密级：　　保密期限：

北京郵電大學

硕士学位论文

题目：基于参与式感知的移动互联网应用的设计与实现

学　　号：2012140754

姓　　名：薛潇剑

专　　业：计算机技术

导　　师：王文东

学　　院：网络技术研究院

2014 年 12 月 20 日

声明

独创性（或创新性）声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：_____ 日期：_____

关于论文使用授权的说明

学位论文作者完全了解北京邮电大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属北京邮电大学。学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。

本学位论文不属于保密范围，适用本授权书。

本人签名：_____ 日期：_____

导师签名：_____ 日期：_____

基于参与式感知的移动互联网应用的设计与实现

摘　要

随着移动互联网、物联网和社交网络的发展，智能手机和平板电脑逐渐成为人类生活中不可或缺的一部分。对于智能手机而言，已经不再单单是打电话、发短信这些简单的功能，越来越多的感知系统被加入到智能手机当中。但目前感知系统大部分是用在应用程序当中，是被动的。而人们参与的感知呼声越来越高。在这些前提下，本文明确了课题的研究目标，设计并开发了一款基于参与式感知的 Android 平台上的移动互联网应用。

在本文中，首先介绍了参与式感知的研究背景和研究现状，以及 Android 操作系统的相关技术背景；随后，针对本课题的目标进行了详细的需求分析，明确了参与式感知应用的各个功能；接着对各个模块进行了详细的任务分工；然后，本文针对控制模块、数据管理模块和用户接口模块进行了系统的概要设计和详细设计，结合用例图、流程图、时序图和类图等多种方式，描述了各个模块的数据结构定义，流程实现和功能实现；最后，对实现的系统进行了测试和效果演示，并在此基础上对整个论文进行总结，提出了参与式感知应用可以进一步优化的地方。

本文设计的系统有一个优势，就是在开始设计的时候，就考虑到功能的添加等扩展性，采用了抽屉式布局来进行设计搭建，并用了多种设计模式，方便后来开发的人进行扩展和设计。

关键词： Android 操作系统 参与式感知 手机应用开发 框架搭建

DESIGN AND IMPLEMENTATION OF PARTICIPATE SENSING SYSTEM BASED ON MOBILE INTERNET

ABSTRACT

With the development of mobile Internet, Internet of things and social networks, intelligent mobile phone and tablet computer has gradually become an integral part of human life. For the intelligent mobile phones, the function of them is no longer not just have a call or send a text message to other people, more and more sensory systems have been added to the intelligent mobile phones. But the current sensing system is mostly used in the application passively, and the people more and more want to be involved in. In this premise, the mission of this thesis is clear: to design and develop an application based on participate sensing on Android platform.

First of all, this thesis introduces the background and present research situation of participate sensing, and the technical background of Android system. Secondly, according to the characteristics of the system, define the requirement and the function of participate sensing system clearly. Then we divide our tasks according to the modules. Thirdly, we introduce the modules that we really take part in, and show the brief design and detail design, combined with use cases diagram, flow chart, sequence diagram, class diagram. We also show the definition of data structure and the realization of the functions and process. At last, based on the software testing, the thesis sums up the whole paper, and shows where we can further optimize the participate sensing application.

The advantage the system has is that, we consider a lot of augmentability such as adding new function. We use a variety of framework to design and

build, to make later development more convenient.

KEY WORDS Android System, Participate sensing, Mobile Application Design, Framework Building

目录

第一章 绪论 1
1.1 研究背景 1
1.2 当前研究现状 2
1.3 课题任务 2
1.4 论文的章节安排 3
第二章 Android 平台参与式感知应用的背景介绍 5
2.1 Android 操作系统 5
2.1.1 Android 操作系统介绍 5
2.1.2 Android 体系结构 6
2.1.3 Android 环境下的编程开发 8
2.2 移动端的感知体系 9
2.3 移动端的网络通信方式 10
第三章 Android 平台参与式感知应用的需求分析 11
3.1 功能性需求 11
3.1.1 框架搭建需求 12
3.1.2 数据管理需求 14
3.1.3 数据可视化需求 15
3.1.4 可配置需求 16
3.1.5 国际化需求 16
3.2 非功能性需求 17
第四章 Android 平台参与式感知应用的概要设计 19
4.1 Android 平台参与式感知应用的总体设计 19
4.2 控制模块设计 20
4.2.1 设置模块设计 21
4.2.2 出错管理模块设计 23
4.2.3 流程管理模块设计 23
4.2.4 日志模块设计 25
4.3 用户接口模块设计 25
4.3.1 主动采集接口设计 25
4.3.2 信息展示接口设计 28
4.3.3 其他接口设计 31

第五章 Android 平台参与式感知应用的详细设计 33
5.1 数据管理模块设计 33
5.1.1 数据库模块 34
5.1.2 文件模块 37
5.1.3 系统函数模块 38
5.2 控制模块设计 39
5.2.1 设置模块设计 39
5.2.2 出错管理模块设计 41
5.2.3 流程管理模块设计 43
5.2.4 日志模块设计 46
5.3 用户接口模块设计 47
5.3.1 主动采集接口设计 48
5.3.2 信息展示接口设计 49
5.3.3 其他接口设计 53
第六章 Android 平台参与式感知应用的测试与效果展示 55
6.1 测试与验证 55
6.1.1 测试与开发环境 55
6.1.2 单元测试 56
6.1.3 功能测试 56
6.1.4 性能测试 57
6.1.5 测试总结 58
6.2 效果展示 59
6.2.1 数据部分 59
6.2.2 PM2.5 部分 59
6.2.3 照片墙部分 60
第七章 总结与展望 63
7.1 论文工作总结 63
7.2 问题与展望 63
参考文献 65
致 谢 67

第一章　绪论

1.1 研究背景

近年来，随着移动互联网、物联网和社交网络的发展，智能手机和平板电脑逐渐成为人类生活中不可或缺的一部分。而智能手机，正在成为人与人之间交流的主要平台之一。无论是在餐桌上，休息前，上班途中，大部分人都会使用手机进行娱乐消遣。早在 2013 年，中国的手机用户已经超过了 10 亿，智能手机用户占一半以上，也就是说，中国基本上每两个人，就会有一个人使用的是智能手机。而对于智能手机，不论是现在流行的 IOS 操作系统，还是 Android 操作系统，有了很多的方式可以对周围的信息进行采集。而在这个背景下，智能手机快速发展，开发商进行了多种尝试，使得手机已经不仅仅是一个纯粹的通信设备，而是在内部，拥有通信、计算、感知等多种功能的设备。这就为参与式感知提供了良好的硬件条件。

对于传感网络，传统的设备非常简单，他们是由固定的传感器组成。这些固定的设备对信息进行搜集，然后通过固定的网络上传到某个节点。然后通过这些固定地方的数据来进行分析。然而，随着当前互联网络的快速发展，智能设备上传感器的多样化，现有的依靠智能设备来进行数据收集，使采集数据更加方便、智能，可以无时无刻的传递信息。传感的感念正经历着从传统的固定传感器到以智能穿戴设备为代表的微型传感器这样一个不断进化的发展过程，人的“参与”越来越多。因此，“参与式感知”[19]的概念也应运而生。

“参与式感知”，就是参与更多一些，人的参与更加重要。对于感知得到的数据，都是由“人”来主动进行创建，然后通过选择上传到网络上。也就是说“参与式感知”以人为主，用户出于个人或经济兴趣，有意识的响应感知需求，用户既是数据的提供者又是数据的消费者。这也与普通的“众包”有所区别，可以认为参与式感知是一种结合了传感数据的采集、移动应用和用户主动参与的一种模式。因为随着智能手机的普及，当前情况下用户用于共享数据的设备一般都是智能手机，而共享的途径则普遍是移动互联网。而人的需求不断增加，更加想要体验到周围的情况，越来越想要感受到周围的环境，越来越多的上下文相关服务迅猛发展，各种新的人机交互方式层出不穷。为满足这一用户需求，越来越多的传感器件和传感技术被运用到智能手机上，催生了一系列基于智能手机感知的应用，并掀起了移动感知的研究热潮。

1.2 当前研究现状

由于智能手机的快速发展，越来越多的传感器件和传感技术被应用到智能手机上，因此，当前也诞生了好多根据这些传感设备或技术开发的一些应用，以及对这些设备进行的一些研究等。

应用最广泛的是基于位置的服务（Location Based Service，LBS）,各种地图应用层出不穷，应用潜力巨大，各大互联网公司都在积极开展 LBS 技术研究及业务开发，类似“查找附近好友”、“查找附近加油站”等 LBS 功能应用层出不穷，但这些系统构建相对独立，数据管理困难，用户隐私，室内 LBS 等问题也会出现，这些问题也需要来慢慢解决[1]。

对于传感器的应用，不只是特定的位置，还可能会根据人的位置移动来进行交互服务。有些软件用这些记录用户轨迹，来预测人的行走方式，但这些都是被动的，而没有主动的交互，也因此称不上是“参与式”的感知。

除了地理位置这些 LBS 服务[17]，当前的学术工业界也有运用其他的感知技术和器件来达到某种目的的。

如对于照片[18]，近年来，用户越来越倾向于通过智能手机等移动设备拍照，这种方式就是运用了更加常见的传感器——照相机。但是目前这个应用领域还存在着很多的不足。如大部分只是进行拍照，稍微好一点的就是通过图像识别来进行图书阅读转化文字等[2]。但这些不能感受到周围的环境，只是能让生活更加方便等，还做不到参与式的感知，大部分是用来对自己进行一次生活质量的提升，让生活更加便捷。

对于声音，现在有了很多智能化的语音交互系统，小到语音传递聊天（如 QQ，微信等），大到语音识别，自动帮助系统（Sirl 等）。这些都是很好的感知系统使用典范。而对于其他感知系统，如光照，开发不多，只是有自动调整屏幕亮度等用于提高用户方便的功能。

感知系统有很多的研究方式，但是大多是服务为主。本课题主要研究的是“参与式感知”，主体为人，以人为本，来进行相应的操作，并给人一个良好的周边环境的反馈，从而达到研究“参与式感知”的目的。用来给科研等提供一个良好的探路平台。

1.3 课题任务

本课题的任务是，在 Android 平台上，开发一款应用，能够使用基本所有的现有 Android 市场常见的传感设备或技术，为用户提供相应的传感参数，使用户能够感知周围环境。并且，用户能够参与进来，通过与应用的交互，获得到更深一步的周围

信息。这款软件从没有到开发到第一个版本，都是本课题的任务。以下是涉及到的几个方面：

一是用户交互（User Interface，UI）界面。UI 设计则是指对软件的人机交互、操作逻辑、界面美观的整体设计。好的 UI 设计不仅是让软件变得有个性有品味，还要让软件的操作变得舒适、简单、自由、充分体现软件的定位和特点。在本文中，用户需要可以进行主动的采集，可以查看已经采集到的信息，并且，可以对采集到的信息进行统计，这些，都是一个 UI 负责承担的。为了使得用户操作方便，设计一些良好的接口，使得用户在不需要太多说明的情况下便可以方便使用。

二是构建一个良好的框架体系。在本课题中，有多个模块需要多人开发，这就需要有一个专门的模块进行沟通协调。譬如，使得数据管理模块能够方便的为 UI 提供查询接口，数据采集模块可以将采集到的信息方便的存储到数据管理模块中。而且，对于出错的部分，需要有一个错误管理模块，并且能够记录下日志，方便开发人员查阅。这些，都是框架需要做的内容。

三是一些功能相关的设计。主要是通过对需求的分析，进行了模块的划分。在此基础上，对数据管理模块，控制模块和用户接口模块进行了调研及分析。对于参与式感知，数据管理要做到为其他模块能够提供存储和查询的接口，使得其他模块在使用的过程中，感受不到内部的实现。控制模块需要协调各个模块之间的交流，并且能在程序异常的时候进行及时的处理，而且可以通过控制模块能够进行采集频率，界面显示等一些设置。用户接口模块能够向用户展示最近的周边生活环境，使得用户能够主动的采集并查看自己想要的信息，甚至可以查看周围的其他人在采集的信息。

在这款参与式感知应用的开发中，本人参与了需求分析，概要设计，详细设计及具体代码的实现与测试等工作，涉及到了软件开发的系统流程，框架搭建，界面设计等工作。

1.4 论文的章节安排

第一章绪论，介绍了本课题的背景提出，论述了当前课题的研究概况，及课题的研究背景及目的。

第二章是相关技术背景的介绍。主要介绍了主要的开发平台 Android 操作系统的体系结构，以及 Android 操作系统下如何编程开发，还有关于采集和通信的简单介绍，明确了本课题论文的技术背景。

第三章是需求分析。从功能性上和非功能性上，介绍了 Android 平台参与式感知应用的需求定义，在此基础上，明确了本课题应用程序需要具有的功能。

第四章是概要设计。细化了 Android 平台参与式感知应用的各个模块，以处理流程和消息序列着手，着重对控制模块和用户接口模块进行设计，细化了功能。

第五章是详细设计。通过类图等方式详细介绍了对数据管理模块，控制模块和用户接口模块，从功能实现数据结构等方面进行了说明。

第六章是测试以及效果展示，对需求分析的各种需求进行不同方面的测试，并在最后对一个最终成果进行了展示。

第七章主要介绍了项目成果，并提出了进一步的可以改进和提高的地方。

第二章　Android 平台参与式感知应用的背景介绍

本文的目标是实现一个 Android 平台下的参与式感知应用系统。基于这个目标，我们有必要了解一下 Android 相关的技术背景和编程开发模式、感知器采集以及其他实现相应功能需要的背景知识。

2.1 Android 操作系统

2.1.1 Android 操作系统介绍

Android 操作系统是基于 Linux 开发而来的操作系统，同 Linux 一样，是一个开源的操作系统。目前，该操作系统主要用于智能移动设备的开发，例如，手机、平板电脑。目前有很多基于 Android 的操作系统，但没有什么统一的中文，中国大陆地区较多人使用“安卓”或“安致”[3]。

Android 操作系统刚开始是由 Andy Rubin 进行开发的，主要用来支持智能手机。到了 2005 年被 Google 所收购。随后，Google 协同多家软硬件开发商和运营商，继续研发了 Android 操作系统，并且对其进行了改良，使之更加友善。按照 Google 一贯的作风，在不久后就对此开放了源代码供人参考。随着第一步以 Android 为操作系统的智能手机问世以来，越来越多的设备运用了此开源系统，不仅有平板电脑，还有电视、数码相机、游戏机等设备也使用了 Android 操作系统。随着人们使用的增多，Android 操作系统逐渐超过其他的智能设备的操作系统，位居世界第一的领先地位。2013 年的第四季度，Android 平台手机的全球市场份额已经达到 78.1%[3]。

Android 操作系统是一款独立于 IOS 和 Windows Mobile 之外的一款智能操作系统，目前已经成为最主流手机操作系统之一。

Android 操作系统是一个开源的操作系统，这个特性也导致了很多新的操作系统出现，如 MIUI 等。但这些都是基于 android 操作系统的，兼容并支持所有的 android 平台上的软件，因此，对于一款智能手机软件，在 android 上开发就能支持大部分的手机设备。

本课题基于 android4.x 平台。该系列的版本是最新的 android 版本，经过几年的沉淀，不仅功能十分强大，而且十分高效、稳定。相对于老的版本，有了很多新的功能，并且在界面等方面做了一些建议，代码上做了更便于界面编写的情况，更加适合软件的开发。

2.1.2 Android 体系结构

如上述所说，Android 操作系统的底层建立在 Linux 操作系统之上，该平台是由操作系统层、中间件层、用户界面层和应用软件层四层所组成。它是采用一种软件叠层（Software Stack）的方式进行构建。这种软件叠层结构可以使得每层之间相互不干扰，明确各自的分工。这种结构可以保证每层之间的耦合度尽可能的低，当一层的层内或层下产生变化后，上层应用程序不需要做出任何改变就可以适应其变化。Android 体系结构如下：

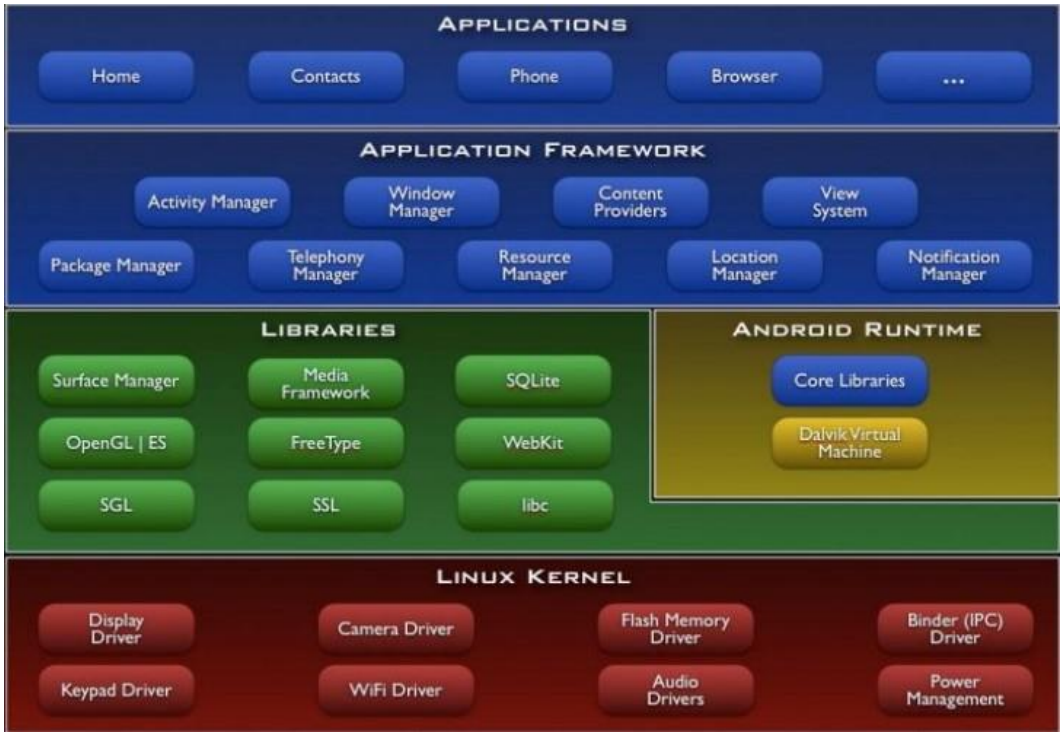

图 2-1 Android 系统的体系结构[4]

2.1.2.1Linux 内核

Android 操作系统建立在 Linux2.6 之上。对于 Linux 内核，其本身就提供了足够的安全性以及相关的进程管理、内存管理、网络协议和模型驱动等核心服务系统。除此之外，Linux 内核也是设备硬件和软件之间的桥梁[5]。在此基础上，Android 还做了一些修改，如电源管理等，但由于大部分不会在这一层开发，故不再详细描述。

2.1.2.2 系统运行库和 Android 运行环境

Android 操作系统包含了一套可以被底层应用程序所使用的 C/C++库。一般来说，Android 应用程序的开发者不能直接调用这些 C/C++库集，但是，可以通过这些层之上的应用程序框架来调用这些库。下面列出一些在本课题研究中常用的一些核心库

[5]：

- 界面管理(Surface Manager)：可以用来对一些显示相关的系统访问进行管理，可以对界面的 2D、3D 效果进行管理融合。
- 媒体库(Media Framework)：这个库是 Android 操作系统多媒体的核心部分，是 Android 多媒体对外的主要接口组成，包括图片、音频、视频等的操作和管理。
- SQLite：是一个数据库，在 Android 操作系统中，可以用来向应用程序提供的可以使用的轻量级关系型数据库。

还有其他的库就不在本文中一一介绍了。

而对于 Android 的运行环境，我们一般意义上指的主要是虚拟机技术。对于每一个 Android 应用程序来说，他们都属于一个虚拟机的实例，各自运行在自己所创建的进程当中。

2.1.2.3 应用程序框架

Android 的应用程序框架(Application Framework)为上层（应用程序层）的开发者提供 API，这样，可以使得开发者非常方便的应用下层的库来编写出极其丰富和新颖的应用程序。开发者可以通过应用程序框架，很方便的访问位置信息、运行后台服务、设置闹钟、向状态栏添加通知等功能。当然，这需要服从框架执行的一些安全限制。下面是主要使用的部分服务[6]：

- 活动管理器(Activity Manager)：管理应用程序生命周期，提供通用的导航回退功能。
- 内容提供者(Content Provider)：使应用程序能访问其他应用程序的数据或共享自己的一些数据。
- 视图(View System)：丰富的、可以扩展的视图集合，可以通过这些来构建一个简单的应用程序，非常重要。
- 资源管理器(Resource Manager)：提供访问一些非代码的资源，如图片，音频等。

通过这些服务，我们可以做一些应用程序而不需要考虑底层是怎么实现的，非常方便。

2.1.2.4 应用程序层

应用程序层(Applications)，顾名思义，就是一个提供各种应用程序运行的平台。包括主屏幕、联系人、电话、浏览器等众多的核心应用，大部分应用都是使用的 JAVA 语言进行开发的，且大多数的应用程序开发，都是在本层开发出来的。这层对外开

放，是一个非常有用的平台。

2.1.3 Android 环境下的编程开发

2.1.3.1 组件机制

Android 开发的时候，一般在应用程序层进行相应的开发。一个 Android 应用程序通常是由一个或者多个基本组件组成。在 Android 应用中，最常用的组件是 Activity，但是，还有许多其他的组件需要开发者知道，现在进行一个大概的介绍[5]。

- Activity。应用程序表示层。Activity 是 Android 应用中负责与用户进行交互的组件，是一个窗口式的容器。Activity 为 Android 应用提供了一个可视化的用户界面，对于一个 Android 应用程序，如果其包含多个界面，那么这个应用程序将会包含多个 Activity，多个 Activity 组成 Activity 栈，当前活动的 Activity 位于栈顶。
- Service。Service 就是我们所说的后台服务。他和 Activity 的地位是相同的，也代表着一个单独的 Android 组件。Service 与 Activity 不同之处在于：service 通常需要在后台运行，一般不需要与用户进行交互。因此，service 组件没有用户图形界面。
- Intent。严格的说，intent 并不属于 Android 应用的组件。但是，它对于 Android 应用程序的作用非常大，因为它是 Android 应用程序内不同组件之间跳转的通信载体。当 Android 应用程序在运行的时候需要在不同组件之间进行跳转的时候，通常就需要 intent 来实现。Intent 可以启动应用中另一个 Activity 或者 Service 组件。也就是说，intent 可以作为任意两个组件之间的载体，只是不同的组件使用 intent 的机制稍微不同而已。

还有一些其他的组件，如 ContentProvider、BroadcastReciever 等，由于本课题不经常用，故不再进行详细的介绍了。

2.1.3.2 消息处理机制

Android 的系统采用了分层系统的思想，在上个小章节已经进行了简单的介绍。而 Android 的消息处理机制，就是应用程序框架层提供的系统服务功能。Android 的消息处理机制，主要包括以下五个方面：

- Message。消息。这个可以理解为线程间通讯的数据单元。比如，后台在处理完自己的任务后，需要对用户界面进行更新，则可以发送一个更新的 message 给用户界面所处的线程。
- Message Queue。消息队列。是用来存放通过 Handler 发布的消息，为一个队

列，符合先进先出的原则。

- Handler。处理者。属于 Message 的主要处理者。负责对消息队列里的消息进行添加，并负责对从队列里取出的消息进行处理。
- Looper。循环器。是消息队列和处理者之间的桥梁，可以从消息队列中循环取出消息，并且交付给对应的处理者去处理。
- 线程。每一个线程里面可以含有一个循环器对象以及一个消息队列结构。例如，主线程一般是界面线程，在主线程启动的时候，启动程序会替主线程建立一个消息队列，以此可以进行消息的管理。

2.1.3.3 权限机制

对于 Android 操作系统来说，他是应用到手机等移动端的操作系统。这就涉及到一个权限的问题。不像 PC 客户端的操作系统，正常情况下，操作系统可以拥有 PC 操作的所有权限。而对于 Android 操作系统，本身最基本的是个移动操作系统，这就涉及到 PC 端没有的一些功能，如短信，电话本等私人信息。因此，Android 操作系统需要更加强大的权限机制。

对于 Android 操作系统下的编程开发，需要用到哪些权限，都必须要在一个 Mainifest.xml 的文件内声明，这样，在使用的过程中，才能调用对应权限，而在安装应用程序的时候，用户可以观察到该应用程序需要哪些操作系统的权限，以此来决定是否使用该应用程序。这样，Android 操作系统的安全增加了不少。

Android 操作系统中，在应用开发的时候可以申请的权限有很多，包括但不限于获取网络状态、获取电池状态、使用照相机、拨打电话等。

2.2 移动端的感知体系

随着科技的发展，现在的手机越来越多的集多种功能为一身，不再是简单的通讯设备了。

越来越多的智能设备为了满足人们的需要，开发了很多硬件传感器，包括加速计、陀螺仪和气压表，来作为人的感官的补充。检测物理和环境属性的传感器为增强移动应用的用户体验提供了令人激动的创新机会。现代设备中包含了越来越多的传感器硬件，它们为用户交互和应用程序开发提供了新的可能性，比如，增强了现实性、基于语音识别的输入和环境定制方案等。

本文主要介绍 Android 相关的传感器，一般的智能手机设备都会包含以下几个传感器类型[8]。

- 温度传感器。通过该传感器可以获取到以摄氏度表示的温度。表示的是智能

手机所处的环境的温度。

- 加速度传感器。该传感器是一个三轴的，可以返回三个坐标轴的当前加速度。
- 陀螺仪传感器。该传感器可以得到三个坐标轴上的设备的角加速度。
- 方向传感器。顾名思义，表示为三个轴的角度的组合。
- 光线感应传感器。可以返回一个以勒克斯为单位的值，用来描述周围环境光的亮度。

还有一些其他的传感器，比如在原始的移动设备上就具有的声音传感器，非常常见的定位系统等。在这里就不一一介绍了。传感器不仅可以用来作为应用程序的调优辅助设备，也可以用来直接进行采集，作为一个环境侦测设备来使用。

2.3 移动端的网络通信方式

手机和 PC 端在网络通信上有所不同。在手机上进行网络连接有多种方式。可以通过运营商的流量，目前常用的有 3G、4G 网络，也可以像 PC 那样通过 WIFI 等连接网络运营商，然后进一步连接网络。也就是说，在不同的环境下，可能会出现不同方式的连接，连接方式不固定且有时候不稳定。所以，本课题有必要使用一种比较通用的网络方式来进行客户端与服务器之间的网络通信[3]。

在本课题的应用程序中，我们在客户端与服务器之间采取的数据通信方式为通过 HTTP 协议。原因有以下几点：

- 广泛性。当前，几乎所有的智能手机，包括 web 端和客户端在内，都支持 HTTP 协议。而且在工业界，HTTP 协议也是主流之一。很少有其他协议可以做到这一点。
- HTTP 协议是应用层协议，它是一种稳定的，提供面向连接的传输协议。
- HTTP 协议是双向传输协议，支持客户端/服务器模式。
- 简单快速：客户向服务端请求服务时，只需要传送请求的方法和路径即可。请求方法常用的有 GET、HEAD、POST。这些方法规定了客户与服务器之间联系的类型，通信速度很快。

这样，在本课题的应用程序中，我们经常运用 HTTP 的 GET 和 POST 请求，来上传或下载自己需要的信息。

第三章　Android 平台参与式感知应用的需求分析

通过对用户的需求分析，我们将此感知应用的需求分为功能性需求和非功能性需求两大类。在本次章节中，首先对功能性需求进行分析，对各个功能进行细化描述，提出整个应用的明确需求。其次，对非功能性需求进行明确，以此来提高系统的通用性，稳定性和鲁棒性。

3.1 功能性需求

在开发之前，先介绍一下本课题研究内容的使用场景，如图 3-1 所示。

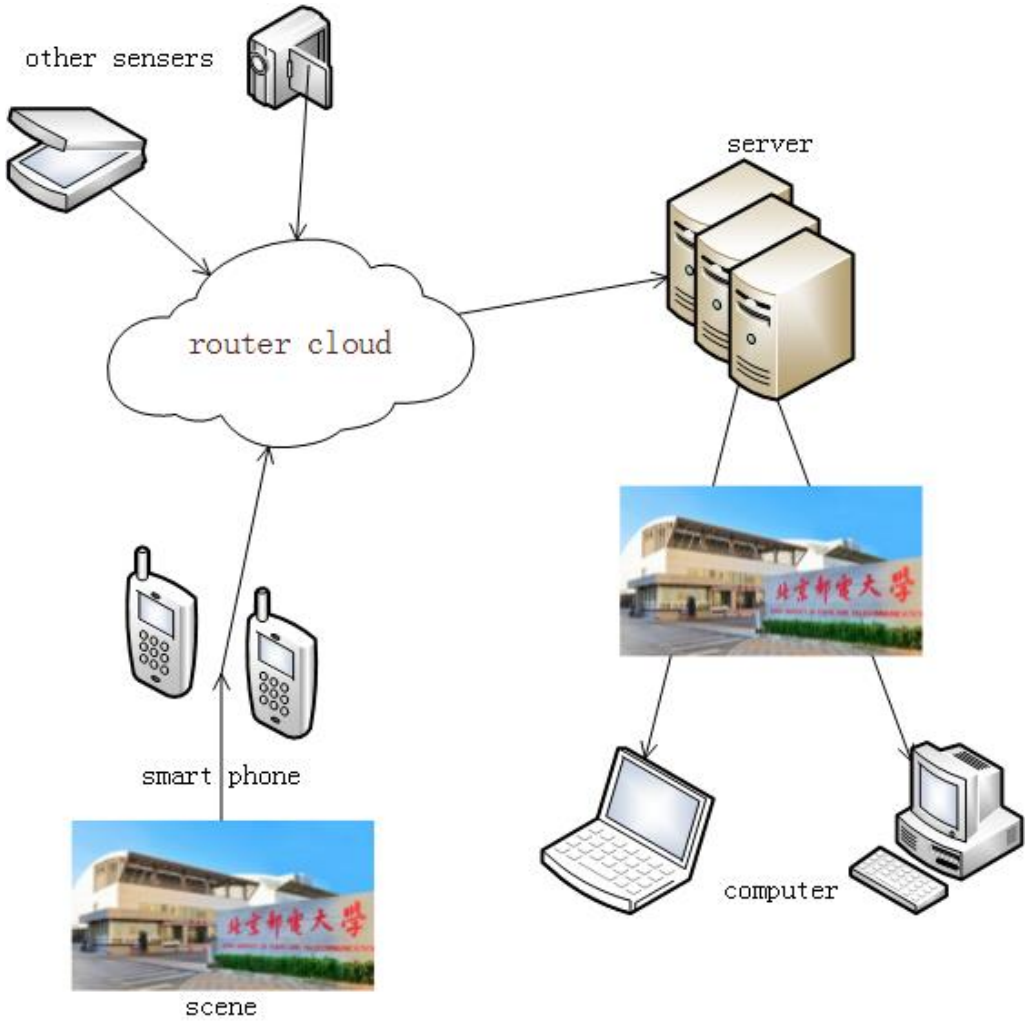

图 3-1 参与式感知使用场景

从图中我们可以看到，参与式感知主要是通过终端采集数据，然后上传到服务器中，进而对数据分析整理，使得其他终端可以方便查看或分析。

Android 平台参与式感知应用的功能性需求主要包括以下方面：

- 实现对应用的一个框架搭建，方便对后来的功能进行扩展。
- 实现对周围环境的感知功能，包括声音，光照，坐标，方向等数据的收集和整理。
- 实现对照片的采集和处理，并且能够通过照片获得相应的周边环境信息。
- 实现后台服务，可以在退出应用的时候依然可以进行数据的采集。
- PM2.5 预测功能。能够通过特定的一些照片信息，和后台沟通，得到当前所照照片的 PM2.5 值。用来对参与式感知应用进行印证说明。
- 实现对采集到的信息进行管理，如添加删除等，方便进行大量的分析。
- 实现对采集到的信息进行展示，实现数据可视化，方便用户自己分析周围的环境。
- 实现对用户的唯一标示，以确保可以根据用户周边环境进行用户个性化分析。
- 实现对一些配置的设置，可以使用户方便的根据个人爱好进行配置。
- 实现国际化的应用软件，比如可以根据系统的语言变换应用的相应语言等方式来进行国际化的处理，方便全球通用。

以上这些是本次版本实现的相关功能。大概可以概括为框架搭建、信息采集、照片收集、后台服务、数据存储、数据可视化、用户系统、可配置和国际化。在这里，主要介绍作者参与的模块：框架搭建、数据存储、数据可视化、可配置和国际化。

3.1.1 框架搭建需求

对于应用来说，框架是整个或部分系统的可重用设计，表现为一组抽象构件及构件实例间交互的方法。可以说，一个框架是一个可复用的设计构件，它规定了应用的体系结构，阐明了整个设计、协作构件之间的依赖关系、责任分配和控制流程，表现为一组抽象类以及其实例之间协作的方法，它为构件复用提供了上下文关系。因此构件库的大规模重用也需要框架。

对于参与式感知的移动互联网应用来多，可能需要感知的方面很多，这就需要一个良好的框架来进行终端软件的整理，找出各个感知系统的共同点和不同点，地宫一个可以复用的设计组件，通过插件等方式进行加载和导入，方便多人开发。

作为一个刚刚起步的应用，框架搭建是根本。有一个良好的框架，可以对以后新加的功能做一个方便扩展。整体框架图如图 3-2 所示。

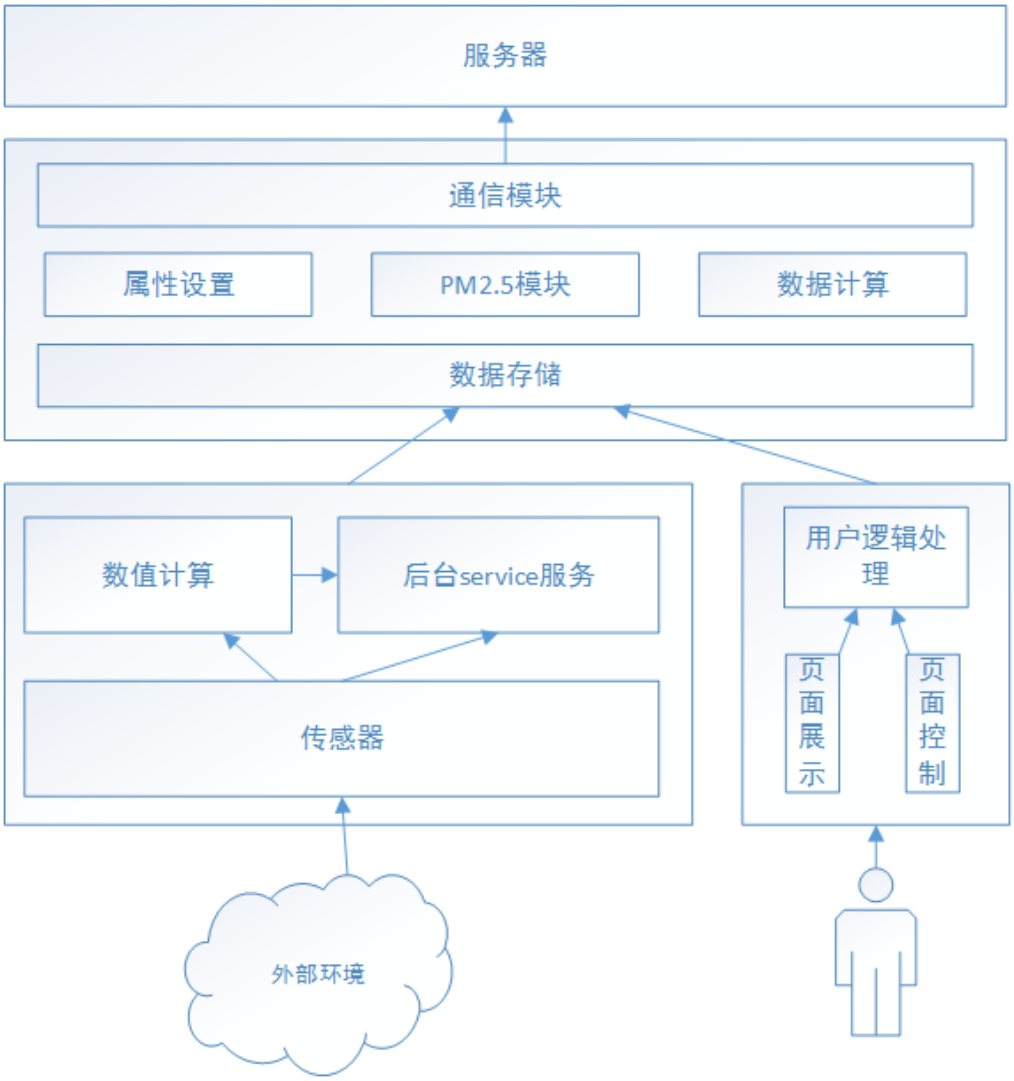

图 3-2 参与式感知应用整体框架

3.1.2 数据管理需求

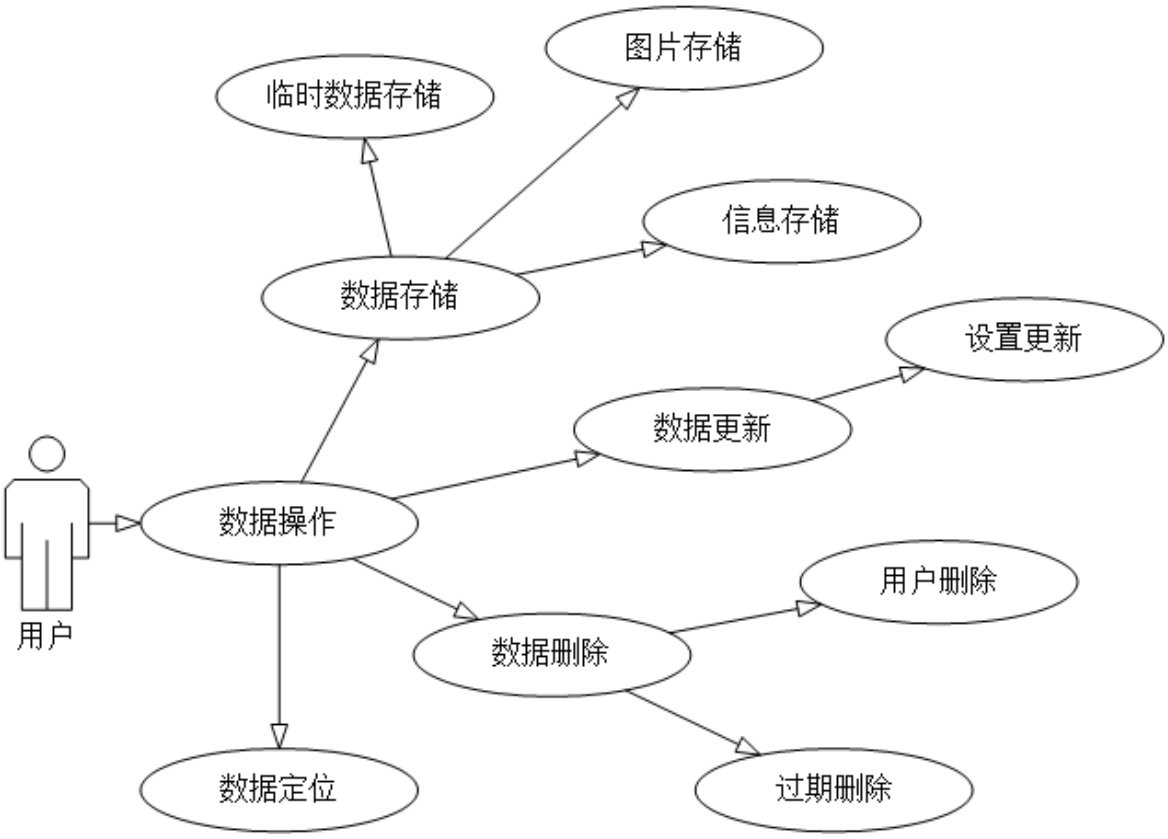

图 3-3 数据管理用例图

数据管理是参与式感知应用的核心需求。用户需要对周围环境进行相应的感知，就需要进行相应的数据操作。当他查看当前周围光照强度的时候，需要看到数据；当他需要拍照的时候，需要数据；当他想要查看以往的周边环境的时候，也需要数据。所以，数据管理是非常重要的。主要包含以下几个方面：

- 临时数据存储：很多情况下只需要知道最终的结果，中间的处理都是临时数据。这些数据若较小的话，存储在内存中，若较大的话（如照片），则存储在临时目录下，在程序退出的时候删除。
- 图片存储：用户需要拍摄照片获取周围的环境。这种情况下，照片是需要存储的。否则，退出程序后不能再次观看，那可就不好了。
- 信息存储：除了照片外，还有一些其他的环境参数需要进行存储。如声音，光照强度等。这些没有图片资源，只是一些数值数据，因此和图片不同，需要另外一种存储方式。
- 设置更新：有一些可以设置的配置项，可以让用户进行设置。这些设置信息是需要存储的，而且用户是可以进行更新数据，来达到自己的个性配置。
- 用户删除：当有些数据用户觉得不需要的时候，可以进行删除。这样的话，

有一个良好的删除方式，可以让用户很容易的进行操作。

- 自动删除：当数据量存储过大的时候，我们需要能够自动清理很早之前的数据，这样就能够使得用户的手机不会因为本课题应用导致存储占用过大。
- 数据定位：该应用会把数据上传到后台服务器。但是我们在本地存储的数据是不能被删除的，且不能被重复上传的。这就需要有个数据定位功能，能够准确的定位初未上传的数据所在位置。

对于这些用户的数据需求，我们应该采用非常便捷快速的方式使得用户获得所需。

3.1.3　数据可视化需求

数据可视化就是常说的数据展现。主要就是把需要告诉用户的感知到的信息以某种方式展现出来，直观的呈献给用户，使得用户能够快速方便的感知到自己所处的周围环境。由于数据分为图片存储，信息存储。故在这个课题当中，数据可视化也分为两个部分：

- 图片视图：图片的可视化，可以根据用户自身的需要展现对应的图片。而且，在用户拍完照等图片操作之后，能够有一个良好的通知系统使用户感知到图片已经采集完成并可以以某种方式展现出来。不仅如此，当用户想看到服务器上其他的人拍的照片的时候，需要能够得到所需要的图片。但是由于服务器上照片过多，还需要进行适当的处理来达到查看照片的目的。
- 信息展示：对于非文件的其他数据信息，如光照、声音等，同样需要给用户一定的感知方式。可以以图表或者文字说明的形式来达到展现数据的目的。具体展现什么样的内容，有两个需求。第一，能够非常及时准确的展示用户当前所处的环境，如声音、光照等，以数值的形式呈现。第二，能够展示用户近期所处的环境，给用户一个对比方式，让用户能够清晰的感觉到自身最近所处的周边环境的变化，达到对比的目的，以图表的形式展现。

对于数据展现，是本次课题研究的主要内容之一，也是 UI 界面部分需要关注的重点之一。

3.1.4 可配置需求

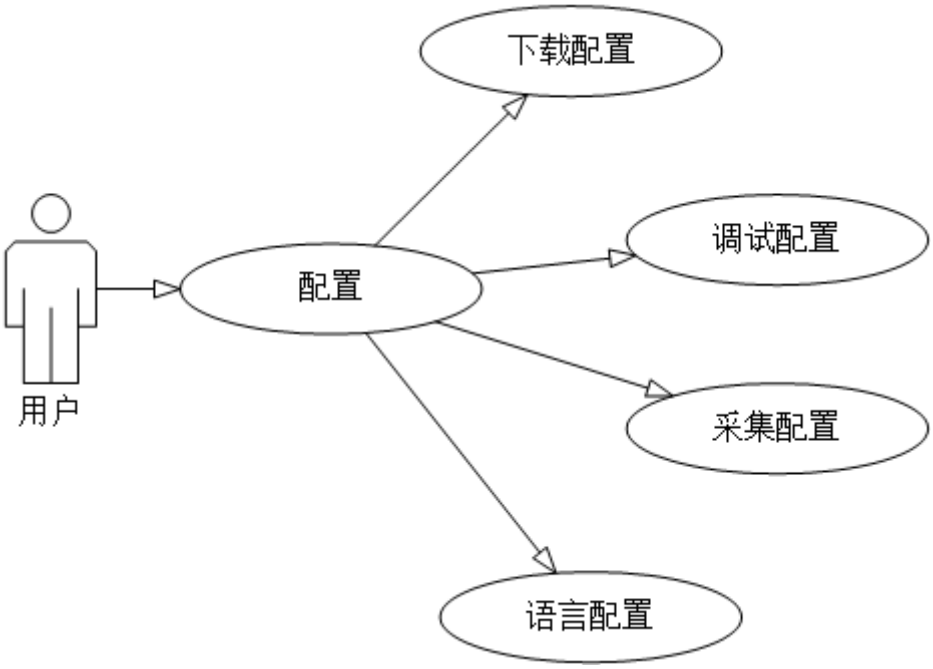

图 3-4 可配置用例图

对于一个手机应用程序，应该对用户开放各种配置信息，而不是强迫用户去做一些事情，因此，虽然对于配置可有可无，但是配置可以使得自身的应用更加的健壮，能够使用户或者开发者灵活的通过设置一些配置信息来达到自己所期望的目的。

本课题应用的具体配置需求，需要根据具体的实现来决定。目前有以下几点：

- 下载配置：对于上文说的从服务器上得到照片，但是不能得到太多，需要适量。此配置就是表示需要获得距离目前多长时间内的照片。
- 调试配置：对于开发者来说，当上传照片或者数据的时候，大部分是在开发的过程中为了调试而上传的一些无用的照片数据等。为了便于区分真正使用和调试，暂时在此课题的第一个版本中使用这个配置，方便测试开发。
- 采集配置：对于采集信息，有时候是需要后台进行定时采集的。这种情况下，需要的是经过用户同意来进行定时采集。于是，用户应该可以配置定时采集的开关已经定时采集的频率等。
- 语言配置：虽然有国际化（见下一个需求）来自动配置语言。但是，也可以允许用户自己设置自身需要的语言。目前只支持英语和中文。

3.1.5 国际化需求

对于一个智能手机应用程序，而且作为一个课题研究性质的应用程序，需要使得大部分人能够参考和使用，这就要求不仅需要在中国可以使用中文的人可以使用，

也最少可以使得非中文的国家大部分能够使用。因此，需要使得程序国际化。

国际化需求就是，需要能够使得本课题的应用程序能够自己适应不同地区的市场，提供一个自适应的、更加友好的界面，而并不需要改变程序内部的逻辑功能。本课题只是中文和英文的支持，当手机的操作系统的环境是中文的时候，便显示中文的字体，如果手机操作系统的环境是英文的，那么，则显示英语字体，而不需要重新写代码逻辑。

不仅语言，对不同的手机屏幕，虽然不能保证布局完全相同，但相对应的布局还是需要且应该保持一致的。本课题另一个国际化的需求就是针对页面布局，能够保证大部分 Android4.0 以上版本的手机操作系统在不同的手机屏幕分辨率下，能够保持正常的显示，而不会因为屏幕大小不同，显示不同的、样式变形的布局。

3.2 非功能性需求

除了需要满足功能性需求外，系统的非功能性需求也非常重要，功能性需求影响着一个应用程序的功能相关，是否能达到目的，而非功能性需求直接影响着用户的使用体验。本课题的应用程序的非功能性需求主要有以下几个方面：

界面简洁，方便易用。对于一个 Android 手机的应用程序，需要界面简洁大方，使用户能够在不阅读相关文献的基础上快速使用。不仅要符合 Android UI 的响应风格，而且要有对应良好的图标和适当的文字提示，使得用户极其容易上手。

- 兼容性需求。对于快速发展的互联网时代来说，兼容性已经逐渐成为了一个应用不可或缺的非功能性约束。对于本课题的应用程序来说，虽然开发在 Android4.0 版本之上，不能兼容以前的 Android 版本，但是，要求能够兼容后续的版本已经不同的 Android 操作系统，如 MIUI 等其他的基于 Android 的操作系统。在系统设计的时候，应该考虑能够向下兼容，支持以后的 Android 版本，而不需要频繁的修改程序代码和框架。
- 性能需求。首先，不能出现严重的卡顿，严重影响用户的使用体验。在响应用户请求的时候，需要能够快速准确的处理用户的请求。需要等待的地方，要有相关的提示说明。其次，对于内存使用方面，不能有内存泄露等重大错误发生，而且内存占用应该尽可能的小，不会对其他的后台程序造成显著影响。
- 稳定性需求。此课题的应用程序应该能够保证系统的功能的稳定性，能够有效地实现功能性需求，而不会出现崩溃等现象的发生。若真的遇到错误的话，要有相应的日志处理操作，而且要有良好的提示信息提示系统出现问题，使用户能够方便接受。而且通过日志信息，能够快速的定位发生的问题所在。

- 可扩展性需求。本课题的应用程序还处在第一个版本的状态，功能不健全，稳定性可能达不到完美，而且将来肯定有需要扩展的功能、用途等。对于这样的情况，应该使得应用程序能够方便的扩展，能够在添加新功能的时候不需要大幅度的改变代码结构，方便插入新功能。
- 本系统不应与其他 Android 手机软件发生冲突，从而不影响用户的使用。

第四章　Android 平台参与式感知应用的概要设计

Android 平台下的参与式感知应用集成了很多的功能，如采集数据，数据管理，数据可视化等诸多功能。我们在概要设计的时候将他们分成各个模块，明确分工，这样更加利于开发。下面我们首先先从整体设计上看一下感知应用的模块构造，然后对作者参与的模块进行进一步的概要设计说明。

4.1 Android 平台参与式感知应用的总体设计

根据参与式感知应用的需求和实现目标，整个系统的总体模块的划分如图 4-1 所示。

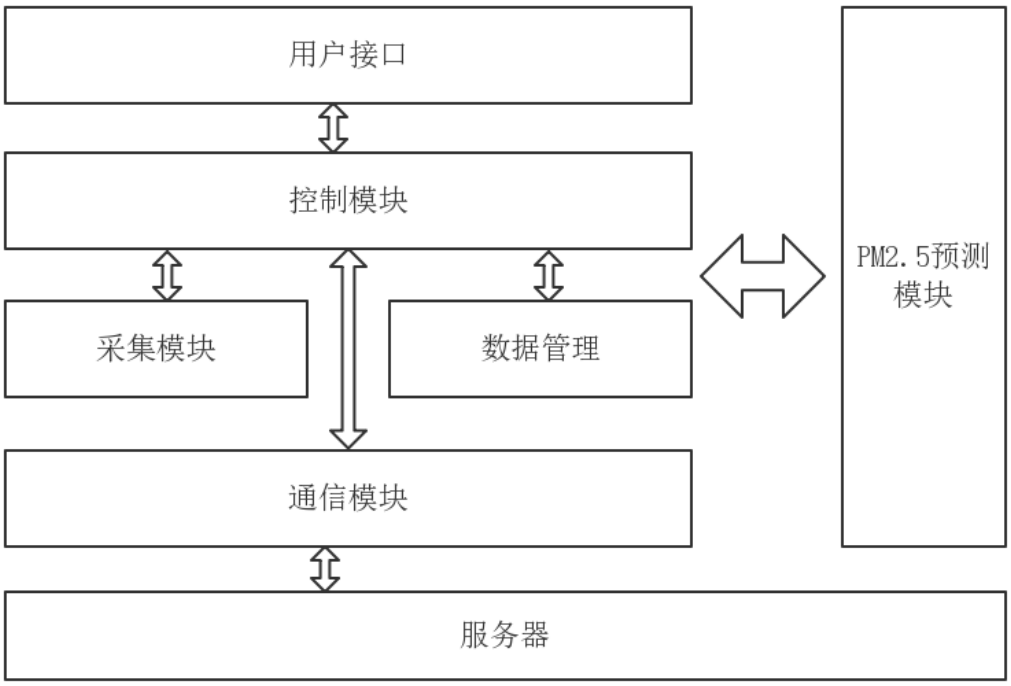

图 4-1 参与式感知应用的整体模块划分

我们在大的方面，将 Android 参与式感知应用分为 6 个模块。从上到下具体内容如下：

- 用户接口模块。这个是本课题应用的直接面向用户的一个模块。该模块主要负责用户与应用的交互。用户通过该接口可以进行应用功能设置、查看相关信息等操作。该模块同时负责通过控制模块来调用其他模块。

- 控制模块。控制模块作为一个统筹全局的模块，在整个 MVC 框架中属于关键的 Control 层。目的是为了协调其他管理层之间的相互关系，运用各种手法，尽可能的降低各模块之间的耦合，达到模块直接独立开发并能协调运作的模块。使他们相互合作完成参与式感知应用的各个功能。
- 采集模块。作为一个参与式感知的应用，采集数据是必不可少的，因此，采集模块应运而出。主要负责声音、光照等数据信息的采集，以及照片、音频等文件信息的采集。
- 数据管理模块。数据管理模块不会直接向用户开放。这个模块主要是用来对用户设置信息的存储管理，对一些采集到的感知环境内容进行存储管理，对一些标记信息进行存储管理，以及对以上信息的读取修改等管理。此模块是参与是感知应用系统的重要数据支撑模块。
- 通信模块。该模块主要的功能是与服务器进行连接，数据交换的作用。主要是用来通过 GPRS 或者 WIFI 网络经过 HTTP 等网络协议进行采集数据和用户信息的上传或者下载，以及一些其他相关信息的上传下载。该模块是整个参与式感知应用与后台的接口。
- PM2.5 预测模块。该模块作为一个独立于整个参与式感知平台的模块，功能非常明确。该模块是通过特定的几张照片进行图像识别与分析，并和后台通信，得到对应的模型数据，然后通过这些新的模型数据，对新的相似照片进行一个 PM2.5 值的预测。此模块是用来进行参与式感知数据的验证和实验，获取感知器收集到的数据并对用户进行展示，是对参与式感知方式的一种说明。
- 服务器模块。服务器是用来进行大量数据存储和分析的模块。通过网络通信等方式与客户端进行沟通，得到相应的数据并进行分析，把需要的结果展示在客户端或 web 界面。是一个不可或缺的部分，是客户端应用之间相互沟通的桥梁。

由于本课题应用实现的是第一个版本，涉及到的模块多且大，每个模块下面都有很多需要实现的功能和细分模块，因此在做这个应用实现的时候，我们采用小组开发，并进行了明确的分工。作者主要负责客户端部分的用户接口、控制模块和数据管理模块。由于数据管理模块属于支撑模块，在本章不予介绍。在本章中，主要对控制模块和用户接口模块进行详细的分析。

4.2 控制模块设计

控制模块不仅有设置模块，出错管理模块，还需要做到统筹全局，做到各个模

块之间的沟通能够独立协调进行，是模块之间的桥梁，属于流程管理模块。而且，还需要对一些操作进行日志处理，这样就可以在测试中查看日志，方便查看程序运行状况，也就是日志模块。下面详细介绍各个细节模块之间的设计。

4.2.1　设置模块设计

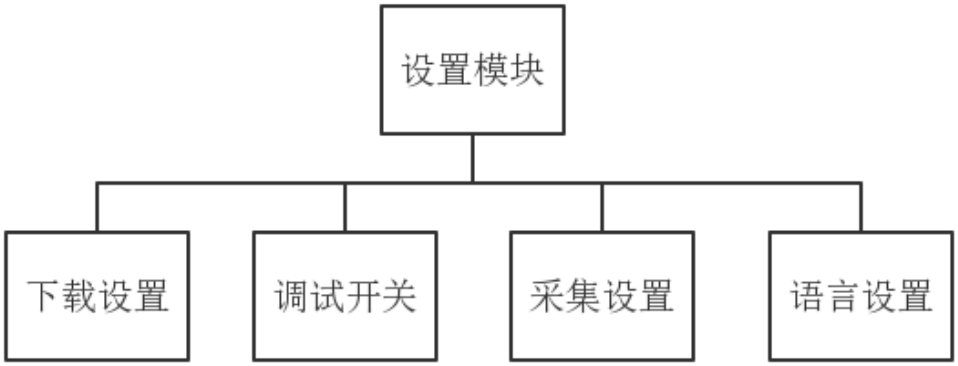

图 4-2 设置模块内部架构图

当用户进入设置模块的时候，可以选择多个不同的设置项进行设置。这些设置有些是针对作者所参与开发的几个模块，有些是针对他人开发的模块，为统筹全局的设置，隶属于控制模块。图 4-2 显示了目前可以设置的配置信息。分为以下四种：

4. 2. 1. 1 下载设置

当用户从服务器获得多个周围环境的图片的时候，需要向服务器发送请求。但由于图片过多，不可能把服务器上的图片全部下载下来。否则，不只是手机存储空间不够，网络带宽也是让人不能忍受的。因此，增加了此设置信息，可以设置距离当前时间一段时间内的照片来获取。

该设置是一个选项，可以从 3 个小时到 1 周之内的大致时间段内都可以选取。当用户点击该设置后，会可以选择 3 小时、6 小时、24 小时、3 天和一周这几个选项。这些时间就表示了在需要获得服务器上数据的时候，会获得多久之内的数据。默认为 3 个小时。

在这个模块中，我们主要的操作是了解当前数据的大概发布日期以及设置获取的数据时间。

4.2.1.2 调试开关

在程序员进行编程实验的时候，为了测试某些采集的功能，会进行相应的采集，而这些采集数据在测试上传功能的时候会一并上传到服务器上，对服务器会造成影响。

调试开关是一个 2 位选项，用户可以开启或关闭调试状态。

当用户开启调试状态的时候，所有的数据收集都会在数据中的一个调试选项中设置为 TRUE。而用户关闭调试状态后，调试选项会被设置为 FALSE。默认为 FALSE。

在这个模块中，我们主要的操作就是查看当时的模式是否为调试，以及对调试模式的开关控制。

4.2.1.3 采集设置

既然是参与式感知，就会有采集模块在不同的工作。而在控制模块下的采集设置是对采集频率的一种控制。目前主要是针对后台采集（被动采集）来进行设置。

该模块有两个功能可以进行设置。

- 后台采集开关。它是一个 2 选项开关，用户可以开启或者关闭后台采集功能。当用户开启后台采集功能的时候，本课题应用就会在退出的时候开启一个服务进行后台采集功能的运作。选择关闭选项的话，则关闭后台采集功能。默认开启。
- 采集频率设置。这个是一个多项选择，只有在后台采集开启后才会生效。用户可以选择 1 小时、2 小时、3 小时、6 小时和 12 小时这五个选项来确定后台采集的频率。当用户选中这五个选项之一且开启了后台采集开关，在程序退出后，后台采集会以此频率为基准。默认为 1 个小时。

在这个模块中，用户主要的操作就是查看当前后台采集的状态及后台采集的频率，以及对后台采集参数的一些修改。

4.2.1.4 语言设置

由于需要国际化，本课题应用需要支持多种语言之间的切换。目前支持两种语言：英语，中文。在这个设置选项中，有三个选项。

用户可以选择“默认”，“英语”和“中文”三个选项中的一个。其中，

- “英语”选项指的是设置应用程序语言为英文。
- “中文”选项指的是设置应用程序语言为中文。
- “默认”选项指的是设置应用程序语言为当前程序所在系统的语言。

用户选择这三个程序中的以一个，本课题应用程序语言会进行相应的变更。

在这个模块中，用户主要的操作就是查看当前的语言设置以及根据自己的喜好进行相关的语言变换。

4.2.2　出错管理模块设计

出错管理，不属于任何功能性需求的一个模块。但是在开发使用中，又占据着非常重要的一个部分。这个模块，不与任何模块有关系，而是与这个应用程序有关系。我们在编写程序时，不能预测到所有的用户操作和所有的隐蔽错误所在。为了防止在有些非预期的环境下出现崩溃而不能找到错误原因，此模块就是起到出现崩溃能友好的提示用户并进行相应记录方便开发者进行错误定位的功能。流程图如图 4-3 所示。

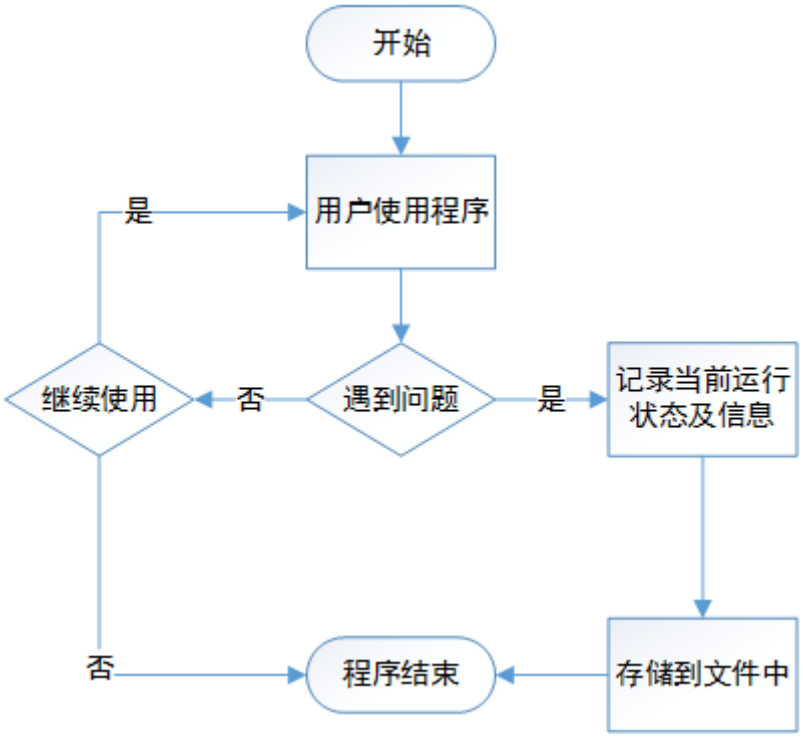

图 4-3 错误管理模块流程图

用户无法直接使用此模块。当用户使用本课题程序崩溃的时候，出错管理模块会起作用，首先会提醒用户当前程序遇到问题，需要关闭。其次，对程序的当前状态进行记录，包括运行的 Android 设备的型号，Android 版本，以及当前程序崩溃时的堆栈情况。然后把这些记录存储到手机存储卡内，以此来方便开发人员快速定位出错原因。

4.2.3　流程管理模块设计

为了尽可能的实现各个模块之间的解耦合，作者加入了控制模块。控制模块除

了上述的几个模块之外，最主要的模块就属于流程管理模块。该模块是用来把各个模块之间联系起来，使得他们能够方便的进行沟通。而且只需要提供相应的接口，无需知道对方如何实现的。这样就可以尽可能的解耦合。

流程管理模块主要针对的是数据管理模块与其他模块之间的流程、用户接口模块和其他模块之间的流程，采集模块和其他模块之间的流程，PM2.5 分析模块与其他模块之间的流程这几个方面来进行管理和统一的。做到把数据和界面相分离，是 MVC 模式中的 Controller 层。具体的数据流图如图 4-4 所示。

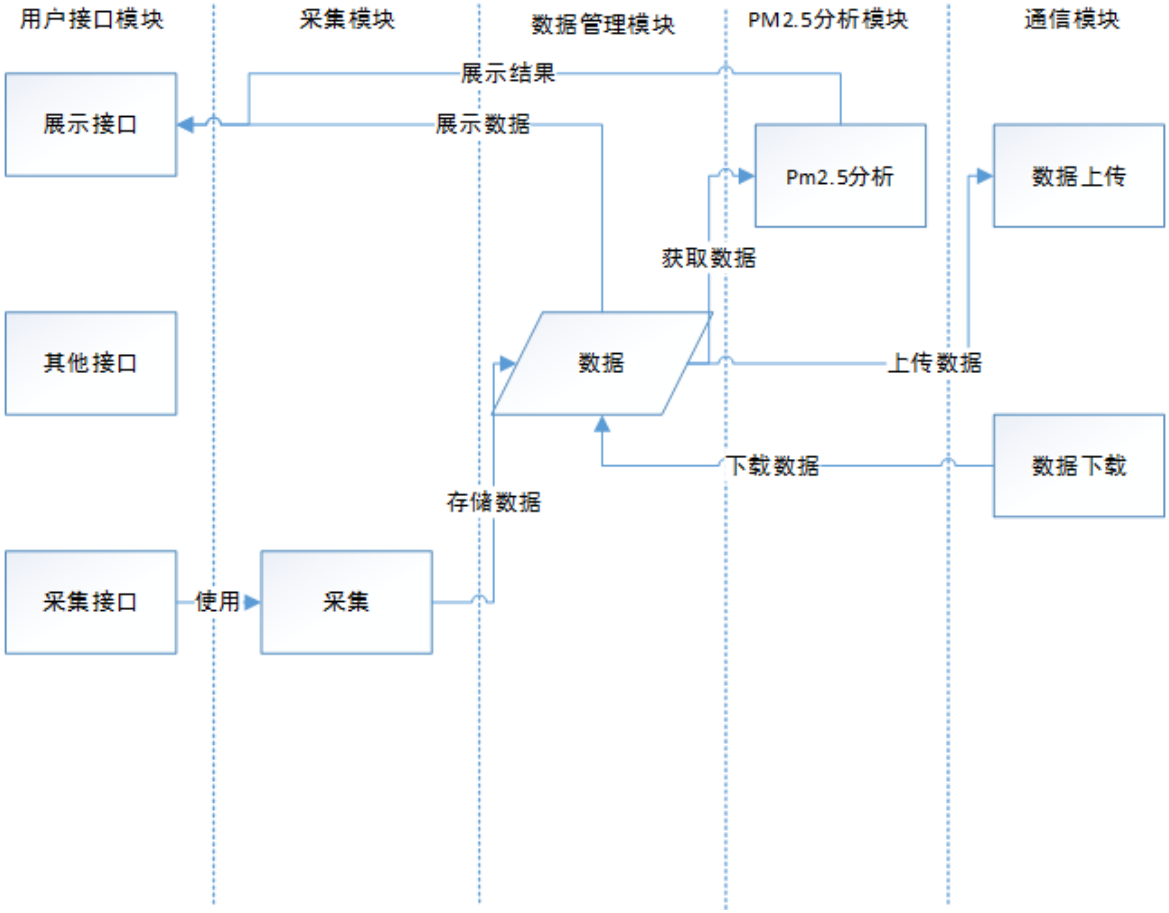

图 4-4 客户端流程管理中的数据流图

图 4-4 是一个简化的数据流图，可以看出，在这个流程管理的过程中，数据管理模块是其他模块的支撑，所有模块都会和数据管理模块相交互。而此流程管理模块，重点是保证数据管理模块与其他模块之间的接口，然后对其他模块也进行相应的接口定义，使得除了数据管理模块之外，其他模块也能够很好的低耦合的相互运作。

流程管理模块主要是用来管理一些数据流和一些操作流的相关步骤，达到各个模块之间的低耦合，从而控制各个模块之间的交互流程。该模块贯穿各个模块，而不是单独存在的。

4.2.4　日志模块设计

日志模块和出错管理模块相似，只是日志模块是贯穿整个应用运行过程当中的，而出错管理模块是在程序遇到未知的错误而导致崩溃的时候才会启动模块功能。可以认为，出错管理模块是日志模块的一个疏漏准备，是日志模块的一个“查漏补缺”。

日志模块是对软件主动或非主动的所有可能操作进行一个记录，包括但不限于以下几个方面：

- 对用户操作的一个记录。包括用户点击某些重要的功能按钮，用户主动进入某个模块。都是用户主动操作的记录。
- 对模块接口调用的记录。包括某些模块从数据管理模块中的数据库取数据，或者向数据库存入数据。这些是针对各个模块的对外接口所记录的信息。
- 系统运行异常的记录。包括后台服务的可捕捉的运行异常，数据库的异常操作，程序运行过程中可以获取到的异常操作，以及其他资源的异常使用等。这些是针对可以预见到的异常进行的日志。

日志模块包括但不限于以上几个方面，还有其他的需要日志模块的地方就不再一一叙述了。日志模块属于控制模块，向其他模块提供接口，而且会分为一定的级别方便用户或开发者进行查看。级别包括“警告”，“错误”，“正常”三个级别。用户可以通过级别来观看自己想关注的日志。

4.3 用户接口模块设计

用户接口（User Interface）模块，就是平常所说的 UI 模块。主要是用来给用户提供一些接口，方便用户操作，能够使用户在不需要看任何说明文件的情况下，就能够快速上手使用。本课题应用中，对于用户接口模块，我们按照详细功能可以分为主动采集接口、信息展示接口、信息上传接口等。以下进行说明。

4.3.1　主动采集接口设计

主动采集按照采集的类型主要分为两类，一类是照片，一类是数据。这两个方式有一些不同，若为照片，存储的是文件和相关信息，若为数据，则只存储数据。

4.3.1.1 照片主动采集

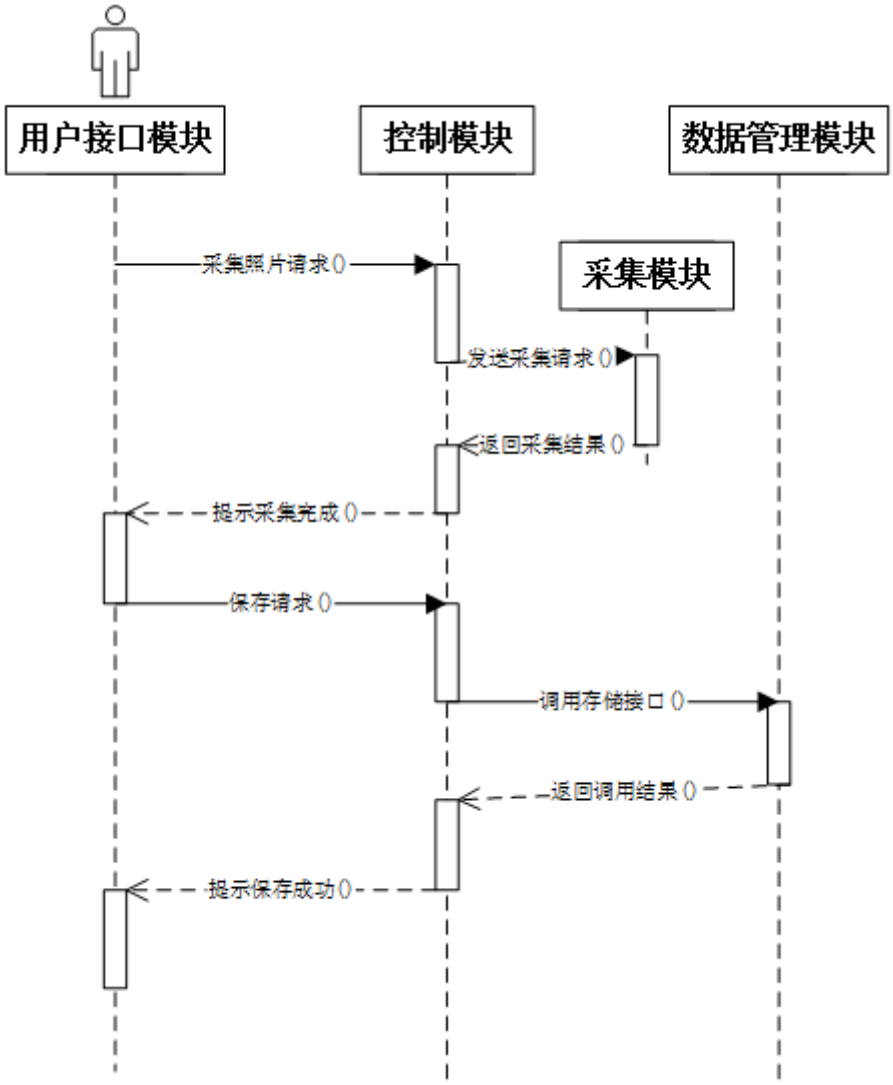

图 4-5 照片主动采集时序图

通过时序图 4-5，我们可以了解到，一次用户通过用户接口主动采集照片的过程如下：

1) 用户从界面发起采集照片请求，即点击“拍照”按钮。
2) 控制模块调用采集模块接口进行采集，将采集完成的消息返回给用户。即弹窗提醒用户照片完毕，是否保存。
3) 用户从界面发起保存照片请求，即点击“保存”按钮。
4) 控制模块调用数据管理模块的接口对数据进行保存。
5) 调用完成后，提示用户保存成功。

4.3.1.2 数据主动采集

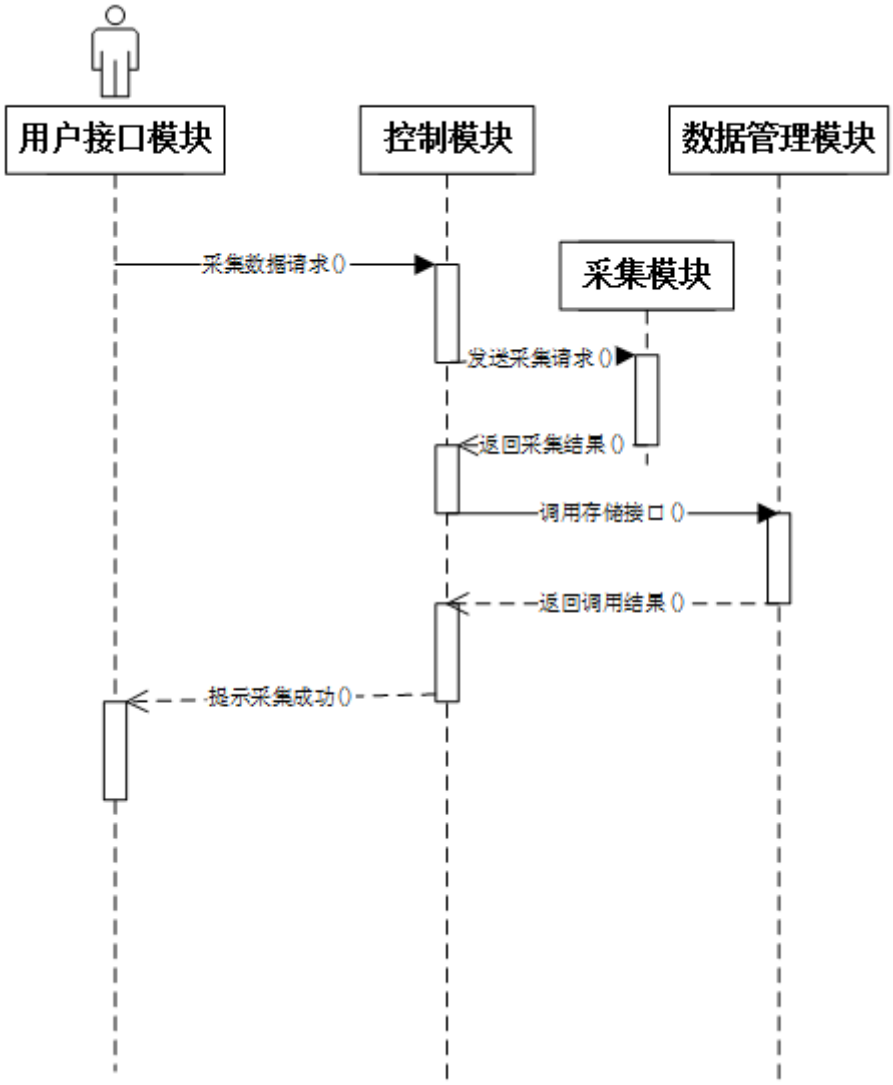

图 4-6 数据主动采集时序图

通过时序图 4-6，我们不难发现，数据的主动采集与照片略有不同。流程如下：

1) 用户发起数据采集请求。即在数据展示页面点击“刷新”按钮。

2) 控制模块调用采集模块进行数据采集。

3) 采集数据完成后，控制模块继续调用数据管理模块对数据进行存储。

4) 存储完成后，提示用户采集成功。即在界面上显示采集到的结果。

数据采集流程与照片不同的地方在于，照片是个文件，占用空间相比较较大，而且对于照片来说，大部分是由于用户的不同操作（如拍照方向等）导致不同的采集结果，因此要用户来决定是否采集成功，成功则保存，否则取消本次采集结果。

对于数据采集，可以完全交给程序进行自动化的采集，数据准确，与用户的行为无关，可以客观反映周围的环境情况，因此不需要用户在保存数据之前验证是否可用。直接保存即可。

4.3.2 信息展示接口设计

针对不同的信息格式，与主动采集接口类似，信息展示接口也会采取不同的方式来展现不同格式的信息。在这里分为数据信息展示、PM2.5 展示、周边环境展示三个方面来进行阐述。

4. 3. 2. 1 数据信息展示

对于单纯的数据信息来说，简单的展示方案就是直接显示数值，也就是如图 4-6 所示，在获取到采集成功的消息之后，直接通过控制模块去数据管理模块读取数据，然后显示在屏幕上即可。

本课题不仅单单显示周围的即时信息，还会显示一周之内用户的周围环境变化趋势。显示的内容包括周边光照强度一周变化趋势以及周边声音强度一周变化趋势。这些变化趋势的显示时序图如图 4-7 所示。

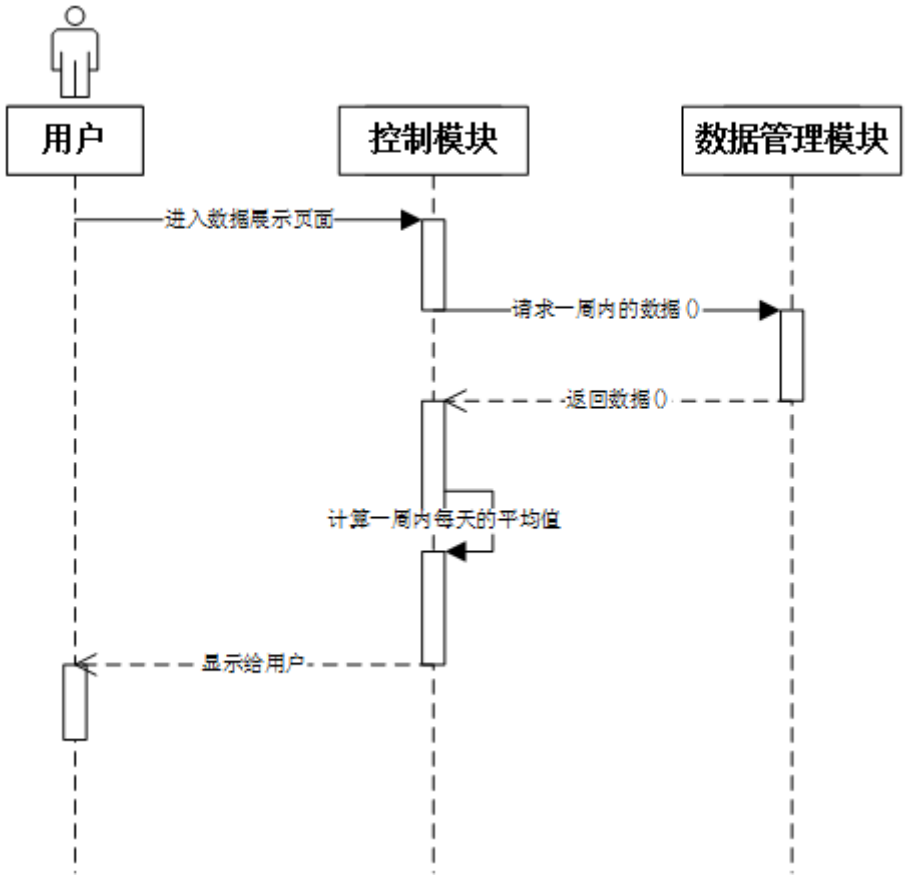

图 4-7 数据信息展示序列图

从序列图可以知道，对于一周之内用户周边环境感知变化的数据展示，大概有以下几个流程：

1) 用户进入到展示的界面。

2) 控制模块向数据管理模块发起请求，请求一周内的所有数据。
3) 控制模块得到数据之后，对每一天的数据进行计算，算出每一天的平均值。
4) 将平均值展现在用户接口上，显示给用户。

4.3.2.2PM2.5 展示

PM2.5 模块作为一个独立开发的模块，作者没有参与其中的开发，但是会调用相关接口进行一系列的操作。本小结就是展示用户 UI 是如何与 PM2.5 模块相沟通的，也是为以后可能开发的其他独立模块做示范。流程图如图 4-8 所示。

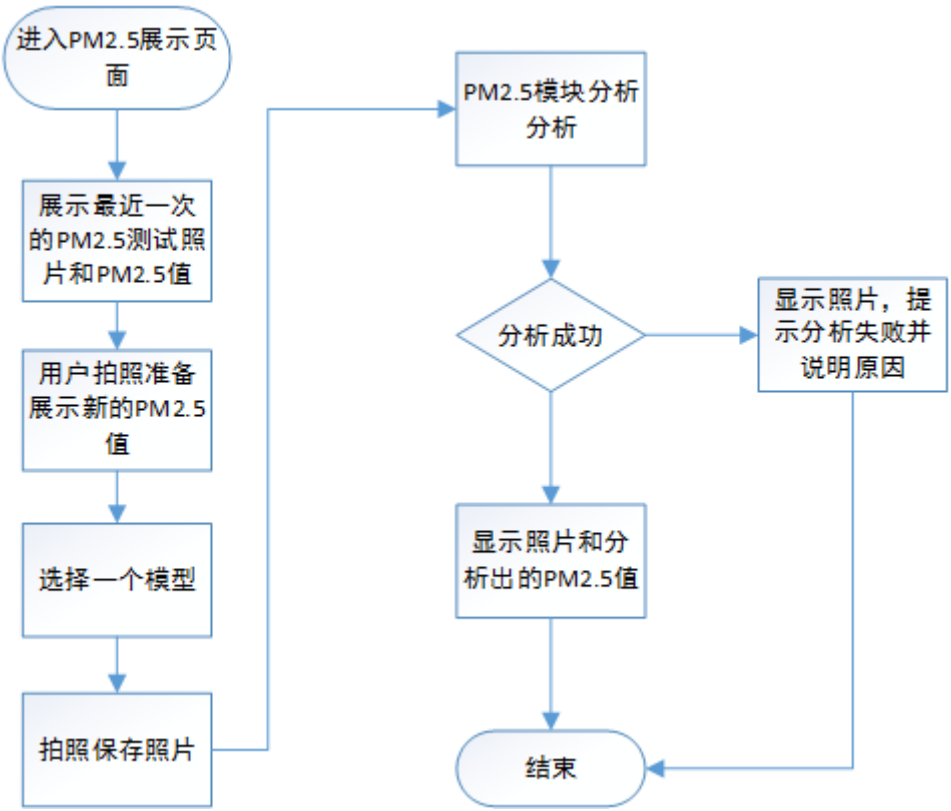

图 4-8 PM2.5 展示过程流程图

根据流程图 4-8 可以知道，PM2.5 分析的过程中，会出现失败的现象，这个原因是因为 PM2.5 模块需要很多的基础条件才能达到分析出结果的目的。如基础照片的张数等。所以用户在开始分析的过程中，并不知道最终的分析结果。在结果出来的时候，用户接口模块会把结果展示给用户，如果分析失败，会弹出提示来说明分析失败的原因，方便用户进行其他的操作。

4.3.2.3周边环境展示

周边环境指的是用户可以查看周围人所处的环境。目前所支持的周边环境都是照片。用户可以通过查看周边环境来观看周围的人照的照片。序列图如图 4-9 所示。

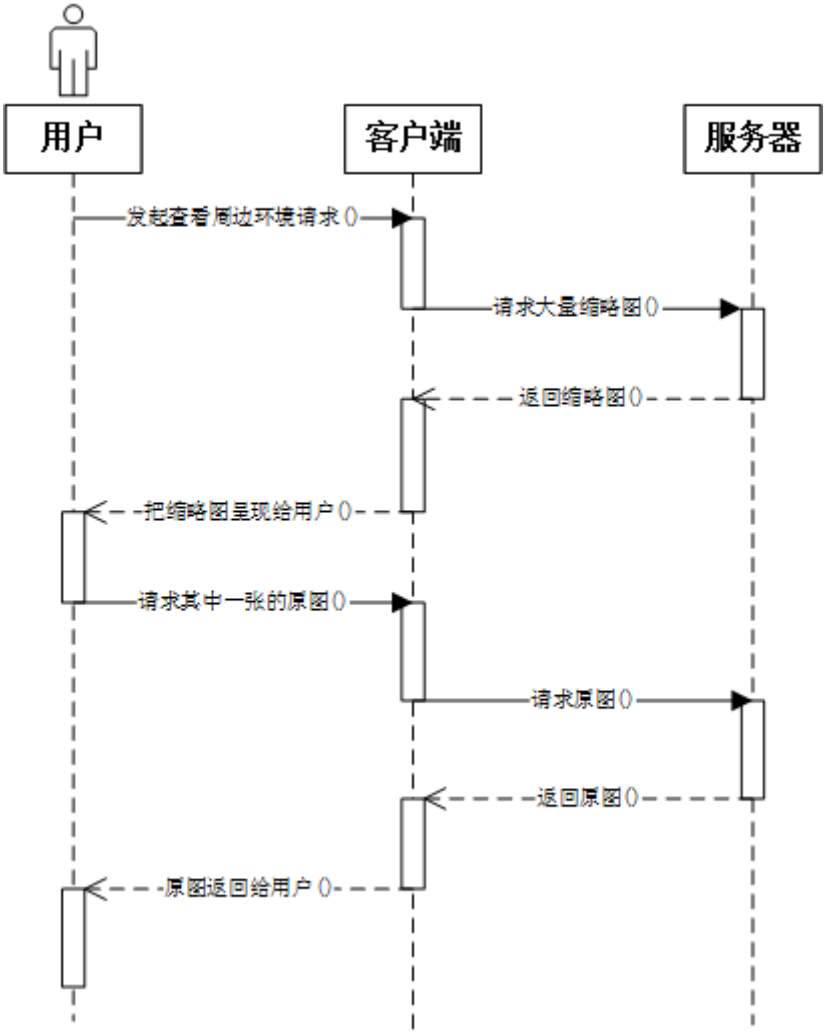

图 4-9 周边环境展示时序图

从时序图 4-9 中可以看到，服务器存储的用户照片在存储原图的基础上，也会进行缩略，存储其缩略图。这样当客户端进行请求的时候，返回的是缩略图大小，不会造成因内存过多而导致的崩溃等错误。这些缩略图是在上传后就完成了。属于服务器模块部分，在这里就不再叙述。从时序图我们可以看到，大致的过程如下：

1) 用户跳转到查看周边环境的界面，即进入“照片墙”界面。
2) 进入页面后，客户端主动向服务器发起请求，请求最近一段时间（详见设置模块）内的照片缩略图。
3) 服务器返回缩略图，并且附带一些信息。如原图地址，缩略图信息等。
4) 客户端以照片墙的形式向用户展示缩略图。
5) 用户可以随意选取一个缩略图请求查看原图。即点击一张缩略图。
6) 客户端根据缩略图内的相关信息去请求原图下载。
7) 服务器返回原图。
8) 客户端将原图展现给用户，方便用户进一步查看。

4.3.3　其他接口设计

用户接口模块不止有以上几个功能，还有一些小的功能，如弹窗提醒，设置信息展示等，由于功能较小，和大部分只是提醒展示用，在其他模块中已经说过，在这里就不再重复了。

第五章　Android 平台参与式感知应用的详细设计

本章根据参与是感知应用的概要设计进行进一步的分析，将主要对数据管理模块，控制模块，用户接口模块进行详细设计的说明，主要结合类图，进一步对每个模块的功能和结构进行分析。

5.1 数据管理模块设计

数据管理模块属于支持模块，主要对数据进行存储管理：采集到的数据进行存储，用户设置进行存储，临时信息进行存储等所有可能产生的需要存储的数据。本模块为其他模块提供了这些信息的访问、修改等接口，包括数据库存储、文件存储、系统函数调用等多种形式。数据管理模块总的调用流程图如图 5-1 所示。

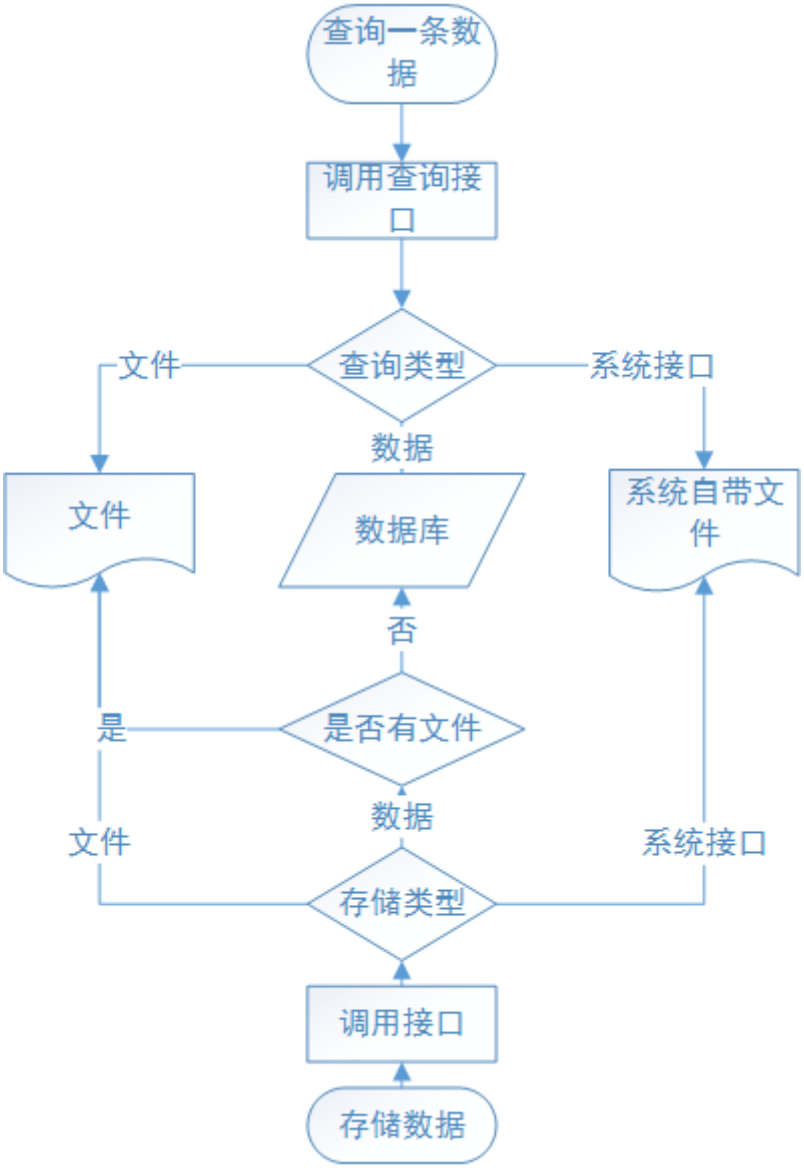

图 5-1 数据管理模块详细流程图

从图中可以看到，对于存储数据，可以单独对文件进行存储，也可以调用系统接口存储到系统自带文件中，也可以向数据库中存储。在向数据库中存储的时候，若存储的是文件，仍然需要再次调用文件接口来进行存储，然后把文件的相关信息存储到数据库中。

下面就针对各个存储方式进行详细的说明。

5.1.1 数据库模块

数据库主要是对采集到的数据进行存储，这些数据都是按照一定格式规定的，而且需要经常性的查找和添加，所以，用数据库存储是非常方便的。用数据库存储有几个优点：一是存储方便，直接调用现成的接口即可。二是读取方便，数据库内部有相应措施能够快速查找。三是结构清晰明了。

本课题应用程序主要使用的是 Android 开发中自带的轻量级数据库 SQLite。使用数据库的模块非常多，在本课题中，主要对采集到的数据分类整理，存储到不同的数据库表中。数据库模块的设计主要包含两大类：文件相关表和数据相关表。

5.1.1.1 文件相关表

表 5-1 文件相关数据库表

内容	类型	说明
索引 id	int	用来作为唯一标识的序列，数据库保证其递增且不重复的
照片文件名	string	用来说明照片文件名是什么，对应于正确的照片
方向（x 方向）	float	照射方向 x 方向
方向（y 方向）	float	照射方向 y 方向
方向（z 方向）	float	照射方向 z 方向
经度	float	位置的经度
纬度	float	位置的纬度
图像 iso（曝光度）	int	图像的曝光度
焦距	float	照相时的焦距
快门时间	DATE	照相时的快门时间
光圈	float	照相时的光圈
Tag	int	用来表示类别（当该值为 0 的时候，表示是正常的照片，1 为视频文件，2 为音频文件。当数值大于 10 的时候，表示为训练集），不会上传，内部鉴别用
宽度	float	图片宽度
高度	float	图片高度

这个表结构就是我们设置的数据库中文件相关表的各个属性和信息，一些主要的参数说明如下：

- 索引 id。这个表项在建表的时候不能为空，为一条数据的唯一标识。也是一个递增的索引。当数据每增加一条，此 id 便会增加 1，并和新的数据存储在一条数据内。
- 照片文件名。对于一个文件来说，是不可能直接存在数据库里的。一是在数据库里很难直接存一个文件，二是文件一般说来，占用空间过大，如果存到数据库里，则会导致数据库空间过大，不利于其他操作。因此，对于一个文件来说，在数据库中存放的是文件在智能手机内部的绝对路径，只是一个字符串，通过它去寻找文件。

- Tag。这个是一个标记位。由于我们的这个数据库只是用来存储文件的，并非只存储照片。而且，对于照片来说，他的用处也不全一样。有的是实时景色，有的是作为 PM2.5 模块的一个模型图片存在的，为了区分这些文件，设置了一个 tag 属性。是一个整数值，当该值为 0 的时候，表示是正常的照片，1 为视频文件，2 为音频文件。当数值大于 10 的时候，表示的是该文件是一个照片且作为 PM2.5 模块的一个模型图片存在的。3-10 是正常的预留位，暂时没有什么效果，但以后有更多的文件类型的话，可以使用他们来定义。该值在上传的时候不会进行上传，只是在客户端内部自己进行判定存在的。

上面的这些属性，在上传的时候会先写入照片内部，作为照片格式的一中说明进行上传，这样可以节省很多空间，只需要进行文件的上传即可。商定好协议之后，在服务器进行解析存储。

5.1.1.2 数据相关表

表 5-2 数据相关数据库表

内容	类型	说明
索引 id	int	用来作为唯一标识的序列，数据库保证其递增且不重复的
光线	float	当前手机周围的光照强度
声音强度	float	当前手机周围的声音强度
充电状态	int	0：未充电；1：充电
电量状态	int	当前电量的剩余百分比
网络状况	int	0：无网络；1：2G；2：3G；3：4G；4：WIFI
经度	float	位置的经度
纬度	float	位置的纬度
时间	DATE	采集的时间

这个表结构，就是我们在本课题中定义的数据相关的信息的存储格式，具体的信息个别说明如下：

- 索引 id。这个表项和文件相关表中的索引 id 有相同的作用。在建表的时候不能为空，为一条数据的唯一标识。也是一个递增的索引。当数据每增加一条，此 id 便会增加 1，并和新的数据存储在一条数据内。
- 充电状态。因为有可能会在充电或者非充电的情况下进行不同的操作，如电量过低时，不上传或后台运行数据，但是在充电的情况下就可以进行相应操作。此数据是一个额外判断量，因此也需要收集。是一个整数，当为 0 的时候，表示的是未充电；为 1 的时候表示的是正在充电。

- 网络状况。同样，网络状况也是参与式感知的一个环节。随着科技的发展，越来越多不同的网络类型出现，网络状况的判别也是很有必要。同时，也可以作为额外判别条件来存在。如当网络为 WIFI 的时候，可以上传大量数据，而为其他状态的时候，可以暂时不上传数据。此属性为一个整数值，当为 0 的时候表示当前的智能手机尚未联网；当为 1 的时候表示的是使用的 2G，2 的时候使用的是 3G，3 的时候使用的是 4G，4 的时候为 WIFI。这样，即使以后再增加新的网络状况，也可以通过新的数值来进行存储表示。

这些数据的相关信息可有可无，为一个整体。当主要采集光照强度的时候，可以不采集其他属性。最后组成 JSON 字符串就可以上传到服务器了。然后服务器对 JSON 字符串进行解析，可以得到对应的数据。

5.1.2　文件模块

按照上个小章节所说，对于文件的存储，在数据库中只存储相应的路径和一些其他的字符串信息，真正的文件是存储在文件系统中的。Android 有很多种方式提供文件操作，这里介绍在我们本项目中使用的方式。

5.1.2.1 资源文件

如果应用程序需要外部资源，那么可以通过 Android 项目建立的时候自动创建的资源文件夹来实现。这种只适合静态的资源存储和调用。如果有个照片显示，需要一个默认图片的时候，使用资源文件是非常方便的。

这些资源文件，是在一个 Android 项目创建的时候就存在的。当把对应的资源文件放置到对应的目录下的时候，在发布 Android 应用程序的时候就会自动包含这些资源。

在程序中，为了访问这些只读的资源，需要调用应用程序的 Resource 对象的打开资源的方法，一遍基于指定的文件接收一个输入流。主要的相关函数如下：

- getResource：获取相应的资源对象。
- getXXX：此函数有多种形式，目的在于根据资源 id 获取相应的实际资源。包括字符串资源，绘图资源，数组资源等。这些资源都是在 XML 中定义的，直接使用即可。

这种方式是 Android 内部提供支持的，开发者不仅可以在开发代码中使用这些资源，也可以在其他的资源文件中直接引用 id 来使用，非常方便。这种资源文件的方式，只适用于一些静态的资源，而不会随着程序的运行而改变的资源。对于默认的图标、字符串等非常方便。

5.1.2.2 存储文件

对于一些非固定的文件资源，如对于照出来的照片，我们要能做到实时存储，这个显然不能存储到正常的资源文件里的。因为他是动态的进行文件存储和查看，而非静态的。因此，需要一些其他的方式进行存储。

Android 操作系统建立在 Linux 操作系统智商，使用的语言大多数情况下是 JAVA，因为我们既可以使用 JAVA 中的文件 IO 操作来进行文件的存储，也可以使用 Android 内部提供的文件 IO 接口。说明大致如下：

- FileInputStream openFileInput(String name)。打开应用程序的数据文件夹下的 name 文件对应的输入流。
- FileOutputStream openFileOutput(String name, int mode)。这个接口是打开应用程序的数据文件夹下的 name 文件对应的输出流。
- File getFileDir()。获取该应用程序的数据文件夹的绝对路径。
- deleteFIle(String)。删除该应用程序下的数据文件夹下的指定文件。

还有很多其他的接口，在这里就不一一列举了。这些接口是 Android 开发环境中的 Context 提供的，方便易用。

但通过上面的接口打开输入输出流的时候，程序所打开的都是应用程序的数据文件夹里的文件，这样所存储的文件大小可能比较有限，毕竟手机内置的存储空间是有限的。

所以，对于手机，有 SD 卡的概念。为了更好的存储应用程序的大文件数据，应用程序需要读、写 SD 卡上的文件。可以通过下列接口实现：

- getExternalStorageState()。这个方法可以用来判断手机上是否插入了 SD 卡，返回 true 表示有 SD 卡。
- getExternalStorageDirectory()。这个函数可以用来获取外部存储器，也就是 SD 卡的目录。通过他再进行一下拼接，就能得到所指定的目录，把得到的大文件数据存储到 SD 卡当中。

5.1.3 系统函数模块

有些时候，应用程序需要少量的数据需要保存，而且这些数据的储存格式都非常简单，都是普通的字符串、标量类型的值等，比如在本课题应用程序中，程序的各种配置信息，数据上传的索引等，对于这类数据，Android 操作系统提供了对应的系统调用函数，SharedPreference 来进行保存。

从实质上来说，SharedPreference 存储是 Android 系统自动完成的，存储在对应应用程序的固定目录下，以 XML 形式来存储。因此，既然是 XML 格式，那么，它

能存储的就是一系列的<key, value>对。

对于 SharedPreference，有一些接口又来获取这些 key-value 对，但是 SharedPreference 接口本身并没有提供写入数据的能力，而是通过其内部接口，调用 edit()方法即可获取所对应的 Editor 对象，通过该对象可以进行写入。接口如下：

- SharedPreference.Editor clear()。清空所有数据。
- SharedPreference.Editor putXxx(String key, String value)。可以存入指定的 key 对应的数值，其中 XXX 可以是 float、int、String 等各种基本类型。这个函数会在没有该值的时候添加一个新的，而有的话，则会覆盖掉旧有的值。
- SharedPreference.Editor remove(String key)。这个可以删除 key 对应的值。

SharedPreference 还有很多的内置功能，都是由 Android 应用程序自己提供的，如可以向其他的应用程序提供属性，也可以读取其他应用程序允许读取的值。这些都是非常容易实现的。在所需要的数据不是太大的时候，可以使用这个。本课题应用程序在配置项，上传索引中使用的该方式。

5.2 控制模块设计

在控制模块当中，最主要的就是调节其他的模块，使整个程序能够快速有效的运行，模块之间尽可能的减少耦合，达到交流顺畅。

控制模块中，可以通过设置模块来让其他模块查看当前设置；可以通过出错管理模块，在程序运行崩溃的时候，查看问题出在哪些模块上，具体原因是什么；可以通过流程管理模块，具体看一下在控制模块当中，如果把几个模块相互连接，让他们相互沟通；可以通过日志模块，来了解如果使得各个模块使用统一的日志接口打印统一格式的日志内容。

下面，就详细介绍一下控制模块中各个子模块是如何实现的。

5.2.1 设置模块设计

设置模块是用来对一些配置进行设置，包括下载设置，调试开关设置，采集设置和语言设置。在设置获取和设定的过程，如图 5-2 所示。

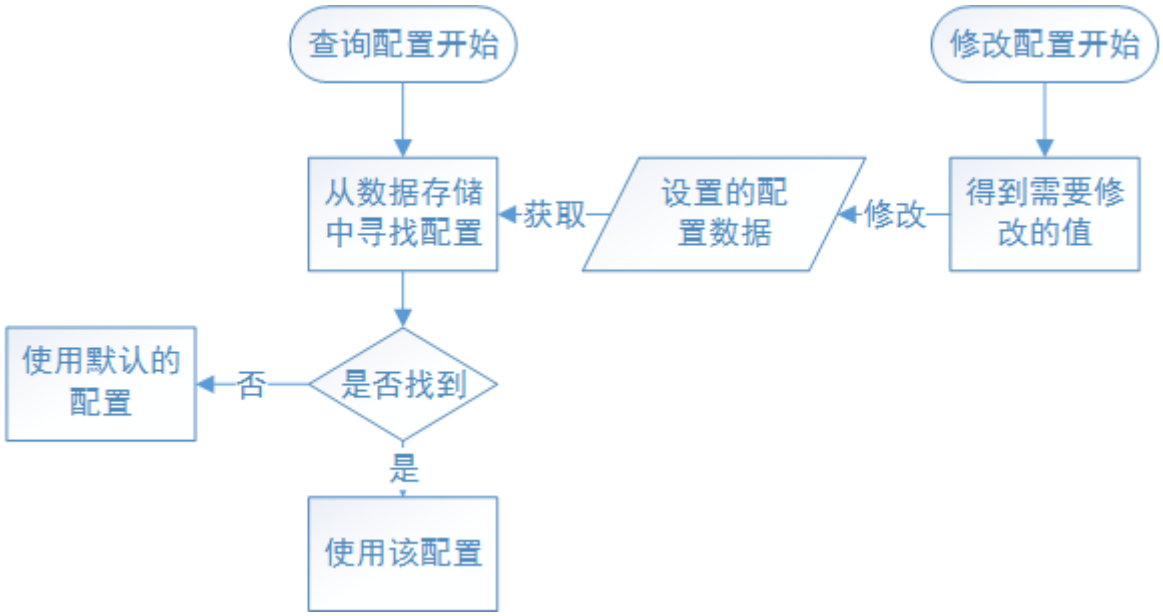

图 5-2 设置模块流程图

对应具体的接口设计如下。

5.2.1.1 下载设置

下载设置是针对照片墙下载图片的时候，距离现在的时间多久的一种设置。主要功能是获取下载的之间周期和设置下载的时间周期。

- 获取下载时间周期：getPhotoWallTime()。返回的是一个整数，指的是时间周期，以小时为单位。
- 设置下载时间周期：setPhotoWallTime(int time)。设置下载时间周期，传入参数为时间，以小时为单位。当用户在设置界面点选时间后调用。

5.2.1.2 调试开关设置

调试开关是针对上传数据时是否有垃圾数据而设定的，若为开，则此次上传为垃圾数据，可以在服务器中删除。具体设置接口如下：

- 获取调试状态：isDebugStatus()。返回一个 bool 值，为 true 证明是调试状态，为 false 说明不是调试状态。
- 设置调试状态：setDebugStatus(bool status)。设置调试状态，当用户选择 off 的时候调用。

5.2.1.3 采集设置

采集设置是针对后台采集的频率进行设置。主要功能是设置采集频率和获得采集频率加以应用。以及针对采集开关的一种设置。具体设置接口如下：

- 获取采集频率：getCollectionFrequence()。获得采集的频率，返回的是一个整数，表示时间间隔，以小时为单位。若返回的值为 0，则表示定时采集关闭。返回值只有大于 0 才有效。
- 设置采集频率：setCollectionFrequence(int timeval)。设置采集的频率。传入参数是时间间隔，以小时为单位。若调用此函数，则不管之前定时采集是否开启，定时采集功能都会自动开启。若传入参数为一个违法值（比如不是整数，或者小于 0），则按照默认的时间间隔 1 小时进行设置。
- 获得定时采集状态：getCollectionStatus()。返回的是一个 bool 值，若为 true，表示的是定时采集开启。否则，表示的是定时采集关闭。
- 设置定时采集状态：setCollectionStatus(bool status)。设置定时采集是否开启，传入 true，则表示开启，否则，表示关闭定时采集。

5.2.1.4 语言设置

对于语言设置来说，只需要设置当前语言即可，会立即生效。因此，该设置项只有一个接口。如下所示：

- 设置当前应用程序的语言。setLanguage(int sel)。对于传入的参数，是一个整数。0 表示的是默认语言，也就是说，和当前 Android 操作系统的语言相一致。当传入的是 1 的时候，表示的是中文。当传入的参数是 2 的时候，表示的是英文。目前该程序只支持这两个语言。在此函数调用的时候，会自动刷新当前的界面和语言。

5.2.2　出错管理模块设计

出错管理模块是用来在整个程序崩溃的时候，进行一种错误记录，以防止莫名其妙的崩溃导致的不知问题所在，故此模块应运而生。该模块的类图如图 5-3 所示。

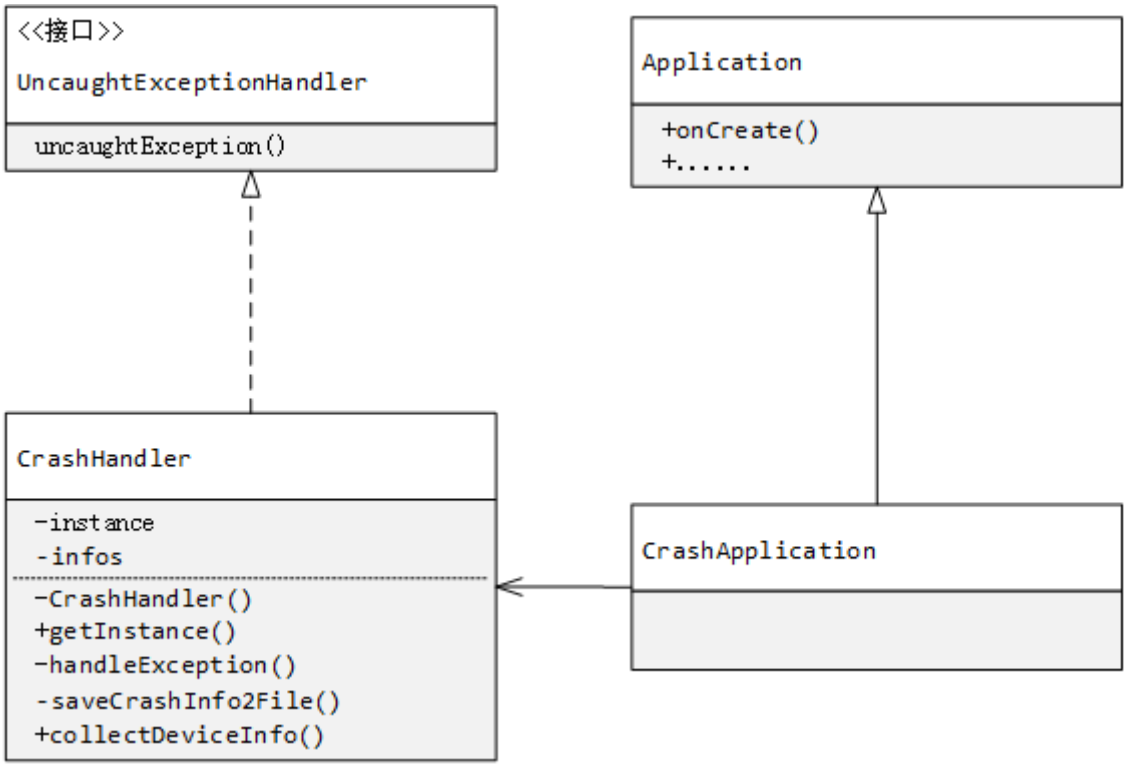

图 5-3 出错管理模块类图

如类图所示，出错管理模块分为两个方面，一个是重写应用，一个是重写异常。

5.2.2.1 重写应用类

对于 Android 应用程序来说，本身自带有应用类，活动类，服务类等。这些类都是一个可以单独活动的类，其中应用类的级别最高，代表着整个应用程序。因此，若想对整个应用程序管理，最好的方法是重写应用类。

应用类（Application 类），包含了很多可以重写的方法，如 onCreate()表示在创建的时候，onDestroy()表示在摧毁的时候，不同的时刻可以重写不同的函数。对此，我们只需要重写创建时候的函数调用，就能很好的在应用程序开始的时候，就把我们自身的东西加入进去，达到目的。

5.2.2.2 重写异常类

异常类有很多，出错管理模块的目的是解决在程序设计的时候未能抓到的异常，因此，对于出错管理模块，我们只针对未抓到的异常进行重写。

对于继承过来的异常句柄，我们采用单例模式，把该类的构造函数设置为私有，这样，再全局调用的时候，只会存在一个该类的实例，进行相应的计算。对于继承了“未抓取异常句柄”类的新类（CrashHandler）的说明如下。

- instance。为这个类的一个实例，为了达成单例模式，每次调用的时候只会对

此实例进行操作。

- infos。为一个 map 的数据接口。表示所有需要储存的信息，包括设备型号，设备名称，出错位置，出错原因等方面。在抓取异常的时候进行赋值。
- collectDeviceInfo()。该函数是用来手机设备信息，包括设备名，设备版本号等。并把这些信息存入到 infos 的数据结构里面。
- saveCrashInfo2File()。该函数是把异常信息保存到文件里。该文件路径已经指定好了。会同时把设备信息和异常信息存入到指定的文件内。返回的是一个 string，表示的是存储的文件绝对路径，方便现在未开发但可能开发的上传错误文件等。
- handleException()。这个函数是作者自定义的错误处理函数，也就是说，当遇到未捕获到的异常的时候，会调用此函数进行处理。在这个过程中，由于已经捕获到了异常，因此，会弹出提示框提醒用户“程序遇到问题，需要关闭”，并会调用 collectDeviceInfo()函数对异常进行分析。然后调用 saveCrashInfo2File()进行保存。返回的是一个 bool 值，成功处理了异常则返回 true，否则返回 false。

重写异常类会被重写应用类在 onCreate()函数中调用，这样，在程序刚开始时，就把重写异常类加载进来，当程序崩溃的时候，按照重写异常类的逻辑进行运行，达到了出错管理的目的。

5.2.3　流程管理模块设计

流程管理模块主要是用来管理一些数据流和一些操作流的相关步骤，达到各个模块之间的低耦合，从而控制各个模块之间的交互流程。下面来说一下本模块中最重要的数据接口部分。

数据接口的类图如图 5-4 所示。

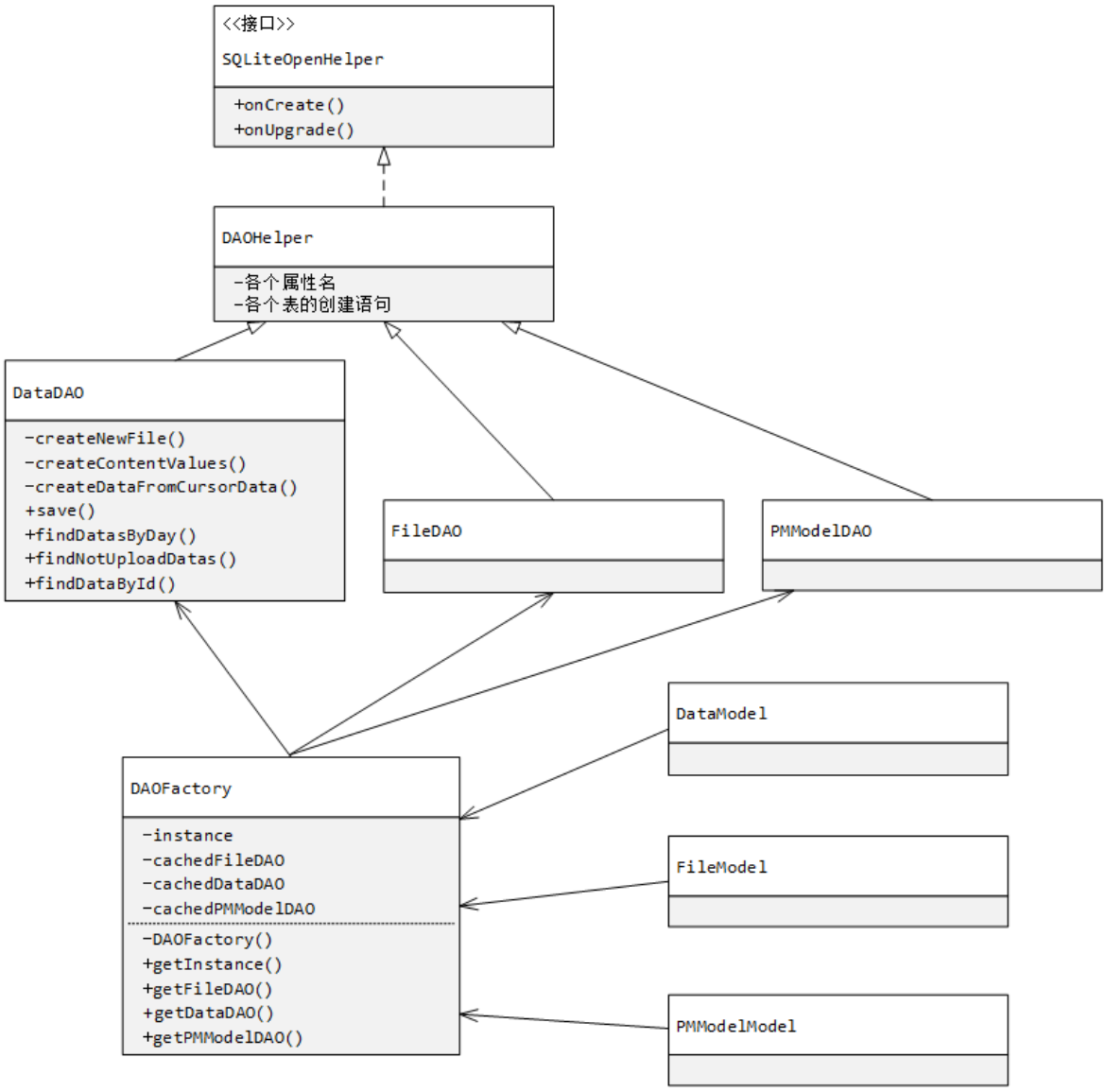

图 5-4 数据接口类图（概括）

如图 5-2 所示，为一个数据接口类图的概括，之所以为概括，是因为很多的类并没有画出其具体的实现，但不难看出，作为流程管理，向其他模块提供接口，一个数据的流程是复杂的。每个流程管理都是如此。整个应用程序采用了 MVC 的框架，数据流程管理也不例外，分为三层：Model 层，View 层，Control 层。在这里进行详细的说明。

5.2.3.1Control 层

Control 层就是所说的控制层。在 MVC 中用来控制数据的走向，对外提供接口，直接和数据库相连。对照类图，Control 层包括除了 model 结尾命名之外的所有类。具体如下：

- SQLiteOpenHelper 类。这个是 Android 操作系统自带的一个接口，用来管理

数据库的创建和数据库的版本，提供了 onCreate 和 onUpdate 两个接口函数。前者表示当数据库第一次创建的时候所执行的函数，后者代表着数据库版本更新所执行的函数。

- DAOHelper 类。这个类继承了 SQLiteOpenHelper。具体实现了该接口。另外，在私有成员变量中，定义了各个属性的字符串以及定义了数据库中各个表创建以及删除时候的语句。在创建函数重写的时候，进行了数据库和数据库表的创建。当更新的时候，重写为删除数据库表和重新创建数据库表两个步骤。
- DataDAO 类。继承了 DAOHelper 类。这个类是关于数据库操作的具体实现。对外的接口有 save()，findDataById()等。对于 save()函数，传入的参数是一个数据类，这样，对于传入的数据，通过对应的 SQL 语句来进行存储，并把存储过后重新赋值的数据类返回，供其他接口调用。其他函数也是类似的功能，不过具体的传入参数和返回值不太一样。DataDAO 这个类的作用就是直接对数据库进行操作，为其他模块提供接口。同样，FileDAO 和 PMModelDAO 也是同样的作用，分别是直接对数据库进行文件和 PM2.5 相关数据的存储和查找等操作。在这里不再细说。
- DAOFactory 类。这个类是一个单例模式。在本课题应用程序中，此类的实例只会存在一个。这样可以避免出现多个数据对同一数据库进行操作，出现导致数据不同步的问题。在这个类中，主要的函数是 getXXXDAO()，其中 XXX 代表着上面说的三个类。当有人调用的时候，若第一次调用，则创建一个对应的实例，否则直接取已有的实例进行数据库操作即可。

综上所述，数据流程管理的控制层其实用了一个设计模式中的工厂模式。当需要哪些对应的类，只需要进行调用相关的函数即可获得，而对应类内部如何实现，外部不需要知道，只知道统一的 DAOHelper 和对应的接口即可。而且，要想增加新的数据库表的操作，只需要改很少一部分就能够实现，降低了各个类之间的耦合性。对于其他的流程管理，Control 层类似，在这里就不再一一叙述了。

5.2.3.2Model 层

Model 层就是所谓的模型层。在类图中表现出来的是 DataModel、FileModel 和 PMModelModel 这三个类。

每个类都表示对应数据的一个模型，也就是对应表中具有哪些属性。如 DataModel 类的私有成员变量定义如下：

```
public class DataModel {
    private Long m_id = null;
    private Float m_lightIntensity = null; //表示光照强度
```

```
    private Float m_soundIntensity = null; // 表示声音强度
    private Date m_createTime = null; // 表示创建的时间
    private Integer m_chargeState = null; // 表示充电状态，0 为未充电，1 为充电
    private Integer m_batteryState = null; // 表示电量百分比，以整数表示
    private Integer m_netState = null; // 表示的是网络状况，是否连了 WIFI
    private Float m_longitude = null; // 表示采集时所处位置的经度
    private Float m_latitude = null; // 表示采集时所处位置的纬度
};
```

这些包括了数据库中数据相关表的光照强度、声音强度等所有信息。另外，除了在该类中定义正常的私有变量来进行存储，还会定义与对应的 DAO 完全一致的函数，如也会定义 save()函数等。这些函数通过调用 DAOFactory 类中的 getDAO()函数得到对应的数据操作类，然后执行相应的操作函数进行操作。在类创建的时候，提供了各种可能的构造函数，有默认的，也有全部参数传递的，方便使用者进行调用。

其他的 Model 类也是类似的情况，Model 层主要是用来对数据的一种归纳，然后通过调用 Control 层中对数据直接进行操作的类，对外提供接口。用户完全不用知道内部是如何实现的，只需要调用接口即可。

5.2.3.3View 层

View 层就是所说的界面层。大部分是需要用户接口模块来进行调用的。主要是包括各种界面的调用等。当用户需要储存的时候，新建一个对应的模型类，然后进行接口调用，保存，就可以完整的实现保存新的数据的功能。同样，当用户需要观察一个数据而进行调用的时候，只需要通过 get 函数就可以获得对应的数据。get 函数多种多样，在控制层中进行真正的控制。

View 层可以通过直接使用 model 层的东西，而不需要知道控制层是如何实现的，通过 MVC 这个框架模型，很好的解决了耦合性。

5.2.4 日志模块设计

日志模块的作用就是通过一个统一的接口，使得整个应用程序在运行的过程中，能够调用该统一接口，进行统一的格式记录，达到标准化的目的。类图如下：

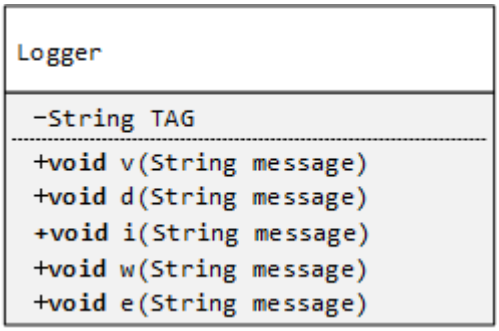

图 5-5 日志模块类图

如图 5-5 所示，日志管理模块的类比较简单，主要有一个私有成员变量和 5 个函数组成。说明如下：

- TAG。指的是一个固定的标识，用来说明日志的拥有者，在这里固定为“theSenser”。这样当日志打印出来的时候，在每个日志之前，都会告知属于“theSenser”的应用。
- v()。该函数表示记录日志级别为 verbose，一般用来调试用。传入参数为需要打印的信息。
- d()。该函数表示记录日志级别为 debug，一般用来调试用。传入参数为需要打印的信息。
- i()。该函数表示记录日志级别为 information，一般用来直接输出信息。传入参数为需要打印的信息。
- w()。该函数表示记录日志级别为 warning，一般用来输出警告，证明程序走到了不想走到的地方，但不影响整个程序的运行。传入参数为需要打印的信息。
- e()。该函数表示记录日志级别为 error，一般用来输出错误信息，程序运行错误，正常用来打印可以让程序崩溃的错误。传入参数为需要打印的信息。

上面的一个数据，五个函数都是全局的静态的变量和函数，这样能够不需要新建对应的类就可以进行使用，方便有效。

5.3 用户接口模块设计

用户接口模块，主要是用来给用户提供一些接口，方便用户操作，能够使用户在不需要看任何说明文件的情况下，就能够快速上手使用。按照概要设计来说，主要分为三大类，下面一一进行详细设计介绍。

5.3.1 主动采集接口设计

主动采集分为照片主动采集和数据主动采集。

5.3.1.1照片主动采集

对于照片，我们主动采集就是直接点击“拍照”按钮，根据不同的场景采取不同的设施。接口函数为 takephoto()。

此函数时总调用函数，用户使用的时候直接调用即可。对于不同的界面，按钮都是同一个。因此，在函数内部中，要先自身判断一下当前的拍照场景属于哪个页面，分两种情况，调用两个不同的函数：

- takePhotoForPredictOrSetModel()。当页面处于“PM2.5 预测”的界面的时候，该函数被调用。该函数会唤起 PM2.5 模型的页面，然后用户可以根据模型选择，然后会传入不同的参数给采集模块的接口，进行照相。
- takePhotoForRealTimeScene()。当页面处于“实时景色”的界面的时候，该函数会被调用。该函数会调用采集模块无参数的接口，意为只负责拍照，而无需添加其他参数。这样，拍下来的照片处于原生态，不会被用来进行 PM2.5 建模，也不会用来做其他事情，单纯的用来展示。

这样，当用户点击“拍照”按钮的时候，会根据不同的场景，进入不同的二次画面，让用户进行不同情况下的照片采集。

5.3.1.2数据主动采集

同照片主动采集类似，用户对于数据的主动采集直接点击按钮“刷新”即可，会根据不同的场景采集不同的数据。接口函数为 refreshDataView()。

当调用此函数的时候，对于当前所处的不同界面，会有不同的调用。但是按钮丢失同一个，也就是说，在函数内部中，要先自身判断一下当前的拍照场景属于哪个页面，两种情况，如下所示。

- 当处于光照强度数据展示的页面时，获得光照强度所处的页面的页面句柄，同时该页面句柄也是一个类，调用该类内部的刷新函数即可。类内部的刷新函数调用的是采集模块的光照采集接口，返回一个数值后，显示到界面上。
- 同样，对于声音强度的数据展示页面时，会先获得声音所处的页面的页面句柄，调用该类内部的刷新函数。刷新函数会调用采集模块的声音采集接口，在得到采集的数值后，反馈给页面部分。

但是，与照片的主动采集不同，照片主动采集是对待页面分别刷新。而数据的主动采集，由于数量较小，而且采集越多越好，因此，每点击一次“刷新”按钮，

就会对所有的数据采集页面进行刷新。不在当前显示范围内的页面，也会进行刷新，只是看不到动态的变化罢了。当用户切入到另一个数据显示的界面后，显示的采集数据为最近一次点击刷新的时候。

5.3.2　信息展示接口设计

所谓的信息展示，是指针对不同的信息格式，会采取不同的方式来展现不同格式的信息。在这里分为数据信息展示、PM2.5 展示、周边环境展示三个方面来进行阐述。

5. 3. 2. 1 数据信息展示

数据信息在这里主要包括光照强度和声音强度。其他的诸如地理位置，面朝方向等，暂时作为辅助的数据进行存储，便于后台进行相关信息的收集和计算。因此，这里主要对光照强度和声音强度的数据信息展示进行详细设计。单独拿光照强度进行举例说明，类图如图 5-4 所示。

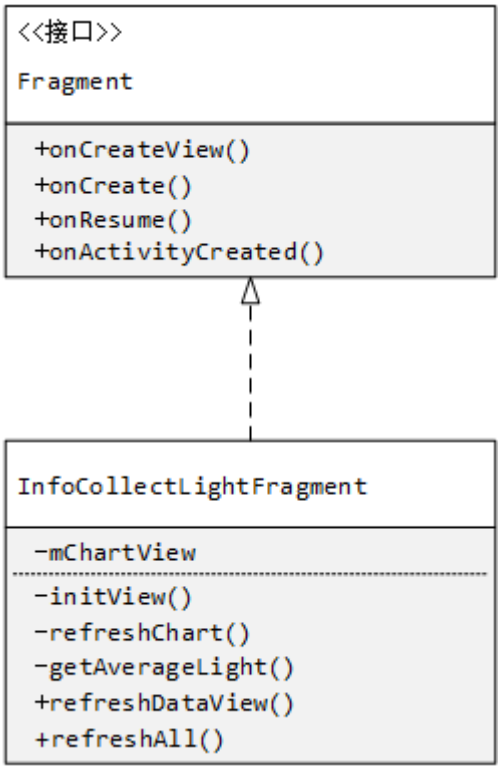

图 5-6 光照强度展示类图

如类图 5-6 所示，光照强度的信息展示类继承与 fragment 这个接口。在 fragment 里有几个接口函数，分别表示对应的生命周期会调用的函数，目前，我们只用了如下四个。

- onCreateView()。表示当该 fragment 中的布局被创建的时候，所执行的函数，

一般是用来加载布局的时候使用。

- onCreate()。当该 fragment 被创建的时候，所执行的函数，一般是用来初始化数据的时候使用。
- onPause()。当该 fragment 暂停的时候，所执行的函数，一般指的是切换到其他的界面。
- onResume()。当该 fragment 恢复的时候，所执行的函数，一般指的是从其他节目恢复回来。

然后，用一个新的类继承并实现这些接口，用上面这 4 个接口实现界面的展现，下面的接口用来处理逻辑。

- initView()。这个是用来初始化图表的，在这个函数中，会先计算 7 天前的日期，然后通过数据管理模块获取每天的光照强度采集数据列表，然后将每天的平均值显示在图表上。被 onCreate()调用。
- refreshChart()。表示的是刷新图表，即获取每天的光照强度采集数据列表，然后将每天的平均值显示在图表上，会被本模块调用。
- refreshDataView()。表示刷新当前数据。即将获取到数据显示在界面上，会被本模块和主动采集模块调用。
- getAverageLight()。表示计算一个数据列表的加和平均值，被 initView 调用。

通过这些函数，就很容易让此模块或者其他模块调用对应的接口来进行数据展示。同样，对于声音强度，也是同样的方式进行展示。由于开发量较大，这里的图表显示使用了开源的工具进行显示，加速了开发效率。

5.3.2.2PM2.5 展示

PM2.5 展示的主要依据是利用同组的其他人开发的 PM2.5 预测模块来进行的，是对参与式感知的数据展示的一种另一种表达形式，是对参与式感知的积极探索。

1) PM2.5 的预测是基于采集、计算、信息展示为一体的，流程如下。
2) 用户在 PM2.5 预测的页面的时候，点击“拍照”按钮进入 PM2.5 预测模块。
3) 用户选择或新建一个模型来进行拍照预测。
4) 拍照结束后，会结束当前拍照页面，返回模型模块界面，此时弹窗显示当前正在预测。
5) 用户此时点击同一个模型会提示正在预测，不允许预测。
6) 用户返回显示界面会看到正在预测的字样。
7) 用户在步骤 2 的时候，可以长按一个模型查看模型的具体详情。

我们发现，在 PM2.5 展示的时候，界面繁多，功能多样，需要展示结果的页面很多，为了方便开发和后续的代码迭代，我们选用了下列的方式进行编写。类图见

5-5 所示。

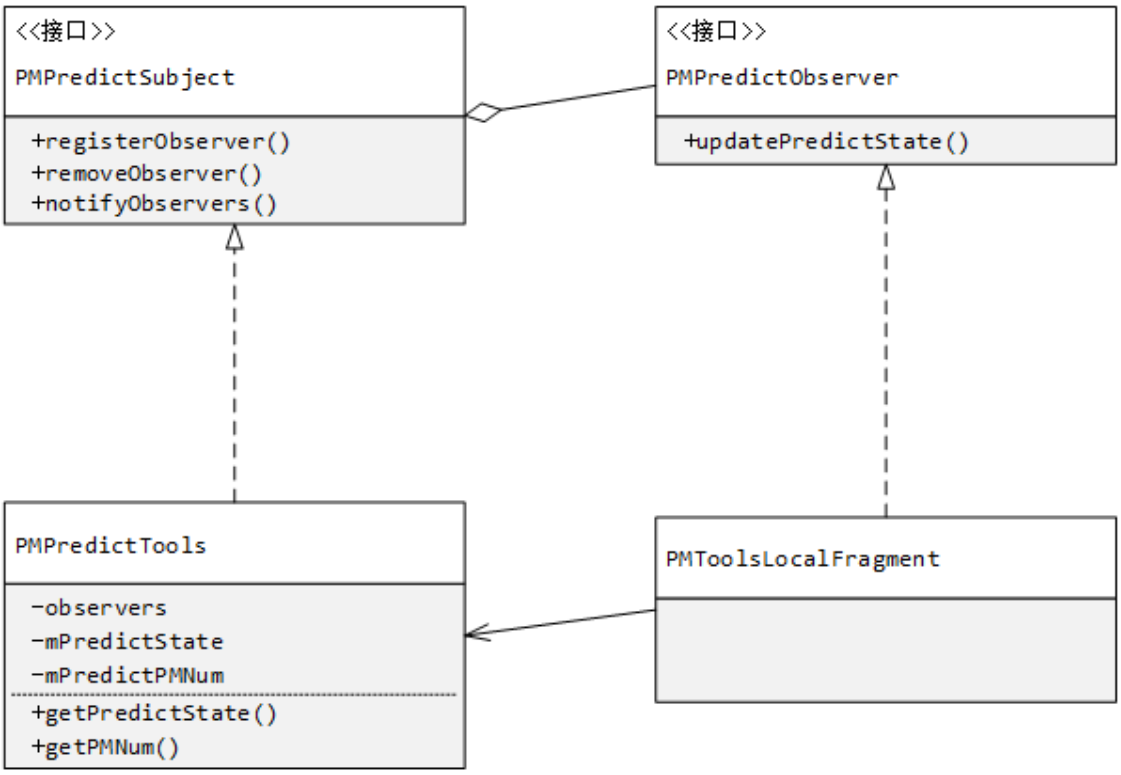

图 5-7PM2.5 展示框架类图

图 5-7 表示了在 PM2.5 展示这个模块内，所具有的框架类图。该框架采用了观察者的优秀设计模式，在类图中只写出了关键的几个接口函数和类函数，下面进行一一说明。

首先，有两个纯接口，PMPredictSubject 表示的是被观察者，在本 PM2.5 显示模块中表示的是 PM2.5 预测模块。而 PMPredictObserver 表示的是观察者，在本 PM2.5 显示模块表示的是界面。

真正的继承被观察者接口的是 PMPredictTools，为控制模块中 PM2.5 模块与其他模块交流的流程管理模块的一部分，属于 MVC 模式中的 Control 层。该模块可以调用 PM2.5 预测模块的接口进行相应的建模分析，PM2.5 值的预测等。这里继承了被观察者，主要实现了以下方式。

- observers。为一个列表，表示的是观察者的列表。
- mPredictState。为当前预测的状态，为一个整数。包括正在预测，预测失败，预测成功等状态。
- mPredictPMNum。只有在预测状态为预测成功的时候有效，表示当前预测的有效 PM2.5 的值。
- getPredictState()。得到当前的预测状态。
- getPMNum()。得到当前的 PM2.5 值。
- registerObserver()。是继承过来的需要实现的接口。表示对观察者的注册。当

有观察者来临的时候，调用此接口就能够对观察者进行注册。

- removeObserver()。继承过来的需要实现的接口，表示解注册。当不需要观察者来观察该类的时候，调用此函数。
- notifyObservers()。表示的是通知观察者。根据此类在不同的 PM2.5 预测阶段可以调用此函数来通知观察者，这样观察者就能够知道有相应的时间发生，然后通过 get 函数来获得当前的状态。该函数会依次遍历 observers 里的所有观察者，依次调用对应的观察者中的 updatePredictState()函数来提醒观察者有事件发生。

同样，对于观察者来说，比较简单，继承 PMPredictObserver 接口即可。当被观察者有状态改变的时候，观察者可以通过 updatePredictState()函数的实现来进行对应的操作，一般情况下，当收到通知后，会调用被观察者的 get 函数来获取对应的状态，进而来进行不同的处理。比如，在预测成功的时候显示“预测成功”，并且再次去获取 PM2.5 值；在预测失败的时候显示“预测失败”，而不会继续显示预测的值。

观察者模式是一个非常好的处理这种一个变化，其他模块都需要知晓的情况。在 PM2.5 展示的过程中，需要得知 PM2.5 预测状态的函数，都属于观察者，而被观察者只有一个，那就是 PM2.5 预测模块的接口。

通过这种方式，我们可以很容易编写代码来进行 PM2.5 的展示，无论当前用户处于哪个需要展示的页面，都能获得准确的最新信息。

5.3.2.3 周边环境展示

周边环境模块是指用户可以上传自身所处的环境照片，并且可以下载他人上传的照片进行查看的模块。

该展示采用了照片墙作为信息展示的一个窗口，非常简单，继承一个 View 的接口，然后自身实现一个每行有三个按钮的可滚动的视图，同时对点击和手势进行监控，当向下滚动的时候，从服务器请求一部分数据，若数据为空，则显示已经到底，若数据存在，则显示对应的数据。

对于点击事件，在每个数据里面都会存有对应原图的下载地址。当用户进行点击后，会向原图的下载地址请求下载，当下载成功后，则显示对应的图片。

在这个过程中，由于数据过于复杂。通信没有经过通信接口模块，而是直接与此相连。全程为 json 格式进行传输。自定义的通信的接口如下：

对于请求的通信，request_type 表示请求类型，request_maxnum 表示请求的最大照片数量，begin_time 表示请求的开始时间。例如下面的例子：

```
{   “request_type”:”photo_list”, // 请求类型
    "request_maxnum": "50", // 请求照片的最大数量
    “begin_time”:” 174940285028” } // 请求的照片的起始拍照时间
```

对于返回的格式。response_type 表示请求类型，response_num 代表返回的照片的实际数量 response_photos 表示返回的照片数组，每个由 photo 数据由 pack_url（压缩图片路径），src_url（原图路径），photo_time(该照片拍摄时间)组成。例子如下：

```
{"response_type": " photo_list",    // 请求类型
    “response_num”:”2”,   // 请求数量
    “response_photos”:[   // 照片列表
        {
            “pack_url”:”XXXX.jpg”,   // 压缩图片的路径
            “src_url”:”ddaf.jpg”, // 原图片路径
            " photo_gps": { "gps_lat":"102.12","gps_log":"23.45"}, // 照片 gps 信息
            “photo_time”:”174940285028”  // 照片拍摄时间
        },
        {
            “pack_url”:”daf.jpg”,
            “src_url”:”rat.jpg” ,
            " photo_gps": { "gps_lat":"102.12","gps_log":"23.45"},
            “photo_time”:”174940285028”
        }
] }
```

这个例子返回了 2 张照片，信息分别如内部所示。

5.3.3　其他接口设计

对于其他的接口，有很多，但大部分在前面各个模块设计中已经包含到了。对于用户接口模块，采用了一个抽屉式的布局，方便开发。这个布局采用了开源项目的框架，就不在这里详细叙述了。只需要调用他们提供的接口，就能很方便的完成界面的开发。

第六章　Android 平台参与式感知应用的测试与效果展示

本章主要对系统进行验证和实现，首先介绍系统的环境，然后对系统进行验证测试和效果展示。证实其可用性和可开发性。

6.1 测试与验证

6.1.1 测试与开发环境

对于 Android 平台上的应用开发，本课题系统采用以下的开发工具和环境进行相关开发与测试：

- Windows 操作系统
- Eclipse 开发平台，用于开发代码。
- JAVA 语言，对于不需要底层开发的 Android 应用程序来说，一律使用 JAVA 进行开发。
- Samsung GALAXY S4。一个 Android4.4 操作系统的实体机，用于对操作系统进行测试。

服务器端和 PM2.5 模块的开发工具不在本课题考虑范围内，因此不做介绍。

测试环境图如图 6-1 所示。

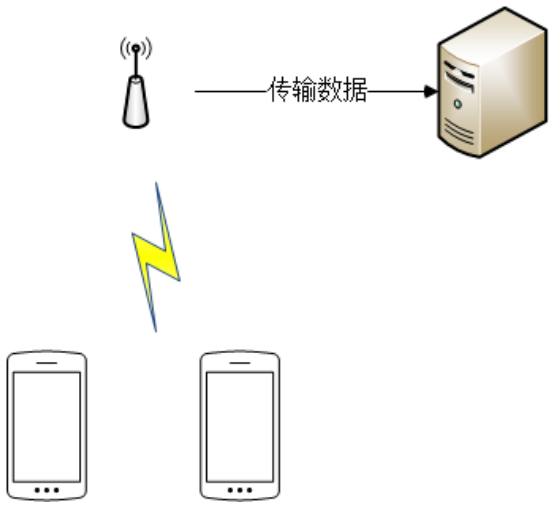

图 6-1 测试环境图

如图我们可以看出，在测试环境中，我们采用的是局域网，智能手机通过无线路由器与网络进行连通，同时多台手机进行测试。

6.1.2 单元测试

单元测试主要对参与式感知应用的各个模块中的相关函数进行测试，伪造一个输入，来查看输出是否符合要求。

作者所做的单元测试只是针对作者参与的模块，包括数据管理模块，控制模块，和用户接口模块。

单元测试不是一个容易的过程。为了达到单元测试的目的，在开发的过程中专门写了一个测试的页面，用来测试各个模块的输入输出。如数据管理模块中的存储过程，用户可以根据提示自己输入数据，然后通过查询来查看是否存储在数据库中。控制模块中的设置模块，通过设置一些数据，然后点击测试里面的对应按钮，可以查看当前设置的值。

通过以上的方式，进行各个接口的单元测试，各个接口与预期相符，符合接口的设计。

6.1.3 功能测试

我们对功能的测试是一种集成测试，是把各种模块连通进行测试的一种方式。在这里，我们根据需求分析，按照展示的内容进行划分，包括对 PM2.5 的采集与预测，照片的采集与展示，信息数据的存储与展示等功能。

6.1.3.1PM2.5 的功能测试

表 6-1PM2.5 功能测试测试用例表

编号	功能描述	测试用例描述	预期结果
1-1	进行新模型预测	拍照->新建模型->命名->照相完成	返回模型预测失败，原因为模型较少
1-2	旧有数量不足的预测	拍照->选择数量较少的模型->照相完成	返回模型预测失败，原因为模型较少
1-3	旧有数量充足的预测	拍照->选择数量较多的模型->照相完成	返回预测成功，并显示预测的 PM2.5 值
1-4	预测过程中的状态显示	拍照->选择其中一组模型->照相完成	再次点击该组模型，显示正在预测

6.1.3.2照片采集与展示的功能测试

表 6-2 照片采集与展示功能测试测试用例表

编号	功能描述	测试用例描述	预期结果
2-1	无网条件下，照相上传	拍照->上传	上传失败
2-2	有网条件下上传	拍照->上传	上传成功
2-3	有网条件下上传，查看照片墙	拍照->上传->重新进入照片墙	上传的照片在照片墙内显示
2-4	查看别人上传照片	拍照->上传->另外一台手机进入照片墙	上传的照片在照片墙内显示

6.1.3.3 信息采集与展示的功能测试

表 6-3 信息采集与展示功能测试测试用例表

编号	功能描述	测试用例描述	预期结果
3-1	收集当前信息	刷新	显示当前的采集数据
3-2	显示一周内的信息	采集一周数据->刷新	图表显示最近一周内周围环境变化曲线

在上述测试中，考虑了各种的正常条件与边界条件。经过各种情况的多次组合测试，测试用例全部通过。测试结果表明，本系统已经实现了参与式感知的所有初步功能，符合功能需求。

6.1.4　性能测试

6.1.4.1 内存使用测试

在 Android 应用程序中，可以通过使用 eclipse 中的 DDMS（Dalvik Debug Monitor Service）来查看当前的一些性能使用状态。我们在运行的过程中，通过 DDMS 调试工具分别查看对应的内存使用情况。如图 6-2 所示。

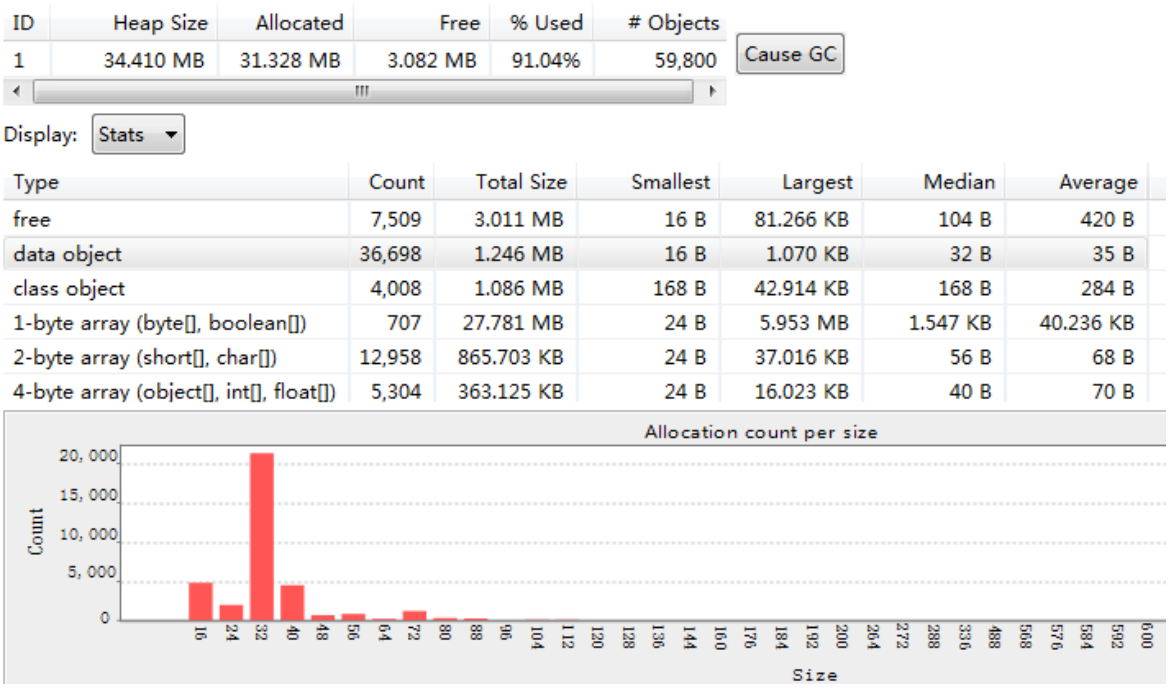

图 6-2 参与式感知应用内存使用情况

如图所示。在本课题应用的运行当中，使用的内存 31MB 多，其中，数据占用的内存为 1.2MB 多，图片占用的内存（1-byte array，bitmap 占用，表示的是图片占用的内存）为 27MB 多，由于本课题应用会处理大量的图片，这些数据是符合预期的。

在 DDMS 开启期间，不断地使用本课题应用来进行一些操作，内存值没有明显的增加和减少，这样可以证明，本课题应用的开发没有内存泄露的情况出现。

6.1.4.2 稳定性测试

在开发的过程中，遇到崩溃的现象时，会调用控制模块进行错误管理，将错误内容写入到文件中，方便了排查。

在开发结束后，使用多台手机共同上传或者下载，并进行多次不同的操作，未出现崩溃的现象，稳定性良好。

通过这些测试，可以初步得到性能的一些使用情况，内存消耗较少，稳定性较高，满足需求。

6.1.5 测试总结

经过各种情况的多次组合测试，测试用例全部通过。测试结果表明，本系统已经实现了参与式感知的所有初步功能，符合功能需求。系统的稳定性较好，在使用的过程中内存使用不高，能够在很多 Android 4.x 平台以上正常运行，符合非功能性

需求。

6.2 效果展示

6.2.1 数据部分

对于数据采集模块，效果如图 6-3 所示。

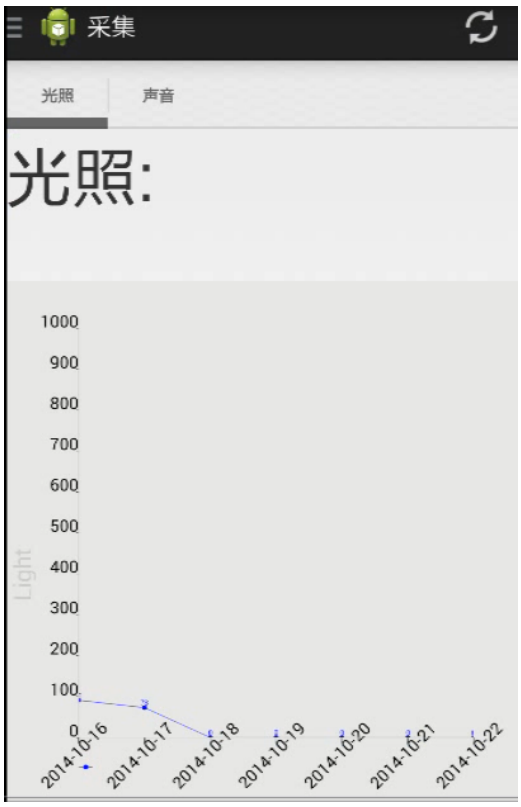

图 6-3 数据部分效果展示

可以看到，上半部分为实时信息展示，此时并没有按下右上角的刷新键。下半部分为进一个星期每天的平均值。由于测试手机不常用，在几天之内光照几乎为 0，也为正常现象。

6.2.2 PM2.5 部分

效果图如图 6-4 所示。

图 6-4PM2.5 展示效果图

对于此时的界面，是预测完的界面，预测的结果已经显示在了左上角。界面为英文，是调整了设置项为英文的结果。

6.2.3 照片墙部分

效果图如图 6-5 所示。

图 6-5 照片墙效果展示

如图可以看到，对于照片墙来说，以墙的方式展现了数据，当点击对应图片的时候，可以看到放大后的原图。

第七章　总结与展望

7.1 论文工作总结

随着移动互联网、物联网和社交网络的发展，智能手机和平板电脑逐渐成为人类生活中不可或缺的一部分。而智能手机，正在成为人与人之间交流的主要平台之一。人们越来越习惯于社会发展给人们带来的舒适生活，拍照，录音无所不在，而随着传感器的发展，人们除了天气，也越来越关心各自周围环境的变化。在此基础上，本课题论文应运而生。本文基于 android 操作系统的平台，进行了参与式感知应用的研发，设计并开发了对应的应用，来为广大用户和以后的开发者做个开头。

在介绍完相关背景后，本文首先介绍了 android 平台下的技术背景和一些开发所需要的基本知识。然后，通过对需求分析的详细叙述，通过用例图等多种方式，慢慢阐述出了一个应用开发的需求背景。然后根据需求，对所需要做的应用进行了概要设计，通过流程图等方式进行了相应的模块划分。再次，本文着重对作者负责的模块进行了详细设计分析，结合了类图、数据结构等多种方式，来进行了详细的设计阐述。在概要设计和详细设计期间，又对大的模块进行了更进一步的细分，使结构更加清晰明了。最后，在设计并编码实现的基础上，对系统进行了验证并给出了效果演示图。

此外，除了功能性方面的需求外，本课题项目还遵循高内聚，低耦合的思想。采用了多种框架、设计模式相结合的方式来构建系统。使得系统内部各个模块相对独立，易于开发。也有利于模块划分，明确各个模块职责，利于以后模块功能的扩展。

本次参与式感知应用既有传统的感知体系，如光照声音，照片等，也有尝试性的感知与当代技术相结合的突破，如 PM2.5 的预测。给用户以新奇和基础，方便用户使用。

限于篇幅，本文中不可能给出所有在开发过程中的全部细节，只能从作者参与的模块入手，大体分析，详细讲解，涵盖了 android 操作系统应用的大部分主要功能。

7.2 问题与展望

目前，虽然已经基本完成了参与式感知应用的开发，但还是有很多不足之处。

用户界面简单。目前的用户界面大部分采用了基本的元素，没有使用过于华丽的界面接口，主要开发工作侧重于功能方面。因此，用户界面稍显简单，以后可以

更加美化一些。

系统的安全性。目前系统没有考虑过安全性，只是实现了大部分功能。程序几乎没有反破解的机制，在设计代码的时候未考虑此因素，以后可以加强。

用户隐私性问题。由于本次开发工作量较大，对于用户接口来说，只是上传了对应手机的 IMEI 码，并且在任何数据传输的过程中没有进行加密。这些可能会破坏安全性和用户隐私，我们以后需要在这方面进行加强。

减少内存占用率。目前只针对正常情况进行了内存优化。当有很多照片的时候，内存可能还是会不够用，或者还有一些其他没有考虑的情况。这些都是以后发展的目标。

参考文献

[1]李怀瑜，朱瀚，肖汉，等. 基于位置的参与式感知服务 [J]. 北京大学学报:自然科学版, 2014, 50(2): 341-347.

[2]刘威. SoPhoNet--基于参与式感知的社交拍照感知、分享与展示[D]. 浙江: 浙江大学, 2012.

[3]百度百科[EB/OL]. http://baike.baidu.com.

[4]CSDN[EB/OL]. http://blog.csdn.net/wangloveall/article/details/8033725

[5]李刚. 疯狂 Android 讲义[M]. 第二版. 北京：电子工业出版社，2013.

[6]戈振兴，边静. Android 体系结构剖析 [J]. 科技信息，2011,12: 204-205.

[7]黄蓉. Android 消息处理机制的研究 [J]. 黑龙江科技信息，2012,33: 87-87.

[8]Reto Meier. Android 4 高级编程 [M]. 余建伟，等译. 第三版. 北京：清华大学出版社，2013:433-435.

[9]Eric Freeman,Elisabeth Freeman 等. Head First 设计模式[M]. 东南大学出版社, 2005.

[10] CSDN[EB/OL]. http://blog.csdn.net/xiaanming

[11]官方 Android 开发手册[EB/OL]. http://developer.android.com/

[12]杨玥，於志文，周兴社等. I-Sensing:面向智慧校园的参与感知应用系统 [J]. 小型微型计算机系统，2013,34(9):2205-2210.

[13]於志文，於志勇，周兴社等. 社会感知计算：概念、问题及其研究进展 [J]. 计算机学报, 2012,35(1):16-26.

[14]张鑫. 手机应用软件测试方法概述 [J]. 计算机光盘软件与应用, 2013,12:280-282

[15]Xiang Sheng, Jian Tang and Xuejie Xiao. Sensing as a Service: Challenges, Solutions and Future Directions[J]. IEEE SENSORS JOURNAL, 2013,13(10):3733-3741.

[16]N. D. Lane. A survey of mobile phone sensing[J]. IEEE Commun, 2010,48(9):140-150.

[17] K. K. Rachuri, C. Mascolo, M. Musolesi, and P. J. Rentfrow. SociableSense: Exploring the trade-offs of adaptive sampling and computation offloading for social sensing[J]. Mobile Comput. Netw. 2011,17:73-84.

[18]D. Kirkpatrick. The facebook effect: the inside story of the company that is connecting the world[M]. Simon & Schuster, 2011.

[19]Burke J A, Estrin D, Hansen M, et al. Participatory sensing[M]. World Sensor Web Workshop, Citeseer,2006:1-5.

[20]Lane N D, Miluzzo E, Lu H, et al. A survey of mobile phone sensing[J].

Communications Magazine, IEEE, 2010, 48(9): 140-150.

[21]Lee J S, Hoh B. Dynamic pricing incentive for participatory sensing[J]. Pervasive and Mobile Computing, 2010, 6(6): 693-708.

致 谢

研究生两年半的学业生活即将结束，在北京邮电大学网络技术研究院宽带网研究中心，我度过了一段充实精彩的生活。繁忙的学习生活，丰富的实验室项目实践，使我的学习和工作能力得到了充分的锻炼，分析问题和解决问题的能力得到了极大的提升。

感谢我的导师王文东教授，在这两年半的学习和工作生活正，他给予了我们很多指导和关怀，在繁忙期间每周抽出时间与我们探讨问题，让我在研究生期间有明确的奋斗目标，能力也得到很大的提升。在此想王老师表示衷心的谢意。

同时也感谢龚向阳教授和阙喜戎副教授，在科研实践当中，他们给予了我们很多的支持和帮助，对我们的工作内容和进展提供了宝贵的意见。在此感谢他们。

感谢同项目组的朱致远、刘肖阳、赵露名、袁龙运、李莹、徐登佳、周萌和周雪松。在研究生期间，我们共同研究和讨论问题，甚至加班加点的工作。没有大家团结一心和辛勤的劳动，就不可能有今天的成果。

感谢我的家人，在我工作最忙最辛苦的时候，给我强大的精神支持。正是他们对我的支持和理解，使我坚定了信息，努力工作。

最后，感谢各位评阅老师在百忙之中抽出宝贵时间评阅！