“活动召集令”安卓应用程序

详细设计文档

目录

[1引言 3](#_Toc339826339)

[1.1编写目的 3](#_Toc339826340)

[1.2项目背景 3](#_Toc339826341)

[1.3定义 3](#_Toc339826342)

[1.4 使用文字编辑以及画图工具 3](#_Toc339826343)

[2总体需求 4](#_Toc339826344)

[2.1需求概述 4](#_Toc339826345)

[2.2软件结构 5](#_Toc339826346)

[2.2.1应用总体架构： 5](#_Toc339826347)

[2.2.2软件功能结构图 6](#_Toc339826348)

[6](#_Toc339826349)

[2.3涉及的相关技术 7](#_Toc339826350)

[3程序描述 8](#_Toc339826351)

[3.1相关函数 8](#_Toc339826352)

[3.2性能 8](#_Toc339826353)

[3.3输入项目 9](#_Toc339826354)

[3.4输出项目 9](#_Toc339826355)

[3.5数据结构 9](#_Toc339826356)

[3.5.1链表 9](#_Toc339826357)

[3.5.2树 10](#_Toc339826358)

[3.7算法 10](#_Toc339826359)

[3.7.1全局排序算法 10](#_Toc339826360)

[3.7.2 时间碰撞算法 10](#_Toc339826361)

[3.7.3地理位置识别算法 11](#_Toc339826362)

[3.7.4心跳策略以及算法 11](#_Toc339826363)

[3.7.5内容审核算法 11](#_Toc339826364)

[3.8接口 11](#_Toc339826365)

[3.8.1 客户端内部接口 11](#_Toc339826366)

[3.8.2服务器接口 11](#_Toc339826367)

[3.8.3接口调用方式 12](#_Toc339826368)

[3.9数据库设计 12](#_Toc339826369)

[3.9.1活动发起者信息表 12](#_Toc339826370)

[3.9.2 用户活动表 13](#_Toc339826371)

[3.9.3 活动总分类表（频道） 13](#_Toc339826372)

[3.9.4 活动子分类表 13](#_Toc339826373)

[3.9.6用户评论表 14](#_Toc339826374)

[3.10代码托管 14](#_Toc339826375)

[3.11界面初步设计 14](#_Toc339826376)

[4团队成员以及开发进度安排 15](#_Toc339826377)

[4.1成员： 15](#_Toc339826378)

[4.2进度安排 15](#_Toc339826379)

[5存在的局限以及未来的发展 15](#_Toc339826380)

# 1引言

## 1.1编写目的

编写详细设计文档主要是为了清晰该应用开发相关的内容，包括项目背景、软件结构、功能等，便于开发团队对项目开发有一个明确的认识，提升开发效率，减少不确定性。

## 1.2项目背景

该项目产生背景：比赛需要

该比赛是移动互联网开放合作联盟主办的以“创所未想，携手逐梦”的2012移动互联网应用创新大赛。

活动主页：<http://app.omia.cn/Default.aspx>

## 1.3定义

所有函数名都以AC开头，AC是Activity Convene（活动召集）缩写。如发布活动函数为AC\_PublishAct

## 1.4 使用文字编辑以及画图工具

(1)MS word 2010

(2)MS visio 2003

(3)yEd Graph Editor

(4)UIDesigner

# 2总体需求

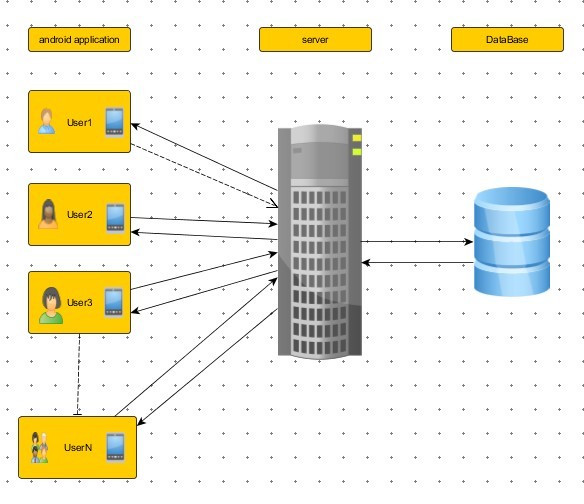
## 2.1需求概述

基于“尽最大可能为陌生人之间的认识建立桥梁”的思想，我们提出了“活动召集令”这样一款移动应用。旨在为当下这样的现状提供一种解决方式：虽然通信技术、网络普及程度进一步提高，但我们所处的活动圈子却越来越小，平时要运动或者出去游玩时，能够联系到的朋友并不多。大多数人会在论坛、QQ群、人人帖子上发布类似求伴信息，但效果并不明显。如果有一款专门的应用，提供“聚集人员来活动”，那效果会更加明显。该应用就如同电商里的淘宝，满足需求更快捷。后期还可以与恋爱社交、团购活动、KTV、电影促销等结合，体现出其商业价值。

举个例子：某学院的研究生只有10个人，其中基本上男女都有另一半，除了A同学。平时休息时间那些人都会陪自己的男（女）友。基本聚不到一块，然而A同学很喜欢集体运动，篮球、羽毛球等，A很苦恼找不到玩伴，他尝试了很多途径，QQ群，人人、微博，终于费尽千辛万苦找到了一个有着相同苦恼的女生。倘若用了我们的应用，只需要将自己的活动需求“一起打篮球”发到平台上，那么不只一个学院的人，甚至整个学校的人都能够看到，有相同需求的人直接就能加入，使得这样的联系更加方便。

## 2.2软件结构

### 2.2.1应用总体架构：

说明：终端应用（客户端）与服务器端进行http双向通信。整体架构设计成restful模式，即终端应用发送post、delete，get等http请求，将操作数据发送到后台，后台服务器接收请求，进行信息的存储，修改，删除操作。服务器为无状态模式，便于分布式部署。后端进行消息的返回，以及实时消息的推送。

### 2.2.2软件功能结构图

### 

功能详述：

1. 基于微博、人人、QQ、（之后增加杭州电信或者移动的接口）等开放接口，具体就是整合它们的用户资源，免去注册登录的麻烦。同时能够将我们的活动分享到这些社交平台上。提高活动的普及度以及互动性。
2. 发布活动时必须加上地址标签，这个标签依靠GPS定位确定可靠吗？还是说要手动输入。地址标签即具备了“社区建立”的这样类似的功能。不过这里存在一个问题，比如用户A发布了活动1，标明地址为：信雅达；而用户B发布了活动2，标明地址为：杭州信雅达科技;用户C发布活动3，标明地址为杭州信雅达科技公司。该如何识别他们的地理位置？其实他们发的是同一个地址。这是个问题.
3. 活动是分类的，比如聚餐、户外活动（爬山）、团购、运动（跑步）、KTV，学习（去图书馆）、旅游、技术开发大会等等。这里界面可以参考app store的布局。当然其他的界面也可以。
4. 以往活动频道。以往活动还是有必要保留的，虽然说它们存在的意义或者价值没有当前的活动大，但是具备参考价值，很多人初始接触这款软件，对它能带来什么或者发布什么样的活动没有概念。以往活动能够起到一定的启发作用。当然我们没必要原原本本地将历史数据展现，可以简单的列表下什么时候什么地点举行了什么活动以及评价如何。
5. 发布活动的功能，就像论坛中发布帖子一样，但我们的活动的排列不是那样，可以借鉴iphone手机上排放应用那样或者唱吧那样的排版。每个活动有时间期限以及人数限制，性别限制。当超过时间则从当前的界面中删去，只能到往期活动中查看。人数达到后，表明活动邀请已经成功，因为其已经不能够再接受新加入的人了，所以就将该活动从当前的活跃活动界面中删除，移到已经活动成功的界面。
6. 加入活动的功能，点击活动图标进去之后有加入按钮，以及一些信息（姓名、联系方式、性别等）录入的文本框。还有留言互动的功能。
7. 推荐列表功能（这里的推荐列表采用的是全局排序算法，不是针对单个用户的个性化推荐（按照某用户的注册时选择的感兴趣标签，而且即使是那样也有局限，因为用户很可能漏选感兴趣的标签）），系统将对正在进行的活动的参与人数进行统计，根据活动的活跃数（参加的人数）降序排列。

除全局排序推荐外，还可以在页面最下面根据位置地理信息，推荐最近地区的前几项活动。

1. 全网（或按频道）搜索功能以及摇一摇功能，搜索可以按照地区、时间、活动等关键字进行。增加摇一摇主要是因为用户有几种，一种是已经知道自己要参加某类活动，还有的就是随便逛逛，随机地选择活动。
2. 活动排行榜，按照不同的标签进行设置，按照“性别比”、“活动人数”等。
3. 频道订阅功能，例如你对旅游感兴趣，可以订阅，如果有新活动加入，后台能够推送相应的订阅信息给你。
4. 后台系统能够自动检测用户选择的活动是否存在时间冲突，并给以提示。

## 2.3涉及的相关技术

1. android客户端开发技术
2. Nodejs后台服务器开发技术
3. Mongodb数据库开发技术
4. Socket以及HTTP网络通信技术
5. 基于MQTT协议的长连接消息推送技术
6. 第三方开放平台接口使用

# 3程序描述

## 3.1相关函数

|  |  |
| --- | --- |
| 功能函数 | 含义 |
| AC\_PublishAct | 发布活动（发布活动信息，包括活动类型、名称、时间、人数限制、地点等） |
| AC\_JoinAct | 加入活动（提交用户信息，包括姓名或昵称、联系电话、性别等） |
| AC\_Comments | 留言互动 |
| AC\_GetRecList | 获取推荐列表（针对具体的活动频道进行活动的排序） |
| AC\_RSS | 频道订阅 |
| AC\_MsgPush | 后台信息推送 |
| AC\_Locate | 自动定位 |
| AC\_SearchAct | 活动搜索（可按照活动类别、人数、时间等关键字） |
| AC\_Shake | 摇一摇（类似随便逛功能） |
| AC\_CheckTimeCollision | 检测两次活动是否存在时间冲突，如果是，则给予提示 |
| AC\_GetHotActList | 获得热门活动列表，这有别于之前的推荐列表，针对的是全局活动 |

## 3.2性能

(1) 较高的并发处理能力,单台服务器达到3000 req/s

(2) 处理速度较快，基本请求在2秒之内返回结果

(3) 电量 < 10%

(4) 稳定性好，后台推送能够达到不掉线，消息100%送达

(5) 界面美观简介，应用打开速度快

## 3.3输入项目

(1)待发布的活动信息(活动名称、时间，人数限制，地点由自动地位函数提供，只需选择即可)

(2)选择的活动,加入信息

(3)取消活动的信息

(3)评论

(4)搜索的关键字

## 3.4输出项目

(1)已发布的活动信息（活动名称、活动时间、地点、发起人、已参加的人数、男女比例等）

(2)发布活动的成功与否的确认信息

(3)用户的评论

(4)对于活动的发布者，还可以查看当前加入的人员的信息

(5)加入活动的确认信息

(6)热门活动信息

(7)往期活动信息

(8)推送的用户订阅的新信息

(9)用户搜索的活动结果

(10)摇一摇的活动结果

(11)时间冲突检测的提示

## 3.5数据结构

### 3.5.1链表

(1)热门活动链表

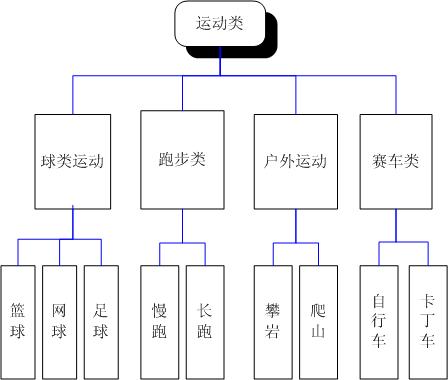
(2)往期活动链表

(3)搜索返回的结果链表

### 3.5.2树

运动的分类采用树的结构

如：



## 3.7算法

### 3.7.1全局排序算法

每一个频道，该频道下的活动按照参加人数（即热门程度）进行降序排列，从高到底进行分布（这是一种粗粒度的排序，当然之后可以进行优化，可以将性别比等其他因素放入参考，因为很多人还是希望活动中男女比例平均，那样比较有趣）

### 3.7.2 时间碰撞算法

同一个用户会对多个活动感兴趣，但万一活动A与活动B时间恰好处在同一个时间段，这就造成了冲突，因为用户不可能同时参加两个活动，而如果系统对这没有预警，用户又不知情，会造成不便，带来失望感，因此系统需要实现检测用户选择活动的时间是否重叠。

### 3.7.3地理位置识别算法

由于我们采用的是自动定位，会存在这样的问题，用户A发布了一个活动S，地址是杭州师范大学；而用户B也发布了一个活动H，地址是杭师。当用户C搜索杭师的活动时，活动S和H其实都是满足条件的，我们就需要对地址进行一定的处理，防止搜索不准。

### 3.7.4心跳策略以及算法

客户端与服务器之间保持长连接，需要定时发送心跳信息来表明是否还在线。需要研究ｇｐｒｓ／ｅｄｇｅ，ｈｓｐａ／ｕｍｔｓ，ｗｉｆｉ不同网络制式下的keepalive time。

### 3.7.5内容审核算法

系统需要具备敏感内容的检测功能，避免发布如“反动游行”、“抢劫”这类的不好的活动。审核不通过，则拒绝发布。更甚者，冻结该用户账户。

## 3.8接口

### 3.8.1 客户端内部接口

参看前面的相关功能函数

### 3.8.2服务器接口

|  |  |  |
| --- | --- | --- |
| 接口类型 | 接口 | 含义 |
| 写入接口 | activity/publish | 发布一条活动信息 |
| activity/edit | 修改活动信息 |
| activity/cancel | 删除活动信息 |
| activity/join | 加入活动 |
| activity/quit | 退出活动 |
| comment/message | 发布评论留言 |
| comment/cancel | 删除评论 |
| 读取接口 | activity/getact | 获取所有活动 |
| activity/name/detail | 查看某个活动的具体信息 |
| activity/gethot | 获得热门的活动列表 |
| activity/channel | 获取某个种类的所有活动 |
| activity/random | 摇一摇的随机活动列表 |
| activity/notify | 订阅信息推送 |
| activity/getallmem | 获得某项活动所有的参与者信息 |

### 3.8.3接口调用方式

采用http请求

基于restful架构的接口，客户端发送post或者get等http请求给服务器，具体数据包含在queryString中；服务器接收请求，路由到相应的处理函数中，进行处理，并将结果以json格式返回给客户端。

## 3.9数据库设计

### 3.9.1活动发起者信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| nickname | 用户昵称 | Varchar(50) | 主键 |
| authcode | 第三方平台授权码 | Varchar(100) |  |
| apipath | 第三方认证接口地址 | Varchar(200) |  |
| sysname | 平台用户名 | Varchar(50) |  |

### 3.9.2 用户活动表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| actid | 活动id | int | 主键 |
| actname | 活动名称 | Varchar(50) |  |
| actchannel | 活动隶属于的频道 | int | 外键 |
| address | 活动地址 | Varchar(100) |  |
| attenlimit | 参加者限制人数 | int |  |
| actstart | 活动开始时间 | date |  |
| actterminal | 活动截止时间 | date |  |
| actinitial | 活动发起人 | Varchar(50) | 外键 |

### 3.9.3 活动总分类表（频道）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| channelid | 频道id | int | 主键 |
| channelname | 频道名称 | Varchar（50） |  |

### 3.9.4 活动子分类表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| subcategoryid | 子类别id | int | 主键 |
| subcatename | 子类别名称 | Varchar(100) |  |
| parentcate | 父类别 | int | 外键 |

3.9.5活动参加者信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| joinid | 参加者id | int | 主键 |
| memname | 参加者姓名 | Varchar(20) |  |
| phone | 联系电话 | string(11) |  |
| gender | 性别 | int | 0：男；1：女 |
| actid | 活动id | int | 外键 |

### 3.9.6用户评论表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 含义 | 类型 | 备注 |
| commentid | 评论id | int | 主键 |
| content | 内容 | Varchar(200) |  |
| owner | 评论者 | Varchar(50) |  |
| actid | 活动id | int | 外键 |

## 3.10代码托管

使用Github用于代码托管，因为团队成员分布比较散，这样便于协同开发

环境：Windows平台（**GitHub for Windows 1.0**）以及linux平台

## 3.11界面初步设计

# 4团队成员以及开发进度安排

## 4.1成员：

尤志强 杭州师范大学阿里巴巴商学院

冯燕华 杭州师范大学美术学院研究生

景宏刚 杭州信雅达科技有限公司 安卓应用工程师

周子江 浙江大学软件学院研究生

顾天桥 杭州电子科技大学计算机学院

## 4.2进度安排

开发周期：11月

文档确定以及相关技术准备：11月5号~11月8号

正式开发：11月9号~11月30号

测试：12月1号~12月3号

部署：12月4号

上线：12月5号

# 5存在的局限以及未来的发展

初始阶段，该应用仅面向学生、白领群体，用于活动召集功能，解决找不到同伴的问题。在这阶段，产品不考虑盈利。

之后待用户群变大，接受程度提升，可以考虑加入一些商业运作，比如团购活动、商家促销，与一些公司合作。以及引入广告等。

另外，针对使用过程中用户反映的问题，做出相应的技术改进，以更好地满足用户需求。