密级： 保密期限：

北京郵電大學

硕士学位论文

题目： 基于 MQTT 协议的通用电子标签系统

学　　号：2017180064

姓　　名：罗复翰

专　　业：电子与通信工程

导　　师：孙文生

学　　院：信息与通信工程学院

2020 年 2 月 15 日

中国 · 北京

密级：　　保密期限：

北京郵電大學

硕士学位论文

题目：基于 MQTT 协议的通用电子标签系统

学　　号：2017180064

姓　　名：罗复翰

专　　业：电子与通信工程

导　　师：孙文生

学　　院：信息与通信工程学院

2020 年 2 月 15 日

BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

Thesis for Master Degree

Title : GENERAL ELECTRONIC LABEL SYSTEM BASEDON MQTT PROTOCOL

Student No.: 2017180064

Candidate: FuhanLuo

Major: Electronic and Communication Engineering

Supervisor: WenshengSun

Institute: School of Information and Communication Engineering

February 15st 2020

基于 MQTT 协议的通用电子标签系统

摘 要

在一些商场、超市和医院等场所，都采用纸质媒介进行信息的传递，这样的传递方式容易造成信息的丢失，同时更换纸质操作不方便，既造成了大量纸张的浪费，也污染了环境。由于物联网的兴起，人们对智能化、可操作、低功率的电子标签有很大的需求，如果将生活中的纸质信息替代成电子显示标签，那么必然需要一款应用程序和一套管理系统将多个电子标签设备信息进行整合以及统一管理，同时可以实时更新标签信息，电子标签的出现将极大的满足人们的生活要求并提高工作效率。

在这样迫切需求的驱使下，通过对智能终端软件的开发和 MQTT 协议相关知识进行了解，系统地分析了通用电子标签的设计需求，本文设计并实现了一套基于 MQTT 推送协议的通用电子标签系统，该系统包含基于 Linux 系统的服务端系统、手机客户端为控制系统和电子标签信息显示系统，主要对这三个方面进行设计。服务端采用的微服务架构进行搭建的后端平台，主要用来进行与数据的读写操作，将手机控制端编辑的内容推送给电子标签显示端。服务器端是以 B/S 的结构进行设计，主要分为应用服务器和代理服务器，应用服务器是以 MVC 思想进行设计，代理服务器的核心是 mosquitto 服务器，应用服务器用来控制和监控用户信息和电子标签信息，代理服务器是手机 APP 和服务器之间传输的桥梁。手机客户端是基于 React Native 框架技术进行开发设计的，通过对用户的分析，确定手机客户端的功能模块，电子标签显示设备端是以 MSP430 主芯片进行设计低能耗的电路，对于电子标签设备传输是通过 ESP8266 芯片设计的无线传播方式，对于三者的通信传播设定了格式，确保传播之间的安全性。为了使传输速率得到提升，设置用户信息表，模板信息表以及电子标签信息表，系统模型最终将手机 APP 端发布的消息可以在某个具体的电子标签中准确显示标签信息。

最后，本文对该系统进行各个模块的测试，验证通用电子标签系统的可行性和稳定性。实验结果显示可以很好的满足设计需求，各部分功能模块可以很好的实现，采用 MQTT 推送协议的通信方案可以很好的满足系统设计。

关键词: 电子墨水屏；无线电子标签；MQTT；微服务；React Native

GENERAL ELECTRONIC ABEL SYSTEM BASED ON MQTT PROTOCOL

ABSTRACT

In some shopping malls, supermarkets, hospitals and other places, paper media is used for the transmission of information, which is easy to cause the loss of information, and it is not convenient to replace the paper operation, which not only causes a lot of paper waste, but also pollutes the environment. Due to the rise of the Internet of things, people's intelligence, operational, low power electronic tags show is very important, if the paper information instead of in the life into electronic display label, so it need a application and a set of management system will be more electronic equipment tag information integration, and unified management, and real-time update the label information, the emergence of electronic tags will greatly promote the requirements of people's lives and work efficiency.

Driven by such urgent needs, by understanding the development of smart terminal software and related knowledge of MQTT protocol, the design requirements of universal electronic tags are systematically analyzed. This paper designs and implements a set of universal electronic tag systems based on MQTT push protocol This system includes a Linux-based server system, a mobile client as a control system, and an electronic label information display system. These three aspects are mainly designed. The back-end platform built by the micro-service architecture adopted by the server is mainly used to read and write data, and push the content edited by the mobile phone control terminal to the electronic label display terminal. The server is designed based on the B / S structure. It is mainly divided into application server and proxy server. Application server is designed based on MVC. The core idea of proxy server is mosquitto server. Application server is to control and monitor user information and electronic label information.

Proxy server is the bridge between mobile APP and server. The mobile client is developed and designed based on the React Native framework technology. Through analysis of the user, the functional modules of the mobile client are determined. The electronic label display device is designed with the MSP430 main chip to design a low-energy circuit. For electronic label device transmission, The wireless transmission method designed by ESP8266 chip sets the format for the three communication transmissions to ensure the security between transmissions. In order to improve the transmission rate, a user information table, a template information table, and an electronic label information table are set up. The message finally released by the mobile phone APP end of the system model can accurately display the label information in a specific electronic label.

Finally, this paper tests the modules of the system to verify the feasibility and stability of the universal electronic label system. The experimental results show that the design requirements can be well met and the functions of each part can be realized. At the same time, the communication scheme using the MQTT push protocol can well meet the system design.

KEY WORDS: Electronic ink screen; Wireless electronic label; MQTT; Microservice; React Native

目 录

第一章 绪论

1.1 研究背景与意义

随着各国环保意识的增强，对资源浪费的行为逐步引起人们的注意，尤其是在超市、商场、小型卖场以及信息标识的地方等场合下，常见使用以纸质或者其他材料作为载体进行信息标识的现象产生环境污染，当此类以实体材料为载体的信息需要修改或者撤换，又容易造成资源的二次浪费[1]。基于 MQTT 协议的通用电子标签系统，是以电子标签来标识需要的信息，进而可以节约资源，使人们生活水平得到提高。

目前无线电子标签在可变信息标志和静态信息显示牌领域已经得到了广泛应用，例如：在会议室场合座位人员信息牌、各大商场的商品价签、公共汽车的车站牌、在医院病患者的病历牌、加油站的油价表、餐馆的电子点菜单和各行各业商品零售业标签显示牌、公告牌等场所，都具有良好的市场应用前景。随着物联网的不断发展和科技的不断进步，如何让物联网设备之间的消息更好的传输是目前需要解决的问题。与传统物联网相比，物联网设备在网络环境方面，始终存在设备处理数据能力慢、带宽小、稳定性不足的缺点，这导致在消息的传输上需要更加严格的要求。

MQTT（Message Queue Telemetry Transport）协议是由 IBM 开发的轻量级的消息传输协议，以其开销性能小、准确性高和稳定性传输等优点在各种物联网应用场景中得到了广泛认可，其基于主题的发布订阅特点，可简便实现在物联网平台中端到端的一对一或一对多的消息传输，已成为目前物联网通信协议标准中的重要竞争者。物联网平台在实际应用中将会面临两大问题：如何利用物联网节点设备来最大化的节省资源和如何在不稳定、低带宽的网络环境中实现快速稳定的消息传输[2]。对此，MQTT 在计算能力弱、低带宽、以及不稳定的网络环境方面做了优化以及改进，使其在智能手机终端和嵌入式终端设备上的消息传输，对终端设备的流量和耗电量在资源利用率上得到了最大化的优化。目前，国际物联网消息传递标准协议中便包含着 MQTT 协议。由此看来，选用 MQTT 协议来设计开发通用电子标签系统是有广泛的应用价值的。

由于物联网的很多应用场景都在生活中得到了普遍应用和推广，使得物联网平台开发的意义和获得的商业价值也有了充分的展现。国内外各大互联网企业统统涌入物联网，搭建以物联网为基础的管理平台为用户策划一套完备的物联网云平台解决方案。目前在市场上熟为人知的有腾讯物联网通信、QQ 物联智能硬件开放平台、AWS IoT、微软 Azure IoT、Google Cloud 的物联网平台、IBM Watson IoT 等等。这些物联网云管理平台都能很好的支持 MQTT 协议，也是作为云平台的接入的首选协议之一。物联网云服务方案商基础服务和开发逻辑处理系统中的重要部分是 MQTT 消息订阅发布系统，它能够很好的来解决设备与云端之间如何建立稳定有效的双向连接，以便支撑大量设备的数据录取、性能监测、参数

变更等更多的物联网适用场景。MQTT 消息订阅发布系统的事务运算能力不但会在云平台提供全局服务时受到影响，而且会直接决定云平台可同时支持的最大访问数量和在物联网云平台解决方案的可靠程度。

基于 MQTT 协议在物联网系统从端到端网络通信应用中有得天独厚的优点，本文主要描述的是以 MQTT 协议通信为基础，研究基于 MQTT 协议的通用电子标签系统的设计。这对于推广 MQTT 协议在物联网中的运用，提供了轻量级的物联网系统端到端的消息传输解决机制，推动物联网系统端到端网络通信协议的进步具有重要价值。

1.2 国内外研究现状

目前在商场中，商品种类繁多、品类不计其数且商品价格更新次数较多；在医院中，各种疾病患者的各种信息都需要及时更新替换；在会议室中，每个座位的开会人员也需要频繁更新，这样使得传统纸质标签更新方式已不能满足现在生活中各类标签快速更新的需求。可变信息标志和静态信息显示牌是获取公众信息服务的重要工具和手段。目前信息显示材料普遍存在着能耗大、体积大、视角小以及易造成光污染等问题，采用电子墨水技术的显示屏可以弥补这些缺点。考虑到现有电子墨水屏的发展水平，要将其作为解决实际的城市信息发布媒介屏幕尺寸较小以及与信息发布中心建立连接标准等问题。

电子墨水屏无需像 LCD 屏背光发射，它是利用自然环境光打在显示屏上，再折射到眼睛的原理。这种方式模拟了墨水与纸张的特点，环境光越强，其显示效果越清晰。由于没有了闪烁，所以在长时间阅读时，眼睛不容易感到疲累。通用电子标签技术可以运用到多种场合，得到了人们的认可。在国外，以电子墨水屏为显示的标签信息可以成功运用到阅读电子书产品。澳大利亚已在公交站牌得到应用，公交站可以准确显示公交班次和时间信息。对于产品网络连接也是通过移动 WIFI 通信，使用时间也很长[3]。对于高强度的阳光照射和风环境下还可以正常工作，说明其可以适应恶劣环境。在国内，电子墨水屏显示的电子标签的使用也逐渐多了起来，在一些医院、超市、会议和商店商品都可以见到[4]，但是都是单一使用，容易造成浪费，所以设计通用电子标签系统，很有应用前景。

1.3 研究内容与目标

通用电子标签系统设计包含电子标签显示设计、APP 客户端设计和服务端设计。对于硬件电路设计，主要采用自顶而下先设计模块间交互的模块化的思路。电子墨水屏 PCB 是主要由大连奇耕公司的 GDE 系列墨水纸（e-ink）显示模块、TI 公司的 MSP430F5529IPN 控制芯片、ESP8266-12F 无线通信模块、电源管理等四大模块有机组合构成。对于硬件电路的程序设计，设计一个基于 TCP 之上的 MQTT 协议的通信传输客户端，用户界面主要包含可订阅不同墨水屏硬件端发布的 MQTT 主题的选择面板控件和内置二值图片的自动解码十六进制的功能以用来实现图片传输的一定分辨率大小的用户设计图框等两大块。通用电子标签系统通信流程主要是 APP 通过 MQTT 协议发送数据，经 ESP8266 的 WIFI 模

块透传后发给 MSP430 主控芯片，最后将 16 进制数据发给终端墨水纸进行显示。本系统主要采用了诸多有低功耗性能的 ICs，如 ESP8266 和 MSP430 系列等，最大特点是工作时长长，一次充电理论上可以工作数个月，而且对于最大耗电的设备 ESP8266 采用 MSP430 控制 PMOS 管将其电源端进行物理断电以保证绝对低功耗。其次，用户可以通过客户端实时控制电子标签，最后进行系统的测试以及下一步的优化。

1.4 论文组织架构

本课题在对电子墨水屏标签与现有物联网协议调研分析的基础上，将物联网 MQTT 协议与墨水屏相结合，提出一种新的电子墨水屏物联网架构，设计并实现了基于物联网 MQTT 协议的通用电子标签系统，并对所设计和实现的通用电子标签进行了各项指标测试。论文总共有六个章节，章节安排及具体内容如下：

第一章为绪论，讲述了研究课题的背景，设计通用电子标签系统的目的，以及国内外对于电子标签系统的研究进展提出了基于 MQTT 协议的通用电子标签系统，提出新时代通用电子标签的改造及与新物联网技术结合的需求，说明了物联网在物联网架构中的重要作用，通过调研传统显示标签及当前主要物联网应用层协议，分析了 MQTT 协议在电子标签系统上使用的优势。

第二章阐述了通用电子标签系统设计开发所需要用到的相关技术进行分析，并详细介绍应用到电子标签的设计思想，此部分主要介绍通用电子标签系统所使用的相关技术，从技术特点层面对系统的可行性方案进行分析。

第三章阐述了整个通用电子标签系统进行总体需求分析和架构设计，并详细描述了系统中的电子标签显示端、智能手机客户端和服务端的设计。

第四章阐述了通用电子标签系统中的电子标签显示端、智能手机客户端和服务端的实现。

第五章阐述了通用电子标签系统各方面的测试，本文通过系统测试和单元测试对系统性能进行了测试，通过测试证明通用电子标签系统的可行性。

第六章总结了本课题的主要研究工作，对所设计的通用电子标签系统仍需改进的方面以及未来电子标签系统发展进行了展望。

第二章 关键技术研究

本章介绍系统设计运用到的技术方案和研究思想，对原理特点进行分析，介绍与研究电子标签系统设计思路和设计相关技术，对深层次的原理进行探索，主要技术由通信 MQTT 协议的相关技术、微服务架构的原理、REST 技术、Spring MVC 和 React Native 框架组成。

2.1 即时通信 MQTT 协议

2.1.1 协议介绍

MQTT 协议是一套轻量级跨平台的基于发布/订阅的消息传输协议，其优势在于设计思想是开放、简单、轻量、易于实现。其工作环境处于低带宽、计算处理器配置低、内存资源有限和网络传输受限[5]。MQTT 协议主要有发布和订阅功能。在 MQTT 协议下有多个客户端需要发布和订阅信息，这时候会通过一个代理服务器去发布信息，服务器指派另一个代理服务器去订阅，订阅主题相同就能让它们进行通信。客户端的发布信息通过代理服务器指定发送话题信息，客户端接收由代理服务器传送到应用服务器进行信息的订阅，同时会通过 PUBACK 服务返回一个值去判断是否传输成功。

2.1.2 协议特点

本小节将通过叙述 MQTT 协议的具体功能特性和工作原理，主要分析了电子标签端的信息应用设计，通过 MQTT 协议在电子标签协议中的设计及传输方式，分析如下[6]：

(1)设计思路

MQTT 协议主要是通过发布/订阅进行信息的传递，通用电子标签系统设计可以在手机 APP 端口和服务对其直接进行运用。在 MQTT 协议下可以进行对象与对象的行为模式。二者设计下，在 MQTT 协议上多了一个事件管道。发布者和订阅者都是从两个管道传递消息事情，互相分开来调整依赖性强的情况。这种发布和订阅的推送方法不是耦合连接，通过分开传输，手机 APP 和服务器直接进行交互，也是通过服务器进行控制。服务器推送模式的消息传输流程如图 2-1 所示。

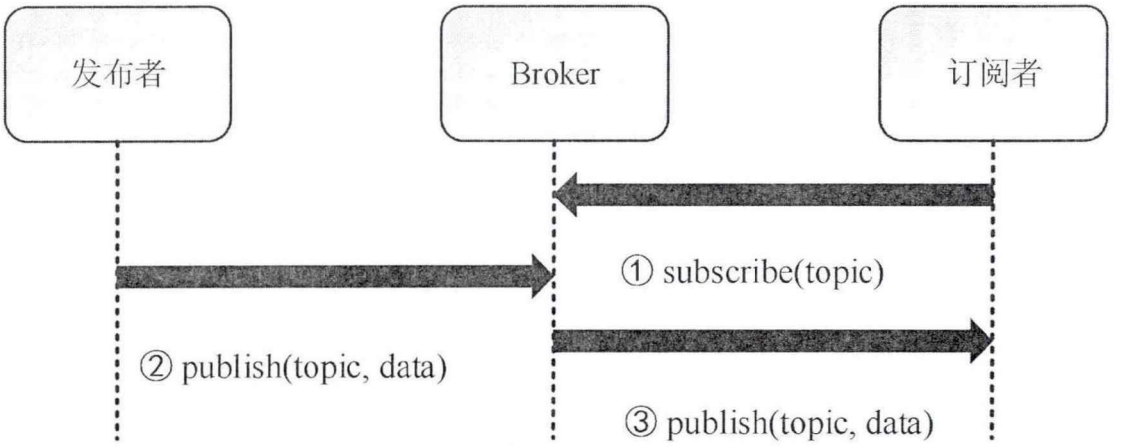

图 2-1 推送模式示意图

从上图 2-1 中可知，每一个客户端就是一个个的发布者或是订阅者，手机 APP 客户端是通过发送到端服务器进行标签信息传输，它们之间通过代理服务器 MQTT Broker 作为桥梁进行消息转发。首先订阅者向代理服务器订阅主题，然后发布者向代理服务器推送主题，最后代理服务器将该主题的数据推送给订阅者，订阅者便能接收到消息，这样一次消息传输完成。

对于发布者和订阅者来说通过代理服务器进行传输非常有优势，它可以降低订阅者对自身开发设备的性能要求，减少订阅者和发布者本身软件的设计要求，减少损耗。

(2)身份标识

ClientID 是每一个客户端的身份标识，而代理服务器通过 ClientID 来识别客户端，这样就能清楚的知道客户端与服务端之前的网络连接状态。假如在网络传输过程中数据一致，数据信息就会保存下来。在相同的时间内服务器和客户端 ClientID 会通过网络进行连接，若客户端在发送相同的 ClientID 进行交互，则之前的网络连接 ClientID 就会断开连接，新用户则进行传输。

在 MQTT 协议下，客户端之间的连接是的非常重要。现在设计的客户端都是通过代理服务器进行信息的订阅，然后再进行传输的，等接收端有信息发布时候就会接受推送。但网络传输存在不稳定的情况，就需要客户端进行重新连接通过调用初始函数。由于调用初始函数的代码是通用代码，所以客户端和服务器之间连接的 ClientID 是随机产生的数，客户端和 ClientID 连接就会中断，即使中断连接，客户端也会显示以前存在的信息，这时显示的信息视为重新连接，影响客户端的运行。在实际应用中，根据要求来进行新的 ClinetID 进行连接实现传输。

(3)清除标识

在客户端连接时，该清除标记需要由客户端来设置。若值为 0 则表示如果之前 subscribe 成功的客户端断线，要推送给客户端数据时需要空出时间进行连接。服务端断开有重新进行连接的时候，客户端又会重新接收到新的数据。

(4)心跳机制

由于系统设计需要长时间连接，所以需要很强的稳定性，心跳机制可以满足这一要求。同时客户端和服务器之间的连接可以由心跳机制进行连接，客户端通过在 CONNECT 报文中设置 Keep Alive 的值。通过心跳机制设置时间间隔进行操作，由代理服务器判断是否接受信息，若没有接收到客户端信息，则执行断开连接；若接收到信息，就会执行更新操作。心跳机制可以根据系统模型设置需要的时间间隔。

(5)遗愿消息

遗愿消息表示的是客户端不是在发送 DISCONNECT 消息中断，出现了异常中断，这时候需要服务器来做的一些措施。客户端和服务器进行信息传输的时候，报文可以指定位置填入主题(Will Topic)和消息(Will Message)。若客户端出现故障，无法连接网络，服务器会发布信息到主题上，客户端也会重新收到通知信息。

(6)主题

在主题(Topic)的设计方面，MQTT 协议采用通配符的层级设计，支持通配符“#”，“+”以及“/”。当客户端订阅主题时，可以通过一次请求来实现订阅多个主题的目的。这是通过主题筛选器(TopicFilter)来明确的，表示订阅所匹配到的多个主题。主题筛选器中的特殊符号主要有三。一是层级分隔符“/”： 一个主题可以拥有多个级别，而级别之间用斜杠字符来进行分隔；二是单层通配符设置为“+”；三是多层通配符设置为“#”：支持一个主题内任意级别话题。我们通过详细的了解主题这一概念对于后文中的通用电子标签系统设计提供了很大的帮助。

2.1.3 物联网中 MQTT 协议的应用

上一小节通过对 MQTT 协议自身的研究和特有的推送模式，我们知道了 MQTT 协议具有以下应用特性，这些特性对于物联网云平台领域的应用需求来说，MQTT 协议有着得天独厚的优势。

首先是 MQTT 协议的开销小。在客户端发布的消息体中，其消息最小可缩小到两个 Byte，这就是它与其他协议的不同之处。相比于 MQ 和 HTTP 协议，它们都拥有比 MQTT 协议高很多的单独消息开销[7]。在 HTTP 协议下进行通信传输，是一种短连接的方法，当客户端需要发送新的信息，这时需要重建连接才能传输信息。MQ 和 MQTT 传输是通过长连接来实现的，设置系统传输方法可以进行改进；第二是在网络环境的可靠性。对于 HTTP 协议来说，在网络出血异常断线的情况下，若想恢复网络重连，则需要额外的编写代码来实现，另外可能还会导致一些幂等性问题的产生，而 MQ 和 MQTT 协议，因为它们底层支持自动恢复功能，所以它们都能够自动从异常断开中恢复，较为简单和便捷，减少人力开发的成本；第三是 MQTT 协议功耗低。当时 MQTT 的设计初衷就是如何做到

设备的功耗最低化问题，而 HTTP 协议却没有考虑该问题，导致在设备的功耗方面，明显比 MQTT 协议消耗的更多。还有一个特性就是推送系统。通过服务端主动去推送消息，可以让物联网云管理平台对外提供更好的实时传输服务，这相比通过客户端轮询的模式来说，主动推送可以更好的节省资源和提高效率。另外在推送方面还要考虑到消息的可靠性，确保数据不会被非法用户窃取，所以也不能选用第三方推送服务作为推送系统的核心方案。在 HTTP 协议中，可以使用 COMET 的方法可以实现推送消息，但是需要耗费客户端和服务端更好的资源，不能让其资源最大化，而在 MQ 和 MQTT 协议中，它们本身都能够原生支持消息推送，很好的保证了推送系统的实现。

综上所述，MQTT 协议本身就是为物联网应运而生的产物，总结特性有如下几点：一是 MQTT 协议是通过发布/订阅方式进行服务器和手机 APP 端进行信息交互的。它们之间可以一对一，一对多方式进行传输；二是可以通过主题信息进行发布传输；三是通过 MQTT 协议服务质量可以得到提升；四是 MQTT 协议能够提供原生的异常分析机制，每当有 Client 异常断线时，将使用遗愿消息机制告知其它相关的 Client。

2.2 微服务架构的原理

常见的后端开发系统，如企业资源计划, 客户关系管理等信息管理系统，都是使用的的整体式架构(Monolithic Architecture)来搭建而成的。虽然整体式架构在开发初期阶段时可以快速的构建，但是在后期维护更新阶段上却变得越来越棘手。如今处于移动互联网快速发展的时代，各大互联网企业也不得不针对移动智能终端设备来进行频繁的界面设计更新和修改，这要求各模块的系统能快速维护上线，需要不间隔的添加或删除某个模块，这就导致了传统型整体式架构越来越为难，越到后期越撑不下去。虽然我们可以对整体式架构中的一些 API 接口函数进行相关标准化并进行重复使用，但是如今市场的快速变化需求需要更好的应变能力，导致整体式架构凸显其局限性，因此微服务架构反而比传统架构越来越受企业的青睐[8]。

微服务指的是一种架构的特性，它能够将大型复杂后端管理系统拆分成多个微服务，而每个微服务组件只需要一项业务或功能，并且对业务的执行得到充分的保证。正是这种微服务架构特性，使得各项微服务都能被单独部署，也可单独开发，解除了各个微服务之间的依赖关系。

微服务架构的特性是与全局式应用比较而来的，单个应用是将所有业务和功能都放在一个进程里面，如果要进行分布式部署和业务扩展的话，则必须要将之前的进程进行全部拷贝。但与微服务对比来说，微服务只需将各项功能业务按照计划进行剥离开来，各项服务运行在不同的单独进程中进行维护，如果需要进行

分布式部署和业务扩展，只需要将不同的业务功能进行按需复制，大大提高了后端开发人员的效率和可维护性。通过研究表明，在整体上微服务架构可以根据不同的业务场景从而具备不一样的特性，但总体上仍具备以下通用特点：

(1)软件应用组件化通过微服务来实现。微服务中的组件能独立进行升级改造或换成所想要的模块，然后在整个软件应用中进行重新上线部署，使得系统各组件层次结构化。

(2)微服务依赖业务功能。微服务的组件都是通过具体开发业务或者功能而来的，因此微服务的项目开发团队是基于跨智能的组织方式，既要有实际应用开发人员，又要有数据交互开发人员，由于在业务划分明确且细致入微，因此一个项目团队一般不需要太多人员。

(3)产品特性。这是微服务的特点，微服务的生命周期是需要一个项目团队来共同参与，从方案设计到线上部署以及运作维护的全部步骤都需要承担起来。这与传统架构的设计开发与部署团队有着差异，微服务提倡的是软件系统的产品化方式，走开发部署一体化的模式。

(4)相互独立。微服务架构是各模块间是相互独立的，同时网络通信模块与各应用模块也是相互独立的，有助于降低各模块之间的依赖性。

(5)“去中心化”管治。与整体式架构采用的一样的编程技术平台的模式略有不同，微服务仅仅对业务功能进行划分，但对其完成目标不作明确限制，提倡采纳不同的技术来实现独自的模块，即便使用不同的开发语言，但相关接口明确便没有问题，这样能更好的找到不同的业务功能其合适的方式。

(6)“去中心化”数据管理。传统的整体式架构通常是对数据层进行统一管理，而微服务架构仅仅对本业务模块负责，在数据交互持久化时，每个微服务都可单独使用其数据库，并且不同的业务模块上可安装不一样的数据库服务。

(7) 自动化统一设施。有了云平台的推动，使得微服务技术架构在实现上更加的自动化，有效的降低了微服务发行、运作和维护的困难程度及人力成本，通过云平台的方法可以更好推动微服务的发展。

(8)异常处理维护。各微服务是相对独立的模块，单个模块出现异常故障虽然不会造成全局性能的影响，但出现的每个后果都是需要每个模块必须结合自己的异常处理和容错机制，负责好自己的模块功能，因此更需要关心去了解与微服务相关的性能监测。

(9)演进式的设计。微服务各模块可快捷的进行更新换代，因此整个系统的框架结构也会随着反复的更新而进行更替。微服务的区分和管治一直是微服务的重难点，其区分也是随着每个模块的不停更替而进行相应地更新。同时在传统型架构中，在之后的业务演进中也可采取微服务的方法来演进。并在更替过程中，

可能会存在区分的不合法，这就会进行相关模块的合并和与之分解。因此微服务架构是个不断演进式架构设计。

通过以上了解微服务的特性，本系统也是采取这种微服务架构的形式，来实现需要的微服务模块应用。在整个微服务架构的基础上对于其自身应用需求功能开发相应的参数和 API 接口，这样可以实现该系统的独立模块布署和上线运行，对于系统研发的扩展性和灵活性都得到了充分体现。

2.3 REST 技术原理

REST 是 Representational State Transfer 的英文缩写，语义为表述性状态转移，其概念最早由 HTTP 协议规范的编写者提出，HTTP 架构便是 REST 风格的经典体现。早期提出时便获得了众多程序编程人员的一直好评，主要优点有扩展性强和便于简单操作。由于分布式计算的蓬勃兴起和移动互联网时代的来临，信息和业务的共享需求也随之越来越频繁。在 REST 的设计方面主要要遵守以下规则[9]：网络上能被识别一切对象都称之为资源，资源是不可或缺的，每一个资源都会自身唯一的身份标识，资源采用使用 HTTP 协议定义好的统一接口进行交互，交互过程中都是在无状态下进行的操作，而操作之后的资源也不会改变资源的身份标识。

源于此设计规则和特性，REST 才能够如此简单的进行编程操作，因其对架构采取了相应的限制，在对 REST 的实现把控资源的方面限制在 7 个，如新建、修改、删除等，HTTP 将网址 URL 的交互方法进行限制，只有 GET, POST, PUT 和 DELETE 这四种方式；REST 还能提升系统的上下伸缩性，因为其明确约束的交互方法是没有状态的，所以在分布式、集群适用场景中，就不需要思虑上下文问题，同时在服务端只需提供资源以及对资源的交互即可，大大降低了服务器的总体性能开销，提高了生产效率。

通用电子标签系统内实现的所有微服务，都是采用的 REST 风格的 API 接口，这样有利于 API 接口的管治和整个系统上线时的统一调用，是一种统一的风格样式。

2.4 Spring MVC 框架介绍

MVC (Model-View-Controller)是一种非常出名的设计模式，该模式尤其适用于在网页端应用领域上，其主要是通过将系统分离成模型、视图、控制器这三部分。模型主要用来是对底层对象数据的封装，视图主要是用户和数据交互操作之间的中间件，控制器主要是用来分发数据的请求，每当接收到数据时将回馈到系统模型之中[10]。

Spring MVC 便包含了 MVC 中的的重要观念，MVC 为控制器和后台逻辑的

相关处理提供了有效的渠道。通过反转控制的参与，使得业务处理变得相对独立，只需相应的参数配置就能改变底层模型。Spring 的 Web MVC 模块是围绕前端控制器而制作的，前端控制器不仅可以对语义处理和话题解析，同时还能处理上层应用的分派命令，支持数据的上传操作。前端控制器通过使用后台逻辑的相关处理来决定哪个处理需要去处理接收的请求，该映射对应的只是用于标识处理特有的模式，它只有一种方式 ModelAndView handleRequest(request,response)的控制器 API 接口的调用。Spring MVC 具有很多方面的优点，对对象都有着明确的划分，如前端网页操作、请求命令处理映射、图形解析器等，各对象都单独负责该有的特定业务功能，快捷高效，因而其分工明确到位，扩展方便迅速，并能够其他的 Spring 架构很好的结合使用，具备优良的适配性和灵活性，提供了数据考证和格式化的机制原理，并能在本地进行解析数据，还可支持开发国际化编程方式。

Spring MVC 在处置用户请求的一般步骤分别是：(1)用户发送请求，前端网页操作器接收到用户发来的请求，依据消息详情分配与之对应的解析器进行操作。(2) 前端网页操作器将请求发给请求命令处理映射，把请求命令映射成一个页面控制器和多个拦截器组成的实体对象，这样便完成了请求的映射处理过程。(3)映射操作发送给适配器，并对相应的操作进行适配。(4)适配完成之后，选择处理器进行相应操作方法的调取，并会返回对应的用户数据和视图。(5)图形解析器接受相应的用户数据和视图之后做处理，获得明确的视图。(6)根据处理的视图进行烘托。(7) 前端网页操作器将接手管理权，将烘托后的视图返回给用户。Spring MVC 是一个基于 Action(动作)的 MVC 框架。该框架体现了 HTTP 协议中的请求/响应特点，在该框架中，用户的所有请求都表明了所需要操作的动作。而这就是通过把每个请求 URI 地址映射到一个可操作的方法上来实现。同时，也把请求的参数映射到了对应函数的参数之中。Spring MVC 架构可以与 Spring Cloud 完美搭配，后者是高效有用的微服务实现容器，所以本文研究的通用电子标签系统中的后端项目便采取了这种框架来进行构建。

2.5 React Native 相关技术

React Native——是 Facebook 在 2015 年 4 月开源的跨平台智能手机终端应用编程框架。它作为 Facebook 早前在原生移动应用程序平台上开源的 React 的衍生产物，以一种全新的方式来呈现和解析移动端网页，使它们高度动态化和响应用户请求，所以前端开发技术人员仅需很少的学习成本就可以进入智能手机终端应用的开发。

2.5.1 React Native 简介

Native App 开发的主要优势是其性能好和功能性强，但一直处于苹果公司的管制之下，对于开发人员来说，非常的不方便，另外对于软件版本的维护来说，也造成了极大的困难。因此想要脱离苹果公司的限制，Facebook 之前曾换了个技术角度去设计开发移动 HTML5，但是结果却适得其反，耗费了大量的人力和财力不说，其研究性成果也并不显著，与 Native APP 比较存在较大的差距，在性能和功能方面都不令人满意。于是在 2012 年 9 月，Facebook 单方面宣布在移动 HTML5 研发道路上存在明显的问题（Betting on HTMLS was a mistake.）。后来便对移动 HTML5 技术研发进行了抛弃，努力寻求新的解决方案，以图达到更好的性能指标。虽然 Facebook 又转型原生 App 的开发后，但是其公司内部还在技术开发新的移动程序设计方案，并对内部应用程序就是通过这项技术方案来研发的[11]。这便有了 React Native 框架的由来。直至 2015 年 4 月，Facebook 便正式发行 React Native 框架技术，并将该技术通通采用开发源代码展示。最开始在开发平台上具有平台的限制，当时只能支持 Mac OS 系统，并且在上线运作项目时也只能支持手机 IOS 系统。在 2015 年 9 月，Facebook 便发行了 React Native for Android 版本，能够将市场其余用户数量使用的智能终端平台 Android 系统也采用 Web 和底层平台上的 JS 的技术，体现了 React Native 框架的能进行跨平台操作，且维护性强。

2.5.2 React Native 特性

第一点就是在效能方面，与底层自有软件 APP 比较而言，没有什么很大的区别。React Native 框架其提供的组件是对 IOS 系统底层接口的包装，并展现出可用的调用函数。虽然是使用的是 JS 来开发客户端，但在实质上是调取底层函数，以及系统自有的页面模块。因而在用户体验度方面，与自带软件程序相比，其效果不分上下。

第二点是在编程效率方面，非常容易快速上手。React Native 采用的是 JS 进行编程，语法上来说是非常灵活的，并且能够让前端开发人员直接跳转到智能移动终端的开发，大大的降低了学习成本和人力物力。

第三点是模块化分离开发。React 的理念被 React Native 所秉承。它的理念是将手机应用程序设计成不一样的模块化组件，模块之间毫不影响，而且每个模块都需要被当作成一个单独的模型来映射，就像建造房屋一般实现智能移动终端的开发。这种开发编程方式不仅在开发代码上具有鲜明的层次结构，而且适用性高，可复用性强，维护性强。而且在一些其他手机应用程序的模块通常也能被复用同一个模块，这些模块可以被很简便的直接拿来取用，就比如第三方的模块库，就可以快速的加入到项目当中[12]。其具体设计图如图 2-2 所示。

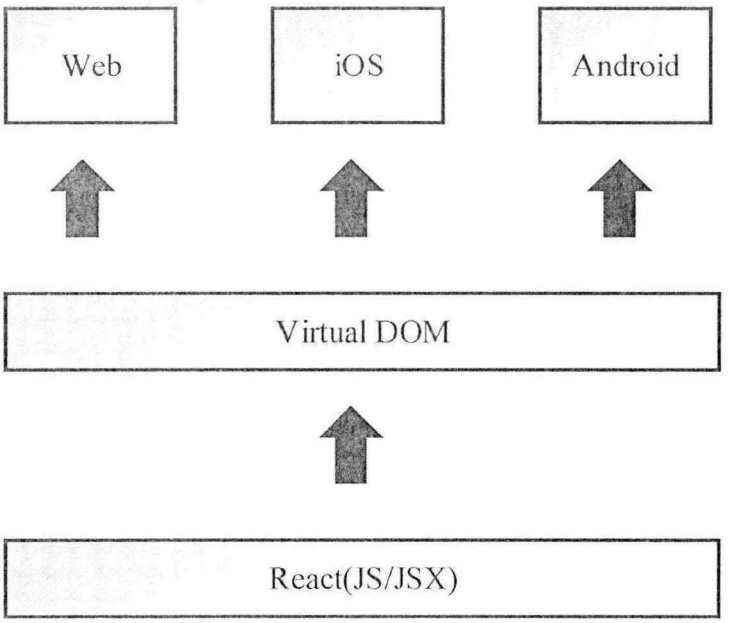

图 2-2 React Native 设计图

React Native 主要特点有如下几点：

(1)支持原生 APP 组件

由于 React Native 本身自带的丰富组件编写的界面，所以可以在不同平台终端上永远保持一样的样式与风格，这对于不同平台的客户来说，体验度都是一样的。

(2)异步执行

JS 应用开发代码和原生 IOS 或 Android 平台之间所有的网络通信机制都是使用的异步执行来处理的。

(3)触屏处理

React Native 本身有个与 IOS 上 Responder Chain 响应链事件处理机制相一致的请求响应模式，在对移动端复杂触屏命令上也做了进一步的改进。

(4)可伸缩的布局模式

React Native 采用的与 CSS 的弹性布局相似的布局，这样的布局有利于开发人员快速搭建普通的界面布局，利用可伸缩的方式来对齐和分布容器中数据的空间，让界面组件可以兼容不同样式的屏幕大小，这样为布局和样式给予了较大的机动性。

(5)Polyfills 功能

通过 npm 来安装所想要依赖库将其加入到 React Native 框架当中，这样对于开发人员的侧重点主要是在视图层代码的撰写方式上，而勿需担心移动端默认浏览器对这些引入的依赖库的适配问题。

(6)可扩展性强

React Native 框架也能和原生视图组件很好的结合搭配使用，在接口设计方

面，扩展性强，可以让较多的组件都采用自定义的形式来进行获得。

目前智能移动终端开发场景下最优秀的跨平台开发框架之一便是 React Native，不断通过各方面资源进一步的优化和调整，使得在 React Native 下开发，在应用程序安全和隐私方面也具备了可靠性保证，在大大降低了内存使用的同时，对于开发的应用具备高水准的应用 APP 框架结构也有了保证。

在面临实际的开发成本上面，通过 React Native 框架可以很容易开发一个 APP 中占有 90%的 UI 界面，APP 设计的 UI 界面需要兼容安卓与 IOS 操作系统，通过相同代码可以控制不同的操作系统。并且能通过一些方法函数(例如在 flex 布局去把界面自动调整屏幕分辨率不一致的智能手机，而不用开发人员去设计和掂量计算视图中的定位与大小)进行即时远程模拟测试。React Native 的远程程序测试与 Cordova 的便捷性能的提高变化是翻天覆地的。当开发人员启动了调试重载以后，会由于程序代码的变动，APP 界面和功能也随之自动更新，这样对于在开发人员在进行构建界面时，可减少开发时间。开发者可以在 XCode 中实时浏览到程序打印的日志信息。在 React Native 代码方面，采用的是热部署方式。对于 IOS 开发人员而言，苹果系统官方的审批步骤需要长时间的等待操作。而 React Native 框架本身采用的是 JS 去搭建的应用程序，它具备有代码实时编译的特点，这样可以让应用程序如同网页般进行热加载，同时能实时发行，所以开发人员只需在苹果官方商城上架时审批一次，若还有新的版本的出现也无需进行再次审批，省去了大量的等待时间；采用 React Native 开发能高效降低移动应用安装包实际大小。开发的应用程序项目若比较简单，功能单一时，React Native 设计会比原来的所占的内存要大，但是开发的应用程序项目若功能多样性，界面设计复杂，这时 React Native 开发将会比原生开发的安装包的体积小，内存容量小。若安装包所占的内存容量太大，这时在 React Native 框架重写程序，安装包体积将会减小。

第三章 基于 MQTT 的通用电子标签方案设计

3.1 整体架构设计

本文是基于 MQTT 通用电子标签系统的设计，主要通过对手机 APP 端、服务器端和电子标签三个部分进行设计。服务器是采用微服务架构原理，通过数据库之间进行信息传输。服务器中的代理服务器是连接手机 APP 端和服务器之间的桥梁。手机 APP 端要满足人们生活的需求，手机 APP 控制电子标签的信息，通过不同场所的要求，设置需要的标签模板进行信息传送。最后在电子标签显示端输出需要的标签信息，同时可以作为测试系统的性能的准则，三者的关系如图 3-1 所示。

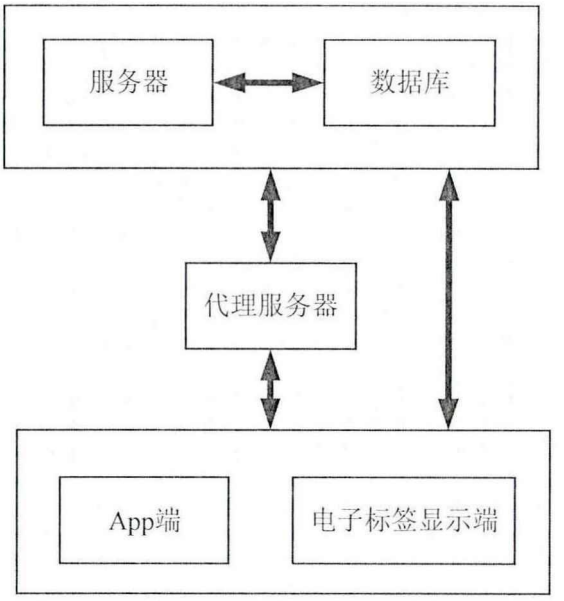

图 3-1 系统架构图

在设计通用电子标签系统时，考虑到开发的可实现性。设计出来的产品要有安全、可操作和便携等优势，手机 APP 端要根据 APP 开发准则设计用户需求方案，同时要考虑到系统开发过程中存在的问题，例如：设备卡死、显示信息错误和数据传输不稳定等常见问题[3]。

通用电子标签系统的运行是当用户需要标签时，通过手机 APP 进行设置，传送到电子标签端进行显示，传输过程中通过 MQTT 协议进行，本节主要根据通用电子标签系统的设计要求和用户需要的功能进行设计系统框架。

手机 APP 端是通过对信息标签进行编辑，通过代理服务器进行发布相对应的主题信息，同时也可以通过服务器管理终端。数据推送过程是用户通过 APP 进行标签信息的发布，把编辑好的标签信息发送给代理服务器。代理服务器对标签信息进行储存消息列表。用户也可以登录服务器端进行后端管理。服务器和手

机 APP 端之间的标签信息进行传输，都是通过 JSON 格式[14]进行发送给电子标签显示端，电子标签信息将被显示出来。

3.1.1 服务器设计

服务器分为代理服务器和应用服务器，手机 APP 端 HTTP 协议进行发布和通知，电子标签端使用 MQTT 协议进行通信。服务器端设计是在一个物理服务器上搭建两个服务器，分别为 HTTP 服务器和 Mosquitto 服务器。

服务器是以 B/S（Browser/Server）结构进行设计。从系统的逻辑功能看（如上图 3-1），应用服务器主要控制电子标签信息进行发布和订阅，同时实现对用户信息进行管理，应用层有多个接口用来交互数据。用户可以通过手机 APP 管理电子标签，对标签信息进行管理和编辑，实现电子标签信息的传递。应用服务器设计采用 MVC 的思想，设置成前台显示层和后端控制层。应用服务器是以 MVC 核心思想进行开发的，可以分为前台显示层和后端控制层进行信息传输：前台显示层主要对电子标签信息进行显示，后台控制层设计包含管理员登陆服务器账号和查看电子标签基础设备的信息等功能。

应用服务器从具体开发考虑，采用 MVC 设计思想，可以划分为前台显示层和后端控制层。前台显示层的功能是对编辑的电子标签进行显示；后台控制层设计包含管理员登陆验证、电子标签设备管理、消息编辑和信息发布等功能。详细介绍如下[15]。

服务器功能设计与手机 APP 端进行通信的服务器使用 Spring Boot 框架搭建，实现快速的微服务开发。使用@SpringBootApplication 进行初始化定义，使用封装好的语句对服务进行相应开发。使用@Restcontroller 直接对 HTTP 的请求进行处理，使用@RequestMapping 来处理地址映射请求。

对业务层和数据层进行设计，定义好自己 Java Bean 类，使用@Autowired 调用 Dao 层中的接口。可处理手机 APP 需要的电子标签选型功能和编辑信息发送功能。同时将信息存入数据库或者传递给 MQTT 服务器以用来发布订阅，实现服务器端内部的通信以及服务器到手机 APP 的通信过程。

使用 Navicat Premium 12 对数据库进行管理，直接可以看到数据库的更新和加入情况，对于调试过程中有很大的帮助。在服务器端的 Mapper 里定义对数据库进行操作的语句。

Mosquitto 服务器是一个 MQTT 代理服务器，作为 Server 端，发布订阅供硬件电路接收。通信基于 MQTT 协议，连接服务器端和硬件端，使硬件端作为 Client 端，定义 MyMQTT 类，使用 QoS1 订阅等级进行通信。整个服务器部分的功能是：手机 APP 端通信的服务器将接收的内容先进行存储到数据库中，同时也作为消息的发布者，将接收到的消息通过主题来发布到 Mosquitto 代理，当有客户端去订阅该主题的时候，Mosquitto 代理将推送消息到客户端，也就是硬件端，

从而实现了整个系统的功能。

3.1.2 手机客户端设计

对于手机 APP 开发需要评估电子标签信息显示电路的性能，对于手机 APP 设置对于功能模块化设计是否符合用于需求，以及功能的实现完成度。对于 UI 界面是否美观，操作是否流畅，在程序语言开发中通过 MQTT 协议连接网络服务端，对 APP 开发进行调试。通用电子标签系统的手机 APP 客户端开发主要采用的是 React Native 框架技术，使用 React Native 原生提供的脚手架工具 React Native-cli 模块来构建通用电子标签系统的基本框架，同时 React Native 框架中提供路由组件 react-navigation 实现在各个界面之间的正常跳转以及配置参数的传输，还有各个界面的整个生命周期的全局控制和管理，并采用丰富的界面 UI 组件对本系统进行规划和布局，通过 npm 方式对通用电子标签系统中的软件资源包进行管理，还依附于开源社区的强大功能，对于一些复杂的常用组件可以直接使用 npm 进行安装依赖，并在通用电子标签系统 APP 开发中直接进行引入使用，借助于第二章提到的 React Native 的优良特性，构建通用电子标签系统平台，从而保证了用户非常友好的体验。同时也使得开发的 APP 客户端软件具备跨平台的特性。手机 APP 客户端软件使用基于主题的订阅发布模式，使客户端使用电子标签显示端能够及时接收手机用户推送的消息[70]。手机 APP 开发功能模块设计如图 3-2 所示。

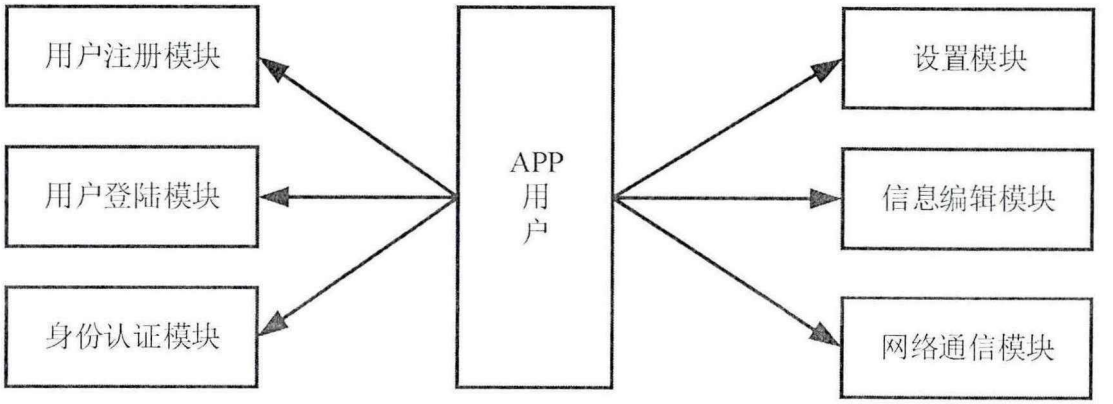

图 3-2 手机客户端总体设计

对于各个模板介绍详细如下[71]。

用户注册模块：用户注册模块主要面向新用户，新用户首次登录时需要进行注册。注册的用户名和密码会存储进入数据库中，在用户第二次登录时，可以通过验证进行登录。这个模块保证安全性的认证，让用户在认证后可以使用相应的服务。用户注册后会提示注册成功，在之后将会对实际效果进行展示，也将测试时出现的结果进行展示。用户注册流程框图如图 3-3 所示，对于首次登录的用户需要注册或者使用开发者的用户名及密码，注册时邮箱未填会进行提示。

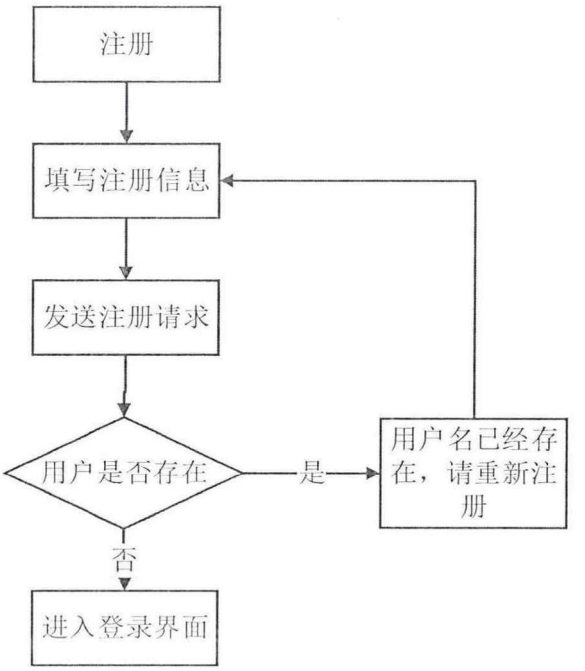

图 3-3 用户注册流程图

用户登录模块：用户注册之后，通过输入正确且合法的用户名和密码进行登录，手机 APP 会把用户信息发送服务器进行对比，若用户名和密码一样，则登陆成功，反之系统提示用户名或者密码错误，需要重新输入。用户登录流程图如 3-4 所示。

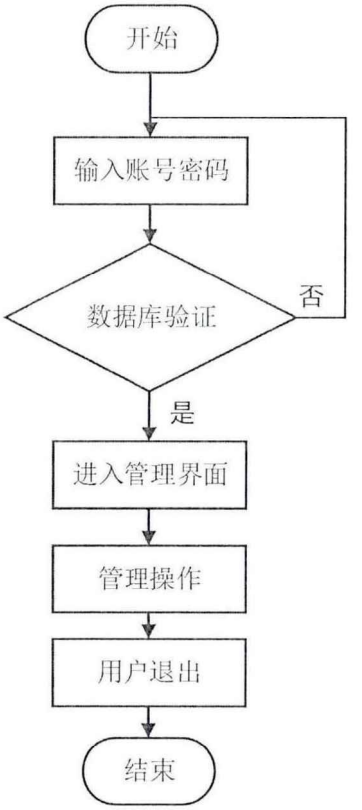

图 3-4 用户登录流程图

身份认证模块：身份认证模块是对个人信息的一种保护，采用 Spring Cloud JTW 机制搭建安全认证模块，在微服务架构下进行信息之间的交互，对用户信息进行安全验证和权限控制。

信息编辑模块：信息编辑模块分两个部分，一个是编辑电子标签的信息，另外一个编辑用户个人信息，两个部分都会存储在数据库中，但是目前只显示一个电子标签的编辑信息。在这个地方尝试了两种方案，第一种是将文字接收后直接显示在电子标签上，但是在大的屏幕上时，演示的效果并不好，所以在大的屏幕上采用了直接显示图片的方法。

网络通信模块：网络通信传输是需要保持手机 APP、服务器和电子标签三者之间建立联系，可以通过制定好的网络协议进行标签信息的交互。

模板设置模块：针对通用电子标签系统，用户可以提前预置几个模板，方便电子标签的快速编辑，有利于用户的长期使用。针对编辑好的模板还可以生成预览样式，看是否符合用户的需求。

日志记录模块：用户每次登录操作都会保留下来，对电子标签信息的修改都有历史记录。手机 APP、服务器和电子标签显示端三部分记录传送的电子标签信息。

设置模块：设置用户个人信息，同时对电子标签模板的信息进行修改，选择适合自己的模板和字体大小。

3.1.3 电子标签显示端设计

电子标签系统硬件包括低功耗控制器系统模块、低功耗墨水屏显示模块、电源管理电路、电子标签系统线性稳压模块、PMOS 开关电路和无线电子标签通信传输模块。

1．低功耗控制器系统模块

(1)主芯片的选择

MSP430[18]系列芯片是德州仪器（TI）公司出品的低功耗处理器，主要满足各种低功耗应用场景，MSP430 系列芯片众多，功耗指标并不完全相同，随着技术的更迭，MSP430 系列芯片的新款 G 系列提高了芯片的处理速度，在 AM（Active Mode）模式下电流超过 F 系列几乎 50μA，功耗指标过差，所以本次设计采用 MSP430F5529 芯片，3.3V 工作模式的电流使用 RAM 存储读写，8MHz 大约在 180μA，睡眠模式（LPM3）时电流大约 2.1μA。同时 MSP430F5529 芯片可选休眠模式更多，时钟也更多，FLL 的时钟选择更多，对于开发来讲适用性更强。同时可提供多个 I/O 输出输入口，可拓展性高[19]。

(2)MSP430F5529 主控程序设计

主控程序设计对于整个系统的低功耗指标来讲至关重要，主控程序的逻辑需要对系统的设计进行适配，完成的功能也要能达到预期。

本论文中的设计是让MSP430F5529在不使用时处于LPM3的工作状态，使用主控芯片的接受中断和定时中断。主控程序的逻辑实现如图3-5所示。

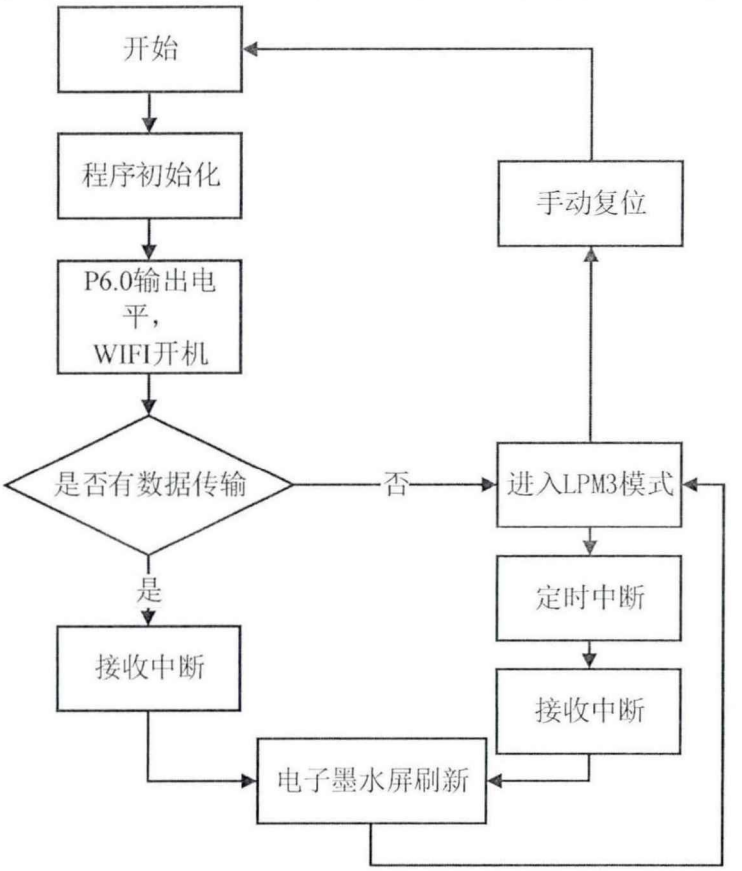

图3-5 MSP430F5529程序控制流程图

2．低功耗显示屏设计

电子墨水屏含有电子墨水胶囊，胶囊内部包含的液体电荷，在不同的电压下可以通过电泳显示技术展现不同的信息。自2007年亚马逊的Kindle阅读器发布以来各种电子纸阅读器层出不穷。国内也出现了以汉王为代表的一大批电子纸品牌，例如翰林，水墨轩等，但是在核心技术方面由于国内起步较晚，相关的品牌大多数都在使用国外的专利，自主知识产权较少，这造成了电子墨水屏成本较高。目前市场上还是以LCD屏为主，还有其他的种类像是OLED屏也有相对较少的市场，随着国内以京东方为代表的显示屏公司的开发技术的进一步发展，OLED屏的使用市场在逐步扩大，有望逐步超过LCD屏的市场份额。电子墨水显示屏具有以下特性[29]

(1)本身功耗低

电子墨水屏可以在没有电源的情况下连续显示画面，只有画面变换的时候，比如翻页时，才需要消耗少量功耗，例如在Kindle已经关机情况下还可以清楚的显示画面。这种特性极大地将降低了电源功耗，这也是电子墨水屏自身的优势之处。

(2)护眼、可视角度大

传统的LCD屏显示原理是利用背光发射，光线需要一直穿过屏幕，由于在不断的刷新，所以出现快速的闪烁，但是以我们的肉眼来看是不能发觉的，但这一客观现象，会导致用眼疲累过度。但电子墨水屏的工作原理是通过自然光照射在显示屏，在折射到肉眼中，这种方式没有了闪烁，一旦屏幕刷新后就不再变化，在长时间的阅读时眼睛也不容易感觉得到疲劳。因此，这种工作原理带来的优势使其在能够更好的适用于电子书阅读器等产品。

(3)本身轻薄

目前，已有的电子墨水屏厚度仅有为1.2毫米，所以我们在对硬件嵌入式设备进行设计时可以考虑可以减轻显示屏的重量，对用户体验度和满意度能够得到更大的提升。

电子墨水屏的显示电路原理图如图3-6所示。

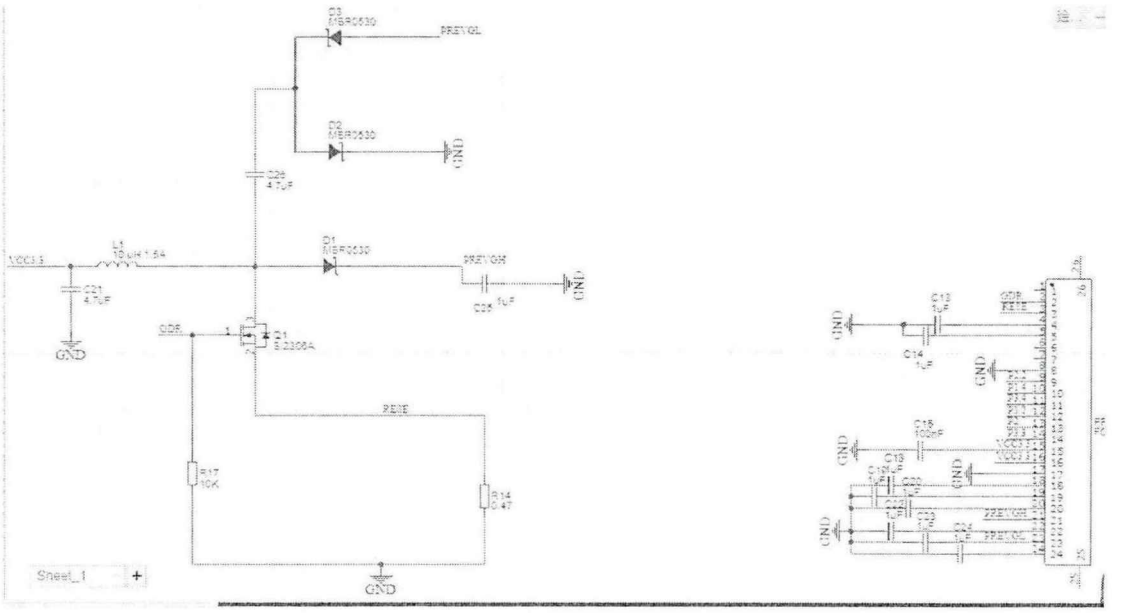

图3-6 电子墨水屏外围电路原理图

3．电子标签低功耗睡眠设计

电子标签虽然断电之后可以使用很长时间，但若需要工作年限长，仍需要设计电路睡眠模式。睡眠模型下，设置一个时间节点，若到了时间节点则电路正常运作，接受APP发送的电子标签信息进行通信传输，APP没有发送标签信息则继续进入睡眠模式。睡眠时间周期为10s，唤醒时间周期为5ms。时序如图3-7所示。

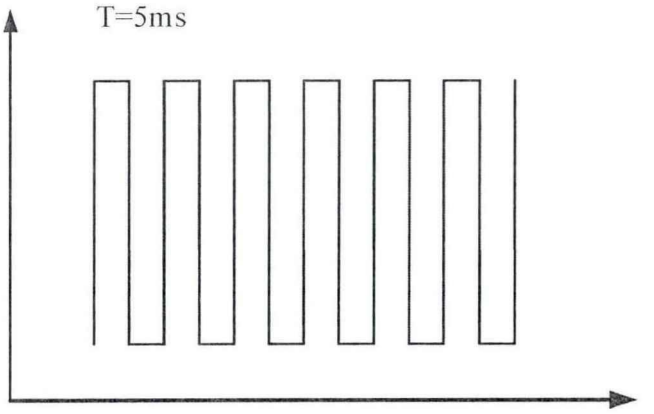

图 3-7 电子标签变更内容时序图

电池电源管理芯片使用的是 TP4056，封装为 SOP-8_EP_150mil，这款芯片的优点是可以保证恒压 4.2V 充电，对于我所准备使用的 3.7V 锂电池可以很好的适配，同时可以在不用外接电源时限制电池的漏电流低至 2μA。原理图如图 3-8 所示。

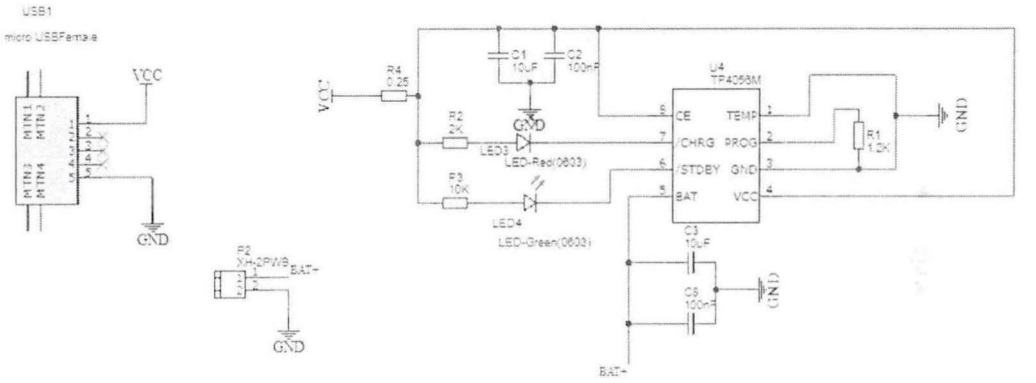

图 3-8 电源管理模块原理图

4．线性稳压模块设计

此模块保证整个电路工作于 3.3V，将电池或者电池电源管理输出的 3.7V 或 4.2V 电压降至 3.3V，保证主控芯片 MSP430F5529 的正常工作以及 WIFI 模块 ESP8266 的正常工作。原理图如图 3-9 所示，C27 和 C28 为滤波电容，跨接电源和地。

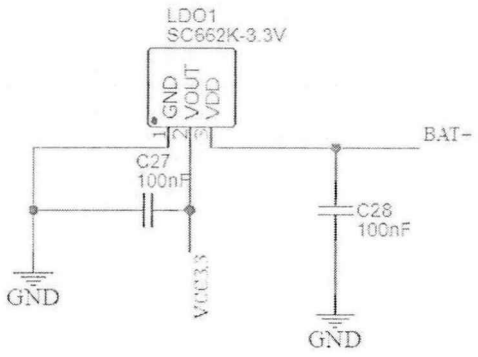

图 3-9 线性稳压模块原理图

5．开关电路模块设计

开关电路模块原理图如图 3-10 所示，正常工作时，两个 P 区与衬底之间的 PN 结均为反偏，漏极的电压 Vds 为负值，栅极对源极的电压 Vgs 也为负值。

当 DS 间加负向电压时，源极和漏极导通，当 Vds 增大漏极电压达到阈值时，沟道在漏极关断。当漏极电压为零时，若 Vgs<0，则当 Vgs 增大到一定值时，PMOS 管导通电路导通。

P 沟道 MOS 管开关电路工作模式高电平断开，低电平导通。Drain 端接后面电路的 VCC 端。

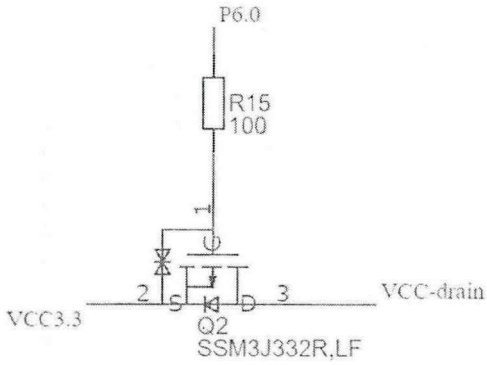

图 3-10 开关电路模块原理图

Vgs>-0.6V 断开 Vg>3.3V-0.6V=2.7V；

Vgs<-1.8V 导通 Vg<3.3V-1.8V=1.5V。

VCC_3V3 即源极接整个电路的 VCC3.3 输入，栅极接 MSP430F5529 的 6.0 管脚，主控芯片通过控制引脚输出高低电平的语句来控制开关导通或者断开。芯片输出高电平为 3.12V，满足关断条件，可以保证 WIFI 断电。

6．WIFI 无线通讯模块设计

目前在物联网应用，WIFI 技术应用最为广泛，其传输距离为 100-300M，传

输速率为 300M/bps 适合通用电子标签的开发系统中[21]。WIFI 技术是通过无线网卡和接收器进行网络传输，无线网卡对网络信号进行转换，接受器是连接无线局域网和无线设备端进行传输的。WIFI 技术具有以下特点：传输速率高，比其他无线传输方式都快；基于工作模型下对于能源的消耗也非常低，由于无线模块的独有的传送方式可以保护信息传输的安全性；WIFI 进行传输时由于通过无线网络和局域网，可以提高稳定性，减少信息传输时的丢失。

WIFI 通讯模块是产品中常用的通讯方式，WIFI 的通信方式是由 TTL 信号和串行通信口进行通信。传统的网络通信方式都是由串口来传输数据的，而 WIFI 通信模块是通过 Uart 接口来实现的，即串口通信 WIFI 模块。本次系统设计的 WIFI 网络通信标准是也是用串口进行传输，基于 IEEE802.11 和 TCP/IP 网络协议进行通信。设置 WIFI 无线通讯需要注意以串口传递为主，供电电压为 3.3V。

现在市场上主流的 WIFI 模块有美国 TI 公司的 CC3200 的 WIFI 模块，联发科的 MT7681 的 WIFI 模块，美国高通公司的 AR9331 的 WIFI 模块和乐鑫公司的 ESP8266 的 WIFI 模块，通过表 3-1 对比这几种型号模块的传输方式、优点和缺点，选出最适用于本系统的 WIFI 模块进行开发。

表 3-1 WIFI 模块芯片对比

序号	型号模块	传输方式	优点	缺点
1	TI CC3200[22]	支持 IEEE802.11 b/g/n 通信协议	外设接口丰富；拥有多种通信协议传输方式；可以进行加密设计工作模式；能耗损失低	电路设计较复杂，且价格昂贵
2	联发科 MT7681[23]	支持 IEEE802.11 n/b/g 无线协议	嵌入式硬件设备，耗能很低	屏蔽了部分代码，仅能做一些简单的远程控制
3	高通 AR9331[24]	本地有线网络转换为 WIFI 热点，实现网络共享；热点信号到 WIFI 热点的转换。	根据用户的不同需求进行板型的结构设计；扩展接口非常丰富；	

4	乐鑫 ESP8266[25]	编程软件编写程序下载到模块，MCU 可以运行应用程序	开发过程简单；作为处理器；进行 WIFI 无线数据的传输；接口丰富与各种嵌入式设备的连接十分方便	

经过对比几种芯片的特点，本系统选用乐鑫的 ESP8266 WIFI 模块，从传输方式和优缺点，根据使用环境，选择 ESP8266 通信模块。

(3)ESP8266 型 WIFI 控制硬件设计

ESP8266WIFI 模块的开发板是 ESP-12E 模块，ESP-12E 模块内部电路包含了定时器电路、下载电路、驱动电路等。它的优点是传输速度快、高速存储和处理大量信息。同时模块的 IO 接口接口多，可以嵌入式设备，方便开发者进行开发[26]。ESP-12E 模块如图 3-11 所示。

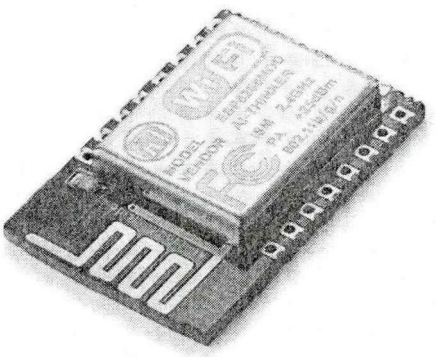

图 3-11 ESP-12E 模块

(4)无线通信方式设计

ESP8266 通信模块的通信方式可以进行加密设置。它的三种模式是 Soft-AP、Station 和兼容模式。提供无线连接服务的是 AP(Access Point)模块，其他终端设备可以通过无线网络进行数据传输，由路由器的无线网络进行连接。在本文的通用电子标签系统中，ESP8266 通信方式采用的是兼容模式。

Soft-AP 模式：在 Soft-AP 下设置 ESP8266 工作模式设置成一个路由器，通过路由器引入无线局域网。若没有路由器，则 ESP8266 就会通过无线网络传输，减少了组成网络的成本。在这种模式下，手机 APP 可以接入无线网络进行互联网访问[27]。通信过程如图 3-12 所示。

图 3-12 Soft-AP 模式通信过程

Station 模式：在 Station 模式时 ESP8266 工作是一种智能终端。模块通过接入无线局域网进行操作，电子标签数据信息会上传到云端服务器。手机 APP 用户可以从云平台上下载电子标签信息数据。同时通过手机 APP 也可以实时监测工作状态并且发送监控指令，数据信息也可以相互交互。通信过程如图 3-13 所示。

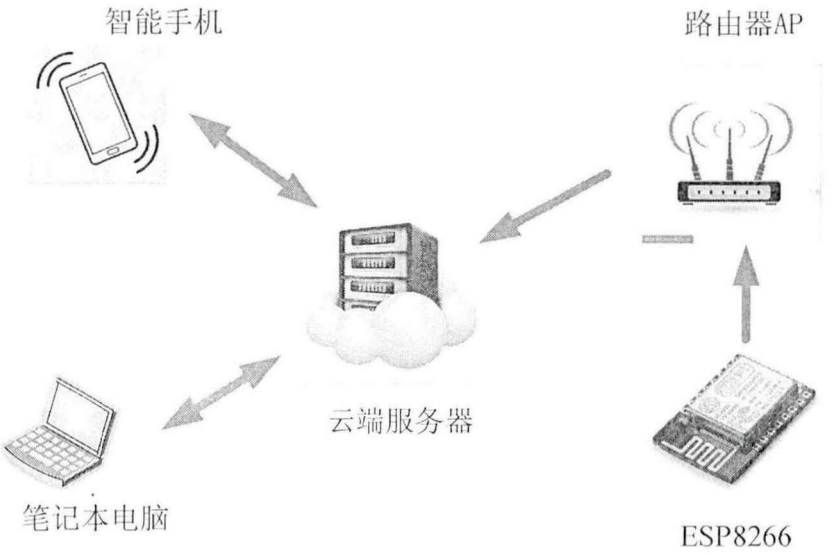

图 3-13 Station 模式通信过程

兼容模式：ESP8266 在兼容模式可以备连接无线路由器，手机设备也可以连接无线网络是本次选择的重要工作模式。在兼容模式下，手机信息可以传送到云平台，与此同时可以直接对云平台进行控制。工作模式下的智能设备需要连接到网络通信中去。如图 3-14 所示为工作模式的设置方法[28]。

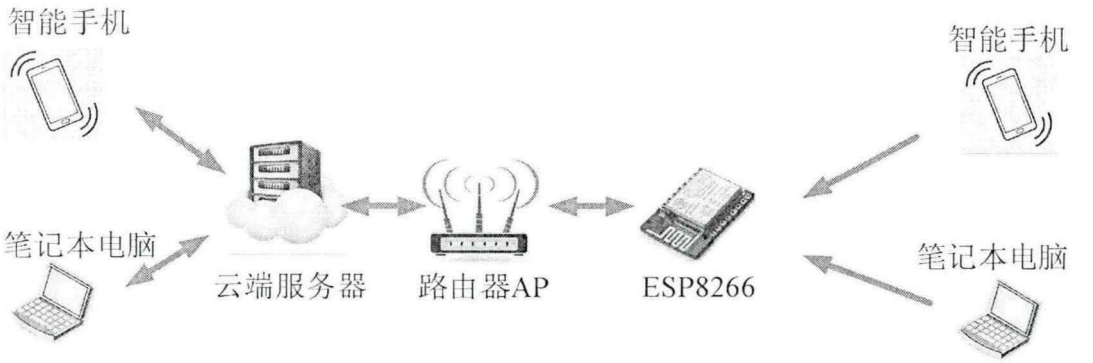

图 3-14 Station + Soft-AP 模式通信过程

(5) WIFI 模块电路原理图及功能

WIFI 模块需要烧录编写的 SDK，根据芯片手册，确定在烧录和运行时的高低电平，这一点对于整个系统的运行至关重要。DIP1 开关在整个系统中是在开发过程中增加的保护开关，FLASH 开关为烧录使用，在成品时可去掉。WIFI 模块原理图如图 3-15 所示。

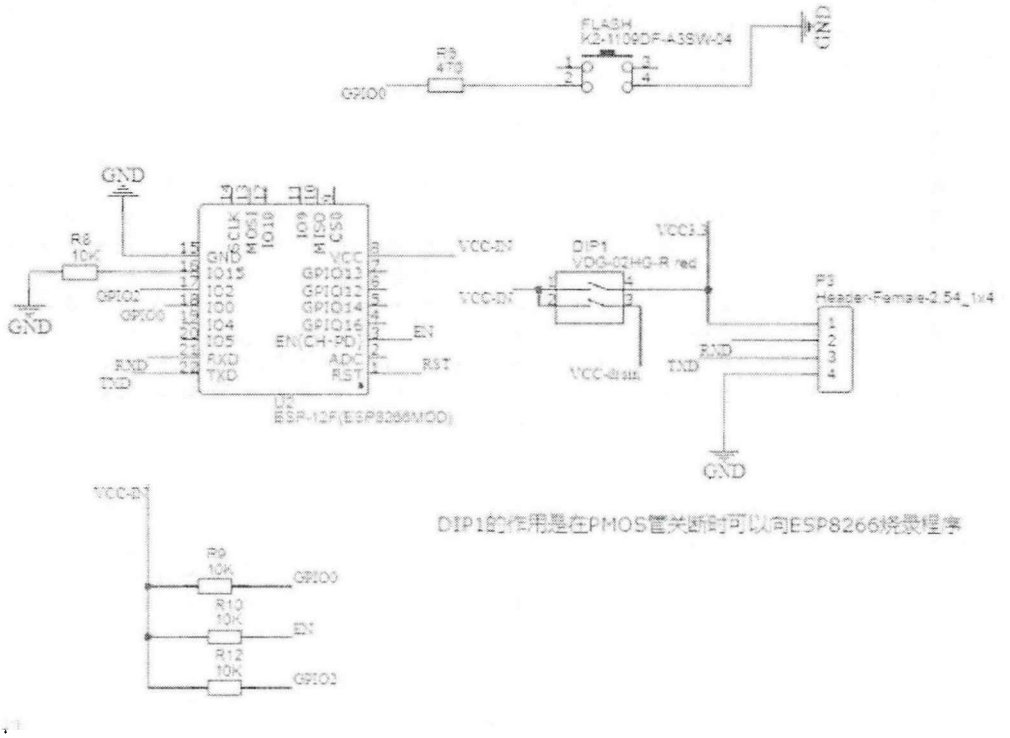

图3-15 WIFI 模块原理图

在开发过程中曾出现无法烧录的情况，经检查是供电电压不稳定的原因，ESP8266 模块需要的 3.3V 电压必须稳定，否则会出现无法烧录的错误。

3.2 系统安全模块设计

在信息进行传递时，电子标签的信息需要进行加密设置，保护电子标签信息

在传递过程中的安全性，所以需要设置网络安全模块。在网络传播过程中需要网络验证机制，当手机用于无线网络中，这时属于活跃网络，手机可以执行获取信息的操作，但网络连接不上 WIFI，客户端会提示需要检查网络或请连接网络后再试[29]。

3.3 通信协议相关设计

3.3.1 通信协议格式设计

系统中需要大量标签信息和用户信息进行交互，所以要首先规定好通信传输协议。根据系统设计要求，手机 APP 端、服务器和电子标签端需要进行数据交互，主要是在 HTTP 协议。在 MQTT 协议下服务器中的代理服务器可以进行标签信息的交互，一般 MQTT 协议通信格式如表 3-2 所示。

表 3-2 MQTT 消息格式

Bit	7	6	5	4	3	2	1	0
Byte1	Message Type(8)					DUP flag	Qos level	RETAIN
	1	0	0	0	0	0	1	×
Byte2	Remaining Length							

3.3.2 数据交互格式设计

手机 APP 和电子标签端在数据交互时采用 JSON 格式，此种格式数据传输短小，此时属于高效传输，加快了系统的升级，用户体验也方便。JSON 格式下是通过集合模式进行通讯，通信过程中键的类型是 String 类型，数据值是字符串、数字和一些简单的类型进行传送。获取电子标签的设备信息为{ device_name：“tags”，position：“dx010101”，number：“000001”，data：“hello world” }。其中，device_name 是设备号，position 是当前电子标签的位置，data 是当前电子标签显示的信息，number 是电子标签的编号。通过这个消息可以获取设备的属性信息。

3.3.3 数据通信设计

数据通信模块是整个手机 APP 端可以与服务器正常连接的基础，就像是前文提到的本论文中手机 APP 端和服务器端通信使用 HTTP 协议。数据传输使用 JSON 格式的数据，但是需要注意的是 JSON 格式的接口数据类型只有 6 种，分别为 Number，String，Boolean，Array，Object，Null。数据通信选用的协议选择要适配整个设计的期望，不能开销过大，造成对手机和服务器的占用过多。

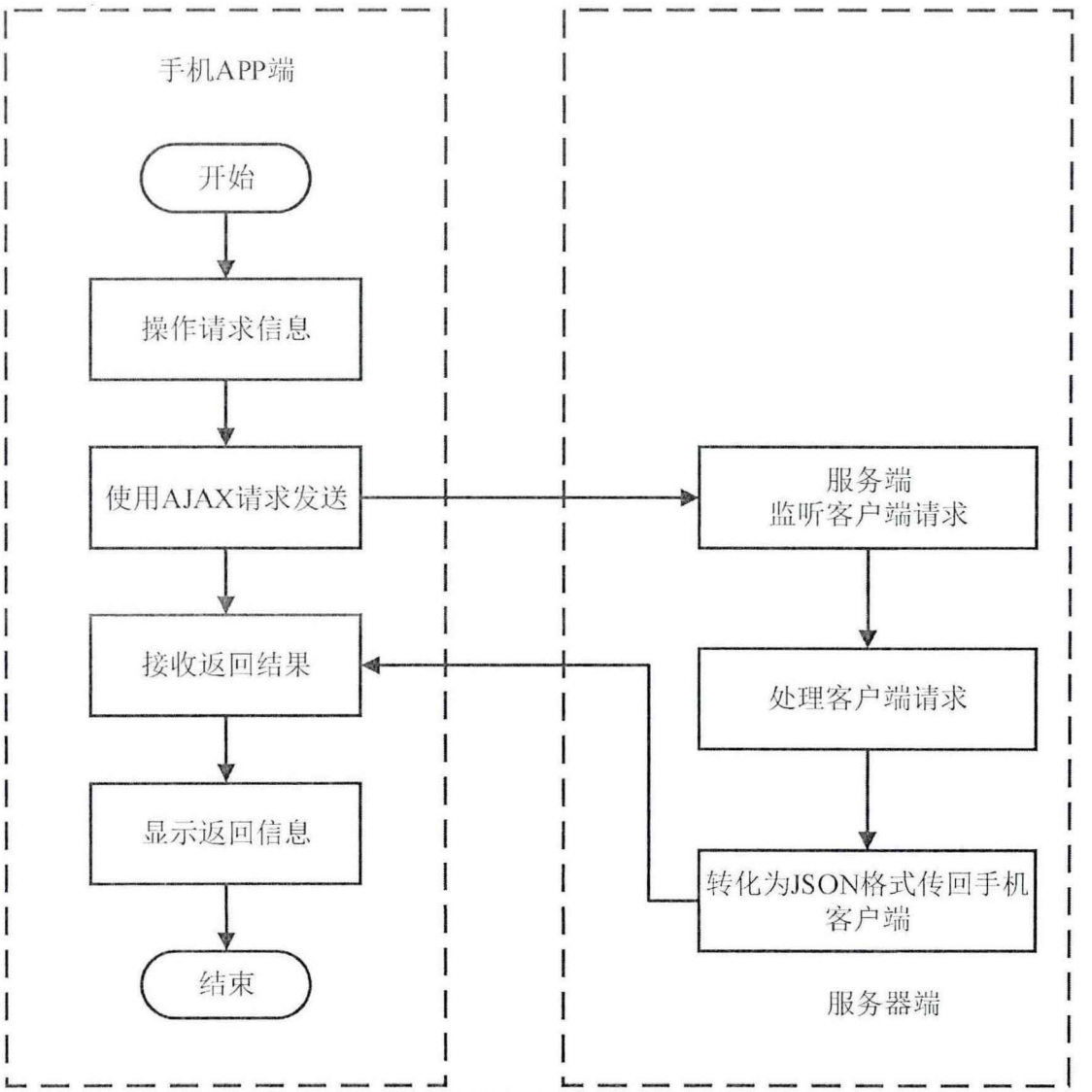

图 3-16 手机 APP 端与服务器端通信及数据传输

3.4 数据库设计

数据库是整个信息系统的核心模块，由于通用电子标签系统设计复杂，需要多种数据库直接进行交互，需要满足以下要求[30]：

(1) 底层设备的多样性。通用电子标签系统有很多电子标签信息，电子标签设备包含不同设备属性值和内容，需要把每一种设备标签拆分开来，这样会增加数据库表，由于手机 APP 用户信息也增加数据库，这时候需要添加新的数据表，一直增加数据表也会造成系统出现错误。

(2) 上层应用操作的可统计性：通用电子标签系统的服务端是一种管理系统，可以对用户信息进行管理，也需要对信息标签需要管理，实时监控信息的动态信息，实行管理员的功能。

根据上述要求，通用电子标签数据库设计需要确认数据库，手机 APP 客户端需要监控电子标签设备的属性状态和标签信息；服务器需要监控手机 APP 的

用户信息，同时监控电子标签设备属性状态以及状态变更信息，及时修改标签信息。从研究对象来看，需要把手机 APP 客户端和电子标签设备端作为监控实体，这样对实体的标识需要共同的属性值。手机 APP 信息包含用户名和密码；电子标签设备包含显示系统标签；服务器也需要权限值和用户名及密码。

3.4.1 电子标签设备信息

如表 3-3 所示，电子标签的信息表包括设备编号，地点两个信息。设备编号为主键，不能为空[31]。

表 3-3 电子标签属性表

字段名	字段标识	数据类型	字符长度	备注
设备编号	dev_id	Varchar	20	主键，不为空
地点信息	Position_id	Varchar	20	不为空

此数据表是为了存储电子标签的信息，在设备选型时可供选择，所以二者的信息都不能为空。

3.4.2 电子标签显示信息

如表 3-4 所示，数据信息表存储的是手机端发来的编辑后的信息。ID 为主键，字段中的主题，名称，备注，以及图片都是能在电子标签中显示的内容，所以需要存储在数据库中。

此表内的信息由服务器端控制更新，当有更新时就会发布订阅，与电子标签选型的信息表存在着绑定关系，当指定了某几个电子标签更新信息时，那么，服务器只会发布相应的订阅，以供硬件 WIFI 模块开机时进行更新操作。

表 3-4 电子标签信息表

字段名	字段标识	数据类型	字符长度	备注
ID	id	Int	10	主键自增
编辑主题	title	Long Text	Max:255	
编辑名称	content	Long Text	Max:255	
备注 1	remark_1	Long Text	Max:255	
备注 2	remark_2	Long Text	Max:255	
备注 3	remark_3	Long Text	Max:255	
图片地址	picture	Varchar	Max:255	
编辑人姓名	name	Tiny Text	Max:255	可为空
编辑时间	edit_time	Timestamp		
编辑人职称	position	Tiny Text	Max:255	可为空
编辑描述	description	Tiny Text	Max:255	可为空
编辑备注	remark	Tiny Text	Max:255	可为空

3.4.3 用户信息

如表 3-5 所示，用户注册登录的信息是用户进入服务的认证信息，以保证用户可以正常使用，使用户可以正常登录。是为了存储每个用户的个人信息，相当于完善了每个人在整个系统中的存在，对于后续开发可通过对每个用户的个人信息进行审核并开放不同的更新权限。用户的密码和用户名也必须保证安全性，防止有不法分子利用窃取的账号发布不良信息，这在之后的继续开发的过程中是必须要保证的。保护信息安全是对用户的负责，也是对设计者自身的负责。由于涉及个人信息存储，所以用户信息的安全必须得到保护，使用更具有安全性的验证机制，防止被劫持出现信息泄露的情况[32]。

表 3-5 用户个人信息表

字段名	字段标识	数据类型	字符长度	备注
用户名	username	Varchar	M<=255	主键，不为空
密码	email	Varchar	M<=255	
邮箱	password	Varchar	M<=255	不为空
姓名	nickname	Varchar	Max:255	
性别	gender	Char	Max:255	
年龄	age	Tiny Text	Max:255	
联系方式	tel	Varchar	Max:255	
个人职称	position	Tiny Text	Max:255	
备注	remark	Varchar	Max:255	

3.4.4 模板信息

如表 3-6 所示，模板信息表存储的是各种不同样式的模板内容，模板内容包括主题，名称，备注，以及存储在服务器端的图片的地址。这样用户可以快捷的编辑电子标签上的信息进行发布显示[33]。

表 3-6 模板信息表

字段名	字段标识	数据类型	字符长度	备注
ID	id	Int	10	主键自增
模板名称	module_name	Varchar	Max:255	
编辑主题	title	Long Text	Max:255	
编辑名称	content	Long Text	Max:255	
备注 1	remark_1	Long Text	Max:255	
备注 2	remark_2	Long Text	Max:255	
备注 3	remark_3	Long Text	Max:255	
图片地址	picture	Varchar	Max:255	
编辑人姓名	name	Tiny Text	Max:255	可为空
模板生成时间	edit_time	Timestamp		

第四章 基于MQTT协议的通用电子标签实现

系统模型中，手机APP端、电子标签显示端和服务器端是通过MQTT协议对电子标签信息进行发布和订阅。服务器分类代理服务器和应用服务器，在数据传输的过程中需要代理服务器，代理服务器为Mosquiito，其可以作为连接的桥梁；应用服务器是在Linux系统中开发的。手机APP客户端是在Windows环境下开发的，数据库采用自带的mysql数据库。

4.1 服务器功能实现

4.1.1 Web服务模块

在通用电子标签系统上工作的Web服务器可以通过其底层的网关设备接口进行数据交互，同时通过底层设备数据包设计参数页面。

Web服务器在调取后端API接口服务就能给用户显示相关页面信息和数据。而其余请求命令比如：用户注册和登陆、配置电子标签设备联网操作、对电子标签进行管理和维护、电子标签信息内容的发布以及个人基本信息的编辑等等，通用电子标签的后端服务系统统一使用Java语言进行编程开发。

在通用电子标签系统上运作Web服务器大体有三个备用方式：Apache，Lighttpd和Nginx。

若Web服务器出现失误，还有备用方式进行操作，主要方式有：Apache、Lighttpd和Nginx三种方式。首先，因其Apache中包含模块组件十分丰富，能够具备十分良好的性能，并且所有的模块代码都能动静态所编译这一优良特性被誉为全世界排名第一的Web服务器。同时它也存在一定的问题和隐患，对于通用电子标签系统来说，采用Apache服务器有以下缺点：一是占有系统的内存空间较大；二是不支持Linux系统内核中的一种可扩展IO事件处理机制，同时将会阻塞前端发来的处理请求；三是对于FastCGI接口和PHP编程方式不能很好的提供服务[34]。

Lighttpd服务器和Nginx服务器的工作方法是一样的，对系统的内存要求特别少，CPU使用率也比较低，所以系统的性能得到了提升。对于Linux系统的开源支持也是十分到位的。但是Lighttpd服务器有着明显的问题，那就是运行不太稳定，可靠性不能得到保障，但系统工作本身有自身的问题对系统也比较大。

最后一个服务器选择是Nginx服务器[35]，它的选择来说也是最好的。它的第一个优点是在系统的占用空间上也较小，只有200多byte；它的第二个优点是高效，Nginx是采用的是异步的方式接收请求机制，这样大大的降低了在高并发情况下能源的消耗；它的第三个优点是能够很好兼容FastCGI接口和PHP编程语

言；它的第四个优点就是相对 Lighttpd 服务器来说将更加稳定，不容易产生较多的系统问题。

Nginx 服务器是一种静态工作模式，服务器是以 PHP 或者 Java 语言进行编程，与 HTTP 协议结合起来进行传输，它们之间添加 FastCGI 或者 Servlet 可以保障 Nginx 服务器接受发布请求之后，可以保证通用电子标签系统的服务器正常运行，本次系统设计是采用 Servlet 来作为 Java 后端程序的容器。

4.1.2 数据交互模块

数据交互模块是电子标签和手机 APP 客户端之间，手机 APP 客户端和服务器，以及电子标签和服务器之间的标签信息交互需要数据之间进行正常操作。这几个数据库和读写相关的模块可以实现系统 API 接口之间进行数据交互。

(1)电子标签内容的变更 API：changedTagsContent

该函数处理的是由手机 APP 客户端上传的电子标签信息内容，电子标签显示端就会进行变更，如表 4-1 所示。

表 4-1 接口 changedTagsContent 描述

int changedTagsContent(@RequestBody Tag tag)	
参数	tag：电子标签中所包含的全部信息
返回值	数值为 0 表示数据库执行失败 数值为 1 表示数据库需要更新，操作执行成功
执行与判断	1：检查该电子标签设备在数据库表 DeviceInfo 中是否存在，如果存在，转步骤 2；否则返回 0。 2：执行更新操作事务：更新数据库表 DeviceInfo 中设备的显

(2)添加模板内容 API：addModuleContent

该函数处理的是由手机客户端上传的模板信息的添加操作，如表 4-2 所示。

表 4-2 接口 addModuleContent 描述

int addModuleContent(@RequestBody Module module)	
参数	module ： 模板中所包含的全部信息
返回值	数值为 0 表示数据库执行失败 数值为 1 表示数据库可以进行添加，操作执行成功
执行与判断	步骤:向数据库中的表 moduleInfo 插入模板内容。

(3)删除模板内容 API：deleteModuleContent

该函数处理的是由手机客户端对模板进行的删除操作，如表 4-3 所示。

表 4-3 接口 deleteModuleContent 描述

int deleteModuleContent(String modulename)	
参数	modulename ： 模板名称
返回值	数值为 0 表示删除数据库相关表项失败； 数值为 1 表示数据库删除操作执行成功。
执行与判断	步骤：执行删除事务：删除数据库表 moduleInfo，如果失败返回 0，成功则返回 1。

(4)用户注册 API：addUser

该函数处理的是由手机客户端进行的用户注册操作，如表 4-4 所示。

表 4-4 接口 addUser 描述

int addUser(Person person)	
参数	person：注册用户包含的基本信息
返回值	数值为-1 表示数据库中数据用户名重复 数值为 0 表示数据库插入操作执行失败 数值为 1 表示数据库插入操作执行成功
执行与判断	1:检查数据库表 User 中是否与 person 结构体中的 username 有重复，若有则返回-1；否则转下一步。 2：向表 User 中插入表项，若插入成功返回 1;否则返回 0。

(5)删除用户 API：deleteUser

该函数处理的是用户删除操作，如表 4-5 所示。

表 4-5 接口 deleteUser 描述

int deleteUser(String username)	
参数	username：用户名
返回值	数值为 0 表示数据库删除操作执行失败； 数值为 1 表示数据库操作执行成功。

执行流程	1：检查数据库表 User 中是否与 person 结构体中的 username 是否存在，若不存在则返回 0；否则转下一步。 2：执行删除事务：删除表 User 中该用户的表项，若失败则返回 0，成功则返回 1。

(6)用户信息变更 API：changedUserInfo

该函数处理的是手机客户端上传的用户信息，如表 4-6 所示。

表 4-6 接口 changedUserInfo 描述

int changedUserInfo(Person person)	
参数	person：用户名需要修改的基本信息
返回值	数值为 0 表示数据库操作失败 数值为 1 表示数据库更新操作事务执行成功
执行流程	1：检查数据库表 User 中是否与 person 结构体中的 username 存在，若没有则返回 0；否则转下一步。 2:执行更新操作事务:更新数据库表 User 中用户的基本信息，如果事务执行成功，返回 1，否则返回 0。

4.1.3 信息处理模块

信息处理模块是系统服务端的核心模块，控制手机客户端用户的基本信息。电子标签内容的更新也都是单独方式进行控制。其主要通过 MQTT 通信线程和信息处理线程来实现，MQTT 通信线程是服务器和手机 APP 端进行交互，传输用户的信息和电子标签信息。信息处理线程需要处理标签信息进行编辑、修改和转发[36]。

信息处理模块是在 Linux 系统上进行编程，后端设计由用 Java 语言编写。系统开发是通过 MQTT 通信线程进行发布，当接收信息时，会以报文形式进行，会按照话题创建新的信息处理线程，也就是说信息处理线程可能不止一个，其过程如图 4-1 所示。

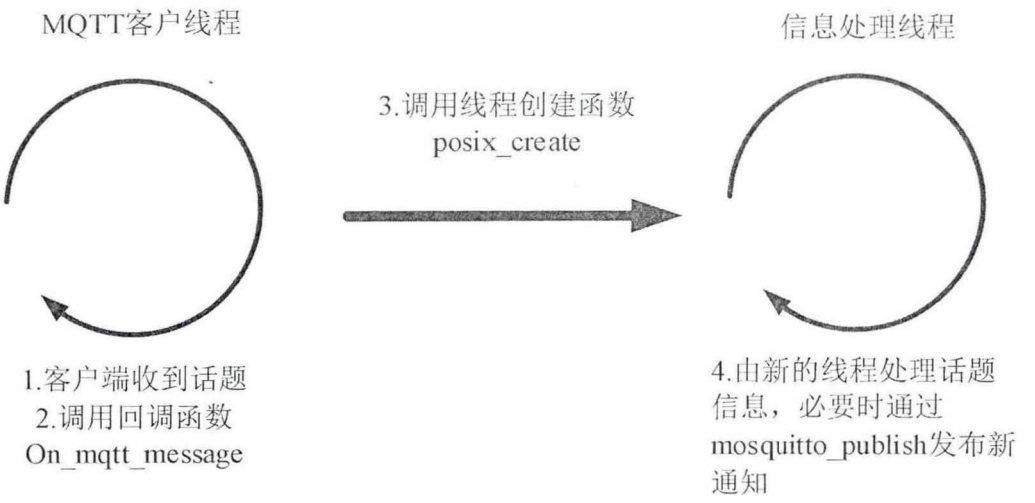

图 4-1 信息处理进程中线程交互图

4.1.4 Mosquitto 通信模块

Mosquitto 通信模块对代理 MQTT 服务器进行正常运作和配置正确参数。通用电子标签系统选用 Mosquitto 开源代理服务器。本小节将从三个方面仔细阐述下 Mosquitto 服务模块的配置设计：话题设计、安全机制和配置。

(1)话题设计

MQTT 客户端首先与代理服务器建立正常的连接之后，便能向指定的话题发布消息内容，MQTT 代理服务器将把消息体的内容推送到已经订阅该主题的 MQTT 客户端，这是 MQTT 消息传输的形式，在通用电子标签系统中，进行 MQTT 协议网络通信交互的实体类共有四类：服务器端的数据交互模块，手机移动用户 APP 端，电子标签显示端。

(2)Mosquitto 服务器的安全机制

MQTT 协议本身在网络通信方面，有着各种加密形式。Mosquitto 服务器的安全机制有三个重要定义：身份标识、认证请求、授权控制。身份标识指的是 MQTT 代理服务器可以通过的 ClientID、UserID 和 SSL 证书这三个中一个或者多个来验证它的具体标识。Mosquitto 服务器将会通过以下两种方法进行身份标识验证：一种方法是通过安全传输协议进行传输的 SSL 证书验证；第二种方法是通过含有秘钥的 ClientID 标识来进行验证。认证请求指的是 Client 和 Mosquitto 双方识别其身份标识：Client 通过 TSL 的方式来识别 Mosquitto，Mosquitto 通过 SSL、密钥或者两者都含有的方式来识别 Client。由于授权控制并不属于 MQTT 协议中的一分子，MQTT 协议只是承载着在网络环境中实体之间消息体的传输和交互操作，所以授权控制只由 Mosquitto 来进行的全局管控。

(3)Mosquitto 服务的安全配置

通用电子标签系统使用的 MQTT 服务的基本步骤是：第一步是以 ClientID 来识别 Client，因为 ClientID 是唯一身份标识码，这个唯一性将由系统得到保证；

第二步是在物联网云管理平台计算能力有限的情况下，将采用预共享密钥的安全套接层协议来作为网络通信的安全措施，同时也能提高通用电子标签系统的实时更新性；第三步由 Mosquitto 服务器进行授权控制，电子标签显示端设备使用授权控制则由服务器的管理中心进行维护。Mosquitto 的运行配置文件 mosquitto.conf 中便包含了 Mosquitto 服务器所运作的基本参数配置。下面主要更加关注安全方面的配置，如表 4-7 所示。

表 4-7 Mosquitto 安全方面的配置

配置名称	设定值	意义
psk_hint	mobisys_smart_home	当定义该属性时，Mosauitto 的监听进程默认使用基于预共享密钥的安全传输层通信协议。
use_identity_as_username	false	本系统使用客户端用户名作为认证基础
ciphers	AES	设定发布服务器支持的加密方式
allow_anonymous	false	是否允许接受匿名客户端

4.2 手机客户端功能实现

手机客户端应用软件开发过程中 APP 的 UI 设计是重要的一环，决定了用户的体验感。本系统中，当用户通过客户端登陆后，会进入终端软件的主界面。主界面是集成配置模式、工作模式、个人信息设置以及客户端设置等模块。根据以上章节对手机客户端的分析及设计，本章节对手机客户端功能进行实现。

4.2.1 注册和登陆功能

用户从应用商店找到 APP 软件进行下载，用户打开界面提示登录，若没有账号，需要去注册用户，如图 4-2 所示；若用户已有账号，需要输入账号和密码就可以登录，如图 4-3 所示。

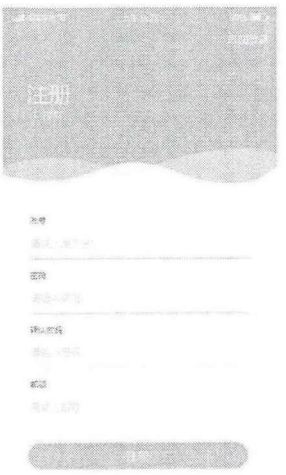

图 4-2 用户注册界面

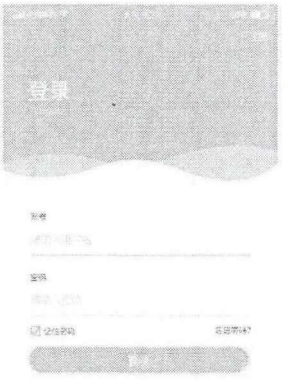

图 4-3 用户登录界面

用户通过注册用户名和密码，可以直接登录。用户注册信息会在SharedPreference文件中进行记录，手机APP在开发过程中需要用SharedPreference来储存用户名和密码。调用Context.getShreadPreferencesa(String name，int mode)，/data/data/包名/shared_pref目录下，以name命名创建一个的xml文件中，选择mode的值为MODE_PRIVATE，只有这样才能确保用户名和密码的安全性能。服务器也可以接受用户注册的信息和已经存在的用户信息，方便进行系统进行监控和查询。

4.2.2 个人信息编辑功能

用户通过输入正确用户和密码在APP端的我的工作栏修改自己的基本信息同时个人信息可以存储在数据库中但不显示。设置主界面和信息编辑界面分别如图4-4和4-5所示。

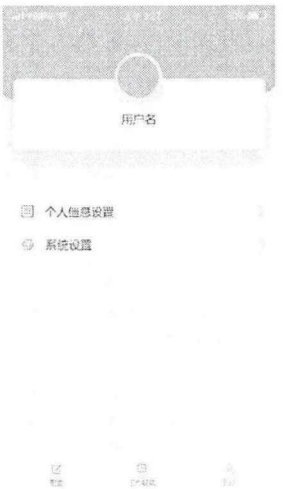

图 4-4 设置主界面

图 4-5 个人信息编辑界面

这一部分主要是为了安全起见，记录每一个人的更改记录，在之后出现问题的时候可以对相关人员进行追责。保证更改的可溯源性。

4.2.3 设备选型和编辑功能

对多终端的开发与实现，目的是可以指定电子标签显示屏将实时更新数据，使得并不是所有的电子标签显示都更新数据，这是为了用户的使用方便而着想，快捷方便的修改电子标签上的信息，比如更改某种商品的价格或者病人的病例报告。设备选型和信息编辑界面如图 4-6 所示。

图 4-6 设备选型和信息编辑界面

4.2.4 模板列表功能

针对通用电子标签系统，手机智能终端用户可以在界面上提前预置生成几个模板，方便日后对电子标签的快速编辑，有利于用户的长期使用。针对编辑好的模板还可以生成预览样式，看是否符合用户的需求。模板列表和编辑界面如图 4-7 所示。

图 4-7 模板列表和编辑界面

4.2.5 客户端设置功能

客户端设置功能模块主要是用来设置用户信息和系统设置。用户端通过登录 APP。在 APP 主界面中进行用户个人基本信息的变更和系统 APP 基本设置。用户能够及时更新自己的个人信息以及按自己需要更换界面的字体样式大小。

第五章 系统测试

系统测试是评价开发产品的性能好坏的体现，主要对手机 APP 客户端和电子标签显示电路进行测试。手机 APP 端需要测试开发的软件功能是否可以正常运行，电子标签显示是否可以正常显示。

5.1 手机客户端功能测试

系统设计问题可以转换成数学问题，通过数学公式可以将问题转换成从输入定义域取值映射到输出值域方式，由输出值可以判断出系统设计的性能问题。本文对系统各个功能进行测试。测试结果如表 5-1 所示。

表 5-1 功能测试结果

功能	用例描述	测试结果	是否合格
注册	输入合法用户名和密码	注册成功，并跳转登录界面	合格
	输入已注册过的用户名	主界面提示用户已被注册	合格
登录	输入正确的用户名和密码	登录成功，并进入主界面	合格
	输入正确的用户名和错误密码	主界面提示用户密码错误	合格
	输入错误的用户名	主界面提示用户不存在	合格
设备列表	查询所有电子标签在线数量	主界面返回可查看的电子标签列表	合格
电子标签发布	输入需要显示的主题、名称、备注，还可以插入图片等	主界面能预览生成的显示样式	合格
模板列表	查询所有的模板数量列表	主界面显示已有的模板列表数量	合格
添加模板	输入需要的主题、名称、备注等，还可以插入图片等默认样式	主界面的模板列表数量+1，信息和添加的一致	合格
修改模板	输入模板编号和想修改的模板内容	刷新该模板，内容以及修改成功	合格
删除模板	输入模板编号	模板列表中已经移除当前模板	合格

个人信息设置	输入想修改的个人信息	再次刷新个人信息已经修改成功	合格
字体设置	输入想调节的字体大小	界面的所有字体大小都已经改变	合格

5.2 电子标签显示测试

5.2.1 显示结果测试

我们分别从三个场景进行测试，分别选择超市、医院和会议标签。在超市场景中，在手机 APP 模板一中，通过对信息进行编译，在电子水墨屏端显示结果如图 5-1 所示。

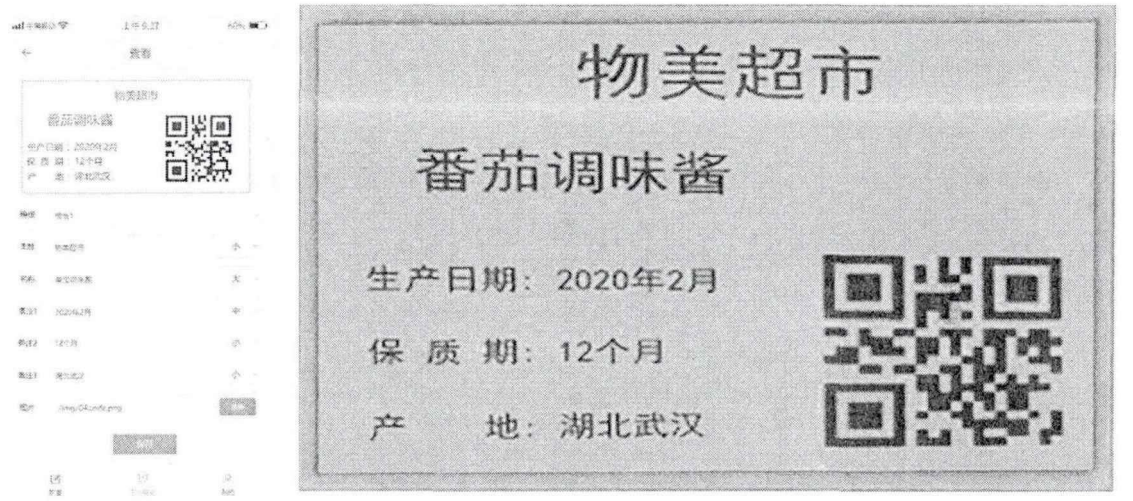

图 5-1 超市标签

依次使用模板二和模板三，对会议标签和医院标签进行结果测试，结果如图 5-2 和 5-3 所示。

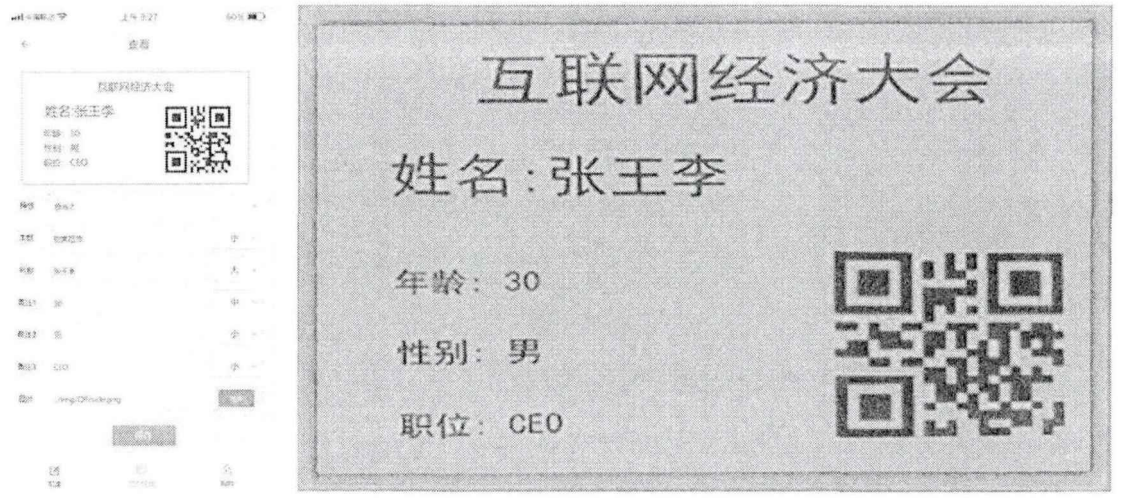

图 5-2 会议标签

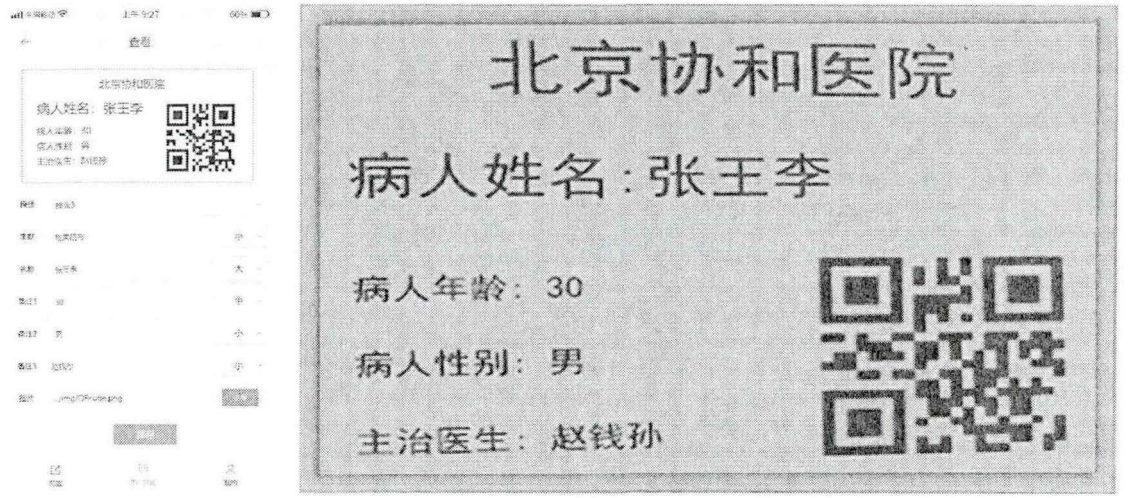

图 5-3 医院标签

5.2.2 显示功耗指标测试

本次统计分为工作和动态两个数据。为直观反映功耗参数，功耗数据用mAh表示。测试结果分别如表 5-2 和 5-3 所示。

表 5-2 硬件电路静态工作参数

电源电压	模块名称	测量次数（次）	电流平均值（μA）	功率损耗（每天）/（mAh）
3.3V	电子墨水屏	3	0.01	2.4×10^{-4}
3.3V	MSP430F5529	10	2.4	5.76×10^{-2}
3.3V	WIFI 模块	10	1.1	2.64×10^{-2}
总计			3.51	8.4024×10^{-2}

可以看出，硬件电路静态功耗平均值为8.4024×10^{-2}mAh每天，按照所用的4200mAh锂电池，该电路理论上可以满足设计之初的3年目标，但是电路不是一直处于静态状态，所以需要综合动态参数来计算。

表 5-3 硬件电路动态工作参数

电源电压	模块名称	测量次数（次）	电流平均值（mA）	能量损耗（每天）/(mAh)
3.3V	电子墨水屏	3	8	192
3.3V	MSP430F5529	10	0.18	4.32
3.3V	WIFI 模块	10	170	4080
总计			178.18	4276.32

在电源电压3.3V时，整个电路可以正常工作。但是静态参数值相较理论值偏高，可能是由于进入芯片的睡眠模式时，电路有额外损耗，但是符合产品需求，综合电路的动态参数来看，每次数据传输应尽可能的缩短时间，定时刷新的话，

每12 个小时自动刷新一次；手动刷新的话，用户不要频繁的刷新电子标签，这样可有效的提高整个电路的使用寿命，功耗指标接近预期，也是可以接受的。

第六章 总结与展望

本论文首先介绍了目前无线电子标签系统的国内外发展现状，讨论了基于MQTT 协议的通用电子标签系统带来的深刻影响以及巨大变革，通过介绍了MQTT 协议特点分析、电子墨水屏技术特点、微服务架构原理、REST 和 Reat Native 等相关技术性的描述。接着对通用电子标签系统进行总体需求分析设计，通过需求分析制定合适的总体系统架构，然后对手机 APP 端、服务器端和电子标签显示端和系统服务端的数据库进行了详细设计。在完成了系统总体设计和详细设计之后，紧接着对系统服务端、手机 APP 端、电子标签显示端介绍了实现，对于系统服务端的实现主要讲述 Web 服务模块，MQTT 服务器信息交互以及数据库交互操作接口设计的实现；对于手机 APP 端的实现讲述了各功能模块以及界面设计的实现；对于电子标签显示端的实现主要讲述了低功耗显示屏模块和WIFI 通信模块的功能实现。最后本课题对系统功能进行了各项测试，测试结果均达到了预期结果。

另外该系统还可以适用于很多场景之下，用户可以通过 APP 端方便管理和编辑电子标签。但本系统还有很多方面需要进一步完善和提升，主要包括以下几个方面：

(1)如何增加无线电子标签系统的功能？可以增加传感器元件，拓展整个系统的硬件部分，目前所设计出的硬件电路只是初步满足低功耗指标，但是后续可以在此基础上增加，例如 PM2.5，甲醛传感器等传感器，因为这个类传感器数值的变化一般不会突变，对于系统可以集成上去，在主控进入休眠时也可以将传感器关闭。并不会造成更多的能源损耗，也可以满足要求较高的低功耗要求。

(2)针对不同的使用场景进行服务器端二次设计。对服务器端进行对支撑更多业务的开发设计，后续部署在云端进行大规模接入测试，对相应的功能进一步完善，支撑整个业务需求，对未来的大规模试用进行可行性的测试。对服务器完成的功能进行进一步的优化设计。

(3)手机 APP 功能进一步优化，UI 设计优化由于时间的原因，本次开发出来的 APP 界面粗糙，虽然符合用户的操作便的要求，但是整体上来说 UI 设计需要进一步改进，手机功能需要添加选择字体，需要服务器端将相应的信息进行编码，在 MQTT 协议的基础上进行数据的传输。对使用的协议进行开发论证对使用的协议进行自己的开发，保证安全性和开销的平衡。

参考文献

[1] 浩钿, 王猛, 陈聪,基于无线通信和墨水屏的电子标识技术研究[J]. 无线互联科技, 016(009):3-5.

[2]张志永. 电子墨水技术在智慧城市应用研究[J]. 工业控制计算机, 2016, 029(003):106-107

[3]Joseph A M, Nagendra B, Bhoje Gowd E, et al. Screen-printable electronic ink of ultrathin boron nitride nanosheets[J]. Acs Omega, 2016, 1(6): 1220-1228.

[4]Gelbman A. Smart electronic label employing electronic ink: U.S. Patent 6,753,830[P]. 2004-6-22.

[5]姚丹, 谢雪松, 杨建军, et al. 基于 MQTT 协议的物联网通信系统的研究与实现[J]. 信息通信, No.159(3):39-41.

[6]杨鹏. 基于MQTT协议的信息推送平台系统的设计与实现[D]. 电子科技大学.

[7]贾军营, 王月鹏, 王少华. 基于 MQTT 协议 IM 的研究和实现[J]. 计算机系统应用, 2015(7):9-14.

[8]Shiva Subhashini Pakalapati, G. Govardhana Chary, Atul K. Yadaw. A prosthetic hand control interface using ESP8266 Wi-Fi module and Android application[C]// 2017 4th International Conference on Innovations in Information, Embedded and Communication Systems. IEEE, 2017.

[9]贾军营, 王月鹏, 王少华. 基于 MQTT 协议 IM 的研究和实现[J]. 计算机系统应用, 2015(7):9-14.

[10]许金喜, 张新有. Android 平台基于 MQTT 协议的推送机制[J]. 计算机系统应用, 2015, 24(1):185-190.

[11]顾亚文. 基于 MQTT 协议的通用智能家居系统设计与实现[D]. 西安电子科技大学.

[12]Terry A A. Electronic ink technologies: showing the way to a brighter future[J]. Library Hi Tech, 2001.

[13]洪华军, 吴建波, 冷文浩. 一种基于微服务架构的业务系统设计与实现[J]. 计算机与数字工程, 2018, 46(1):149-154.

[14]米沃奇. 深度剖析微服务架构的九大特征[J]. 电脑知识与技术：经验技巧, 2016(10):105-110.

[15]吴振宇. 基于 Web 的物联网应用体系架构和关键技术研究[D]. 北京邮电大学, 2013.

[16]薛峰, 梁锋, 徐书勋, et al. 基于Spring MVC框架的Web研究与应用[J]. 合肥工业大学学报(自然科学版), 2012, 35(3):337-340.

[17]冯博. 基于 React Native 框架的兴趣社区 Android 客户端设计与实现[D]. 哈尔滨工业大学.

[18]刘爽, 史国友, 张远强. 基于 TCP/IP 协议和多线程的通信软件的设计与实现[J]. 计算机工程与设计, 2010, 31(7):1417-1420.

[19]杨家炜. 基于 Spring Boot 的 web 设计与实现[J]. 轻工科技, 2016(7):86-89.

[20]杨鹏. 基于 MQTT 协议的信息推送平台系统的设计与实现[D]. 电子科技大学.

[21]潘婷婷. React Native 在 APP 开发中的应用研究[J]. 无线互联科技, 2016(19):142-143.

[22]范国闯,钟华,黄涛,冯玉琳. Web 应用服务器研究综述[J]. 软件学报(10):1728-1739.

[23]王阅蓁. 移动应用的 web 与 native 混合编程模式研究与实现[D]. 电子科技大学.

[24]张凯, 陈峰, 杜警,等. 城市公交实时位置手机查询系统设计[J]. 计算机工程与科学, 2014, 36(7):1296-1300.

[25]黄刚, 雷小燕. 电子标签的超低功耗应用设计[J]. 电子测量与仪器学报, 2010, 24(10):979-984.

[26]Wang Y, Hu Y. Design of electronic shelf label systems based on ZigBee[C]//2013 IEEE 4th International Conference on Software Engineering and Service Science. IEEE, 2013: 415-418.

[27]陈东坡, 何乐年, 严晓浪. 一种低静态电流、高稳定性的 LDO 线性稳压器[J]. 电子与信息学报, 2006, 28(8).

[28]王磊, 周慧, 蒋国平. 基于 WiFi 的自适应匹配预处理 WKNN 算法[J]. 信号处理(09):31-38.

[29]严萍, 张兴敢, 柏业超, et al. 基于物联网技术的智能家居系统[J]. 南京大学学报：自然科学版(01):33-39.

[30]Kodali R K. An implementation of MQTT using CC3200[C]//2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT). IEEE, 2016: 582-587.

[31]杨东轩, 刘硕, 王嵩. 低功耗电子号牌在排队系统中的设计与实现[J]. 计算机应用与软件, 2018, 35: 12.

[32]Chichao L, Jiaqi G, Jingping R, et al. A wireless mesh network based temperature

monitoring system for blast furnaces[C]//2017 13th IEEE International Conference on Electronic Measurement & Instruments (ICEMI). IEEE, 2017: 265-269.

[33]Kodali R K, Mahesh K S. A low cost implementation of MQTT using ESP8266[C]//2016 2nd International Conference on Contemporary Computing and Informatics (IC3I). IEEE, 2016: 404-408.

[34]Azmi, Noraini, Sudin, Sukhairi, Kamarudin, Latifah Munirah, 等. Design and Development of Multi-Transceiver Lorafi Board consisting LoRa and ESP8266-Wifi Communication Module[J]. IOP Conference Series: Materials Science and Engineering, 318:012051.

[35]陈文宇. 面向对象的关系数据库设计[J]. 电子科技大学学报, 2002, 31(1):53-56.

[36]于金刚, 耿云飞, 杨海波. 基于 MQTT 协议的消息引擎服务器的设计与实现[J]. 小型微型计算机系统, 2016, 37(10):2238-2

致谢

本论文是在导师的悉心指导下完成的，从论文的选题到论文的撰写，无不渗透着导师的心血，我的硕士论文即将完成，我的学生生涯也即将结束，在此我要衷心地感谢在研究生学习期间所有给予我关心和帮助的人。

首先在这里我要感谢我的导师孙文生老师，在我历时将近两个月的毕业设计以及论文的撰写过程中，老师给我提供了很多的宝贵意见，让我对于论文的写作更加的严谨，老师严肃的科学精神，严谨的治学态度，丰富渊博的学识，敏锐的学术思维，精益求精的工作态度以及侮人不倦的师者风范，为我营造了一种良好的精神氛围，使我在生活和学习中受益颇多。在实验室的三年中，从研究方向到项目实践，孙老师都给予了详细的指导和不懈的支持，在此谨向孙老师致以诚挚的谢意和崇高的敬意。我相信，在以后的工作和生活中，孙老师的教诲仍将是指引我人生的灯塔。

在此，还要感谢身边所有的同学和朋友。感谢他们在生活中对我的无私帮助，是你们在我论文设计过程中与我一起探讨问题，并提出了很多宝贵的意见和想法，使我能及时的发现问题并做出改进，在此表示深深的谢意。

借此机会，特别感谢含辛茹苦培养我长大的父母以及我的家人，感谢他们在研究生三年来对我的鼓励和支持，他们的理解和支持是我在求学阶段能够克服重重困难的重要因素。

最后，衷心感谢在百忙之中评阅论文和参加答辩的各位专家、教授。