**解密陌生人系统开发**

**开发团队：Silence**

**指导老师：杨红梅**

**组长：石飞琪**

**组员：林宇强 张壳**

**备注：山东科技大学信息科学与工程学院**

**目录**

[一、概述 3](#_Toc7008)

[1.1 开发背景 3](#_Toc5071)

[1.2 开发目标 3](#_Toc10957)

[1.2.1 新用户注册 3](#_Toc27423)

[1.2.2 查找陌生人 3](#_Toc18356)

[1.2.3 解密游戏 3](#_Toc4358)

[1.2.4 通讯录 4](#_Toc12492)

[1.2.5 与陌生人语音聊天、图片分享 4](#_Toc21367)

[1.3 参考资料 4](#_Toc17903)

[1.4 设计原则 4](#_Toc32206)

[二、 需求分析 5](#_Toc3144)

[2.1 需求陈述 5](#_Toc21189)

[2.1.1 新用户注册 5](#_Toc11521)

[2.1.2 查找陌生人 5](#_Toc19176)

[2.1.3 解密游戏 5](#_Toc2642)

[2.1.4 通讯录 5](#_Toc25573)

[2.1.5搜索好友 5](#_Toc9765)

[2.1.6 与陌生人语音聊天 6](#_Toc28464)

[2.1.7图片分享 6](#_Toc11937)

[2.1.8表情发送 6](#_Toc708)

[2.1.9修改自己的详细信息 6](#_Toc26725)

[2.1.10关于软件App设置界面 6](#_Toc1302)

[2.2运行环境 6](#_Toc11061)

[三、 主要模块设计 7](#_Toc21874)

[3.1内部接口设计 7](#_Toc3424)

[3.2 数据库逻辑设计 7](#_Toc23634)

[3.2.1联系人内容提供者——数据库： 7](#_Toc31545)

[3.2.2聊天消息内容提供者——数据库： 8](#_Toc27513)

[3.3 安全保密设计 8](#_Toc9035)

[3.4主要技术难题攻关 8](#_Toc21460)

[3.4.1附近的人： 8](#_Toc9518)

[3.4.2发送语音信息，分享图片 9](#_Toc1902)

[3.4.3嵌入式表情开发 11](#_Toc8246)

[3.4.4 APP整体框架： 11](#_Toc4090)

[3.4.5联系人的首字母排序： 11](#_Toc26694)

[3.4.6用户找回密码功能： 11](#_Toc9925)

[3.4.7Android系统兼容性 12](#_Toc6800)

[3.4.8用户登录过程XMPP连接的维护 12](#_Toc24838)

[四、实现 13](#_Toc22798)

[4.1 编码 13](#_Toc15337)

[4.2 代码约定 13](#_Toc27374)

# 一、概述

## 1.1 开发背景

随着陌陌的上市，SoLoMoGlo（Social + Local + Mobile + Global）概念持续发酵，基于移动社交的商业模式层出不穷，相关产品众多。移动社交领域的持续火热，让其成为大学生创业的一个重要选择。基于这样的背景，我们要完成一个基于游戏促进互动的陌生人社交类移动应用。

与微信这类熟人社交软件不同，陌生人交友软件最基本的要求是建立人与人之间的关系。然而通过具有趣味的游戏，更能促进人与人之间的社交关系。主要社交场景简述如下：用户首先注册个人信息，然后通过摇一摇功能找到附近的陌生人。找到陌生人后需要完成一个趣味小游戏完成解锁，然后才能看到该陌生人的信息。接着就可以与该陌生人进行语音沟通、图片分享等社交活动。

## 1.2 开发目标

### 1.2.1 新用户注册

首先用户需要自拍一张照片，然后输入用户昵称、性别等信息，确认后完成注册。

### 1.2.2 查找陌生人

已注册用户进入查找陌生人功能后，用户通过摇动手机查找距离自己一公里范围内的陌生人，然后将这些陌生人昵称作为标记显示在地图上。用户点击某个陌生人进入解密游戏环节。

（注：用户摇动手机后App自动将其位置信息公布出去，供其他用户查找。）

### 1.2.3 解密游戏

用户需完成一个经典有趣的拼图解密游戏后才能看到该陌生人的照片和性别等信息。然后用户可将该陌生人加入自己的通讯录，同时系统发送一条推送消息给该陌生人，提示谁将其设为联系人，并同时将该用户自动加入到其通讯录中。

### 1.2.4 通讯录

通讯录以列表的方式显示用户的联系人。用户可以在此删除联系人。用户可以点击某个联系人进入聊天环节。

### 1.2.5 与陌生人语音聊天、图片分享

在聊天环节，用户可以发送表情，发送语音，或者发送图片功能实现多渠道的社交方式。

## 1.3 参考资料

Openfire3.9.3开源框架及其相关文档和Api；

Asmack4.0.0及其相关文档和Api；

百度地图开放Api及其相关文档；

Android4.1Api开发文档；

Android4.4Api开发文档；

## 1.4 设计原则

Openfire开源框架采用XMPP即时通讯协议实现，整个底层采用xml进行数据传输和交互，通过定义好的xml数据包格式传输相应的命令实现整个服务器的操控和配置。Openfire还制定了一套强大的插件机制，使开发者可以自己写一个Openfire的插件来为服务器添加额外的功能。

asmack包是XMPP协议提供给Android的开源jar包，内部实现了XMPP协议的基本即时通讯的功能，底层和Openfire一样采用格式化的xml进行数据传输，因此，也很容易通过自己编写的格式xml数据包来实现额外的功能。

只要在服务器和客户端实现相同的接口，便可通过这个XMPP的协议为软件系统添加自己需要的功能，软件的可拓展性高，适合二次开发。

# 二、 需求分析

## 2.1 需求陈述

### 2.1.1 新用户注册

首先用户需要自拍一张照片，然后输入用户昵称、性别等信息，确认后完成注册。

### 2.1.2 查找陌生人

已注册用户进入查找陌生人功能后，用户通过摇动手机查找距离自己一公里范围内的陌生人，然后将这些陌生人昵称作为标记显示在地图上。用户点击某个陌生人进入解密游戏环节。

（注：用户摇动手机后App自动将其位置信息公布出去，供其他用户查找。）

### 2.1.3 解密游戏

用户需完成一个经典有趣的拼图解密游戏后才能看到该陌生人的照片和性别等信息。然后用户可将该陌生人加入自己的通讯录，同时系统发送一条推送消息给该陌生人，提示谁将其设为联系人，并同时将该用户自动加入到其通讯录中。

### 2.1.4 通讯录

通讯录以列表的方式显示用户的联系人。用户可以在此通过长按删除联系人。用户可以点击某个联系人进入聊天环节，可以点击联系人的头像查看联系人的详细信息。

### 2.1.5搜索好友

可以通过账户名，昵称搜索的方式查找陌生人，并查看信息，添加好友。

### 2.1.6 与陌生人语音聊天

在聊天界面，用户可以点击语音发送按钮进行语音的录制，并把语音发送给对方。

### 2.1.7图片分享

在聊天界面，用户可以点击图片发送按钮，选择手机中的图片发送给对方，实现交互的图片分享。用户可在聊天界面点击聊天记录中的图片进行图片的全屏观看。

### 2.1.8表情发送

软件还内嵌了一套表情供用户使用，用户可以在聊天界面通过点击表情发送按钮，调出表情面板，选择自己想发送的表情进行发送。

### 2.1.9修改自己的详细信息

可以通过任意界面内点击自己的头像进入设置界面，对自己的详细信息进行设置修改。

### 2.1.10关于软件App设置界面

可以对软件的当前账户进行删除，或者设置账户的状态，设置软件的通知栏提醒，声音提醒开关，清空App的聊天记录，聊天图片以及缓存，恢复软件的初始设置等操作。

## 2.2运行环境

客户端：

该App可运行在Android4.0以上的Android系统。

服务器端：

服务器端理论上可运行在Windows和Linux下，Linux下未经测试，Windows下测试通过。

# 主要模块设计

## 3.1内部接口设计

XMPP内部通过xml格式的数据进行客户端服务器的信息交互。通过客户端服务器建立的Socket生成一个交互的输入输出流，通过这个输入输出流可进行客户端与服务器的数据交互。

下面简述客户端对xml数据的接收和发送，服务器端的原来与此相同：

Xml接收：通过将Socket中的输入流中的字节数组数据分装到XmlPullParser中，通过识别xml中的命名空间找到软件系统中接收这种xml格式的模块，在这个模块中通过PULL的方式对该xml数据进行解析，将数据封装到对应的数据结构中，达到解析服务器发送的数据的目的。

Xml发送：通过将本地的数据封装到相应的数据结构中，在这个数据结构中实现toXML()方法，将数据结构中的数据封装成固定的xml数据格式字符串，再将这个字符串通过Socket的输出流发送到服务器，实现客户端对服务器的数据发送。

通过这个方式实现了服务器和客户端的数据交互，只要在xml的命名空间中指定正确，该xml数据就会被送到软件系统中的对应模块，达到xml数据控制多种交互信息的目的。

## 3.2 数据库逻辑设计

### 3.2.1联系人内容提供者——数据库：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 默认值 | 主键 | 备注 |
| 1 | \_id | Integer | 0 | 是 | 行ID |
| 2 | account | TEXT |  |  | 联系人JID |
| 3 | name\_by\_me | TEXT |  |  | 联系人备注 |
| 4 | sort | TEXT |  |  | 联系人排序字符串 |
| 5 | section | TEXT |  |  | 联系人首字母 |
| 6 | nickname | TEXT |  |  | 联系人昵称 |
| 7 | real\_name | TEXT |  |  | 联系人真实姓名 |
| 8 | email | TEXT |  |  | 联系人邮箱 |
| 9 | gender | TEXT |  |  | 联系人性别 |

### 3.2.2聊天消息内容提供者——数据库：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 类型 | 默认值 | 主键 | 备注 |
| 1 | \_id | Integer | 0 | 是 | 行ID |
| 2 | who\_id | TEXT |  |  | 消息发送方JID |
| 3 | body | TEXT |  |  | 消息主体 |
| 4 | type | TEXT |  |  | 消息类型 |
| 5 | status | TEXT |  |  | 文件传输状态 |
| 6 | session\_id | TEXT |  |  | 当前对话方JID |
| 7 | session\_name | TEXT |  |  | 当前对话方名称 |
| 8 | unread | Integer | 0 |  | 消息是否未读 |
| 9 | time | TEXT |  |  | 消息发送或接收毫秒时间 |
| 10 | progress | Integer | 0 |  | 文件传输进度 |
| 11 | filepath | TEXT |  |  | 文件路径 |

## 3.3 安全保密设计

Openfire服务器使用BlowFish进行用户名网络传输过程中的加解密，而找回密码插件的加密方式采用MD5进行加密，插件可根据需要更改加密方式。

## 3.4主要技术难题攻关

### 3.4.1附近的人：

通过摇一摇上传自己的位置信息（经纬度）到服务器，服务器查找半小时内在一公里范围内分享上传自己的位置信息的陌生人，并把附近的人信息回传到请求客户端，客户端再把附近的人显示出来。整个过程信息交互采用xml数据包进行传递，具体信息包如下所示：

请求包：

<iq id=”” from=”” to=”” type=”get” >

<query xmlns = "urn:xmpp:rayo:lbsservice" >

<item username=”” updatetime=”” gender=”” nickname=”” longitude = "" latitude = "" />

</query>

</iq>

其中<iq>结点是XMPP协议中的IQ数据包头结点，设置它的子节点为<query> ，命名空间为:urn:xmpp:rayo:lbsservice，提供让服务器端识别该数据包，并流向附近的人插件，<query>的子节点<item>为位置信息主体，上传该位置信息的用户，性别，昵称，更新时间以及经纬度，提供这些数据给服务器查找附近的陌生人。

服务器端接收到这个数据包后，查找位置信息数据库，查找到半小时内上传过位置信息的一公里范围内的陌生人，并把附近的人信息回传到请求客户端：

回传IQ数据包信息如下：

<iq id=”” from=”” to=”” type=”result” >

<query xmlns = "urn:xmpp:rayo:lbsservice" >

<item username=”” updatetime=”” gender=”” nickname=”” longitude = "" latitude = ""

jid=””/>

</query>

</iq>

username：用户名

updatetime：上传位置信息的时间

gender：用户性别

nickname：用户昵称

longitude ：经度

latitude ：纬度

jid：用户XMPP唯一的JID

信息内容和发送数据包类似，多了一个XMPP内部的用户专有JID，用来实现添加该好友的功能，其中每个<item>结点表示一个附近的人，回传信息包可以有多个<item>包。

客户端接收到回传的位置信息之后，开启百度地图界面，把收到的附近的人根据经纬度显示在地图上。用户点击地图上的一个陌生人，便会根据收到的JID向服务器请求该陌生人的详细信息，在UI线程上，把界面跳转到解谜游戏界面，等用户把解密游戏攻关之后，便可以把后台请求加载好的详细信息显示在陌生人用户详情上，用户可以选择是否添加为好友。

### 3.4.2发送语音信息，分享图片

用户聊天界面下发送语音信息即是软件开启Android的录音模块，把语音信息录音，再通过服务器把语音文件传送到对方，对方点击播放语音即实现了语音发送的功能。

用户聊天界面下，发送图片即是把图片文件通过服务器发送到对方。

综上，唯一的技术难题便是文件传输。

Asmack的文件传输功能有一定的缺陷，导致Android客户端难以实现文件传输功能，于是我们便设想出三种实现文件传输的方式：

1、将传输的文件转换成Base64编码字符串，并把字符串通过XMPP的Message数据包像普通的聊天消息传送给对方，在这种Message上进行标记即可实现此功能。但是这种方式有个致命的问题即是允许传输的单个文件大小限制较小，在传输一些大图片无法实现。

1. 模仿XMPP原本的文件传输的协议，自行开发出一个Openfire的文件传输代理，通过Openfire实现双方的文件信息交互，再通过一端开启一个ServerSocket充当临时文件服务器，由发送端开启一个Socket把文件传送过去。这个方式的弊病即是开启的Socket难以被Openfire服务器调控，文件传输过程中出现网络问题导致的丢包等问题难以被服务器调控保证接收方收到的文件的可靠性。
2. 模仿XMPP的原本的文件传输协议，由服务器代理通过Message实现文件的基本信息的双方协商，再由发送方通过HTTP协议将文件传输到HTTP服务器，并把文件信息告知接收方，接收方通过这个文件信息连接HTTP服务器即可找到对方发送的文件并进行下载。下载完成后HTTP服务器会将该文件删除，保证服务器的空间。这个方式使得服务器在文件传输过程中可以及时检测传输文件的可靠性，并可在服务器端进行检测，文件出现问题及时通知发送方重新发送，避免了接收方收到的损坏文件导致流量的浪费；而一旦文件上传成功而接收方由于网络原因接受失败，则可直接在接收端重新到HTTP服务器进行下载，避免了发送方反复发送相同文件浪费流量。
3. 工程中对一和三方法进行了实现，

方法一：

// 得到文件输入流

FileInputStream inputStream = new FileInputStream(file);

// 转化成字节数组

byte[] buffer = new byte[1024];

int len = 0;

ByteArrayOutputStream outputStream = new ByteArrayOutputStream();

while ((len = inputStream.read(buffer)) > 0) {

outputStream.write(buffer, 0, len);

}

// Base64编码

String result = Base64.encodeBytes(outputStream.toByteArray());

将传输的文件转换成Base64编码字符串，再进行发送。

方法三：

通过HTTP服务器暂存文件时间文件发送

Message信息包格式如下：

<message id="" to="" type="chat">

<subject>file-transfer</subject> //文件传输的标记

<subject xml:lang="zh">file-transfer</subject>

<fileinfo>

<fileName>IMG20150520002516.jpg</fileName> //文件名

<saveFileName></saveFileName> //文件储存名，唯一标示符

<mime\_type>image/jpg</mime\_type> //文件类型

<status>sending</status> //文件发送状态

<date>1432135551162</date> //文件发送的毫秒时间

<md5>25c66b3e7bf676edc47e6905018cd299</md5>//文件的md5校验码

<path>/storage/sdcard0/DCIM/Camera/IMG20150520002516.jpg</path>//文件在发送方的储存路径

</fileinfo>

</message>

1. 其中的saveFileName采用UUID类获取一个唯一时空标识符加上原本文件名作为文件在服务器的储存文件名，避免了多人同时上传同一文件造成的冲突。
2. 上传到HTTP服务器的文件在服务器通过saveFileName获取的hashCode后八位字节转换成的Integer将文件分配到一个特定文件夹中，这是为了将文件打散，从而尽可能的避免冲突和增快检索文件的速度；而接收方同样也可以通过saveFileName得到目标文件的具体地址，从而下载到正确的文件。这样打散文件既加快服务器效率，也方便了收发双方的文件检索。

### 3.4.3嵌入式表情开发

通过将表情图片文件存放在drawable文件夹下，通过识别发送和接收信息中的特定编码，将特定编码转换成表情图片显示出来，即用户以为发送了图片，其实只是发送了一个编码字符串，实现用户聊天的多样化，并简化后台的服务器功能实现。

表情发送面板由聊天界面中的表情发送按钮控制其显示和隐藏。

表情编码由\开头，后面三位数字代表drawable中的一个表情，软件先嵌入51个表情，分成三页，三位数字中的第一个数字表示是页数，后两个数字表示该页中的17个表情中的对应表情，从111到127，该编码字符串由正则表达式（“**[/]{1}[1-3]{1}([1]{1}[1-9]{1}|[2]{1}[0-7]{1})**”）进行匹配识别，匹配成功便把四个编码字符转换成表情显示出来，匹配失败就按照原字符串输出显示。

EditText通过SpannableString实现图文并茂，TextView通过SpannableStringBuilder实现图文并茂。

### 3.4.4 APP整体框架：

该APP采用双进程协同工作，一个后台的IMService独占一个进程，维护和Openfire服务器的连接，所有的服务器连接和数据交互操作都在这个进程（由XmppManager进行操作）中执行；而另一个进程用来控制UI的刷新和交互，所有用户交互都在这个进程中实现。

既然前台后台分处不同进程，进程间交互比较困难，故让XmppManager对aidl文件接口实现，使得后台进程的XmppManager对象能够在UI进程中得到一个实例。通过在XmppManager中实现相应的服务器连接和数据交互的方法，就能够在UI进程中通过这个XmppManager实例实现和服务器的交互操作。

即时信息和联系人的刷新通过在IMService进程中将收到的新消息储存在数据库联系人内容提供者和消息内容提供者，在UI进程中通过内容观察者检测对应内容提供者的数据变化，并把变化的信息在UI中进行刷新，实现后台Service维护服务器信息交互，UI前台用户交互协同作用的框架。

### 3.4.5联系人的首字母排序：

通过将常用姓氏的中文和拼音进行对应，并把拼音以及非汉字的字符串转换成小写字母生成一个用来进行排序的字符串sort，该字符串的首字母就是联系人按照字母分类的首字母T，首字母相同就按照排序字符串sort进行排序。

### 3.4.6用户找回密码功能：

当用户忘记密码需要找回时，可通过注册时提供的邮箱找回密码。通过向服务器提供账户名和邮箱，向服务器发送一个IQ数据包，在Openfire服务器端实现一个找回密码的插件，接收这个IQ数据包，对用户的邮箱进行校验，校验通过就向目标邮箱发送一条密码的验证码，同时通知客户端跳转到设定密码的界面，用户到邮箱查看验证码之后，可通过这个验证码对账户密码进行重设，从而实现找回密码的功能。

由于Openfire内部对密码进行封装和加密，因此要从服务器内部获取密码难度较大，因此考虑在账户注册和登录时后台通过加密向服务器端额外发送一组账户密码，通过服务器添加的插件来维护用户的账户密码，实现对账户密码的找回功能。

用户注册时向服务器传输账户、邮箱、密码关联信息

<iq id=”” from=”” to=”” type=””>

<query xmlns = "urn:xmpp:rayo:findpassword" >

<findpassword type = "" password = "MD5加密" account = "" email = "" username=""/>

</query>

</iq>

用户登录时向服务器提交更新此次的密码

<iq id=”” from=”” to=”” type=””>

<query xmlns = "urn:xmpp:rayo:findpassword" >

<findpassword type = "password" code = "" username = "" password =””/>

</query>

</iq>

用户找回密码通过验证码向服务器请求重设密码的请求

<iq id=”” from=”” to=”” type=””>

<query xmlns = "urn:xmpp:rayo:findpassword" >

<findpassword type = "code" account = "" email = "" username=""/>

</query>

</iq>

type:辨别是上传更新密码还是请求重设密码，password表示上传更新密码，code表示重设密码

username:用户名

account:用户JID帐号

password:用户密码，通过MD5加密，此处加密算法封装成方法，可通过更换方法来变更加密

算法

code:验证码

email:Email邮箱

### 3.4.7Android系统兼容性

Android4.4的拍照和图库功能：因为Android的Api开始，图库的相关Api以及通过图库内容提供者获取的图片Uri有所不同，因此，软件在第一次启动的欢迎界面对Android系统进行检测，并把Android系统版本存入到共享参数中，在App中关于拍照和图库的实现时对共享参数进行检测，以Android4.4为分界线采取两种处理方案，从而实现功能的系统兼容性。

### 3.4.8用户登录过程XMPP连接的维护

登陆界面实现登录功能前，App应该和服务器没有连接，进行登录操作之后才创建和服务器的XmppConnection，启动IMService后台来维护这个连接。要实现这个过程，需要在登陆界面自行创建一个和Openfire服务器的连接，进行账户登录的验证操作，验证成功之后再启动XmppManager的后台IMService维护创建好的连接，并跳转到联系人界面，从而实现对Xmpp连接的维护。

# 四、实现

## 4.1 编码

服务器客户端统一采用UTF-8中文编码方案。

## 4.2 代码约定

App包名定为com.silence.im

App名称为：解密陌生人

SDK版本为： 最低：Android-7 目标版本：Android-10

申请百度地图Key为：TtGGWZgk3IZAMfgEDfiBTGYQ