虚拟火炬 APP 概要设计

文档目录

文档描述

版本记录

版本	日期	作者	说明
V0.1	2014-1-22	单志萍、孙秀丽、何晓磊、曹海林	

缩略语与术语

1 概述

针对目前全球性环境问题及青少年健康问题突出，例如全球气候变暖、海洋污染、森林锐减等。南京借助此次举办青奥会的契机，宣传绿色、环保，传递奥运精神的同时，将这些社会性的环境及健康问题，展现给青少年，并号召他们贡献出自己的一份力。

超凡团队希望通过青奥会虚拟火炬传递 APP 达到：

(1) 将南京国际化城市形象推广至全球；

(2) 倡导绿色、环保、节俭办青奥的青奥理念；

(3) 传递奥林匹克精神「卓越」、「友誼」及「尊重」；

(4) 宣传积极、运动、健康的生活方式；

(5) 寓教于乐，着重青少年文化教育各方面建设；

(6) 将青奥圣火平等传遍 205 个奥林匹克国家；

创新火炬传递形式，植入更多互动体验，优化用户体验，让全球青少年乐在其中；

针对不同国家地区惯用语言不同，制作多语言版本，为全球的青少年提供青奥委会实时概况及参与虚拟火炬传递，实现 205 个国家共同参与，传递青奥圣火；

活动阐述：砺砺会按照官方指定的传递路线，将火炬在各个国家里进行传递；

用户跟随砺砺，沿传递的路线同步火炬传递（用户不能超过砺砺）；

每传递到一个传递点，将获取查看进入下一关的条件；

用户参与互动游戏，满足条件后，即可进入下一个传递点。

亮点阐述：地图沿途上设置有神秘宝箱，抵达后，可打开宝箱获得意外惊喜。

活动步骤：圣火取火 → 圣火传递 → 点燃南京青奥圣火

2 虚拟火炬客户端设计

2.1 功能设计

2.1.1 分享功能

用户通过分享到社会平台（腾讯微博、QQ 空间分享）获取火炬步数。

2.1.2 取火功能

用户进入每一个打开取火页面，模拟真实奥运取火。用户通过移动火炬到凹凸镜范围内一定时间后点燃火炬，成功进入下一关卡。

2.1.3 传递功能

用户可以通过跑步、答题、小游戏、分享已经邀请好友获取每一关卡需要的步数；关卡的不同需要的步数不同（数据通过服务器控制）。

2.1.4 答题功能

通过获取获取火炬所在的国家的以奥运、各地文化、环保常识等小知识；参与知识互动问答，用户可获得火炬传递的步数。

2.1.5 跑步功能

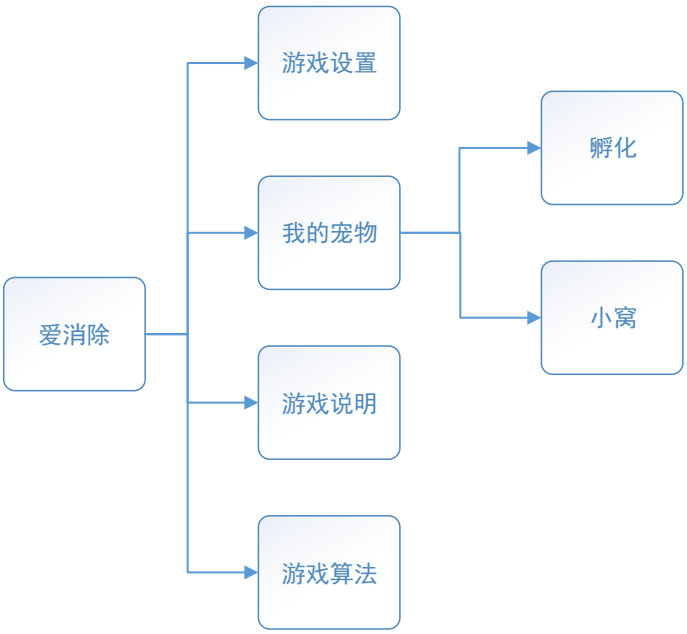
APP 内植入 LBS 定位系统，用户选择通过每天跑步来获取火炬传递步数，根据用户的地理位置移动，测算运动路程，获取步数。

2.1.6 小游戏功能

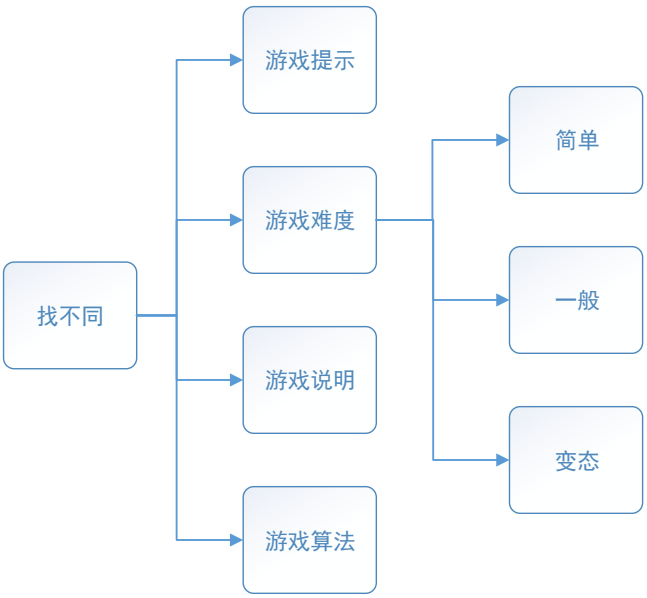
APP 预先植入各种不同小游戏（天天爱消除、找不同）模拟 Diamond Dash；用户通过

小游戏获取火炬步数；

模拟天天爱消除小游戏结构图：



模拟找不同小游戏结构图:



2.1.7 Bump 功能

用户通过 bump 向好友传递自己的奥运火炬给对方；完成传递后，好友可以在用户的传递火炬点为起点向前继续传递火炬。

2.1.8 步数统计

统计用户所有互动形式的步数、腾讯微博收听听众以及 QQ 好友获取的步数、以及目前所达到的奥运城市提醒。

2.1.9 用户登录

以腾讯微博、qq 号进行登陆注册，也可以以邮箱进行登陆。

2.2 模块划分

本客户端分为 8 个模块：

- 分享模块：用户登录社会平台分享自己的火炬步数已经目前所达到的奥运城市
- 取火模块：模拟真实的奥运取火
- 传递模块：不同的奥运国家传递场景搭建
- 答题模块：宣传不同的奥运传递国家的公益性常识
- 小游戏模块
- BUMP 模块
- 统计模块：统计用户所有互动形式的步数
- 登录模块：以腾讯微博、qq 号进行登陆注册，也可以以邮箱进行登陆，弱化邮箱登陆方式

2.3 与系统其他部分交互

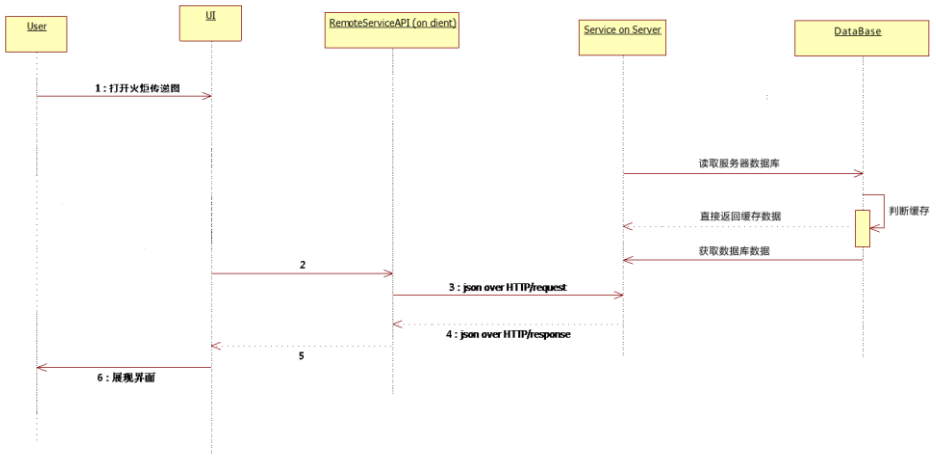
2.3.1 与后台服务器接口列表

- 获取答题接口：通过 JSON over HTTP 获取数据
- 获取神秘宝箱接口：通过 JSON over HTTP 获取数据

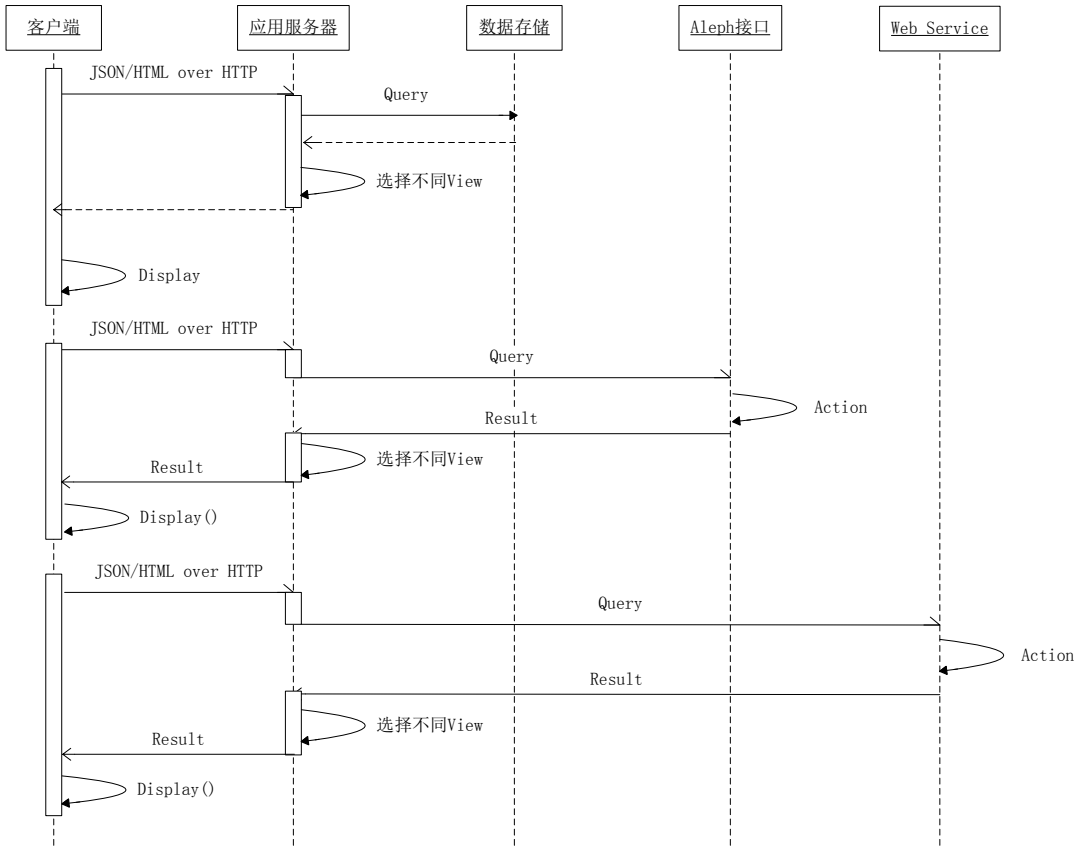
2.3.2 与神秘宝箱后台的交互

- 获取最新数据：客户端通过 HTTP 请求获取宝箱分布数据

- 清空缓存


User
UI
RemoteServiceAPI (on dient)
Service on Server
DataBase
1：打开火炬传递圈
读取服务器数据库
判断缓存
直接返回缓存数据
获取数据库数据
2
3：json over HTTP/request
4：json over HTTP/response
5
6：展现界面

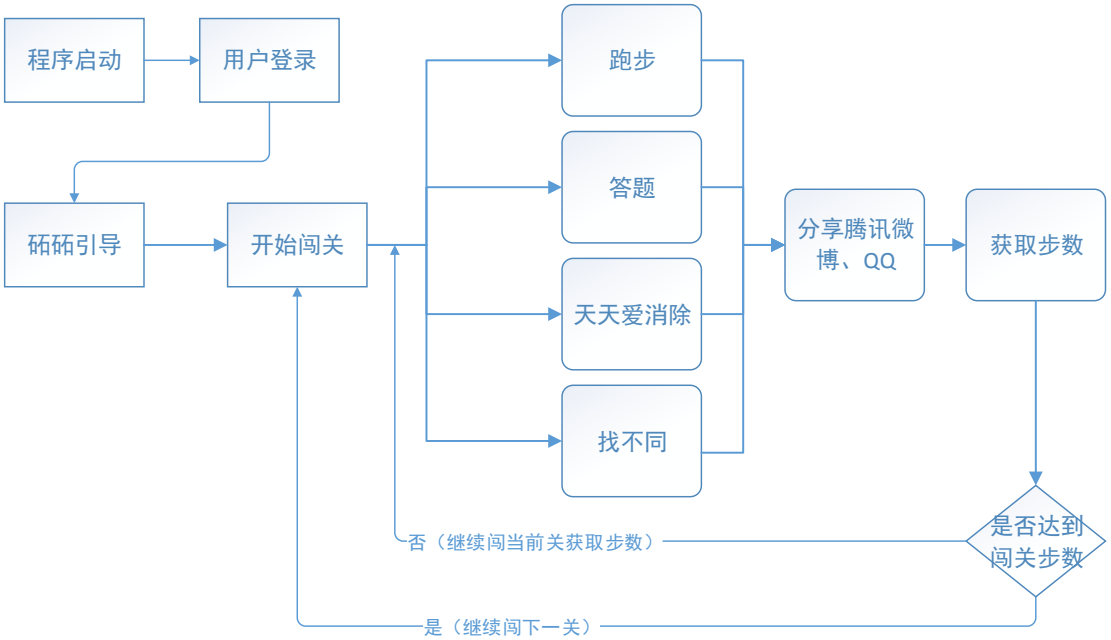
- 数据交互


客户端
应用服务器
数据存储
Aleph接口
Web Service
JSON/HTML over HTTP
Query
选择不同View
Display
JSON/HTML over HTTP
Query
Action
Result
Result
选择不同View
Display()
JSON/HTML over HTTP
Query
Action
Result
Result
选择不同View
Display()

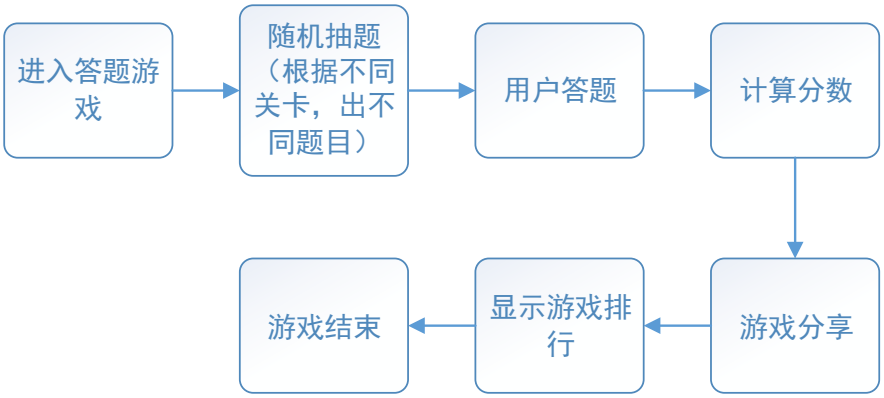
2.3.3 腾讯微博 API

2.4 重要流程

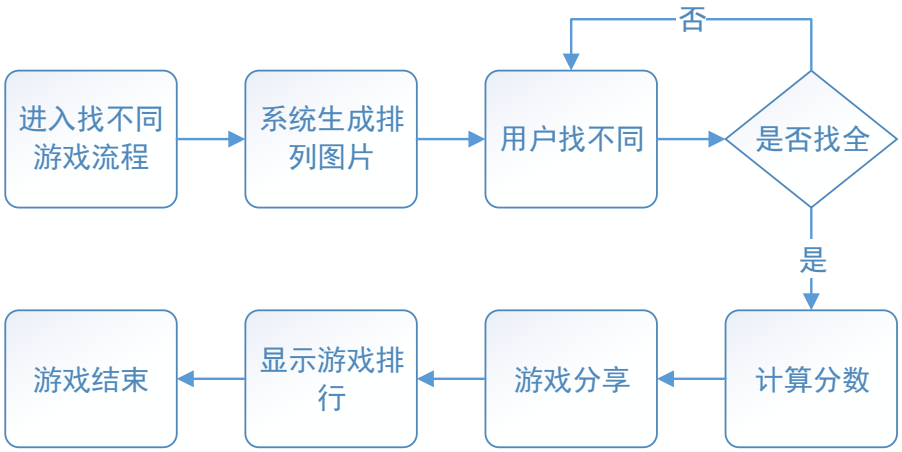
2.4.1 整体流程



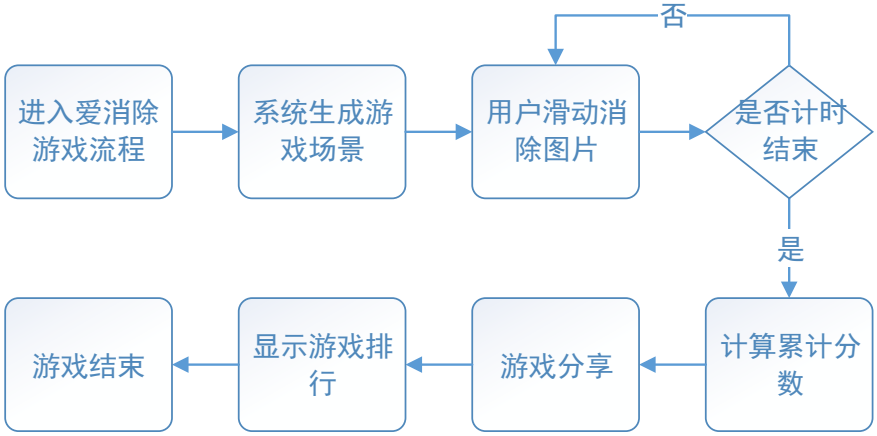
2.4.2 答题游戏流程



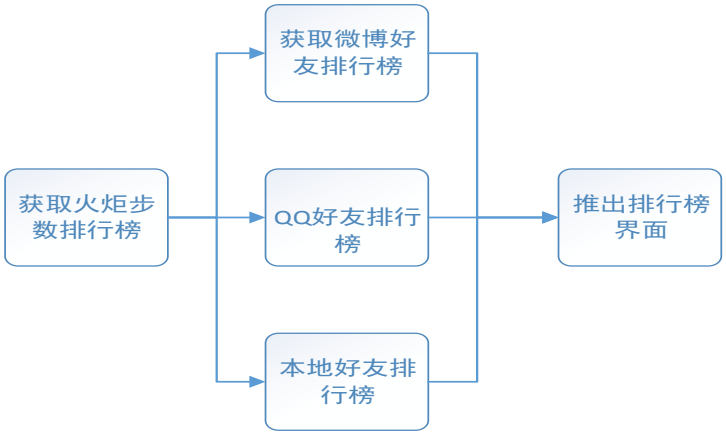
2.4.3 找不同游戏流程



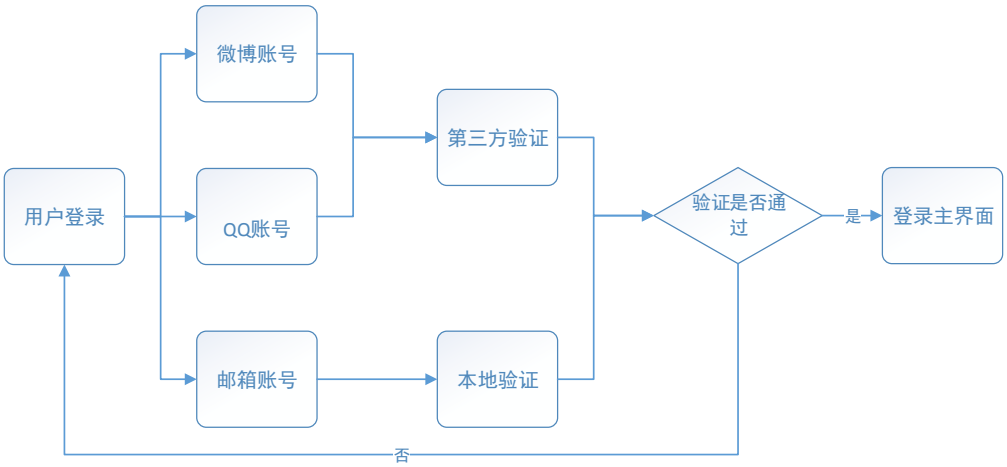
2.4.4 爱消除游戏流程



2.4.5 获取火炬步数排行榜流程



2.4.6 登陆流程



3 服务器设计

3.1 应用接口服务系统

3.1.1 功能设计

- 微博粉丝，获取微博用户的粉丝列表数据接口；
- 微博好友，获取微博用户好友列表数据接口；
- 分享，获取分享到社交平台的分享内容接口；
- 登录，验证客户端用户的信息接口；
- 注册，通过客户端注册用户信息接口；
- 保存步数，保存用户的火炬步数接口；
- 获取步数，返回用户的火炬步数接口；
- 获取宝箱，获取宝箱的数据接口；
- 获取答题，获取题目列表数据接口；

3.1.2 模块划分

- 用户管理模块；

- 分享模块；
- 宝箱管理模块；
- 题目管理模块；
- 步数统计模块；

3.2 后台管理系统

3.2.1 功能设计

- 用户信息，存储用户的相关信息。
- 宝箱信息，配置客户端显示神秘宝箱信息。
- 答题信息，管理客户端答题信息。
- 步数统计，统计用户的火炬步数以及目前所达到的奥运城市。

3.2.2 模块划分

- 用户管理模块；
- 宝箱管理模块；
- 题目管理模块；
- 步数统计模块；
- 系统配置模块；
- 信息推送模块；

3.3 性能评估

与并发用户数相关的概念还包括“并发用户数”、“系统用户数”和“同时在线用户数”，下面用一个实际的例子来说明它们之间的差别。

假设有一个系统有 1000 万个使用用户——这就是说，可能使用该 OA 系统的用户总数是 1000 万名，这个概念就是“系统用户数”，该系统有一个“在线统计”功能（系统用一个全局变量记数所有已登录的用户），从在线统计功能中可以得到，最高峰时有 200 万人在线（这个 200 万就是一般所说的“同时在线人数”），那么，系统的并发用户数是多少呢？

根据我们对业务并发用户数的定义，这 200 万就是整个系统使用时最大的业务并发用户数。当然，200 万这个数值只是表明在最高峰时刻有 200 万个用户登录了系统，并不表示实际服务器承受的压力。因为服务器承受的压力还与具体的用户访问模式相关。例如，在这 200 万个“同时使用系统”的用户中，考察某一个时间点，在这个时间上，假设其中 40%的用户在较有兴致地看系统公告（注意：“看”这个动作是不会对服务端产生任何负担的），20%的用户在填写复杂的表格（对用户填写的表格来说，只有在“提交”的时刻才会向服务端发送请求，填写过程是不对服务端构成压力的），20%部分用户在发呆（也就是什么也没有做），剩下的 20%用户在不停地从一个页面跳转到另一个页面——在这种场景下，可以说，只有 20%的用户真正对服务器构成了压力。因此，从上面的例子中可以看出，服务器实际承受的压力不只取决于业务并发用户数，还取决于用户的业务场景。

在实际的性能测试工作中，测试人员一般比较关心的是业务并发用户数，也就是从业务角度关注究竟应该设置多少个并发数比较合理，因此，在后面的讨论中，也是主要针对业务并发用户数进行讨论，而且，为了方便，直接将业务并发用户数称为并发用户数。

（1） 计算平均的并发用户数： C = nL/T

（2） 并发用户数峰值： C’≈C+3 根号 C

公式（1）中，C 是平均的并发用户数；n 是 login session 的数量；L 是 login session 的平均长度；T 指考察的时间段长度。

公式（2）则给出了并发用户数峰值的计算方式中，其中，C’指并发用户数的峰值，C 就是公式（1）中得到的平均的并发用户数。该公式的得出是假设用户的 login session 产生符合泊松分布而估算得到的。

实例：

假设有一个系统，该系统有 3000 个用户，平均每天大约有 400 个用户要访问该系统，对一个典型用户来说，一天之内用户从登录到退出该系统的平均时间为 4 小时，在一天的时间内，用户只在 8 小时内使用该系统。

则根据公式（1）和公式（2），可以得到：

C = 400*4/8 = 200

C’≈200+3*根号 200 = 242

虚拟火炬 APP 并发情况介绍

假设系统有 2000 万用户,平均一天有 200 万用户访问系统,对一个典型用户,一天只能从登陆到退出系统平均时间为 1 小时,在一天时间内,用户只在 8 个小时内使用该系统.则根据公

式计算:

（1） 计算平均的并发用户数： C = nL/T

（2） 并发用户数峰值： C’≈C+3*根号 C

则平均的并发用户数是：c=2000000*1/8=250000

并发用户数峰值： C’≈251500

一旦我们找出了并发用户数，其他的一些系统属性值也可以由它推出。

对于 Web 应用，请求率（比如说单位时间内的请求数，有时候视为点击率）是另一个进行容量规划的重要因素。如果可以通过样本得到用户的平均数值是 r，那么可以容易得出：

如果平均每个用户每 5 分钟发起 1 个请求，那高峰时平均总的请求率就大概是 62875 请求/分钟，即 1048 请求/秒。以每台服务器并发请求数 400 计算，需要 3 台应用服务器。具体每分钟请求次数需最终商讨后确定。

影响的参数包括：

高峰时平均总的请求率/分钟 =(nL/T +3*根号(nL/T))*R

n 是每天平均使用系统的人数；L 是平均使用长度；T 指考察的时间段长度。R 为平均每个用户每分钟发起的请求数。

4 数据库设计

关键表结构数据字典说明

APPHOMECONFIG

描述：首页配置

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	title	标题				
2	code	编码				
4	url	链接				
5	sort	排序				
6	func_imag	背景图片				
7	updateDate	修改时间			√	

APPHOMETOPCONFIG

描述:首页顶部配置

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	code	编码				
2	url	链接				
4	sort	排序				
5	func_imag	背景图片				
6	updateDate	修改时间				

USERS

描述：参与用户表

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	Type	1.腾讯微博 2.QQ 3.Email 注册				
2	Nick_Name	昵称				
4	Token	第三方平台 token				
5	User_Indentiy	第三方标识 1.来源为腾讯微博对应腾讯微博 ID 2.来源为 QQ 时对应 QQ 号 3.来源为 Email 注册时,对应 Email				
6	Head_Img	头像, 用户头像地址（中图），50×50 像素				
7	Password	密码				
8	Country	所在国家				
9	Province	所在州、省				
10	City	所在城市				
11	Gender	性别，m：男、f：女、n：未知				
12	description	签名				
13	created_at	注册时间				
14	birth_day	出生天				
15	birth_month	出生月				
16	birth_year	出生年				
17	edu	教育情况				
18	IsTorch	是否有火炬。				

User_Game_Diamond_Gash

描述：记录用户天天爱消除游戏相关信息

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	USERID	关联用户表 ID				
2	PlayBeginTime	游戏开始开始时间				
4	PlayEndTime	游戏结束时间				
5	Score	分数				
6	GateNumber	用户所创关数				
7	OlympicCountry	用户玩此游戏时所在的火炬传递地点				

User_Game_Photo_Hunt
描述：记录用户找不同游戏相关信息

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	USERID	关联用户表 ID				
2	PlayBeginTime	游戏开始时间				
4	PlayEndTime	游戏结束时间				
5	Score	分数				
6	GateNumber	用户所创关数				
7	OlympicCountry	用户玩此游戏时所在的火炬传递地点				

User_Game_Run
描述：记录用户跑步游戏相关信息

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	USERID	关联用户表 ID				
2	PlayBeginTime	跑步开始时间				
4	PlayEndTime	跑步结束时间				
5	Score	步数				
6	BeginLongitude	开始经度				
7	BeginLatitude	开始纬度				
8	EndLongitude	结束经度				
9	EndLatitude	结束纬度				
10	OlympicCountry	用户玩此游戏时所在的火炬传递地点				

User_Quiz
描述：记录用户答题记录

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	USERID	关联用户表 ID				
2	PlayBeginTime	答题开始时间				
4	PlayEndTime	答题结束时间				
5	Score	得分				
6	OlympicCountry	用户答题时所在的火炬传递地点				

Quiz_Detail
描述：用户答题详细情况

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值

1	User_QuizID	关联 User_Quiz 的 ID				
2	QuizID	所答题目				
4	QuizAnswer	用户答案				

Quiz_Info
描述：系统题题库

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	id	Id 标识				
2	title	试题名称				
4	OptionA	选择 A				
5	OptionB	选择 B				
6	OptionC	选择 C				
7	OptionD	选择 D				
8	OlympicCountry	该题目对应的奥运国家				
9	Answer	该题目正确答案				

备注：判断题只有 OptionA 和 OptionB 两个

User_Participant
描述：用户参与情况

序号	字段名称	字段描述	字段类型	长度	允许空	缺省值
1	UserID	关联用户基本表主键				
2	OlympicCountry	用户当前火炬传递所在的国家				
4	Torch_Relay	火炬传递时间				
5	IsHistory	0 表示是历史 1 表示是当前地点				

5 客户端安全问题

苹果 iOS 是一个封闭的系统它保护了用户免受恶意软件的攻击，目前移动恶意软件的主要目标就是 Android 平台。Android 上的流行软件普遍存在安全缺陷或安全漏洞。漏洞频发的原因可能有很多，将从数据存储、网络通信、密码和认证策略这三个角度介绍 Android 软件中比较常见的安全缺陷或安全漏洞，分析产生问题的原因，介绍可能的攻击方法，并给出解决问题的建议。

5.1 数据存储

Android 软件可以使用的存储区域分为外部（SD 卡）和内部（NAND 闪存）两种。除

了大小和位置不同之外，两者在安全权限上也有很大的区别。外部存储的文件没有读写权限的管理，所有应用软件都可以随意创建、读取、修改、删除位于外部存储中的任何文件，而仅仅需要申明 READ_EXTERNAL_STORAGE 和 READ_EXTERNAL_STORAGE 权限。内部存储则为每个软件分配了私有区域，并有基于 Linux 的文件权限控制，其中每个文件的所有者 ID 均为 Android 为该软件设立的一个用户 ID。通常情况下，其他软件无权读写这些文件。

关于数据存储可能出现的问题包括以下几种。

问题 1：将隐私数据明文保存在外部存储

例如，聊天软件或社交软件将聊天记录、好友信息、社交信息等存储在 SD 卡上；备份软件将通信录、短信等备份到 SD 卡上等。如果这些数据是直接明文保存（包括文本格式、XML 格式、SQLite 数据库格式等）的，那么攻击者写的软件可以将其读取出来，并回传至指定的服务器，造成隐私信息泄露。

解决办法：较好的做法是对这些数据进行加密，密码保存在内部存储，由系统托管或者由用户使用时输入。

问题 2：将系统数据明文保存在外部存储

备份软件和系统辅助软件可能将用户已安装的其他软件数据保存至 SD 卡，以便刷机或升级后进行恢复等；或者将一些系统数据缓存在 SD 卡上供后续使用。同样的，如果这些数据是明文保存的，恶意软件可以读取它们，有可能用于展开进一步的攻击。

问题 3：将软件运行时依赖的数据保存在外部存储

如果软件将配置文件存储在 SD 卡上，然后在运行期间读取这些配置文件，并根据其中的数据决定如何工作，也可能产生问题。攻击者编写的软件可以修改这些配置文件，从而控制这些软件的运行。例如，如果将登录使用的服务器列表存储在 SD 卡中，修改后，登录连接就会被发往攻击者指定的服务器，可能导致账户泄露或会话劫持（中间人攻击）。解决办法：对这种配置文件，较安全的方法是保存到内部存储；如果必须存储到 SD 卡，则应该在每次使用前判断它是否被篡改，例如，与预先保存在内部的文件哈希值进行比较。

问题 4：将软件安装包或者二进制代码保存在外部存储现在很多软件都推荐用户下载并安装其他软件；

用户点击后，会联网下载另一个软件的 APK 文件，保存到 SD 卡然后安装。也有一些软件为了实现功能扩展，选择动态加载并执行二进制代码。例如，下载包含了扩展功能的 DEX 文件或 JAR 文件，保存至 SD 卡，然后在软件运行时，使用 dalvik.system.DexClassLoader

类或者 java.lang.ClassLoader 类加载这些文件，再通过 Java 反射，执行其中的代码。如果在安装或加载前，软件没有对 SD 卡上的文件进行完整性验证，判断其是否可能被篡改或伪造，就可能出现安全问题。攻击者可以使用称为“重打包”（re-packaging）的方法。目前大量 Android 恶意代码已采用这一技术。重打包的基本原理是，将 APK 文件反汇编，得到 Dalvik 指令的 smali 语法表示；然后在其中添加、修改、删除等一些指令序列，并适当改动 Manifest 文件；最后，将这些指令重新汇编并打包成新的 APK 文件，再次签名，就可以给其他手机安装了。通过重打包，攻击者可以加入恶意代码、改变软件的数据或指令，而软件原有功能和界面基本不会受到影响，用户难以察觉。如果攻击者对软件要安装的 APK 文件或要加载的 DEX、JAR 文件重打包，植入恶意代码，或修改其原始代码；然后在 SD 卡上，用其替换原来的文件，或者拷贝到要执行或加载的路径，当软件没有验证这些文件的有效性时，就会运行攻击者的代码。攻击结果有很多可能，例如直接发送扣费短信，或者将用户输入的账户密码发送给指定的服务器，或者弹出钓鱼界面等。

解决办法：软件应该在安装或加载位于 SD 卡的任何文件之前，对其完整性做验证，判断其与实现保存在内部存储中的（或从服务器下载来的）哈希值是否一致。

问题 5：全局可读写的内部文件

如果开发者使用 openFileOutput(String name,int mode)方法创建内部文件时，将第二个参数设置为 Context.MODE_WORLD_READABLE 或 Context.MODE_WORLD_WRITEABLE，就会让这个文件变为全局可读或全局可写的。开发者也许是为了实现不同软件之间的数据共享，但这种方法的问题在于无法控制哪个软件可以读写，所以攻击者编写的恶意软件也拥有这一权限。如果要跨应用共享数据，一种较好的方法是实现一个 Content Provider 组件，提供数据的读写接口，并为读写操作分别设置一个自定义权限。

问题 6：内部敏感文件被 root 权限软件读写

如果攻击者的软件已获得 root 权限，自然可以随意读写其他软件的内部文件。这种情况并不少见。大量的第三方定制 ROM 提供了 root 权限管理工具，如果攻击者构造的软件伪造成一些功能强大的工具，可以欺骗用户授予它 root 权限。即便手机安装的官方系统，国内用户也大多乐于解锁、刷 recovery 并刷入 root 管理工具。在 Android 2.2 和 2.3 中，存在一些可以用于获取 root 权限的漏洞，并且对这种漏洞的利用不需要用户的确认。

解决办法：因此，我们并不能假设其他软件无法获取 root 权限。即便是存在内部的数据，依然有被读取或修改的可能。

前面提到，重要、敏感、隐私的数据应使用内部存储，现在又遇到 root 后这些数据依

然可能被读取的问题。如果攻击者铤而走险获得 root 权限（被用户觉察或者被安全软件发现的风险），那理论上他已拥有了系统的完整控制权，可以直接获得联系人信息、短信记录等。此时，攻击者感兴趣的软件漏洞利用更可能是获得其他由软件管理的重要数据，例如账户密码、会话凭证、账户数据等。例如，早期 Google 钱包将用户的信用卡数据明文存储，攻击者获取这些数据后，可以伪装成持卡人进行进一步攻击以获得账号使用权。这种数据就是“其他由软件管理的重要数据”。所以关于数据存储这个问题并没有通用的解决方法。需要根据实际情况寻找方案，并在可用性与安全性之间做出恰当的选择。

5.2 网络通信

Android 软件通常使用 WiFi 网络与服务器进行通信。WiFi 并非总是可信的。例如，开放式网络或弱加密网络中，接入者可以监听网络流量；攻击者可以自己设置 WiFi 网络钓鱼。此外，在获得 root 权限后，还可以在 Android 系统中监听网络数据。

问题 1：不加密地明文传输敏感数据

最危险的是直接使用 HTTP 协议登录账户或交换数据。例如，攻击者在自己设置的钓鱼网络中配置 DNS 服务器，将软件要连接的服务器域名解析至攻击者的另一台服务器；这台服务器就可以获得用户登录信息，或者充当客户端与原服务器的中间人，转发双方数据。早期，国外一些著名社交网站的 Android 客户端的登录会话没有加密。后来出现了黑客工具 FaceNiff，专门嗅探这些会话并进行劫持（它甚至支持在 WEP、WPA、WPA2 加密的 WiFi 网络上展开攻击！）。

解决方法：很显然—对敏感数据采用基于 SSL/TLS 的 HTTPS 进行传输。

问题 2：SSL 通信不检查证书有效性

在 SSL/TLS 通信中，客户端通过数字证书判断服务器是否可信，并采用证书中的公钥与服务器进行加密通信。然而，有开发者在代码中不检查服务器证书的有效性，或选择接受所有的证书。例如，开发者可以自己实现一个 X509TrustManager 接口，将其中的 checkServerTrusted 方法实现为空，即不检查服务器是否可信；或者在 SSLSocketFactory 的实例中，通过 setHostnameVerifier(SSLSocketFactory. ALLOW_ALL_HOSTNAME_VERIFIER)，接受所有证书。做出这两种选择的可能原因是，使用了自己生成了证书后，客户端发现证书无法与系统可信根 CA 形成信任链，出现了 CertificateException 等异常。这种做法可能导致的问题是中间人攻击。在钓鱼 WiFi 网络中，同样地，攻击者可以通过设置 DNS 服务器

使客户端与指定的服务器进行通信。攻击者在服务器上部署另一个证书，在会话建立阶段，客户端会收到这张证书。如果客户端忽略这个证书的异常，或者接受这个证书，就会成功建立会话、开始加密通信。但攻击者拥有私钥，因此可以解密得到客户端发来数据的明文。攻击者还可以模拟客户端，与真正的服务器联系，充当中间人做监听。

解决方法：是从可信 CA 申请一个证书。但在移动软件开发中，不推荐这种方法。除了申请证书的时间成本和经济成本外，这种验证只判断了证书是否 CA 可信的，并没有验证服务器本身是否可信。例如，攻击者可以盗用其他可信证书，或者盗取 CA 私钥为自己颁发虚假证书，这样的攻击事件在过去两年已有多次出现。事实上，移动软件大多只和固定的服务器通信，因此可以在代码中更精确地直接验证是否某张特定的证书，这种方法称为“证书锁定”（certificate pinning）。实现证书锁定的方法有两种：一种是前文提到的实现 X509TrustManager 接口，另一种则是使用 KeyStore。具体可参考 Android 开发文档中 HttpsURLConnection 类的概览说明。

问题 3：使用短信注册账户或接收密码

也有软件使用短信进行通信，例如自动发送短信来注册、用短信接收初始密码、用短信接收用户重置的密码等。短信并不是一种安全的通信方式。恶意软件只要申明了 SEND_SMS、RECEIVE_SMS 和 READ_SMS 这些权限，就可以通过系统提供的 API 向任意号码发送任意短信、接收指定号码发来的短信并读取其内容，甚至拦截短信。这些方法已在 Android 恶意代码中普遍使用，甚至 2011 年就已出现拦截并回传短信中的网银登录验证码（mTANs）的盗号木马 Zitmo。因此，这种通过短信注册或接收密码的方法，可能引起假冒注册、恶意密码重置、密码窃取等攻击。此外，这种与手机号关联的账户还可能产生增值服务，危险更大。较好的实现方式还是走 Internet。

5.3 密码和认证策略

问题 1：明文存储和编码存储密码

许多软件有“记住密码”的功能。如果开发者依字面含义将密码存储到本地，可能导致泄漏。另外，有的软件不是直接保存密码，而是用 Base64、固定字节或字符串异或、ProtoBuf 等方法对密码编码，然后存储在本地。这些编码也不会增加密码的安全性。采用 smali、dex2jar、jd-gui、IDA Pro 等工具，攻击者可以对 Android 软件进行反汇编和反编译。攻击者可以借此了解软件对密码的编码方法和编码参数。

解决办法：使用基于凭据而不是密码的协议满足这种资源持久访问的需求，例如 OAuth 对外服务的弱密码或固定密码。另一种曾引起关注的问题是，部分软件向外提供网络服务，而不使用密码或使用固定密码。例如，系统辅助软件经常在 WiFi 下开启 FTP 服务。部分软件对这个 FTP 服务不用密码或者用固定密码。在开放或钓鱼的 WiFi 网络下，攻击者也可以扫描到这个服务并直接访问。还有弱密码的问题。例如，早期 Google 钱包的本地访问密码是 4 位数字，这个密码的 SHA256 值被存储在内部存储中。4 位数字一共只有 10000 种情况，这样攻击软件即便是在手机上直接暴力破解，都可以在短时间内获得密码。

问题 2：使用 IMEI 或 IMSI 作为唯一认证凭据

IMEI、IMSI 是用于标识手机设备、手机卡的唯一编号。如果使用 IMSI 或 IMEI 作为用户认证的唯一凭据，可能导致假冒用户的攻击。首先，应用要获取手机的 IMEI、手机卡的 IMSI 并不需要特殊权限。事实上，许多第三方广告库回传它们用于用户统计。其次，得到 IMEI 或 IMSI 后，攻击者有多种方法伪造成用户与服务器进行通信。例如，将原软件重打包，使其中获取 IMEI、IMSI 的代码始终返回指定的值；或修改 Android 代码，使相关 API 始终返回指定的值，编译为 ROM 在模拟器中运行；甚至可以分析客户端与服务器的通信协议，直接模拟客户端的网络行为。因此，若使用 IMEI 或 IMSI 作为认证的唯一凭据，攻击者可能获得服务器中的用户账户及数据。