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 1](#_Toc433291765)

[1.1 编制目的 1](#_Toc433291766)

[1.2 词汇表 1](#_Toc433291767)

[1.3 参考资料 1](#_Toc433291768)

[2. 产品概述 1](#_Toc433291769)

[3. 逻辑视角 1](#_Toc433291770)

[4. 组合视角 3](#_Toc433291771)

[4.1 开发包图 3](#_Toc433291772)

[5. 接口视角 5](#_Toc433291773)

[5.2 用户界面层的分解 5](#_Toc433291774)

[5.3 业务逻辑层的分解 5](#_Toc433291775)

[5.3.1 业务逻辑层模块的职责 5](#_Toc433291776)

[5.3.2 业务逻辑层模块的接口规范 5](#_Toc433291777)

[5.4 数据层的分解 13](#_Toc433291778)

[5.4.1 数据层模块的职责 13](#_Toc433291779)

[5.4.2 数据层模块的接口规范 13](#_Toc433291780)

[6. 信息视角 18](#_Toc433291781)

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

1. 接口视角

5.2 用户界面层的分解

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所示。

5.3.1 业务逻辑层模块的职责

业务逻辑层模块的职责如下表所示。

|  |  |
| --- | --- |
| 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下诸表所示。

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.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返回订单全部信息 |
| TransitInfo.enterBarcodeCustomer | 语法 | public ResultMessage enterBarcodeCustomer(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返回该订单货运状态和物流轨迹 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| transitInfoDataSer.findOrder(String  barcode) | 根据条形码号获得订单信息 | |
| transitInfoDataSer.findTransit(String barcode) | 根据条形码号获得物流信息 | |
| DatabaseFactory.getOrdersDatabase | 得到orders数据库的服务的引用 | |

orderNewLogic模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderNew.Create | 语法 | public Order Create(Order order) |
| 前置条件 | 单据的各项信息已输入并确认 |
| 后置条件 | 将该新增单据上传至数据库，返回上传状态 |
| 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.add(OrderPO po) | 在数据库中添加OrderPO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 |
| 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 | |

cityLogic模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| City.addCity | 语法 | public ResultMessage addCity(String name); |
| 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.enterDistance | 语法 | public ResultMessage enterDistance(long distance); |
| 前置条件 | 总经理已选定了两个城市，distance符合输入规则 |
| 后置条件 | 将两城市距离信息加入数据库，或更新数据库已有信息，返回添加结果 |
| City.enterPrice | 语法 | public ResultMessage enterPrice(double price); |
| 前置条件 | 总经理已经选定了两个城市，price符合输入规则 |
| 后置条件 | 将两城市价格信息加入数据库或更新数据库已有信息，返回添加结果 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| cityDataSer.add(cityPO po) | 新增城市对象 | |
| cityDataSer.update(cityPO po) | 更新城市对象信息 | |
| DatabaseFactory.getCityDatabase | 得到city数据库的服务的引用 | |

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 ResultMessage 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 ResultMessage 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 ResultMessage 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() |
| 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 | |

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, int 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.checkStock | 语法 | public ResultMessage checkStock(Date start, Date end, String id) |
| 前置条件 | 用户输入了开始时间和结束时间 |
| 后置条件 | 从数据库中获取开始时间和结束时间之间的出入库情况，计算合计，统计数据，返回结果 |
| 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, String id) |
| 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 shelfId, 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, String id) |
| 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 shelfId, 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.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 |
| 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(String id, int w, int part) | 在数据库中设置ID相应仓库和相应区的警戒线比例 | |
| stockDataSer.getInOut(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入库单 | |
| 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 po, String id) | 在数据库中ID相应仓库添加一个架的信息 | |
| stockDataSer.deleteShelf(String shelfId, String id) | 在数据库中ID相应仓库删除一个架的信息 | |
| stockDataSer.reviseShelf(ShelfPO po, String id) | 在数据库中ID相应仓库修改一个架的信息 | |
| stockDataSer.getShelf(String shelfID, String id) | 在数据库中获得ID相应仓库一个架的信息 | |
| 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) |
| 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.getDiary(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.getCollects(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) |
| 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<OrderPO> 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。 | |
| DatabaseFactory.getOrderDatabase | 得到orders数据库的服务的引用 | |

5.4 数据层的分解

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下诸表所示。

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 orderPO 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orderPO结果 |
| TransitInfoDataSer.findTransit | 语法 | public ArrayList<String> 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物流轨迹和货运状态 |

OrderNewDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderNewDataSer.add | 语法 | public boolean add(OrderPO 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 |

workOrgManDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| WorkOrgManDataSer.findWork | 语法 | public ArrayList<workPO> findWork(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WorkPO结果 |
| WorkOrgManDataSer.addWork | 语法 | public boolean addWork(WorkPO 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.deleteWork | 语法 | public boolean deleteWork(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的人员 |
| 后置条件 | 在数据库中删除相应的人员 |
| WorkOrgManDataSer.reviseWork | 语法 | public boolean reviseWork(String id,WorkPO) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的人员 |
| 后置条件 | 在数据库中更新相应ID人员的信息 |
| WorkOrgManDataSer.checkWork | 语法 | public WorkPO checkWork(String id) throws RemoteException |
| 前置条件 | 数据库中存在相应ID的人员 |
| 后置条件 | 返回相应人员的信息 |
| 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(String id,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 |

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.update | 语法 | Public Boolean update(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 |

AccountDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| AccountDataSer.findAccount | 语法 | public ArrayList<AccountPO> findAccount(String keywords) 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 po) 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(String id) throws RemoteException |
| 前置条件 | 无 |
| 后置条件 | 返回该id对应的期初信息 |

StockDataSer的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| StockDataSer.getWarning | 语法 | public ArrayList<Integer> getWarning() 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 id,int w) 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.getInOut | 语法 | public ArrayList<Order> getInOut(Date start, Date end) 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收款单和付款单列表 |
| StockDataSer.getStock | 语法 | public StockPO getStock()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()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 |

CheckDataSer模块的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| CheckDataSer.findDiary | 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<CollectPO> 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<CollectPO> 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<HallPO> 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 |

OrderApproveDataSer模块接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| OrderApproveDataSer.update | 语法 | public boolean update(ArrayList<OrderPO> 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 OrderPO 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 |

1. 信息视角