快递管理系统 MSE(Management System of Expressage)

软件详细设描述文档

V1.0

完整版

一组

成员：吕德超、李昊朔、阙俊杰、罗铉斌

2015-11-5

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **日期** | **变更原因** | **版本号** |
| 全体人员 | 2015-10-21 | 最初草稿（不完整） | V1.0 草稿 |

# 引言

## 编制目的

本报告详细完成对快递管理系统的详细设计，达到指导后续软件构造的目的，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。

## 词汇表

|  |  |  |
| --- | --- | --- |
| 词汇名称 | 词汇含义 | 备注 |
| MSE | 快递管理系统 |  |
| checklist | 订单查询 | 订单查询 |
| listin | 单据输入 | 订单输入、收件人信息输入 |
| transfer | 中转接收 | 接收与派件、中转接收、录入中转单 |
| load | 装运管理 | 车辆装车管理、装运管理 |
| vehicle | 车辆管理 | 司机信息管理、车辆信息管理 |
| finance | 财务管理 | 收款单管理，结算管理，成本管理，查看成本收益表，统计报表，查看经营情况 |
| store | 仓库管理 | 入库管理，出库管理，库存管理 |
| bill | 期初建账 | 期初建账 |
| account | 账户管理 | 账户管理，帐户查询 |
| staff | 员工管理 | 人员机构管理，用户信息管理 |
| policy | 策略管理 | 策略管理 |
| approve | 审批单据 | 审批单据 |
| user | 用户 |  |
| order | 订单 |  |
| receiver | 收件人信息 |  |
| loadorder | 装车单 |  |
| driver | 司机信息 |  |
| arriveorder | 到达单 | 中转中心到达单 |
| receiveorder | 接收单 | 营业厅接收单 |
| sendorder | 派件单 |  |
| payeeorder | 收款单 |  |
| changeorder | 中转单 |  |
| storeinorder | 入库单 |  |
| storeoutorder | 出库单 |  |
| payorder | 付款单 |  |
| startinfo | 期初信息 |  |
| salary | 薪水信息 |  |
| constant | 常量信息 |  |

## 参考资料

1. IEEE标准
2. 快递管理系统MSE用例文档
3. 快递管理系统MSE结构逻辑设计文档

# 产品概述

参考快递管理系统用例文档和快递管理系统软件需求规格说明文档中对产品的概括描述。

# 体系结构设计概述

参考快递管理系统概要设计文档中对体系结构文档的概述。

# 结构视角

## 业务逻辑层的分解

业务逻辑层的开发包图参见软件体系结构文档图3

### Userbl模块

#### 4.1.1.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Checklistbl模块

#### 4.1.2.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Listinbl模块

#### 4.1.3.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.3.2具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Transferbl模块

#### 4.1.4.1中转接收

###### 模块描述

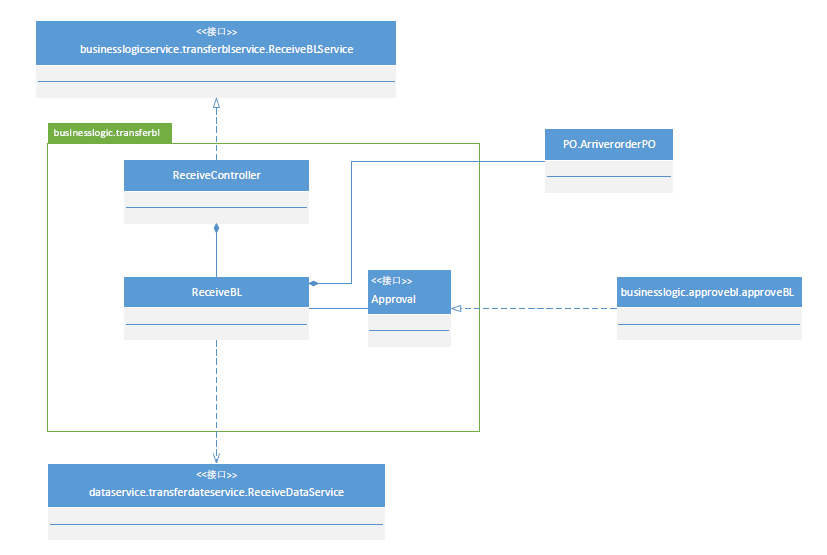
中转接收模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

中转接收模块的职责及接口参见软件系统结构描述文档。

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，我们会添加businesslogicservice.transferblservice.ReceiveBLService接口。业务逻辑层和数据层之间添加dataservice.transferdateservice. ReceiveDataService接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了ReceiveController，这样ReceiveController会将中转接收的业务逻辑处理委托给ReceiveBL对象。ArriverorderPO是作为到达单的持久化对象被添加到设计模型中去的。Approval是为了将单据提交给总经理审批而产生的接口。

中转接收模块的设计如图：



中转接收模块各个类的职责如表

|  |  |
| --- | --- |
| 模块 | 职责 |
| ReceiveController | 负责实现中转接收界面所需要的服务。 |
| ReceiveBL | 具有新建到达单、设置消息提醒、查看消息提醒、存储新建的到达单等功能，能够帮助完成中转接收界面所需要的服务。 |

###### 模块内部类的接口规范

**ReceiveController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Receive.build | 语法 | public void build(LoadorderVO lv,ArriveState as) |
| 前置条件 | 已创建一个ReceiveBL领域对象，且输入符合输入规范 |
| 后置条件 | 调用ReceiveBL领域的build方法 |
| Receive.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 已创建一个ReceiveBL领域对象 |
| 后置条件 | 调用ReceiveBL领域的checkRemind方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Receive.build | 新建一张到达单 | |
| Receive.checkRemind | 查看消息提醒 | |

**ReceiveBL的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Receive.build | 语法 | public void build(LoadorderVO lv,ArriveState as) |
| 前置条件 | 启动一个新建到达单回合 |
| 后置条件 | 系统将新建的到达单提交给总经理审批 |
| Receive.setRemind | 语法 | public void setRemind(LoadorderVO lv) |
| 前置条件 | 有货物到达中转中心 |
| 后置条件 | 增加一条提醒消息 |
| Receive.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 启动一个新建到达单回合 |
| 后置条件 | 删除该条提醒消息 |
| Receive.saveOrder | 语法 | public void save(ArriverorderVO av) |
| 前置条件 | 到达单已通过总经理审核 |
| 后置条件 | 将到达单存储 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Approval.checkOrder(PO) | 提交给总经理审核单据 | |
| ReceiveDataService.insert(ArriverorderPO av) | 插入一个持久化对象ArriverorderPO | |

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.4.2录入中转单

###### 模块描述

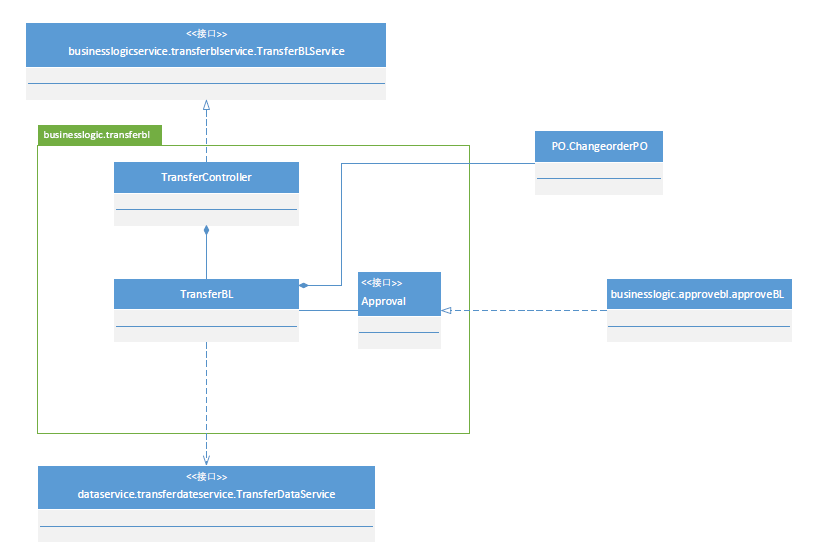
录入中转单模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

录入中转单模块的职责及接口参见软件系统结构描述文档。

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，我们会添加businesslogicservice.transferblservice.TransferBLService接口。业务逻辑层和数据层之间添加dataservice.transferdateservice. TransferDataService接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了TransferController，这样TransferController会将录入中转单的业务逻辑处理委托给TransferBL对象。ChangeorderPO是作为中转单的持久化对象被添加到设计模型中去的。Approval是为了将单据提交给总经理审批而产生的接口。

录入中转单模块的设计如图：



录入中转单模块各个类的职责如表

|  |  |
| --- | --- |
| 模块 | 职责 |
| TransferController | 负责实现录入中转单界面所需要的服务。 |
| TransferBL | 具有新建中转单、设置消息提醒、查看消息提醒、查看库存信息、存储新建的中转单等功能，能够帮助完成录入中转单界面所需要的服务。 |

###### 模块内部类的接口规范

**TransferController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Transfer.build | 语法 | public void build(int[][][][] location,String car\_number,String monitor) |
| 前置条件 | 已创建一个TransferBL领域对象，且输入信息符合输入规范 |
| 后置条件 | 调用TransferBL领域的build方法 |
| Transfer.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 已创建一个TransferBL领域对象 |
| 后置条件 | 调用TransferBL领域的checkRemind方法 |
| Transfer.checkInventory | 语法 | public ResultMessage checkInventory() |
| 前置条件 | 已创建一个TransferBL领域对象 |
| 后置条件 | 调用TransferBL领域的checkInventory方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Transfer.build | 新建一张中转单 | |
| Transfer.checkRemind | 查看消息提醒 | |
| Transfer.checkInventory | 查看此时中转中心仓库库存信息 | |

**TransferBL的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Transfer.build | 语法 | public void build(int[][][][] location,String car\_number,String monitor) |
| 前置条件 | 启动一个录入中转单回合 |
| 后置条件 | 系统将中转单提交总经理审批 |
| Transfer.setRemind | 语法 | public void setRemind() |
| 前置条件 | 有库区库存值达到提醒值 |
| 后置条件 | 增加一条提示消息 |
| Transfer.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 启动一个录入中转单回合 |
| 后置条件 | 删除该条消息 |
| Transfer.checkInventory | 语法 | public ResultMessage checkInventory() |
| 前置条件 | 启动一个录入中转单回合 |
| 后置条件 | 返回各个库区的库存信息 |
| Transfer.saveOrder | 语法 | public void save(ChangeorderVO cv) |
| 前置条件 | 中转单通过总经理审批 |
| 后置条件 | 将中转单存储 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Approval.checkOrder(PO) | 提交给总经理审核单据 | |
| Inventory\_management.checkInventory\_now | 查看此时中转中心仓库库存信息 | |
| TransferDataService.insert(ChangeorderPO cp) | 插入一个持久化对象ChangeorderPO | |

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.4.3具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.4.4具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Loadbl模块

#### 4.1.5.1装运管理

###### 模块描述

装运管理模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

装运管理模块的职责及接口参见软件系统结构描述文档。

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，我们会添加businesslogicservice.loadblservice.ShippingBLService接口。业务逻辑层和数据层之间添加dataservice.loaddateservice.ShippingDataService接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了ShippingController，这样ShippingController会将装运管理的业务逻辑处理委托给ShippingBL对象。LoadorderPO是作为装运单的持久化对象被添加到设计模型中去的。Approval是为了将单据提交给总经理审批而产生的接口。

装运管理模块的设计如图：



装运管理模块各个类的设计

装运管理模块各个类的职责如表

|  |  |
| --- | --- |
| 模块 | 职责 |
| ShippingController | 负责实现装运管理界面所需要的服务。 |
| ShippingBL | 具有新建装运单单、设置消息提醒、查看消息提醒、存储新建的装运单等功能，能够帮助完成装运管理界面所需要的服务。 |

###### 模块内部类的接口规范

**ShippingController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Shipping.build | 语法 | public void build(ChangeorderVO cv,String car\_number,String monitor,String transfer) |
| 前置条件 | 已创建一个ShippingBL领域对象，且输入符合规范 |
| 后置条件 | 调用ShippingBL领域的build方法 |
| Shipping.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 已创建一个ShippingBL领域对象 |
| 后置条件 | 调用ShippingBL领域的checkRemind方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Shipping.build | 新建一张装运单 | |
| Shipping.checkRemind | 查看消息提醒 | |

**ShippingBL的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Shipping.build | 语法 | public void build(ChangeorderVO cv,String car\_number,String monitor,String transfer) |
| 前置条件 | 启动一个新建装运单回合 |
| 后置条件 | 系统将装运单提交给总经理审批 |
| Shipping.setRemind | 语法 | public void setRemind(ChangeorderVO cv) |
| 前置条件 | 需要装运的中转单上的货物已出库完毕 |
| 后置条件 | 增加一条提示消息 |
| Shipping.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 启动一个新建装运单回合 |
| 后置条件 | 删除该条消息 |
| Shipping.saveOrder | 语法 | public void save(LoadorderVO) |
| 前置条件 | 到达单已经通过总经理审核 |
| 后置条件 | 将中转单存储 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Approval.checkOrder(PO) | 提交给总经理审核单据 | |
| ShippingDataService.insert(LoadorderPO lp) | 插入一个持久化对象LoadorderPO | |

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.5.2具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Vehiclebl模块

#### 4.1.6.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.6.2具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Financebl模块

#### 4.1.7.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.7.2具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.7.3具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.7.4具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Storebl模块

#### 4.1.8.1入库管理

###### 模块描述

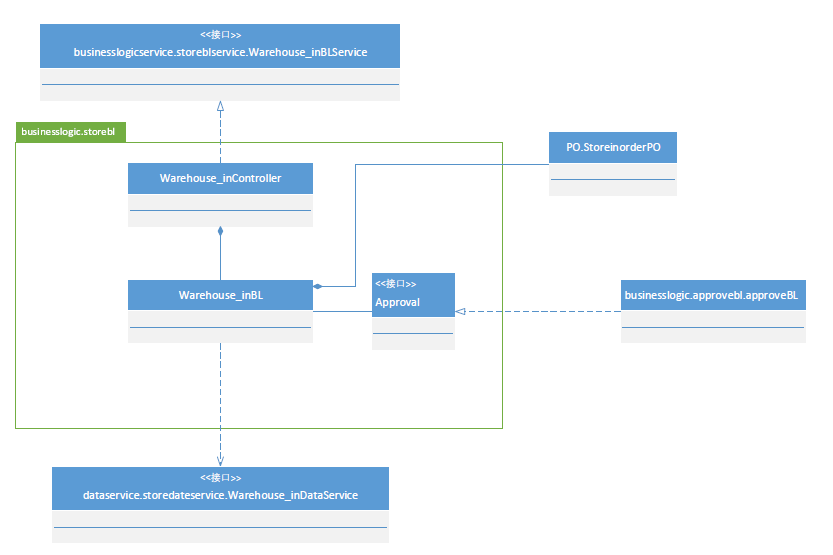
入库管理模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

入库管理模块的职责及接口参见软件系统结构描述文档。

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，我们会添加businesslogicservice.storeblservice.Warehouse\_inBLService接口。业务逻辑层和数据层之间添加dataservice.storedateservice.Warehouse\_inDataService接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了Warehouse\_inController，这样Warehouse\_inController会将对入库管理的业务逻辑处理委托给Warehouse\_inBL对象。StoreinorderPO是作为入库单的持久化对象被添加到设计模型中去的。Approval是为了将单据提交给总经理审批而产生的接口。

入库管理模块的设计如图：



入库管理模块各个类的设计

入库管理模块各个类的职责如表

|  |  |
| --- | --- |
| 模块 | 职责 |
| Warehouse\_inController | 负责实现入库管理界面所需要的服务。 |
| Warehouse\_inBL | 具有新建入库单、设置消息提醒、查看消息提醒、存储新建的入库单等功能，能够帮助完成入库管理界面所需要的服务。 |

###### 模块内部类的接口规范

**Warehouse\_inController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Warehouse-in.build | 语法 | public void build(ArriverorderVO av,int jia,int wei) |
| 前置条件 | 已创建一个Warehouse\_inBL领域对象，并输入符合输入规则 |
| 后置条件 | 调用Warehouse\_inBL领域对象的build方法 |
| Warehouse-in.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 已创建一个Warehouse\_inBL领域对象， |
| 后置条件 | 调用Warehouse\_inBL领域对象的checkRemind方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Warehouse-in.build | 创建一张入库单 | |
| Warehouse-in.checkRemind | 查看消息提醒 | |

**Warehouse\_inBL的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Warehouse-in.build | 语法 | public void build(ArriverorderVO av,int jia,int wei) |
| 前置条件 | 启动一个入库管理回合 |
| 后置条件 | 系统将入库单提交总经理审批 |
| Warehouse-in.setRemind | 语法 | public void setRemind(ArriverorderVO av) |
| 前置条件 | 货物录入到达单 |
| 后置条件 | 增加一条消息提示 |
| Warehouse-in.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 启动一个入库管理回合 |
| 后置条件 | 删除该条消息 |
| Warehouse-in.saveOrder | 语法 | public void save(StoreinorderVO sv) |
| 前置条件 | 入库单通过总经理审批 |
| 后置条件 | 将入库单存储 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Approval.checkOrder(PO) | 提交给总经理审核单据 | |
| Warehouse-in DataService.insert(StoreinorderPO sp) | 插入一个持久化对象StoreinorderPO | |

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.8.2出库管理

###### 模块描述

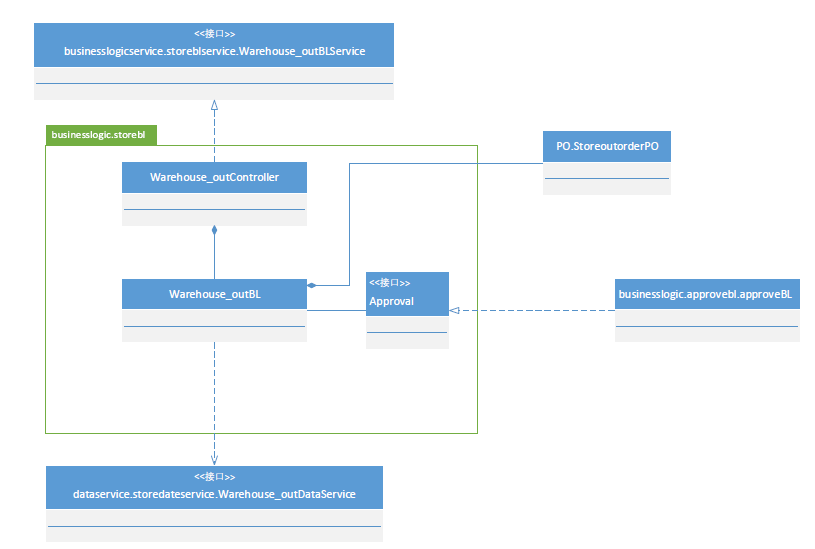
出库管理模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

出库管理模块的职责及接口参见软件系统结构描述文档。

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，我们会添加businesslogicservice.storeblservice.Warehouse\_outBLService接口。业务逻辑层和数据层之间添加dataservice.storedateservice.Warehouse\_outDataService接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了Warehouse\_outController，这样Warehouse\_outController会将对出库管理的业务逻辑处理委托给Warehouse\_outBL对象。StoreoutorderPO是作为出库单的持久化对象被添加到设计模型中去的。Approval是为了将单据提交给总经理审批而产生的接口。

出库管理模块的设计如图：



出库管理模块各个类的设计

入库管理模块各个类的职责如表

|  |  |
| --- | --- |
| 模块 | 职责 |
| Warehouse\_outController | 负责实现出库管理界面所需要的服务。 |
| Warehouse\_outBL | 具有新建出库单、设置消息提醒、查看消息提醒、存储新建的出库单等功能，能够帮助完成出库管理界面所需要的服务。 |

###### 模块内部类的接口规范

**Warehouse\_outController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Warehouse-out.build | 语法 | public void build(ChangeorderVO,String number) |
| 前置条件 | 已创建一个Warehouse\_outBL领域对象，并输入符合输入规则 |
| 后置条件 | 调用Warehouse\_outBL领域对象的build方法 |
| Warehouse-out.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 已创建一个Warehouse\_outBL领域对象， |
| 后置条件 | 调用Warehouse\_outBL领域对象的checkRemind方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Warehouse-out.build | 创建一张入库单 | |
| Warehouse-out.checkRemind | 查看消息提醒 | |

**Warehouse\_outBL的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Warehouse-out.build | 语法 | public void build(ChangeorderVO,String number) |
| 前置条件 | 启动一个出库管理回合 |
| 后置条件 | 系统将出库单提交总经理审批 |
| Warehouse-out.setRemind | 语法 | public void setRemind(ChangeorderVO cv) |
| 前置条件 | 货物录入中转单 |
| 后置条件 | 增加一条消息提示 |
| Warehouse-out.checkRemind | 语法 | public ResultMessage checkRemind() |
| 前置条件 | 启动一个出库管理回合 |
| 后置条件 | 删除该条消息 |
| Warehouse-out.saveOrder | 语法 | public void save(StoreoutorderVO sv) |
| 前置条件 | 出库单通过总经理审批 |
| 后置条件 | 将出库单存储 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Approval.checkOrder(PO) | 提交给总经理审核单据 | |
| Warehouse-out rDataService.insert(StoreoutorderPO sp) | 插入一个持久化对象StoreoutorderPO | |

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.8.3库存管理

###### 模块描述

库存管理模块承担的需求参见需求规格说明文档功能需求及相关非功能需求。

库存管理模块的职责及接口参见软件系统结构描述文档。

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，我们会添加businesslogicservice.storeblservice.Inventory\_managementBLService接口。业务逻辑层和数据层之间添加dataservice.storedateservice. Inventory\_managemenDataService接口。为了隔离业务逻辑职责和逻辑控制职责，我们增加了Inventory\_managemenController，这样Inventory\_managemenController会将对库存管理的业务逻辑处理委托给Inventory\_managemenBL对象。StorePO是作为库存信息的持久化对象被添加到设计模型中去的。

库存管理模块的设计如图：



库存管理模块各个类的设计

入库管理模块各个类的职责如表

|  |  |
| --- | --- |
| 模块 | 职责 |
| Inventory\_managemenController | 负责实现库存管理界面所需要的服务。 |
| Inventory\_managemenBL | 具有查看库存、盘点库存、设置警戒值、设置提醒值等功能，能够帮助完成库存管理界面所需要的服务。 |

###### 模块内部类的接口规范

**Inventory\_managemenController的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Inventory\_management.checkInventory\_now | 语法 | public ResultMessage checkInventory(int store\_qu) |
| 前置条件 | 已创建一个Warehouse\_outBL领域对象 |
| 后置条件 | 调用Inventory\_managemenBL 领域的 |
| Inventory\_management.checkInventory | 语法 | public ResultMessage checkInventory(Data startdata,Data overdata) |
| 前置条件 | 已创建一个Warehouse\_outBL领域对象，且输入时间段符合规范 |
| 后置条件 | 调用Inventory\_managemenBL 领域的checkInventory方法 |
| Inventory\_management.take\_stock | 语法 | public void take\_stock (int store\_number,int store\_qu, Data startdata,Data overdata,string way) |
| 前置条件 | 已创建一个Warehouse\_outBL领域对象，且库区，时间段和查看方式符合规范 |
| 后置条件 | 调用Inventory\_managemenBL 领域的take\_stock方法 |
| Inventory\_management.setWarnData | 语法 | public void setWarnData (int store\_number, int warn\_value) |
| 前置条件 | 已创建一个Warehouse\_outBL领域对象，且输入警戒值 |
| 后置条件 | 调用Inventory\_managemenBL 领域的setWarnData方法 |
| Inventory\_management.setRemindData | 语法 | public void setRemindData (int store\_number, int remind\_value) |
| 前置条件 | 已创建一个Warehouse\_outBL领域对象且输入提醒值 |
| 后置条件 | 调用Inventory\_managemenBL 领域的setRemindData方法 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Inventory\_management.checkInventory\_now | 查看当前各个库区的存储状态 | |
| Inventory\_management.checkInventory | 显示指定时间段各个库区出入库数量和金额 | |
| Inventory\_management.take\_stock | 显示或输出指定库区在指定时间段所有出入库的订单信息 | |
| Inventory\_management.setWarnData | 设置仓库警戒值比例 | |
| Inventory\_management.setRemindData | 设置不同库区提醒值比例 | |

**Inventory\_managemenBL的接口规范**

|  |  |  |
| --- | --- | --- |
| **提供的服务（供接口）** | | |
| Inventory\_management.checkInventory\_now | 语法 | public ResultMessage checkInventory(int store\_qu) |
| 前置条件 | 中转中心业务员启动一个录入中转单回合 |
| 后置条件 | 系统显示各库区库存量 |
| Inventory\_management.checkInventory | 语法 | public ResultMessage checkInventory(Data startdata,Data overdata) |
| 前置条件 | 启动一个查看库存回合 |
| 后置条件 | 系统显示各个库区出入库数量和金额 |
| Inventory\_management.take\_stock | 语法 | public void take\_stock(int store\_number,int store\_qu, Data startdata,Data overdata,string way) |
| 前置条件 | 启动一个盘点库存回合 |
| 后置条件 | 根据查看方式，显示或输出制定库区所有出入库的订单信息 |
| Inventory\_management.setWarnData | 语法 | public void setWarnData (int store\_number, int warn\_value) |
| 前置条件 | 启动一个管理警戒值回合 |
| 后置条件 | 系统更新警戒值 |
| Inventory\_management.setRemindData | 语法 | public void setRemindData (int store\_number, int remind\_value) |
| 前置条件 | 启动一个管理提醒值回合 |
| 后置条件 | 系统更新提醒值 |
| **需要的服务（需接口）** | | |
| 服务名 | 服务 | |
| Inventory\_managementDataService.check(int store\_number) | 查看当前中转中心最新的StorePO | |
| Inventory\_managementDataService.  updata(StorePO sp，int store\_number) | 更新当前中转中心StorePO | |
| Inventory\_managementDataService.  checknumber(int store\_number, Data startdata,Data overdata) | 查看StoreNumber.txt | |
| Inventory\_managementDataService.  addnumber(int store\_number,Data,int storein\_number,int storeout\_number,int storein\_money,int storeout\_money) | 更新StoreNumber.txt | |
| Inventory\_managementDataService.  checkorder(int store\_number,int store\_qu, Data startdata,Data overdata) | 查看StoreOrder.txt | |
| Inventory\_managementDataService.  addorder(int store\_number,OrderPO op) | 更新StoreOrder.txt | |

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Billbl模块

#### 4.1.9.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Accountbl模块

#### 4.1.10.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Staffbl模块

#### 4.1.11.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Policybl模块

#### 4.1.12.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

#### 4.1.12.2具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

### Approvebl模块

#### 4.1.13.1具体用例

###### 模块描述

**自己补全**

###### 整体结构

根据体系结构的设计，我们将系统分为展示层、业务逻辑层、数据层。每一层之间为了增加灵活性，我们会添加接口。比如逻辑层和业务逻辑层之间，**自己补全**

###### 模块内部类的接口规范

**自己补全**

###### 业务逻辑层的动态模型

**自己补全**

###### 业务逻辑层的设计原理

**自己补全**

# 依赖视角



图3 快递管理系统客户端开发包图



图4 快递管理系统服务器端开发包图