目录

[功能概要设计 3](#_Toc455682754)

[业务顺序图 4](#_Toc455682755)

[系统工作流程图 5](#_Toc455682756)

[拣货 5](#_Toc455682757)

[订单分配给拣货员 5](#_Toc455682758)

[小车运货架到拣货员面前 5](#_Toc455682759)

[拣货员拣一个货架上的货 6](#_Toc455682760)

[小车将货架运回仓储区 7](#_Toc455682761)

[订单切换 8](#_Toc455682762)

[补货 9](#_Toc455682763)

[补货员对商品扫码，小车取货架 9](#_Toc455682764)

[小车运货架到补货员面前（同拣货） 9](#_Toc455682765)

[补货员将商品放到货架 10](#_Toc455682766)

[小车将货架运回仓储区（同拣货） 10](#_Toc455682767)

[商品切换 10](#_Toc455682768)

[盘点 10](#_Toc455682769)

[按货架盘点 10](#_Toc455682770)

[按商品盘点 11](#_Toc455682771)

[按拣货员/订单盘点 11](#_Toc455682772)

[软件模块图 12](#_Toc455682773)

[软件UML图 13](#_Toc455682774)

[包图 13](#_Toc455682775)

[类图 13](#_Toc455682776)

[数据表结构 14](#_Toc455682777)

[持久记录 14](#_Toc455682778)

[实时状态记录 16](#_Toc455682779)

[日志、异常记录 17](#_Toc455682780)

[通讯协议 18](#_Toc455682781)

[协议格式 18](#_Toc455682782)

[功能码 18](#_Toc455682783)

[路径规划 20](#_Toc455682784)

[选择策略 20](#_Toc455682785)

[待确定问题 21](#_Toc455682786)

# 功能概要设计

1. 产品上架
2. 扫码仓库库位条码/直接手动输入库位编号（库区-通道-货架-层号-格号）
3. 再依次扫码产品/直接录入产品ID(按箱/组合上架产品暂不考虑)
4. 选择订单开始打包商品，绑定订单和盛放商品的容器，简称订单容器
5. 获取订单中的产品
   1. 选择小车设备（可能当前有多个设备可用）和目标货架（可能有多个货架上有对应商品）
   2. 分配设备取货（设备自行制定行走路线，给出相对坐标值，只需要向减少间距方向移动即可）
   3. 拣货员将商品从货架下，扫码/直接出库产品，然后拿至订单容器
   4. 设备归位货架（同b，都属于在两点间通过设备移动货架）
6. 循环2中步骤，直至订单完成，换新订单
7. 安排设备充电

# 业务顺序图

员工

中央系统

小车设备

到指定充电桩充电

进入待机状态

充电完成

小车空闲或电量低于40%

选择小车、货架

按预定路线，将货架送至拣货员，排队等候

送货架回指定位置

订单完成，换新订单

(扫描)订单拣货、扫码商品补货

拣货员拿商品下架、补货员上架

盘点员选货架盘点

选最近空闲小车

按预定路线，将货架送至盘点员，排队等候

盘点员记录实际商品名称规格和数量

# 系统工作流程图

拣货（暂不考虑通过设备运送订单货架，仅通过人工选择订单容器）

### 订单分配给拣货员

拣货员按开始按钮

中央系统为拣货员的N个空的订单容器绑定N个新订单（商品交叉）

初始订单分配完成

### 小车运货架到拣货员面前

中央系统安排小车排队等候

中央系统发现，拣货员面前已有小车在

否

是

中央系统完成，通过小车将货架运到拣货员前

小车按计划路线找到货架，并将货架运至拣货台

中央系统为小车规划好行走路线，并实时监控小车位置

中央系统找到订单中产品对应货架（最近拣货员），找到空闲小车（最近货架）

拣货员已分配订单

拣货员拣一个货架上的货小车将货架运回仓储区

中央系统检测扫码商品跟预期是否相同

是

否

拣货员放回后重拿商品，并扫码

中央系统识别出当前货架对应当前拣货员的一个商品并指示商品位置

拣货员将商品从货架拿出，并扫码

小车在拣货员面前

中央系统识别商品对应拣货员的订单容器，点亮订单容器的提示灯

拣货员将商品放入订单容器，并关闭订单容器的提示灯

中央系统检查货架上有其他产品，属于当前的N个订单中

是

否

拣货员完成一货架拣货

否

否

小车在拣货员面前

中央系统根据货架上SKU的出货频率，为货架算出新位置

中央系统为小车规划好行走路线，并实时监控小车位置

小车按计划路线将货架运至目标位置，并放下货架

中央系统判断小车电量符合标准

是

中央系统找到空闲的充电桩，并规划好路线，监控小车到充电桩充电

中央系统判断是否有新的搬运任务

小车充电完成，通知中央系统后进入待机状态

是

中央系统安排小车进行新一轮搬运

订单切换

拣货员将商品放入订单容器，并关闭订单容器的提示灯

是

不做操作

中央系统判断订单中商品是否完成

否

中央系统点亮订单完成指示灯

打包员换上空的订单容器，并关闭订单完成指示灯

中央系统为空的订单容器分配一份新的订单

订单切换完成

## 补货

补货应该是先给出补货计划，补货计划是对于订单的预期以及当前库存量得到的，所以，补货商品是在库存中有预期的对应库位的。

操作是拣货的逆向过程，区别有两点：1，定位货架的查询条件不同：当补货员扫描商品后，根据要补货的数量和货架自身空位量，以及最近一周跟补货商品的销售相似度，综合考虑来决定为哪个货架进行补货，从而选择哪个货架；而拣货需要是订单中的产品，所以只需要考虑，订单需要移动的货架数量、商品先进先出和距离拣货员的距离作为考虑因素。2，货架往返的执行结果不同：对于要补货的货架是上架商品，拣货是相反的下架商品。

### 补货员对商品扫码，小车取货架

补货员扫描商品二维码

中央系统算出当前数量的商品预期的所有货架

中央系统找到空闲的设备，并规划出相应的路径

中央系统安排小车取货架

小车取货架完成

### 小车运货架到补货员面前（同拣货）

### 补货员将商品放到货架

中央系统检测扫码库位跟预期是否相同

是

否

补货员重新扫码

中央系统识别出当前货架，并指示当前商品在货架的位置

补货员扫描库位码

小车在拣货员面前

补货员将指定数量的商品放入库位

中央系统检查货架上有其他位置，属于当前商品

否

拣货员完成一货架补货

是

### 小车将货架运回仓储区（同拣货）

### 商品切换

补货员换个商品进行扫码

## 盘点

盘点操作跟拣货类似，两个区别分别是：1，对于货架的选择条件，此处是随机一个货架；2，没有对货物的上下架，仅检查核实，操作之后没有商品的增减。

### 按货架盘点

随机选中一台货架，给出系统中货架上的产品及数量，由盘点员确认对应的实际结果。

### 按商品盘点

随机选中一款商品，给出系统中的货架上的产品及数量，有盘点员确认对应的实际结果。

### 按拣货员/订单盘点

# 软件模块图

业务模块

设备移动货架

设备排队

设备充电

应用模块

设备状态监控

商品库存增删查改

订单状态监控

拣货台信息

基础资料

设备信息

商品信息

库位信息

充电桩信息

订单信息

基础功能

设备通信

选车\货架策略

路径规划

# 软件UML图

## 包图

各资料信息表

数据访问层库

业务逻辑层库

UI应用层包

基础算法库

## 类图

基础信息类实体

基础功能类库

IDAL

DAL

DALMysql

DALRedis

数据访问类库

UI模块

拣货员、补货员使用的提示界面

订单的监控界面

小车的监控界面

# 数据表结构

持久记录（用于初始化）

|  |
| --- |
| **拣货员表** |
| 人员ID：1 |
| 姓名：张三 |
| 性别：男 |
| 权限：11 |
| 职位：管理员 |
| 年龄：20 |
| 手机：150XXX |
| 联系地址：南山蛇口 |

|  |
| --- |
| **货架信息表** |
| 货架ID：1 |
| 货架编码(6位：仓库2+行数1+列数2+层数1)：02A271 |
| 绝对坐标X，Y：2，3 |
| 货架层数：4 |
| 货架面数：2 |
| 各面（分号分隔）每层格数：01020201; 01020301 |
| 类型（冗余字段：1小2中3大11冷）：1 |

|  |
| --- |
| **商品货架关系表（bak）** |
| 商品ID(通过条码/ID唯一定位)：1 |
| SKU ID（同个SKU包含多个商品）：1 |
| 货架ID：1 |
| 状态（0上架1出库）：0  {定期备份并清除已出库产品} |

|  |
| --- |
| **商品信息表（bak）** |
| 商品ID(通过条码/ID唯一定位)：1 |
| 货架面号（最多4个面）：1 |
| 库位号（当前货架所有库位自下到上，自左到右编号）：2  {此库位仅用于显示给拣货员：若将库位，层号信息分别用数据记录表示，数据冗余不增加查询和计算效率} |
| 商品名称及规格（用于显示）：水杯；红色300ml |
| 生产日期：2015-07-01 |
| 过期日期：2016-12-31 |
| 尺寸规格(mm)：20\*200\*2000 |
| 重量(g)：200 |
| 上架时间（扫码/指派）：2016-07-01 10:51:50 |
| 出库时间（扫码/指派）：2016-07-10 10:51:50 |
| 状态（0上架1扫码出库2未扫码出库）：0 |

|  |
| --- |
| **设备信息表** |
| 设备ID：1 |
| 设备序号（可读编号）：S0012 |
| 状态（1待命2取货3运货4充电9故障）：1 |
| 绝对坐标X，Y：（33，44） |

|  |
| --- |
| **充电桩** |
| 序号ID：1 |
| 可读序号：DZ002 |
| 绝对坐标X，Y：2，3 |
| 状态（1空闲2工作中9故障）：2 |

|  |
| --- |
| **补货/拣货台** |
| 序号ID：1 |
| 可读序号：HT002 |
| 绝对坐标X，Y：2，3 |
| 类型（1补货2拣货）：1 |
| 并发订单数：6 |

实时状态记录（用于跟踪调试 - Mysql）

|  |
| --- |
| **实时订单表**  订单开始拣货时增加记录，每次录入商品更新记录 |
| 订单号：5542144 |
| 商品总数：5 |
| 商品信息（SKU ID，数量）：1,2;3,1;4,1;6,1 |
| 拣货员ID：3 |
| 取货设备（冗余字段：设备编号）：1,2 |
| 取货商品（冗余字段：SKU ID，条码ID，数量;）：1,1,1;1,2,1;3,3,1 |
| 已取数量：3 |
| 状态（0完成1拣货中2异常）：1 |

|  |
| --- |
| **设备取货表**  给设备发布取货任务时增加记录，拣货员拿到商品后更新状态 |
| 商品ID（通过条码/ID唯一定位）：1 |
| SKU ID（同个SKU包含多个商品）：1 |
| 货架ID：1 |
| 商品数量：1 |
| 状态（1取货中2已出库）：1 |
| 设备ID：1 |
| 开始取货时间：2016-07-01 13:50:10 |
| 扫码出库时间：2016-07-01 13:59:10 |

|  |
| --- |
| **设备状态表**  设备连接后，每秒增加5条记录（前期先用Mysql，吞吐支持不到时再独立） |
| 设备ID：1 |
| 状态（0候命中1取货中2运货中3电量低4充电中11故障12失联）：1 |
| 绝对位置X，Y：22，33 |
| 当前时间：2016-07-01 13:50:10 |
| 备注：电量低时记录电量，取货中时记录商品名称，运货中时记录货架编号 |

日志、异常记录每步操作（用于历史备案 - 文档/Nosql）

|  |
| --- |
| **日志表** |
| 记录时间：2016-07-01 11:20:50 |
| 对象类型（1设备2拣货员）：1 |
| 操作步骤（功能名称）：设备取货 |
| 日志内容（操作记录）：设备接收取货指令（XXX） |

|  |
| --- |
| **异常记录表** |
| 记录时间：2016-07-01 11:20:50 |
| 对象类型（1设备2拣货员）：1 |
| 操作步骤：设备取货 |
| 异常内容：发信息（XXX）获取设备（XXX）状态，响应超时 |

# 通讯协议

## 通信协议格式

上位机向下位机发送信息是双向的，即不论上位机查询小车设备的状态，还是向设备发送需执行的命令，都需要设备给予信息反馈。

设备主动向上位机发送信息是单向通信，即不需要上位机回复。

|  |  |
| --- | --- |
| 保留位1 | 头文件属性 |
| 保留位2 | 头文件属性 |
| 功能码1 | 0x01 |
| 功能1数据高字节 | 0 |
| 功能1数据低字节 | N1 |
| 。。。。。。 | 多功能保留位 |
| 功能码4 | 0x04 |
| 功能4数据高字节 | 0 |
| 功能4数据低字节 | N4 |
| 数据位1 | 0xAF |
| 数据位2 | 0xAF |
| …… | …… |
| 数据位N1 | 0xAF |
| 若有其他功能的数据则依次顺序排序，若没有则不保留位置（包括功能1） | |
| 校验码 | 0xAF |

## 功能码

|  |  |  |
| --- | --- | --- |
| 功能名称 | 功能码 | 数据格式 |
| 上位机：查询状态 | 0x10 | 工作状态（1 Byte）{0闲置，2充电，3取货架，4送拣货台，5排队，6归位货架，1故障}，电量（1 Byte）{百分比值}，X坐标（2 Byte），Y坐标（2 Byte），Z坐标（1 Byte） |
| 上位机：  安排充电 | 0x20 | 充电桩ID（1 Byte），第一步X（2 Byte），第一步Y（2 Byte）, 第一步Z（1 Byte）, ……,第i步X（2 Byte），第i步Y（2 Byte）, 第i步Z（1 Byte）……, 最后一步X（2 Byte），最后一步Y（2 Byte）, 最后一步Z（1 Byte） |
| 上位机：  移动到位置等待 | 0x21 | 第一步X（2 Byte），第一步Y（2 Byte）, 第一步Z（1 Byte）, ……,第i步X（2 Byte），第i步Y（2 Byte）, 第i步Z（1 Byte）……, 最后一步X（2 Byte），最后一步Y（2 Byte）, 最后一步Z（1 Byte） |
| 上位机：  去找货架 | 0x22 | 货架ID（2位），第一步X（2 Byte），第一步Y（2 Byte）, 第一步Z（1 Byte）, ……,第i步X（2 Byte），第i步Y（2 Byte）, 第i步Z（1 Byte）……, 最后一步X（2 Byte），最后一步Y（2 Byte）, 最后一步Z（1 Byte） |
| 上位机：  运货架到拣货台 | 0x23 | 拣货台ID（2位），第一步X（2 Byte），第一步Y（2 Byte）, 第一步Z（1 Byte）, ……,第i步X（2 Byte），第i步Y（2 Byte）, 第i步Z（1 Byte）……, 最后一步X（2 Byte），最后一步Y（2 Byte）, 最后一步Z（1 Byte） |
| 上位机：送回货架到仓储区 | 0x24 | 货架ID（2位），第一步X（2 Byte），第一步Y（2 Byte）, 第一步Z（1 Byte）, ……,第i步X（2 Byte），第i步Y（2 Byte）, 第i步Z（1 Byte）……, 最后一步X（2 Byte），最后一步Y（2 Byte）, 最后一步Z（1 Byte） |
| 小车：  当前状态 | 0x30 | 工作状态（1 Byte ）{0闲置，2充电，3取货架，4送拣货台，5排队，6归位货架，1故障}，电量（1 Byte）{百分比值}，X坐标（2 Byte），Y坐标（2 Byte），Z坐标（1 Byte） |
| 小车：  电量低 | 0x31 | 电量（1 Byte）{百分比值），X坐标（2 Byte），Y坐标（2 Byte），Z坐标（1 Byte） |
| 小车：  遇到障碍 | 0x32 | 障碍距离（1 Byte），X坐标（2 Byte），Y坐标（2 Byte），Z坐标（1 Byte） |
| 小车：  超载 | 0x33 | 货物重量（2 Byte），X坐标（2 Byte），Y坐标（2 Byte），Z坐标（1 Byte） |
| 小车：  货物不稳 | 0x34 | 货物重量（2 Byte），X坐标（2 Byte），Y坐标（2 Byte），Z坐标（1 Byte） |
| 小车：  未知异常 | 0x39 | X坐标（2 Byte），Y坐标（2 Byte），Z坐标（1 Byte） |

# 路径规划

直接按坐标接近来决定行走方向，转弯越少越好

选择策略（移动总曼哈顿距离最短）

选订单：

初始订单时，选择有更多共同商品，或者在相同货架的订单；

换新订单时，选择跟当前拣货员待拣商品更多重合的订单

选货架：

拣货时，选择拥有更多待检订单商品、靠近拣货台的货架

补货时，选择跟对应SKU出货频率接近的货架，同时满足更接近（大于）当前商品尺寸空库位的货架

选小车：

最靠近货架的空闲小车

待确定问题：

1. 对接原OMS/WMS数据（订单、产品）的方案

关键点：

* 订单信息：仅需要可以定位到商品即可（商品ID），需要OMS提供接口获取订单对应的产品（通常商家可能不愿意提供订单的客户信息）
* 产品所在库位信息：跟WMS系统绑定，或者用WMS中产品信息进行初始化
* 库位、拣货台、充电桩及设备信息：独立导入/录入

1. 数据存储方案

关键点：

* + 基础信息存储在持久数据库Mysql中（数据格式相对固定不考虑Nosql，收费不考虑Sqlserver，选择Mysql）
  + 多个小车位置状态改变频繁，每秒更新5次小车位置，Mysql可以胜任100台左右小车，若更多小车，则需要引入实时数据库ExtremeDB（会有高并发不考虑内存数据库Sqlite）

1. 对设备的路径规划及运动控制是设备独立实现，不采用中心调度
2. 入库/补货阶段如何通过商品二维码跟商品SKU做绑定