基于 MongoDB 的移动信息分享平台①

王振铎[1], 王振辉[2], 王红刚[1], 陈绥阳[1,3]

[1](西安思源学院 电子信息工程学院, 西安 710032)
[2](西安翻译学院 工程技术学院, 西安 710021)
[3](西安交通大学 信息科学系, 西安 710043)

摘 要: 目前应用系统已经由 PC 转移到移动设备上, 高校正处于建设智慧校园的高峰期. 急需构建基于移动设备的信息分享平台. 在数据库选择上选择 NoSql 数据库中的 Mongodb 作为数据存储数据库, 根据高校的特点, 设计并实现了一个信息分享平台. 实际应用表明, 该平台操作简便、运行稳定, 方便了师生的使用.

关键词: 信息分享; MongoDB; Andriod; 移动

Mobile Information Sharing Platform Based on MongoDB

WANG Zhen-Duo[1], WANG Zhen-Hui[2], WANG Hong-Gang[1], CHEN Sui-Yang[1,3]

[1](School of Electronic & information, College of Siyuan, Xi'an 710038, China)
[2](School of Technology and Engineering, Xi'an Fanyi University, Xi'an 710105, China)
[3](School of Information and Science, Xi'an Jitong University, Xi'an 710049, China)

Abstract: At present, the application system has shifted from PC to mobile devices, and the university is in the peak period of building the wisdom of the campus. It is in urgent need of building information sharing platform based on mobile devices. At first, MongoDB is chose to store data. According to the characteristics of the University, an information sharing platform is designed and implemented. Practical application shows that the system has the advantages of simple operation, stable running, convenient using for the teachers and students.

Key words: information sharing; MongoDB; Andriod; mobile

随着微信、微博等一系列移动应用的推广, 人们对移动互联网有了更深入的认识. 手机不再是一个传统单一的打电话、发短信的工具. 它可以解决工作和生活中的任何问题. 人们的工作、生活、娱乐正在从 PC 互联网向移动互联网过度. 大多数高校都加快了智慧校园的建设工作. 这一举措无疑加快了高校网络设备和电脑等设备的更新换代. 为师生提供了良好的硬件设施. 但其上运行的是大多是教学和行政办公系统, 缺乏具有高校特色的应用. 笔者查阅文献发现, 还没有基于移动设备的高校信息分享平台. 为此, 在我校智慧校园建设的基础上, 开发了该平台.

1 系统概述

系统的建设目标是为全校人员提供了一个交流的平台, 学校的领导、各职能部门、教师和学生, 均可以利用此平台发布消息、通知以及新闻. 教师和学生可以随时掌握学校、班级的各种信息、以及学校周边的生活环境. 系统功能结构图如图 1 所示.

(1) 登录注册: 为了方便系统的维护和管理, 系统有注册和登录的功能.

(2) 发布新闻: 系统中的用户可以随时、随地发布自己的所见所闻, 平台上的用户随时可以查阅.

(3) 发表评论: 新闻信息发布后, 其他阅读者可以发表自己的见解和想法.

(4) 添加好友: 平台针对高校, 应体现信息共享的优势, 所以, 在新闻发布的过程中, 可以随时将朋友加入到平台中.

(5) 直接拨号功能: 根据用户注册时提供的电话,

① 基金项目:陕西省教育厅自然科学研究课题(14JK2087)
收稿时间:2015-07-29;收到修改稿时间:2015-09-16

系统提供直接拨号通话功能.

(6) 即时聊天功能: 文字交流功能, 类似聊天工具.

(7) 周边信息查询: 为入学新生提供学校周边信息查询的功能, 使他们尽快熟悉学校周边的各种信息.

(8) 照片上传: 发布信息可以上传照片, 使系统的功能更加扩展.

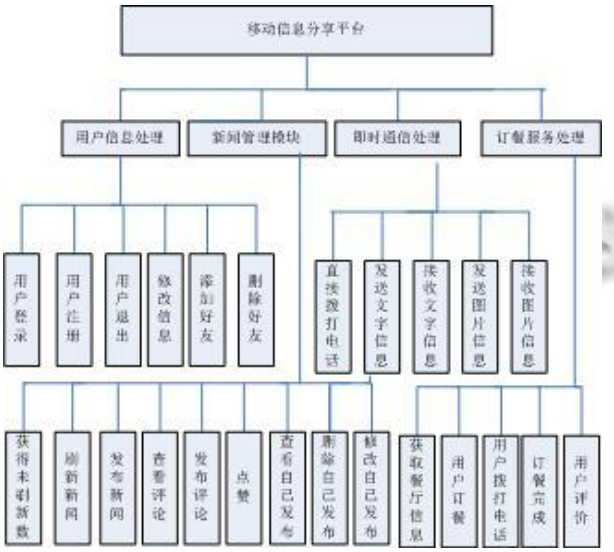

图 1 系统功能结构图

该系统的开发的软件环境是, 如表 1 所示.

表 1 系统开发环境

软件	版本
JDK	1.7
Tomcat	7.0
MongoDB	2.6
Eclipse	jee-keple
ADT	23
SDK	4.0.3

2 系统的体系结构设计

为了充分发挥客户端的处理能力、增强用户体验, 系统采用 C/S 架构, 这中结构的优点是很多工作可以由客户端处理后再提交给服务器. 本系统的客户端安装在 Android 智能手机中, 负责与用户交互, 用户有什么请求, 可以通过操作客户端程序来实现, 程序会根据用户的操作来判断用户的请求, 判断完成后会自动向服务器发送请求. 当服务器收到客户端的请求后, 首先判断用户的请求类型, 再做相应处理, 将处理结果返回给客户端, 客户端在将服务器返回的结果显示给用户.

2.1 客户端(Android 端)架构设计

客户端发送请求, 服务器收到请求之后, 将数据封装到响应对象中, 接收请求数据.

在 Android 技术中适合这项工作的有两个技术: 一个是在 Activity 中开辟一个子线程, 通过子线程来发送请求, 但是这种方式的不足之处在于, 此时的子线程必须要一个开辟它的 Activity 对象才可以. 而当界面不存在的时候, Activity 对象也将出栈, 也就是说 Activity 对象不存在, 若发生这种情况, 就无法接收网络请求数据了. 在这种情况下, 就必须采用 Android 组件 Service 来完成这项工作. Service 不依赖任何对象, 它是完全独立存在的, 就算 Activity 对象不存在, 它也可以照常工作.

但 Android 中 Service 组件是独立存在的, 如何才能把 Activity 和 Service 联系起来呢? Android 提供了另一种机制, Broadcast, 利用 Broadcast 就可以在 Activity 和 Service 之间传递数据, 这样一来就可以将用户在 Activity 上的操作类型传递到 Service 中, 在通过服务向服务器发送请求, 此时就可以将服务器传回来的数据通过广播带到 Activity 中并显示的客户端界面上. 客户端架构如图 2 所示.

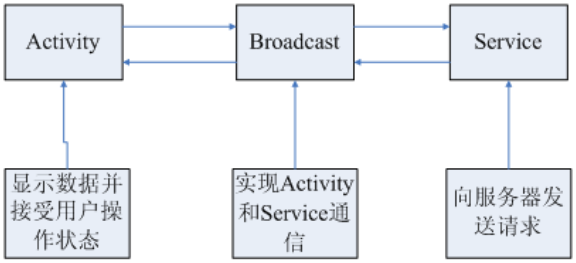

图 2 Android 总体架构

2.2 服务器端架构设计

服务器端的开发采用分层的思想[1]. 服务器端的核心是控制层, 客户端发送的 Request 请求, 由 Servlet 拦截, 控制层通过调用后台处理模块, 实现请求所需的业务功能, 然后将 Response 响应结果返回给客户端. 服务端的总体结构后有以下层次组成: 控制层、业务层、数据库层. 其中, 控制层, 控制页面数据的流向, 用 Servlet 实现. 业务层, 实体层和数据访问层, 用 JavaBean 实现. 数据库层, 是永久保存数据的地方,

用 MongoDB 实现. 服务器端架构如图 3 所示.

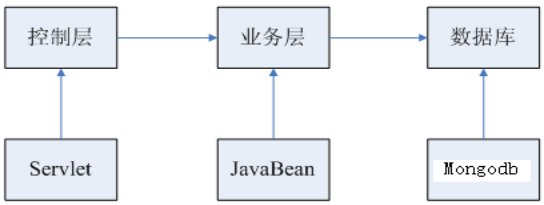

图 3 服务器端架构

3 MongoDB选择依据

MongoDB 是一个高性能、开源、模式自由(schema-free)的文档型数据库, 是云计算平台中 NoSql 数据库用的较多的一个[1]. 它在许多场景下可用于替代传统的关系型数据库, 具有以下特性:

(1) 跨平台: MongoDB 服务端可运行在 Linux、Windows 等平台上, 支持 32 位和 64 位应用, 默认端口为 27017. 最好运行在 64 位平台, 因为 MongoDB 在 32 位模式运行时支持的最大文件尺寸为 2GB.

(2) 高效的传统存储方式: 支持二进制数据及大型对象(如照片或图片).

(3) 面向集合的存储: 适合存储对象及 JSON 形式的数据.

(4) 支持丰富类型的数据存储.

以上三个特性, 使其非常适合大数据的要求, 具备网站实时数据存储所需的复制及高度伸缩性[2]. 同时, 它可以非常适合存放大尺寸, 非重要价值的数据, 非常适合互联网上数据的存储. 其它基于手机信息共享系统大多采用 MySql 关系数据库系统, 笔者首次采用云计算环境下的 MongoDB 数据库作为信息共享平台的后台数据库.

4 数据库设计

为了提高系统的性能, 本系统的数据库由服务器端的数据库和 Android 客户端数据库两部分构成. 其中服务器端采用 MongoDB, 客户端采用 SQLite 数据库, 以下是具体的设计:

4.1 服务器端数据库设计

服务器端数据库包括以下 10 张数据表:

(1) 用户表: (用户编号, 昵称, 姓名, 手机号码, 密码, 性别, 年龄, 头像编号, 电子邮件, 个性签名, 个人简介);

(2) 头像表: (头像编号, 头像状态, 头像地址);

(3) 好友关系表: (甲用户编号, 乙用户编号,);

(4) 新闻信息表: (新闻编号, 用户编号, 新闻文字信息, 发布时间, 点赞次数, 经度, 信息, 纬度信息, 发布时所在地址信息);

(5) 新闻评论表: (评论编号, 新闻编号, 用户编号, 评论的文字信息, 评论时间, 点赞次数);

(6) 图片表: (图片编号, 新闻编号, 商品编号, 商品评论编号, 图片地址);

(7) 餐厅信息表: (餐厅编号, 餐厅名称, 餐厅简介, 头像编号, 联系电话, 联系人姓名, 餐厅地址, 经度信息, 纬度信息);

(8) 商品表: (商品编号, 餐厅编号, 商品名称, 商品原价, 商品现价, 商品总数, 评分, 添加时间);

(9) 订单表: (订单编号, 商品编号, 用户编号, 餐厅编号, 价格, 完成状态, 下单时间);

(10) 商品评论表: (商品评论编号, 用户编号, 评论文字, 评分, 评论时间)

4.2 Android 客户端数据库设计

Android 端的数据库与服务端的数据库的区别在于, 服务端存储的是所有的数据, 而 Android 本地数据库(SQLite)[3], 只保存用户的个人数据.

这样做的优点在于能够提高用户体验. 试想上网速度不稳定时, 当启动程序, 从远程的服务上读取信息, 可能是字符信息, 也可能是二进制信息, 有时会等很久, 服务器才有响应, 在这种情况下, 数据没有回来之前, 用户是什么都看不到的, 即界面上什么也没有, 为了避免这种情况, 保证用户在服务器有无响应的情况下, 用户界面能够正常显示, 我们设计了本地数据库, 将上一次的数据先存储到本地, 在启动程序的时候, 先从本地获取数据, 这样速度加快了, 用户几乎感受不到延迟现象, 当远端服务的实时数据传回之后, 将实时数据显示的界面上, 这样效果就更好.

经过分析, 需要将以下信息保存在本地数据库, 分别是:

(1) 新闻信息表: (序号, 新闻编号, 用户编号, 新闻文字信息, 发布时间, 点赞次数, 经度, 信息, 纬度信息, 发布时所在地址信息);

(2) 联系人表: (序号, 用户编号, 昵称, 姓名, 手机号码, 密码, 性别, 年龄, 头像编号, 电子邮件, 个性签名, 个人简介);

(3) 聊天记录表(序号, 用户 1 编号, 用户 2 编号,

发送者编号, 文字内容, 图片地址, 时间).

5 关键技术

5.1 Json 格式数据的解析与封装

JSON 是一种轻量级的数据交换格式[4]. 它基于 JavaScript 的一个子集. JSON 采用完全独立于语言的文本格式, 使用了类似于 C 语言家族的习惯. 这些特性使 JSON 成为理想的数据交换语言. 系统中 MongoDB 数据采用的是 JSON 格式存储, 原因是 JSON 格式数据的传输性能好, 而且有利于 Android 客户端的接收和解析. 以下给出解析和封装的例子, 以 Java 语言为例, 假设 Json 格式如下:

```
String strJson = "{"students":[
{"name":"Jack","age":12},{"name":"Vista","age":23},
{"name":"Kaka","age":22},
{"name":"Hony","age":31}]}";
```

解析过程如下:

```
try { JSONObject jo = new JSONObject(strJson);
JSONArray jsonArray = (JSONArray) jo.get("students");
for (int i = 0; i < jsonArray.length(); ++i) {
JSONObject o = (JSONObject) jsonArray.get(i);
System.out.println("name:" + o.getString("name") + "," +
"age:"+o.getInt("age")); }
} catch (JSONException e) {
  e.printStackTrace();
}
```

需要封装的格式如下:

```
"{'deviceid':'C300772','uid':1,'pwd':1,'mdate':'2015-1-4
21:57:00','curr_version':'1.0'}"
```

封装方法如下:

```
Map<String, String> map = new HashMap<>();
map.put("deviceid", tm.getDeviceId());
map.put("uid", userName);
map.put("pwd", userPassword);
SimpleDateFormat sdf=null;
sdf=new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
map.put("mdate", sdf.format(new Date()));
map.put("curr_version", Constants.VERSION);
Gson gson = new Gson();
String strJson = gson.toJson(map);
```

说明: 代码中 Gson 是谷歌提供的一个类, Gson 是 Google 提供的用来在 Java 对象和 JSON 数据之间进行映射的 Java 类库[5]. 可以将一个 JSON 字符串转成一个 Java 对象, 反之亦然.

5.2 头像的拍摄选取裁剪

本系统的登录功能, 允许用户添加头像标志, 在朋友圈中, 大多使用自己的照片, 但照片或用手机随机拍摄的个人照太大, 不能作为登录的图像标志, 这时就需要对照片进行裁剪, 使其满足头像的尺寸要求, 故本系统中, 笔者自行编写了一个头像选取裁剪程序, 代码如下:

```
.定义拍照标志位
static final int PHOTO_REQUEST_TAKEPHOTO = 1;
. 从相册中选择
static final int PHOTO_REQUEST_GALLERY = 2;
. 结果
static final int PHOTO_REQUEST_CUT = 3;
. 调用系统的拍照功能
Intent intent=null;
intent=newIntent(MediaStore.ACTION_IMAGE_CAPT
URE);
. 指定调用相机拍照后照片的储存路径
intent.putExtra(MediaStore.EXTRA_OUTPUT,Uri.from
File(tempFile));
startActivityForResult(intent,PHOTO_REQUEST_TAK
EPHOTO);
.调用画册选择图片
Intent intent = new Intent(Intent.ACTION_PICK, null);
intent.setDataAndType(MediaStore.Images.Media.EXTE
RNAL_CONTENT_URI,"image/*");
startActivityForResult(intent,PHOTO_REQUEST_GAL
LERY);
.图片裁剪
Intentintent=newIntent("com.android.camera.action.CRO
P");
intent.setDataAndType(uri, "image/*");
. crop 为 true 是设置在开启的 intent 中设置显示的 view
可以剪裁
intent.putExtra("crop", "true");
. aspectX aspectY 是宽高的比例
intent.putExtra("aspectX", 1);
intent.putExtra("aspectY", 1);
```

```
. outputX,outputY 是剪裁图片的宽高
intent.putExtra("outputX", size);
intent.putExtra("outputY", size);
intent.putExtra("return-data", true);
startActivityForResult(intent,PHOTO_REQUEST_CUT)
;
.将图片制作成 Bitmap
Bundle bundle = intent.getExtras();
if (bundle != null)
 Bitmap photo = bundle.getParcelable("data");
}
```

5.3 基于 LBS 的定位与反地址编码

系统的周边信息查询功能, 需要对地址进行定位和反地址编码[6], 该功能实践步骤如下:

1)申请密钥

下面为申请密钥的地址, 在此就不在赘述.

http:.lbsyun.baidu.com/apiconsole/key

2)添加必备 jar 包, 如下图 4 所示.

图 4 jar 包图

3)配置清单文件添加开发秘钥

```
<application>
<meta-data
android:name="com.baidu.lbsapi.API_KEY"
android:value="开发者 key" />
</application>
```

同时需要添加所需权限, 这里就不赘述了.

4)核心代码

```
.初始化搜索模块, 也可去掉地图模块独立使用
GeoCoder mSearch = null;
. 定位相关
LocationClient mLocClient;
. 定位初始化
mLocClient = new LocationClient(this);
mLocClient.registerLocationListener(new
MyLocationListenner());
LocationClientOption option = new
LocationClientOption();
option.setOpenGps(true);. 打开 gps
option.setCoorType("bd09ll"); .坐标类型
option.setScanSpan(1000);
mLocClient.setLocOption(option);
. 初始化搜索模块, 注册事件监听
mSearch = GeoCoder.newInstance();
mSearch.setOnGetGeoCodeResultListener(new
MyGetReverseGeoCodeResult());
    .jingdu 代表经度信息, weidu、代表纬度信息
public class MyLocationListenner implements
BDLocationListener {
public void onReceiveLocation(BDLocation location) {
if (location == null) return;
.获得经纬度
double jingdu = location.getLongitude();
double weidu = location.getLatitude();
.封装经纬度信息
LatLng ptCenter= new LatLng(weidu, jingdu);
. 反 Geo 搜索
mSearch.reverseGeoCode(new
ReverseGeoCodeOption().location(ptCenter));
                }
            }
public void onReceivePoi(BDLocation poiLocation) {}
    }
获得地址编码监听器:
下面的 address 就是得到的地址.
public class MyGetReverseGeoCodeResult implements
OnGetGeoCoderResultListener{
public void onGetGeoCodeResult(GeoCodeResult arg0)
{}
public void
onGetReverseGeoCodeResult(ReverseGeoCodeResult
result) {
if (result==null|| result.error !=
SearchResult.ERRORNO.NO_ERROR) {return;
        }else {
String address = result.getAddress();
```

```
}}
  };
```

6 系统测试

为了保证系统的正常使用, 对系统进行了测试, 系统的测试环境如下:

(1) 服务器环境:

系统:Windows 7

硬件: 4G 内存、双核 CPU 主频 2.2GHz、500G 硬盘

(2) 软件环境:

Eclipse IDE for Java EE Developers、tomcat7.0、MongoDB2.6

(3) Android 手机:

系统: Android 4.4.2

硬件: 1G 内存、四核 CPU、主频 1.3GHz、8GSDcard

同时, 经过详细的系统测试, 各功能均能够正常使用, 界面响应操作流畅, 证明系统的可操作性、性能均达到了预期要求.

在测试过程中, 我们关注了功能和性能两个部分, 二者中最重要、最关键的是系统的性能. 这里我们用常用的关系数据库 Mysql 和本系统采用的 MongoDB 数据库以新闻表为例进行了测试和比较. 测试数据定为 1 亿条, 在插入数据时, MongoDB 执行速度 0.058 秒, Mysql 执行速度 0.153 秒, 效率远远低于 MongoDB 数据库, 而在查询性能上, MongoDB 的速度约为 0.003 秒, 也是远远高出 Mysql 数据库 3 倍左右时间的.

该实验证明, 此共享平台选择 MongoDB 数据库以及采用的开发技术是正确的.

7 结语

本文应用多种新技术开发一个基于 Android 的移动信息分享平台, 主要技术有 Java、服务器技术、MongoDB 数据库、Http 协议、Android 开发相关技术, 以及基于百度的 LBS 定位和反编码等. 系统具有技术新、功能全面、操作简便等特点, 可以满足移动信息共享的需求, 经过系统测试, 证明系统具有一定的先进性, 并具有实际的应用价值. 但本系统仍有功能拓展的空间, 同时数据的安全性在后期使用中, 需要进一步加强[6].

参考文献

1 于源,王宁,胡刚,邵珂珂.基于 Qt 和 ARM 的无线点菜系统的设计与开发.数字技术与应用,2015,(1):158–159.

2 邸铮.MongoDB 在煤炭行业 GIS 数据存储方面的应用.煤炭技术,2013,32(4): 200–202.

3 严霄凤,张德馨.大数据研究.计算机技术与发展,2013,(4): 168–172.

4 曹宇,陈海峰.基于 JSON、JavaScript、HTML5 和前端存储技术的均衡运算框架.实验室研究与探索,2014,(5):116–119.

5 沈美,于翔.浅析在 Android 系统中 JSON 和 GSON 的用法.电脑编程技巧与维护,2014,(24):81–83.

6 陈伟鹤,李文静,朱江.基于社交网络好友攻击的位置隐私保护模型.计算机工程与科学,2015,(4):692–698.