目 录

[第一章 综合设计的进展情况 2](#_Toc26501)

[1.1针对工程问题的方案设计 2](#_Toc12075)

[1.1.1本系统适用范围： 2](#_Toc2391)

[1.1.2功能摘要 2](#_Toc11694)

[1.1.3运行环境约束 2](#_Toc5507)

[1.1.4系统分工： 3](#_Toc22311)

[1.1.5具体需求： 3](#_Toc32710)

[1.1.6系统功能的具体实现及关系，见下图1-1-6： 5](#_Toc29301)

[1.1.7 ER图，见图1-1-7. 5](#_Toc30002)

[1.1.8条件与限制 6](#_Toc15187)

[1.2针对工程问题的推理分析 6](#_Toc8662)

[1.2.1后台交互 6](#_Toc6088)

[1.2.2歌词显示 7](#_Toc31986)

[1.2.3歌曲推荐 7](#_Toc28192)

[1.2.4社交化 7](#_Toc6022)

[1.3针对工程问题的具体实现 8](#_Toc18357)

[1.3.1前端总结 8](#_Toc2160)

[1.3.2后端总结 9](#_Toc3866)

[1.4知识技能学习情况 13](#_Toc10879)

[1.4.1.几大类的学习与掌握 13](#_Toc27736)

[第二章 存在问题与解决方案 15](#_Toc17445)

[2.1存在的主要问题 15](#_Toc6242)

[2.2解决方案 15](#_Toc2289)

[第三章 前期任务完成度与后续实施计划 16](#_Toc29216)

[参考文献 17](#_Toc8286)

# 综合设计的进展情况

## 1.1针对工程问题的方案设计

1.1.1本系统适用范围：

音乐播放器主要为手机提供播放音频的功能，适合所有喜爱音乐的人群使用。

### 1.1.2功能摘要

（1）使用音乐播放器播放歌曲；

（2）浏览歌曲列表；

（3）同步显示歌曲歌词

（4）拖动进度条调节播放进度

（6）后台播放歌曲

（7）上一曲/下一曲，播放/暂停等基本功能

（8）登录注册、找会密码（在线功能）

（9）好友系统（点赞、评论）

（10）数据库维护（包括歌曲的增删改查）、用户账号维护（包括修改用户信息，删除用户信息等）

### 1.1.3运行环境约束

操作系统：android2.2——4.2

数据库：QLite

软件平台：ADT

### 1.1.4系统分工：

### 付鼎：后台功能的实现（包括登录，注册，推荐/在线播放歌曲）

耿睿：UI界面的制作与优化

龚子成：在线功能的实现（包括歌曲评论，点赞等）

程弈茗：聊天界面（分享歌曲的实现，好友系统），滚动歌词

### 1.1.5具体需求：

（1）后台播放音频文件

描述：对指定音频文件进行后台播放，当程序被切换后台依旧可以播放。

加工：用户可在播放界面进行播放、暂停、下一曲/上一曲、播放模式切换、

进入播放列表、拖动播放进度操作。允许对本模块执行查询功能。

  查寻功能：点击播放按钮，获取自定义类Music，该类存储歌曲的信息。

 （2）显示歌曲歌词

  描述：对播放的歌曲的歌词进行同步显示。

  加工：当音乐播放时，自动搜寻对应的歌词并添加到播放界面，如果没

有歌词则不显示。允许对本模块执行查询功能。

  查询功能：查询歌曲对应的LRC歌词文件。

 （3）自动获取手机内音频文件

  描述：对手机内的歌曲进行自动加载。

  加工：打开软件进入到欢迎界面，欢迎界面两秒后自动跳转播放主界面，

在该界面完成自动遍历手机内的音频文件，并将数据保存到数据库

的表中，允许对本模块执行新增功能。

（4）显示歌曲列表

  描述：对获取的歌曲使用列表显示出来。

  加工：点击列表内的歌曲可以播放该曲目。允许对本模块执行查询功能。

  查询功能：查询数据库里歌曲的相关信息，添加到列表里；

  输出：歌曲名称与歌唱者显示在列表里。

（5）进度条显示歌曲播放进度，并可以拖动调整播放进度

 描述：实时更新歌曲的播放进度。

  操作：拖动进度条可调整歌曲播放进度。允许对本模块执行修改功能。

修改功能：拖动进度条修改音乐播放的进度。

  输出：歌曲播放进度被改变

（6）可在线点赞评论

描述：听歌的时候可以在评论区编写并发表评论，点赞。

操作：在评论输入框中编写评论，在点赞按钮上点赞。

输出：评论自动调整对齐

（7）登录/密码找回

描述：在主界面上可以登录进入本地用户，若忘记密码可以找回

操作：输入账号密码登录，若密码错误则拒绝登录。点击忘记密码会通过

邮箱发送随机八位密码进行重置。

## 1.1.6系统功能的具体实现及关系，见下图1-1-6：

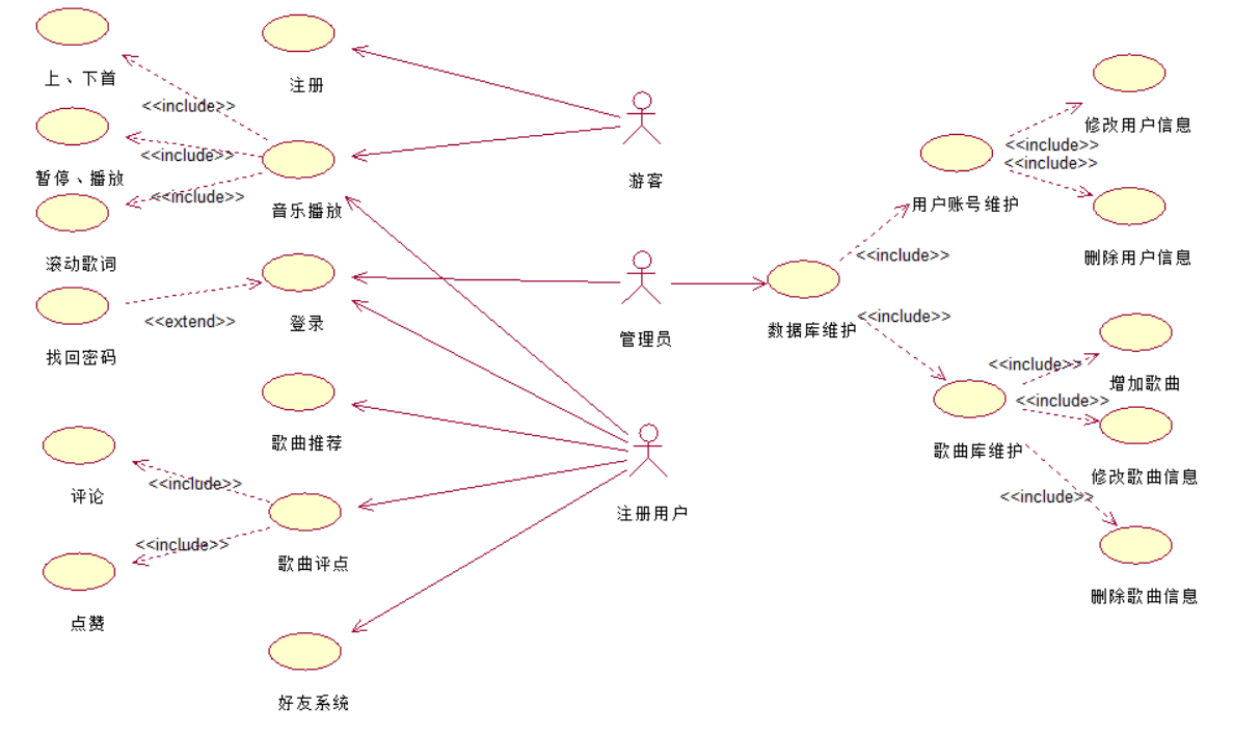


图1-1-6

## 1.1.7 ER图，见图1-1-7.

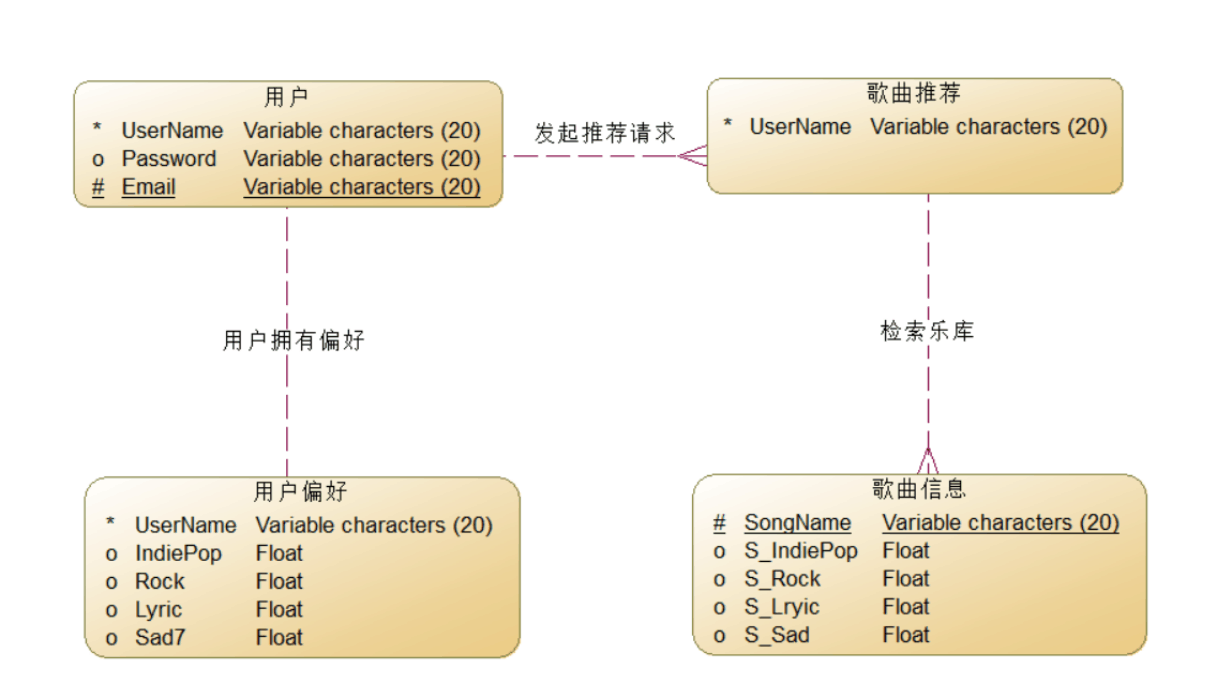


图1-1-7

### 1.1.8条件与限制

（1）开发人员开发经验不足；

 （2）编程人员对开发工具不熟悉，编程经验非常有限；

# 1.2针对工程问题的推理分析

## 1.2.1后台交互

原理：

是基于HttpClient，它是 Apache Jakarta Common 下的子项目，可以用来提供高效的、最新的、功能丰富的支持 HTTP 协议的客户端编程工具包。

步骤：

1. 创建HttpClient对象。

2. 创建请求方法的实例，并指定请求URL。

3. 调用setParams方法来添加请求参数

4. 调用HttpClient对象的execute(HttpUriRequest request)发送请求，该方

法返回一个HttpResponse。

1. 调用HttpResponse的getHeaders(String name)方法可获取服务器的响应头；

该对象包装了服务器的响应内容。程序可通过该对象获取服务器的响应内容。

6. 释放连接。无论执行方法是否成功，都必须释放连接。

## 1.2.2歌词显示

　歌词同步的实现思路很简单：获取歌词文件LRC中的时间和歌词内容，然后在指定的时间内播放相应的内容。获取不难，难就在于如何在手机屏幕上实现歌词的滚动。通过MotionEvent事件来判断用户对歌词的操作，以实现进度同步和歌词大小调整。

## 1.2.3歌曲推荐

原理：潜在因子算法，这种算法是在NetFlix（美国知名流媒体提供商）的推荐算法竞赛中获奖的算法每个用户（user）都有自己的偏好，比如A喜欢带有小清新的、吉他伴奏的、王菲等元素（latent factor），如果一首歌（item）带有这些元素，那么就将这首歌推荐给该用户，也就是用元素去连接用户和音乐。每个人对不同的元素偏好不同，而每首歌包含的元素也不一样。每个用户对每首歌都这样计算可以得到不同用户对不同歌曲的评分矩阵。从而实现歌曲推荐功能。

## 1.2.4社交化

原理：我们知道，传统的HTTP协议是无状态的，每次请求（request）都要由客户端（如 浏览器）主动发起，服务端进行处理后返回response结果，而服务端很难主动向客户端发送数据；而对于涉及实时信息的Web应用却带来了很大的不便，WebSocket的工作流程是这 样的：浏览器通过JavaScript向服务端发出建立WebSocket连接的请求，在WebSocket连接建立成功后，客户端和服务端就可以通过 TCP连接传输数据。因为WebSocket连接本质上是TCP连接，不需要每次传输都带上重复的头部数据，所以它的数据传输量很小。

## 1.3针对工程问题的具体实现

### 1.3.1前端总结

1.3.1.1歌词方案设计与实现

自定义View来实现解析lrc歌词同步滚动、上下拖动、缩放歌词等功能

关于歌词有以下几个功能：

1. 实现歌词同步滚动的功能，即歌曲播放到哪句歌词，就高亮地显示出正在播放的这个歌词；  
   2）实现上下拖动歌词时候，可以拖动播放器的进度。即可以不停地上下拖动歌词，当手指离开屏幕时候 即从当前拖动到的歌词位置播放。  
   3）实现歌词的字体大小可以进行缩放的功能。即双指在屏幕进行缩放操作时，歌词的字体大小也进行相应的缩放操作。

1.3.1.2歌词显示原理

LRC歌词文件：lrc是英文lyric（歌词）的缩写，被用做歌词文件的扩展名。以lrc为扩展名的歌词文件可以在各类数码播放器中同步显示。

LRC歌词文件的格式：lrc歌词文本中含有两类标签：一是标识标签 ，二是时间标签。

1）标识标签

标识标签，其格式为“[标识名:值]”，主要包含以下预定义的标签：

[ar:歌手名] [ti:歌曲名] [al:专辑名] [by:编辑者(指lrc歌词的制作人)] [offset:时间补偿值] （其单位是毫秒，正值表示整体提前，负值相反。这是用于总体调整显示快慢的，但多数的MP3可能不会支持这种标签）。

2）时间标签

时间标签，形式为“[mm:ss]”或“[mm:ss.ff]”(分钟数:秒数.毫秒数)，数字须为非负整数，

比如”[12:34.50]”是有效的，而”[0x0C:-34.50]”无效。

时间标签需位于某行歌词中的句首部分，一行歌词可以包含多个时间标签

(比如歌词中的迭句部分)。当歌曲播放到达某一时间点时，MP3就会寻找对应的时间标签并显示标签后面的歌词文本，这样就完成了“歌词同步”的功能。

解析lrc歌词文件：从assets目录下读取test.lrc歌词文件内容，则可以调用上面的getFromAssets(String fileName)方法得到歌词的文本内容。

## 1.3.2后端总结

安卓端与服务器进行交互，是基于HttpClient，它 是 Apache Jakarta Common 下的子项目，可以用来提供高效的、最新的、功能丰富的支持 HTTP 协议的客户端编程工具包。

1.3.2.1服务器端连接步骤：

1. 创建HttpClient对象。

2. 创建请求方法的实例，并指定请求URL。如果需要发送GET请求，创建HttpGet对象；如果需要发送POST请求，创建HttpPost对象。

3. 如果需要发送请求参数，可调用HttpGet、HttpPost共同的setParams(HetpParams params)方法来添加请求参数；对于HttpPost对象而言，也可调用setEntity(HttpEntity entity)方法来设置请求参数。

4. 调用HttpClient对象的execute(HttpUriRequest request)发送请求，该方法返回一个HttpResponse。

5. 调用HttpResponse的getAllHeaders()、getHeaders(String name)等方法可获取服务器的响应头；调用HttpResponse的getEntity()方法可获取HttpEntity对象，该对象包装了服务器的响应内容。程序可通过该对象获取服务器的响应内容。

6. 释放连接。无论执行方法是否成功，都必须释放连接

代码：

|  |
| --- |
| *//设置URL地址*  **private static** String *ProcessURL*=**"http://210.41.99.178:8080/MusicService/finds.action?"**;  *//创建一个HttpClient对象* HttpClient httpclient = **new** DefaultHttpClient(); *//远程登录URL* processURL=*ProcessURL*+**"userName="**+userName+**"&email="**+password; Log.*d*(**"远程URL"**, processURL); *//创建HttpGet对象* HttpGet request=**new** HttpGet(processURL); *//请求信息类型MIME每种响应类型的输出（普通文本、html 和 XML，json）。允许的响应类型应当匹配资源类中生成的 MIME 类型* request.addHeader(**"Accept"**,**"text/json"**); *//获取响应的结果* HttpResponse response =httpclient.execute(request); *//获取HttpEntity* HttpEntity entity=response.getEntity(); *//获取响应的结果信息* String json = EntityUtils.*toString*(entity, **"UTF-8"**); *//JSON的解析过程* **if**(json!=**null**){  JSONObject jsonObject=**new** JSONObject(json);  result=jsonObject.get(**"message"**).toString();  } **if**(result==**null**){  json=**"登录失败请重新登录"**; } *//创建提示框提醒是否登录成功* AlertDialog.Builder builder=**new** AlertDialog.Builder(find.**this**); builder.setTitle(**"提示"**)  .setMessage(result)  .setPositiveButton(**"确定"**, **new** DialogInterface.OnClickListener() {   @Override  **public void** onClick(DialogInterface dialog, **int** which) {  dialog.dismiss();  }  }).create().show(); |

后台登陆代码：

|  |
| --- |
| //设置编码格式  **this**.response.setContentType("text/json;charset=utf-8");  **this**.response.setCharacterEncoding("UTF-8");  //初始化JSON  Map<String,String> json=**new** HashMap<String,String>();    **try**{  //初始化会话  SessionFactory sf = **new** Configuration().configure().buildSessionFactory();  Session session = sf.openSession();  //创建事务  Transaction tx = session.beginTransaction();  //创建SQL语句  String sqlQuery = "select u from UserInfo u where u.username='"+ userName + "'";  //执行SQL语句  Query lQuery = session.createQuery(sqlQuery);  //取出结果  ArrayList userList = (ArrayList) lQuery.list();  //关闭事务、会话  tx.commit();  session.close();  //新建User对象  UserInfo user = **new** UserInfo();  //判断返回结果是否为空  **if** ((**null** != userList) && (userList.size() > 0)) {  //取出返回的结果  user = (UserInfo) userList.get(0);  //放入登陆结果信息  **if** (user.getPassword().equals(password)) {  json.put("message", "登录成功!");  }**else**{  json.put("message", "账户或密码错误!");  }  }  }**catch**(HibernateException e){  e.printStackTrace();  }  //写入HttpServletResponse  **byte**[] jsonBytes = json.toString().getBytes("utf-8");  response.setContentLength(jsonBytes.length);  response.getOutputStream().write(jsonBytes);  response.getOutputStream().flush();  response.getOutputStream().close(); |

找回密码：

|  |
| --- |
| UserInfo user = **new** UserInfo();  **if** ((**null** != userList) && (userList.size() > 0)) {  user = (UserInfo) userList.get(0);  **if** (user.getEmai().equals(email)) {  String temppasswod=**null**;  //随机初始化8位密码  TempPassword temp=**new** TempPassword();  //发送邮件  Email e=**new** Email();  temppasswod=temp.getpassword();  e.send(email,temppasswod);  user.setPassword(temppasswod);  //将新密码保存到数据库中  session.save(user);  json.put("message", "重置密码成功！请在邮箱中查询！");  }**else**{  json.put("message", "NO！");  }  } |

## 1.4知识技能学习情况

## 1.4.1.几大类的学习与掌握

1.4.1.1.Activity —— 应用表示层（基类Activity）

应用程序中的每个屏幕都是通过继承和扩展基类Activity来实现的。

同一应用中的每个Activity是相互独立的。程序启动后显示的第一幅画面是应用程序的第一个Activity（默认窗口），而后可以根据需要从这个Activity启动另一个新的Activity。

Activity利用View来实现应用中的GUI（用户直接通过GUI和应用程序做交互）。Activity窗口内的可见内容通过基类View提供。使用Activity.setContentView()方法设置当前Activity中的View对象。

1.4.1.2.Service —— 没有可见的用户界面，但能够长时间运行于后台（基类Service）

运行于应用程序进程的主线程中，因此Service不会阻塞其他组件和用户界面。 Service是不能自己启动的，必须通过Context对象（如一个Activity）调用startService或bindService方法来启动（用这两种方法启动的Service的生命周期不同）。

Broadcast Receiver —— 用户接收广播通知的组件（基类BroadcastReceiver）

Android中的广播要么来自于系统，要么来自普通应用程序。 很多事件都可能导致系统广播，如手机所在时区发生变化，电池电量低，用户改变系统语言设置等。

1.4.1.3.Content Provider —— 为解决应用程序间数据通信、共享的问题（基类ContentProvider）

在Android中，每个应用程序都是用自己的用户ID并在自己的进程中运行。这样的好处是，可以有效地保护系统及应用程序，避免被其他不正常德应用程序所影响，每个进程都拥有独立的进程地址空间和虚拟空间。

Content Provider可以将应用程序特定的数据提供给另一个应用程序使用。其数据存储方式可以是Android文件系统、SQLite数据库或者其他合理的方式。

当数据需要在应用程序间共享时，我们就可以利用ContentProvider为数据定义一个URI。之后，其他应用程序对数据进行查询或者修改时，只需要从当前上下文对象获得一个ContentResolver, 然后传入响应的URI就可以了。

1.4.1.4.Intent—连接组件的纽带

以上4种基本组件中，除了Content Provider是通过Content Resolver激活外，其他3种组件Activity、Service和Broadcast Receiver都是由Intent异步消息激活的。

Intent在不同的组件之间传递消息，将一个组件的请求意图传给另一个组件。因此，Intent是包含具体请求信息的对象。

Intent是一种运行时绑定（runtime binding）机制，它能够在程序运行的过程中连接两个不同的组件。通过Intent，你的程序可以向Android表到某种请求或者意愿，Android会根据意愿的内容选择适当的

# 第二章 存在问题与解决方案

## 2.1存在的主要问题

1．缺陷描述：音乐清单有乱码，音乐无名称，查看不方便

  缺陷影响：其他音乐都有名称，音乐无名称，查看不方便

  推迟原因：目前的日志为了调试方便，显示了很多其它信息，在项目正式发布时会统一处理的。

2．缺陷描述：数据字典种类修改，默认值设置后，在调用该数据字典种类的数据字典，默认值无显示

  缺陷影响：数据字典种类的默认值设置后，不能显示设置的默认值，相当于数据字典种类默认值设置功能未实现

推迟原因：该功能暂时不好实现，需要和和系统的默认语种一起处理。

3.缺陷描述：多媒体添加，文件上传功能未实现。

 缺陷影响：文件上传功能未实现。

推迟原因：该功能暂时不好完成，在下个版本中完成。

## 2.2解决方案

在项目开始的时候应该制定编码标准，数据库标准，需求变更标准，开发和测试人员都严格按照标准进行，减少因为开发，测试不一致而导致的问题同时也可以降低沟通成本。正确布置测试环境，减少因为测试环境，测试数据库数据的问题而出现的无效bug。

  开发人员解决bug的时候，填写bug原因以及解决方式，方便bug的跟踪。开发人员在开发版本上发现bug，可以通知测试人员，因为开发人员发现的bug.很有可能在测试版本上出现。

**第三章 前期任务完成度与后续实施计划**

前期完成度略微滞后，功能实现进度完成，但存在未修复的bug，功能运行也有待进一步优化，团队合作能力强，通过频繁的交流使项目开发过程变得流畅

付鼎：功能设计：登录，注册，推荐/在线播放歌曲；

ppt制作；

后台服务器端总结：技能学习情况，方案设计与实现。

耿睿：UI界面制作，真机适配，功能跳转逻辑实现，界面测试及优化；

安卓端界面总结：技能学习情况，方案设计与实现；

龚子成:功能设计：常用功能，歌曲评论，歌曲点赞。

后台安卓端总结：技能学习情况，方案设计与实现。

程弈茗：功能设计：聊天界面（分享歌曲），好友系统；

安卓服务器端总结：技能学习情况，方案设计与实现。

付鼎与龚子成负责后端服务器与数据库开发，程弈茗与耿睿实现前端功能与界面。

**参考文献**

1. Lesaek A.Maciaszek（澳）.需求分析与系统设计: 44-76
2. 李刚.疯狂web讲义. （北京）：230-364