**软件工程课程设计**

**《智能物流管理系统项目》**

**软件需求规格说明书**

**小组：目标是成为抽象带师**

**设计组学生：**

**阿琪 1120161708**

**周赫斌 1120161737**

**万伯阳 1120161725**

**指导教师：汤世平**

**北京理工大学**

**2019年11月**

|  |  |
| --- | --- |
| 项目名称 | 智能物流管理系统 |
| 文档 | **概要设计说明书** |
| 版本 | **V1.0** |
| 作者 | **阿琪** |
| 最后更新时间 | **2019-11-14** |

目录

[1 引言 4](#_Toc24616452)

[1.1 编写目的 4](#_Toc24616453)

[1.2 项目背景 4](#_Toc24616454)

[1.3 定义 4](#_Toc24616455)

[1.4 参考资料 4](#_Toc24616456)

[2 任务概述 5](#_Toc24616457)

[2.1 目标 5](#_Toc24616458)

[2.2 运行环境 5](#_Toc24616459)

[3 系统分布设计 5](#_Toc24616460)

[3.1 表示层服务： 5](#_Toc24616461)

[3.2 应用逻辑服务： 5](#_Toc24616462)

[3.3 数据存储服务 6](#_Toc24616463)

[4 层次分布图 7](#_Toc24616464)

[5 数据结构设计 7](#_Toc24616465)

[5.1 用户 7](#_Toc24616466)

[5.2 订单 8](#_Toc24616467)

[5.3 货物 8](#_Toc24616468)

[5.4 物流记录 9](#_Toc24616469)

[5.5 仓库 9](#_Toc24616470)

[5.6 路径 10](#_Toc24616471)

[5.7 承运商 10](#_Toc24616472)

[6 算法设计 10](#_Toc24616473)

[6.1 仓库间货物调度算法： 10](#_Toc24616474)

[6.2 仓库内货物摆放位置调度算法 11](#_Toc24616475)

[6.3 货物摆放空间最优化算法 11](#_Toc24616476)

[6.4 最优路径计算算法 12](#_Toc24616477)

# 引言

## 编写目的

为了明确软件大概的设计方向和大致功能模块的实现，编写本文档。

## 项目背景

随着电子商务的兴起，我国物流产业的业务量暴增，从业人员的数量也随之迅速增长。但是由于物流行业的成本高居不下，同行业之间的低价竞争，也使得物流企业的利润不升反降。同时，作为劳动密集型产业，众多物流企业规模小，信息化程度低，缺少物流标准化体系，过于激烈的竞争降低服务质量等都现如今急需解决的问题。

本系统将物流网络系统和运输系统深度结合，实现智能运输、智能识别和自动分拣的“物自（智）流”过程，提高物流企业的品质和效率。

## 定义

该软件是针对物流服务行业的系统化、智能化、方便快捷化的信息管理和物流调度平台。

## 参考资料

附录8(软件设计文档编写纲要)

《智能物流管理系统用户需求》

# 任务概述

## 目标

实现信息的系统化管理，物流调度的自动化设置。使所有使用该软件的用户都极大的方便工作物流。

## 运行环境

PC端windows7、windows10及以上版本；手机端IOS系统、安卓系统

# 系统分布设计

## 表示层服务：

用户登陆注册界面网页版及手机版；

业务办理服务；

用户投诉及反馈服务；

用户信息查询服务

管理员信息管理服务；

承运商信息查询服务；

## 应用逻辑服务：

用户信息增删改查服务；

订单信息增删改查服务；

投诉业务及客服业务转接服务；

信息通知服务；

仓库调度服务；

运输路线计算服务；

服务器同步更新服务；

## 数据存储服务

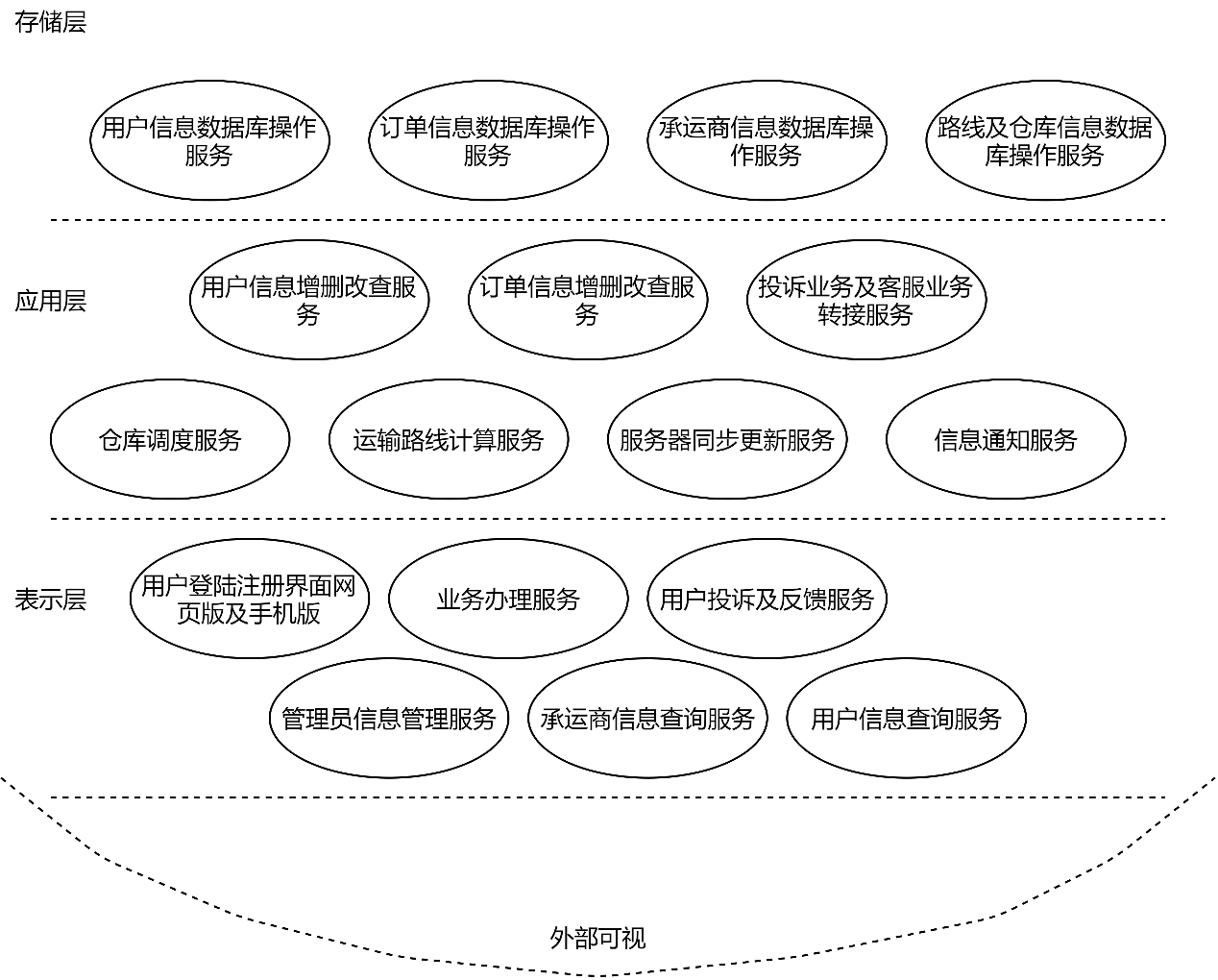
用户信息数据库操作服务；

订单信息数据库操作服务；

承运商信息数据库操作服务；

路线及仓库信息数据库操作服务；

# 层次分布图



1. 层次分布图

# 数据结构设计

## 用户

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段**  **说明** | **名称** | **类型** | **长度** | **处理说明** |
| 用户ID | User\_ID | int | 10 | 无 |
| 用户名 | User\_userName | char | 变长 | 用户名只包含汉字数字及英文字母 |
| 密码 | User\_password | string | 变长 | 密码只包含ASIIC码 |
| 用户权限等级 | User\_authority | short | 1 | 1-10之一代表客户和工作人员的权限等级，由用户指定 |
| 姓名 | User\_name | char | 变长 | 任意字符 |
| 性别 | User\_gender | bool | 1 | 1:男性 0:女性 |
| 身份证号 | User\_IDNumber | int | 19 | 无 |
| 手机号码 | User\_CellphoneNumber | int | 11 | 无 |
| 邮箱地址 | User\_EMailAddress | char | 变长 | 无 |
| 常用发送地址 | User\_CommonAddress | Char | 变长 | 无 |

## 订单

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段**  **说明** | **名称** | **类型** | **长度** | **处理说明** |
| 订单ID | Order\_ID | Int | 10 | 无 |
| 运送货物ID | Order\_CargoID | Int | 10 | 无 |
| 订单提交人ID | Order\_SubmitterID | Int | 10 | 无 |
| 订单审核人ID | Order\_AuditorID | Int | 10 | 无 |
| 出发地地址 | Order\_DepatureAddress | Char | 变长 | 任何字符 |
| 目的地地址 | Order\_DestinationAddress | Char | 变长 | 任何字符 |
| 特殊要求 | Order\_SpecialRequirement | Char | 变长 | 任何字符 |
| 支付情况 | Order\_PayoutState | bool | 1 | 0：未支付 1：支付 |
| 运费数额 | Order\_Freight | Double | 1 | 无 |
| 运送方式 | Order\_DeliveryType | enum | 1 | Truck:卡车  Train:火车  Airplane:飞机  Ship:海运 |
| 完成情况 | Order\_Performance | bool | 2 | 00：未递送  01：运送中  10：已抵达  11：已接收 |

## 货物

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段**  **说明** | **名称** | **类型** | **长度** | **处理说明** |
| 货物ID | Cargo\_ID | Int | 10 | 无 |
| 货物提交人ID | Cargo\_SubmitterID | Int | 10 | 无 |
| 货物名称 | Cargo\_Name | Char | 变长 | 无 |
| 货物长度 | Cargo\_Length | Double | 1 | 单位cm |
| 货物宽度 | Cargo\_Width | Double | 1 | 同上 |
| 货物高度 | Cargo\_Height | Double | 1 | 同上 |
| 大概价值 | Cargo\_Value | Double | 1 | 单位￥ |
| 货物种类 | Cargo\_Type | Enum | 1 | Small:普通小件  Big:普通大件  Special:特殊  Fresh:生鲜 |
| 特殊要求 | Cargo\_Demand | char | 变长 | 无 |

## 物流记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段**  **说明** | **名称** | **类型** | **长度** | **处理说明** |
| 货物ID | Record\_CargoID | int | 10 | 无 |
| 订单ID | Record\_ID | Int | 10 | 无 |
| 记录人ID | Record\_AdministratorID | Int | 10 | 无 |
| 货物下一站地点 | Record\_NextPos | Int | 1 | 代表地点的序号 |
| 货物状态 | Record\_CargoState | enum | 1 | 待定 |
| 记录时间 | Record\_Time | date | 1 | 无 |
| 承运商ID | Record\_CarrierID | int | 10 | 无 |

## 仓库

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段**  **说明** | **名称** | **类型** | **长度** | **处理说明** |
| 仓库ID | Warehouse\_ID | Int | 10 | 无 |
| 仓库名 | Warehouse\_Name | Char | 变长 | 无 |
| 已中转货物数 | Warehouse\_TransitNumber | Int | 1 | 无 |
| 事故数 | Warehouse\_Accident | Int | 1 | 无 |
| 大型货物总仓储能力 | Warehouse\_LargeCapacity | Int | 1 | 无 |
| 小型货物总仓储能力 | Warehouse\_SmallCapacity | Int | 1 | 无 |
| 当前大型货物剩余仓储能力 | Warehouse\_LargeInventory | Int | 1 | 无 |
| 当前小型货物剩余仓储能力 | Warehouse\_SmallInventory | int | 1 | 无 |

## 路径

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段**  **说明** | **名称** | **类型** | **长度** | **处理说明** |
| 路径ID | Edge\_ID | Int | 10 | 无 |
| 起始仓库点 | Edge\_StartWarehouse | Int | 1 | 无 |
| 终止仓库点 | Edge\_EndWarrhouse | Int | 1 | 无 |
| 路径承运商ID | Edge\_CarrierID | Int | 10 | 无 |
| 已发送货物数 | Edge\_SentCargo | Int | 1 | 无 |
| 运送中事故数 | Edge\_SendingCargo | Int | 1 | 无 |
| 货物完好评级 | Edge\_CargoIntact | Double | 1 | 无 |
| 运费评级 | Edge\_Freight | Double | 1 | 无 |
| 运送时间评级 | Edge\_Time | double | 1 | 无 |

## 承运商

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段**  **说明** | **名称** | **类型** | **长度** | **处理说明** |
| 承运商ID | Carrier\_ID | int | 10 | 无 |
| 承运商名称 | Carrier\_Name | Char | 变长 | 无 |
| 承运商价格 | Carrier\_Price | Double | 1 | 无 |
| 运货能力评级 | Carrier\_Capability | double | 1 | 无 |

# 算法设计

## 仓库间货物调度算法：

**算法思想：**当仓库满时，应当设定所有送达该仓库的路线的权值为充分大，以停止往该仓库送货。若有发往该仓库的货物已经开始发送，在到达后没有足够的空间容纳货物，则先在仓库外等待，直到仓库有足够的容量容纳货物。当仓库的货物容量下降到80%以下，则重新设定送达该仓库的路线，以使得它能正常工作。

**算法实现：**在每次最新的仓库库存数据上传后，统一更新所有仓库的点的数据。

仓库权值 = （1-仓库当前大型货物仓储能力 / 仓库总大型货物仓储能力） + （1-仓库当前小型货物仓储能力 / 仓库总小型货物仓储能力）

当仓库权值大于1.95时将之设为无穷大，并停止路径设定，当新计算出来的仓库权值小于1.6且路径设定被停止时，重新启动该仓库的路径设计。

|  |
| --- |
| double val = （1 – Warehouse\_LargeInventory / Warehouse\_LargeCapacity） + (1 - Warehouse\_SmallInventory / Warehouse\_SmallCapacity); //计算仓库权值  int Design = 1; // 当前仓库是否进行路径设计  if (val > 1.95) {  val = 0x3f3f3f3f3f;  Design = 0;  }  else {  if (val < 1.6) {  Design = 1;  }  } |

## 仓库内货物摆放位置调度算法

**算法思想：**对于每个仓库，将仓库按矩形计算，保存该矩形的长宽。将该仓库对应的一整个大矩形分成若干较小的矩形，该较小的矩形应该能容纳可能入库的最大批次货物，同时在矩形之间保留能够容纳自动运货机器人或工人移动的最小道路。对于所有入库的货物，将其考虑为一个长方体（通过货物摆放空间最优化算法得出）。能够得到该长方体的长宽高。根据长宽高的属性，将货物分为小体积货物和大体积货物两类。小体积货物定义为长宽高都不超过30cm的货物，大体积货物定义为长宽高中存在一维超过30cm的货物。所有危险品都应当该增加长宽高至少达到大体积货物的标准。小体积货物和大体积货物应当分开存放。对于小体积货物，统一安排货架盛放。对于大体积货物，首先考虑对于不要求顶部向上的货物翻转长宽高使得占地面积最小。由于大体积货物一般重量不轻，所以不考虑叠放。因为对于大体积货物重新排列的成本太高，所以只考虑在空隙处放下物品。从俯视图看，每个货物是一个二维的矩形。把所有货物的长宽上取整到10cm的倍数，然后暴力枚举每个货物的四个顶点，如果以这个点为左上顶点可以放下待处理物品就放下。

## 货物摆放空间最优化算法

因为货物形状不一，且包含特殊要求，情况过于复杂，无法用一种单一算法概括，这里分类讨论。

该算法优先考虑有破损、爆炸的风险以及有各种特殊要求的货物。对于可叠放、无特殊要求、破损风险较小的普通货物（后称普通货物）则后考虑。

首先对于有爆炸风险的货物，不应将之与其他可能会发热或运送途中易发生碰撞的零散货物摆放在一起。应安全摆放该种货物。可将货物的长宽高在货物的实际长宽高基础上适当提升，以便留出一定安全距离。同时有危险的货物附近应当做好消防措施，如准备灭火器等。

对于上方无法放置物品的货物，当它是小体积货物时放在货架上则不会有问题，当它是大体积货物时上方不会堆叠其他货物。

## 最优路径计算算法

算法思想：每个仓库为一个点V**i**，每条路径为一条边E**ijk**，代表点i到点j之间的第k条路径。每条边有权值D**ijk**，该权值由该路径的平均通过时间，事故发生率和运费加权求和得出。

在每次路径数据更新时，对于每两个仓库之间的所有路径，找出权值D最小的路径，用P**ij**表示，代表直接从一个仓库到达另一个仓库的最优路径。

在得到所有P**ij**后用Dijkstra算法得出仓库之间的最短路。

算法实现：

路径权值 = C1 \* 路径平均通过时间 + C2 \* 运费 + C3 \* 事故发生率

(C1 C2 C3需要由客户即物流公司自己设定)

Pij = min { Dijk | Eijk }

调用Dijkstra算法更新所有点之间的最短路径。

|  |
| --- |
| // Dijkstra算法  //  初始化,设从0开始  for i=[0,n)      dist[i] = map[0][i]  visit[0] = true;    for i=[1,n)      //  寻找最短路径(s,t)，同时把t加入S集合      min = MAX\_VALUE      for j=[0,n)          if !visit[j] && dist[j]<min              min = dist[j]//记录最小值和最小值的下标  min\_j = j        visit[j] = true        //  松弛边(t,v)，其中v为顶点      for k=[0,n)          if !visit[k] && dist[k]>dist[j]+tab[j][k]              dist[k] = dist[j]+tab[j][k] |