实时校车

软件系统设计文档

中国矿业大学小煤球队

队员：吴栋、宋磊、张镇洲

指导老师：余接情

**目录**

1. **文档简介** ................................................................3

1.1文档目的与范围........................................................3

1.2读者对象..............................................................3

1.3参考文献..............................................................3

1. **软件简介** ................................................................5

2.1软件设计背景...........................................................5

2.2软件需求分析...........................................................5

**3.开发、测试与运行环境**.....................................................7

3.1运行环境...............................................................7

3.2开发环境...............................................................7

3.3开发平台简介...........................................................7

**4.软件系统结构分析**.........................................................9

**5.功能模块划分及简介**......................................................11

5.1登录/注册模块........................................................11

5.2定位与地图模块.......................................................15

5.3上传共享位置数据信息.................................................16

5.4校车实时位置计算模块.................................................17

**6.数据库设计概述**..........................................................20

6.1数据库环境说明.......................................................20

6.2表汇总和表设计.......................................................21

**7.用户界面设计概述**........................................................23

7.1登陆/注册界面........................................................23

7.2主界面...............................................................24

7.3校车实时信息展现.....................................................25

**1 文档简介**

1.1文档目的与范围

本软件设计文档编写的目的是说明程序模块的设计考虑，包括程序描述、输入输出、算法和流程逻辑等，为软件程序编写和系统维护提供基础。

1.2读者对象

本文档的预期读者为系统设计人员、软件开发人员、软件测试人员和项目评审人员。

1.3参考文献

[1]王进. 实时公交查询系统的优化设计和实现[D]. 北京邮电大学, 2013.

[2]杨荫. 提醒平台[D]. 华南理工大学, 2013.

[3]李紫阳. 基于Android平台的视频客户端的设计与实现[D]. 西安电子科技大学, 2014.

[4]庄丽. 企业订单管理系统的设计与实现[D]. 吉林大学, 2012.

[5]马志祥. 基于J2EE的海量数据虚拟存储管理平台的设计与实现[D]. 电子科技大学, 2014.

[6]唐芳. 实时公交信息查询系统无线移动客户端的设计与实现[D]. 北京邮电大学, 2012.

[9]郝晓龙. 基于B/S架构的库存管理系统的分析与设计[D]. 北京邮电大学, 2013.

[8]刘宁. 基于Android平台的LBS应用设计与实现[D]. 西安电子科技大学, 2014.

[9]陈武. 基于XMPP的企业级即时通信系统的研究与实现[D]. 武汉理工大学, 2009.

[10]王永青. 基于PHP的物联网用户登录与数据解析的设计与实现[D]. 武汉理工大学, 2012.

[11]张琦. 基于PHP的实验室公共服务系统的设计与实现[D]. 西安电子科技大学, 2014.

[12]陈盛. 基于VB的企业人事管理系统的设计与实现[D]. 电子科技大学, 2012.

[13]任俊国. 多数据源可控查询技术的研究与应用[D]. 山东科技大学, 2012.

[14]满勤. 基于J2EE的B2C电子商务系统的设计与实现[D]. 电子科技大学, 2013.

[15]白云鹏. 实时公交换乘系统的研究与开发[D]. 电子科技大学, 2013.

[16]郭宏昌. 基于Android手机的实时公交查询系统设计与实现[D]. 西安市交通运输管理处, 2012.

[17]宫文浩. 基于Android平台的火车公交出行软件设计[D]. 广西师范大学, 2013.

[18] Lei Lin. An Android Smartphone Application for Collecting, Sharing and Predicting 2 Border Crossing Wait Tim [D]. the State University of New York, 2014.

[19] Andres Monzon,Sara Hernandez,Rocio Cascajo. Quality of Bus Services Performance: Benefits of Real Time Passenger Information Systems[J]. Transport and Telecommunication,2013,14(2).

[20]Md. Matiur Rahman,S.C. Wirasinghe,Lina Kattan. Users' views on current and future real‐time bus information systems[J]. J. Adv. Transp.,2013,47(3).

[21] CHEN Peiji. Real-time bus passenger flow statistics scheme based on light-sensitive wireless sensor network[A]. 中国自动化学会控制理论专业委员会、中国系统工程学会.第35届中国控制会议论文集（F）[C].2016:5.

[22] 黄洁. 智能公交系统乘客满意度和行为意愿分析[D].华中科技大学,2009.

[23] Lin, L., Q. Wang and A. W. Sadek. Border Crossing Delay Prediction Using Transient Multi- 2 server Queueing Models. Transportation Research Part A: Policy and Practice, Vol. 64, pp. 65- 3 91, 2014.

[24] Steinfeld, A., J. Zimmerman, A. Tomasic, D. Yoo, and R. D. Aziz. Mobile Transit Information from Universal Design and Crowdsourcing. Transportation Research Record: Journal of the Transportation Research Board, Vol. 2217, No. 1, pp. 95-102, 2011.

**2 软件简介**

2.1软件设计背景与意义

众多高校都有多个校区，学生往返不同校区大多会选择搭乘校车。校车准确的抵达和出发时间是每位师生最为关注的信息，而运行时刻表是大多高校了解校车运营信息的主要途径。运行时刻表为每位师生提供了一份详细的计划运营时刻信息。受实际情况约束，计划运营时刻信息往往不够准确，会出现晚点现象。以中国矿业大学为例，学校在安排车辆时，往往会在同一时间段内投入若干辆车，每辆车从一个校区到另一个校区后紧接着会再返回原先那个校区，如此循环。当路况良好时，校车基本按照时刻表时间运营，但路况拥堵时，校车就很难遵循时刻表上的时间来运营。此外，出于运营成本考虑，中国矿大在闲时时段未设置明确的时刻表，即两校车发车间隔为半个小时左右，各站点抵达时间不定。实际运营过程中，校车通常会比预期时间晚20~40分钟，而如此长时间的莫名等待对于大部分人而言是难以忍受的，尤其是在北方寒冷的夜晚。为此，借助当前科技手段准确而方便地掌握当前校车的准确位置信息及抵达时间显得尤为重要。

本作品将借助大数据手段及手机APP为广大师生的出行提供实时校车信息服务。在本APP中，每位用户既是信息的提供者也是信息的享用者，需要各位用户获取校车实时位置信息的同时贡献自己的位置信息，通过挖掘用户的实时位置信息获取当前的校车位置及预计抵达时间。通过本APP，用户可以实时查询校车的实时运营信息，包括校车的运营路线、发车时间、实时位置信息、抵达某一站点的时间预计。

2.2软件需求分析

系统的需求主要有：1）校车时刻表及路线查询；2）校车实时位置信息查询；3）抵达时间预计；4）地图服务。具体地，用户在乘坐校车的时候使用手机定位系统，将自己的位置信息上传至云数据库；等车用户则从云数据库中获取这些共享的数据，通过聚类分析后得到当前正在运行的一辆或者多辆校车的实时位置；最后，根据预测分析模型来预测校车到达某一位置的时间。

**3 软件开发、测试与运行环境**

3.1 运行环境

|  |  |
| --- | --- |
| 运行平台 | Android智能手机 |
| Android版本 | 5.0以上 |

3.2 开发环境

|  |  |
| --- | --- |
| 操作系统 | Windows10 |
| 开发语言 | Java |
| JDK版本 | Jdk 1.8 |
| 开发平台 | Android studio2.3.3 |
| 服务器 | AVOS Cloud |

3.3开发平台简介

Android Studio开发平台

3.3.1 简介

Android Studio 是一个Android集成开发工具，基于[IntelliJ IDEA](http://baike.baidu.com/item/IntelliJ%20IDEA). 类似 [Eclipse](http://baike.baidu.com/item/Eclipse/61703) [ADT](http://baike.baidu.com/item/ADT)，Android Studio 提供了集成的 Android 开发工具用于开发和调试。

在[IDEA](http://baike.baidu.com/item/IDEA)的基础上，Android Studio 提供：

1.基于[Gradle](http://baike.baidu.com/item/Gradle)的构建支持

2.Android 专属的重构和快速修复

3.提示工具以捕获性能、可用性、版本兼容性等问题

4.支持[ProGuard](http://baike.baidu.com/item/ProGuard) 和应用签名

5.基于模板的向导来生成常用的 Android 应用设计和组件

6.功能强大的布局编辑器，可以让你拖拉 UI 控件并进行效果预览

3.3.2优势

Android studio相比较于其他开发平台的优点有：

1、Google推出的

2、速度更快

3、UI更漂亮

4、更加智能

5、整合了Gradle构建工具

6、强大的UI编辑器

7、内置终端

8、更完善的插件系统

9、完美整合版本控制系统

3.3.3 工程结构

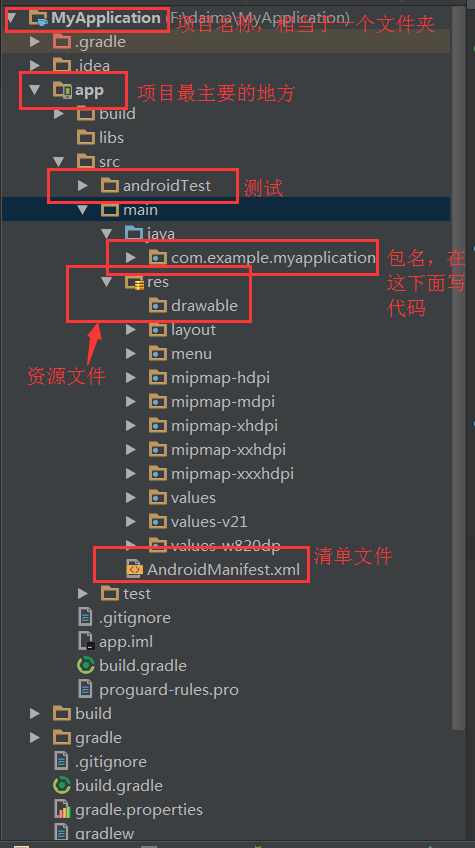
****

图1 Android studio工程结构

**4 软件设计**

4.1体系结构

本系统采用客户端/服务器（Client/Server）模式，客户端用Java 语言开发，在 Android 平台上运行。服务器端使用LeanCloud一站式云服务器，调用LeanCloud提供的数据服务接口进行数据访问，校车位置信息来源于用户的手机定位信息。用户的定位信息来源于高德地图API。客户端通过 3G、4G或者WiFi网络访问服务器端查询接口，服务器端将查询到的实时数据以List格式返回给客户端，系统的架构如图3所示

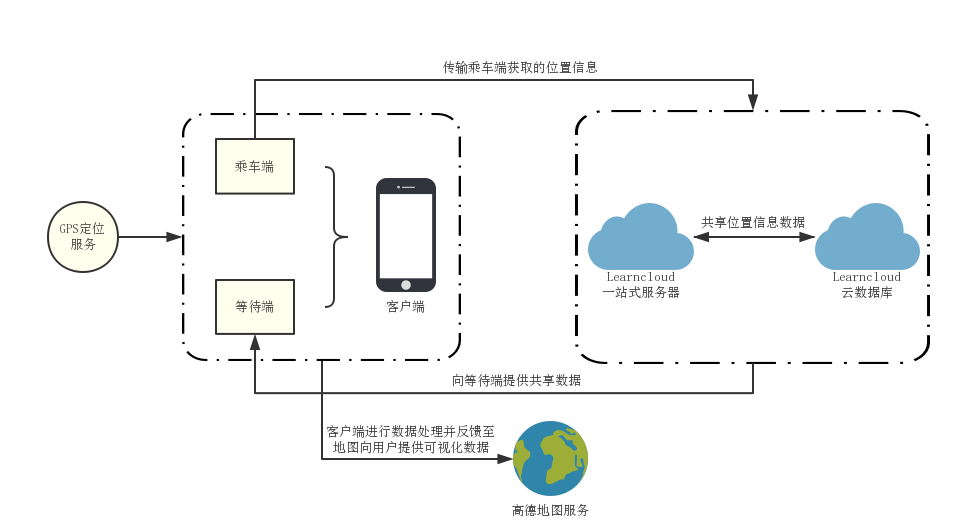


图2 系统架构图

表 系统说明

|  |  |
| --- | --- |
| **客户端/服务器** | **说明** |
| 客户端 | 实时校车系统客户端指手机客户端，客户端完成软件和相关查询页面的需求，与云服务器进行交互，发起查询请求，通过与第三方的地图服务器交互将结果向用户展示。其中手机客户端需要支持定位功能和上网功能，这样才能向服务器实时汇报地理位置信息。手机端同时也是数据处理端，在获取到全部的共享数据后做聚类分析，完成校车的实时位置定位和校车运行状态、到站时间预测等功能。  在本项目中采用小米4s作为本系统的实验机，Android版本为7.0。 |
| 服务器端 | 使用leancloud服务器来存储与中转数据，在这里可以汇总当前在线用户所共享的所有位置信息数据，并向发起数据访问的客户端传送数据。在这里既可以作为数据库来存储数据，也可以作为服务器处理客户端的访问请求。 |

4.2总体设计

结合以上需求，系统设计如下：



图3 系统功能图

**5 软件功能模块划分及简介**

5.1登录/注册模块

该模块包括用户注册、登录等。

界面有用户名和密码输入框，输入框识别用户输入的信息，在用户点击登录按钮时，系统首先判断用户名、密码输入是否为空，如果某项为空，则弹窗提示并要求重新输入；随后以用户名向数据库查询数据，得到数据库中对应的密码，如没有返回数据则表示没有该用户，弹窗提示用户输入的用户名有误应重新输入，若有返回数据，则把返回的密码与用户输入的密码比较，相同则用户名密码输入成功，则登录成功，跳转到主界面并返回用户名，否则提示用户输入密码有误并进行重新输入。

用户点击注册按钮后，弹出注册对话框，提示用户进行注册。用户注册需要输入用户名、密码及确认密码。用户在进行注册时，客户端首先判断用户输入手否由漏项，若某项为空，则弹出对话框提示。其次，判断用户两次输入密码是否一致。然后以用户名向数据库中查询用户名相同的数据，若有返回的数据则表示该用户名已经使用，提示用户重新输入新的用户名，最后则向数据库中写入用户名和密码，并退出注册界面转至登录界面提示用户登录。

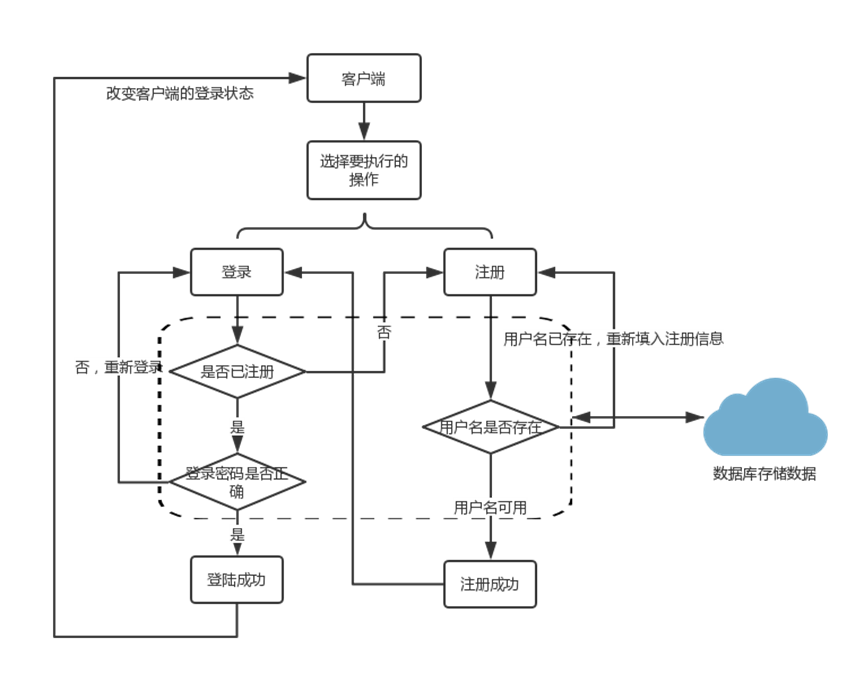
5.1.1登陆/注册界面交互流程：

图4 用户注册/登录交互流程图

5.1.2实现方法：

1、登录界面

①设置全局变量username（用户名）、password（密码）

②添加文本框改变监听事件，监听用户输入内容,在用户输入完毕后将输入的内容赋值给相应的全局变量，方便在其他模块使用这些用户数据。  
 public void afterTextChanged (Editable s) {  
 username = user\_text.getText().toString();*//将用户名文本框 的内容赋值给username* password = pass\_text.getText().toString();//将密码文本框的内容赋值给password}

③根据获取的用户名username和密码password可以来判断输入是否为空，如果用户名和密码未输入，则弹窗提示“用户名或密码未输入，请检查后再登录！”。

if (TextUtils.*isEmpty*(username) || TextUtils.*isEmpty*(password)) {

//弹窗提醒

}

④登录功能是使用数据库查询接口，这里是使用用户输入的用户名向数据库的“UserInfo”表中查询数据“password”，leancloud的查询接口使用需要先实例化要查询的表，然后执行查询条件：

AVQuery<AVObject> avQuery = new AVQuery<>("UserInfo");//实例化表  
avQuery.whereEqualTo("username", username);//查询

leancloud会将查询条件所在的行的全部数据返回，在查询的回调函数中然后根据需要提取相应的数据“password”，与用户输入的密码对比，如相同则登录通过，登录成功并跳转至主界面：

avQuery.findInBackground(new FindCallback<AVObject>() {//回调函数  
 public void done(List<AVObject> list, AVException e) {//查询后的执行函数  
 if (有查询结果) {  
 String get\_pass = list.get(0).getString("password");  
 if (密码相等) {  
 *//账号密码匹配，登录成功，跳转界面并传值*

}}}）；

2、注册界面

①注册界面同登录界面一样需要添加文本监听事件，监听用户的输入，设置全局变量reUserName(注册用户名)、rePassword(注册用户密码)、reRePassword(吃输入密码)，并将输入内容赋值给相应的全局变量。

public void afterTextChanged (Editable s) {  
 reUserName = reUserText.getText().toString();*//将用户名文本框的内容赋值给reUserName*  
 rePassword = rePassText.getText().toString();*//将密码文本框的内容赋值给rePassword*  
 reRePassword=reRePassText.getText().toString();*//密码文本框的内容赋值给reRePassword*  
}

②根据获取的用户名reUserName和密码rePassword以及reRePassword可以来判断输入是否为空，如果其中某项未输入，则弹窗提示“用户名或密码未输入，请重新填写！”；同时检查两次输入密码rePassword以及reRePassword是否完全一致，若一致则进行下一项判断，若不一致则弹窗提示“两次输入密码不一致，请重新输入”

public void onClick(View v){  
 if(TextUtils.isEmpty(reUserName)||TextUtils.isEmpty(rePassword)||TextUtils.isEmpty(reRePassword)){

*//弹窗提示*

}else if(!rePassword.equals(reRePassword){

*//弹窗提示*

}

③注册功能是使用数据库的写入接口，与查询相同，需要先实例化要操作的表，然后执行添加数据

AVObject User = new AVObject("UserInfo");  
User.put("username", username);  
User.put("password", password);

执行成功的回调函数中添加注册成功的后续操作，提示用户是不是使用所注册的用户直接登录，在“确定”选项添加登录功能。

User.saveInBackground(new SaveCallback() {*//注册成功的回调函数*    
 if(数据写入成功){  
 *//写入成功，弹窗提示是否直接登录，是则跳转至主界面并传回用户名*

}   
 });

④界面跳转功能

Android在两个activity间跳转是可以直接使用finish（）来结束当前界面，但是如果要在两个界面之间传递参数，则需要使用Intent接口：

从主界面跳转到登录界面，在登录界面返回时传回用户名参数，则需要使用Intent跳转：

public void 页面跳转函数(View v){Intent intent = new Intent();  
 intent.setClass(MainActivity.this,LoginActivity.class);  
 MainActivity.this.startActivityForResult(intent,requestCode);  
 }

登录界面需要使用finish方式关闭，并在关闭前执行传值函数：

Intent mIntent = new Intent();  
mIntent.putExtra("name", username);  
*// 设置结果，并进行传送*LoginActivity.this.setResult(resultCode, mIntent);  
finish();

使用这种方式跳转界面，在跳转后的界面关闭时会执行一个回调函数，在这里可以处理传回的参数

*// 回调方法，从第二个页面回来的时候会执行这个方法*  
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
 *String name = 登录界面返回的用户名；*

*用户名输入框写入返回的用户名*  
 }  
}

5.2定位与地图模块

通过地图服务可以获取用户当前位置信息，包括有当前的经纬度坐标和获取时间、当前速度等数据，因此地图服务是该系统功能实现的核心。根据高德地图Android开发文档了解到，高德地图使用的是火星坐标系，不能直接使用手机GPS获取的经纬度坐标，需要转为火星坐标系才能正常显示点位信息，这里这里为了避免转换，将直接使用高德地图SDK的定位服务，直接获取火星坐标系所对应的位置信息。

5.2.1高德地图SDK简介

高德地图 Android SDK 是一套地图开发调用接口，开发者可以轻松地在自己的Android应用中加入地图相关的功能，包括：地图显示（含室内、室外地图）、与地图交互、在地图上绘制、兴趣点搜索、地理编码、离线地图等功能。本系统主要使用了高德地图的地图显示与定位服务，地图显示使用2D地图。

5.2.2地图服务的实现

①在工程的配置文件AndroidManifest中添加高德地图的2D地图服务和定位服务：

<meta-data  
 android:name="com.amap.api.v2.apikey"  
 android:value="aff69cd7956149a182effe4aec07980a" />//地图服务秘钥  
 *<!-- 定位需要的服务 -->* <service android:name="com.amap.api.location.APSService"></service>

②加载地图并设置地图中心点为南湖校区

*//加载地图  
mapView* = (MapView) findViewById(R.id.***map***);  
*mapView*.onCreate(savedInstanceState);*// 此方法必须重写***aMap** = *mapView*.getMap();  
*//启动定位*initLocationService();

③设置全局变量*latitude、longitude*来记录定位信息

*//定位回调函数*  
 public void onLocationChanged(AMapLocation amapLocation) {  
 if (定位成功) { *latitude*=定位的纬度;*//获取纬度  
 longitude*= 定位的经度;*//获取经度 }}*

5.3上传共享位置信息数据

乘车端用户将车辆位置信息上传至数据库，等候端用户向数据库发送数据查询请求，并下载校车静态或动态信息，包括校车运行线路、站点分布、运营时刻表、该时间校车所处的大致位置、校车到达该站点所需要的时间等信息。

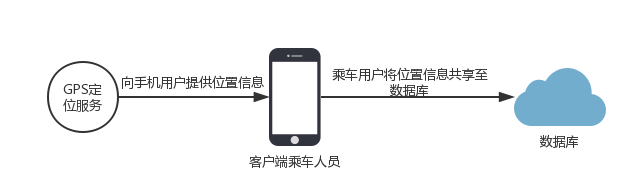
用户在乘坐校车时打开共享功能，则系统会启动线程starupload，该线程是每两秒执行一次上传，将用高德地图定位接口所获取的位置信息上传至数据库中，用户再次点击可以停止上传。

图5 上传共享位置信息数据流程图

实现方法

1. 用户交互控件为一个floataction控件，点击此控件可执行上传线程，再次点击可以关闭上传线程，为控件添加点击监听事件：

public void onClick(View v){   
 if(未启动上传){*//如果没有开始上传数据，则启动上传* Function.*StartUpload*();*//启动上传共享位置数据信息*   
 }else if(已经启动上传){  
 Function.*CloseUpload*();*//关闭上传*   
 }   
 }

1. 线程介绍：用Timer线程实现和计划执行一个任务的基础步骤：

* 实现自定义的TimerTask的子类，run方法包含要执行的任务代码
* 实例化Timer类，创建计时器后台线程。
* 实例化任务对象 (new RemindTask()).
* 制定执行计划。这里用schedule方法，第一个参数是TimerTask对象，第二个参数表示开始执行前的延时时间（单位是milliseconds，这里定义了2000）

public static void StartUpload(){

if(mTimer == null){

mTimerTask = new TimerTask(){

@Override

public void run() {

handler.sendMessage(msg);

}

};

mTimer = new Timer();

mTimer.schedule(mTimerTask, 1000,2000);

}

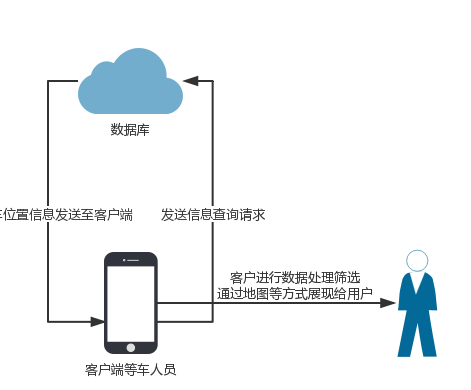
}

1. 使用handler每2秒处理一次数据，将获取到的位置信息数上传至数据库

protected static Handler *handler* = new Handler(){  
 public void handleMessage(final Message msg){  
 switch(msg.what){  
 case *LOAD\_PROGRESS*:  
 上传位置信息数据至表“bus\_location”   
 }  
 super.handleMessage(msg);  
 }

5.4校车实时位置计算

校车实时位置信息由客户端计算所得：①以系统设计原则来说，数据处理应放在服务端，但是LeanCloud云服务器的云端处理需要收费，在这里本系统是为了验证设计的正确性，所以将处理端也置于客户端中。②数据处理的目的：因为从数据库中获取的共享数据是当前所有在线用户所共享的数据，会包含有多辆校车的运营数据，需要使用使用数据处理的相应算法，自动识别多个用户所共享的相同校车的数据以及不同用户共享的不同校车的数据，并返回位置坐标。③算法返回的位置坐标在高德地图服务中显示。

图6 校车实时位置解算过程图

5.4.1聚类分析

以当前时间为条件，查询前2秒内的所有共享数据，这些数据包含3秒内由用户上传的，所有在运行的校车（有用户共享位置）的位置点，聚类分析是要把属于同一辆校车的，由不同用户或相同用户共享的数据筛选出来，相当于把数据分组，以获得同一辆校车在这2秒内所经过的位置点（用户共享的位置点）。再以此数据，可以计算出校车在这2秒内的位置（以平均值代替，将校车视为静态）。聚类分析是每秒执行以此，将算法返回的位置作为当前校车的位置，2秒的时间依据车辆的车速（50km/h-60km/h）可得误差范围为（25m-30m）。

5.4.2算法实现

算法主要是分组的思想，根据车辆的车速分析，车辆在2秒的时间内的形成在30米内，以极限的思想来考虑，在获取数据的获取时间范围内，得到了同一辆车的位置信息，而且这两个数据的时间间隔为2秒，则这两个数据的位置间的距离为30米，也就是说把校车在2秒内的最大行程30米作为一个阈值即判断依据。因此我们可以通过以下几个步骤去进行算法的实现。

1、聚类算法（K-Means）：

1. 从运数据库中读取2秒内上传的位置信息；
2. 设第一个点为第1个识别点，一次判断下一个点与第一个识别点的距离，若距离小于30米则认为该点与第一识别点在同一辆校车上；若该点与第一识别点的距离大于30米，则设该点为第二识别点，然后一次判断下一个点与两个识别点之间的距离（由于同一时刻最多有两辆校车，即从文昌发往南湖和南湖发往文昌，所以该算法设置两个识别点最多可以识别三辆校车）；如果该点与与识别点1的距离小于30米，则该点在校车1上，如果该点与识别点2的距离小于30米，则该点在校车2上，若该点与识别点1及识别点2的距离均超过了30米则该点位于校车3上；
3. 将表示同一辆校车位置的点存入同一数组；
4. 剔除粗差（Ransac算法）；
5. 将剔除粗差后的数组内达到位置做平均值表示校车位置，分别计算出没量校车的位置并添加在地图上。

2、剔除粗差（Ransac算法）

聚类后说形成的校车位置的点集赢分布在以识别点为圆心，30m为半径的圆上，且由于校车一直运动，可认为校车在5s内走的轨迹近似为直线。

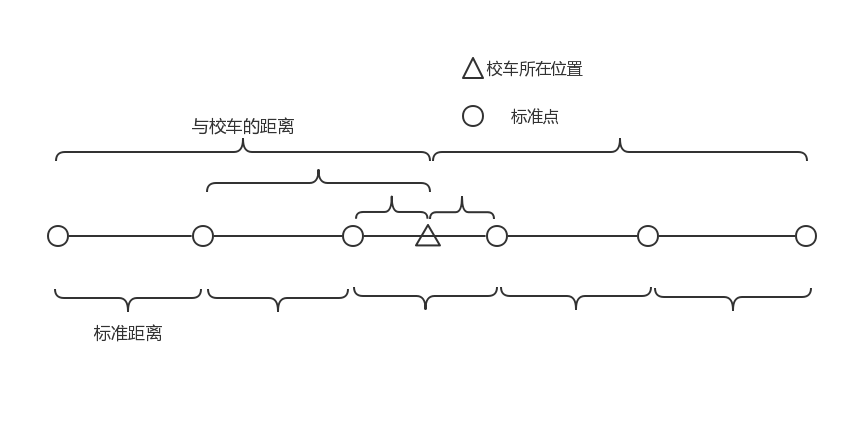
遍历点集中的任意两点（ai，aj）：

1. 其他点到ai，aj连成的直线距离如果小于10m，则写入数组中；
2. 如果写入a数组中的个数大于原点集中的个数的75%，则保留a数组，否则置为空；若a数组不为空，则比较K=[hk]/k(hk为第k个点到该直线的距离)来判断哪条直线及数组的选取更优，如果k值比前一个k值小，则用新的a数组来替换前一个a数组；
3. 从第②步中所得的最终数组为最优数组，将该数组返回到聚类算法中。

5.5抵达时间预计

目前校车的抵达时间是通过从云数据库中读取校车时刻表信息，然后通过与当前时刻比对直接给出与当前时刻最近的三个时刻的预计抵达时间。等车端用户结合地图上所显示的校车位置和给出的预计到达时间对校车的发车时间有个直观的认识。

为了更准确的得到校车的发车时间，我们想到了以下两种方式来更准确地预测校车的发车时间，但是由于时间问题这部分功能还在调试中，以下只是介绍这两种方法的构思。

1、①由于校车有两条路线，我们第一步要做的是对校车所走的路线进行判断。首先计算各标准点之间的距离，存在一个数组内，然后计算校车与各个标准点之间的距离，若校车与前后两点的距离的和与标准距离相等，则可确定校车在这一路段，同时能确定校车在南北哪条线。

②其次判断使用者所处的位置以及使用者所要前往的目的地，因为只有两个校区，因此我们可以根据乘车用户离哪个站点最近来判断，若在南湖那边则认为去往文昌，若在文昌那边则认为去往南湖。

③判断一定时间内校车在标准路段序号的变化，若减小则认为校车去往南湖，若增大则认为去往文昌。同时若发现校车在三食堂附近，则认为校车去往文昌，在终点附近则认为去往南湖。

以上就是对乘客乘车方向以及校车行驶轨迹的判断，接下来就是判断校车抵达的剩余时间。

2、①优先计算与使用者要乘坐的方向相同的校车，若校车若在的路段序号小于使用者所在的路段序号，则直接计算剩余距离，记为状态0;若大于，则认为错过该校车，不计算剩余距离，记为状态2。然后计算与使用者方向相反的校车，计算校车到终点的距离与该方向起点与使用者的距离的和，记为状态1。

②获取各路段的路况，畅通则视为满速，缓行则视为满速的百分之六十，堵塞则视为满速的百分之二十，若无法获取数据则视为满速的百分之八十，所剩距离除以各段速度再相加即为剩余时间。

**6 数据库设计概述**

6.1数据库环境说明

6.1.1 AVOS Cloud云数据库简介

AVOS Cloud是AVOS中国团队在2013年9月发布的应用开发一站式后端服务，为开发者提供数据存储 、实时消息、消息推送以及统计分析等服务。AVOS Cloud 的主要服务包括：无模式(schema free)的对象存储(Object Storage)、“云代码”扩展(Cloud Code)、跨平台的消息[推送](http://baike.baidu.com/item/%E6%8E%A8%E9%80%81/9908161)(Push Notification)、实时数据统计(Mobile Analytics)和社交组件(Social Component)，涵盖移动开发多方面的需求。针对每一个移动平台，不论是iOS、Android，还是Windows Phone、Web，都提供有原生的SDK，开发者可以简单集成使用。

AVOS Cloud 旨在帮助开发者处理好后台系统的各种挑战，最大限度地缩短产品开发周期，让开发者更专注于商业模式和用户体验。基于AVOS Cloud来开发应用，开发者无须考虑是使用MySQL还是[NoSQL](http://baike.baidu.com/item/NoSQL/8828247)来存储数据，更不用担心用户量和流量的快速增长。同时，集成AVOS Cloud的消息[推送](http://baike.baidu.com/item/%E6%8E%A8%E9%80%81/9908161)和数据统计服务，依托于云平台的[分布式处理](http://baike.baidu.com/item/%E5%88%86%E5%B8%83%E5%BC%8F%E5%A4%84%E7%90%86/3352171)能力，开发者可以得到更多大数据所带来的[附加价值](http://baike.baidu.com/item/%E9%99%84%E5%8A%A0%E4%BB%B7%E5%80%BC/3259317)。本系统主要使用了AVOS Cloud云数据库的数据存储（Storage）服务，所以将主要介绍此项服务。

AVOS Cloud提供的存储服务和我们熟知的[云存储](http://baike.baidu.com/item/%E4%BA%91%E5%AD%98%E5%82%A8/8326238)、云硬盘、网盘不一样，我们提供的是schema-free的对象（Object）存储服务。这里所说的“对象”，是程序开发中的数据逻辑单元，例如一个“用户”、一条“微博”、一则“评论”等等，都是“对象”。换言之，是面向程序开发的云存储服务。

AVOS Cloud的存储系统采用了分布式架构，提供了多副本的冗余支持，以保证数据极高的可用性；在数据量迅速增长的情况下，可以自动分片处理，支持PB级别的数据规模；另外，对于每一条数据（每一个对象），都提供有严格的[访问控制](http://baike.baidu.com/item/%E8%AE%BF%E9%97%AE%E6%8E%A7%E5%88%B6/8545517)协议，保证数据绝对安全。

在基本的“对象”增删改查操作之外，还提供“云代码”扩展的机制，基于方便开发者定制自己特有的功能，来完成更加复杂的业务逻辑，让后台逻辑可以按需灵活扩展；并且也支持网站托管。

除此之外，AVOS Cloud还支持图片、视频、音乐等大文件的存储，并且默认提供CDN服务，让数据分发可以[全网加速](http://baike.baidu.com/item/%E5%85%A8%E7%BD%91%E5%8A%A0%E9%80%9F/7943274)。

6.1.2数据存储特点与介绍

数据存储（LeanStorage）是 LeanCloud 提供的核心功能之一，它的使用方法与传统的关系型数据库有诸多不同。使用 LeanStorage 的特点在于

1.不需要单独维护表结构。

2.数据可以随用随加，这是一种无模式化（Schema Free）的存储方式。

3.所有对数据的操作请求都通过 HTTPS 访问标准的 REST API 来实现。

4.我们为各个平台或者语言开发的 SDK 在底层都是调用统一的 REST API，并提供完整的接口对数据进行增删改查。

LeanStorage 在结构化数据存储方面，与 DB 的区别在于：

1.Schema Free／Not free 的差异；

数据接口上，LeanStorage 是面向对象的（数据操作接口都是基于 Object 的），开放的（所有移动端都可以直接访问），DB 是面向结构的，封闭的（一般在 Server 内部访问）；

2.数据之间关联的方式，DB 是主键外键模型，LeanStorage 则有自己的关系模型（Pointer、Relation 等）；

6.2表汇总和表设计

实时校车系统的数据基础是共享位置信息数据和校车运行数据，可分为静态数据和动态数据，其中静态数据包括校车的时刻表、站点位置、校车的线路轨迹等，动态数据包括校车的实时位置数据、实时路况等。

本系统中校车静态数据的获取主要就是通过校官网的发布的校车时间信息，然后用这些原始数据设计数据库结构，将这些静态信息完整的、有效的组织起来，存储到数据库中，并完成校车数据的更新工作，为实时校车系统提供静态数据的支持。在动态数据即用户共享位置信息方面，由于本系统需要接收用户上报的实时位置数据，因此必须预先设计好用户上报的数据格式以及相应数据的存储格式以便系统对其的各项分析和处理。

6.2.1 用户信息

用户信息表包括有用户名和密码两个字段，类型均设置为string类型，且密码不能为空。

表2 用户信息表

Table 2 User Information Table

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 说明 | 备注 |
| Usename | String | 用户名 | 主键 |
| Password | String | 密码 | 不能为空 |

6.2.2 校车信息

1）校车时刻表：校车时刻表应包含始发站、令时、发车时间三个字段。数据来源为校官网发布的校车时刻表。

表3 校车信息表

Table 3 school bus information table

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 说明 | 备注 |
| Season | Number | 日期 | 1为工作日，2为周末 |
| Type | Number | 始发站 | 1为南湖，2为文昌 |
| Departure time | Date | 发车时间 |  |

2）共享位置信息数据表：该表为整个数据库中最重要的表，此表存储的为所有由用户所共享的位置信息数据，应包含有位置信息、共享者、共享时间等字段。

表4 共享信息表

Table 4 Shared information table

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 说明 | 备注 |
| Position | Geopoint | 位置 |  |
| User | String | 共享人 |  |
| Creatat | Date | 创建时间 |  |

**7 用户界面设计概述**

7.1 登录/注册界面

登录界面包括用户名及密码输入框，登录按钮，注册页面跳转按钮，以及帮助提示按钮。

注册页面包括用户名、用户密码、确认密码三个输入框以及确认注册和返回取消注册按钮等。

同时，登录注册界面也包含了等用户在进行错误操作或操作确认时进行相关提示的功能及对话框。

图7 登录/注册界面展示

7.2 主界面

主界面分为两部分，底部菜单栏+功能区，菜单栏有三个菜单分别为“地图”、“我的”以及“时刻表”选项，地图选项是地图展示功能，包括显示自己的实时位置、展示校车实时位置、校车站点位置；

时刻表选项的功能是是获取校车信息数据，用于展示校车时刻表信息；

****我的选项是打开用户的个人中心，个人中心可以展示个人信息（头像、用户名等），并提供登录界面跳转按钮，显示用户状态（登录/非登录），以及进行校车实时信息的查询与操作。

图8 主界面展示：地图界面，“我的”界面

7.3校车实时信息展现

用户发送查询校车的请求后，客户端从云数据库下载近几秒内乘车用户的上传数据，经过客户端解算后通过地图反馈给等车端用户。地图上可显示用户当前所在位置、校车当前所在位置以及校车说走过的路径等，同时打开时刻表可以显示最近一辆校车到达所需要的时间。

图9 不同时间显示校车的动态位置



图10 时刻表中显示最近班次校车预计到达时间