**基于RFID技术的航空行李  
人工分拣辅助系统**

—详细设计

**密级：秘密**

**版本：0.1**

**2016-7-28**

目录

# 版本修订记录

类别：A-增加；M-修改；D-删除

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 修订日期 | 版本号 | 修订类别 | 描述 | 修改人 |
| 2016-7-15 | 0.1 | A | 建立文档。 | 贾环宇 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 文档背景

## 文档目标

本文档为“基于RFID技术的航空行李人工分拣系统”详细设计文档，旨在对系统的详细信息进行阐述，并描述技术方案中的各种技术细节。

## 背景

北京首都国际机场含有包括T1、T2、T3在内的三个航站楼，其中T1航站楼由海南航空全部租用，用以办理海南航空集团旗下的各种业务。由于T1航站楼修建历史较久，其行李分拣仍然采用人工的方式，效率低下，出错率较高。



图 1 这里是图1

图1展示了T1航站楼行李分拣装置。来自值机处的行李通过传送带运往T1航站楼地下，传递至由两条互相连接的环形传送带所构成的分检池上。工人在分检池旁某个地点对行李实施分拣，将传送带上的行李按照不同的航班分别放置在对应运输车上。分检池的形状和位置如图2所示。

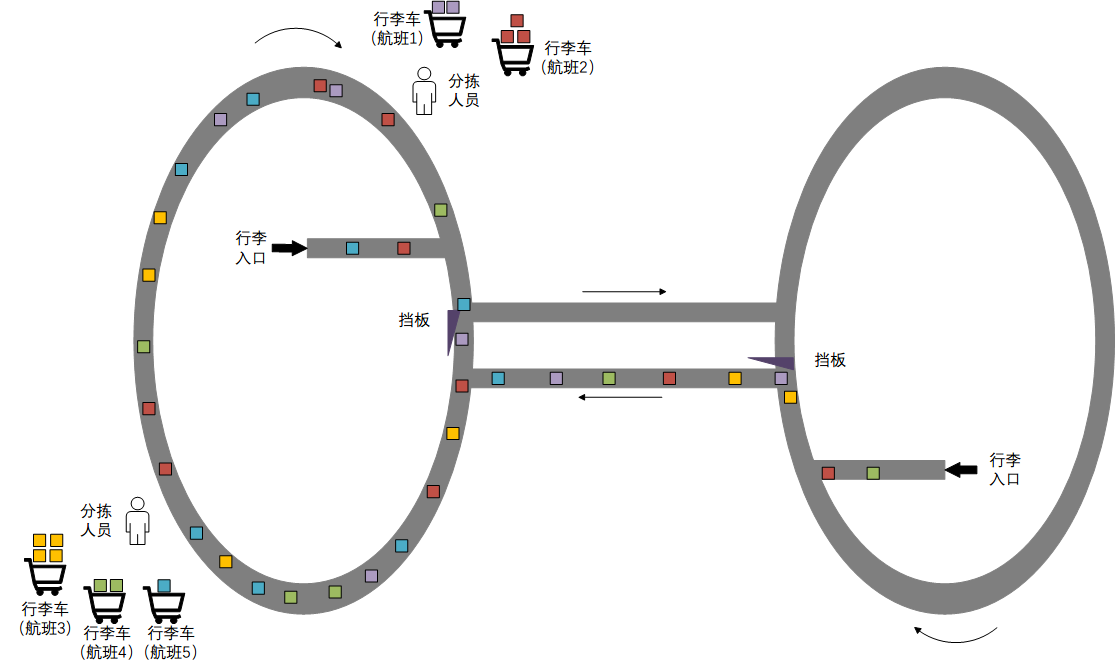


图 2 这里是图2

两个传送带均为椭圆形，中间由两条履带相连接。在环形传送带和履带相连的地方安装了一个挡板，通过人工控制挡板的开闭可以决定行李是通过履带传送到另一端还是继续留在本条传送带中。分拣人员分布在分检池的外侧，当有行李经过时工人需要翻看行李的标签信息，对行李的目的地进行人眼识别。如果行李目的地和自己负责的航班相匹配，则将行李放入行李集装箱或行李拖车上。如果行李目的地不匹配，则让行李继续在分拣池内继续循环传送。

因为乘客办理行李托运无统计特性，所以在分拣过程中不能预知多少行李会被放入拖车，也无法预知下一个行李的到达时间。工人必须换班轮流守候在分拣转盘附近，即使没有任何行李到达也需如此。如此粗放的行李分拣方式，给行李分拣人员带来了不便，也导致了不少行李的丢失和错放等情况的发生。

本项目的目的是研发个人行李实时监控及分拣辅助系统，通过显示屏等手段即时跟踪行李位置，方便行李分拣人员的及时分拣。通过本系统能够有效控制行李丢失事件，显著提高人工分拣效率，能够为上层人性化应用提供基础服务。

## 系统目标

根据上述业务分析，本项目拟研发航空行李人工分拣辅助系统，所要达到的目标有：

1. **向后兼容性**。现有的分拣设施基于条形码和打印码结合的混合方式，分拣人员一般使用人眼识别得出行李的相关信息。本项目试运行阶段必须兼容现有的分拣流程和方法，即除电子标签之外，还需要同时提供条形码和打印码两种方式。
2. **降低行李丢失率**。最基本的功能需求，要求实时准确监控和追踪每个行李的运送路径；要求遇到行李状态异常时，系统能够及时进行有效提示；要求遇到行李丢失时，系统能够准确定位和追踪到行李。
3. **虚拟化显示行李位置**。人工行李分拣方式造成了巨大的人力资源浪费。一方面，即使在没有行李被传送到是也需要有人在分拣盘附近等候；另一方面，当航班和行李数量比较多的时候，行李会很容易被拣错，使其成本升高。本系统拟采用可视化界面，虚拟化显示当前行李的位置，以增强工人的分拣效率，减小分拣错误。

## 术语定义

1. 北京首都国际机场

北京首都国际机场（ICAO：ZBAA，IATA：PEK），简称首都机场或北京机场，为[中华人民共和国](http://baike.baidu.com/view/1256.htm)和北京联外主要的国际机场，是目前[中国](http://baike.baidu.com/view/61891.htm)最繁忙的[民用机场](http://baike.baidu.com/view/700374.htm)，也是[中国国际航空公司](http://baike.baidu.com/view/125670.htm)的基地机场，其中1号航站楼于1980年启用。自2008年起，海南航空转场1号航站楼，使其成为海南航空集团（包括海南航空公司HU、大新华航空CN、天津航空公司GS、首都航空公司JD等）国内航班的专用航站楼。

1. 值机

值机是民航的一种工作，就是为旅客办理乘机手续（换登机牌、收运旅客的托运行李、安排旅客的座位）。旅客需要在候机楼的值机柜台办理值机手续。按照相关规定，所携带的行李超过一定重量或体积限制的，应作为托运行李托运。托运行李将被打上对应标签，并利用传送带传入航站楼底下的行李分拣池。

1. 分拣池

需要托运的行李通过传送带，最终被运送到分拣池，再由工人进行分拣装车，再运送到飞机上。图三所示为单个传送带的形状。

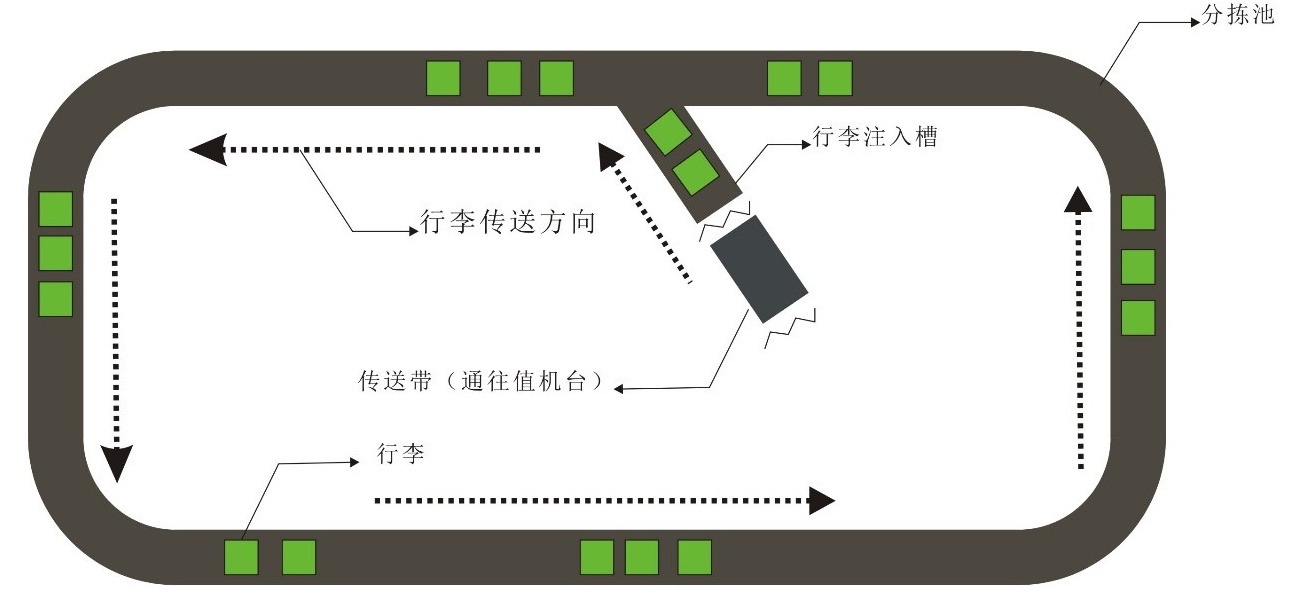
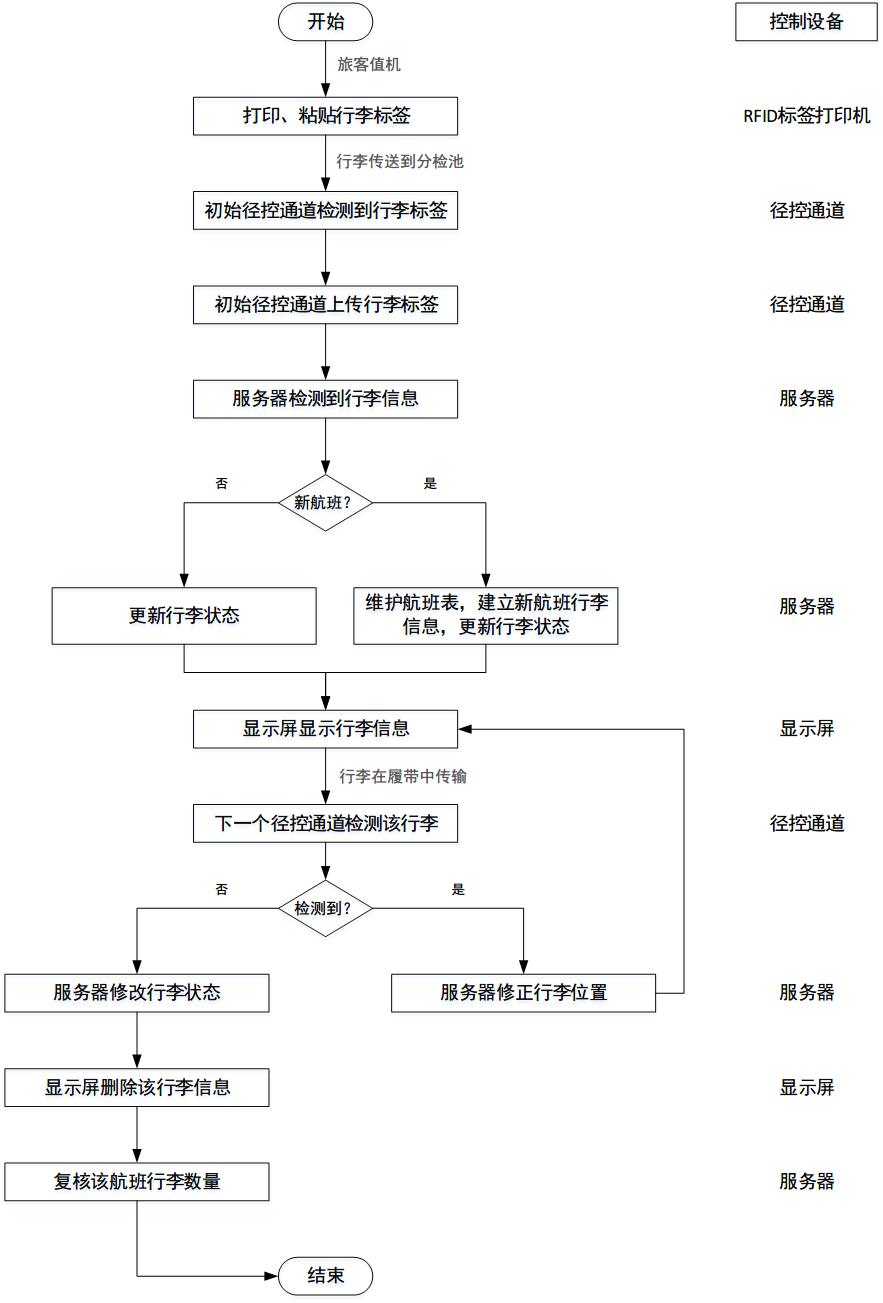


图 3 这里是图3

系统相关的分检池则如图2所示，由两个相同大小的传送带互相连接组成。工作人员通过控制挡板的开闭来决定是否让一条传送带上的行李传送到另一条传送带。当分检池运行时，分拣人员站在分检池外侧，拣出自己负责航班的行李，并放置于对应的行李车上。

## 参考文献

# 业务流程



# 系统方案

3.1 行李标签打印

3.2 径控通道检测

3.3 服务器检测行李信息

3.4 服务器维护航班表

3.5 服务器更新行李状态

3.6 显示屏显示行李信息

3.7 行李位置修正

3.8 航班行李数量复核

# 系统体系架构

本小节介绍了系统的总体架构，采用松耦合的基于消息总线的架构。

## 硬件资源

本系统所涉及到的硬软件资源包括：

|  |  |
| --- | --- |
| 资源名称 | 资源描述 |
| 径控通道 | 用于检测行李标签的通道 |
| 航班信息 | 航班号、航班行李总数等 |
| 分拣池 | 每个分拣池分为若干个分拣区 |
| 虚拟化显示器 | 用于显示分拣池中的行李当前位置 |

## 软件架构图

Message Bus

1. Flight Proxy

(2) Server

(4) Track Tunnel

(5) Virtualization

(3) Administrator

图4：软件体系架构图

上图显示了系统的体系架构，本系统设计的模块异常繁多，尤其是模块之间的消息传递跟是纷繁复杂。采用基于消息总线的松耦合机制具有良好的扩展性和异构性。本系统包含以下几个独立的消息组件：

1. 离港系统代理（Flight Proxy）：组件编号为M1。该模块定期对离岗信息进行查询，当查询到对本系统感兴趣的航班后，向消息总线推送个人行李信息。
2. 中心服务器(Center Server)：组件编号为M2。该模块维护了系统的所有状态信息。
3. 管理客户端（Administrator）：组件编号为M4。该模块可以显示当前系统维护的所有硬软件资源信息，并且可以对其进行关联操作，有UI界面。
4. 径控通道代理（Track Tunnel Proxy）：组件编号为M5。一个径控通道对应一个代理模块，每检测到一个行李之后代理会将行李的位置消息进行推送。
5. 虚拟化显示（Virtualization）：组件编号：M8。该模块连接虚拟化显示屏，当接收到行李的位置信息时，及时更新行李的状态。

## 部署图



图5：系统部署图

上图显示了系统的部署图，整个系统通过高速局域网相连，关键技术的计算采用集中式加局部分布式的方式。同时，通过Web服务器将必要的信息通过互联网向用户或第三方应用开放。

# 功能模块

本小节对各个消息组件进行了详细描述。

## 消息总线

消息总线是对消息推送和分配的组件，本系统采Apache ActiveMQ消息系统。Apache ActiveMQ纯Java实现的、开源的、符合JMS1.1规范的标准消息系统，也是目前最流行和功能最强大的消息系统。JMS是一种与厂商无关的 API，用来访问消息收发系统。它类似于 [JDBC](http://baike.baidu.com/view/25611.htm)(Java Database Connectivity)：这里，JDBC 是可以用来访问许多不同关系数据库的 API，而 JMS 则提供同样与厂商无关的访问方法，以访问消息收发服务。JMS 使您能够通过消息收发服务（有时称为消息中介[程序](http://baike.baidu.com/view/17674.htm)或路由器）从一个 JMS 客户机向另一个 JMS客户机发送消息。消息是 JMS 中的一种类型[对象](http://baike.baidu.com/view/2387.htm)，由两部分组成：报头和消息主体。报头由路由信息以及有关该消息的元数据组成。消息主体则携带着应用程序的数据或有效负载。根据有效负载的类型来划分，可以将消息分为几种类型，它们分别携带：简单文本 (TextMessage)、可序列化的对象 (ObjectMessage)、属性集合 (MapMessage)、字节流 (BytesMessage)、原始值流 (StreamMessage)，还有无有效负载的消息 (Message)。

JMS有以下元素组成: (1) JMS提供者, 连接面向消息中间件的，JMS接口的一个实现。提供者可以是Java平台的JMS实现，也可以是非Java平台的面向消息中间件的适配器。(2)　　JMS客户, 生产或消费基于消息的Java的应用程序或对象。(3) JMS生产者, 创建并发送消息的JMS客户。(4) JMS消费者, 接收消息的JMS客户。(5) JMS消息, 包括可以在JMS客户之间传递的数据的对象。 (6) JMS队列, 一个容纳那些被发送的等待阅读的消息的区域。队列暗示，这些消息将按照顺序发送。一旦一个消息被阅读，该消息将被从队列中移走。(7) 　JMS主题, 一种支持发送消息给多个订阅者的机制。

### 主题消息分类

主题消息是由某个模块发布，其他模块监听的消息，采用发布者/订阅者模型，是本项目最基础的通信方式，下表总结了最基本的主题消息。

#### 通用消息

##### 5.1.1.0.1. 消息概览

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | | 关联 | 发出端 | 接收端 | | 类别 | 描述 |
| 0 | **0.1.1** | **0.2.1** | 所有 | 所有 | 发布 | | 设备启动/关闭 |
| **0.1.2** | **0.2.2** | 所有 | 所有 | 发布 | | 信息确认 |
| **0.2.1** | **0.1.1** | 所有 | 所有 | 订阅 | | 设备启动/关闭 |
| **0.2.2** | **0.1.2** | 所有 | 所有 | 订阅 | | 信息确认 |

##### 5.1.1.0.2. 详细消息格式

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编号** | **名称** | **参数** | **类型** | **描述** |
| **0.1.1** | StatusSwitch设备启动/关闭 | source | int | 设备编号 |
| component | Int | 设备类型 |
| CommandType | bool | 开/关信息 |
| **0.1.2** | Confirm信息确认 | souce | int | 设备编号 |
| component | int | 设备类型 |
| success | bool | 指令执行情况 |

#### 航班代理相关

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 编号 | | 关联 | 发出端 | 接收端 | 类别 | 描述 |
| 1 | **1.1.1** | **2.2.4**  **6.2.1**  **8.2.1** | M1 | M2,M8 | 发布 | 新加个人行李 |
| **1.1.2** | **2.2.5**  **6.2.2**  **8.2.2** | M1 | M2,M8 | 发布 | 删除个人行李 |
| **1.1.3** | **2.2.1**  **3.2.8**  **4.2.3**  **6.2.3**  **8.2.3** | M1 | M2,M4,M8 | 发布 | 关闭航班 |
| **1.4.1** | **4.3.1** | M1 | M4 | 响应 | 响应资源发现 |

#### 服务器相关

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | | 关联 | 发出端 | 接收端 | | 类别 | 描述 |
| 2 | **2.2.1** | **1.1.3** | M1 | M2,M4,M8 | 订阅 | | 关闭航班 |
| **2.2.4** | **1.1.1** | M1 | M2,M8 | 订阅 | | 新加个人行李 |
| **2.2.5** | **1.1.2** | M1 | M2,M8 | 订阅 | | 删除个人行李 |
| **2.2.6** | **5.1.1** | M5 | M2,M8 | 订阅 | | 跟踪路径信息 |
| **2.2.9** | **8.1.1** | M8 | M2 | 订阅 | | 新航班的第一个行李入池 |
| **2.4.3** | **4.3.1** | M2 | M4 | 响应 | | 响应资源发现 |

#### 管理器相关

##### 消息概览 （未全)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | | 关联 | 发出端 | 接收端 | | 类别 | 描述 |
|  | **4.1.12** |  | M4 | M8 | 发布 | | 配置虚拟显示器 |
| **4.1.13** | **8.2.8** | M4 | M8 | 发布 | | 通知信息 |
| **4.2.5** | **1.1.3** | M1 | M2,M4,M8 | 订阅 | | 关闭航班 |
| **4.3.1** | **1.4.1**  **2.4.3**  **3.4.3**  **5.4.1**  **6.4.1**  **7.4.1**  **8.4.1** | M4 | M1,M2,M5,M8 | 查询 | | 资源发现 |

#### 跟踪通道相关

行李在整个分拣过程中所处的状态以及所在的位置可以由各个监控设备对该行李在不同时间的检测结果得到。换言之，根据各个监控设备对同一件行李的监控结果的组合，我们可以获知该行李所处的位置以及状态信息。下面以一件行李为例，对其在整个分拣过程所处的状态以及位置变化进行分析，捕获在该过程中各个监控设备所需发送以及接受的消息。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | | 关联 | 发出端 | 接收端 | 类别 | | 描述 |
| 5 | **5.1.1** | **2.2.6**  **8.2.4** | M5 | M2,M8 | | 发布 | 跟踪路径信息 |
| **5.4.1** | **4.3.1** | M5 | M4 | | 响应 | 响应资源发现 |

消息内容及格式：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编号** | **名称** | **参数** | **类型** | **描述** |
| **0.2.1** | 设备启动/关闭 | source | int | 设备编号 |
| component | int | 设备类型 |
| messageType | message | 打开或关闭设备 |
| **0.2.2** | 跟踪路径信息 | epc | int | RFID标签信息 |
| time | Time | 该行李的读取时间 |
| trackTunnelId | int | 该径控通道的编号 |
| location | Location | 该径控通道的位置信息 |
| RSSI | float | 读取该RFID标签的信号强度大小 |
| poolId | int | 分拣池编号 |

#### 虚拟化显示屏相关

在得到各个行李在不同监测设备中的状态之后，实时显示模块对这些信息进行整合，建立推测模型，获得各个行李的实时位置以及状态信息，然后将其显示出来。表（4）给出了实施显示模块的消息机制。

**表（4）：实时显示模块的消息机制。**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | | 关联 | 发出端 | 接收端 | 类别 | | 描述 |
| 8 | **8.1.1** | **2.2.9** | M8 | M2 | | 发布 | 新航班的第一个行李入池 |
| **8.2.1** | **1.1.1** | M1 | M8 | | 订阅 | 新加个人行李 |
| **8.2.2** | **1.1.2** | M1 | M8 | | 订阅 | 删除个人行李 |
| **8.2.3** | **1.1.3** | M1 | M8 | | 订阅 | 航班关闭 |
| **8.2.5** | **3.1.5** | M3 | M8 | | 订阅 | 识别到应该提取的行李 |
| **8.2.8** | **4.1.13** | M4 | M8 | | 订阅 | 通知信息 |
| **8.4.1** | **4.3.1** | M8 | M4 | | 响应 | 资源发现 |

消息内容及格式：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **编号** | **名称** | **参数** | **类型** | **描述** |
| **0.3.1** | 设备启动/关闭 | source | int | 设备编号 |
| component | int | 设备类型 |
| messageType | message | 打开或关闭设备 |
| **0.3.2** | 航班入池 | epc | long | RFID标签信息 |
| time | Time | 该行李的读取时间 |
| trackTunnelId | int | 该径控通道的编号 |
| location | Location | 该径控通道的位置信息 |
| poolId | int | 分拣池编号 |

## 离岗系统代理

离岗系统代理定期对离岗系统进行查询，当有新的行李增加且该行李的航班号符合系统的期望时，提取该行李的信息并推送给消息总线。

## 中心服务器组件

## 径控通道

首先对行李在整个分拣过程中可能处于的位置进行分析，得到以下四种位置关系：上游传送带（行李注入槽）；分拣池；地面；行李拖车。其中，根据行李被放置在地面的原因不同，我们将地面分为两类：

1. 地面A：行李遭受撞击从分拣池滑落；
2. 地面B：为节省时间，人工将通过腕带提示器检测的行李放置在地面（复核通道与分拣池之间，尚未通过复核通道检测）；



**图(1)：行李分拣池平面示意图。**

接着对行李在这几种位置之间的转化条件进行分析，得到以下转化关系：

**表(1)：行李所处位置变化触发关系。**

|  |  |
| --- | --- |
| 位置转化 | 触发条件 |
| 上游传送带🡪分拣池 | R1径控通道检测到该行李首次进入分拣池。 |
| 分拣池🡪行李拖车 | 复核通道检测并核实该行李已成功分类。 |
| 分拣池🡪地面 | 行李即未通过径控通道的二次监测，又未通过复核通道的检测。 |
| 地面🡪分拣池 | 径控通道在一段时间之后重新检测到某件行李。 |
| 分拣池🡪行李拖车 | 行李成功通过复核通道的检验； |

对分拣池🡪地面这一转化过程的原因分析如下：

* 行李由于遭受撞击或其他原因从分拣池上滑落到地面；
* 行李已通过腕带提示器检测，但尚未通过复核通道。

对地面🡪分拣池这一转化过程的原因分析如下：

* 人工将滑落行李重新放置在分拣池中；
* 腕带提示器检测到该行李不属于该航班，故人工将该行李重新放回分拣池中。



**图（2）：行李在各个位置之间的转化关系示意图。**

由以上分析可知，**一**件行李的成功分拣可被认为是该行李首先通过径控通道的监测之后，又通过了所属复核通道的检测。为保障每件行李能够被成功分拣，我们需要对每件行李在整个分拣过程中所处的位置进行实时监控，并显示其所处位置信息，以提高分拣员的分拣效率。另一方面，由于行李在分拣池上所处的状态并不稳定，例如：行李由于遭受其他行李的撞击而从分拣池中滑落到地面，或者在经过分拣口时被人工调整位置。因此如何利用径控通道来对行李在各个位置进行准确定位，并对行李滑落等突发事件做出判断是该节主要讨论的问题。

整个分拣系统共安装三台径控通道，若干腕带阅读器（将阅读器归为两类，一类为分拣口1，另一类为分拣口2）和两台复核通道，分别标记为*R1*，*R2*，*R3*，*H1*，*H2*，*D1*，*D2*。各个径控通道和复核通道所处的位置如图(1)所示。我们首先分析各台检测仪器对一个行李的检测状态的可能组合。

**表(2)：七台检测仪器状态组合。**

|  |  |
| --- | --- |
| 判断条件  *R1R2R3H1H2D1D2* | 原因 |
| 1 0 0 0 0 0 0 | 行李在径控通道*R1*与*R2*之间滑落； |
| 1 1 0 0 0 0 0 | 行李在径控通道*R2*与*R3*之间发生滑落； |
| 1 1 0 1 0 0 0 | 行李在分拣出口*A1*被人工分拣，但尚未通过复核通道； |
| 1 1 0 1 0 1 0 | 行李在分拣出口*A1*被人工分拣； |
| 1 1 1 0 0 0 0 | 行李处于分拣池中，待分拣； |
| 0 1 1 0 0 0 0 | 行李在径控通道R3与R1之间发生滑落； |
| 0 1 1 0 1 0 0 | 行李在分拣出口*A2*被人工分拣，但尚未通过复核通道； |
| 0 1 1 0 1 0 1 | 行李在分拣出口*A2*被人工分拣； |
| 其他 | 不可能出现情况； |

## 虚拟化显示

在本课题中，需要将各台监控设备信息汇集，然后对行李所处位置进行实时仿真并显示。方便起见，假设在整个分拣过程中，传送带都在匀速运动。不妨将传送带抽象为一个四个顶角都为圆角的矩形，如图（3）所示。通过对以上各个仪器监控状态的分析，我们将行李在整个分拣过程所处状态分为三类：



**图（3）：传送带抽象表征。**

* 待分拣状态：行李在传送带上匀速运动，等待被分拣；
* 已分拣状态：行李已通过复核通道检验，成功分类；
* 丢失状态：行李由于滑落或在分拣出口被人工分拣但尚未未通过复核通道；

待分拣状态下，行李按照时间序列在传送带上匀速运动，系统计算各件行李的位置，并进行显示。易知每件行李在传送带上的位置可以通过行李在传送带上的搁置时间以及传送带的运行速度计算得到。当行李被下一个径控通道检测到时，程序首先对该行李的位置进行判断，调整该行李的位置。即，将行李从所处旧位置处删除，在新位置处进行显示。当系统检测到某件行李已被成功分拣，则将其从显示屏上去除。当行李处于丢失状态时，我们对其处理如下：首先根据各个检测器的检测状态的组合状态得出该行李的丢失原因以及丢失位置，然后将该行李从显示屏去除，将显示屏上行李累计丢失数量+1。最后虚拟显示该件丢失行李可能所处的位置。

### 行李显示模块要素：

本小节描述行李显示模块的设计要求以及设计要素。在此基础上，给出整个显示模块的设计框架，并分析该框架的可行性和健壮性。

分析整个行李分拣流程，我们将行李显示模块的主要作用概括为一下五个方面：

* 显示各件行李与其对应航班信息；
* 实时显示行李所处位置进行，协助分拣员进行行李分拣，提高行李分拣效率；
* 报告行李丢失事件，并给出丢失的行李可能出现的地点，协助分拣员寻找丢失行李；
* 显示行李的统计信息，对已分拣，待分拣，以及丢失的行李统计数据进行实时动态显示，提高分拣效率；
* 当行李运行到距离对应分拣口一定距离之内时，对该行李进行高亮显示，提醒分拣员进行分拣。如图(5)所示。



**图（5）：行李状态显示变化。**



**图（6）：显示界面设计。**

# 附录