基于无源超高频 RFID 技术的仓库管理系统解决方案

1.1 工作过程

射频识别技术是通过线圈式天线阅读器对射频识别标签（Passive Integrated Transponder, PIT）所发送信号进行接收以实现对标记个体的监测。双通道读写器通过天线发送一定频率的超高频信号，当电子标签进入读写器辐射场的有效工作区域内，发生电磁反向散射耦合，电子标签被激活，将自身存储的信息通过内置天线发送出去，读写器通过接受天线接收到电子标签发送来的信号，对标签芯片中的存储器进行读/写操作，数据经过读写器解调和编码后送到主机并通过主机发送至 Iot 的数据存储，并在云端的 LinkDevelop 中进行处理，最后上行回传图 1-1 和图 1-2 就是我们模拟场景（实验室）的读写器和天线。

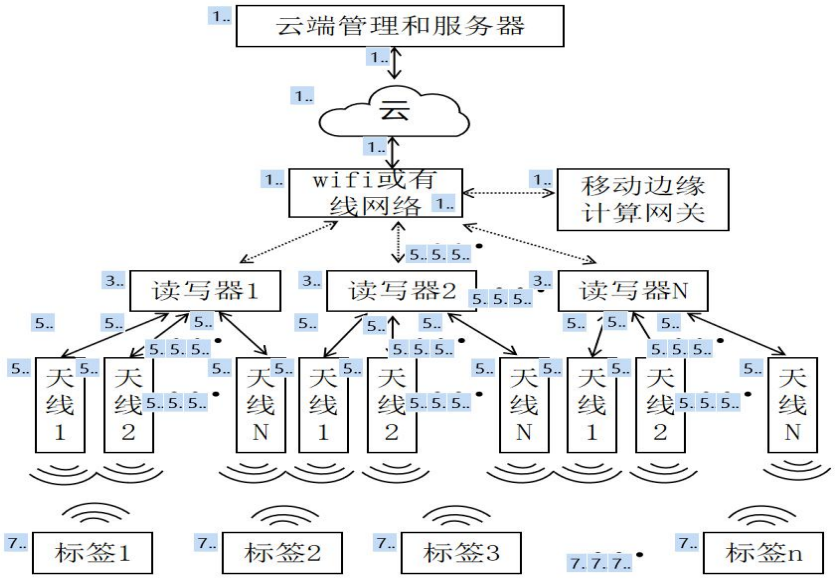
图 1-1

图 1-2

射频识别由阅读器和天线线圈组成。两者之间的信号和能量传输采用磁场耦合的方式，射频识别监测系统的操作简易和轻便，方便野外监测，RFID 标签在经过天线线圈时，阅读器会记录下标签独特的编号及记录的时间。射频识别系统的供电方式可由蓄电池、太阳能或是交流电转直流电，可供电压在 7~20 V，国内外一般采用 12V 左右;其中，RFID 标签能量的供给由天线阅读器提供，实现了标签的小型化，提高了标签的使用寿命。

1.2 工作原理

我们的系统由云端管理和服务器、移动边缘计算网关、超高频 RFID 读写器网络、天线、标签等部分组成，关系如下：云端管理和服务器通过 wifi 或宽带连接移动边缘计算网关来控制读写器网络。每个读写器又分别驱动多根天线去读取标签。



在定位算法中，首先采用 K nearest neighbour 算法三步计算得出坐标。第一步计算其欧几里得距离，第二步计算 k 个参考点的系数，第三步计算目标横纵坐标。而其后再通过卡尔曼滤波器处理异常坐标以及突发情况。

一、K nearest neighbour算法

1.测算欧几里得距离 $E_n = \sqrt{\sum_{m=1}^{4}(T_m - R_{n,m})^2}$

2.计算K个参考点的系数 $w'_k = \frac{1/E'^2_k}{\sum_{k=1}^{K} 1/E'^2_k}$

3.计算目标标签横纵坐标 $(x,y) = \sum_{k=1}^{K} w'_k(x_k, y_k)$

二、Kalman Filter （卡尔曼滤波器）

1.计算估计值 $X[i+1]=(1-K[i+1])x[i]+K[i+1]^*z[i+1]$

2.计算卡尔曼增益系数 $K[i+1]=(P[i]+Q)/(P[i]+Q+R)$

3.计算估计值方差 $P[i+1]=(1-K[i+1])P[i]$

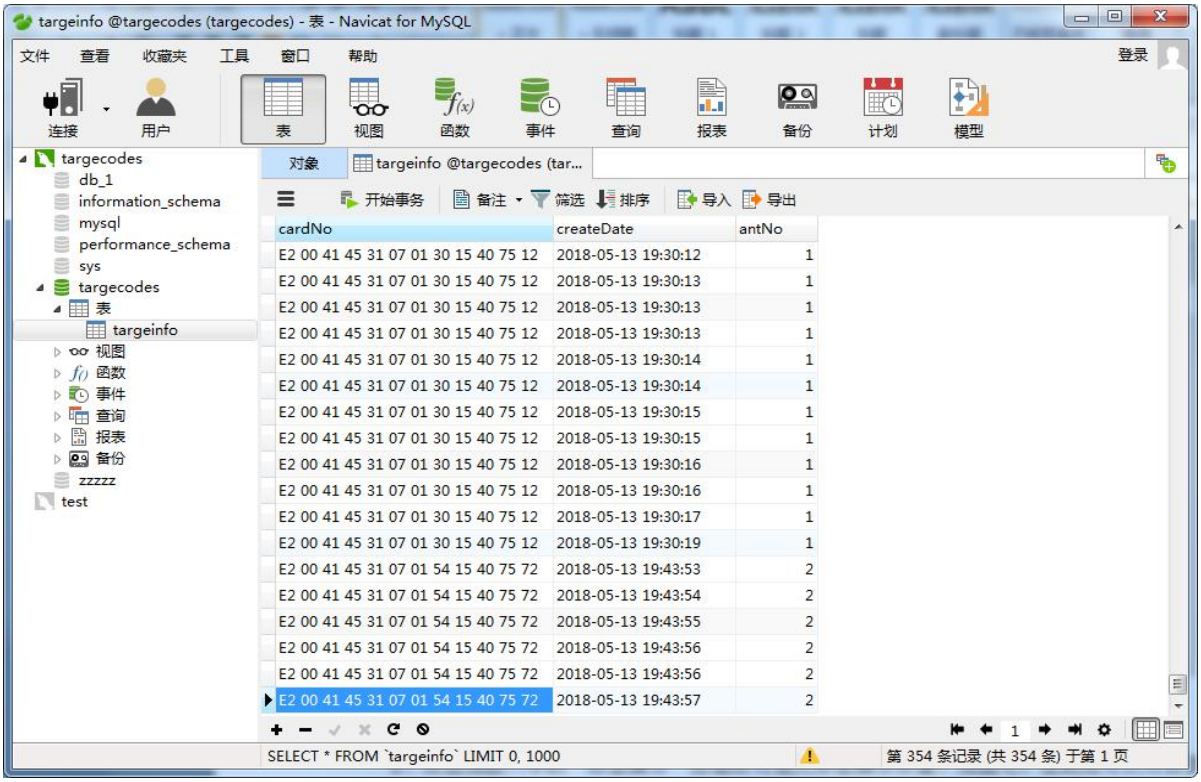
边缘计算网关控制读写器的软件界面，可以支持由多台读写器组成的读写器网络。而云端功能包括出入库扫描，库存管理，人员或物品的信息修改和查询以及轨迹追踪。

边缘计算网关控制读写器

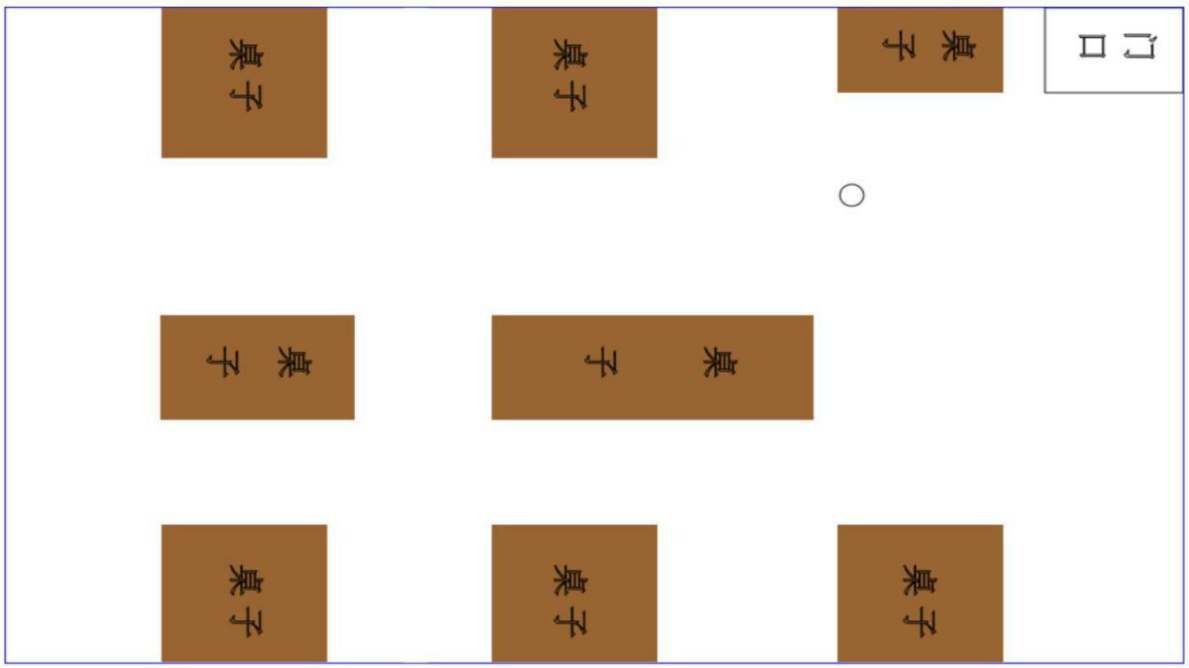
1.2.1 系统的环节组成。

出入库扫描

库存管理

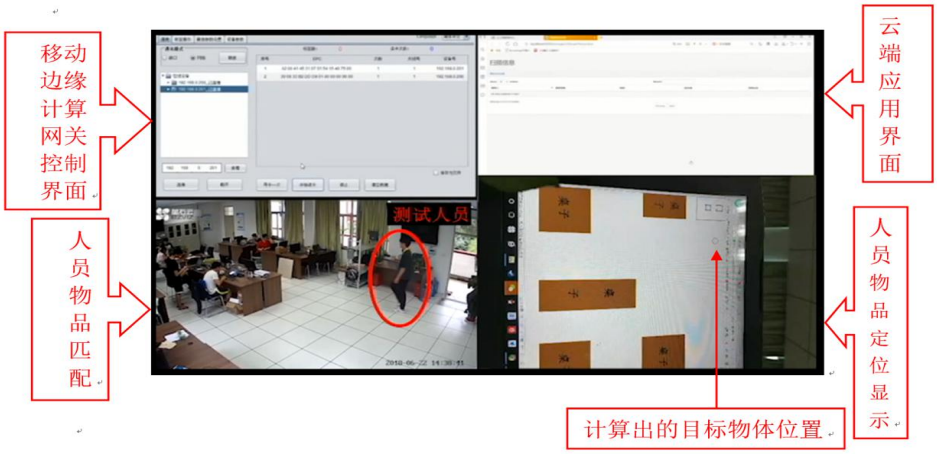


数据库信息插入



轨迹追踪

我们在实验室中模拟仓库场景并做的测试。测试人员携带配有标签的货物进入之后，移动边缘计算网关控制读写器读取标签信息并回传云端，云端做数据存储和定位显示。



1.2.2 上下行控制

接收 LinkDevelop 平台下行的指令，并且在收到固定指令后将本地数据上行传到 LinkDevelop 平台中，将最新的状态显示在 LinkDevelop 平台上，如图 1.2.6 所示。

```
JS connectLD.js
var cardNo,createDate,antNo,rssi,host,type;

device.on('connect', () => {
  // 连接成功
  console.log('connect successfully');
  device.serve('property/set', params => {// 接收下行指令，存在params中
    console.log('receieve params:', params);
    console.log('post props:', params.cardNo);
    var temp;
    temp = new Number(params.cardNo);
    if(temp>0){// 判断是否要上行数据
      console.log("准备上传");
      var i = 0;// 记录目前上传的数据位置
      console.log(result);
      while(i<=temp-1){
        cardNo = result[i].cardNo;
        createDate = result[i].createDate.toString();
        antNo = result[i].antNo.toString();
        rssi = result[i].rssi.toString();
        host = result[i].host;
        type = result[i].type;
        console.log(cardNo,createDate,antNo,rssi,host,type);
        i++;
        var val = {
          cardNo:cardNo,
          createDate:createDate,
          antNo:antNo,
          rssi:rssi,
          host:host,
          type:type
        }
        device.postProps(val);
        console.log("上传成功");
        if(i>=result.length){
          console.log("数据库没有更多数据可以上传！上传结束。");
          break;
        }
      }
    }
  });
});
```

图 1.2.6

1.2.3 后端代码

下图 1.2.7 为自动读取 T 的 js 代码。

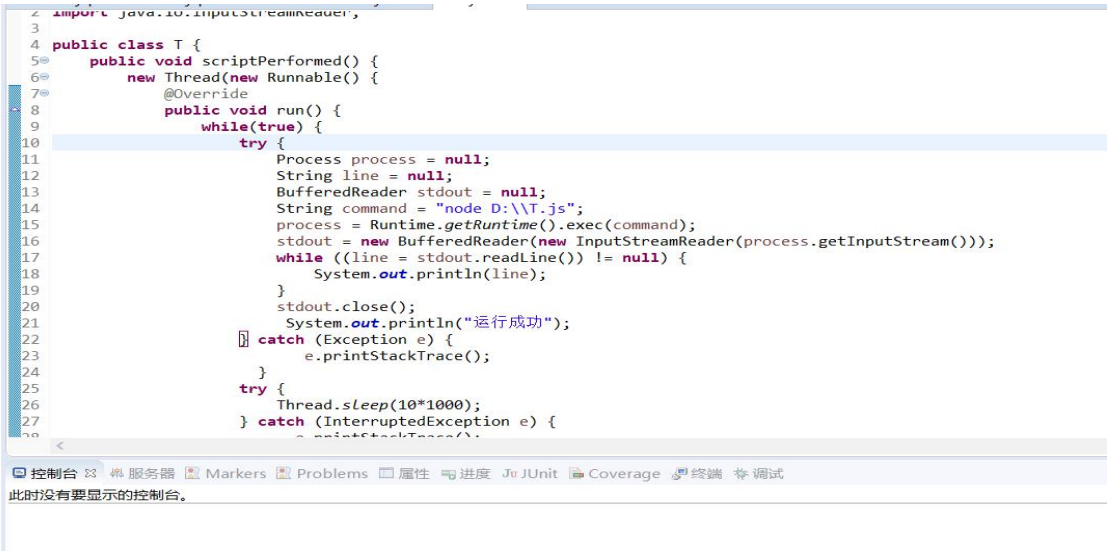


图 1.2.7

1.3 感知层技术

无源超高频 RFID 标签工作在 860 兆赫兹～960 兆赫兹，读写器可以实现对标签的高速读取，符合 ISO-18000-6C（EPC G2）协议；具有对标签读取敏感度判断算法，稳定读取距离可达 15 米；卓越的多标签识别能力，每秒钟能读取 200 张以上的标签；支持 RSSI 相对强弱指示。标签附着在物品上或嵌入工作人员的工作牌中，读写器置于基站出入口或给保护区管理员配置手持读写器，可以实现对范围中的物品进行实时的全方位的盘点、监测、管理等功能。

1.3 传输层技术

读写器使用 ISO-18000-6C（EPC G2）协议读取标签数据。读写器可以通过有线网、wifi、3G/4G 或蓝牙等方式把读回的标签数据回传到移动边缘计算网关（手机、平板、电脑等）。移动边缘计算网关通过 TCP 协议把数据回传到 LinkDevelop。

1.4 控制层技术

LinkDevelop 控制移动边缘计算网关（手机、平板、电脑等）发送指令控制读写器读取标签 EPC 编号、ID 号、RSSI 信号强度等数据，并将获取的标签数据存入数据存储中，在云上实现人员与物品、物品的出入状况的自动显示，实现物品的自动计数和人员物体进出的实时登记。

1.5 软件开发技术

移动边缘计算网关（手机、平板、电脑等）上，在 Eclipse 集成开发环境中使用 java 语言设计读写器控制程序，发送指令给读写器，以获取标签 EPC 编号和 RSSI 等数据信息。云端服务器上，在 IntelliJ IDEA 中运用 JavaScript 语言以及 Data V 进行网页界面设计，实现了从页面实时检测携带标签的工作人员和物品进出监测区域的管理，在 Navicat for MySQL 中运用 MySQL 数据库实现标签数据的云存取功能。

1.6 云应用

云端应用实现活动范围的有效监测。物品携带标签进入基站范围内，读写器会辨别某标签是否有历史记录，并给物品一个进入的标记，待物品拿走后，系统会将物品与该标签解绑，并持续更新物品信息，管理者能够随时查询被检测者的各种信息，如位置，进出时间等。

物品带着标签离开监测区，读写器读取已被标记过的物品信息，并将他们绑定，能在前端网页上显示某个时间某物品离开。

1.7 可视化应用

该系统将物品及工作人员的信息做了可视化应用，管理员可以在后台管理的界面中对物品的名称、类别、物品位置、出入时间等信息进行增删改查。