密级：　　保密期限：

北京郵電大學

硕士学位论文

题目：基于参与式感知平台的高并发数据处理系统的设计与实现

学　　号：2012111366

姓　　名：徐登佳

专　　业：计算机科学与技术

导　　师：龚向阳

学　　院：网络技术研究院

2015 年　2 月　4　日

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。（保密的学位论文在解密后遵守此规定）

保密论文注释：本学位论文属于保密在__年解密后适用本授权书。非保密论文注释：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：_____

基于参与式感知平台的高并发数据处理系统的设计与实现

摘　要

近年来，随着通信、微电子、集成电路和软件等技术的飞速发展，移动电话已经从普通的通信工具转变为兼具计算、感知、通信能力的智能终端。参与式感知的基本思想就是由人携带手机等移动终端设备收集并共享感知数据信息，并基于这些信息提供各种新型服务。参与式感知区别于传统传感网络，更加注重“人”的参与，使感知系统具有了覆盖面广、数据丰富、不需要额外硬件投资等优点，使得低成本大范围收集感知数据成为可能。使用移动终端的传感器进行数据收集，其过程会产生大量的原始数据，服务器需要对原始数据进行接收和存储，提供给后续的数据融合、数据分析和数据可视化部分进行处理。因此，在参与式移动感知平台中，服务器对于数据的稳定接收和高效存储具有重要的意义，是其他模块功能的基础。

本论文以参与式感知平台为基础，设计并实现了一个参与式感知平台的服务器端并发数据处理模块。本模块作为平台客户端和服务器的通信接口，实现对大量客户端的并发数据的稳定接收和高效存储，为客户端模块提供下载和查询接口，并为激励模块提供任务推送接口，将客户端和服务器的采集、上传、分析、查看功能结合到了一起。

本文首先介绍了参与式感知的研究背景和发展现状，然后对参与式感知平台进行了简单介绍，接下来对参与式感知平台的并发数据处理系统，进行了详细的设计和实现；然后，对系统的功能和性能进行测试。论文最后对全文进行了总结，对未来的工作进行了展望，并总结了作者在研究生期间的所有工作和成果。

关键词　参与式感知　移动协作　并发　消息推送

DESIGH AND IMPLEMENTATION OF CONCURRENT DATA PROCESSING SYSTEM BASED ON PARTICIPATORY SENSING

ABSTRACT

With the improvement of communication technology and related hardware level, an ordinary mobile phone is gradually transforming from a communication tool to an intelligent terminal equipped with large numbers of advanced built-in sensors with rich computing, sensing and communication capabilities. In this context, a new data collection method, participatory sensing was put out. Different from the traditional sensor network generating data by the sensors, participatory sensing pay more attention to the "people" . It uses conscious or unconscious users' participation, so that perceptual system has a wide coverage, data-rich, no additional hardware investment. In this way, a wide range of low-cost sensing data collection system is possible.

During the process of sensor data collection by the mobile terminals, it will produce large amounts of raw data, which needs to be received and stored by the server. The data would be provided for the subsequent data integration, data analysis and data visualization modules. In the participatory sensing platform, a server capable of receiving and storing data efficiently is significant, and is the basis of other modules.

In this thesis, a participatory sensing platform which is able to collect, upload, analysis and view data is designed and implemented. In the client, user can collect real-time data and upload it, while the server receive the data and analyze the data. As the interface between client and server, in order to achieve a stable and efficient concurrent data processing, a concurrent data processing system on the server side is designed.

First, this thesis introduces the related theories and technical background of participatory sensing and incentive strategy. Second, this thesis presents the quirement analysis and system design of the participatory sensing platform. Third, the detailed design and implementation of each subsystem are highlighted.

Then unit and integration testing are adopted, which indicates that the entire system achieves the prospective design goals. In the end, the thesis summarizes the whole work of the system, looking forward to future work, and summarizes the work and achievements of the author during the post-graduate period.

KEY WORDS participatory sensing mobile cooperation concurrency push

目 录

第一章 绪论

1.1 研究背景

随着通信技术的发展与相关硬件水平的提高，移动电话从普通的通信工具逐渐转变为内置大量先进传感器的智能终端，具备了丰富的计算、感知和通信能力。在此背景下，一种新的数据收集方式——参与式感知（Participatory Sensing）被提了出来[1]。区别于传统传感网络由传感器产生、搜集、上传信息的过程，参与式感知更加注重“人”的参与，数据由用户创建、筛选或者控制，然后上传。它利用有意识或无意识的用户参与，使感知系统具有了覆盖面广、数据丰富、不需要额外硬件投资等优点，使得低成本大范围收集感知数据成为可能。

移动智能终端集成了多种传感器设备，并可以采集相应的感知数据，移动智能终端的携带者可以主观决定是否将感知数据共享。另外，人类的社会活动使得移动终端经常能与其他终端相遇，具有相互协作完成感知任务的机会。与固定部署的传感网络技术相比，参与式感知技术以用户信息分享为基础，用户不仅是参与式感知系统的使用者，同时也是内容和服务的提供者，更是最终的受益者[2]。它具有以下优点：1）部署成本极低。参与式感知技术使用移动终端的感知及无线通信功能收集、上传数据，同时，参与式感知的服务开发部署难度低，便于大规模部署；2）强大。手机有多少种传感器，人们就可以获取多少种数据，就可以上传多少种数据，这使参与式感知网络能够搜集丰富的感知数据，而移动终端用户的固有流动性决定了参与式感知史无前例的时空覆盖范围；3）均衡。感知的主体是人，人具有高度灵活性和移动性，人们可以通过协作的方式动态分配任务，可随时根据要求调整上传内容，使得数据获取更加均衡。

尽管参与式感知技术有着广阔的应用前景和诸多优点，但在实际部署和运行的过程中依然面临着巨大的挑战与不足[3]。首先，作为节点的移动终端设备具有运动上的随机性，节点数量分布的不均匀性，节点拥有资源不同产生的异构性，这些都导致节点提供的参与式感知服务具有差异性和动态不确定性；其次，移动终端设备的剩余能量、计算能力、存储空间等资源相对有限，这对于数据收集过程产生了比较大的限制；再次，数据收集过程受参与者的影响，由于参与者不同的文化、教育背景等，收集到的感知数据在有效性和可信性方面都需要进行鉴别和处理[4]。

参与式感知系统一般采用集中式的控制方式，包含感知、网络、应用三个层面，移动终端收集的感知数据通过无线通信接口上传至中央服务器进行处理。感知任务能够在

终端上人为触发，服务器自动触发或者基于当前上下文环境触发。在中央服务器上，数据经过分析、处理后，能够以多种形式展示给用户。使用移动终端的传感器进行数据收集，其过程会产生大量的原始数据，在参与式感知过程中，各个移动终端分别将所有原始数据上传至中央服务器。服务器需要对原始数据进行接收和存储，提供给后续的数据融合、数据分析和数据可视化部分进行处理。在参与式移动感知平台中，由于数据的异构性和数据访问的并发性，服务器对于数据的稳定接收和高效存储具有更为重要的意义，是服务器端其他功能的基础。

在移动互联网高速发展的大前提下，参与式感知技术会获得更多的应用和更好的发展。每个人都是一台数据传感器和环境检测器，所上传的数据将不仅用于自身，而且可以建立起有关全体人类行为和周边环境的海量数据库，系统的中央服务器必须能够进行高并发和高稳定的数据处理，参与式感知才能在科学研究、灾害救援、以及商业应用等多个方面有更大的作用。

1.2 课题主要研究内容

本课题以移动参与式感知平台为基础。该平台实现感知数据从采集、上传、存储到处理和展示的过程，同时实现激励任务的触发和下发过程。本论文主要在参与式感知平台的基础上，对平台的服务器端并发数据处理模块进行设计和实现。因此本论文的主要研究内容包括：参与式感知平台的整体需求和设计，服务器端数据处理模块的设计和实现，对服务器端数据处理模块（即网络接口模块）的测试工作。

参与式感知平台以 C/S(Client/Server)结构为基础，由移动终端和服务器、网站构成。移动终端用户设备及其所嵌入的多种传感器作为基本感知单元，收集感知数据。数据可以在客户端进行本地存储和展示，在网络状态良好且电池电量充足的情况下，客户端会将数据传送到服务器端。服务器接收各种传感器数据和客户端拍摄的照片数据，进行存储和分析。网站具有数据展示的功能，能将收集的数据在网页上展示，并且通过使用谷歌地图提供的接口，能将数据与其经纬度信息作关联性的展示。本平台在服务器端预留了接口，数据分析模块可以使用该接口在系统中添加不同的分析算法，对于手机的数据进行分析和利用，因此系统具备可拓展性。

服务器端数据处理模块作为服务器和客户端之间的通信接口，为了保证平台的稳定工作，需要提供稳定的数据处理能力。数据处理能力包括：1）数据接收功能。该模块作为服务器与客户端的通信接口，需要处理参与式感知网络中的并发数据上传请求，保证客户端数据的上传能够稳定进行；2）数据存储。接收到了大量图片和感知数据后，模块提供稳定高效的数据存储功能；3）数据查询功能。为客户端提供查询数据库的接口；4）推送功能。为了提高协作感知的效率，平台引入了激励机制，因此需要为其提供消息推送的接口，使激励能够推送到客户端。

基于参与式感知平台的特性，本文侧重于对上述服务器端数据处理模块的功能实现进行研究。包括并发数据接收能力的实现和优化、数据存储的实现和优化、任务推送功能的实现。最后，为了测试服务器端数据处理模块和参与式感知平台的功能，设计相应的测试用例，对模块功能和系统整体功能进行验证，并对系统的性能进行了测试。

1.3 主要工作

本文主要工作是调研现有的参与式感知平台的实现方案和技术，整理项目需求，设计一个基于移动协作的参与式感知平台，然后对其中的服务器端并发数据处理系统进行了概要设计和详细设计，并对其进行功能实现和测试。

具体主要工作如下：

技术调研。充分了解参与式感知的研究背景、参与式感知平台的设计思路，调研并发数据处理和数据推送的相关技术。

参与式感知系统和高并发数据处理子系统的需求分析。充分分析参与式感知平台的功能需求，并根据功能设计详细的应用场景，在此基础上，完成对参与式感知系统的并发数据处理模块的用例描述。

参与式感知平台的服务器端数据处理模块的设计。根据模块的功能需求和应用场景，对模块进行设计。包括模块与其他模块的数据接口和调用关系，以及模块内部子模块的划分和子模块的实现流程等。

服务器端数据处理模块的实现和测试。根据设计方案对参与式感知平台的服务器端数据处理模块进行了实现。为了验证数据处理模块的功能和处理性能，设计了测试用例进行测试。

1.4 论文结构

本文的主要内容是参与式感知平台的服务器端并发数据处理系统的设计与实现。论文章节内容安排如下：

第一章，绪论。介绍了参与式感知的研究背景和参与式感知平台的基本架构，并介绍了论文作者的相关工作。

第二章，相关技术和概念。介绍参与式感知的技术背景以及发展现状，对于本系统中需要用到的相关技术，如并发数据处理和推送技术等进行介绍。

第三章，系统的需求分析及总体设计。首先说明系统功能需求，通过用例图说明的方式对参与式感知平台进行需求分析。然后分析了服务器端的并发数据处理模块的需求，通过数据流图分析了模块的逻辑功能和数据的逻辑变换过程。然后介绍本模块与其他模块的数据接口和调用关系，以及消息流程等内容。

第四章，网络接口模块的设计与实现。详细介绍了参与式感知平台的服务器端并发

数据处理系统（即网络接口模块）的内部结构，包括子模块划分以及各个子模块的类设计、数据库设计和内部流程。

第五章，网络接口模块的测试。介绍服务器端并发数据处理系统的测试环境，功能测试和性能测试的测试用例和测试结果及对测试结果的相关分析。

第六章，结束语。对已完成工作进行总结，并指出需要完善的地方和相应建议。

第二章 相关技术和概念

参与式感知是指利用众多移动通信设备的内置传感器部件来进行数据的感知、收集、分析及反馈应用的新型数据服务模式。在参与式感知过程中，各个移动终端分别使用WLAN 网络将所有原始数据上传至服务器后台。服务器需要对原始数据进行接收和存储，提供给后续的数据融合、数据分析和数据可视化部分进行处理。在参与式移动感知平台中，由于数据的异构性和数据访问的并发性，服务器能够提供的数据的稳定接收和高效存储具有更为重要的意义，是服务器端其他功能的基础。

本章首先对参与式感知的产生背景和研究现状做一个概述和总结，然后简要介绍了实现参与式感知平台的高并发数据处理系统的相关技术。

2.1 参与式感知和城市计算

2.1.1 参与式感知

随着移动通信技术的不断演进和发展，当前的用户设备已经从只具备单一的语音通话功能扩展到集各种复杂的多媒体应用于一身。在此前提下，为了更好地完成感知任务，研究人员提出了一种通过不同个体共同完成感知任务的感知数据收集方式——移动协作感知（Mobile Participatory Sensing）[5]。移动协作感知将智能终端用户设备作为基本节点，保证了信息获取的多元化与及时性、实现了感知任务（来源于任务需求方）与感知动作（来源于任务参与者）的充分互动、感知数据的有效收集和分析，进而形成具有明显交互特性的“参与式感知平台”[6]。借助于移动通信技术所带来的优势和它们的广泛应用，参与式感知可以利用用户设备配备的各种类型的传感器来实现大范围的感知数据采集。

参与式感知最先称为城市感知(Urban Sensing)，此命名主要是因为它的应用领域和参与者的地域性。顾名思义，参与式感知的重点在于人，即参与者，这也是参与式感知区别于传统传感器网络的特点。在参与式感知网络中，每一个参与者都是一个传感节点，感知周围的环境并上传数据。人的分析判断能力大大减轻了系统的负担，同时人的移动性扩大了系统的地域覆盖范围，因此系统可以获得实时的，覆盖范围广的，海量的数据。

当前参与式感知网络已经有了一定的应用和研究，参与式感知系统也有多种多样的组织结构，概括起来可以将参与式感知的整体架构简略地用层级结构表示。如图 2-1 所示，参与式感知网络基本可分为感知层、网络传输层、数据处理层和应用层。

感知层：即数据采集层。利用移动设备和设备上内嵌的各种传感器作为客户端节点，实现对声音、光照、噪声、速度、经纬度等多种传感器信息的采集。

网络传输层：采集到的数据必须传输到服务器，才能为后续的数据处理层、应用层功能提供数据基础，因此网络传输层的功能十分重要。网络传输层将感知层收集的数据上传到服务器，提供高效稳定的网络传输功能。

数据处理层：数据处理即对收集的原始数据进行数据存储、数据融合、数据清理等一系列的原始数据加工。主要以数据库中的感知数据为基础，对其进行原始数据的存储和更新、融合和清理、分析和计算等操作。

应用层：数据处理层通过对数据的分析获得相应的信息，应用层通过具体的服务对这些信息进行应用或展示，使采集到的感知数据真正能对参与式感知中的参与者的日常活动和生活环境进行有效的反馈和指导。

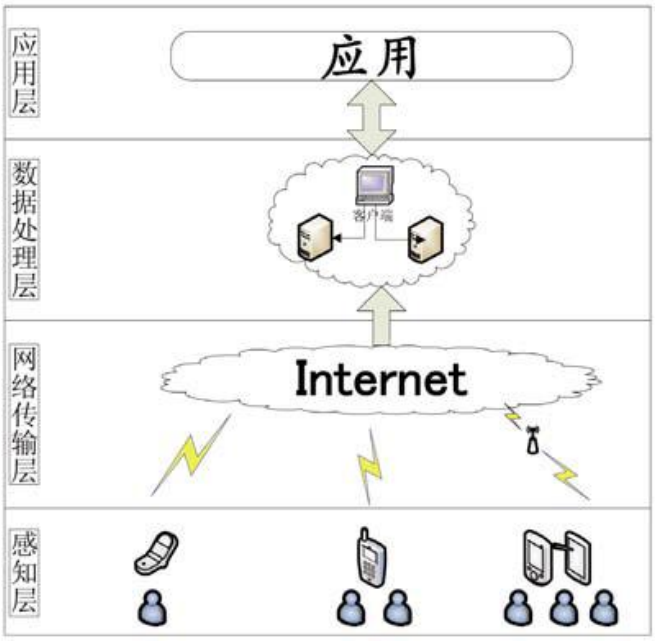


图 2-1 参与式感知框架图[7]

2.1.2 城市计算

参与式感知（Participatory Sensing）是目前移动物联网领域的研究热点之一，近年来，已经涌现了大量创新性的参与式感知应用。其中，城市计算（Urban Computing）是近几年比较比较热门的应用领域。

城市计算[8]通过感知技术获取城市中的多种异构大数据，通过多种高效的数据管理和分析算法对获取的数据进行数据整合分析，使用数据可视化技术将数据收集分析的结果进行展示。城市计算致力于解决当前城市面临的环境、交通、能源等多种挑战，提高人们的生活品质、保护环境和促进城市运转效率。

图 2-2 所示为城市计算的基本框架图。城市计算是一个“多数据多任务”的系统。

从图中可以看出，它分为四个环节，其中的城市感知和数据获取环节需要从城市的环境规划、交通拥堵、能源消耗等各个方面获取多种异构数据，而这些数据正是城市计算的基础。传统的传感器网络覆盖面小、数据类型不丰富、且需要额外硬件投资，必然不能满足需求，而参与式感知、群智感知、移动协作感知等低成本的感知计算方式才是获取数据的主要途径。

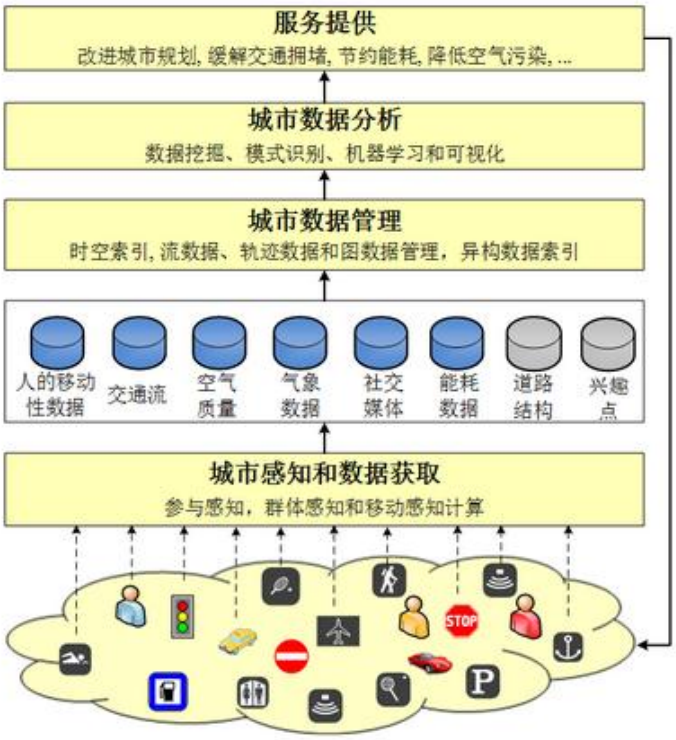


图 2-2 城市计算基本框架[9]

2.1.3 参与式感知中的问题

作为一个新兴的领域，在带来机遇的同时，参与式感知还有很多问题需要研究，主要有以下几个方面：

用户资源优化问题。用户在收集和上传数据过程中，需要耗费客户端资源，如果不考虑数据收集与传输过程中的能量效率而大量上传冗余数据，会造成用户的资源浪费，极大的损害参与者的积极性。同时，服务器对数据进行存储和分析也需要耗费服务器资源，如果用户上传的数据并非服务器所需要的关键数据，必将对服务器资源造成不必要的浪费；

反馈与激励的问题。参与式感知系统中用户的积极参与是系统能够持续运行的关键。如果用户丧失了参与的动力，那么参与式感知系统将无法运行下去。因此，对于参与式感知中的参与者，要对其参与行为进行合理的反馈和鼓励，使其从自身上传感知数据的行为中获益，也就是对用户进行激励，保证用户的参与度。如何通过激励机制最大限度

的调动参与者的积极性，是一个需要深入研究的问题；

隐私保护。参与式感知系统的固有运行机制决定了参与式感知系统中的隐私保护与参与激励存在着矛盾,即为了保护用户隐私,一般会将用户的身份隐蔽,而在激励机制中给用户回报却需要明确相关数据的收集者。参与者个人隐私的保护，关系到参与者的个人信息安全，而个人信息安全又会影响参与者的积极性，因此必须研究相应的措施保障用户个人隐私不被泄露；

数据合理有效的处理和利用。参与式感知的一大弊端就是数据可靠性问题。收集到的感知信息很大程度上受到用户主观因素的影响，导致这些信息的可靠度并不高，同时节点的不确定性和网络情况的复杂性也增加了数据的不可靠性，这就要求后台处理程序有能力鉴别、剔除不可靠的信息。另外，单个参与者作为基本感知单元，其收集的感知数据的价值可能是比较低的，但当大量的平凡数据汇聚到一起，就可以对其进行数据挖掘,从其中获得一些被隐藏的有价值的信息[8]。对于感知数据的挖掘和相关信息的利用，使参与式感知可以对人们的生产生活进行指导。

2.2 服务器性能优化

本课题中，服务器端不仅能接收到如环境 PM2.5 值等简单的文本数据，而且要能接受处理图片数据，在进一步的研究中，可能还需要处理视频音频数据。另外，服务器的并发请求量也可能随着客户端的增多而大量增加。服务器的性能如响应时间、吞吐量会影响客户端用户的使用体验，进而对参与者的积极性产生重要的影响[11]。感知数据的复杂性，也使系统对服务器的实时性和可靠性和服务器的整体性能提出了更高的要求。因此，必须对服务器的性能进行测试，分析其实际的运行情况，进行进一步的优化。

对于网络接口模块的性能测试，是为了检测网络接口模块是否能在客户端并发上传数据时进行稳定、高效的接收处理[12]。性能测试主要针对服务器的响应时间、并发度、吞吐量等参数进行测试。要测试的内容包括：连接速度测试、负载测试、压力测试等[13]。其中，连接速度测试是对系统的响应时间进行测试，即用户在客户端发出上传数据或者查询数据的请求，到接收到服务器返回的处理结果所需的时间。响应时间不仅是服务器性能的重要标志，也影响用户体验和用户参与式感知的积极性。负载测试是为了测试网络接口模块在客户端有并发请求时的工作状况，确保在本系统设计的并发请求下服务器能对数据进行有效的接收和处理。压力测试是指测试系统的并发度限制，以及系统的错误处理和恢复功能，保证系统出现错误时能够及时记录和解决。

服务器的性能优化方案，已经有非常多的方法可以采用，例如线程池技术、缓存技术等。线程池技术，即先在服务器的主线程中预先创建含有限个任务线程的线程池，一旦有客户请求到来就启动这些任务线程来处理客户端到来的连接。超出任务线程处理能力部分的客户就先在任务队列中等待，等任务线程处理完任务后成为空闲的任务线程，

空闲出来的任务线程就可以转而执行等待队列中的任务。缓存技术，是指将需要动态生成的内容暂时缓存在内存里，在一个可以接受的延迟时间范围内，同样的请求不再动态生成，而是直接从内存中读取，这样可以极大地减少了动态内容读取时间。

综上所述，对于服务器的性能优化有一些基本的解决方案[14]：改善服务器架构和设计，改进数据库的设计，改进 Web 容器的性能等。所以优化方案将从服务器性能优化、代码性能优化方面入手，包括 Tomcat Web 服务器的优化配置、图片保存和压缩的效率提高等方面，主要结合参与式感知中特殊的场景和数据特点，对服务器的并发性能进行优化。

2.3 系统开发技术

2.3.1 系统中的通信技术

2.3.1.1 HTTP 和 HTTPS

HTTP[15](Hypertext transfer protocol)，即超文本传送协议，是互联网上应用最广的协议。它定义了浏览器怎样向服务器请求资源，以及服务器怎样把浏览器请求的资源返回。HTTP 是应用层的协议，是互联网上网络节点之间进行可靠文件交换的重要基础。

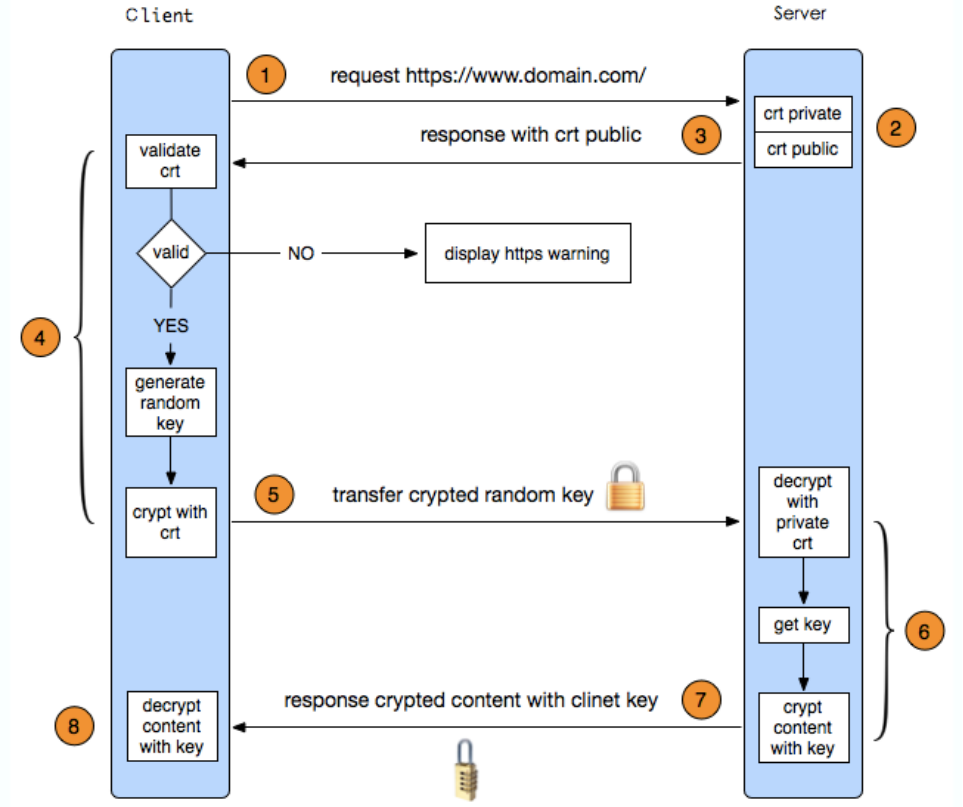


图 2-3 HTTPS 通信过程[16]

HTTPS(Hyper Text Transfer Protocol over Secure Socket Layer)，是使用 SSL（Secure

Sockets Layer）协议对 HTTP 协议传输的隐私数据进行加密的一种协议，可以说是安全版的 HTTP 协议。其通信过程如图 2-3 所示，首先客户端发起 HTTPS 请求；然后服务器接收到请求，进行配置并将自己的证书传送给客户端；客户端解析证书，然后就可以和服务器之间进行信息的传输了，即一方发送加密信息，另一方接收到信息之后进行解密，如上图步骤 5-步骤 8 所示。

在参与式感知中，对于用户隐私的保护是重要的研究方向，隐私保护也会极大的影响用户参与的热情。HTTP 协议用明文方式发送数据，因此不适合传送一些敏感信息；而 HTTPS 协议使用了具有安全性的 SSL 加密，更加安全。但是，HTTPS 协议必须从 CA 申请证书，增加了成本；另外，HTTPS 的复杂性，导致增加了服务器资源消耗，使服务器响应变慢，并不适用于本项目。考虑到本项目并不涉及用户个人身份、信用卡等敏感信息，使用 HTTP 协议暂时可以满足系统需求。

HTTP 协议的主要有以下特点：支持 B/S 模式；客户向服务器请求服务时，只需传送请求方法（GET、HEAD、POST 等）和路径。协议头中的 Content-Type 字段可以对传输的数据内容和类型加以区分，使得通信更加灵活。HTTP 是无连接的，每次连接只处理一个请求，服务器处理完客户的请求，并收到客户的应答后，即断开连接。这种连接和传输数据的方式可以极大的节省传输时间。

由于它的上述优点，在本系统设计时服务器与客户端间的通信采用了 HTTP 协议进行通信。客户端需要向服务器请求资源或上传数据，则对服务器发起请求（get/post），服务器则向客户端返回对该请求的应答，返回的资源可以有很多表达方式（图片、文字、富媒体等）。

2.3.1.2 消息推送

消息推送是简单来说就是服务器向移动终端发送连接，进行消息的发送，使客户端不定时地被动获取服务器的信息[17]。传统的消息推送方式包括三种，第一种是 SMS 消息，通过短信的方式将服务器更新的信息发送给客户端，但是很明显其成本较高，在经济上并不适用于本项目；第二种是长轮询的方式，每隔一段时间就向服务器发起请求获取信息，查看服务器信息更新状况。该方法如果轮询频率较低，则实时性难以保证，而频率较高则会消耗系统网络资源和电能；第三种方式是服务器使用 Push（推送）的方式，当服务器端有新信息了，则把最新的信息主动 Push 到客户端上。这种方式使消息更新的实时性有了很大提高持久连接的推送方案，是实现移动端平台消息推送的最佳选择。

2.3.1.2.1 C2DM 技术

C2DM[18](Android Cloud to Device Messaging)是一个用来帮助开发者从服务器向 Android 应用程序发送数据的服务。如图 2-4 所示，客户端用户首先使用 Android 的客户端向 C2DM 服务器申请注册，服务器接收到注册请求生成标示客户端的注册 ID，然后

将客户端用户名与注册的 ID 在应用服务器进行绑定。应用服务器会向 C2DM 服务器发送信息，C2DM 服务器则将接收到的应用服务器消息推送给客户端，此过程在应用服务器上的信息或者客户端信息更新时发生。经过以上过程，服务器主动下发信息到了客户端。

图 2-4 C2DM 工作流程[19]

C2DM 是谷歌官方提供的消息推送机制，具有资源消耗小，稳定性强的优点。但是由于网络环境的限制，可能无法连接到谷歌官方的服务器，因此本论文无法选择该方案应用到工程实践中。

2.3.1.2.2 XMPP 协议

XMPP[20]（可扩展消息处理现场协议）是基于可扩展标记语言[21]（XML）的协议，它主要用于实现即时通信，通常所见的即时通信软件底层均采用该协议。这个协议允许互联网用户向互联网上的其他任何人发送即时消息，并且消息的传递不依赖于客户端的类型。XMPP 协议中定义了客户端，服务器，网关三个角色，客户端之间发送消息必须通过服务器，服务器同时承担了客户端信息记录，连接管理和信息的路由功能，会对连接到服务器的客户端进行角色验证。网关负责与异构的即时通信系统的互联互通。

2.3.1.2.3 MQTT 协议

MQTT[22]（消息队列遥测传输）是 IBM 开发的一个即时通讯协议，是一个轻量级的适用于传感器和移动设备的通信协议。该协议采用基于代理的“发布/订阅”模式，协议结构简洁、小巧，它可以通过很少的代码和带宽与远程设备连接，适用于要求能耗较低的设备。该协议包括客户端和代理服务器，客户端连接到代理服务器，然后告知服务器本客户端可以接受到的消息类型； 同时，客户端可以发布自己的不同类型的消息，这些消息根据协议的内容，可以被其他客户端获取。但是，该协议的缺点也比较明显，

其服务端组件 rsmb 不开源，而且部署较复杂，并不适用于本系统。

目前国内第三方提供的云推送服务，多是基于对 XMPP 协议的封装实现。考虑到 XMPP 应用的广泛性，综合调研评估各第三方推送服务的稳定性和时长反馈，本论文采用百度提供的云推送 API，并对其消息处理流程进行了改进，以满足本论文的功能性需求。另外，添加了客户端主动获取消息的机制，使激励任务下发流程更具灵活性。

2.3.1.3 进程间通信

多进程和多线程在编程实现和性能上很大的区别。多进程间共享数据复杂，需要远程调用来实现，但数据分开存储导致数据同步比较简单，多线程由于能够共享进程的数据，故数据共享简单，但对数据同步造成了极大地困难；多进程需要为每个进程开辟内存，内存消耗大，CPU 利用率低，多线程占用内存少，线程间切换简单因此 CPU 利用率高；进程的创建与销毁较为复杂，速度较慢，而进程创建销毁比较简单；在编程调试方面，进程具有显著优势，其编程调试难度远小于多线程编程；在可靠性方面，进程之间不会相互影响，而多线程会出现一个线程崩溃导致整个进程崩溃，系统无法运行的现象；多进程能够适用于多核多机的分布式系统，增强系统的可扩展性，但是多线程则无法支持多机的扩展。

综合各方面考虑，为维护系统的稳定性和可扩展性，本系统采用多进程的方式。进程之间需要实现数据的共享、计算结果的传递等功能，因此需要实现进程间通信。本论文选择了远程调用的方式实现进程间的通信。

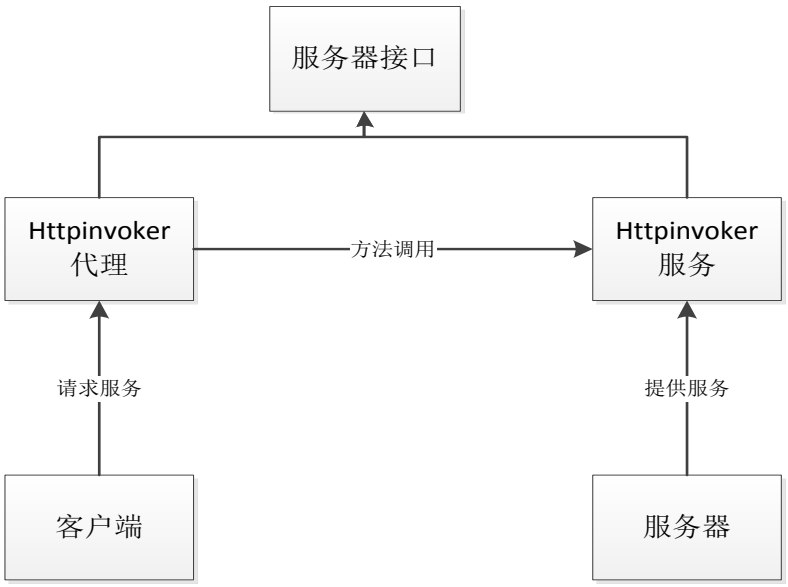


图 2-5 httpinvoker 基本结构

远程调用是指通过客户端进程调用远程服务端服务，等待应答。服务端的服务保持休眠状态直到调用信息到达为止，服务端在接收到调用信息之后，对服务进行执行，完

成服务的计算后返回计算结果。

在服务器设计初期，经各模块需求分析，服务器选择 java 作为开发语言。因此在选择远程调用技术时，在 Java 本身提供的远程通讯协议 RMI 和 Spring 框架提供的 HTTP invoker[23]之间进行了比较。RMI 具有较高的稳定性，应用十分简单，不依赖于框架，只依赖 Java 语言本身，但其不能穿透防火墙，在实际应用中造成了很大困扰。而 HTTP invoker 其效率与 RMI 协议相差无几，但其通过 http 协议访问 web 的端口的方式实现远程调用，能够良好的解决无法通过防火墙的问题。

HTTP invoker 是 spring 框架中的一个远程调用模型，执行基于 HTTP 的远程调用，并使用 java 的序列化机制在网络间传递对象。客户端可以很轻松的像调用本地对象一样调用远程服务器上的对象。其工作方式如图 2-5 所示。Spring 框架提供了一个代理工厂 Bean，使得我们能够将远程调用服务向本地 JavaBean 服务一样配置，极大简化了远程调用服务模型。

远程调用实现分为服务端和客户端两部分，服务端提供服务供客户端调用。采用 HTTP invoker 的方式，服务端将服务部署在网络上，客户端可以通过 http 协议以及 Http Invoker 代理对网络上的服务进行调用。

- 服务提供方配置：

服务的配置需要导出 HTTP invoker 服务，为导出该服务，需要使用 HttpInvokerServiceExporter 类，因此需要先在 Spring 中配置一个 HttpInvokerServiceExporter Bean。通过配置 service-ref 的属性值置入了实现这个服务的 Bean 的引用，可以指明所实现的服务的位置。serviceInterface 属性用于标示这个服务实现的接口，在该接口中标识的服务即为服务器提供的支持远程调用的服务，即需要在接口中写明提供的服务的服务名、参数列表、返回值。

HttpInvokerServiceExporter 的本质是一个 Spring 的 MVC 控制器，它通过 DispatcherServlet 接收来自客户端的请求，并将这些请求转换成对实现服务的 POJO（Plain Ordinary Java Object，简单 Java 对象）的方法的调用，流程如图。

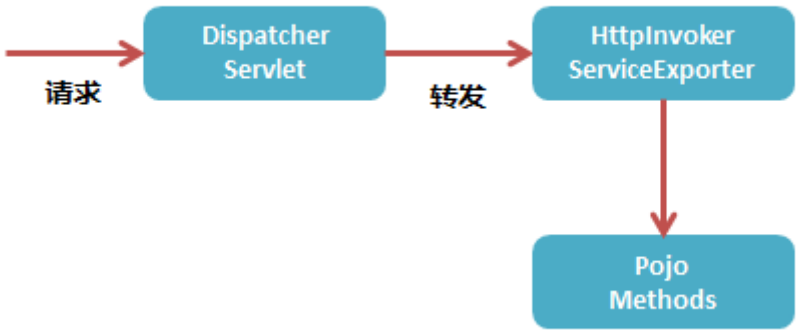


图 2-6 请求接收流程

因为 HttpInvokerServiceExporter 是一个 Spring 的 MVC 控制器，我们首先需要在部署描述符文件（web.xml）中声明 DispatcherServlet，它是一个前段控制器，其任务是将请求的 URL 通过处理器映射来决策其下一步的工作内容，之后配置<servlet-mapping>，表明所提供的服务对应的访问路径。最后，我们在 Spring 的上下文文件中建立一个 URL 的处理器映射，映射 HTTP URL 到对应的服务上面。通过映射配置定义了服务访问的路径。

- 客户端配置：

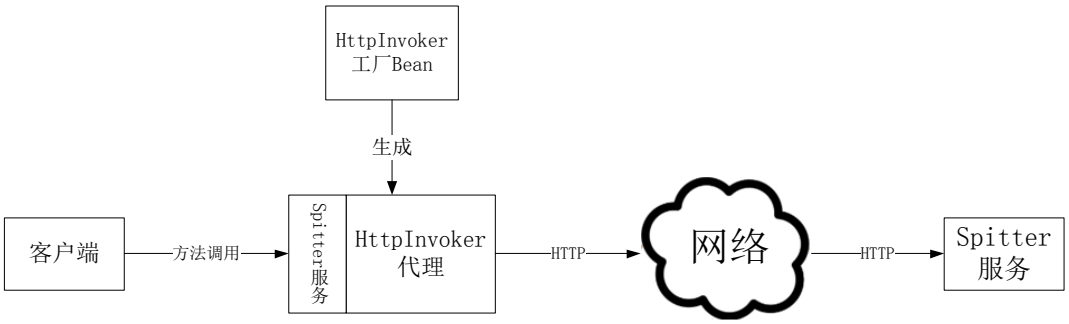


图 2-7 客户端远程调用流程

为了访问基于 HTTP Invoker 的远程服务，客户端必须在 Spring 的应用上下文中配置一个 HttpInvokerProxyFactoryBean 的 Bean 来代理它。HttpInvokerProxyFactoryBean 是一个代理工厂 Bean，用于生成一个代理，该代理使用 Spring 特有的基于 HTTP 协议进行远程通信。

其中，serviceInterface 属性用于标示服务所实现的接口，该接口标示了所能提供的服务。serviceUrl 属性用于标示远程服务的位置。完成上述两个属性的配置后，就能够像调用编写在本地函数一样访问远程服务了。

2.3.1.4 通信数据格式

JSON[24]（JavaScript Object Notation）是一种轻量级的数据交换格式。由于 JSON 是基于 JavaScript 的一个子集，这意味着在 JavaScript 中处理 JSON 数据不需要加载任何特殊的工具包。JSON 采用完全独立于语言的文本格式，但是也使用了类似于 C 语言家族的习惯（包括 C, C++, C#, Java, JavaScript, Perl, Python 等）。这些特性使 JSON 成为理想的数据交换语言。易于人阅读和编写，同时也易于机器解析和生成。相比 XML，JSON 容易阅读、解析速度更快、占用空间更少。

本系统在设计服务器与客户端之间通信接口，以及系统各模块之间接口时，使用 JSON 格式的 API，相比传统的 XML 等格式更加简洁，减少了单次数据传输字节流的大小，减少了数据交换时间，提升了用户体验，同时也便于扩展。

2.3.2 系统开发运行环境

系统的开发环境如下：

- 后台编程语言：Java（JDK 1.7）
- 集成开发环境：Eclipse Java EE
- 服务器端框架：Spring MVC（HTTP invoker）
- 测试工具：Chrome 浏览器 postman 插件，Apache Bench 网站性能工具
- 版本控制：SVN
- 客户端编程语言：Java、android SDK

服务器运行在 Linux 操作系统上，便于进行编程和调试。在保证系统的安全性的同时也提升系统的性能、增强可扩展性。Web 前端部署在了和服务器端同样的环境中。

2.4 本章小结

本章主要介绍了对于项目实现方案的调研和方案的选取，以及服务器性能优化、消息推送和进程间通信的基本知识。着重介绍了参与式感知的概念以及在城市计算中的应用，说明了参与式感知研究中的重点和难点。对于本课题的重点数据并发处理需要用到的几种技术进行了详细的介绍，并且对系统开发的技术方案进行了初步的研究。

第三章 系统的需求分析及总体设计

通过对参与式感知和相关技术的研究，我们对参与式感知平台有了更深入的理解。本章通过对参与式感知平台进行需求分析，明确参与式感知平台的整体功能需求和应用场景。在此基础上，对于平台的服务器端的并发数据处理系统，即网络接口模块进行了分析和总体设计，明确该部分在整个平台中的职责和作用，为详细设计和实现奠定基础。

3.1 系统的需求分析

本课题需要实现一个完整的参与式感知平台系统。在系统中，客户端作为移动的感知节点，通过收集自身的传感器数据和照片数据完成感知数据采集；采集到的感知数据被上传到服务器，由服务器上对应的算法模块对数据进行分析和利用，实现数据融合、轨迹预测、PM2.5 分析等功能。客户端用户不仅能从客户端观察到自己上传的相关数据，还能了解周围的其他用户所采集的数据，同时用户能从自己上传的数据获得奖励。而为了更好的展示收集到的数据和数据分析结果，平台还提供了从网页查看数据的功能。

3.1.1 系统拓扑结构

根据上述的基本需求分析，整个系统的网络拓扑设计如图 3-1 所示。

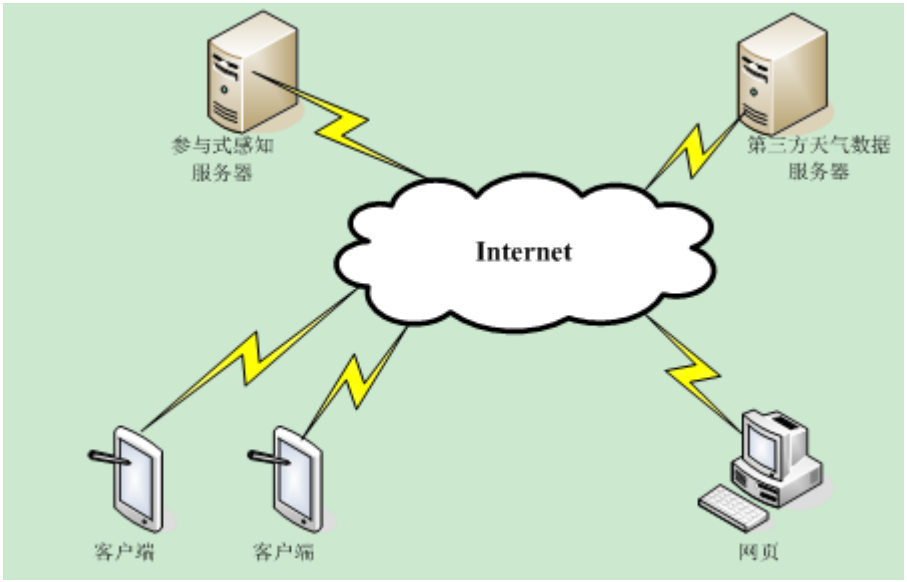


图 3-1 系统拓扑结构

该系统主要包括客户端、参与式感知服务器、第三方天气数据服务器和网页。

参与式感知客户端作为整个参与式感知平台系统的基本感知节点，可以收集自身的光照、声音、螺旋加速器等传感器数据，同时还可以使用客户端的拍照功能收集照片。收集到的照片和数据作为原始数据通过因特网传递到参与式感知服务器上。

参与式感知服务器进行数据的清洗、融合、分析等工作，产生的结果存入参与式感知服务器数据库。当客户端和网页需要查看这些处理结果时，需要向服务器提出查询请求，由服务器从数据库中读取查询结果返回。

在移动端需要通过拍摄照片计算 PM2.5 数据时，在建模过程中需要使用第三方天气数据进行验证。因此为保证建模的顺利进行，需要定时抓取第三方天气数据服务器，并进行存储，以确保在客户端需要数据时能够及时提供。

另外，参与式感知服务器上的激励模块能够根据已经收集到的数据的分布情况和参与者的运动轨迹，产生激励任务，下发到指定的客户端，促使客户端用户进行指定地点或类型的数据采集。在用户完成任务之后，还需要通过一定的算法计算用户应得的奖励。

3.1.2 系统整体需求分析

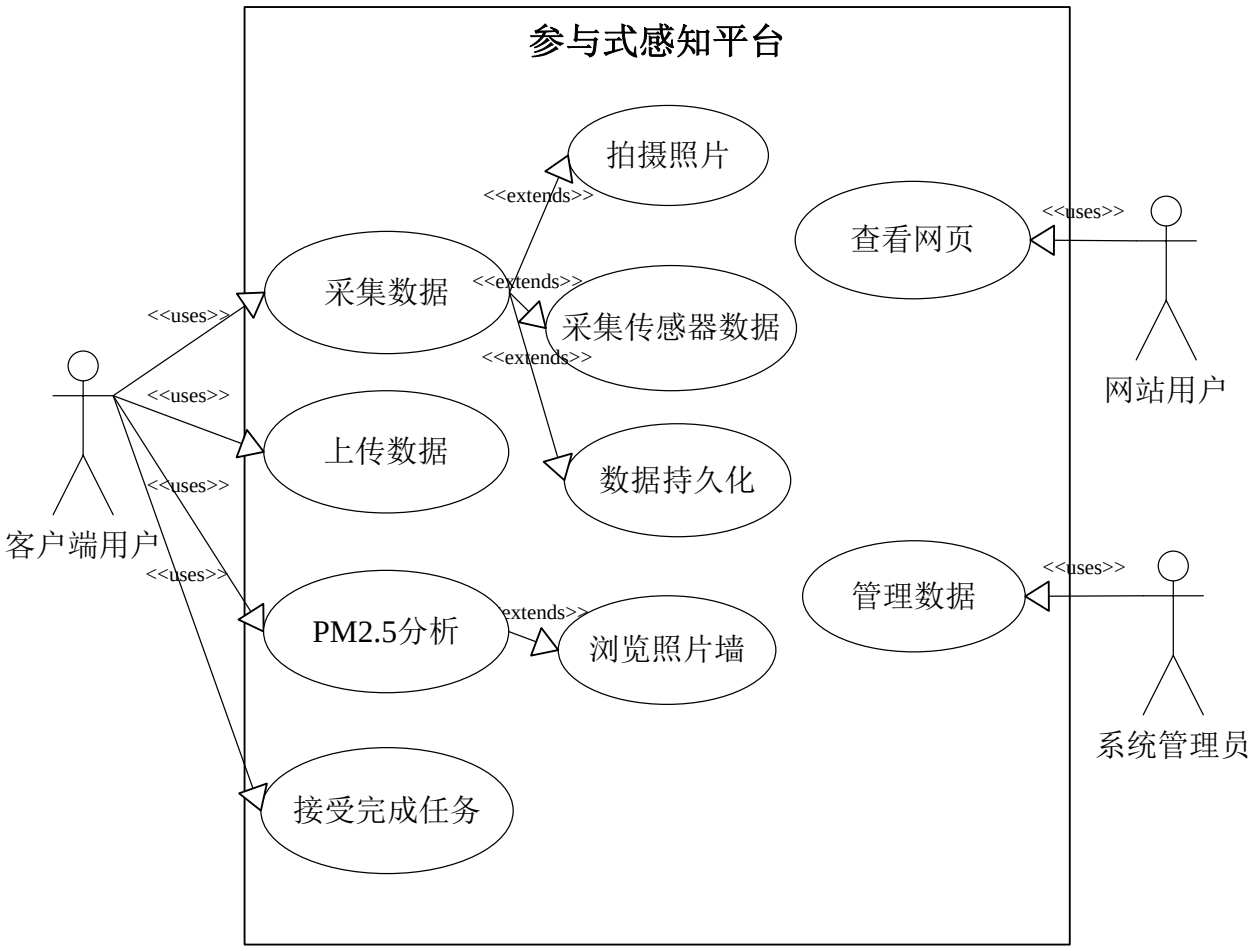


图 3-2 参与式感知平台系统用例图

本课题中的参与式感知平台的重点在于移动协作感知，系统的参与者包括客户端用

户、网站用户和系统管理员，系统需要为用户提供感知数据收集、数据上传、数据展示和激励任务接受完成等几大基本功能。下面将根据用例图，对系统功能进行说明。

网站用户主要使用参与式感知平台的网站查看网页，使系统收集到的各种感知数据能够全面直观的展现给用户。用户在网页上不仅能够按照感知数据的位置信息查看数据，还能够查看到一段时间内平台收集到的感知数据的变化情况，并且能观看到以 POI 的形式展示的照片数据的集合。

为了保证参与式感知平台的健壮性，应该提供系统管理的功能。系统管理员的主要职责是对平台中的数据进行管理，主要包括对数据的添加、查找、删除工作。平台收集到的感知数据有文本、照片等不同的存在形式，需要对他们进行高效的管理，以便更好地对数据进行利用。

客户端用户使用系统客户端，客户端作为参与式感知平台的基本感知节点，必须提供采集数据的功能，这是参与式感知的首要条件，也是平台的核心功能之一。这既包括采集噪声状况、光照情况等传感器信息，也包括拍摄照片；同时，为了防止采集到的数据丢失，平台还提供数据持久化功能，将采集到的未上传的数据进行保存。其次，对于客户端用户，平台能够提供稳定的数据上传功能，方便用户提交自己采集到的数据。此外，为了对于采集到的感知数据进行充分的利用，为客户端用户提供了根据拍摄的图片建模分析 PM2.5 的功能；作为对该功能的补充和完善，客户端用户还能在自己的客户端上观看到自己以及周围的其他用户收集到的图片。最后，作为激励机制在平台中的体现，平台为客户端用户提供任务的接受和完成功能。

- 采集数据

对于参与式感知来说数据采集功能是必不可少的，根据移动设备现有传感器的状况我们所采集的数据有：GPS、光照、声音、电池电量、充电状态、螺旋仪值、网络状态、图片信息等。将其按照采集方式的不同分为采集传感器数据和拍摄照片两种功能，并且在采集照片的时候，会将其他的传感器信息一同写入图片的 EXIF 文件当中。另外，由于受到网络状况的限制，采集到的数据可能无法及时上传，为了避免由于节点的不确定性造成的数据损失，应提供数据持久化功能，将数据保存在本地。

- 上传数据

数据在本地实现采集后，要上传到服务器进行存储和分析，因此需要提供数据上传的功能。不仅包括文本类型的感知数据，还需要上传图片或者其他多媒体数据。平台的数据上传具有突发性，因此平台应该具有处理并发上传的数据的能力。

- PM2.5 分析

为了良好的利用所采集的图片数据，我们在客户端对图片进行了建模分析。根据对同一场景的建模能够计算出，图片所对应的场景的 PM2.5 值。为了使用户更直观的参与到数据收集中，在客户端还能以照片墙的形式观看到客户端所建立的各个模型内包含的图片和当前环境的实景图片。

- 接受完成任务

为了验证激励机制在参与式感知平台中的作用，平台中设置了激励任务的产生和推送功能，能够根据激励机制产生任务促使客户端用户更积极的参与到数据收集中。作为客户端用户，能够使用系统客户端进行接收任务和完成任务。

3.1.3 网络接口模块需求分析

参与式感知平台的服务器端主要职责是对数据进行接收、存储和分析，提供 web 访问的接口。同时，服务器还提供接口使的采集的到的数据可以用来进行数据分析，以验证各种算法。

3.1.3.1 用例图

为了接受客户端上传的数据，服务器端需要提供并发数据处理功能，包括并发数据的上传和下载功能；同时为了进行激励任务的推送，服务器端要实现数据推送的功能。上述功能都是由服务器端的并发数据处理模块进行实现，该模块也称为网络接口模块。该部分的用例描述如下：

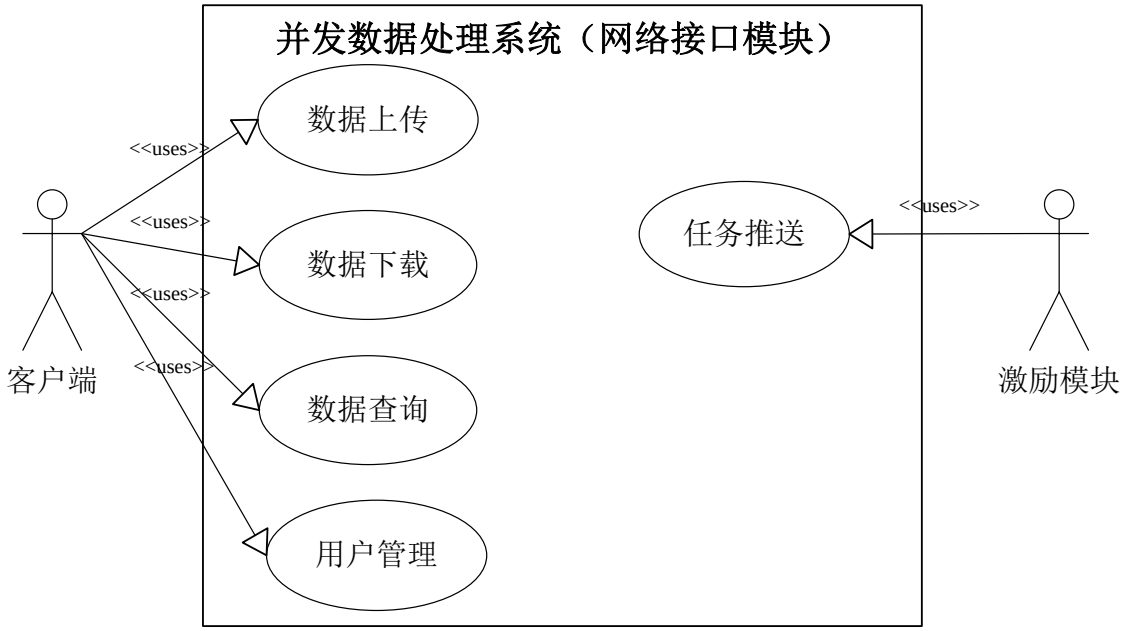


图 3-3 服务器网络接口模块用例图

- 数据上传

网络接口模块提供给客户端的主要功能，接收客户端上传的传感器信息和照片信息。数据上传功能的完整性确保了数据的完整性，是数据存储和应用的前提条件。由于客户端的数量是动态发生变化的，可能有大量客户端同时上服务器发出上传请求的情况，这就对数据上传功能的性能提出了要求。需要支持高并发的访问，并且完成对请求的迅速响应和数据的妥善存储。

- 数据下载

目前的参与式感知平台客户端为用户提供了查看照片墙的功能，从平台的可扩展性考虑，在客户端还可能部署其他的查看历史信息相关的功能。因此为了配合客户端的这些需求，需要网络接口模块提供数据下载功能，使客户端能从服务器下载感知数据和照片文件等。该功能同样需要支持高并发的访问，对下载请求进行迅速和稳定的响应。

- 数据查询

客户端需要对服务器上的数据进行查询，网络接口模块为客户端提供查询接口，接收客户端的查询请求，根据请求的内容进行相应的操作，向服务器数据库请求数据，将正确的查询结果返回给客户端。平台的 PM2.5 分析、照片墙等客户端功能模块都需要网络接口提供的数据查询功能的支持。实现该功能需要明确查询请求和应答的格式，内部的处理流程以及和数据库模块的交互过程。

- 用户管理

客户端需要进行用户注册和用户登录。用户管理负责对用户注册请求进行处理，对用户信息进行识别，将用户身份和用户权限、用户同意上传的数据类型等相关信息写入数据库中。在每一个用户登录时，进行登录验证，并更新用户的登录状态和相关历史记录。在进行任务推送时，进行用户查询，确定任务推送的目标客户端。

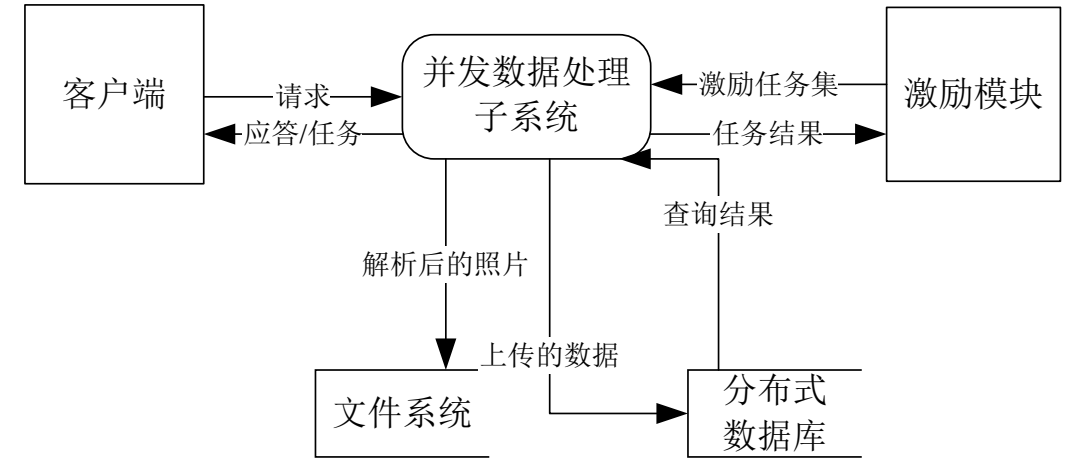
- 任务推送

为了验证激励机制，对用户进行有效的激励，需要发放任务。需要给激励模块提供消息推送的功能，使激励模块能够将任务推送给客户端用户。为了满足多种激励任务的不同需求，不仅提供推送给全体用户的功能，还能够指定推送的目标用户进行推送。

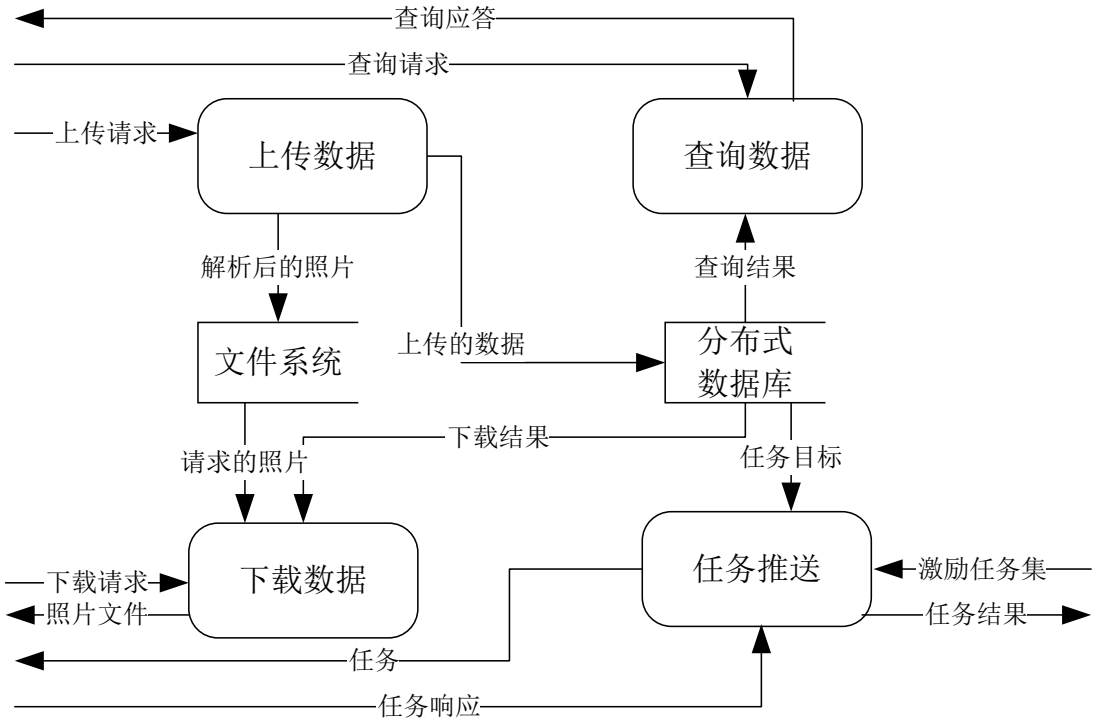
3.1.3.2 数据流图

对本模块的用例分析，明确了其基本的功能需求。在此基础上，需要对其逻辑功能，数据在模块内部的流向及数据处理变换的过程进行分析。图 3-4 为服务器端的并发数据处理子系统（即网络接口模块）的数据流图。

顶层数据流图中外部实体有客户端和激励模块，系统的输入数据流是客户端的各种请求和激励模块的激励任务集，只有这两类输入数据会使系统进行工作，对数据进行加工处理，产生的应答或任务也只会返回到对应实体。图 3-4(a)中，客户端向系统输入的请求包括上传请求、下载请求、查询请求，系统向客户端返回的应答也对应于每种请求；系统从分布式数据库中读取的查询结果也根据查询条件的不同包括下载文件查询结果、任务目标查询结果等。



(a) 顶层数据流图



(b)0 层数据流图

图 3-4 并发数据处理子系统数据流图

0 层数据流图将顶层的并发数据处理子系统进行了分解，从功能上将其分为四个加工：上传数据、查询数据、下载数据、任务推送。上传数据的输入数据流是客户端的上传请求，请求中的照片或数据经过解析和处理后，成为输出数据流，分别存入文件系统和分布式数据库。查询数据的输入数据流是客户端的查询请求，根据查询请求从分布式数据库中读取查询结果，产生输出数据流，即查询应答返回给客户端。下载数据的输入数据流是客户端的下载请求，根据下载请求从数据库读取到要下载的文件，再将照片文件从文件系统中读取出来，返回给客户端。任务推送的输入数据流是激励模块产生的激励任务集，任务推送接收到任务集之后，查询数据库获得任务目标，产生确定的任务推

送到目标客户端；客户端对接收到的任务会做出响应，产生的任务结果作为向激励模块的输出数据流。

3.1.4 非功能性需求

参与式感知平台不仅需要完成功能上的需求，还要在稳定性、可扩展性等方面有更高的要求。这些非功能性需求包括：对并发数据通信的支持，数据通信和存储的稳定性，平台的可扩展性，客户端的易用性等多个方面。

对于并发数据处理系统（即网络接口模块）来说，稳定高效的处理并发的数据请求才能满足整个参与式感知平台的功能性需求，因此，对于并发数据处理系统的性能要提出更高的要求。

- 性能

由于平台的客户端用户数量可能不断增多，因此作为接受处理客户端数据的模块，网络接口模块要能够处理突发大量数据同时上传的情况。支持高并发数据的上传和存储工作，能够尽快写入数据库，避免客户端等待时间过长造成请求超时或者带来较差的用户体验。同时，网络接口模块也需要支持客户端的查询和下载等功能，使客户端的查询请求和下载请求的响应时间尽可能短。

- 稳定性

系统数据库中存在大量数据，网络接口模块需要向数据库进行频繁的数据存储操作，因此读写数据库的过程要稳定，从而能够保证数据的完整性。为了系统的稳定运行，网络接口模块要最大限度的减少数据上传、下载失败的情况，减少系统崩溃的次数，并且应提供容错处理、系统日志等功能，以便在系统发生崩溃时进行问题追踪，以便能及时对问题进行恢复。

- 可扩展性

参与式感知平台的基本设计思想是一个开放性平台，提供接口完成对数据的开放，使在系统中添加新的算法模块时，原有的系统功能不受影响。网络接口模块作为客户端和服务器之间的通信接口，在系统设计时，要充分考虑与其他各模块间的耦合程度、以及提供给其他模块的接口的易用性和兼容性。

3.2 网络接口模块总体设计

本节首先介绍参与式感知平台的整体模块划分和逻辑结构，在此基础上描述了网络接口模块在参与式感知平台中的位置，说明了网络接口模块和其他模块之间的调用关系和消息交互。

3.2.1 系统模块划分

根据系统的功能性需求，将系统主要划分为客户端和服务器两大模块。客户端主要完成数据的采集、上传和 PM2.5 分析预测的功能；服务器负责数据的接收、存储、分析和展示的功能。该系统的模块划分如图 3-5 所示。

- 客户端模块功能概要描述：

用户 UI：为用户提供简单易用的用户 UI，使用户能够方便的进行数据采集和上传的操作，能够向用户展示感知信息的统计图表，展示拍摄照片的瀑布效果，能够展示接受到的激励任务。准确稳定，避免崩溃，避免由于交互歧义对用户造成的困扰。

PM2.5 分析：对指定地点拍摄的图片进行建模，在获得模型之后，对新拍摄的照片能够根据模型预测出该时该地的 PM2.5 值。

控制模块：对客户端的其他模块进行调度和控制，封装对数据库的各种操作。

采集模块：利用客户端系统提供的各种传感器读取 API，读取传感器数据，保存到客户端数据库中。

通信模块：负责在客户端的与服务器之间的通信功能，为用户 UI、PM2.5 分析等模块提供与服务器的通信接口，确保数据上传、下载的顺利进行。

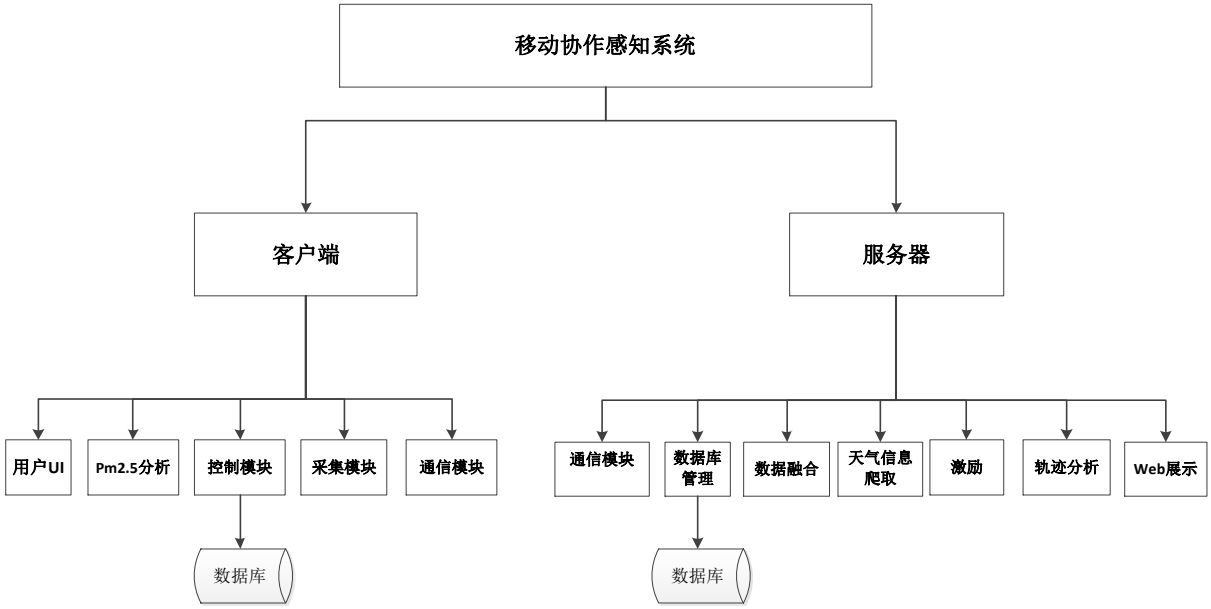


图 3-5 系统模块划分

- 服务器模块功能描述：

通信模块（即网络接口模块）：负责服务器与客户端之间的所有通信功能，确保数据的上传、下载、查询、任务推送能稳定进行。支持并发访问。

数据库管理模块：构建分布式数据库系统，将对数据库的所有操作进行封装，为其他模块提供数据库操作的接口。

数据融合：对客户端收集的数据进行分析、清洗和补齐，使得数据在时空分布中，尽量达到均匀。

天气信息爬取：为客户端的 PM2.5 分析模块服务，定时爬取指定 PM2.5 检测站的数据，用于客户端的 PM2.5 预测模型的建立。

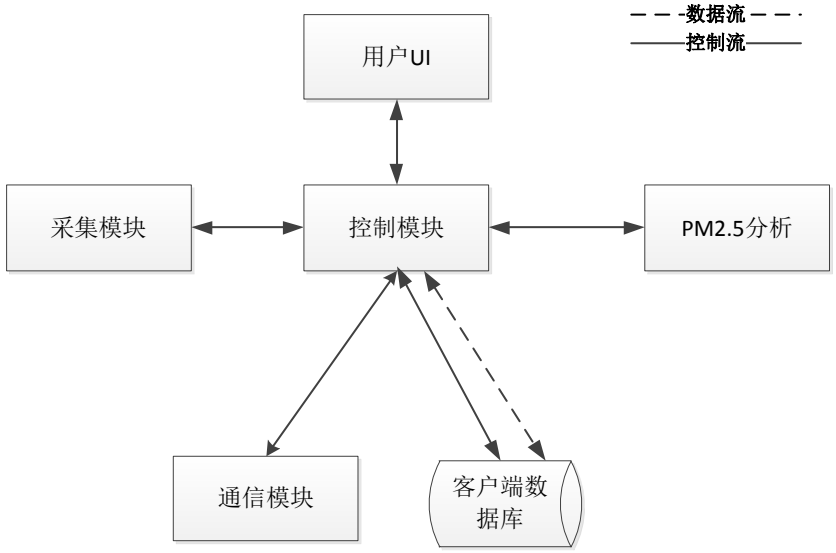
激励模块：数据融合得出的结果中，会一些地区数据较稀疏，即该地区存在数据缺失；由于用户的隐私设置不同，每个用户上传的数据类型可能存在缺失。对于缺失的数据，给可能提供该数据的用户发送任务和报酬，激励用户主动采集系统缺失的数据，并对完成任务的用户给予奖励。

轨迹分析：对用户上传的照片和数据中包含的位置信息进行分析，建模得出用户的运动规则，并利用该运动规则为用户提供相关的服务。

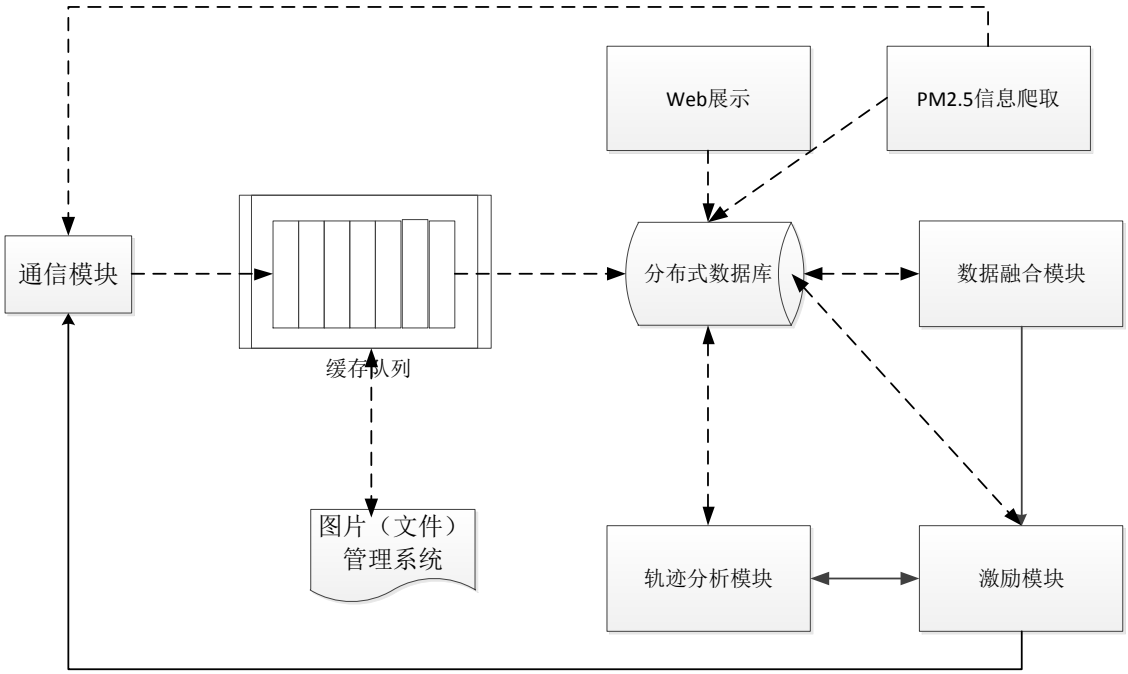
web 展示：对采集的数据进行聚类、分类、展示。在地图上标注信息点，将图片和数据根据点在地图上进行关联展示。

3.2.2 系统逻辑结构

由于整个系统的复杂性，各功能模块各司其职，相互有着密切的联系。系统的逻辑结构如图 3-6 所示。客户端各模块通过控制模块相互联系，服务器端各模块通过分布式数据库管理模块相互联系。其中激励模块与轨迹分析和数据融合模块联系紧密，web 展示模块和 PM2.5 信息抓取模块在服务器端独立运行。



（a）客户端模块逻辑关系



（b）服务器模块逻辑关系

图 3-6 逻辑结构

客户端和服务器都有相对应的通信模块，他们之间进行信息的交互，作为客户端和服务器之间的通信接口。客户端的通信模块负责将客户端收集好的数据以事先协商好的格式发送给服务器的网络接口模块，此外客户端通信模块还负责数据下载、数据查询，即向服务器发送某种格式的请求信息，接收服务器网络接口模块的返回信息，并将其传递给能处理该信息的客户端模块。而与之相对应的服务器的网络接口模块，负责对客户端请求的应答，根据上传数据格式的不同进行相应的存储和数据库操作，针对客户端的请求进行数据库查询，生成对应格式的应答返回给客户端通信模块。与此同时，服务器的网络接口模块还能够将激励模块传递来的激励任务信息主动推送到指定客户端，完成激励任务的推送功能。

缓存队列是网络接口模块和分布式数据库模块之间的缓冲。由于网络接口模块接收数据上传的速度和写入数据库的速度差别较大，当大量数据并发上传时可能导致数据拥塞，影响网络接口模块对客户端的响应时间和吞吐量。网络接口模块将接收到的数据先存入缓存队列，然后由数据库管理模块从缓存队列中读取数据，解决了数据上传和数据写入之间速度差异导致的矛盾。

PM2.5 信息爬取模块主要完成对指定的城市的 PM2.5 监测站点的监测数据的爬取。该模块主要是为了配合客户端的 PM2.5 分析模块的功能需求。服务器的信息爬取模块获得的站点监测数据将存入服务器数据库中，在客户端提出查询请求时，被返回给客户端，作为客户端 PM2.5 分析模块的对比信息，通过误差对比客户端完成模型训练。

数据融合模块、激励模块和轨迹分析模块对收到各种感知数据进行处理。数据融合

模块和轨迹分析模块能够利用数据验证各自的算法，另外，根据这两个模块的计算结果，激励模块能够产生对应的激励任务。激励模块在制定激励策略之时，能够参考两个模块计算结果，从而达到更好的激励效果。

3.2.3 网络接口模块交互设计

服务器端的并发数据处理系统，即网络接口模块，主要为系统服务器端提供通信功能。主要功能包括数据上传、数据下载、数据查询、用户管理和任务推送。

数据上传功能主要接收客户端上传的照片和光照噪声等感知数据，将数据处理后存入服务器的数据库中；数据下载功能为客户端的查看照片墙功能服务，返回客户端需要的照片数据流；数据查询功能为客户端的 PM2.5 分析功能提供服务器数据查询接口，将建模所需的数据查询到并返回客户端；用户管理功能负责进行用户注册和登录，并对用户权限、个人配置等信息进行管理；任务推送功能为服务器的激励模块提供服务，激励模块产生任务，将任务相关内容通知网络接口模块，网络接口模块使用推送功能将任务推送到客户端，并接受客户端的响应，将响应返回给激励模块。本小节主要说明网络接口模块与系统其它模块之间进行不同功能时的调用关系和消息交互。

3.2.3.1 数据上传

客户端用户使用客户端收集到的图片数据、PM2.5 数据和经纬度等信息，会在网络服务较好的情况下上传到服务器，保存到服务器的数据库中。数据上传过程涉及的功能模块和几个模块之间的消息交互的序列图如下图 3-7 所示。上传请求由客户端发起，消息的内容中包括请求的类型、以及具体的数据。服务器的网络接口模块在接收到该请求后，会根据请求的类型调用不同的方法对上传的数据进行不同的处理。

其中，JSON 格式的光照噪声等感知数据的后续处理和照片数据的后续处理是不同的。照片数据需要保存到文件系统中，并进行照片信息解析。初步处理后，网络接口模块将数据写入缓存队列中。如果缓存队列服务正常工作，网络接口模块在写入数据之后将得到写入成功的返回值，接着网络接口模块会给客户端模块同样返回一个上传成功的返回值。

如果上述过程发生异常，例如缓存队列服务未正常开启，网络接口模块无法正常将数据写入缓存队列，将得到写入失败的返回值。网络接口模块会给客户端模块返回一个上传失败的返回值，提示客户端进行重传。

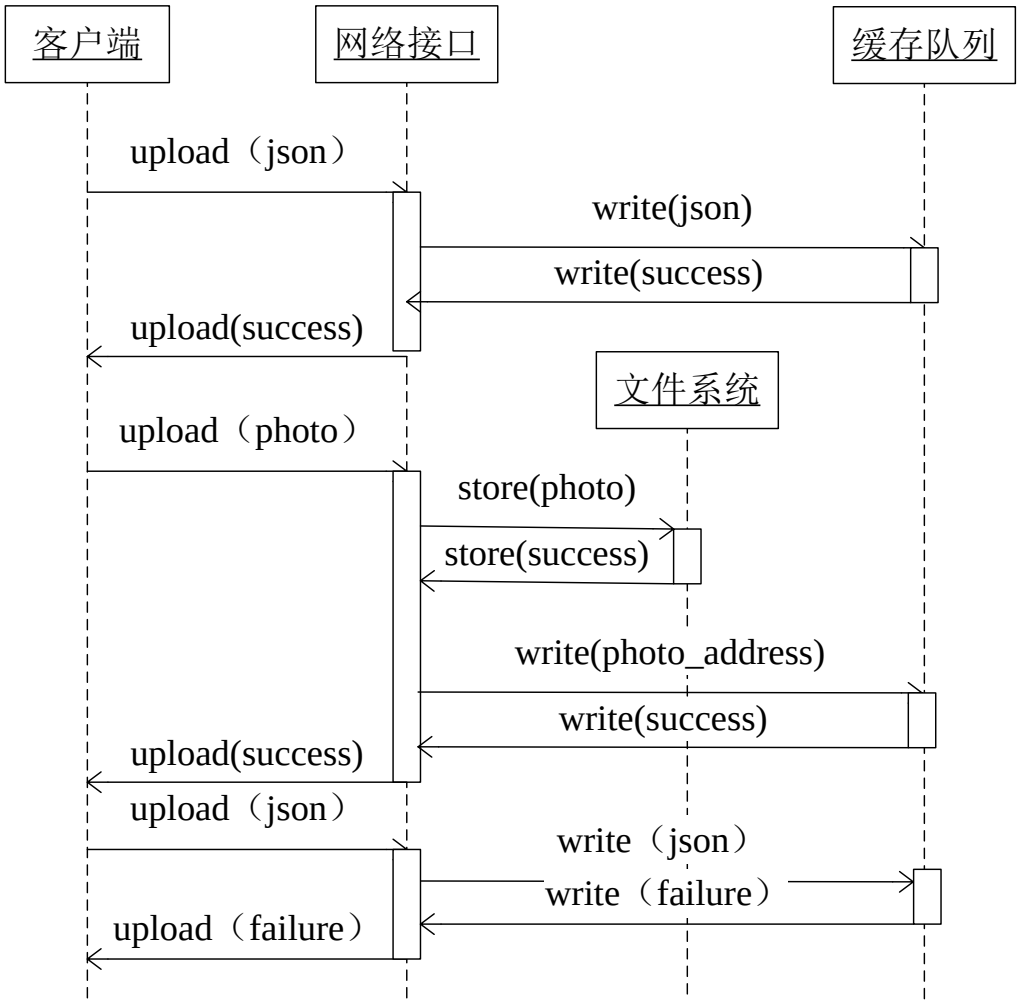


图 3-7 数据上传过程消息图

3.2.3.2 数据下载

客户端为用户提供了查看照片墙的功能，因此需要服务器提供数据下载的功能作为支持。数据下载过程的消息交互图如下图 3-8 所示。客户端首先提出查询请求，请求的内容是 JSON 格式的字符串，按照事先约定的格式表示了客户端要下载的照片的相关条件（照片时间、经纬度等信息）。服务器的网络接口模块在接收到查询请求后，会分析查询条件，根据该条件查询服务器的数据库，获得客户端需要的照片的名称和 URL 等信息。网络接口模块将查询结果以约定的 JSON 格式返回客户端。客户端接收到查询结果，经过分析，得到需要下载的照片的名称和地址，然后根据该信息向服务器下载照片。服务器中保存有客户端上传的所有照片文件，会根据客户端的下载请求返回照片。

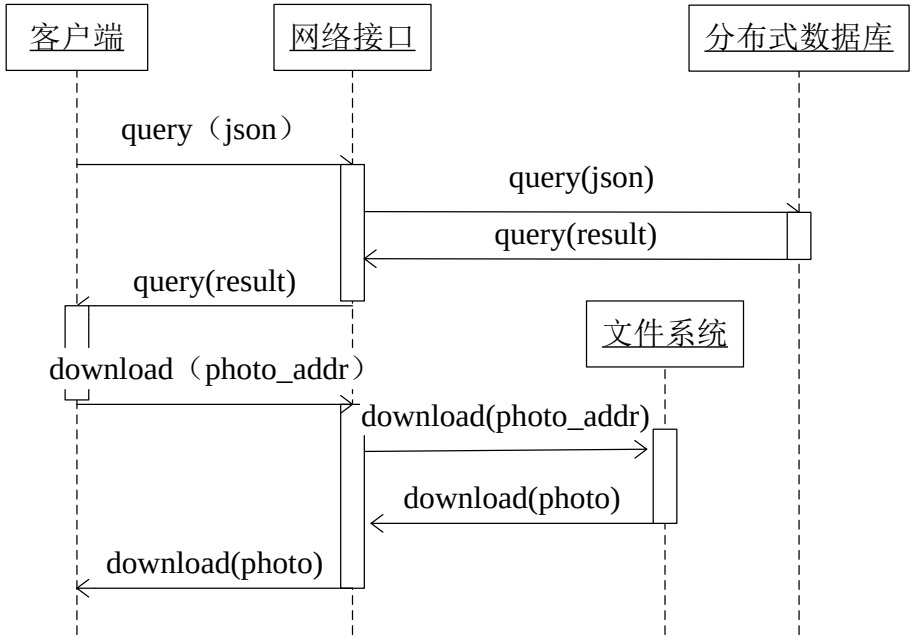


图 3-8 数据下载过程消息图

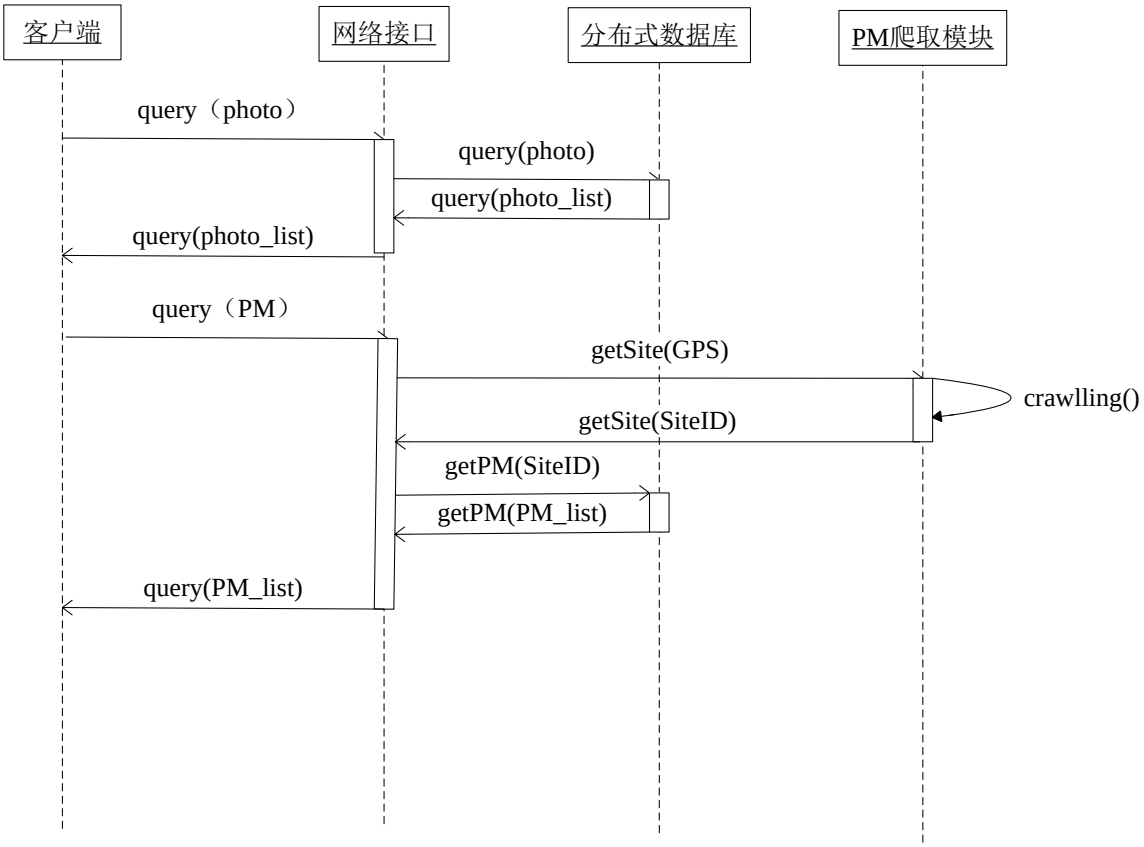


图 3-9 数据查询过程消息图

3.2.3.3 数据查询

客户端提供的 PM2.5 分析预测功能，是参与式感知平台的核心功能之一。该功能的实现包括模型训练阶段和 PM2.5 预测阶段。在第一阶段，主要是通过在客户端拍摄一定量的照片，和环境监测站的 PM2.5 监测数据一起建模，进行模型训练。其中，PM2.5 监测数据是由服务器的 PM2.5 信息爬取模块在网络上爬取之后存入数据库的。因此，服务器为客户端提供查询接口，使客户端能够获得数据库中保存的建模所需的 PM2.5 监测数据。数据查询过程网络接口模块与系统其它模块的消息交互如图 3-9 所示。

客户端首先提出查询请求，请求的内容是 JSON 格式的字符串，其中标示了查询数据类型，并按照事先约定的格式表示了建模所需的数据类型、数据时间等查询条件。服务器的网络接口模块在接收到查询请求后，会首先分析查询条件，调用 PM 爬取模块获得查询条件中的 GPS 信息所代表的实际 PM 监测站点编号。获得该站点编号之后，网络接口模块再根据查询条件的内容和站点编号查询服务器的数据库，获得 PM2.5 相关数据，并以约定的格式返回 JSON 字符串。当客户端的查询请求的格式不正确，或数据库中没有需要的数据时，会导致查询失败，网络接口模块会生成失败原因的信息写入日志文件中，并给客户端返回查询失败的提示。

3.2.3.4 用户管理

用户管理功能包括用户注册、用户登录以及用户个人信息维护。用户管理并不是一个独立的过程，在客户端与服务器进行通信时，用户管理功能经常被启用对用户个人信息进行更新。用户管理动能主要涉及客户端模块、网络接口模块、数据库模块，其消息交互过程如图 3-10 所示。

用户注册请求由客户端用户注册界面发出，请求中包含有用户名，用户密码和用户配置信息。接收到该请求后，将首先查询数据库确认用户是否已经注册过。如果用户已注册过，则注册失败，将结果提示信息返回客户端。数据库中不存在该用户名时，说明用户没有注册过，则向数据库添加一条用户信息记录，为用户进行注册，同时处理注册请求中的配置信息，将其统一写入数据库中。

注册完成后用户可以进行登录，以及登录后的上传文件、查询等各种操作。用户登录请求是从客户端的用户登录界面发出，使用用户名和密码作为请求中的参数进行登录。接收该请求后，需要查询数据库获得用户注册信息。如果查询失败，用户名不存在或者密码不符合，则登录失败，返回提示信息给客户端。查询成功，则登录成功，将提示返回给客户端，用户可以进行后续操作。

用户使用客户端进行上传文件、个人配置、注销等操作时，都会导致用户相关信息发生变化。用户管理功能需要将这些信息的变化及时更新到数据库中。例如，用户进行

了文件上传，则用户信息中用户上传文件数发生了改变，需要将其更新到数据库中；用户注销会引起用户登录次数、登录时间等信息的改变，也需要及时进行更新。用户信息影响用户的可信度、用户所得激励，必须进行及时的更新和管理。

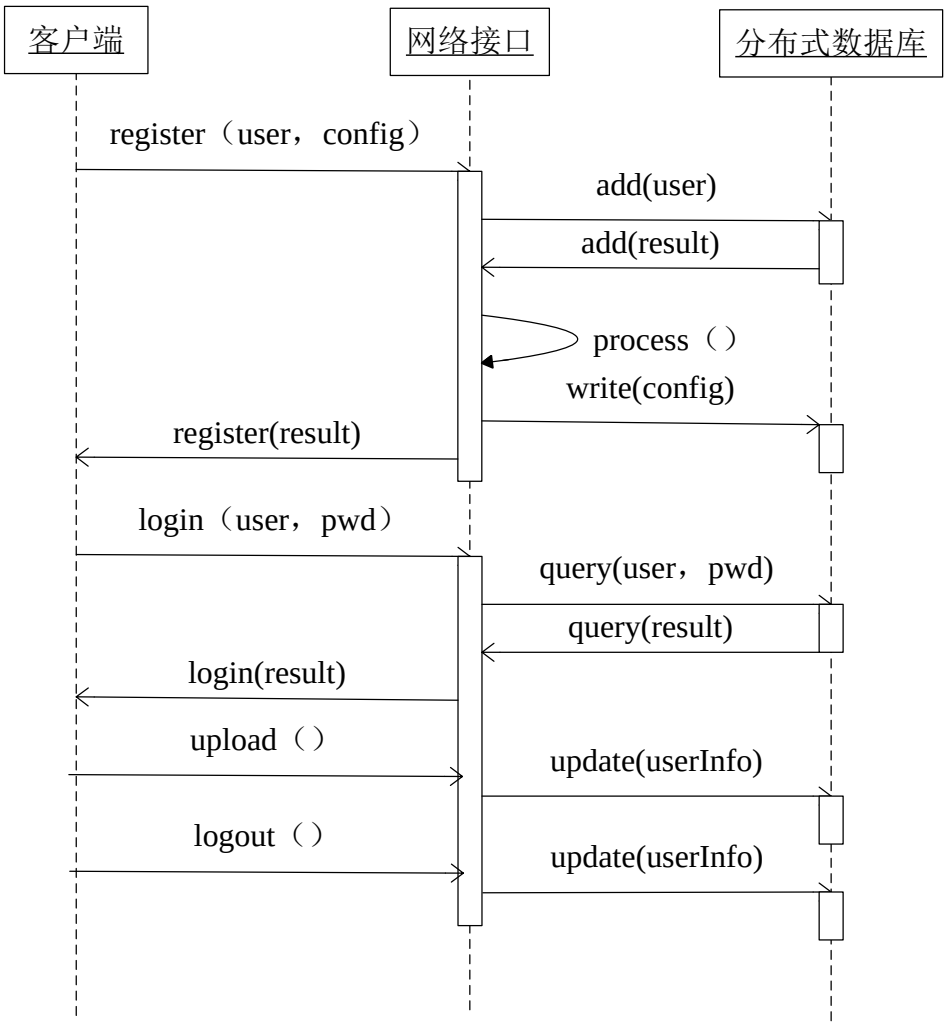


图 3-10 用户管理过程消息图

3.2.3.5 任务推送

任务推送过程是由服务器端的激励模块发起的。在服务器端，数据库模块负责存储上传的感知数据，当采集的数据达到一定规模时，数据库模块将会通知数据融合模块对数据的分布进行重新计算，从而得出数据分布最为稀疏的区域的编号，并将该消息通知给激励模块。激励模块将调用轨迹分析模块的功能，计算出在数据稀疏区域可能存在的用户。激励模块由此产生激励任务，将任务发布给用户，促使用户接受任务，上传我们需要的数据。激励任务包括可能会接受任务的一组用户，可能发给用户的激励以及任务的具体内容。

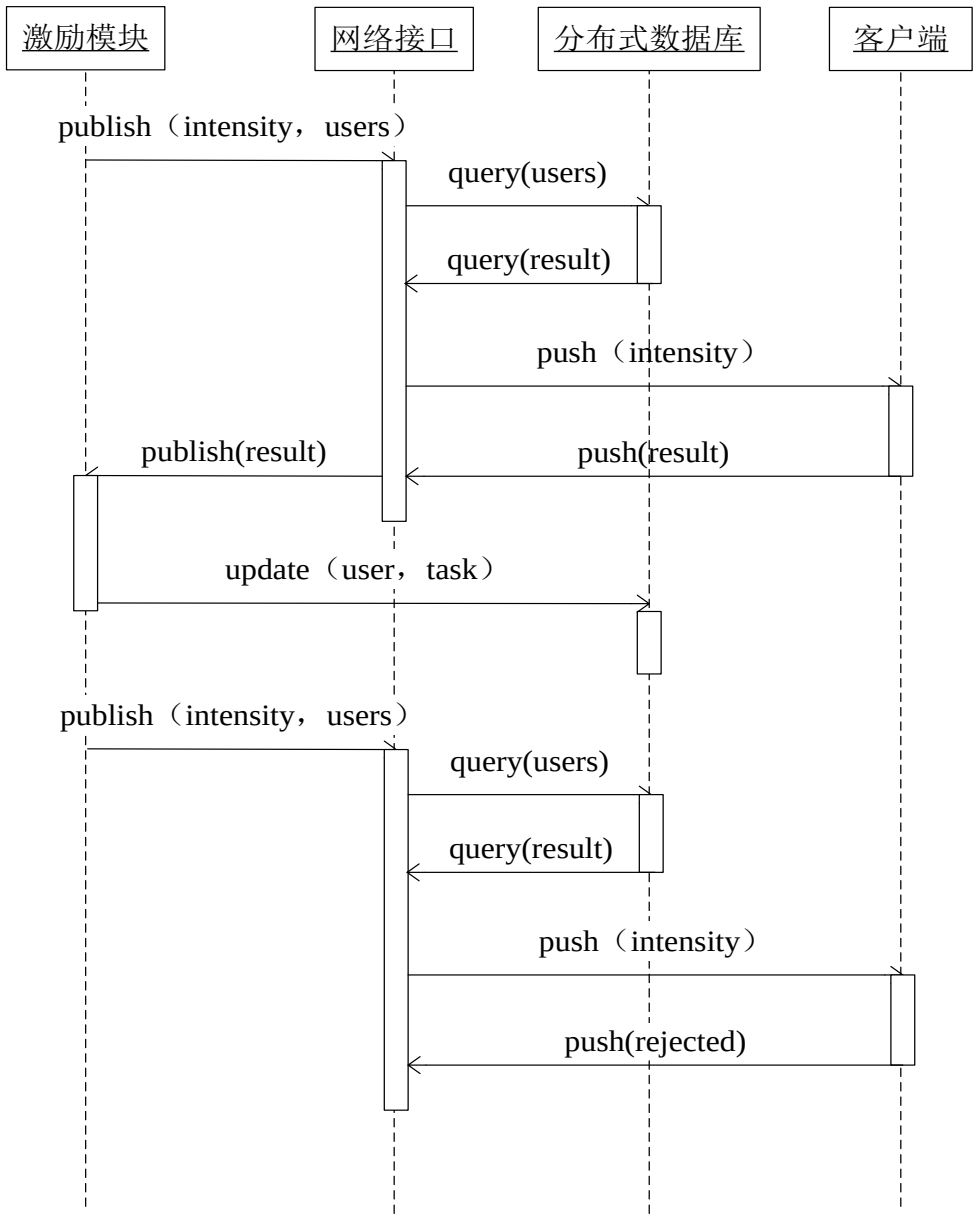


图 3-11 全体推送过程消息图

激励任务根据任务目标群体的不同，可以分为全体推送任务和组推送任务。对于全体推送任务，任务发布者（即激励模块）并不对所有用户进行区分，直接将当前的全体客户端用户作为任务的发布目标；而对于组推送任务，激励模块通过服务器的数据融合模块和轨迹预测模块的相关处理结果，确定了接受任务可能性较大的一组用户，激励模块指定任务只对这一组用户推送，从整体的角度来看，组推送任务更加具有针对性，节约了系统资源。

对于全体推送和组推送两种激励任务，采用不同的方式进行任务下发。激励模块产生任务之后将此激励任务发布给服务器端的网络接口模块。对于全体推送，其模块之间消息交互的基本流程如图 3-11 所示。网络接口模块接收到推送任务的调用请求后，将

查询数据库，获得任务的详细内容，然后将任务向全体客户端推送。推送消息的内容为任务详情，例如请用户到某个地点拍摄照片，以及完成该任务的奖励。用户对任务作出响应（是否参与），网络接口模块接收到该响应，将此结果发送给激励模块，激励模块根据用户的响应对数据库进行更新，即完成任务推送过程。如果用户拒绝接受任务，网络接口模块同样会接收到该响应并将此响应返回给激励模块。

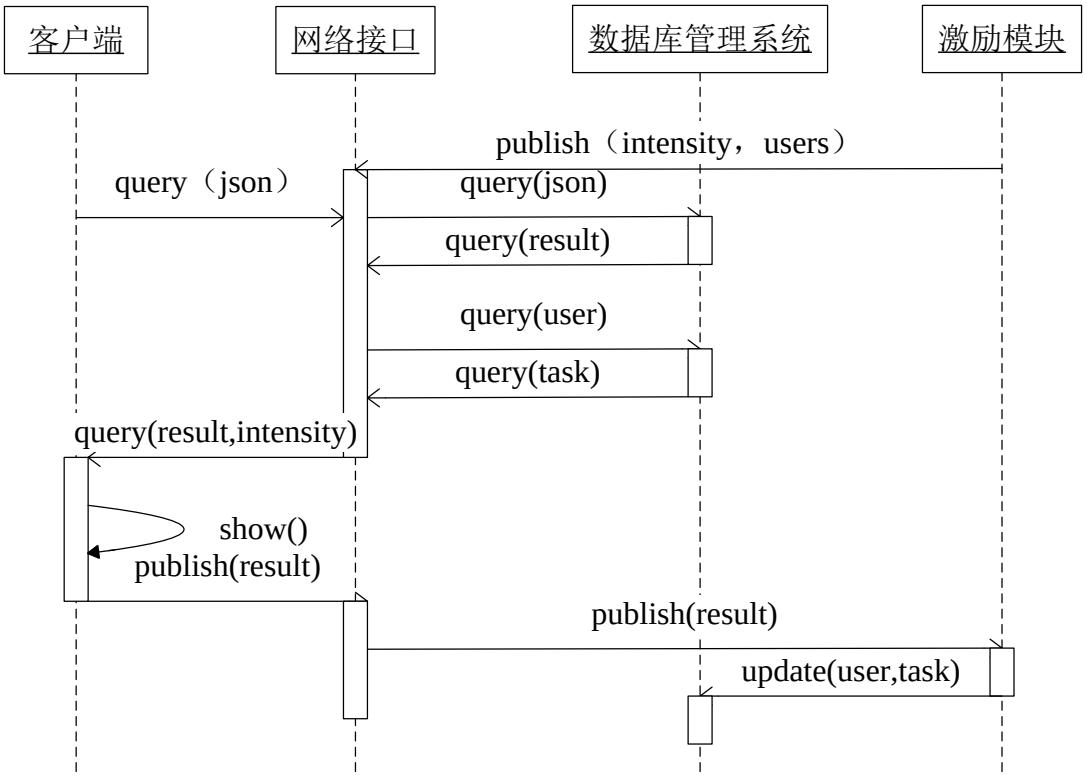


图 3-12 组推送过程消息图

对于组推送的任务发布方式，则采用客户端主动查询的方式将任务发送到客户端。激励模块将产生的激励任务发布给网络接口模块，网络接口模块得知这是一个组推送任务，则不立即进行推送。在客户端进行数据上传、数据查询等操作时，客户端需要向服务器发出请求，网络接口模块在对请求做出对应的应答时，如果该客户端是激励任务的目标用户，将激励任务作为应答的一部分返回给客户端。其模块之间的消息交互流程如图 3-12 所示。当网络接口模块接收到客户端的请求并处理得出结果，需要对客户端进行应答时，它会查询数据库模块，得知该用户是否在推送任务的目标用户组中，如果在用户组中，则继续查询获得任务的详细内容，然后将任务详情加入原本的应答消息中，此时返回给客户端的应答消息将包含对客户端请求的应答以及发送给客户端的任务。客户端对于接受到的应答消息进行处理，如果发现了推送任务，则进行解析展示等相关处理，由用户决定是否接受任务。

3.3 本章小结

本章首先对整个系统的总体功能性需求进行了说明，并对参与式感知平台的服务器端并发数据处理系统的功能通过用例图的形式进行了初步的阐述和说明。此外，对系统的非功能性需求做出了说明。然后介绍了整个平台的模块划分和各模块基本功能，说明了模块之间的逻辑结构。接下来，说明了并发数据处理系统，即网络接口模块的功能和该模块与其他模块的调用关系。通过消息图说明了该模块工作过程中与其他模块的消息交互的基本过程。该模块的详细设计，包括内部子模块划分、流程、接口定义等将在下一章具体阐述。

第四章 网络接口模块的设计与实现

网络接口模块负责服务器的数据接收、数据存储和推送，是客户端和服务器之间的通信接口，在整个系统中是重要的子模块之一。根据功能需求将该模块分为数据并发处理和任务推送子模块。本章将详细介绍网络接口模块的设计与实现。

4.1 网络接口模块内部结构

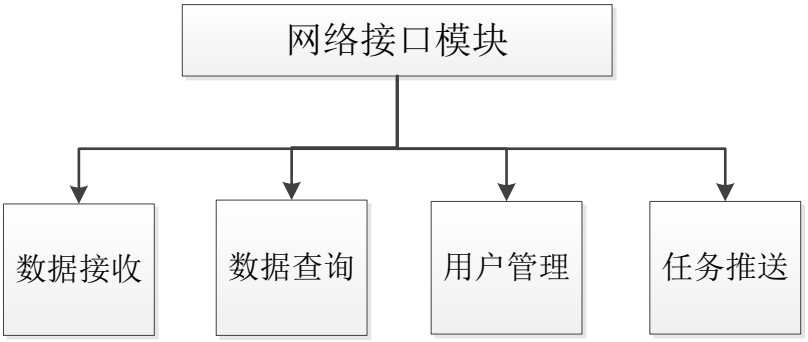


图 4-1 网络接口模块子模块划分

在前一章对网络接口模块的功能需求进行了分析，网络接口模块主要提供了数据并发处理和任务推送两个基本功能，而并发处理功能又可细分为数据接收、数据查询和用户管理功能。因此，网络接口模块的内部结构如图 4-1 所示。

- 数据接收

数据接收模块负责接收客户端上传的数据。在参与式感知平台中，客户端会收集光照、噪声、经纬度等各种传感器的值，还会拍摄照片进行上传。数据接收子模块会接收各种数据，根据数据的类型进行对应的处理。对照片数据进行解析压缩然后保存到文件系统中，然后将照片相关信息存入数据库；对于其他类型的感知数据，直接解析然后保存到数据库。保存成功或者失败，该模块都会产生提示信息，返回给客户端，同时为了在发生异常时进行快速的修复，每次上传过程都会记录到日志中。考虑到并发上传数据的情况，该模块还需对并发请求的情况进行处理。

- 数据查询

由于数据查询和数据下载功能联系比较紧密，因此都在数据查询模块进行实现。客户端的核心功能之一是 PM2.5 预测，即通过用户自己拍摄的照片和已知的 PM2.5 数据建模，模型建立之后能够预测出用户新加入模型的照片的 PM2.5 值。该功能需要查询照片拍摄的时间地点的 PM2.5 值，需要向服务器发出查询请求，查询数据库中的数据。数

据查询子模块能够接收到查询请求，对其进行解析，然后调用服务器的对应接口，进行数据查询操作，将结果以约定的格式返回客户端。

- 用户管理

用户管理模块负责对用户注册、登录、用户信息更新进行管理。用户要使用系统，必须首先进行注册和登录获得系统用户的身份。该身份不仅用于用户上传和下载数据，还用于激励模块推送激励任务和计算任务奖励。用户登录之后进行的各种操作会使用户自身信息发生变化，用户管理模块还负责记录这些变化并进行更新，保证系统各模块在需要与客户端进行信息交互时能了解每个客户端的状态。

- 任务推送

任务推送模块负责实现对参与式感知任务的主动下发功能。感知任务由激励模块产生，通过进程间调用的方式，调用任务推送模块的功能。任务推送模块分析感知任务的内容，确定推送目标、推送信息的内容，将信息推送到指定的客户端目标该模块的实现需要客户端和服务器的共同配合，考虑到客户端网络资源和电池资源的限制，该模块在实现的过程中需要保证消息推送的稳定性、较少的延迟和资源消耗。采用长连接的方式进行消息推送，节约资源。

4.2 并发处理功能设计

由于用户客户端情况、使用习惯的不确定性，在参与式感知平台中，必须提供数据并发处理的功能，满足用户大量并发上传文件、查询数据的需求。

并发处理功能的实现采用线程池与 epoll 技术结合的方式。在网络接口模块的主线程中预先初始化一定数量的线程，当服务器接收到新的客户端请求时，就启动空闲的线程进行处理；当请求的数量超过线程池中的空闲线程数量时，新的请求就会进入等待状态，当线程池中有空闲线程时才会被处理。线程池在实现多线程的同时，避免过多创建线程对系统资源的浪费。同时，使用事件监听的 I/O 策略，避免了旧有的 select 机制的轮询方法，极大的节约系统资源，而内核事件通知的机制，使得对于请求的处理效率更高。

并发处理功能实现的基本流程如下图 4-2 所示，其中，server 模块负责接受客户端请求，并且分发异步回调事件，同时保证整个异步发送请求期间缓冲区的有效性。connection 模块负责发起 socket 异步读请求，并发起异步时间等待，每次处理完一个请求，重设超时值为 60 秒，如果超时时间到，进入 timeout 函数。timeout 函数调用 shutdown(socket)方法会导致读失败，此时没有新的任务发起，此对象会被销毁。request_parser 模块会分析获取到的每一个请求，然后交由 request_handler 模块处理。log 模块记录请求的日志事件，在系统出错时进行查看和恢复。file_cache 模块负责文件缓存，提高文件读写的性能。

在网络接口模块工作时，主函数初始化，创建工作线程 connection 模块，构造函数中设置 60 秒超时，发起 socket 异步读请求，并发起异步时间等待。如果读到数据，则 connection 模块将调用 request_parser 模块分析请求，然后调用 request_handler 模块处理请求，同时调用 log 模块记录 request 日志事件。 request_handler 模块解析请求，将分析出请求的文件的路径和文件名，file_cache 模块尝试从内存中获取文件。如果文件存在且发送成功，记录成功日志，否则记录失败日志。

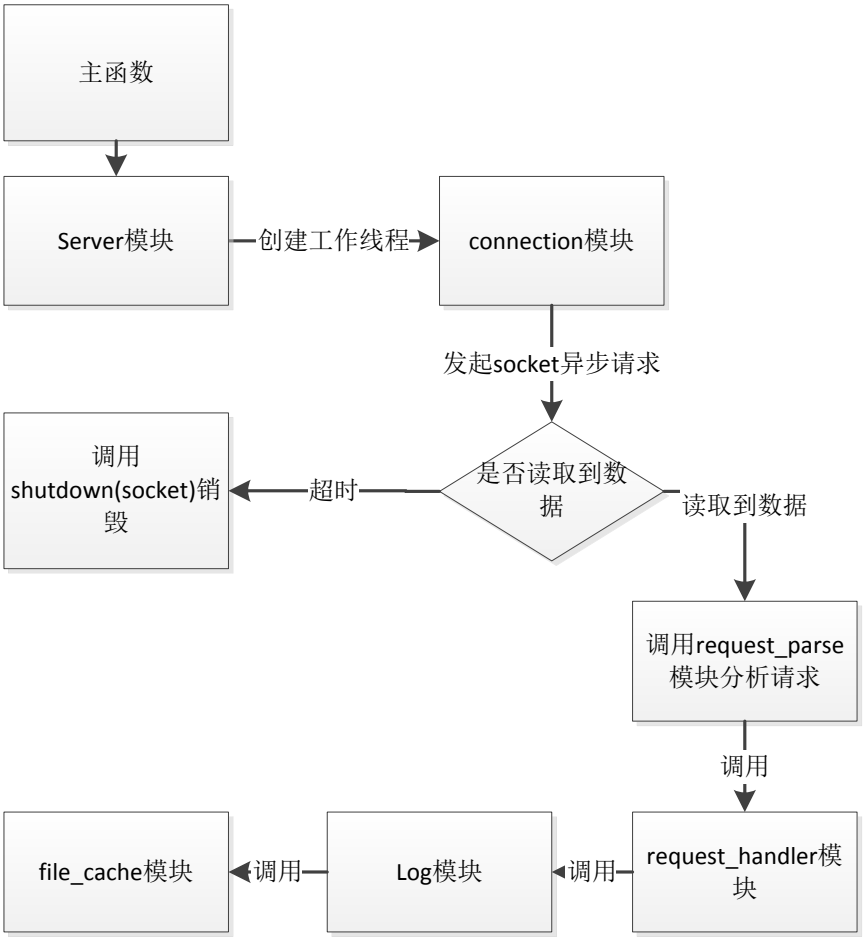


图 4-2 基本工作流程

在上述流程中，采用异步的方式读取请求并处理请求，同时分析请求的数据，并且处理文件的读取。详细的工作流程如图 4-3 所示。在 connection 对象中，设置了 60 秒超时，每次发起 socket 异步读请求的同时发起异步时间等待。每次处理完一个请求，重设超时值为 60 秒。如果超时时间到，进入 timeout 函数，此函数调用 shutdown(socket)，会停止 socket 活动，等待异步读取完成之后销毁 connection 对象。

如果读到数据，即在 60 秒内接受到了请求，则会调用 request_parser 对象分析请求，request_handler 对象处理请求，同时调用 log 对象记录 request 日志事件，记下请求时间.。根据 request_handler 从请求中解析出的文件名，调用 file_cache 的 cache_get 尝试从内存中获取文件，如果获取成功，设置返回结果 200。如果 file_cache 获取失败，内存中没

有此文件，尝试从文件系统中读。如果读取到了请求的文件，在 file_cache 中添加文件，设置返回结果 200。如果 file_cache 中文件数量大于一定值(500 个)，则移除一个最早的，把此文件加入到 file_cache。发送结果给客户端后，将处理时间(当前时间-请求时间)和相关信息记录到日志。如果文件存在且发送成功，记录成功日志，否则记录失败日志。

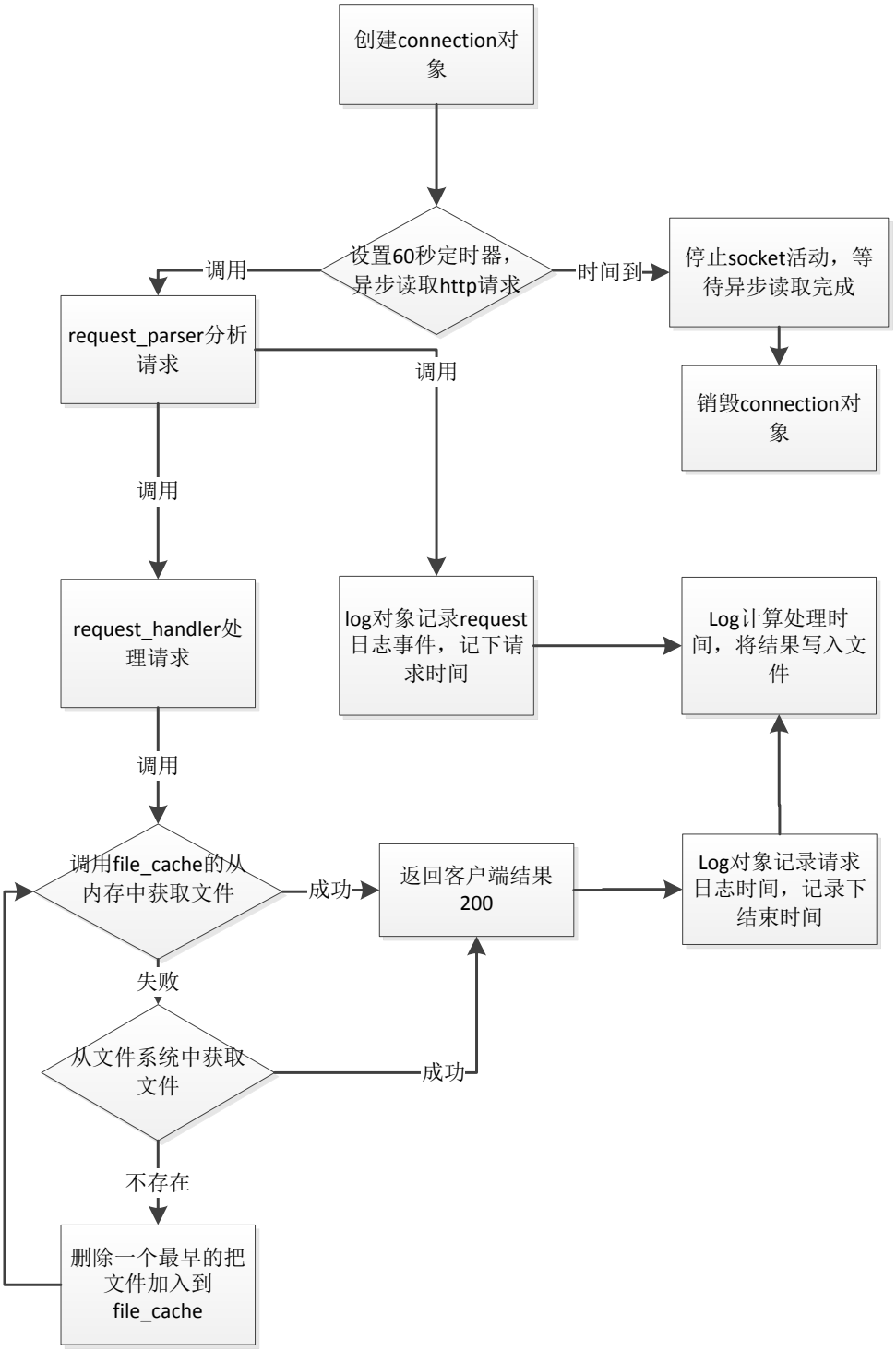


图 4-3 请求处理工作流程

4.3 数据接收子模块设计

4.3.1 类设计

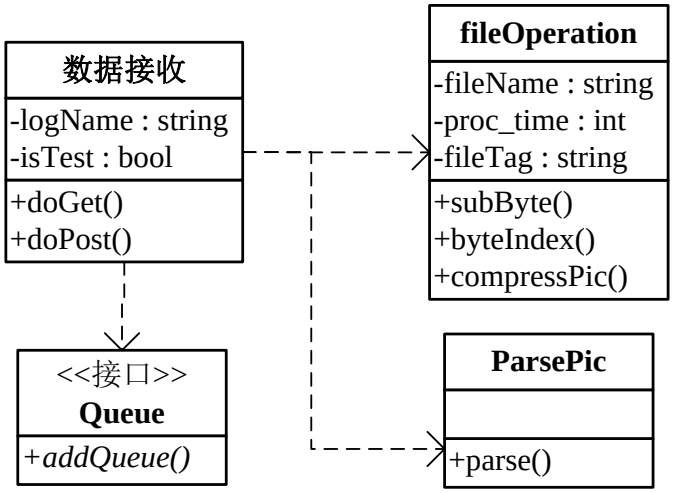


图 4-4 数据接收子模块类图

数据接收子模块的类图如图 4-4 所示。数据接收类的 logName 属性记录了日志文件名称，isTest 标识当前是否处于测试状态。当处于测试状态时，接收的文件会保存到测试文件夹中，便于测试完毕进行删除。方法 doGet 主要用于测试网络连接是否正常，而 doPost 方法则负责接收客户端上传的文件或数据。当上传的是数据时，数据接收类直接进行处理，然后调用 Queue 类的 addQueue 方法将其添加到缓存队列；当上传的是文件时，数据接收类首先会调用 ParsePic 类的 parse 方法解析文件，文件解析成功后需要调用 fileOperation 类对文件的 byte 流进行分析处理。在使用 subByte 和 byteIndex 等方法从 byte 流中获得照片文件之后，再调用 compressPic 进行文件压缩。压缩完成后将原文件和压缩文件的地址与文件信息生成 JSON 字符串，调用 Queue 的 addQueue 将其添加到缓存队列。

4.3.2 数据库设计

图片收集表如表 4-1 所示，主要用于存储图片的相关信息。数据接收模块接收到图片后，对图片格式进行解析，从图片 EXIF 域中解析出的图片拍摄时间、上传用户的用户名、拍摄地点、图片的 PM2.5 预测值等信息。这些信息和图片在服务器文件系统的保存地址都写入到数据库的图片收集表中。该表的主键是图片拍摄时间、用户名、图片经纬度。

表 4-1 图片收集表

属性	类型	描述	主键
userName	String	用户名	√
TimeStamp	string	拍摄时间	√
Lon	Double	经度	√
Lat	double	纬度	√
Content	String	图片地址	
Compress	String	压缩图片地址	
Fpm	Int	图片 PM2.5	
Poi	Int	图片 POI 标识	

表 4-2 感知数据收集表

属性	类型	描述	主键
userName	String	用户名	√
TimeStamp	string	收集时间	√
Lon	Double	经度	
Lat	double	纬度	
Light	Int	光照	
Temp	Double	温度	
Noise	Int	声音	

感知数据收集表如表 4-2 所示，它主要存储了非图片类型的上传数据。客户端上传 JSON 格式的字符串中，包括用户名、数据收集的时间、地点、光照温度噪声等感知数据。数据接收模块接收到感知数据后，同样进行解析，然后把这些数据保存在数据库的感知数据收集表，以供后续的数据融合、轨迹分析等模块使用。

4.3.3 流程设计

数据接收子模块的基本工作流程如图 4-5 所示。基本工作过程是接收到数据，根据数据的类型进行不同的处理，将数据写入缓存队列，给客户端返回上传结果。

客户端发出的数据上传请求中有标示数据类型的标志位，模块接收到请求之后，首先根据该标志位判断上传的是照片还是普通的感知数据。

对于照片类型的上传数据，客户端会在拍摄照片的同时，采集客户端的传感器数据，将其写为 JSON 格式的字符串存入照片的 EXIF 域中。数据接收模块会对 EXIF 域的信息进行解析，如果解析失败，则说明照片格式不正确，不能存入缓存队列。如果照片信息解析成功，则能顺利解析出照片拍摄的时间地点用户名等信息。解析完成后，将照片保存在文件系统中，而把照片的信息和照片的保存地址一起存入缓存队列中。

对于非照片类型，即光照噪声等传感器数据，数据接收模块接收之后进行分析，如果格式正确，则将数据存入缓存队列。保存完成后，给客户端返回上传成功的消息。在处理过程中，如果发生数据格式错误、缓存队列服务未正常开启等问题，将导致数据保

存失败，此时会生成错误提示信息给客户端，提醒客户端进行重传等操作。

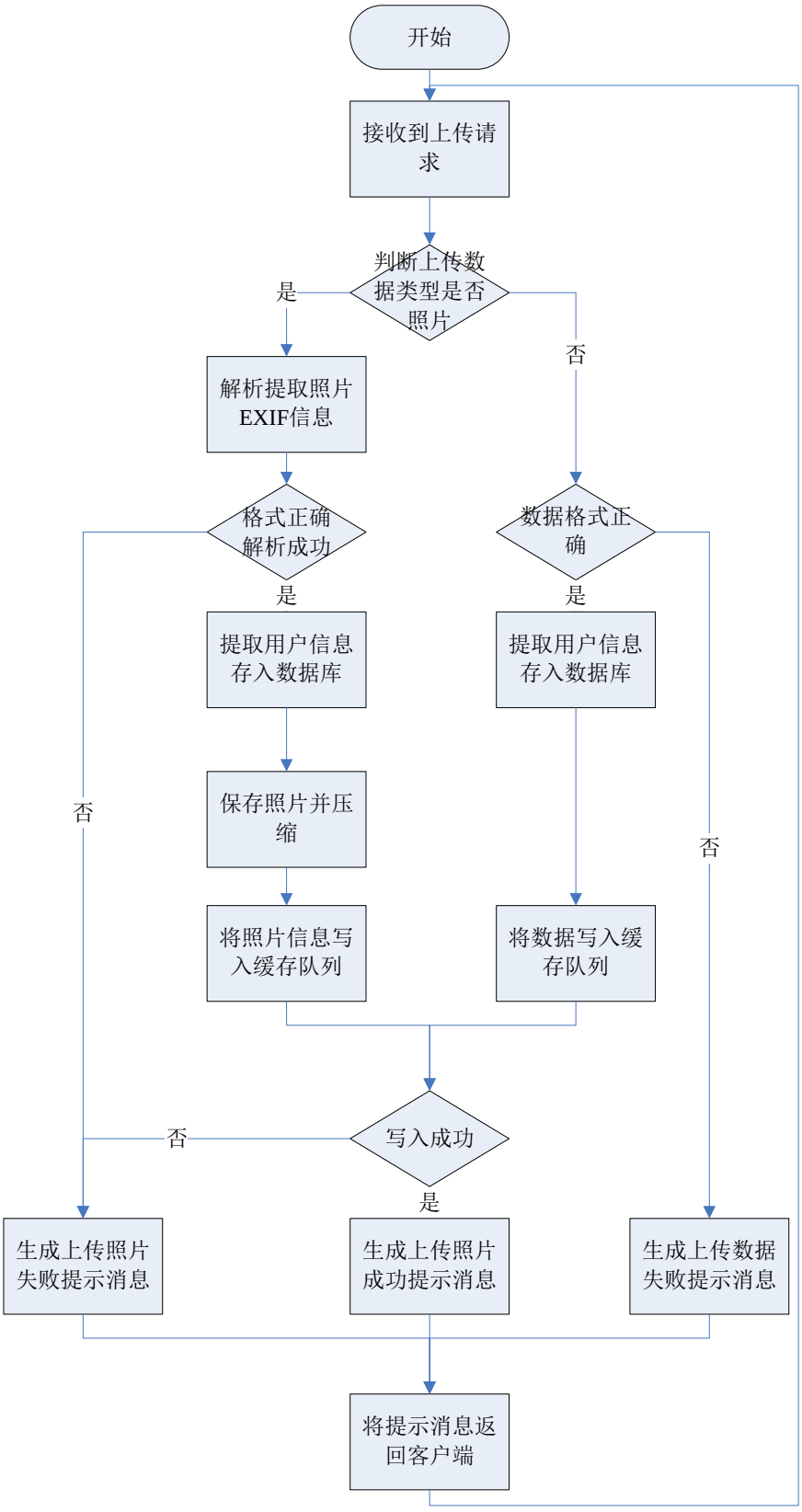


图 4-5 数据接收基本流程图

4.4 数据查询子模块设计

4.4.1 类设计

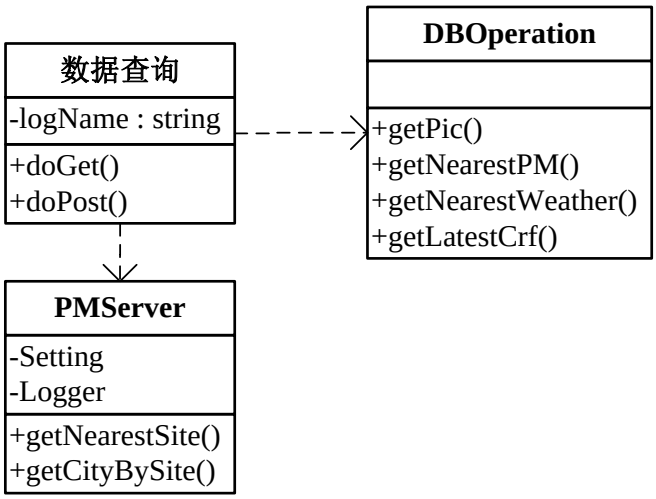


图 4-6 数据查询子模块类图

数据查询子模块的类图如图 4-6 所示。其中，数据查询类的 logName 属性记录了日志文件名称，对查询功能的测试不会对服务器的数据造成影响，因此不用设置测试状态标记 isTest。方法 doGet 主要用于测试网络连接是否正常，而 doPost 方法则负责接收客户端的查询请求，并调用其他两个类的方法完成查询过程。

查询请求主要分三类，当查询的是照片时，数据查询类调用 DBOperation 的 getPic 方法获得符合条件的照片集，并组织成约定的格式返回给客户端。当查询的是天气信息时，数据查询类首先会调用 PMServer 类的 getNearestSite 和 getCityBySite 两个方法解析查询请求中的经纬度信息，获得该经纬度附近最近的观测站和该经纬度所在的城市。然后根据站点 ID 和城市名，调用 DBOperation 类的 getNearestPM 方法可以获得该站点的 PM2.5 信息，getNearestWeather 方法返回站点的天气信息。当查询的是相机模型信息时，数据查询类直接调用 DBOperation 的 getLatestCrf 方法获得结果。

PMServer 类获得站点信息和城市信息时，是通过实时查询网络获得的。它的 Setting 属性标示爬取信息时需要的配置文件， logger 是其爬取数据的日志文件地址。

4.4.2 数据库设计

前面的章节已经提到，数据查询模块主要是为客户端的 PM2.5 预测模块服务，因此涉及到的数据库表格主要是与 PM2.5 建模相关的几张表。站点观测表中保存有服务器爬取到的环境监测站点的信息，主要是站点的 PM2.5、PM10 和 AQI 浓度值，该表的主键是城市名、站点编号和观测时间。天气表中保存的是服务器爬取到的天气站点信息，包

括温度、湿度、压强、风速、总体天气情况等数据，该表的主键同样是城市名、站点编号和观测时间。CRF 模型表保存有不同的手机型号对应的 CRF 模型的描述，主键是手机型号。

表 4-3 站点观测表

属性	类型	描述	主键
City	String	城市名	√
SiteId	string	站点编号	√
Timestamp	String	观测时间	√
Fpm	double	PM2.5 浓度	
Cpm	Double	PM10 浓度	
Aqi	Double	AQI 浓度	

表 4-4 天气表

属性	类型	描述	主键
City	String	城市名	√
SiteId	string	站点编号	√
Timestamp	String	观测时间	√
Rain	Double	降雨量	
Weather	String	天气状况	
Humidity	Int	湿度	
Temp	Double	温度	
Speed	Int	风速	
Pressure	Int	大气压	

表 4-5 CRF 模型表

属性	类型	描述	主键
PhoneID	String	手机型号	√
Model	string	CRF 模型	

4.4.3 流程设计

图 4-7 所示为数据查询模块的流程。数据查询主要为客户端的照片墙功能和 PM2.5 预测功能服务。

数据查询子模块接收到查询请求，首先判断请求的类型。对于目前的系统来说，定义了两种基本的查询类型，即查询照片和查询 PM2.5 监测数据。如果接收到的是查询照片请求，则数据查询模块会对请求进行分析，确定查询条件，然后根据此条件对数据库中的图片收集表进行查询，确定用户需要下载的照片的地址并将其返回给客户端。客户端接收到查询请求的应答之后，会提取出其中的照片地址，从该地址下载照片。如果数据查询模块接收到的请求是查询 PM2.5 监测数据，则数据查询模块首先根据查询请求中的经纬度信息确定查询的位置最近的站点和查询的位置所在的城市，然后根据城市和站点信息查询数据库的站点观测表、天气表和 CRF 模型表，获得 PM2.5 预测模块所需的

数据，将数据以约定的格式返回给客户端。当查询类型错误或查询请求格式异常时，查询会失败，数据查询模块会生成失败的提示信息返回给客户端。

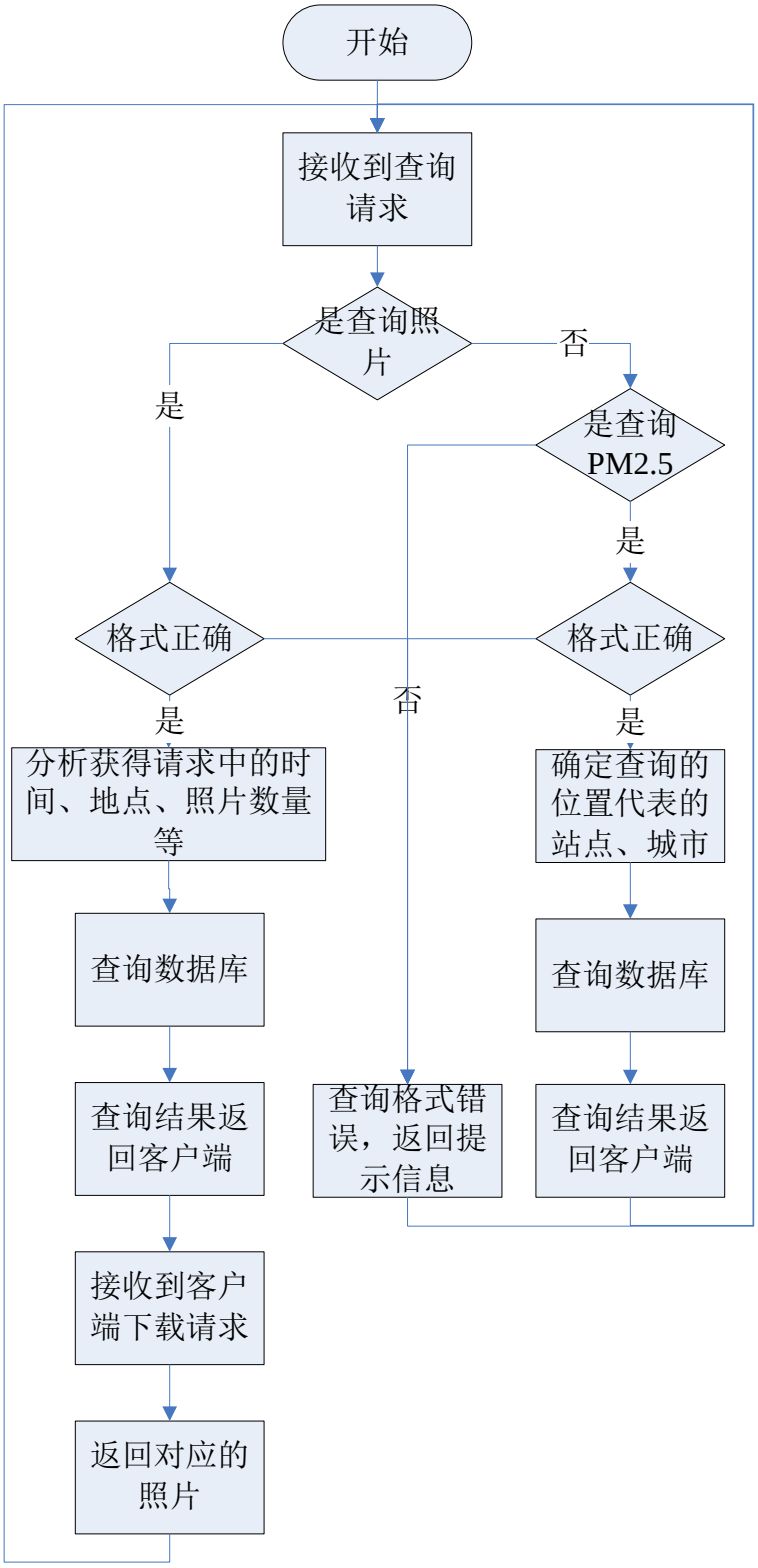


图 4-7 数据查询流程图

4.5 用户管理模块设计

4.5.1 数据库设计

表 4-6 用户信息 UserInfo 表

属性	类型	描述	主键
userId	string	IMEI 标示用户身份	√
userName	string	用户名	
passWord	string	登录密码	
PSType	int	表明用户权激励类型	
incentive	double	激励值	
participate	Int	参与活跃度	
loginTimes	Int	登录次数	
onlineTime	long	在线时间	

用户信息表 UserInfo 如表 4-6 所示，主要用于存储用户的个人信息以及相关的激励信息，以用户 ID 作为主键。用户注册时会在本表中增添一行记录；用户登录时则通过 userId 对记录进行查询；用户上传数据会对本表进行更新，主要更新 participate 信息；用户注销操作则会使登录此时和在线时间信息发生更新。另外，incentive 和 participate 信息会对激励任务的分配产生影响，因此激励模块进行激励任务的发布时，也会对用户信息表进行查询。

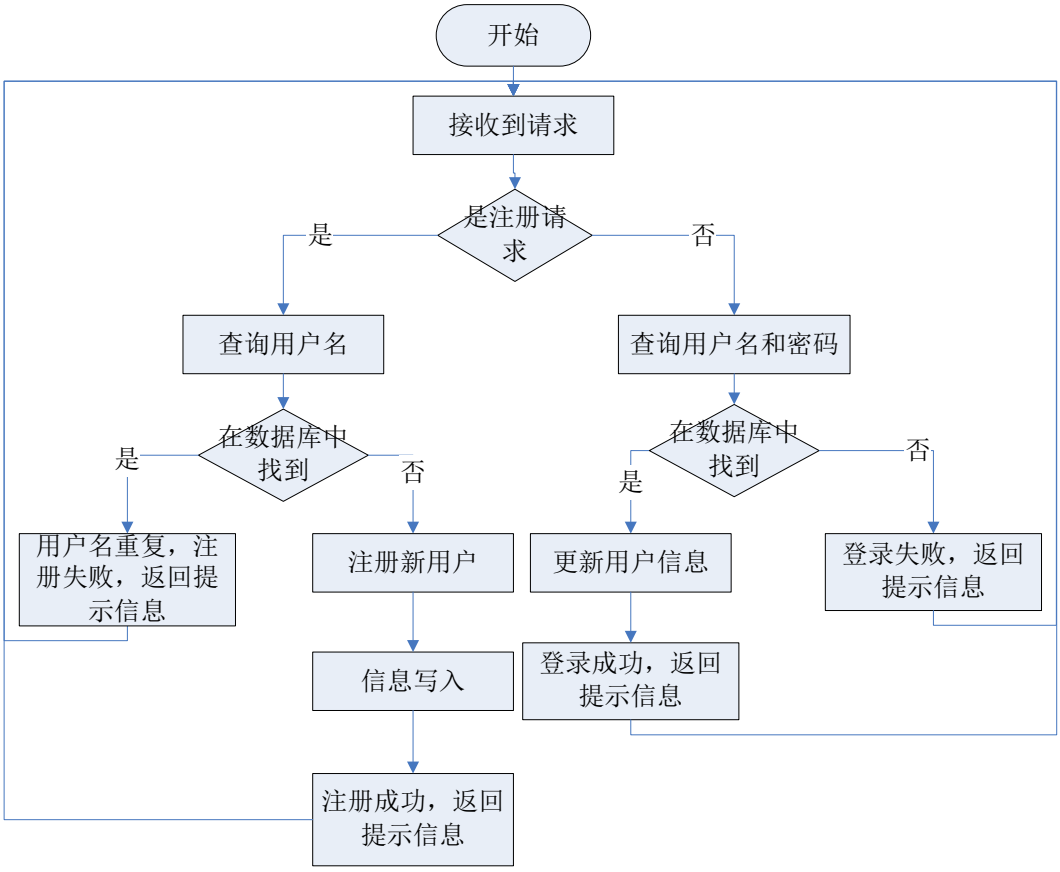


图 4-8 注册登录流程图

4.5.2 流程设计

用户注册登录的流程如图 4-8 所示。注册和登录请求都是由用户管理子模块进行处理的，首先需要判断接收到的是何种请求。接收到注册请求时，本模块会在用户信息表中查找用户名，确定用户是否注册过，未注册的用户名可以注册成功，并将用户相关信息写入数据库。接收到登录请求时，需要根据用户名和密码查看用户信息表；当查找成功时，证明用户可以登录，此时触发用户管理模块的用户信息更新，登录完成。

4.6 任务推送子模块设计

4.6.1 数据库设计

表 4-7 原始激励信息表 Originalincentive

属性	类型	描述	主键
incentiveId	int	自增	√
dataType	int	规定激励所需数据类型	
Lon	double	纬度	
Lat	double	经度	
Time	string	发出本次任务的时间	
payment	double	对于本次任务的激励	
Flag	int	判断本次任务是否已经推送	

表 4-8 用户激励信息表 Userincentive

属性	类型	描述	主键
Id	int	自增	√
userName	string	用户名	
incentiveID	int	激励 ID	
Time	String	响应激励时间	
Pay	double	约定激励	
willing	Int	是否愿意参与	
Flag	int	是否已经响应	

任务推送功能主要和激励模块协同工作，为本模块设计的数据库表也被激励模块和任务推送模块共同访问。

原始任务信息表如表 4-7 所示。它主要存储了本次激励任务所需的数据类型，该数据所在的位置，以及激励和本次任务的推送情况。激励模块产生的新的激励任务存入该表中，然后调用任务推送模块。任务推送模块根据任务编号从表中取出需要推送的任务，写成约定格式的字符串，将任务推送出去，同时对表中的 time 和 flag 属性进行更新，表示该任务已经推送。

用户激励信息表如表 4-8 所示，主要记录对于发送激励用户对于本次激励的响应情况。任务被推送到客户端后，用户会做出是否接受任务的响应，该响应被存入此表中。

通过该表可以查询用户响应激励的历史信息，以及对每次任务的接受情况。当下一次需要发送激励任务时，这些信息也将作为激励模块计算激励和选择目标用户的参考。

4.6.2 流程设计

任务推送模块，需要将激励模块产生的任务推送到客户端。对本系统来说，任务推送不仅能够实现一般的广播，也就是全体推送功能，还需要实现组播，即给指定的一些用户发送任务。

本论文使用云推送 API 来实现信息的推送。其提供两种消息的推送机制：通知和透传消息。通知即展示在系统通知栏的消息，透传则将需要传送的消息传送到目的客户端，同时保证传输的质量即可，而不对传输的消息进行处理。

在如今开放的 API 中，实现了单播和广播的功能，云推送平台开发的接口中实现的组播，是需要对客户端打标签的方式实现，这种标签是由管理员进行配置，静态确定的。但在本论文研究中，每次的激励任务都会有不同的目标用户集合，需要实现动态的组播推送，因此对实现方案进行了重新的设计。

分组推送流程如图 4-9 所示。首先任务推送模块被激励模块调用，说明产生了新的任务。任务推送模块从数据库中读取任务内容和任务目标，判断该任务是否是全体推送。如果该任务是全体推送，则直接调用全体推送接口，为所有用户下发广播消息并等待用户响应。接受该任务的用户，任务推送模块将会记录其响应，更新数据库并等待该用户上传数据。

如果任务目标是一组动态确定的用户，则任务推送模块会将激励任务信息记录而不马上推送，并将有激励任务这一消息通知数据接收和查询模块。当接下来有客户端发出请求时，任务推送模块会首先查询数据库确认该客户端是否推送的目标用户。如果进行查询时未查询到该用户名，则正常返回请求应答，不会进行其他操作；若查询到用户名，说明本用户是目标用户，可以接收激励。然后，将数据库中的激励任务信息取出，将其写入对客户端请求的应答中。客户端模块将对接收到的应答进行解析，发现其中有附加的激励任务信息时，在 UI 界面显示给用户。用户接受或拒绝任务之后，任务推送模块记录用户的响应，等待接收任务的用户上传的数据。

当任务推送模块接收到用户上传的任务数据后，会对其进行保存，并写入数据库。同时查询数据库，确认接收该任务的用户是否已经全部上传了数据。如果不是，则继续等待用户上传数据；如果已经全部上传数据，则更新数据库，表明本次任务已经结束，并结束本次会话。

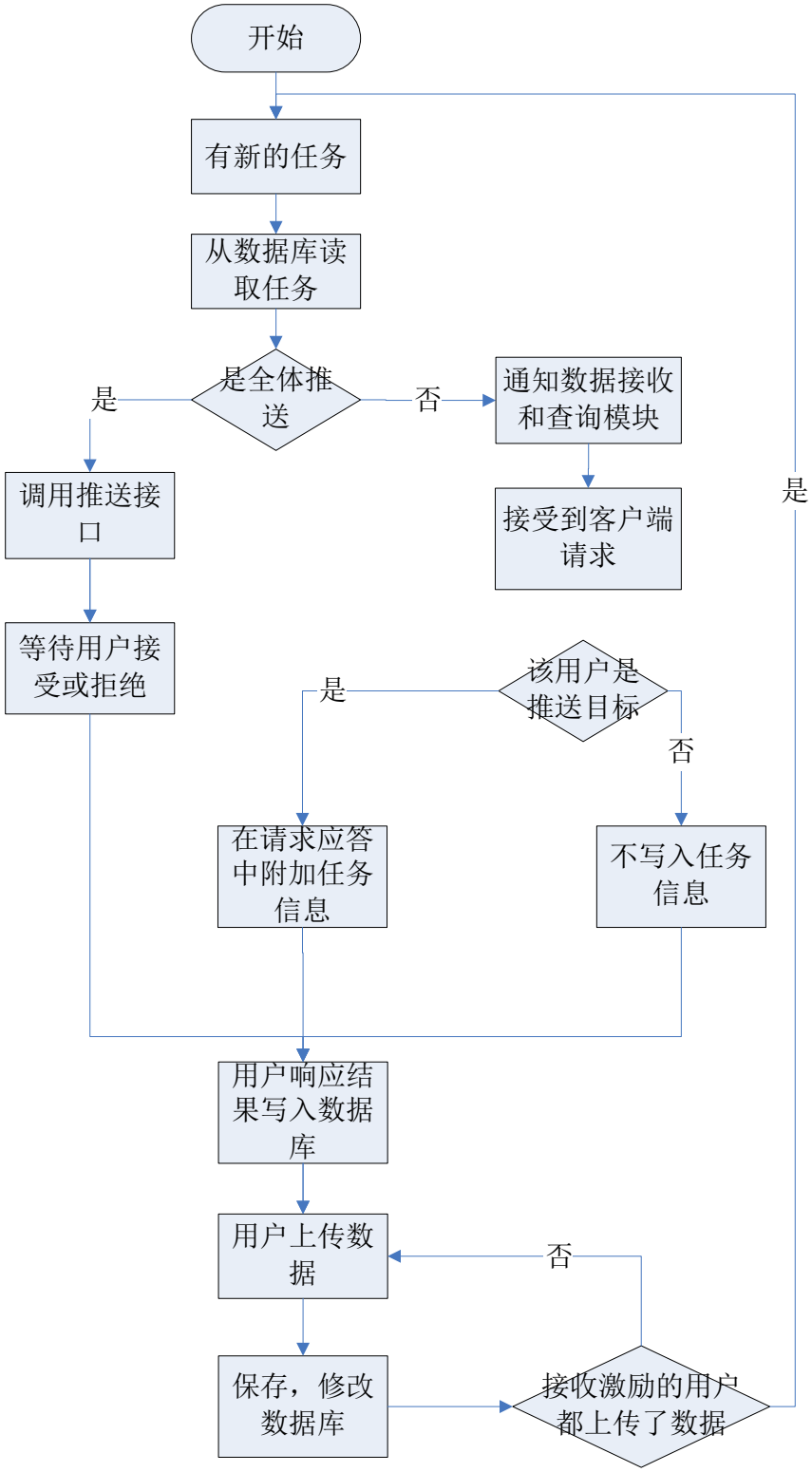


图 4-9 任务推送流程图

4.6.3 功能实现

- 进程间通信

任务推送功能主要是为激励模块服务。在服务器端，网络接口模块和激励模块各种

作为独立的进程进行各自的功能。因此需要采用进程间通信的方式，使激励模块能够调用网络接口模块的推送功能。

采用 Spring 框架下的 HTTP invoker 的方式通过远程调用实现进程间通信。进程间通信不仅使激励模块可以调用任务推送子模块，而且使激励模块能和轨迹分析模块、数据融合模块之间进行信息的交互，从而产生激励任务。

对于任务推送模块而言，激励模块扮演了客户端的角色，任务推送模块成为服务端，将其接口提供给激励模块调用。任务推送模块作为推送功能的提供方，需要将其服务部署到 web 环境中，供激励模块模块进行调用和访问。

在第二章已经提到过，服务的配置需要导出 HTTP invoker 服务。为了导出该服务，需要使用 HttpInvokerServiceExporter 类，这是一个 Spring 的 MVC 控制器，我们首先需要在部署描述符文件（web.xml）中声明 DispatcherServlet，它是一个前段控制器，其任务是将请求的 URL 通过处理器映射来决策其下一步的工作内容，之后配置 <servlet-mapping>，表明所提供的服务对应的访问路径。最后，我们在 Spring 的上下文文件中建立一个 URL 的处理器映射，映射 HTTP URL 到对应的服务上面。通过映射配置定义了服务访问的路径。

激励模块对任务推送模块发起调用请求时，在 HTTP invoker 的处理过程中处于客户端的角色。为了访问基于 HTTP Invoker 的远程服务，客户端必须在 Spring 的应用上下文中配置一个 HttpInvokerProxyFactoryBean 的 Bean 来代理它。HttpInvokerProxyFactoryBean 是一个代理工厂 Bean，用于生成一个代理，该代理使用 Spring 特有的基于 HTTP 协议进行远程通信。 在配置完成后，就能像编写调用本地函数一样对远程服务进行调用了。

- 消息推送

服务器主要利用广播方法来进行消息推送。

```
//建立广播类型
PushBroadcastMessageRequest request = new PushBroadcastMessageRequest();
request.setDeviceType(3); // device_type => 1: web 2: pc 3:android4:ios 5:wp
String messageFormat = "{\"title\":\"%s\",\"description\":\"%s\"}";//定制内容格式
String[] titleContent = new String[2];
titleContent[0] = "我的标题";
titleContent[1] = "通知内容";
request.setMessage(String.format(messageFormat, titleContent));
//调用推送接口
PushBroadcastMessageResponse response =
channelClient.pushBroadcastMessage(request);
```

4-10 广播推送实现

广播推送的实现方式主要代码如图 4-10 所示，建立广播之后，确认广播类型，然后将从数据库取出的激励任务根据定义的格式写入推送消息中。在推送完成后，response.getSuccessAmount()回调函数可以检测推送成功的个数。

单播推送对应的类为 PushUnicastMessageRequest，与广播推送不同的是，在设置单播推送方法时需要绑定用户的 userid，从而确认是对哪一个用户进行推送。因此需要访问数据库的用户信息表，确认用户名对应的客户端，然后向该客户端推送。

4.7 本章小结

本章首先介绍了网络接口模块的内部结构和子模块的划分。网络接口模块被分为数据接收、数据查询、用户管理和任务推送四个子模块。然后针对每个子模块的设计进行了详细的介绍，包括子模块的流程和模块中涉及的数据库表的设计。最后介绍了任务推送模块中用到的进程间通信的实现方式。模块功能的测试，将在第五章进行详细介绍。

第五章 网络接口模块的测试

5.1 测试环境

参与式感知平台的基本测试环境拓扑如图 5-1 所示。

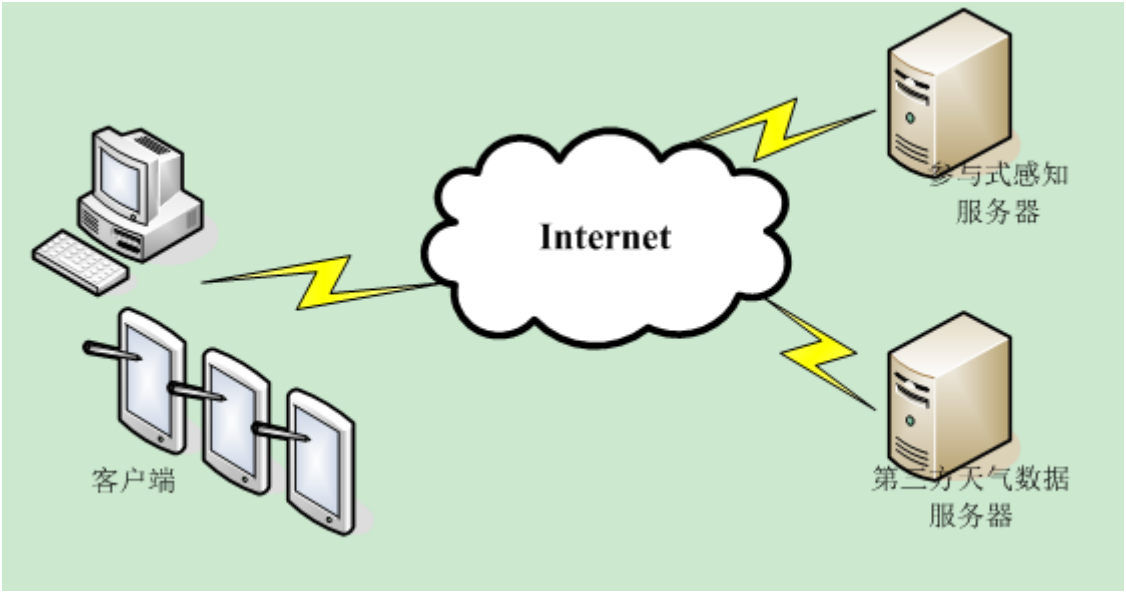


图 5-1 测试环境拓扑

网络接口模块主要作为客户端和服务器之间的通信接口，在测试过程中需要协调客户端、服务器和网站的功能。其中参与式感知服务器使用 ubuntu 系统，并且安装有 HBase 数据库；有多个客户端可以通过无线网络与服务器通信，客户端采用的是三星 Galaxy 系列的产品，应用程序适配 android4.3 以上系统版本；web 端采用 chrome 或者 IE 浏览器进行网页访问；第三方天气数据服务器可以通过互联网访问，爬取天气数据。

本章节的测试使用黑盒测试，黑盒测试又称为功能性测试，是由功能驱动或数据驱动的测试。在已知系统拥有功能的基础上，测试各功能是否能够正常使用。黑盒测试主要基于系统的需求分析和设计，在程序接口进行测试，它只检查程序功能是否按照需求规格说明书的规定正常使用，程序是否能适当地接收输入数据而产生正确的输出信息。本章使用黑盒测试主要希望能发现：功能不正确或遗漏；输入输出错误；数据库访问错误；性能错误。

5.2 网络接口模块功能测试

网络接口模块的主要功能包括数据接收、数据查询、任务推送。分别测试各个子功

能的实现是否达到了要求，和其他模块之间的接口是否正确实现。

5.2.1 数据接收功能测试

数据接收功能是网络接口模块的主要功能，主要负责接收客户端上传的照片和感知数据。对数据接收功能的测试目的是测试客户端上传的文件和数据能否成功接收，能否正确解析文件中的 EXIF 信息，将文件和数据保存并正确写入数据库中。测试分为上传文件测试和上传数据测试。

5.2.1.1 上传文件测试

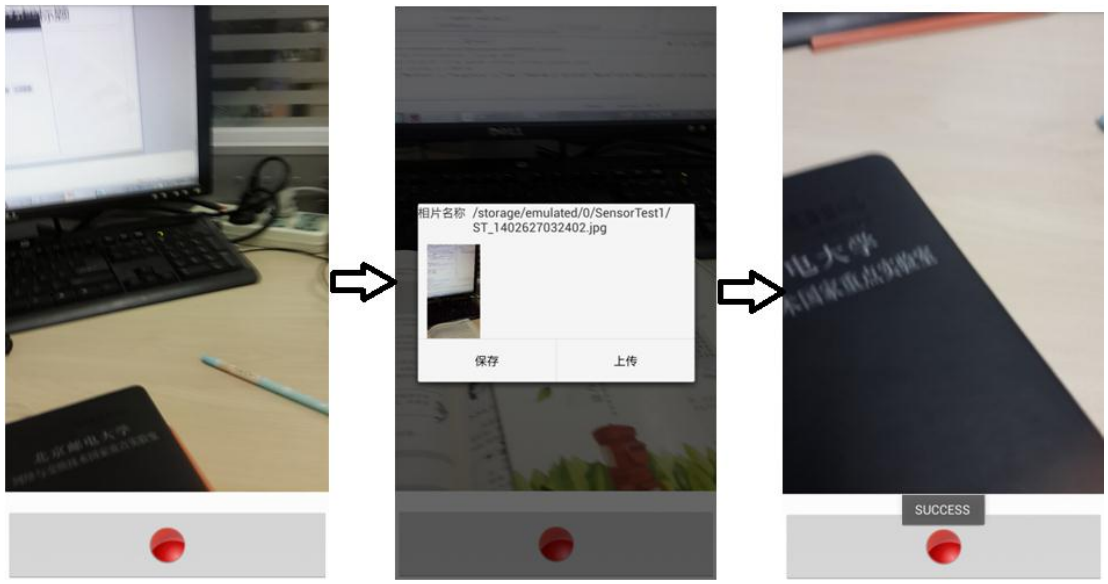
上传文件的测试用例如下：

表 5 -1 数据接收功能测试用例 1

测试用例编号	recv001
测试目标	客户端上传的文件是否能被成功接收，文件中的 EXIF 信息是否正确解析，文件是否进行了压缩和存储，是否正确写入数据库
测试环境	上传文件的地址： http://10.108.107.92:8080/uploadFile/fileServlet
测试步骤	1.在应用的 PM collection 界面点击右上角相机图标，进入拍照界面。 2.拍摄照片，点击“上传”。 3.界面弹出“success”提示。 4.在服务器查看日志文件
预期结果	1．查看日志文件，发现多了一条上传文件的记录； 2．查看服务器的/usr/local/apache-tomcat-7.0.54/webapps/ROOT 文件夹，发现多了对应的照片文件； 3．下载该文件打开，能成功打开，并且是刚上传的照片文件。
实际结果	与预期结果一致
测试结果	通过

上述用例测试了客户端的文件上传功能，测试结果是上传文件功能可以正常工作。图 5-1(a)所示为客户端测试过程中的界面截图，用户成功上传文件后界面会弹出“success”的提示信息。图 5-2(b)所示为上传文件成功后服务器日志文件中的情况，可以看出新上传了一个文件，同时网络接口模块正确解析了文件的 EXIF 信息中包含的标签信息，并

对文件进行了压缩。



(a)客户端上传照片界面

```
2014-09-19 10:17:29,029 [http-bio-8080-exec-144] photo tag: {"pic":{"Time":1411093062000,"Longi
2014-09-19 10:17:29,029 [http-bio-8080-exec-144] filename: zzy_1411093049080.jpg
2014-09-19 10:17:29,029 [http-bio-8080-exec-144] compress result: ok
```

(b)服务器日志文件

图 5-1 客户端上传文件成功

5.2.1.2 上传数据测试

客户端上传非文件的感知数据的测试用例如表 5-2：

表 5 -2 数据接收功能测试用例 2

测试用例编号	recv002
测试目标	客户端上传的（非文件）数据是否能被成功接收，是否被正确解析，是否正确写入数据库
测试环境	上传数据的地址： http://10.108.107.92:8080/uploadFile/fileServlet
测试步骤	1.在应用的“测试”界面点击“上传数据”按钮。 2.界面弹出“success”提示。 3.在服务器查看日志文件
预期结果	查看日志文件，发现多了一条上传数据的记录。
实际结果	与预期结果一致
测试结果	通过

上述用例测试了客户端的感知数据上传功能。图 5-2(a)所示为客户端测试过程中的界面截图，用户成功上传感知数据后界面会弹出“success”的提示信息。图 5-2(b)所示

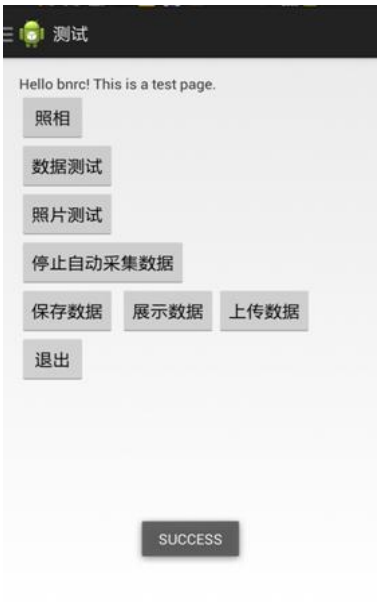
为上传文件成功后服务器日志文件中的情况，可以看出新上传了一个 JSON 字符串，即感知数据。

通过上述测试用例，可以验证网络接口模块的数据接收功能正确工作，与客户端和数据库的接口也按照设计正确实现。客户端进行文件和数据上传时，网络接口模块能正确接收文件，对文件进行解析，并保存到数据库中。

5.2.2 数据查询功能测试

客户端的照片墙、PM2.5 预测功能都需要向服务器发送查询请求，由网络接口模块负责向服务器端其他模块请求数据，再将结果返回客户端。对数据查询功能测试的目的是验证对查询请求能否生成约定格式的查询结果，对一些异常情况是否能够进行处理。

在本小节的测试中使用了测试工具 Postman。它是 chrome 浏览器的一款插件，能够发送任何类型的 HTTP requests (GET, HEAD, POST, PUT)，附带任何数量的参数。通过这个插件可以模拟 post 或者 get 请求，还可以看到请求返回的结果，不论是 JSON 格式还是 XML 格式。同时。因此，我们使用 Postman 来代替客户端发出查询请求，用于测试工作。



(a)客户端上传数据界面

```
2014-09-19 10:33:18,018 [http-bio-8080-exec-146] recv json: %7B%22data%22%3A%5B%7B%22NetState%22
0%2C%22Latitude%22%3A39.963684%2C%22Longitude%22%3A116.35728%2C%22Light%22%3A328%2C%22BatterySta
```

(b)服务器日志文件

图 5-2 客户端上传数据成功

5.2.2.1 照片查询测试

表 5-3 所示为照片查询功能测试用例：

表 5 -3 数据查询功能测试用例 1

测试用例编号	query001
测试目标	测试网络接口模块能否接收到客户端的查询照片请求，进行解析并返回正确的照片列表。
测试环境	查询照片的地址： http://10.108.107.92:8080/queryDB/getPhotoServlet 测试工具：postman

测试步骤	1.使用客户端在不同的时间拍摄了一定量的照片并上传
	2.使用 postman，向上述地址发送 post 请求，请求内容为 json 格式的字符串
	upload={
	“request_type”:”photo_list”,
	"request_maxnum": "50",
	“begin_time”:”1404789038942”
	}
	发送请求。查看返回的信息。
	3.begin_time 为 ms 形式的时间，选择不同的时间进行发送，测试根据时间进行查询的功能是否正常。

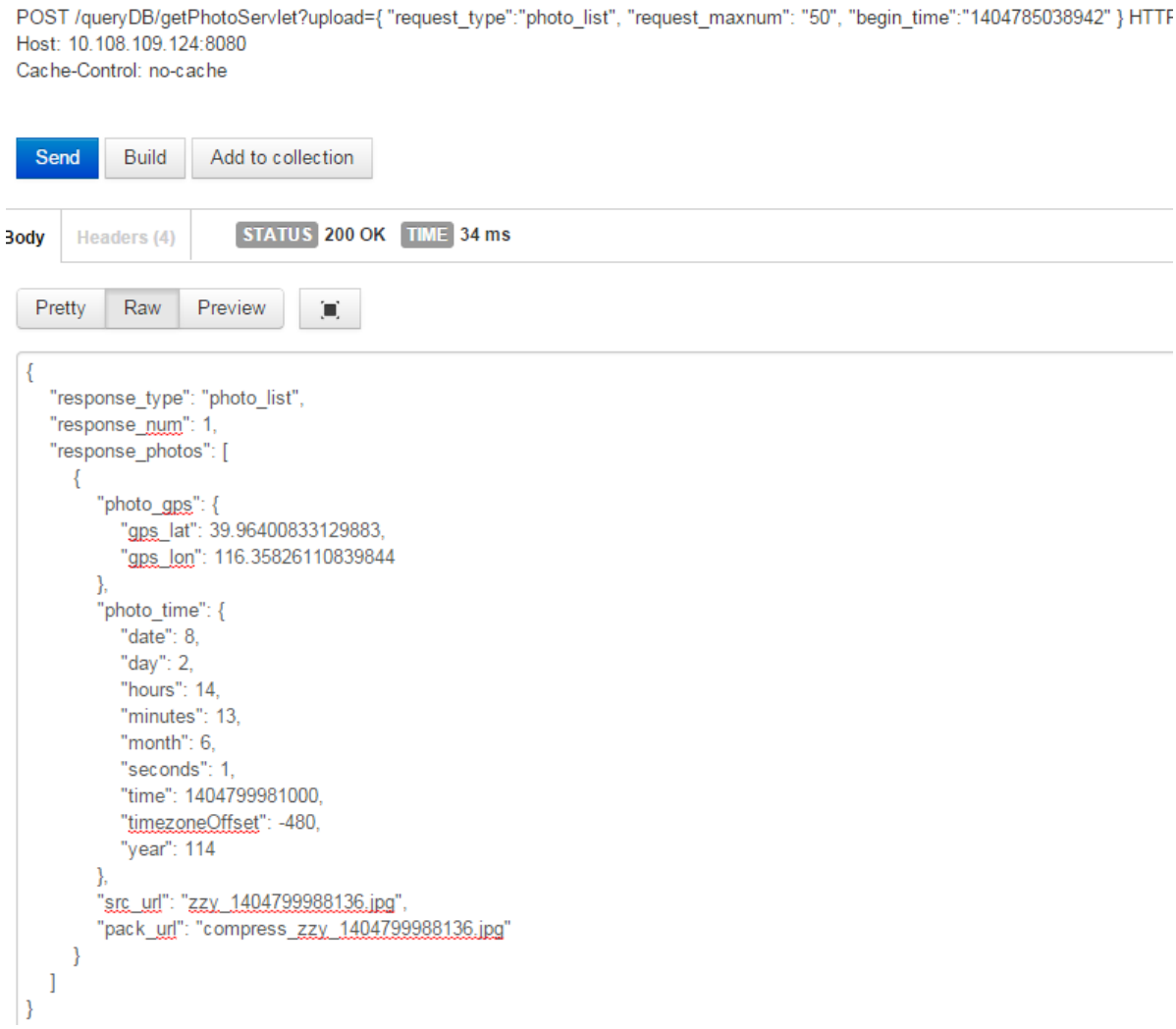
预期结果	有符合条件的照片时，返回规定格式 json 字符串，包括照片数量，照片拍摄的时间地点，以及照片在服务器的 URL。
	{
	"response_type": "photo_list",
	"response_num": 1,
	"response_photos": [
	{
	"photo_gps": {
	"gps_lat": 39.96400833129883,
	"gps_lon": 116.35826110839844
	},
	"photo_time": {
	"date": 8,
	"day": 2,

（续上表）

预期结果	"hours": 14, "minutes": 13, "month": 6, "seconds": 1, "time": 1404799981000, "timezoneOffset": -480, "year": 114 }, "src_url": "zzy_1404799988136.jpg", "pack_url": "compress_zzy_1404799988136.jpg" }] } 查询的时间不同时，会返回从查询时刻开始的不同数量的照片信息。
实际结果	与预期结果一致
测试结果	通过

上述用例测试了网络接口模块的查询照片功能。图 5-3 所示为使用 postman 进行测试的结果界面。图片上半部分显示发出的 post 请求的内容，下半部分为网络接口模块对该请求的响应。当没有照片时将返回如图 5-3(a)所示字符串，其中 response_num 是 0，否则所查询的时间有合适的照片时将返回如图 5-3(b)所示 JSON 字符串，表示没有查到符合条件的结果。

(a)没有符合条件的照片



(b)有符合条件的照片

图 5-3 客户端查询照片结果

5.2.2.2 天气信息查询

天气信息查询功能主要为客户端的 PM2.5 分析功能提供查询接口，根据客户端请求中的查询类型字段确定对哪一信息进行查询，能对 PM2.5、天气情况、相机模型等数据进行查询。

表 5-4 数据查询功能测试用例 2

测试用例编号	query002
测试目标	测试网络接口模块能否接收到客户端的查询站点 PM 请求，进行解析并返回正确的 PM 值。
测试环境	查询的地址： http://10.108.107.92:8080/queryDB/getWeather
测试步骤	1. 服务器上的爬取站点天气信息的程序定时整点运行，将爬取的

（续上表）

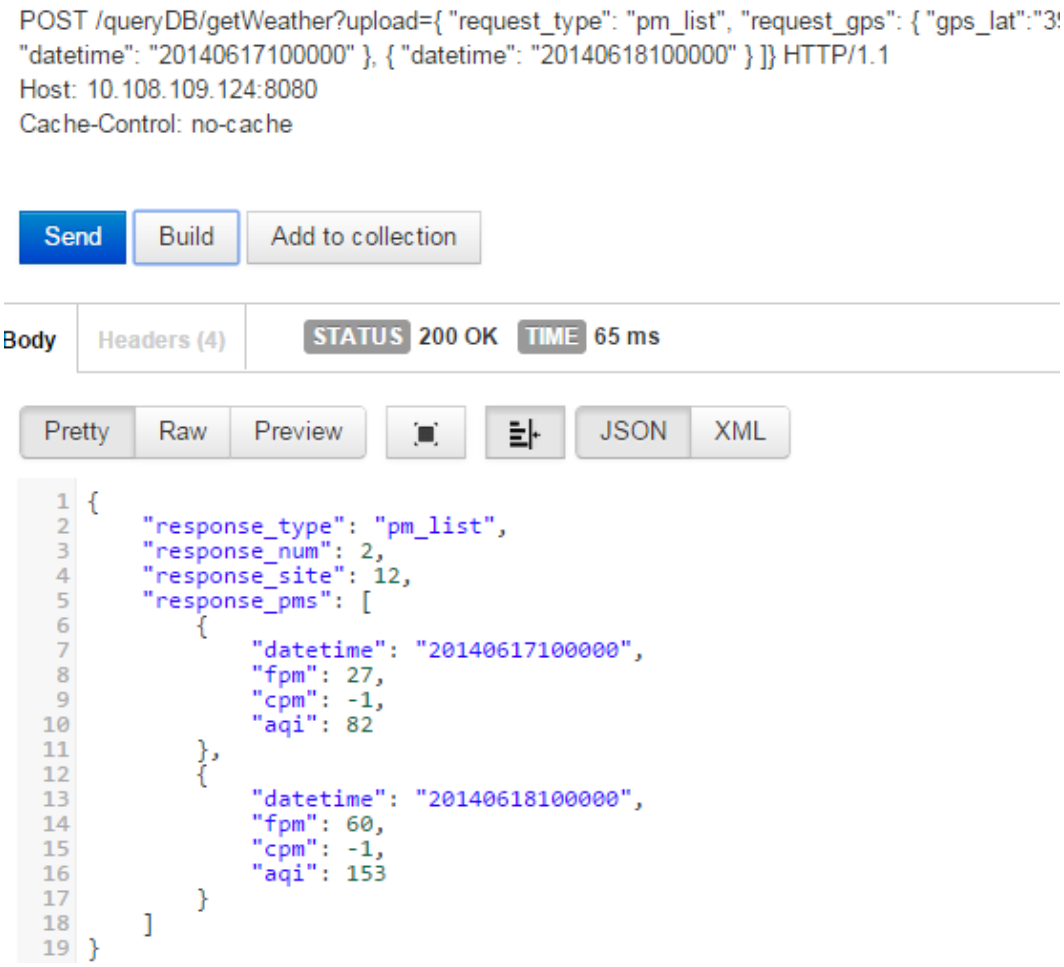
测试步骤	信息存入数据库。
	2. 使用 postman，向上述地址发送 post 请求，请求内容为 json 格式的字符串 upload={
	"request_type": "pm_list",
	"request_gps":
	{ "gps_lat":"39.8549379885","gps_log":"116.4001464844"},
	"request_num": "2",
	"request_datetimes": [
	{
	"datetime": "20140617100000"
	},
	{
	"datetime": "20140618100000"
	}
	]
	}发送请求,查看返回的信息。

预期结果	返回如下 json 格式的字符串，即请求的站点的 pm 信息。 { "response_type": "pm_list", "response_num": 2, "response_site": 12, "response_pms": [{ "datetime": "20140617100000", "fpm": 27, "cpm": -1, "aqi": 82 }, { "datetime": "20140618100000", "fpm": 60, "cpm": -1, "aqi": 153

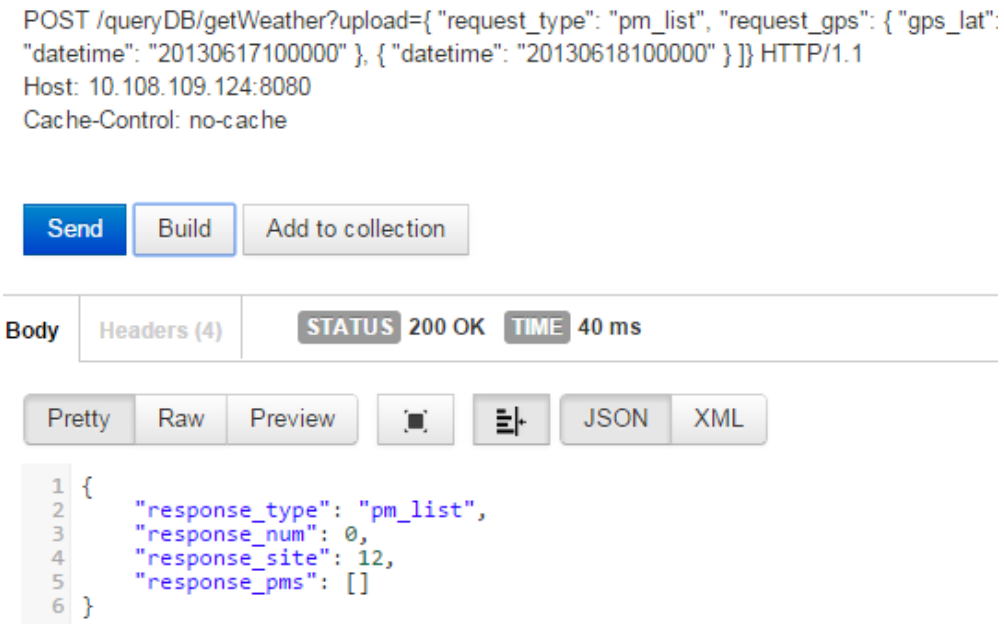
（续上表）

预期结果	}] }
实际结果	与预期结果一致
测试结果	通过

上述测试用例主要测试数据查询功能中的 PM2.5 查询，网络接口模块在接收到查询 PM2.5 的请求之后，将会调用服务器的 PM 信息爬取模块提供的接口，得到请求中的经纬度最近的站点。然后通过该站点到数据库中查询。如果符合条件的结果存在，则返回如图 5-4(a)所示格式的结果，否则如果查询请求中的时间点的数据不存在数据库中，将返回如图 5-4(b)所示的结果，其中 response_num 为 0，表示没有所查询时间点的 PM2.5 信息。



(a) 查询时间的 PM 记录存在



(b)查询时间的 PM 记录不存在

图 5-4 查询某时某地的 PM2.5 信息

天气数据查询与 PM2.5 查询功能都是为了客户端的 PM2.5 分析模块服务的。其数据接口和工作流程也相似。如果符合条件的结果存在，则返回如图 5-5(a)所示格式的字符串；当所查询的数据不存在时，也会产生一个 response_num 为 0 的应答，如图 5-5（b）所示。

表 5 -5 数据查询功能测试用例 3

测试用例编号	query003
测试目标	测试网络接口模块能否接收到客户端的查询城市天气请求，进行解析并返回正确的天气信息。
测试环境	查询的地址： http://10.108.107.92:8080/queryDB/getWeather

测试步骤	1. 服务器上的爬取站点天气信息的程序定时整点运行，将爬取的信息存入数据库。
	2. 使用 postman，向上述地址发送 post 请求，请求内容为 json 格式的字符串 upload={
	"request_type": "weather_list",
	"request_gps":
	{ "gps_lat":"39.8549379885","gps_lon":"116.4001464844"},
	"request_num": "2",
	"request_datetimes": [
	{

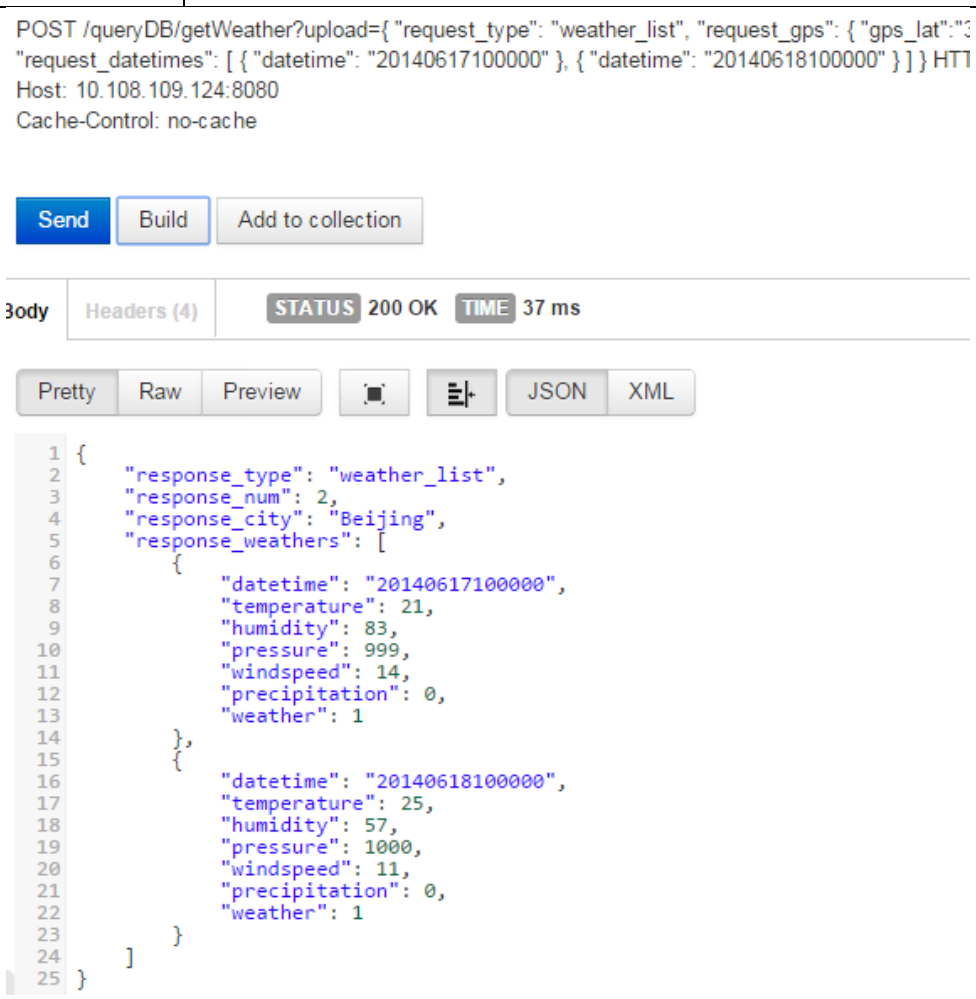
（续上表）

测试步骤	"datetime": "20140617100000"
	},
	{
	"datetime": "20140618100000"
	}
	]
	}发送请求,查看返回的信息。

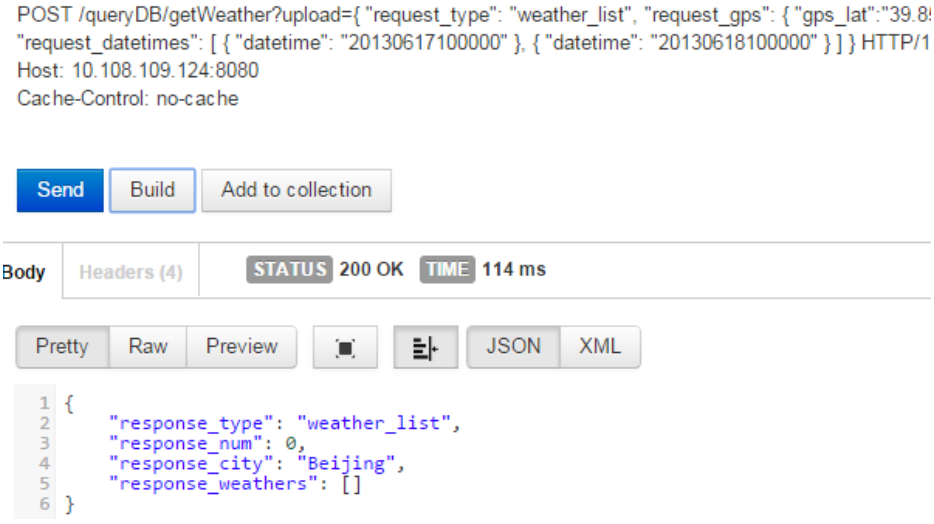
预期结果	返回如下 json 格式的字符串，即请求的城市的天气信息
	{
	"response_type": "weather_list",
	"response_num": 2,
	"response_ city": "Beijing",
	"response_ weahters": [
	{
	"datetime": "20140617100000",
	"temperature": 21,
	"humidity": 83,
	"pressure": 999,
	"windspeed": 14,
	"precipitation": 0,
	"weather": 1
	},
	{
	"datetime": "2014061810000",
	"temperature": 25,
	"humidity": 57,
	"pressure": 1000,
	"windspeed": 11,
	"precipitation": 0,
	"weather": 1
	}
	]
	}

（续上表）

实际结果	与预期一致
测试结果	通过



（a）查询成功



(b)查询失败，没有符合条件的信息

图 5-5 查询某时某地天气信息

表 5-6 数据查询功能测试用例 4

测试用例编号	query004
测试目标	测试网络接口模块能否接收到客户端的查询相机模型 请求，进行解析并返回正确的相机模型信息。
测试环境	查询的地址： http://10.108.107.92:8080/queryDB/getWeather
测试步骤	在数据库中存入了手机型号的相关信息 使用 postman，向上述地址发送 post 请求，请求内容为 json 格式的字符串 upload={ "request_type": "crf_model", "request_ phoneid": "35243003" }发送请求，查看返回信息。

预期结果	返回如下格式的字符串
	{
	" response _type": "crf_model",
	"response_ modelid": "12",
	"response_ model": "35243003"
	}

实际结果	与预期结果一致
测试结果	通过

上述用例测试了数据查询功能能否正确查询相机模型。当服务器数据库中未保存有该相机模型数据时，将返回如图 5-6 所示的字符串。查询成功时，即返回与预期结果相同格式的字符串。

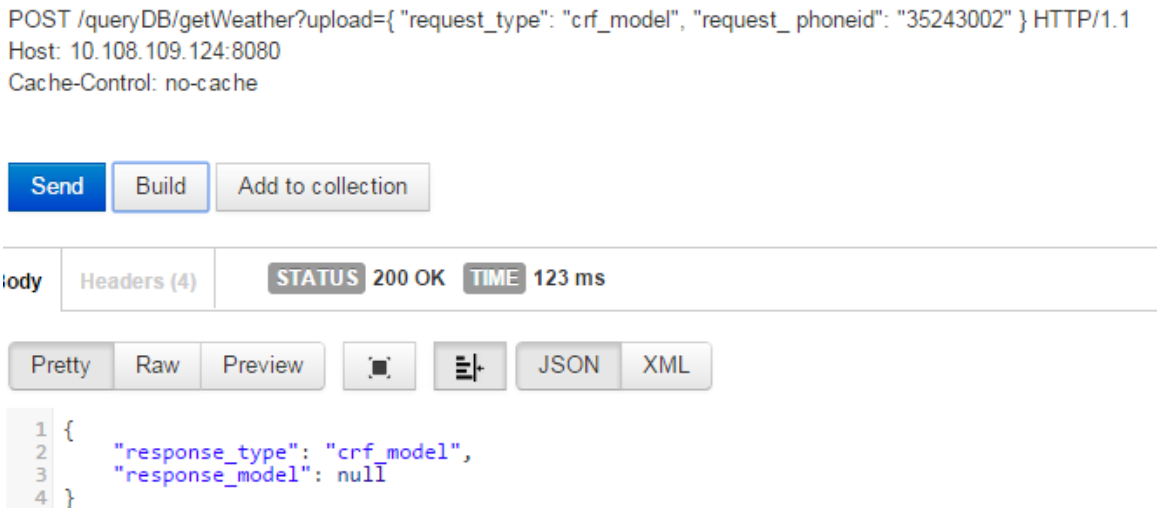


图 5-6 查询相机模型失败

通过本小节的测试，验证了网络接口模块的数据查询功能按照需求和设计实现。网络接口模块与 PM 信息爬取模块，数据库模块，客户端模块之间的数据接口和数据交互流程都是正确的。

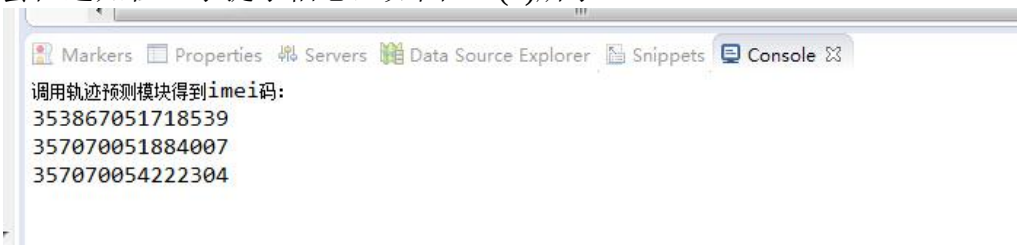
5.2.3 任务推送功能测试

任务推送功能是为了服务器端的激励模块能够发布激励任务而设计实现的。任务推送功能是由激励模块调用的

表 5-7 任务推送功能测试用例 1

测试用例编号	push001
测试目标	测试网络接口模块能否被激励模块成功调用，将激励模块产生的任务推送给指定用户。
测试环境	激励模块、轨迹分析模块正常工作，其中轨迹分析模块的功能可以模拟实现。
测试步骤	1. 激励模块产生了需要获得数据的地点的经纬度信息。 2. 激励模块调用轨迹分析模块，预测将到达指定地点的用户列表。 3. 获得用户名列表之后，调用推送模块进行推送。
预期结果	步骤 2 中，激励模块能成功获得用户名列表。 步骤 3 中，客户端接收到所推送的信息。
实际结果	与预期结果一致
测试结果	通过

上述用例测试了网络接口模块的任务推送功能，包括网络接口模块与激励模块、客户端之间的接口，激励模块与轨迹预测模块的接口。如图 5-7(a)所示，服务器端激励模块成功调用轨迹分析模块，获得相应的 imei 码，即用户 ID 列表。客户端成功接收到推送信息时会在通知栏显示提示信息，如图 5-7(b)所示。



(a)获取用户 id 列表

（b）客户端通知栏显示通知

图 5-7 推送成功

5.3 网络接口模块性能测试

为了保证网络接口模块能够正常稳定工作，不仅需要测试其功能是否正确实现，还需要对其性能进行测试。测试内容主要有：响应时间测试、并发性能测试、稳定性测试。

本小节的测试中使用了开源测试工具 ab（Apache Bench），它是一个简单的性能测试工具，可以模拟 HTTP 并发请求，向指定的地址发送各种类型的 HTTP 请求，并能由测试者设定并发度。

5.3.1 响应时间

响应时间包括网络连接时间、文件传输时间和事务处理时间等，是决定用户体验的重要因素。主要对网络接口模块上传数据和下载数据功能的响应时间进行了测试。测试用例如下：

表 5-8 响应时间测试用例

测试用例编号	perf001
测试目标	测试单个用户上传文件和下载文件的响应时间
测试环境	上传数据的地址： http://10.108.107.92:8080/uploadFile/fileServlet
测试步骤	1.使用客户端拍摄的多张不同大小的照片，进行上传，查看客户端和服务器的日志文件，获得响应时间。 2.打开客户端照片墙界面，选中几张不同的照片放大查看，客户端会下载选中的照片。查看服务器日志文件，获得响应时间。

响应时间测试主要测试了在系统只有一个用户进行上传或下载时，网络接口模块的响应时间。下表所示为测试结果。

表 5-9 响应时间测试结果

文件大小 (单位：M)	0.92	1.01	1.24	1.56	1.73
上传文件响应时间 (单位：ms)	497	516	524	547	560
下载文件响应时间 (单位：ms)	381	388	394	406	412

从表中可以看出，上传文件时网络接口模块的响应时间与文件大小有关，较大的文件所需要的网络传输时间和文件解析、压缩、存储时间更长，会影响响应时间。下载文件时，客户端已知所需文件的 URL，将直接请求该文件，因此减少了文件分析处理的时间，所以同样的文件下载所需要的时间会更短。从平均情况来看，使用客户端拍摄的照片大小在 90%在 1M-1.5M，平均响应时间也比较接近这一区间。

5.3.2 并发度

并发度是是指在同一时刻同时访问服务器的客户端数量。与客户端的通信主要由网络接口模块负责，其中文件并发上传是并发性能的主要指标。因此并发度测试需要测试网络接口模块处理并发文件上传请求的性能。

表 5-10 并发处理性能测试用例 1

测试用例编号	perf002
测试目标	测试文件接收的并发处理能力
测试环境	上传数据的地址： http://10.108.107.92:8080/uploadFile/fileServlet 测试工具：ab
测试步骤	1.选择本应用客户端拍摄的 1.24M 的一张照片，使用 ab 工具，发送 post 请求上传文件，进行并发性测试。 2.分别测试并发度 1，10，30，50，70，90，110 的情况下，服务器对上传文件的处理能力。

图 5-8 所示为使用 ab 工具进行并发性能测试的结果。在并发度为 1，即只有一个用户时，单个请求的处理时间为 520ms，与上一小节测试的响应时间结果相符。随着并发度的提高，单个请求的处理时间稍微变长，在并发度为 110 时，达到了系统并发处理能力的边界值，单个请求处理时间变长。该时间为所有请求的平均处理时间，由于此时并发度达到了边界值，导致部分请求需要排队，不能马上处理，因此平均处理时间变长。

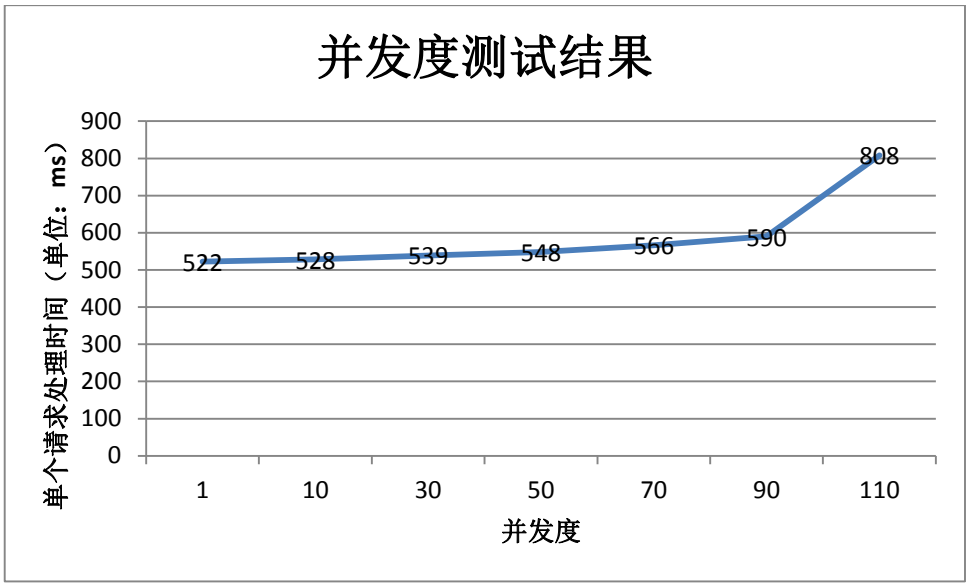


图 5-8 并发测试结果

5.3.3 稳定性

使用测试工具进行的并发性能测试，与使用客户端上传文件相比，在网络传输情况、客户端处理能力等方面都可能存在着差距。使用手机客户端对稳定性进行了测试，稳定性测试是为了验证在长时间的连续工作中，网络接口模块能稳定的进行数据处理，不发生崩溃的情况

表 5-11 稳定性测试用例 1

测试用例编号	perf003
测试目标	测试使用客户端上传文件时，客户端和网络接口模块的稳定性。
测试环境	上传数据的地址： http://10.108.107.92:8080/uploadFile/fileServlet 测试工具：三星 Galaxy 手机客户端 10 台， android4.3 以上
测试步骤	1. 打开客户端的测试界面，连续拍摄 20 张照片。 2. 10 台客户端尽量同时点击上传文件。 3. 查看服务器日志文件和客户端的响应，查看系统的稳定性。 4. 在一天的不同时间分别上传文件或查看照片墙，查看系统的稳定性。

从测试结果来看，步骤 3 中的 200 张照片都成功的被服务器接收，在传输过程中未发生服务器或客户端的崩溃超时等现象，说明系统在短时间内连续处理大量请求具有基

本的稳定性。步骤 4 中，10 位用户分别在一天的不同时间多次上传了文件并点击照片墙查看，未发生失败的情况，说明系统运行较为稳定。

5.4 本章小结

本章详细描述参与式感知系统的服务器端网络接口模块的测试工作，首先介绍系统的测试环境，然后介绍了该模块的功能性测试，接着对与网络接口模块有数据交互的模块进行了集成测试。最后，针对系统的高并发需求，对网络接口模块的处理能力进行了性能测试。

第六章 结束语

6.1 全文总结

本论文主要讨论了参与式感知系统平台的设计方案，并设计实现了平台中网络接口模块的功能，并进行了测试验证。参与式感知平台能够通过参与者收集到的传感器数据，对数据进行存储、分析、可视化展示和共享。

本论文中参与式感知平台的设计，强调了平台的可扩展性，以及对大量数据上传存储的支持，并且提供了数据可视化功能。本平台在后台与前台设计之初，对可扩展功能（数据采集、算法验证）采用插件式管理，便于后期对系统扩展工作的进行。其次本平台采用分布式数据存储技术，为大量数据的存储提供了先决条件。为使得数据能够直观展示，本平台设计了数据可视化的功能，对数据根据地理位置信息进行组织和展示。

本论文中使用移动终端的传感器进行数据收集，其过程会产生大量的原始数据，服务器需要对原始数据进行接收和存储，提供给后续的数据融合、数据分析和数据可视化部分进行处理。在参与式移动感知平台中，由于数据的异构性和数据访问的并发性，服务器对于数据的稳定接收和高效存储具有更为重要的意义，是服务器端其他功能的基础。

本论文首先对参与式感知平台的相关背景和技术进行介绍；之后阐述了系统的需求分析，并对系统进行模块划分，讲述参与式感知平台的网络接口模块与其他模块之间的逻辑关系和消息传递的流程；接着详细阐述了网络接口模块的内部结构、流程和实现方式；最后通过对模块的功能进行了单元测试，并进行了性能测试。

6.2 可继续开展的工作

限于作者自身经历和能力的限制，参与式感知平台和高并发数据处理的研究还存在许多不足需要改进。

- 为文件系统添加数据管理功能。在系统运行过程，需要对服务器中保存的照片文件进行人为的添加删除等操作，添加数据管理功能会使系统功能更加完善。
- 系统并发性能还可以进一步提高。参与式感知平台的数据请求有一些特性，后续研究中应针对这些特性以及用户体验等方面，研究算法或技术对并发性能进行改进。
- 任务推送不使用第三方应用。由于所使用的第三方应用的限制，进行部分

用户推送时需要由用户再自行发出请求进行确认，极大的浪费客户端资源，影响用户体验。在后续研究中，考虑使用 Openfire 等技术自主实现消息推送，使部分用户的组推送能够直接将消息推送到指定的用户，不需二次确认。

6.3 研究生期间主要工作

在研究生期间，论文作者参与的工作主要有：

1. 华为基金项目 IPv6 多归属与重编号 时间：2013.03-2013.09
   - 阅读 IETF 的 IPv6 多归属相关的 RFC 文档和相关 draft，进行总结
   - 研究多归属中的源地址选择机制
   - 搭建 IPv6 多归属实验环境
   - 验证多归属情况下的源地址选择。
2. 华为基金项目 自治网络的 MPLS 协商协议的设计与实现 时间：2013.10-2013.12
   - 参与协商协议的概要设计和详细设计
   - 负责协商协议顶层消息交互模块的设计与实现
   - 负责模块单元测试和集成测试
3. 国家自然基金项目 移动协助感知 时间：2014.2-2014.12
   - 参与系统的需求分析
   - 参与系统总体架构的设计
   - 负责系统网络接口模块的设计与实现
   - 负责进行模块单元测试和集成测试

参考文献

[1] J. Ma，C. Chen， J.P. Salomaa. MWSN for large scale mobile sensing[J]. Signal Process. Syst, 2008, 51: 195-206.

[2] 尹杰. 移动互联网中的 LBS 隐私保护技术研究[D]. 郑州： 解放军信息工程大学, 2012: 1-6.

[3] 刘云浩. 群智感知计算[J]. 中国计算机学会通信, 2012, 8: 38-41.

[4] O. Riva, C. Borcea. The Urbanet Revolution: Sensor Power to the People[J]. Pervasive Computing, 2007, 6(2): 41-49.

[5] 陈静姝，王庆官，张凡，迟学斌. 基于网格计算环境的可视化系统设计与实现[J]. 计算机应用研究, 2007, 8: 248-250.

[6] 悟网不欢．“人+智能终端”，强大传感器的强大未来[EB/OL]. http://www.huxiu.com/article/15388/1.html?f=chouti, 2013-06-04.

[7] 杨娟. 基于监测个人交通方式的一种能量有效的移动感知框架[D]. 西安：西安电子科技大学, 2013: 3-12.

[8] Yu Zheng, Licia Capra，Ouri Wolfson，Hai Yang. Urban Computing: concepts, methodologies, and applications[J]. ACM Transaction on Intelligent Systems and Technology. 2014, 5(3).

[9] 王静远,李超,熊璋,单志广. 以数据为中心的智慧城市研究综述[J]. 计算机研究与发展. 2014, 51(2): 239-259.

[10]杨峥，吴陈沭. 移动计算中的群智感知[EB/OL]. http://www.docin.com/p-744909088.html.

[11]梁明刚,陈西曲. Linux 下基于 epoll+线程池高并发服务器实现研究[J]. 武汉工业学院学报, 2012, 3: 54-59.

[12]黄冬泉,张敏,徐振亚,尹宝林,李伟琴. 高并发事件驱动服务器研究[J]. 计算机工程与科学, 2007, 1:138—141, 148.

[13]李军. 高并发 Web 系统的设计与优化[D]. 北京市：北京交通大学, 2009.

[14]刘苗苗. Web 性能测试的方法研究与工具实现[D]. 西安：西安理工大学, 2007.

[15]王涛. HTTP 协议技术浅析[J]. 中国新技术新产品, 2013, 22: 14.

[16]肖曦,南楠. 基于 HTTPS 的统一通信系统安全设计[J]. 物联网技术, 2011, 5: 67-71.

[17]黄河清,陈文. Android 平台消息推送服务的实现[J]. 电脑编程技巧与维护, 2014, 18:53-55.

[18]Android Cloud to Device Messaging Framework[EB/OL].

https://developers.google.com/android/c2dm/

[19]张长学，张伟，董志明. 移动推送技术面面观[J]. 移动通信, 2011, 35(5): 21-27.

[20]陈航，赵方. 基于服务器推送技术和 XMPP 的 Web IM 系统实现[J]. 计算机工程与设计, 2010, 31(5): 925-928.

[21]Extensible Markup Language (XML) [EB/OL]. http://www.w3.org/XML/

[22]任亨，马跃，杨海波，贾正锋. 基于 MQTT 协议的消息推送服务器[J]. 计算机系统应用, 2014, 23(3): 77-82.

[23]Walls C， Breidenbach R. Spring In Action, Updated For Spring 2.0[M]. Dreamtech Press，2007: 207-236.

[24]介绍 JSON[EB/OL]. http://www.json.org/json-zh.html

致　　谢

时光飞逝，两年半的研究生时光即将结束。研究生学习让我收获了很多，在此，我要感谢所有在研究生求学期间给予过我指导和帮助的人。

我首先要感谢我的导师龚向阳教授。在硕士研究生期间，龚老师在学习上给予了我很大的帮助。龚老师不仅为我们的项目把关，给项目指出研究方向，还在平时的学习和研究中给了我很多指点和帮助，在此谨向龚老师致以诚挚的谢意。同时，我还要感谢王文东教授和阙喜戎副教授，他们经常在百忙之中参与项目例会，为项目研究提出开创性的意见。对我的项目工作和论文写作两位老师也提出了宝贵的建议和意见，在学术交流中，王老师和阙老师对项目的实现方式提出独到的见解，认真的工作风格给我留下了深刻的印象，在这里我要向王老师和阙老师表示由衷的感谢。

感谢所有参与式感知项目组的同学。我们在项目开发中积极配合、团结一致，共同克服困难，完成了课题任务。从他们身上我学到了很多，并且也结下了深厚的友谊。

此外，我还要感谢我的家人，特别是我的父母，他们的关心和鼓励是我一生中最宝贵的财富，他们的支持与鼓励为我提供了坚强的后盾，解除一切后顾之忧。

最后，非常感谢诸位专家教授们在百忙之中评审我的学位论文，并提出宝贵意见。

作者攻读学位期间发表的学术论文

[1] 徐登佳. 基于位置的视频监控系统目标跟踪的设计与实现[EB/OL].北京：中国科技论文在线 [2014-12-18]