**发货程序**

**系统详细设计初稿**

**版本: [1.0]**

**日期: [09/13/2020]**

修订摘要

**发货程序详细设计说明书**

**目录**

[1 介绍 2](#_Toc20736)

[1.1 目的与范围 2](#_Toc6066)

[1.2 假设，依赖与约束 2](#_Toc22630)

[2 系统架构图 2](#_Toc32735)

[3 服务端接口 2](#_Toc5352)

[3.1 安全 7](#_Toc6883)

# 介绍

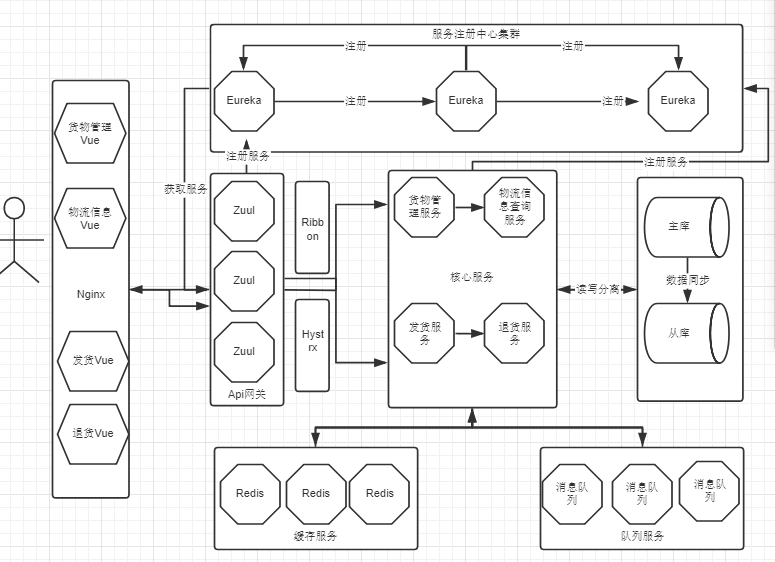
## 目的与范围

本文档致力于从技术角度对发货系统加以说明，包含详细设计、接口设计。

## 假设，依赖与约束

本文档基于需求明确、业务设计合理，详细设计符合业务逻辑，技术能够实现，数据库设计支持现有业务及业务的拓展，接口完整且满足系统需求及日后拓展。

# **系统架构图**

****

# 服务端接口

**货物管理**

|  |  |
| --- | --- |
| **说明项** | **货物管理** |
| **需求简易说明** | **货物的增删改查，价格维护，库存维护等** |
| **通过表单输入货物信息，比如价格，库存等，通过查询列表显示货物列表，通过选择列表某个 项进行货物的修改，比如库存，单价的修改** | |

**数据库**

|  |  |  |
| --- | --- | --- |
| **说明项** | **具体说明** | **操作** |
| **对应表名** | **Good** | **Select/Update/Insert/Delete** |

**类及程序方法**

|  |  |  |
| --- | --- | --- |
| **架构层** | **涉及的相关类** | **相关类中包含的所需要的方法** |
| Controller | GoodController | List<Good> getGoodList(String name)  Good getGood(Integer id)  addGood(Good good)  updateGood(Good good)  deleteGood(Integer id) |
| Service | GoodService | List<Good> getGoodList(String name)  Good getGood(Integer id)  addGood(Good good)  updateGood(Good good)  deleteGood(Integer id) |
| Dao | GoodDao | List<Good> getGoodList(String name)  Good getGood(Integer id)  addGood(Good good)  updateGood(Good good)  deleteGood(Integer id) |
| Entity | Good |  |

**基础信息维护**

|  |  |
| --- | --- |
| **说明项** | **基础信息维护** |
| **需求简易说明** | **物流公司、运输方式数据维护** |
| **物流公司、运输方式的增删改查** | |

**数据库**

|  |  |  |
| --- | --- | --- |
| **说明项** | **具体说明** | **操作** |
| **对应表名** | **infra\_info** | **Select/Insert/Update/Delete** |

**类及程序方法**

|  |  |  |
| --- | --- | --- |
| **架构层** | **涉及的相关类** | **相关类中包含的所需要的方法** |
| Controller | InfraController | List<Infra> getExpressList()  Infra getExpress(Integer id)  addExpress(Infra infra)  updateExpress(Infra infra)  deleteExpress(Integer id)  List<Infra> getTransportWayList()  Infra getTransportWay(Integer id)  addTransportWay(Infra infra)  updateTransportWay(Infra infra)  deleteTransportWay(Integer id) |
| Service | InfraService | List<Infra> getInfraList(InfraEmum infraEmum )  Infra getInfra(Integer id)  addInfra(Infra infra)  updateInfra(Infra infra)  deleteInfra(Integer id) |
| Dao | InfraDao | List<Infra> getInfraList(InfraEmum infraEmum )  Infra getInfra(Integer id)  addInfra(Infra infra)  updateInfra(Infra infra)  deleteInfra(Integer id) |
| Entity | Infra,InfraEmum |  |

**发货流程**

|  |  |
| --- | --- |
| **说明项** | **具体说明** |
| **需求简易说明** | **发货合并及分拆及整体更新，发货的时候库存要考虑并发** |
| 1. **对于发货时默认不做限制，就是一批货发送（数据库中存储的记录是trade表和transportation表关系是1对1（要考虑库存并发情况，通过锁来实现，比如redis的分布式锁，保证某个货物库存数量的一致性）** 2. **如果对于一批货进行拆分运输，则按照拆分数（数据库中存储的记录是trade表和transportation表关系是1对多，比如拆分数是3，则transportation表中是3条运输记录）** 3. **对于已经拆分的货，可以进行合并操作，前提是运输状态还不是运输中（比如transportation表中原先有3条运输记录，现在要合并成2条，做法就是找出2条记录合并成一条记录，另外一条记录不变，效果就是合并后的2条记录，合并完成后总的运输数量还是保持不变）** 4. **对于需要整体按照比例更新运输数量，要考虑库存并发情况，通过锁来实现某个货物库存数量的一致性，整体更新就是对于trade表中运输总数量的修改操作** 5. **整体更新时，通过加减乘除四种方式，在trade表更新运输重数量的时候，自动transportation表中运输记录的更新** 6. **对于退货，也要考虑库存并发情况，因为退货以为这库存的增加，不过不在本次需求范围之内，暂时不考虑** 7. **需要和物流接口对接，能通过Transportation的物流单号进行该运输记录的物流信息，这功能暂时不在本次范围之内** 8. **Transportation表中有个status字段，表示运输状态，比如一开始记录生成的时候是准备状态，然后仓库发货的时候，更新为出库状态，如果物流已揽件，则表示运输状态，所有的合并和拆分及更新数量操作，必须在物流揽件状态前才能更新，否则不允许更新** 9. **对于仓库出库，物流揽件等状态的更新，建议通过消息队列，将状态放入队列，由下游服务进行消费，降低耦合性，但是由于引入了队列，增加了系统复杂度** | |

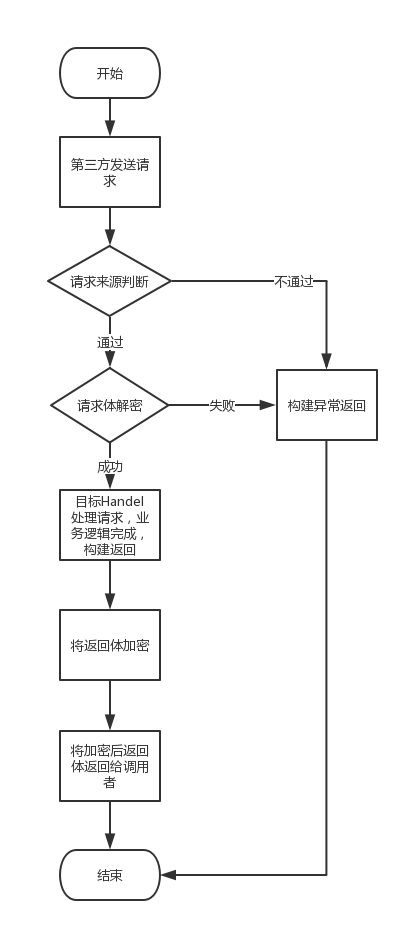
**数据库**

|  |  |  |
| --- | --- | --- |
| **说明项** | **具体说明** | **操作** |
| **对应表名** | **Transportation、trade** | **Select/Insert/Update** |

**类及程序方法**

|  |  |  |
| --- | --- | --- |
| **架构层** | **涉及的相关类** | **相关类中包含的所需要的方法** |
| Controller | TradeController | addTrade(Trade trade)  updateTransportation(Transportation transportation)//修改指定运输记录的信息，比如更新物流信息，更新运输状态等  Trade getTrade(Integer Id)//获取某个贸易交易  List<Transportation> getTransportationByTradeId（Integer tradeId）//获取某个贸易下的所有运输记录信息  splitTransportation(List<Integer> uumber,Integer tradeId)//对于某个贸易交易进行运输批次的拆分，及一批货分几次运输，每次运输运多少货物  mergeTransportation(List<Integer> transportationIdList)//指定哪几批运输批次进行合并成一条运输记录，必须在货物未发货情况下  updaterRootQuantity(Integer tradeId,Integer quantity)//指定贸易交易的总数量更新时，会均匀更新的该贸易交易下的所有运输的数量 |
| Service | TradeService | addTrade(Trade trade)  updateTransportation(Transportation transportation)//修改指定运输记录的信息，比如更新物流信息，更新运输状态等  Trade getTrade(Integer Id)//获取某个贸易交易  List<Transportation> getTransportationByTradeId（Integer tradeId）//获取某个贸易下的所有运输记录信息  split(List<Integer> ids)//对于某个贸易交易进行运输批次的拆分，及一批货分几次运输，每次运输运多少货物  merge(List<Integer> ids)//指定哪几批运输批次进行合并成一条运输记录，必须在货物未发货情况下  updaterRootQuantity(Integer tradeId,Integer quantity)//指定贸易交易的总数量更新时，会均匀更新的该贸易交易下的所有运输的数量  易 |
| Dao | TradeDao | addTrade(Trade trade)  updateTransportation(Transportation transportation)//修改指定运输记录的信息，比如更新物流信息，更新运输状态等  Trade getTrade(Integer Id)//获取某个贸易交易  List<Transportation> getTransportationByTradeId（Integer tradeId）//获取某个贸易下的所有运输记录信息  split(List<Integer> ids)//对于某个贸易交易进行运输批次的拆分，及一批货分几次运输，每次运输运多少货物  merge(List<Integer> ids)//指定哪几批运输批次进行合并成一条运输记录，必须在货物未发货情况下  updaterRootQuantity(Integer tradeId,Integer quantity)//指定贸易交易的总数量更新时，会均匀更新的该贸易交易下的所有运输的数量 |
| Entity | Trade(交易主表） ,Transportation（运输记录表) |  |

## 安全

**在传输方面，所有对外请求均采用加密传输，流程如下：**