前置条件 | | 用户选择添加车辆信息 |
| 后置条件 | | 在数据库中增加该车辆信息，并返回添加结果 |
| WorkOrgMan.deleteVan | 语法 | | public boolean deleteVan (String id) |
| 前置条件 | | 该id对应的车辆信息在数据库中存在 |
| 后置条件 | | 根据id在数据库中删除该车辆信息，并返回删除结果 |
| WorkOrgMan.reviseVan | 语法 | | public boolean reviseVan (VanVO van) |
| 前置条件 | | 该车辆在数据库中存在 |
| 后置条件 | | 在数据库中更新该车辆信息，并返回更新结果 |
| WorkOrgMan.checkVan | 语法 | | public ResultMessage checkVan (String id) |
| 前置条件 | | 该id对应的车辆信息在数据库中存在 |
| 后置条件 | | 根据id在数据库中查找该车辆信息并返回 |
| WorkOrgMan.vanList | 语法 | | public ResultMessage vanList() |
| 前置条件 | | 用户查看车辆信息列表 |
| 后置条件 | | 返回数据库中的车辆信息列表 |
| WorkOrgMan.searchVan | 语法 | | public ResultMessage searchVan(String keywords) |
| 前置条件 | | 用户检索车辆并输入关键字 |
| 后置条件 | | 根据该关键字查找符合条件的车辆并返回车辆信息列表 |
| WorkOrgMan.reviseWage | 语法 | | public boolean reviseWage (ArrayList<WageVO> wages) |
| 前置条件 | | 薪水信息在数据库中存在 |
| 后置条件 | | 在数据库中更新该薪水信息，并返回更新结果 |
| WorkOrgMan.getWage | 语法 | | public ResultMessage getWage() |
| 前置条件 | | 用户查看薪水信息 |
| 后置条件 | | 返回数据库中对应的薪水信息 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| workOrgManDataSer.findWork(String keywords) | | 从数据库中找到含有该关键字的人员，返回数据 | |
| workOrgManDataSer.addWork(WorkPO  po) | | 在数据库中新增一个人员的数据 | |
| workOrgManDataSer.deleteWork(String id) | | 在数据库中删除一个人员的数据 | |
| workOrgManDataSer.reviseWork(WorkPO po) | | 在数据库中修改一个人员的数据 | |
| workOrgManDataSer.checkWork(String id) | | 在数据库中获得一个人员的数据，返回 | |
| DatabaseFactory.getWorkbase() | | 得到Work数据库的服务的引用 | |
| workOrgManDataSer.findOrg(String keywords) | | 在数据库中找到含有该关键字的机构，返回数据 | |
| workOrgManDataSer.addOrg(OrgPO  po) | | 在数据库中新增一个机构的信息 | |
| workOrgManDataSer.deleteOrg(String id  ) | | 在数据库中删除一个机构的信息 | |
| workOrgManDataSer.reviseOrg(OrgPO po) | | 在数据库中修改一个机构的信息 | |
| workOrgManDataSer.checkOrg(String id) | | 在数据库中获得一个机构的信息，返回数据 | |
| DatabaseFactory.getOrgbase | | 得到org数据库的服务的引用 | |
| workOrgManDSer.addDriver(DriverPO  driver) | | 在数据库中新增一个司机的数据 | |
| workOrgManDSer.deleteDriver (String id) | | 在数据库中删除一个司机的数据 | |
| workOrgManDSer.reviseDriver(DriverPO driver) | | 在数据库中修改一个司机的数据 | |
| workOrgManDSer.checkDriver(String id) | | 在数据库中获得一个司机的数据，返回 | |
| DatabaseFactory.getDriverbase() | | 得到Driver数据库的服务的引用 | |
| workOrgManDSer.findVan(String keywords) | | 从数据库中找到含有该关键字的车辆，返回数据 | |
| workOrgManDSer.addVan (VanPO  van) | | 在数据库中新增一个车辆的数据 | |
| workOrgManDSer.deleteVan (String id) | | 在数据库中删除一个车辆的数据 | |
| workOrgManDSer.reviseVan(VanPO van) | | 在数据库中修改一个车辆的数据 | |
| workOrgManDSer.checkVan (String id) | | 在数据库中获得一个车辆的数据，返回 | |
| DatabaseFactory.getVanbase() | | 得到Van数据库的服务的引用 | |
| workOrgManDSer.reviseWage(WagePO wage) | | 在数据库中修改薪水信息 | |
| workOrgManDSer.getWage(String id) | | 在数据库中查找并获取薪水信息 | |

cityLogic模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| City.addCity | 语法 | public boolean addCity(CityVO city); |
| 前置条件 | 增加一个城市 |
| 后置条件 | 将该城市信息加入数据库，返回添加结果 |
| City.reviseCity | 语法 | public boolean reviseCity(CityVO city); |
| 前置条件 | 修改了一个城市信息 |
| 后置条件 | 更新数据库已有信息，返回修改结果 |
| City.reviseCities | 语法 | public boolean reviseCities(CitiesVO city); |
| 前置条件 | 总经理已经选定了两个城市 |
| 后置条件 | 将两个城市间的距离和价格信息加入数据库或更新数据库中已有的距离与价格信息 |
| City.cityList | 语法 | public ResultMessage cityList(); |
| 前置条件 | 需要获取城市对象 |
| 后置条件 | 查找数据库中所有城市并返回城市对象列表 |

|  |  |
| --- | --- |
| 需要的服务（需接口） | |
| 服务名 | 服务 |
| cityDataSer.add(cityPO po) | 新增城市对象 |
| cityDataSer.revise(CityPO city) | 更新城市对象信息 |
| cityDataSer.reviseCities(CityPO cities) | 更新城市之间距离和价格 |
| cityDataSer.getCities() | 查找并获取所有城市对象列表 |

accountLogic模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Account.newCount | 语法 | public Boolean newCount(CountVO count); |
| 前置条件 | 用户新建一套账 |
| 后置条件 | 将新增的一套账加入数据库，返回加入是否成功 |
| Account.checkInitInfo | 语法 | public ResultMessage checkInitInfo(); |
| 前置条件 | 用户选择查看期初信息 |
| 后置条件 | 返回期初信息 |
| Account.addAccount | 语法 | public boolean addAccount(AccountVO account); |
| 前置条件 | 用户输入了新账户信息 |
| 后置条件 | 将新增的账户信息加入数据库，返回加入结果 |
| Account.deleteAccount | 语法 | public boolean deleteAccount(String id) |
| 前置条件 | 用户选择一个账户进行删除 |
| 后置条件 | 根据账户名在数据库中删除该账户信息，返回删除结果 |
| Account.reviseAccount | 语法 | public boolean reviseAccount(AccountVO account) |
| 前置条件 | 修改了账户信息并确认 |
| 后置条件 | 更新数据库中该账户的信息，返回更新结果 |
| Account.accountList | 语法 | public ResultMessage accountList() |
| 前置条件 | 用户选择查看账户列表 |
| 后置条件 | 返回数据库中所有账户的列表 |
| Account.checkAccount | 语法 | public ResultMessage checkAccount(String id) |
| 前置条件 | 用户选择一个账户进行查看 |
| 后置条件 | 返回该账户信息 |
| Account.searchAccount | 语法 | public ResultMessage searchAccount(String keywords) |
| 前置条件 | 用户检索账户并输入关键字 |
| 后置条件 | 根据该关键字查找符合条件的账户并返回 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| accountDataSer.findAccount(String keywords) | 在数据库中查找含有关键词的账户，返回数据 | |
| accountDataSer.addAccount(AccountPO  po) | 在数据库中新增一个账户 | |
| accountDataSer.deleteAccount(String id) | 在数据库中删除一个账户 | |
| accountDataSer.reviseAccount(AccountPO po) | 在数据库中修改一个账户的信息 | |
| accountDataSer.checkAccount(String id) | 在数据库中找到一个账户的信息，返回数据 | |
| DatabaseFactory.getAccountbase() | 得到account数据库的服务的引用 | |
| accountDataSer.addCount(countPO po) | 在数据库中新增一套账 | |
| accountDataSer.findInitInfo(String id) | 根据id在数据库中查找期初信息，返回数据 | |
| accountDataSer.addCount(CountPO  po) | 在数据库中新增一个账户 | |
| accountDataSer.getCount(String id) | 在数据库中获取并返回一个账户信息 | |
| accountDataSer.getWorkers(String id) | 在数据库中获取并返回一个人员信息 | |
| accountDataSer.getVans(String id) | 在数据库中获取并返回一个货车信息 | |
| accountDataSer.getStock(String id) | 在数据库中获取并返回一个货车信息 | |

stockLogic模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Stock.setWarning | 语法 | public ResultMessage setWarning(int w,Part part, String id) |
| 前置条件 | 用户输入了警戒线比例，选择了区 |
| 后置条件 | 修改数据库中的警戒线比例，并返回修改是否成功 |
| Stock.checkWarning | 语法 | public boolean checkWarning(String id) |
| 前置条件 | 用户新建了入库单 |
| 后置条件 | 对比数据库中库存数/库存容量和警戒线比例，返回是否报警 |
| Stock.checkNow | 语法 | public ResultMessage checkNow(String id) |
| 前置条件 | 用户选择了库存盘点 |
| 后置条件 | 从数据库中获取当前库存情况，返回结果 |
| Stock.outputExcel | 语法 | public boolean outputExcel(String location,StockVO stock) |
| 前置条件 | 用户选择导出库存情况和导出位置 |
| 后置条件 | 将生成的Excel导出到相应位置，返回导出是否成功 |
| Stock.shelfList | 语法 | public ResultMessage shelfList(String id) |
| 前置条件 | 用户选择了库存分区 |
| 后置条件 | 读取数据库中的架信息，返回架信息 |
| Stock.addShelf | 语法 | public boolean addShelf(ShelfVO shelf) |
| 前置条件 | 用户输入了新建的架的所有信息，确认新建 |
| 后置条件 | 在数据库中添加新建的架的信息，返回是否新建成功 |
| Stock.deleteShelf | 语法 | public boolean deleteShelf(String id) |
| 前置条件 | 用户选择了一个架，选择删除 |
| 后置条件 | 在数据库中去除相应架的信息，返回是否删除成功 |
| Stock.reviseShelf | 语法 | public boolean reviseShelf(ShelfVO shelf) |
| 前置条件 | 用户修改了架信息，并确认 |
| 后置条件 | 在数据库中修改相应架的信息，返回是否修改成功 |
| Stock.checkShelf | 语法 | public ResultMessage checkShelf(String id) |
| 前置条件 | 用户选择了架，选择查看 |
| 后置条件 | 在数据库中找到相应架的信息，返回信息 |
| Stock.searchShelf | 语法 | public ResultMessage searchShelf(String keyword,String id) |
| 前置条件 | 用户输入了关键词，选择搜索 |
| 后置条件 | 在数据库中找到含有关键词的架的信息，返回信息 |
| Stock.getWarning | 语法 | public ResultMessage getWarning(String id) |
| 前置条件 | 用户选择了库存报警 |
| 后置条件 | 在数据库中获得各区警戒线比例，返回值 |
| Stock.setPoint | 语法 | public boolean setPoint(String id, Date date) |
| 前置条件 | 用户确认盘点 |
| 后置条件 | 在数据库中增加一个盘点截止点，返回是否成功 |
| Stock.checkStockIn | 语法 | public ResultMessage checkStockIn(Date start,Date end,String id) |
| 前置条件 | 用户输入了开始和结束时间 |
| 后置条件 | 在数据库中按照开始时间和结束时间，查找并返回所有入库信息 |
| Stock.checkStockOut | 语法 | public ResultMessage checkStockOut(Date start,Date end,String id) |
| 前置条件 | 用户输入开始时间和结束时间并确认 |
| 后置条件 | 在数据库中根据开始和结束时间，查找并返回所有出库信息 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| DatabaseFactory.getStockData() | 得到Stock数据库的服务的引用 | |
| stockDataSer.getWarning(String id) | 从数据库中获得ID相应仓库的警戒线比例 | |
| stockDataSer.setWarning(int w,Part part,String id) | 在数据库中设置ID相应仓库和相应区的警戒线比例 | |
| stockDataSer.getIn(Date start, Date end, String id) | 在数据库中获得ID相应仓库开始时间和结束时间之间的入库单 | |
| stockDataSer.getOut(Date start, Date end, String id) | 在数据库中获得ID相应仓库开始时间和结束时间之间的出库单 | |
| stockDataSer.getStock(String id) | 在数据库中获得ID相应仓库当前库存的情况 | |
| stockDataSer.getShelves(String id) | 在数据库中获得ID相应仓库当前架的情况 | |
| stockDataSer.addShelf(ShelfPO shelf) | 在数据库中ID相应仓库添加一个架的信息 | |
| stockDataSer.deleteShelf(String id) | 在数据库中ID相应仓库删除一个架的信息 | |
| stockDataSer.reviseShelf(ShelfPO po) | 在数据库中ID相应仓库修改一个架的信息 | |
| stockDataSer.getShelf(String id) | 在数据库中获得ID相应仓库一个架的信息 | |
| stockDataSer.searchShelf(String keyword,String id) | 在数据库中获得含有关键词的架 | |
| stockDataSer.setPoint(String id, Date date) | 在数据库中增加一个盘点截止点 | |

checkLogic模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Check.seeDiary | 语法 | public ResultMessage seeDiary(Date start, Date end) |
| 前置条件 | 输入的日期符合输入规则 |
| 后置条件 | 根据输入的开始和结束日期，返回相应的日志记录 |
| Check.seeStatementSheet | 语法 | public ResultMessage seeStatementSheet(Date start, Date end，String id) |
| 前置条件 | 输入的日期符合输入规则 |
| 后置条件 | 根据输入的开始和结束日期，返回由入款单和收款单信息组成的经营情况表 |
| Check.seeCollectRecord | 语法 | public ResultMessage seeCollectRecord(Date date, String id) |
| 前置条件 | 输入了日期，选择了营业厅 |
| 后置条件 | 根据输入的日期，返回当日的收款记录 |
| Check.total | 语法 | public ResultMessage total(ArrayList<IncomeVO> income) |
| 前置条件 | 用户选择合计收款 |
| 后置条件 | 返回合计的收款总额 |
| Check.getHall | 语法 | public ResultMessage getHall() |
| 前置条件 | 用户选择查看经营情况表 |
| 后置条件 | 返回所有营业厅的名称 |
| Check.outputExcel | 语法 | public boolean outputExcel(ArrayList<IncomeVo> income, String location) |
| 前置条件 | 用户选择导出经营情况表 |
| 后置条件 | 导出经营情况表到相应地址 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| checkDataSer.seeDiary(Date start, Date end) | 根据开始和结束日期从数据库中获得日志记录，返回数据 | |
| checkDataSer.findCollects(Date start, Date end) | 根据开始和结束日期从数据库中获得收款单记录，返回数据 | |
| checkDataSer.getCollectRec (Date date, String id) | 根据给的日期和营业厅编号从数据库中获得当日该营业厅收款记录，返回数据 | |
| checkDataSer.getHall() | 返回所有营业厅的编号和名称 | |
| DatabaseFactory.getDiaryDatabase | 得到diary数据库的服务的引用 | |
| DatabaseFactory.getRecordDatabase | 得到record数据库的服务的引用 | |

orderApprove模块接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| orderApprove.askExamine | 语法 | public ResultMessage askExamine() |
| 前置条件 | 用户请求审批单据 |
| 后置条件 | 返回待审批单据列表 |
| orderApprove.examine | 语法 | public boolean examine(boolean approve,ArrayList<String> id) |
| 前置条件 | 用户决定单据是否通过审批 |
| 后置条件 | 返回审批结果 |
| orderApprove.chooseOrder | 语法 | public ResultMessage chooseOrder(String id) |
| 前置条件 | 用户选择一个订单，查看详情 |
| 后置条件 | 系统返回订单详情 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| orderApproveDataSer.update(  ArrayList<PO> po) | 更新订单审批状态 | |
| orderApproveDataSer.getOrder(String id) | 从数据库中获取相应单据的信息，返回。 | |
| orderApproveDataSer.getExamine(String id) | 从数据库中获取相应单据的审批信息，返回。 | |
| DatabaseFactory.getOrderDatabase | 得到orders数据库的服务的引用 | |

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，包括对持久化数据的增、删、改、查。User逻辑需要的服务由UserDataSer接口提供。由于持久化数据的保存可能存在多种形式：Txt文件、序列化文件、数据库等，所以抽象了数据服务。数据层模块的描述具体如图13所示。



图13 数据层模块的描述

5.4.1 数据层模块的职责

数据层模块的职责如表所示。

数据层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| UserDataSer | 持久化数据库的接口，提供用户数据增删改查等服务 |
| UserDataSerImpl | 基于Mysql数据库的持久化数据库的接口，实现UserDataSer的服务 |
| TransitDataSer | 持久化数据库的接口，提供更新物流信息等服务 |
| TransitDataSerImpl | 基于Mysql数据库的持久化数据库的接口，实现TransitDataSerImpl的服务 |
| OrderNewDataSer | 持久化数据库的接口，提供在数据库新建单据等服务 |
| OrderNewDataSerImpl | 基于Mysql数据库的持久化数据库接口，实现OrderNewDataSer的服务 |
| OrderApproveDataSer | 持久化数据库的接口，提供导入单据信息和保持单据审批状态等服务 |
| OrderApproveDataSerImpl | 基于Mysql数据库的持久化数据接口，实现OrderApproveDataSer的服务 |
| AccountDataSer | 持久化数据库接口，提供账户数据的增删改查和期初建账服务 |
| AccountDataSerImpl | 基于Mysql数据库的持久化数据接口，实现AccountDataSer的服务 |
| CityDataSer | 持久化数据库接口，提供城市数据的增删改查服务 |
| CityDataSerImpl | 基于Mysql数据库的持久化数据接口，实现CityDataSer的服务 |
| WorkOrgManSer | 持久化数据库接口，提供人员机构数据的增删改查服务 |
| WorkOrgManSerImpl | 基于Mysql数据库的持久化数据接口，实现WorkOrgManSer的服务 |
| StockDataSer | 持久化数据库接口，提供库存数据的增删改查服务 |
| StockDataSerImpl | 基于Mysql数据库的持久化数据接口，实现StockDataSer的服务 |
| CheckDataSer | 持久化数据库接口，提供统计报表和日志记录的载入服务 |
| CheckDataSerImpl | 基于Mysql数据库的持久化数据接口，实现CheckDataSer的服务 |

5.4.2 数据层模块的接口规范

数据层模块的接口如表所示。

数据层模块的接口规范

UserDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| UserDataSer.findUser | 语法 | public UserPO 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UserPO结果 |
| UserDataSer.addUser | 语法 | public boolean addUser(UserPO user) throws RemoteException |
| 前置条件 | 同样ID的po在数据库中不存在 |
| 后置条件 | 在数据库中增加一个PO记录 |
| UserDataSer.deleteUser | 语法 | public boolean deleteUser(String id) throws RemoteException |
| 前置条件 | 数据库中存在同样ID的用户 |
| 后置条件 | 在数据库中删除一个PO记录 |
| UserDataSer.reviseUser | 语法 | public boolean reviseUser(UserPO user) throws RemoteException |
| 前置条件 | 在数据库中存在同样ID的用户 |
| 后置条件 | 在数据库中修改该用户的数据 |
| UserDataSer.list | 语法 | public ArrayList<UserPO> list() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中用户数据的列表 |

TransitInfoDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| TransitInfoDataSer.find | 语法 | public PO find(String barcode) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按barcode进行查找返回相应的PO结果 |
| TransitInfoDataSer.findTransit | 语法 | public ArrayList<String> findTransit(String barcode) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按barcode进行查找返回相应订单的物流轨迹和货运信息 |

CityDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CityDataSer.add | 语法 | public boolean add(CityPO city) throws RemoteException |
| 前置条件 | 该city在数据库中不存在 |
| 后置条件 | 在数据库中增加一个city记录 |
| CityDataSer.revise | 语法 | Public Boolean revise(CityPO city) throws RemoteException |
| 前置条件 | 该city在数据库中已存在 |
| 后置条件 | 在数据库中更新该city |
| CityDataSer.reviseCities | 语法 | Public Boolean reviseCities(CityPO cities) throws RemoteException |
| 前置条件 | 该city在数据库中已存在 |
| 后置条件 | 改变城市之间距离和价格 |
| CityDataSer.getCities | 语法 | Public ArrayList<CityPO> getCities() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回城市对象列表 |

AccountDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AccountDataSer.findAccount | 语法 | public ArrayList<AccountPO> findAccount(String keyword) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按关键字查找返回符合条件的账户列表 |
| AccountDataSer.addAccount | 语法 | public boolean addAccount(AccountPO account) throws RemoteException |
| 前置条件 | 同样账户名的账户在数据库中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| AccountDataSer.deleteAccount | 语法 | public boolean deleteAccount(String id) throws RemoteException |
| 前置条件 | 该id对应的账户在数据库中存在 |
| 后置条件 | 在数据库中删除一个po记录 |
| AccountDataSer.reviseAccount | 语法 | public boolean reviseAccount(AccountPO po) throws RemoteException |
| 前置条件 | 该账户在数据库中存在 |
| 后置条件 | 在数据库中更新该账户信息 |
| AccountDataSer.checkAccount | 语法 | public AccountPO checkAccount(String id) throws RemoteException |
| 前置条件 | 该id对应的账户在数据库中存在 |
| 后置条件 | 返回该id对应的账户信息 |
| AccountDataSer.findInitInfo | 语法 | public AcountPO findInitInfo() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回该id对应的期初信息 |
| AccountDataSer.addCount | 语法 | public boolean addCount(CountPO count) throws RemoteException |
| 前置条件 | 同样账户名的账户在数据库中不存在 |
| 后置条件 | 在数据库中增加一个po记录 |
| AccountDataSer.findOrg | 语法 | public ArrayList<OrgPO> findOrg() throws RemoteException |
| 前置条件 |  |
| 后置条件 | 在数据库中获取并返回机构信息 |
| AccountDataSer.getWorkers | 语法 | Public ArrayList<WorkPO> getWorkers(String id)throws RemoteException |
| 前置条件 | 账户名的账户在数据库中存在 |
| 后置条件 | 在数据库中获取并返回人员信息 |
| AccountDataSer.getVans | 语法 | public ArrayList<VanPO> getVans(String id) throws RemoteException |
| 前置条件 |  |
| 后置条件 | 在数据库中获取并返回货车信息 |
| AccountDataSer.getStock | 语法 | Public ArrayList<StockPO> getStock(String id)throws RemoteException |
| 前置条件 |  |
| 后置条件 | 在数据库中获取并返回库存信息 |
| AccountDataSer.getAccount | 语法 | public ArrayList<AccountPO> getAccount(String id) throws RemoteException |
| 前置条件 |  |
| 后置条件 | 在数据库中获取并返回账户信息 |

OrderNewDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderNewDataSer.CenterLoadingOrderDSer.create | 语法 | public boolean create(CenterLoadingPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个CenterLoadingOrder记录 |
| OrderNewDataSer. CenterLoadingOrderDSer.getHalls | 语法 | public ArrayList<String> getHalls () throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中获取并返回营业厅信息 |
| OrderNewDataSer.ArriveOrderDSer.create | 语法 | public boolean create (ArrivePO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个ArriveOrder记录 |
| OrderNewDataSer.ArriveOrderDSer.getCities | 语法 | public ArrayList<String> getCities () throws RemoteException |
| 前置条件 | 新建单据逻辑请求城市信息 |
| 后置条件 | 在数据库中获取并返回城市信息 |
| OrderNewDataSer.ExpressOrderDSer.create | 语法 | public boolean create (ExpressPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer.HallLoadingOrderDSer.create | 语法 | public boolean create (HallLoadingPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer. HallLoadingOrderDSer.getOrgs | 语法 | public ArrayList<String> getOrgs () throws RemoteException |
| 前置条件 | 新建单据逻辑请求机构信息 |
| 后置条件 | 在数据库中获取并返回机构信息 |
| OrderNewDataSer. HallLoadingOrderDSer.getVans | 语法 | public ArrayList<String> getVans() throws RemoteException |
| 前置条件 | 新建单据逻辑请求货车信息 |
| 后置条件 | 在数据库中获取并返回货车信息 |
| OrderNewDataSer.IncomeOrderDSer.create | 语法 | public boolean create (IncomePO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer.IncomeOrderDSer.getCouriers | 语法 | public ArrayList<String> getCouriers () throws RemoteException |
| 前置条件 | 新建单据逻辑请求快递员信息 |
| 后置条件 | 在数据库中获取并返回快递员信息 |
| OrderNewDataSer.PaymentOrderDSer.create | 语法 | public boolean create (PaymentPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer.ReceiveOrderDSer.create | 语法 | public boolean create (ReceivePO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer.ReceiveOrderDSer.getExpress | 语法 | public ExpressPO getExpress (String id) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中获取并返回物流信息 |
| OrderNewDataSer.RecipientOrderDSer.create | 语法 | public boolean create (RecipientPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer. RecipientOrderDSer.getCities | 语法 | public ArrayList<String> getCities () throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中获取并返回城市信息 |
| OrderNewDataSer.SendOrderDSer.create | 语法 | public boolean create (SendPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer.SendOrderDSer.getCouriers | 语法 | public ArrayList<String> getCouriers () throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中获取并返回快递员信息 |
| OrderNewDataSer.StockInOrderDSer.create | 语法 | public boolean create (StockInPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer.StockInOrderDSer.getCities | 语法 | public ArrayList<String> getCities () throws RemoteException |
| 前置条件 | 单据新建逻辑请求城市信息 |
| 后置条件 | 在数据库中获取并返回城市信息 |
| OrderNewDataSer.StockOutOrderDSer.create | 语法 | public boolean create (StockOutPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer.StockOutOrderDSer.getCities | 语法 | public ArrayList<String> getCities () throws RemoteException |
| 前置条件 | 单据新建逻辑请求城市信息 |
| 后置条件 | 在数据库中获取并返回城市信息 |
| OrderNewDataSer.TransitOrderDSer.create | 语法 | public boolean create (TransitPO order) throws RemoteException |
| 前置条件 | 单据已生成，且信息完整 |
| 后置条件 | 在数据库中新增一个Order记录 |
| OrderNewDataSer.TransitOrderDSer.getCities | 语法 | public ArrayList<String> getCities () throws RemoteException |
| 前置条件 | 单据新建逻辑请求城市信息 |
| 后置条件 | 在数据库中获取并返回城市信息 |

workOrgManDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| WorkOrgManDataSer.addDriver | 语法 | public boolean addDriver (DriverPO po) throws RemoteException |
| 前置条件 | 新人员信息已输入完全并确认 |
| 后置条件 | 在数据库中增加一个PO记录 |
| WorkOrgManDataSer.deleteDriver | 语法 | public boolean deleteDriver (String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的司机 |
| 后置条件 | 在数据库中删除相应的司机 |
| WorkOrgManDataSer.reviseDriver | 语法 | public boolean reviseDriver(DriverPO) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的司机 |
| 后置条件 | 在数据库中更新相应ID司机的信息 |
| WorkOrgManDataSer.checkDriver | 语法 | public WorkPO checkDriver(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的司机 |
| 后置条件 | 返回相应司机的信息 |
| WorkOrgManDataSer.findOrg | 语法 | public ArrayList<OrgPO> findOrg(String keywords) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按关键词查找并返回相应的OrgPO结果 |
| WorkOrgManDataSer.addOrg | 语法 | public boolean addOrg(OrgPO po) throws RemoteException |
| 前置条件 | 新机构信息已输入完全并确认 |
| 后置条件 | 在数据库中增加一个PO记录 |
| WorkOrgManDataSer.deleteOrg | 语法 | public boolean deleteOrg(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的机构 |
| 后置条件 | 在数据库中删除相应机构的信息 |
| WorkOrgManDataSer.reviseOrg | 语法 | public boolean reviseOrg(OrgPO) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的机构 |
| 后置条件 | 在数据库中更新相应机构的信息 |
| WorkOrgManDataSer.checkOrg | 语法 | public OrgPO checkOrg(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的机构 |
| 后置条件 | 返回该机构的信息 |
| WorkOrgManDataSer.findVan | 语法 | public ArrayList<VanPO> findVan(String keyword) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按关键词查找并返回相应的VanPO结果 |
| WorkOrgManDataSer.addVan | 语法 | public boolean addVan(VanPO Van) throws RemoteException |
| 前置条件 | 新货车信息已输入完全并确认 |
| 后置条件 | 在数据库中增加一个PO记录 |
| WorkOrgManDataSer.deleteVan | 语法 | public boolean deleteVan(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的货车 |
| 后置条件 | 在数据库中删除相应货车的信息 |
| WorkOrgManDataSer.reviseVan | 语法 | public boolean reviseOrg(VanPO Van) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的货车 |
| 后置条件 | 在数据库中更新相应机构的信息 |
| WorkOrgManDataSer.checkVan | 语法 | public VanPO checkVan(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的货车 |
| 后置条件 | 返回该货车的信息 |
| WorkOrgManDataSer.getWage | 语法 | public WagePO getWage(String id) throws RemoteException |
| 前置条件 | 信息已输入完全并确认 |
| 后置条件 | 从数据库中获取相应人的薪水 |
| WorkOrgManDataSer.reviseWage | 语法 | public boolean reviseWage (WagePO) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的机构 |
| 后置条件 | 在数据库中更新相应的薪水信息 |
| WorkOrgManDataSer.findWork | 语法 | public ArrayList<WorkPO> findWork(String keyword) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 按关键词查找并返回相应的WorkPO结果 |
| WorkOrgManDataSer.addWork | 语法 | public boolean addWork(WorkPO work) throws RemoteException |
| 前置条件 | 新工人信息已输入完全并确认 |
| 后置条件 | 在数据库中增加一个PO记录 |
| WorkOrgManDataSer.deleteWork | 语法 | public boolean deleteWork(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的工人 |
| 后置条件 | 在数据库中删除相应工人的信息 |
| WorkOrgManDataSer.reviseWork | 语法 | public boolean reviseWork(WorkPO work) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的工人 |
| 后置条件 | 在数据库中更新相应工人的信息 |
| WorkOrgManDataSer.checkWork | 语法 | public WorkPO checkWork(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的工人 |
| 后置条件 | 返回该工人的信息 |

StockDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| StockDataSer.getWarning | 语法 | public ArrayList<Integer> getWarning(String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中各个区的警戒线比例 |
| StockDataSer.setWarning | 语法 | public boolean setWarning(int w,Part part,String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 设置相应区的警戒线比例为w |
| StockDataSer.getIn | 语法 | public ArrayList<StockInPO> getIn(Date start, Date end,String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中开始时间和结束时间之间的入库单列表 |
| StockDataSer.getOut | 语法 | public ArrayList<StockOutPO> getOut(Date start, Date end,String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中开始时间和结束时间之间的出库单列表 |
| StockDataSer.getStock | 语法 | public StockPO getStock(String id)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中库存的情况，包括各个架上的货物 |
| StockDataSer.getShelves | 语法 | public ArrayList<ShelfPO> getShelves(String id,String keyword)throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回数据库中所有架的信息 |
| StockDataSer.addShelf | 语法 | public boolean addShelf(ShelfPO shelf) throws RemoteException |
| 前置条件 | 数据库中不含有同样ID的架 |
| 后置条件 | 在数据库中增加一个架的信息 |
| StockDataSer.deleteShelf | 语法 | public boolean deleteShelf(String id) throws RemoteException |
| 前置条件 | 数据库中含有同样ID的架 |
| 后置条件 | 在数据库中删除相应的架 |
| StockDataSer.reviseShelf | 语法 | public boolean reviseShelf(ShelfPO shelf) throws RemoteException |
| 前置条件 | 数据库中含有同样ID的架 |
| 后置条件 | 在数据库中修改相应的架 |
| StockDataSer.getShelf | 语法 | public ShelfPO getShelf(String id) throws RemoteException |
| 前置条件 | 数据库中含有同样ID的架 |
| 后置条件 | 返回数据库中相应架的信息 |
| StockDataSer.setPoint | 语法 | public boolean setPoint(String id, Date date) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 在数据库中增加一个盘点截止点 |

OrderApproveDataSer模块接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderApproveDataSer.update | 语法 | public boolean update(ArrayList<PO> po)throws RemoteException |
| 前置条件 | Order存在于数据库中 |
| 后置条件 | 在数据库中更新订单审批状态 |
| OrderApproveDataSer.getOrder | 语法 | public PO getOrder(String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 从数据库获得并返回相应的单据信息 |
| OrderApproveDataSer.getOrders | 语法 | public ArrayList<PO> getOrder(String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 从数据库获得并返回相应的单据信息 |
| OrderApproveDataSer.getExamine | 语法 | public ArrayList<PO> getExamine (String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 从数据库获得并返回相应的单据审批信息 |

CheckDataSer模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CheckDataSer.seeDiary | 语法 | public ArrayList<DiaryPO> seeDiary(Date start,Date end) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 根据输入的开始和结束日期从数据库中获得日志记录，返回相应的数据 |
| CheckDataSer.findCollect | 语法 | public ArrayList<IncomePO> findCollect(Date start,Date end) throw RemoteException |
| 前置条件 | 无 |
| 后置条件 | 根据输入的开始和结束日期从数据库收款单记录，返回相应的数据 |
| CheckDataSer.getCollectRec | 语法 | public ArrayList<IncomePO> getCollectRec(Date date,String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 根据时间和营业厅名，从数据库中获取并返回该营业厅的所有收款单记录 |
| CheckDataSer.getHall() | 语法 | public ArrayList<OrgPO> getHall() throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 从数据库中获取并返回所有的营业厅编号和名称 |

1. 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。

· UserPO类包含用户的用户名，密码，权限等级属性

· AccountPO类包含账户名称与余额属性

· CitiesPO类包含城市间距离和价格属性

· CityPO类包含区号，名称属性。

· CountPO类包含编号，机构，人员，车辆，库存，账户属性。

· DiaryPO类包含操作日志的时间和操作方式属性

· DriverPO类包含司机的编号，姓名，出生日期，身份证号，手机，车辆单位，性别，行驶证期限属性。

· OrgPO类包含机构的种类，地址，名称属性

· ShelfPO类包含快递编号，入库日期，目的地，区号，排号，架号，位号属性。

· StockPO类包含货物属性。

· VanPO类包含车辆代号，发动机号，车辆号，底盘号，购买时间，服役时间，车辆图片属性

· WagePO类包含薪水发放方式，发放金额，提成份额属性。

· WorkPO类包含工作人员的姓名，性别，职位，所属机构属性。

· ArrivePO类包含到达日期，中转单编号，出发地，货物到达状态属性。

· CenterLoadingPO类包含装车日期，本中转中心中转单编号，出发地，到达地，货柜号，监装员，本次装箱所有托运单号，运费属性。

· SendPO类包含到达日期，托运订单条形码号，派送员属性。

· HallLoadingPO类包含装车日期、本中转中心汽运编号、到达地（营业厅）、车辆代号、监装员、押运员、本次装箱所有订单条形码号、运费属性。

· IncomePO类包含收款日期，收款金额，收款快递员，对应的所有快递订单条形码号。

· PaymentPO类包含付款日期，金额，付款人，付款账号，条目，备注属性。

· ReceivePO类包含收件编号，收件人，收件日期属性。

· RecipientPO类包含中转中心编号，到达日期，中转单编号，出发地，货物到达状态属性。

· ExpressPO类包含寄件人姓名、住址、单位、电话、手机，收件人姓名、住址、单位、电话、手机，托运货物信息，包装费，总费用，订单条形码号，种类属性。

· StockInPO类包含快递编号、入库日期、目的地、区号、排号、架号、位号属性

· StockOutPO类包含快递编号、出库日期、目的地、装运形式（火车、飞机、汽车）、中转单编号或者汽运编号属性。

· TransitPO类包含运输工具种类，装车日期，本中转中心中转单编号，航班号，出发地，到达地，货柜号，监装员，本次装箱所有托运单号，运费属性。

· PO类包含单据信息和是否审批属性

UserPO定义如下

public class UserPO{

private String Username;

private String Password;

private int lev;

public UserPO(String na,String pw,int lv){

Username=na;

Password=pw;

Lev=lv;

}

public void Setna(String na){  
 Username=na;

}

public void Setpw(String pw){  
 Password=pw;

}

public void Setlv(int lv){  
 Username=na;

}

public String Getna(){  
 return Username;

}

public String Getpw(){  
 return Password;

}

public int Getlv(){  
 return lev;

}

}

6.2数据库表

数据库中包含Account表，Cities表，City表，Count表，Diary表，Driver表，Org表，Shelf表，Stock表，User表，Van表，Wage表，Work表，Arrive表，CenterLoading表，Express表，HallLoading表，Income表，Order表，Payment表，Receive表，Recipient表，Send表，StockIn表，StockOut表，TransitPO表。