

**本科生毕业设计**

**基于手持终端的门禁远程鉴权装置的软件设计**

**学 院 信息工程学院**

**专 业 物联网工程**

**班 级 2015级本科1班**

**学 号 5006150019**

**学 生 姓 名 闫英杰**

**联 系 方 式 18790565355**

**指 导 教 师 王奎甫 职称：教授**

2019年5月

**独 创 性 声 明**

本人郑重声明：所呈交的毕业论文（设计）是本人在指导老师指导下取得的研究成果。除了文中特别加以注释和致谢的地方外，论文（设计）中不包含其他人已经发表的研究成果。与本研究成果相关的所有人所做出的任何贡献均已在论文（设计）中作了明确的说明并表示了谢意。

签名：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

　　　　　　　　　　　　　　　　　　　\_\_\_\_\_\_\_\_年\_\_\_\_\_\_月\_\_\_\_\_\_日

**授权声明**

本人完全了解许昌学院有关保留、使用本科生毕业论文（设计）的规定，即：有权保留并向国家有关部门或机构送交毕业论文（设计）的复印件和磁盘，允许毕业论文（设计）被查阅和借阅。本人授权许昌学院可以将毕业论文（设计）的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编论文（设计）。

本人论文（设计）中有原创性数据需要保密的部分为（如没有，请填写“无”）：

学生签名：

　　　　　年　　月　　日

指导教师签名：

　　　　　　　　　　　　　　　　　　　　 　 年　　月　　日

基于手持终端的门禁远程鉴权装置的软件设计

摘 要

为了解决数量庞大的一体化通讯基站钥匙管理繁琐的问题，提出一种基于手持终端的门禁远程鉴权装置。

本论文主要提出了基于手持终端的门禁远程鉴权装置的软件设计方案。该方案基于apache mina网络应用框架。采用GPRS通信技术，借助微信公众平台，使装置和微信用户与服务器进行数据通信。服务器程序接收客户端发来的数据，根据数据中用户和装置的身份标识与数据库中的数据对比进行信息鉴权。若鉴权通过，给装置发送开门指令，若不通过，则反馈给用户鉴权失败。由此实现数据交互和远程信息鉴权。

关键词： 信息鉴权；物联网基站门禁；数据过滤；认证秘钥

**Software Design of Remote Access Control Authentication Device Based on Handheld Terminal**

**ABSTRACT**

This paper put forward a Access Control Remote Authentication Device Based on Handheld Terminal.It aims to solve the problem that manageing a large number of keys to integrated communication base stations.

This paper put forward a Software Design of Remote Access Control Authentication Device Based on Handheld Terminal mostly.It makes the user and the server communicate with eacher other with the help of wechat public platform .It based on apache mina network framework and use GPRS network communication technology.The cilent send data to the server ,the data contain id of users and base stations.Then,after that ,the server contrast the id in data and in data base.if the two kind of data are identical,the server will send instruction to open the door. Otherwise,if not,the server send failed information to users. Thus, data interaction and remote information authentication are realized.

**Key words：**Information authentication; Access control of internet of things; Data filtering; Authentication secret key

**目　录**

[1. 绪论 1](#_Toc8728319)

[1.1 研究的意义 1](#_Toc8728320)

[1.1.1 选题的依据 1](#_Toc8728321)

[1.1.2 选题的意义 1](#_Toc8728322)

[1.2 国内研究现状 1](#_Toc8728323)

[1.3 研究的主要问题及方法 2](#_Toc8728324)

[1.3.1 研究的主要问题 2](#_Toc8728325)

[1.3.2 研究的方法 2](#_Toc8728326)

[2. 系统需求分析 3](#_Toc8728327)

[2.1 硬件需求 3](#_Toc8728328)

[2.2 软件需求 3](#_Toc8728329)

[2.2.1 开发工具 3](#_Toc8728330)

[2.2.2 开发语言 3](#_Toc8728331)

[2.2.3 开发框架 3](#_Toc8728332)

[2.2.4 开发环境 4](#_Toc8728333)

[2.3 性能需求 4](#_Toc8728334)

[3. 系统总体设计方案 5](#_Toc8728335)

[3.1 系统的总体设计 5](#_Toc8728336)

[3.2 数据传输模块和服务器之间的数据交互 6](#_Toc8728337)

[3.3 主控模块和服务器之间的数据交互 7](#_Toc8728338)

[3.4 单片机的选择 8](#_Toc8728339)

[3.5 网络通信技术的选择 8](#_Toc8728340)

[3.6 数据库的选择 9](#_Toc8728341)

[3.7 功能的实现 9](#_Toc8728342)

[4. 系统软件设计 10](#_Toc8728343)

[4.1 系统主程序设计 10](#_Toc8728344)

[4.2 数据库的设计 10](#_Toc8728345)

[4.2.1 数据库的概念结构设计 10](#_Toc8728346)

[4.2.2 数据库的逻辑结构设计 11](#_Toc8728347)

[4.3 网络通信协议的选择 12](#_Toc8728348)

[4.3.1 服务器和客户端的通信协议 12](#_Toc8728349)

[4.3.2 服务器和装置的通信协议 12](#_Toc8728350)

[4.4 数据库的连接 13](#_Toc8728351)

[4.5 客户端连接服务器 14](#_Toc8728352)

[4.6 装置连接服务器 14](#_Toc8728353)

[4.7 装置请求命令码的处理及反馈结果的发送 15](#_Toc8728354)

[4.7.1 装置请求命令码的处理 15](#_Toc8728355)

[4.7.2 反馈结果的发送 15](#_Toc8728356)

[4.8 通过微信公众号给用户发送反馈信息 17](#_Toc8728357)

[4.9 日志的处理 17](#_Toc8728358)

[5. 系统调试 19](#_Toc8728359)

[5.1 系统软件调试 19](#_Toc8728360)

[5.2 系统总体调试 20](#_Toc8728361)

[6. 总结 23](#_Toc8728362)

[参考文献 24](#_Toc8728363)

[附 录 25](#_Toc8728364)

[致 谢 32](#_Toc8728365)

1. 绪论

**1.1 研究的意义**

1.1.1 选题的依据

根据中国信息通信标准所研究表示，截止到第二季度末，我国的4G基站数量已达346万个，5G时代的到来，伴随着飞速提高的网络速度，5G基站数目可能是4G基站的2倍。由于这些场所存在一定的安全性，而且对进出人员要求比较严格。使用传统的机械锁，会出现企业管理大量基站门的钥匙的麻烦，当基站出现故障需要抢修时时，维修员需要去取钥匙维修结束再还钥匙，不可避免的浪费了大量的时间。因此，我们可以将嵌入式技术用于基站管理，解决钥匙管理的麻烦。

随着物联网产业的兴起，生活中处处体现能够感受到物联网带来的便利。微信公众号的开发平台也越来越健全[1]。本文将借助成熟的具有获取动态二维码信息并发送请求信息的公众号,将物联网嵌入式技术用于基站管理，解决钥匙管理的麻烦[2]。

1.1.2 选题的意义

大部分基站管理者为了省去区分钥匙的时间，常常采用通用性机械钥匙，多把锁用同一把钥匙。通用型机械钥匙具有制作成本低、安装便捷等特点，但是并不适用在安全性要求高的基站设备上。随着人工智能的快速发展，指纹、虹膜、面部识别门禁相继出现，极大的提高了人身的安全性。但其往往应用于家庭、公司、学校等场合，不适用于像基站这样大量的流动人员以及时间不固定的进出。

本文提出基于手持终端的远程门禁鉴权装置。将装置ID、服务器地址和时间随机码生成二维码并显示在OLED屏幕上。用户客户端使用微信扫码工具扫描二维码，将二维码中的信息通过数据传输单元发送至服务器。服务器做远程信息鉴权以及日志的记录。此装置取消了传统钥匙，避免了管理钥匙的麻烦。防止无授权的人员进入，日志查询出入记录，极大的提高了基站的安全性。

**1.2 国内研究现状**

当人们意识到传统门锁机械装置非常不安全时，更多的门禁系统应运而生。随着国民经济的发展，我国安防产业迅速发展，产品的种类也不断丰富。从机械锁演变为磁卡锁，以及常见的密码锁。目前市场上有关于集控型钥匙管理系统的研究，它主要分为解锁钥匙管理机系统和解锁钥匙信息管理系统两部分。使用ARM嵌入式技术，通信技术主要基于以太网和GSM的短消息通信技术。门锁采用电磁控制技术、IrDA红外检测光电传感和RFID无线射频技术。基站维修员通过系统提出申请，后台通过系统审核，向维修员发送短信进行回复，管理员打开控制解锁钥匙管理机门，否则将不做任何处理，工作人员也无法进入。

**1.3 研究的主要问题及方法**

1.3.1 研究的主要问题

（1） 网络通信方式的选择，服务器程序框架的选择，数据库的选择，编程环境的搭建。

（2） 数据库的建表，系统执行时数据的流向，服务器程序解析、处理请求，数据交互的实现。

（3） 系统的调试，在调试过程中寻找系统存在的漏洞进行完善。

1.3.2 研究的方法

本系统的主要目的是解决管理大量基站的钥匙的问题，从而提出这个智能门禁远程鉴权装置。由此从以下几个方面研究:

（1） 此设计的研究意义，是否有足够的实用性和安全性，以及未来市场的需求。设计的可行性，考虑设计中可能用到的关键技术是否难以实现[3]。

（2） 国内外的研究动态，搜取多方资源，观察是否有可以参考的价值。

（3） 画出系统的框架、查取相关技术工作原理。

（4） 以使用流程作为大纲，分部分进行设计，维修员扫描门上的二维码，向云服务器发送申请，云服务器判断合法后，给以权限并在屏幕上显示相应的工单，维修员选择工单后进入，系统录入数据库。在执行开门关门后在日志文件中产生相应的反馈。完成维修后锁门，从而达到对进出人员的严格控制，将此流程从理论变为现实。

2. 系统需求分析

**2.1 硬件需求**

本系统硬件核心部分是使用STM32F103VCT6开发板和STC15W4K324单片机。经过对系统的流程以及实现的功能分析之后，选用以下硬件进行实现：

（1） STM32F103VCT6开发板，为该系统的中心模块，用于生成动态二维码，控制电磁锁。

（2） 液晶显示屏，用来显示设备状态和二维码。

（3） STC15W4K324单片机，作为数据传输模块的中心组成部分[4]。

（4） GSM模块采用M6311中移动模组，内嵌传输协议模块用于进行数据传输。

**2.2 软件需求**

2.2.1 开发工具

（1） SpringToolSuite

SpringToolSuite集成了Spring开发框架，是基于Eclipse，但比Eclipse使用更方便。也具有Java集成环境的开发，拥有可扩展的开发平台，带有一套标准的插件，包括Java开发工具包（IDK）[5]。

（2） MySQL

MySQL数据库是关系型数据库，它支持结构化查询语句。它根据需求分表存储，提高了数据库的速度和灵活性。支持JDBC（Java数据库连接)，且客户机连接服务器的个数也没有限制，具有较高的安全性和可移植性[6]。

2.2.2 开发语言

本系统软件设计主要使用Java语言。Java语言使用可跨平台，开发资源分享，面向对象的设计使项目更安全。

2.2.3 开发框架

框架开发具有效率高、提高系统安全性的优点。系统服务器与装置和微信用户之间数据交互使用apache mina框架进行开发，它适用于可用性高的应用程序，往往分为服务器（Acceptor）部分和客户端(Connector)部分[7]。此论文只选择服务器部分开发，该框架注重接口的实现，其基础实现结构为以下文件:

（1） Server.java 设置缓冲区的大小，设置过滤器用于实现数据的编码和解码，端口号的绑定，并开始监听，框架的逻辑流程的处理。

（2） ServerHandler.java 接收到客户端发来消息，处理逻辑业务，即实现功能。

（3） Encoder.java 将应用程序中的对象转化为网络中的二进制。

（4） Decoder.java 将网络中的二进制流转化为应用程序中的对象。

2.2.4 开发环境

系统的后台开发使用的是Java开发，需要配置Java运行所需的环境，JDK版本选择jdk1.8.0\_151版本。在程序的编写过程中需要导入框架所需要的外部jar包，才可以正常运行。系统的硬件数据通信模块和服务器程序进行通信，后台代码运行在服务器上才可以进行正常数据交互。

**2.3 性能需求**

本系统用作基站的安全防护，管理员可以远程通过后台查询记录检查进出的情况，综合实用性及用户体验，对本系统的性能做出以下需求：

1. 数据库中数据无冗余的函数依赖。
2. 软件内代码的功能命名有具体的意义，尽量将功能封装为对象，方便调用。
3. 要求系统中数据传输安全可靠，不容易丢失、响应快。
4. 软件方便操作，使用简单，后期可维护性强[8]。

3. 系统总体设计方案

**3.1 系统的总体设计**

此系统硬件分为主控模块和数据通信模块（DTU）,以STM32F103VCT6单片机作为系统的中央处理器。将设备的ID、服务器地址信息和时间有关的随机秘钥进行生成动态二维码，在OLED显示屏上显示。用户扫描二维码将设备的ID发送至服务器，服务器根据用户关注微信公众号后的openId和对比数据库中的信息。若数据库中存储用户的信息，且装置连接状态正常、注册信息正常，授予开门权限。云服务器根据二维码中的装置ID发送开门指令，DTU将开门指令传送给主控模块,发送开锁指令。STC15W4K324单片机、M6311移动模组共同组成的数据通信模块，同时对外接上一个RS232串口。主控模块连接执行单元，即电磁锁。系统总体设计通过使用流程的角度将总体设计如图3-1所示。

请求数据指令

设备信息

设备ID

微信扫描二维码

云服务器

身份验证

数据通信模块

微信用户显示工单

主控执行开锁

根据openId

验证未通过

验证通过

图3-1 总体设计执行流程图

系统的硬件框架图如图3-2所示。

STM32F103VCT6单片机

显示屏

主控模块

电磁执行单元

云服务器

STC15W4K324单片机

M6311模组

数据通信模块

RS232

图3-2 硬件模型图

系统的软件部分基于apache mina网络应用框架和微信公众号开发平台。将装置和客户端分别和服务器之间进行数据传输。服务器程序主要接收微信发来的数据请求进行解析，首先对比数据库中的微信的唯一标识openId，若对比成功则授予其开门权限，其次对比数据库中装置的状态，若状态正常，服务器则发送开门指令。将对比成功的结果发送至数据传输单元（DTU）,数据传输单元（DTU）再将数据传输给主控（MCU）。主控执行开锁的操作后，电磁锁打开，同时将结果反馈给微信用户。若是对比失败结果则只通过云服务器返回给公众号，并不执行开锁操作。微信公众号将微信服务器和需要的云服务器连接起来，系统中的软件微信公众平台数据交互原理如图3-3所示。

对比结果

数据对比

返回请求结果

申请数据

后台程序

数据库

微信公众号

云服务器

DTU

设备信息

图3-3 数据交互原理图

**3.2 数据传输模块和服务器之间的数据交互**

数据传输单元和服务器通信过程，上电后，主动向服务器发送TCP/IP连接的申请，然后发送上初始化信息。服务器（DSC）根据报文信息判断是否初始化，服务器回复响应报文。报文中携带标志位，0x01表示初始化未完成，0x02表示未进行初始化。若已经初始化，数据传输单元（DTU）会向服务器发送注册信息，注册信息包含数据传输单元的身份信息。服务器给数据传输单元回应响应注册信息，至此完成注册，并维持数据连接。若没有初始化，数据通信单元进行等待，服务器发送给数据传输单元秘钥进行初始化，接下来完成注册信息。若数据传输模块在15秒内等不到服务器发来的秘钥，则进入休眠状态，断开连接。数据传输单元和服务器数据交互流程如图3-4所示。

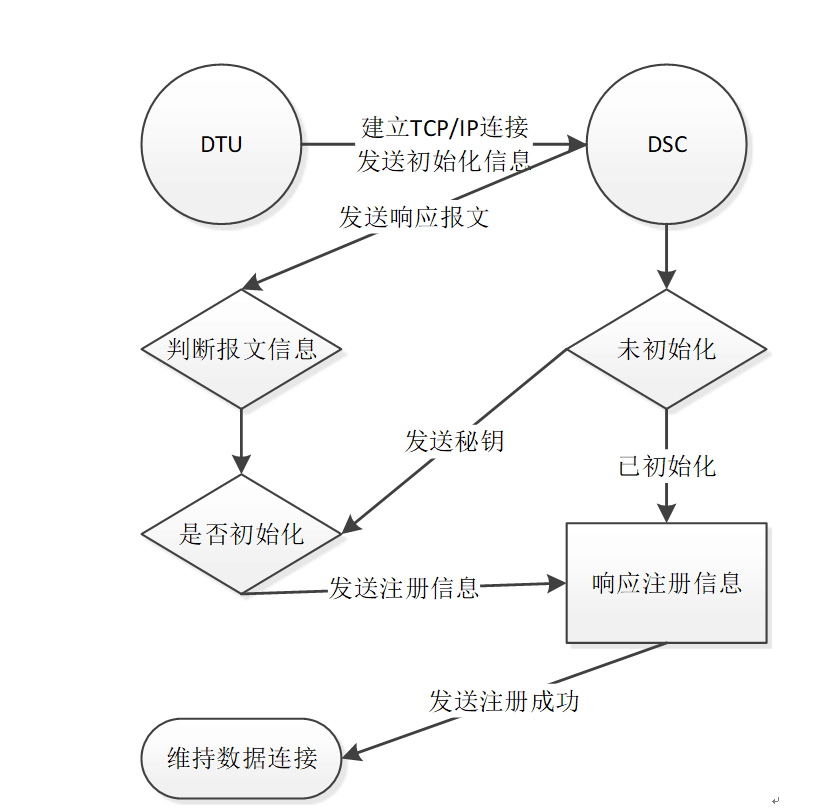
****

图3-4 数据传输单元和服务器数据交互图

**3.3 主控模块和服务器之间的数据交互**

主控向服务器上传数据，主控部分使用串口通信的方式发送到数据传输单元，数据传输单元将数据封装好发送给服务器。服务器发送响应报文给数据传输单元。数据传输单元判断是不是自己的数据，若是发给自己的数据进行接收，若不是发送给主控模块。主控向服务器上传数据如图3-4所示。

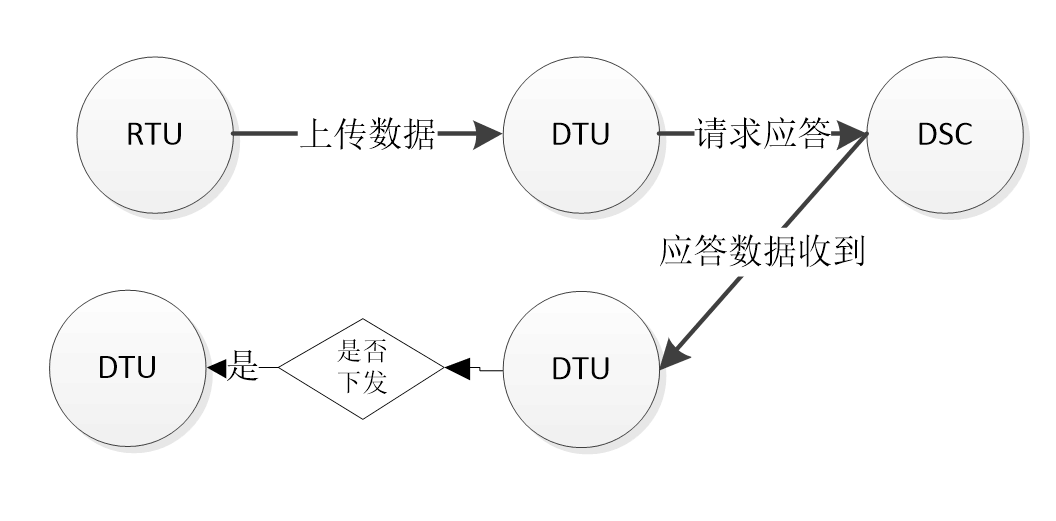


图3-4 主控向服务器上传数据图

服务器向主控部分下发数据同样需要经过数据传输单元，数据传输单元将数据转发给主控单元，主控单元通过数据传输单元向服务器响应信息。服务器向主控下发数据如图3-5所示。

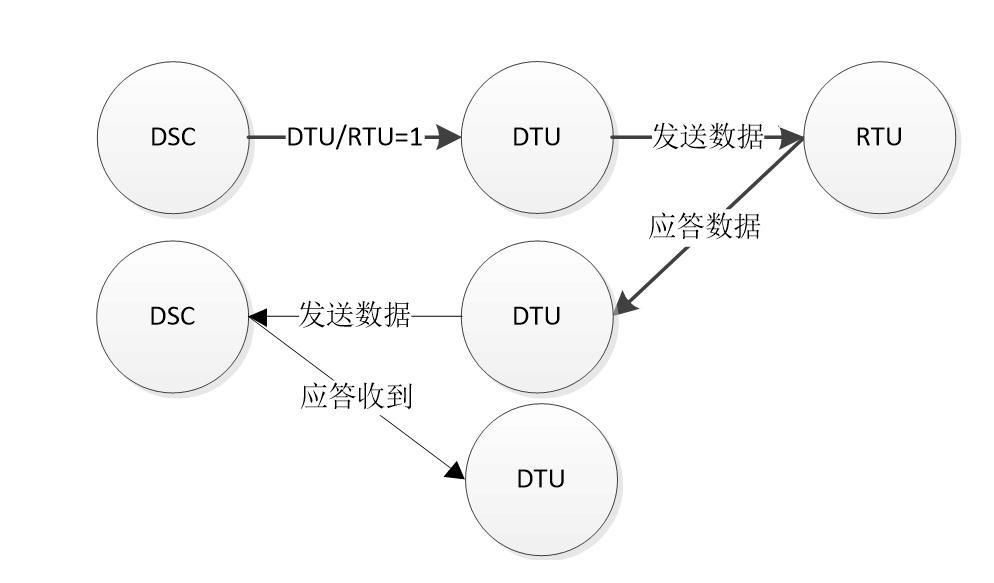


图3-5 服务器向主控下发数据图

指令0x20~0x2F表示发送给DTU的请求数据。0x60~0x6F表示发给RTU的请求数据，DTU收到转发给RTU。0xA0~0xAF表示发给DTU的响应数据。0xE0~0xEF表示发给RTU的响应数据，数据内容非RTU请求RSC应答数据时，RSC仅应答数据收到。为RTU请求数据时，应答需由DTU转发，应答类型由指令类型决定。

0x00~0x0F 来自DTU的数据请求。0x40~0x4F 来自RTU的数据请求，根据数据内容作出应答。0x80~0x8F表示DTU响应数据请求。0xC0~0xCF RTU响应数据，根据数据类型应答。

**3.4 单片机的选择**

此装置使用STM32F103VCT6单片机和STC15W4K324单片机。此单片机是32位的单片机，一次可处理32位数据，并且价格优廉。该芯片采用ARM公司的Cortex-M3内核，具备72MHz的系统主频，内部具有64K字节SRAM为开发提供了很大的便利。可以承担本作品的控制任务。

STC15W4K324单片机，一次只能处理8位，但作为数据传输单元，它的功能足够承担数据传输的任务。选择此芯片不仅可以减轻主控模块的负担，也是更经济的选择。此系列单片机由于更基础，价格低廉，历史比较久，因此在开发过程中可以借鉴成熟的例子。因此选择STC15W4K324单片机作为数据通信单元的重要组成部分。

**3.5 网络通信技术的选择**

近年来，随着网络的迅速发展，使得全社会得自动化程度提高，GPRS对我们每个人来说都不陌生，它不仅是手机中重要的通信方式。GPRS是GSM移动用户可以用的一种服务，它的含义是通用无线分组业务。网络覆盖范围广泛，并可以实现实时通信。它大大缩短了用户呼叫建立的时间。GSM移动只需做软件升级，几乎不必改动硬件。它具有非常多的优点，它的覆盖范围非常广，即使是山区和县镇也可以覆盖到。数据传输率高，它的通信费用低，服务成本也更低。具有很高的实时响应和处理能力，几乎不存在系统延时，实时在线。可以非常好的满足传输的实时性的要求和数据采集[9]。

中国移动提供的GSM模组，提供便利的联网，因此选择GPRS传输。

**3.6 数据库的选择**

MySQL系统相比与常见的SQL Server数据库系统同样支持SQL。并且对于多数个人用户是免费的，体积也很小，因此更简单易用。对于系统来说数据的安全性至关重要，这关系到用户的利益与体验。没有连接用户个数的限制，MySQL不仅可以实现共享，也可以进行访问控制，具有较高的安全性运行速度快。资源齐全，便于开发。

**3.7 功能的实现**

经过对系统的需求分析以及系统的实现，系统正常运行之后将具备以下功能:

（1） 系统数据传输模块正常联网，主控模块生成二维码正常。

（2） 用户扫描公众号后可以提取设备ID发送至服务器。

（3） 服务器正确解析用户发来的数据请求。

（4） 服务器对比数据库中数据后发送指令正确。

（5） 执行单元得到主控模块的指令后能够开锁。

4. 系统软件设计

**4.1 系统主程序设计**

软件部分实现的主要功能为通过云服务器接收客户端、装置发来的数据，放入缓冲区并处理。服务器程序将数据与数据库中的数据进行对比，若用户是数据库中所存储的用户，则表示用户合法。在类中用ischeck()返回一个布尔类型的值用来表示装置状态。若检查装置状态正确，给与开锁权限，将开锁指令通过云服务器发送给数据传输单元（DTU）。开锁成功后将信号反馈给服务器，服务器修改数据库中装置的状态。若用户不合法或者装置状态不正确则都不给开门权限。服务器和数据通信模块之间进行指令传输，两者之间约定指令的含义。DTU主动发送连接请求，服务器根据已经约定好的指令给装置发送秘钥、连接指令或者开门指令，必要时对数据库的操作。装置数据通信模块和服务器之间通过指令传输，其中包括初始化请求报文、认证请求报文、设置IP地址报文、设置秘钥报文、设置keepalive报文、停止数据连发报文、加密秘钥报文、设置身份识别码报文。具体执行细节如图4-1所示。

开门成功/开门失败

用户OpenId

设备Id

DTU数据通信模块

微信扫描二维码

服务器

程序

数据库Ischeck()

UserDao

CabinDao

True/False

请求设置

门信息指令

开门指令

图4-1 执行流程图

**4.2 数据库的设计**

4.2.1 数据库的概念结构设计

在软件设计中，数据是最重要的组成部分，当数据量较大且关系较复杂时，更需要一个好的数据库[10]。在本设计中，主要有用户实体和装置实体。openId用来唯一标识用户，对当前公众号唯一。用户手机号属性，用户名属性，不可为空也不能重复。从现实和逻辑上来看，用户的名字有可能出现重复，而关注公众号后的生成的openId却只有一个。为装置进行普通编号id作为主码，拥有imei属性、state装置状态属性，qrcode二维码属性。imei为国际移动设备识别码,每一个GSM模组都拥有一个。qrcode是将装置信息和随机时间码生成动态二维码。

用户实体和装置实体分别如图4-2、图4-3所示。

用户

图4-2 用户实体图

装置

图4-3 装置实体图

4.2.2 数据库的逻辑结构设计

用户信息表如表4-1所示。

**表4-1 用户表信息（User）**

|  |  |  |  |
| --- | --- | --- | --- |
| 用户编号 | 用户标识 | 用户名称 | 用户手机号 |
| Id(char) | openId(char) | UserName(char) | Tel\_number() |
| 01 | 06\_bmjPTlm6\_2sgVt7hMZOpfL2M | 闫英杰 | 18790565355 |
| 02 | o9Kcn0SD0khSzjJFcHxbNlh2JdKE | 刘晓琳 | 13781977498 |

由于装置主要信息表包括装置序号、装置状态、装置信息二维码qrcode,具体情况如表4-2所示。

**表4-2 装置信息表（Cabin）**

|  |  |  |  |
| --- | --- | --- | --- |
| 装置序号 | 装置状态 | 装置二维码 | Imei |
| Id(char) | state(int) | qrcode(char) | imei(char) |
| 01 | 1 | Null | 861853033622424 |
| 02 | 0 | Null | 861853033622425 |

**4.3 网络通信协议的选择**

4.3.1 服务器和客户端的通信协议

用户客户端与服务器之间选择WebSocket通信协议。WebSocket通信协议基于TCP协议，它与传统常用TCP协议不同。它只需要建立一次连接就可以进行通信，不需要不断发送请求[11]。节省了带宽和CPU占用。服务端和客户端可以主动向对方发送信息。交互双方可以发送二进制数据，在本系统中传输将组合为十六进制，并将其转化为字节数组。最关键特点是本论文选择的mina框架提供WebSocket支持。WebSocket通信协议原理如图4-4所示。

webSocket

client

server

Connection open

data

data

data

time

Connection close

图4-4 WebSocket通信协议原理图

4.3.2 服务器和装置的通信协议

服务器和装置间数据传输选择TCP协议。由于硬件数据通信模块为中国移动M6311模组，内嵌TCP/IP协议，因此选择TCP/IP协议。在开发过程中要了解TCP的特点和传输机制。TCP用校验、序号、确认和重传等机制来保证数据的安全性[12]。在交互过程中使用套接字（端口号；IP地址）来寻找客户端的地址，它唯一的标识了主机上的一个进程[13]。在通信之前必须进行连接，因此具有较高的安全性。TCP报文段有首部和数据部分，这个报文段不仅可以用来传输数据，也可以用来建立连接、释放连接和应答。

系统中数据传输的通信协议如图4-5所示。

DTU数据通信模块

微信扫描二维码

服务器程序

WebSocket

TCP

图4-5 系统中通信协议图

**4.4 数据库的连接**

（1） 下载并导入mysql-connection-jar驱动。任何在导入外部的jar包之后只有进行buildpath被项目引用之后才能真正的发挥作用。

（2） 新建DBHelper.java专用做连接数据库的类定义数据库的IP地址和端口号。数据库的用户名称为超级管理员权限、数据库名称share\_cabin、驱动器名称8.0.15版本驱动器连接。这些在建数据库的时候已经配置成功的参数，这样数据库就连接成功了。

DBHelper.java

String dbIP = "127.0.0.1";

String dbPort = "3306";

String dbName = "share\_cabin";

String dbUserName ="root";

String dbPassword ="287522";

Class.forName("com.mysql.cj.jdbc.Driver");

conn=DriverManager.getConnection("jdbc:mysql://"+dbIP+”:”+dbPort+"/"+dbName+"?autoReconnect=true&useUnicode=true&characterEncoding=utf8&serverTimezone=GMT%2B8&useSSL=false",dbUserName,dbPassword);

**DBHelper.java**

（3） 数据库连接的测试，连接成功如图4-6所示。

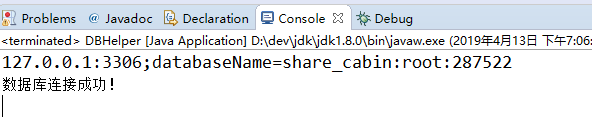


图4-6 数据库测试结果图

**4.5 客户端连接服务器**

在本系统中，用户充当客户端，服务器程序就是服务器端。系统主要使用apache mina框架。本系统是服务器程序只选择mina框架中的Acceptor端进行框架基础之上的开发。

（1） 主程序中逻辑的实现。设置服务器监听端口为8012。设置Filter链，实现数据的过滤，即数据的编码和解码，将网络中的二进制数据和面向对象中的数据进之间的转化。将编码类对象和解码类对象添加到getFilterChain()方法中，并设置接收缓存区大小为2M。可以同时设置读写两个时间，但此处只设置设置读空闲时间setIdleTime为3600。创建业务处理类的对象，调用处理业务类。绑定端口InetSocketAddress(port)的同时就开始了监听。

（2） 数据的解码。网络中数据的传输是二进制，而在服务器应用程序中数据常常是Java对象或者基本类型。因此数据的解码就是将二进制转换为数据对象。传输协议为只需要一次连接就可以传输数据的WebSocket，将WebSocket将mina框架进行结合。

数据的交互协议中，第一个字节暂时没用，不进行说明。第二个字节第1位存放掩码，后7位表示数据传输内容的长度。第三个字节到第六个字节为掩码内容。第七个字节之后为传输的真正内容。

（3） 数据的编码。数据的编码就是将数据对象转换为二进制。定义WebBean类用来接收WebSocket传来的消息。使用message.getContent().getBytes("UTF-8")即可将消息的内容转换为二进制数据流。

**4.6 装置连接服务器**

（1） 主程序中逻辑的实现。设置服务器监听端口ServerPort为8021。设置Filter链，实现数据的过滤，即数据的编码、解码和日志的实现，这里比客户端连接服务器多出一个日志创建，用来日后查询开门记录。设置接收缓存区大小为2048，设置读空闲时间为180，创建业务处理类的对象，调用处理业务类，绑定端口的同时开始监听[14]。

（2） 数据的解码。将二进制转换为数据对象，数据的首部长度为10，剩余均为消息主体。在收到消息后，调用数据格式转换类中的ByteToLong()方法，将字节转化为长整形类型。将前8个字符取出，作为Id,剩下字符作为命令码cmdCode存起来,在处理信息方法中备用。

（3） 数据的编码。将数据对象转换为二进制，数据首部长度为10。将消息主体、数据长度10、命令编码getCode()方法的到的编码、getCode()得到的装置Id和getSn得到编码序列号Sn放入缓冲区IOBuffer中发送给装置数据传输模块。

**4.7 装置请求命令码的处理及反馈结果的发送**

4.7.1 装置请求命令码的处理

首先定义收到码的格式，区分收到的字节分别代表的不同的请求。当收到不同的请求，将会对数据库中装置信息做出改变，或者将指令发送给装置数据传输模块，同时将反馈信息通过sendMsg()发送给用户。命令码和调用方法对照如表4-3所示。

**表4-3 命令码和调用方法对照表**

|  |  |  |
| --- | --- | --- |
| 报文（字节） | 报文含义 | 调用方法 |
| 0x01 | 初始化请求 | CmdInitResp() |
| 0x02 | 发送认证信息请求 | CmdAuthReq() |
| 0x81 | 设置IP地址请求 | CmdSetAddrResp() |
| 0x82 | 设置认证秘钥请求 | CmdSetAuthResp() |
| 0x83 | 设置keepalive响应请求 | CmdSetKeepAliveResp() |
| 0x84 | 设置停止数据连接请求 | CmdCloseConnResp() |
| 0x85 | 查询配置请求 | CmdQueryResp() |
| 0x86 | 设置加密秘钥请求 | CmdSetEncrypResp() |
| 0x87 | 删除装置信息请求 | CmdResetResp() |
| 0x88 | 设置身份识别码请求 | CmdSetIdResp() |
| 0x89 | 查询版本请求请求 | CmdQueryVerResp() |
| 0x40 | 主动上报装置信息 | recvRtuReq(cmd) |
| 0xC0 | 开门信息反馈 | recvRtuResp(cmd) |

4.7.2 反馈结果的发送

在上述处理请求码的过程中，用封装好的请求来判断消息，或者将请求结果返回。在这个过程中xxxReq()作为请求，xxxResp()作为给装置数据传输模块的返回。在此过程中，常常需要查询、修改或者删除数据库中的数据，由此建立一个独立的Dao包，包含CabinDao.java和UserDao.java对数据库中的数据库进行存取[15]。创建PreparedStatement对象，只需要将属性用“?”表达，再进行设置，就可以像数据库中结构化查询。在信息反馈的过程中不仅需要操作数据库中数据而且常常涉及给装置发指令和给用户发送反馈信息。例如更新门状态。涉及对数据库的操作，当门打开后，将结果反馈给数据通信模块，数据通信模块上传至服务器，告诉服务器开门成功。服务器程序对数据库中的装置信息做出更新状态操作，表示门从关闭到打开。当关门之后则告诉服务器进行初始化操作，即表示门关闭，成为初始化状态。例如开门操作，执行设计给用户发送反馈和与数据通信模块进行反馈。

CabinServerHandler.java

private void recvRtuResp(AbstractCommand cmd) {

CmdRtuResp rtuResp = (CmdRtuResp) cmd;

long id = cmd.getId();

Cabin cabin = onlineCabins.get(id);

CmdRscAck ack=new CmdRscAck(0, rtuResp.getId(), rtuResp.getSn());

cabin.getSession().write(ack);

if (rtuResp.isCheck()) {

byte kind = rtuResp.getKind();

byte type = rtuResp.getType();

if ((0xff & kind) == 0x83 && type == 0x01) {

JSONObject obj = cabin.getUserReq(rtuResp.getSn());

if (obj != null && webListener != null) {

JSONObject o = new JSONObject();

if (rtuResp.getState() == 1) {

CabinDoorDao.updateDoorState(obj.getLong("doorId"),1);

String msg = "门已打开！";

MsgCustomer.sendMsg(obj.getString("openId"), msg);

}

webListener.callBack(obj.getString("openId"), obj.getByteValue("code"), rtuResp.getState(), o);

System.out.println("发送反馈信息开门");

} else {

System.out.println("发送反馈错误" + ":" + webListener);

}

}

}

}

**CabinServerHandler.java**

**4.8 通过微信公众号给用户发送反馈信息**

用户在传输过程中为客户端的，建立客户端连接HttpClient client=HttpClients.createDefault();得到的client对象进行执行操作。得到参数组合才可以创建传参数的httpGet对象，进行数据传输。而消息的地址将通过格式化后的sendMsg，accesstoken进行传递。创建对象HttpPost并将消息地址传输进去。在微信公众号开发中，openid作为普通用户的标识。AccessToken相当于各种接口的钥匙，根据appid和secret组合得到用AccessToken、消息三个参数组合成具有统一格式的msg\_format，并用utf-8格式将数据规格化。

MsgCustomer.java

String msgUrl=String.format(sendMsg\_format, access\_token);

HttpPost httpPost = new HttpPost(msgUrl);

String data=String.format(msg\_format, openid,msg);

httpPost.setEntity(new StringEntity(data,"utf-8"));

HttpResponse response = client.execute(httpPost);

data=EntityUtils.toString(response.getEntity());

System.out.println(data);

**MsgCustomer.java**

**4.9 日志的处理**

在日志的编写前，需要下载日志包并导入。在执行的过程中，无论执行是否成功，都会产生日志。因此，日志代码的编写无处不在。

CabinServerHandler.java

LOG.error(session.getId() + " The client " + session.getRemoteAddress() + " disconnect");

**CabinServerHandler.java**

5. 系统调试

**5.1 系统软件调试**

在代码编写完成后，为了测试代码的功能性，进行系统的软件测试。由于在系统中，服务器和两个客户端之间进行数据的过滤，即编码和解码。数据传输模块传输的是二进制数据，根据编码解码器中的功能不能直接打印出来。因此选择模拟一个客户端依据本论文中借助的mina框架与服务端进行数据的传输。测试部分分为客户端、服务端、客户端处理、服务端处理四个部分，详细过程如下:

（1） 建立MinaServer.java，编写服务器的主程序。创建端口号为7080，创建继承自Acceptor类的IoAcceptor。设置过滤器编码和解码，调用电脑系统控制台能过识别的的LineDelimiter.WINDOWS.getValue()进行编解码。另外设置缓冲区大小和读取空闲时间。接着进行自己的session会话处理，绑定端口号进行监听，开始监听服务器发来的会话连接。

（2） 建立MinaClient.java，编写客户端的主程序。设置客户端的IP地址为本地地址，端口号为7080。设置与客户端的相似的编解码，缓冲区大小为1024，空闲时间为20秒，接着进行自己的会话处理。

（3） 建立Myhandler.java，继承自IoHandlerAdapter()。出现异常状况时IoHandlerAdapter()，客户端接收到服务器的messageReceived()，通过session.write()将信息发送到客户端，当字符等于quit时调用session.close()关闭连接;。

（4） 建立MyClientHandler.java，实现封装类的代码。出现异常状况时IoHandlerAdapter()、客户端接收到服务器的messageReceived()、以及会话的打开与关闭。

运行服务器程序，等待客户端的连接，服务器开始监听，显示监听端口7080，执行结果如图5-1所示。

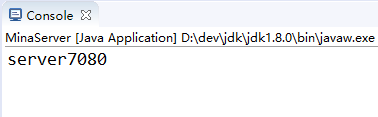


图5-1 客户端监听状态图

运行客户端程序，此时建立与服务器的连接，服务端显示结果如图5-2所示。

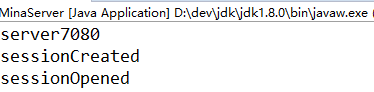


图5-2 客户端和服务端建立连接成功图

此时客户端收到服务器发送数据结果如图5-3所示。

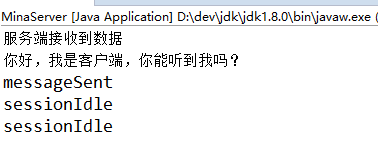


图5-3 客户端收到数据结果图

在客户端收到服务器发送的数据之后结果如图5-4所示。



图5-4 服务器收到数据结果图

当发送的数据等于字符串quit时表示退出连接结果如图5-5所示。

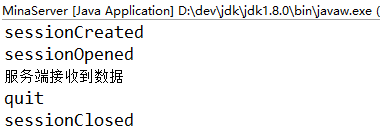


图5-5 正常关闭连接结果图

异常关闭，结束程序编译时关闭结果如图5-6所示。

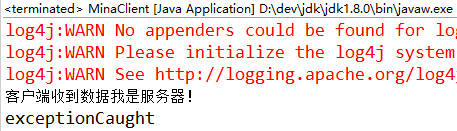


图5-6 异常关闭连接结果图

**5.2 系统总体调试**

（1） 将硬件编写的hex文件烧写进主板中，当把程序烧写进去后，硬件数据主控模块生成二维码。没有连接数据传输模块时，设备显示未联网，不能发送数据。主控显示情况如图5-7所示。



图5-7 设备未联网图

（2） 连接数据传输模块。硬件数据传输模块内存服务器的IP地址123.207.221.93，上电后给服务器发送心跳信号，与服务器建立连接。设备通过数据传输模块给服务器发送初始化信息得到认证秘钥，发送认证信息，可以正常联网，主控情况显示如图5-8所示。

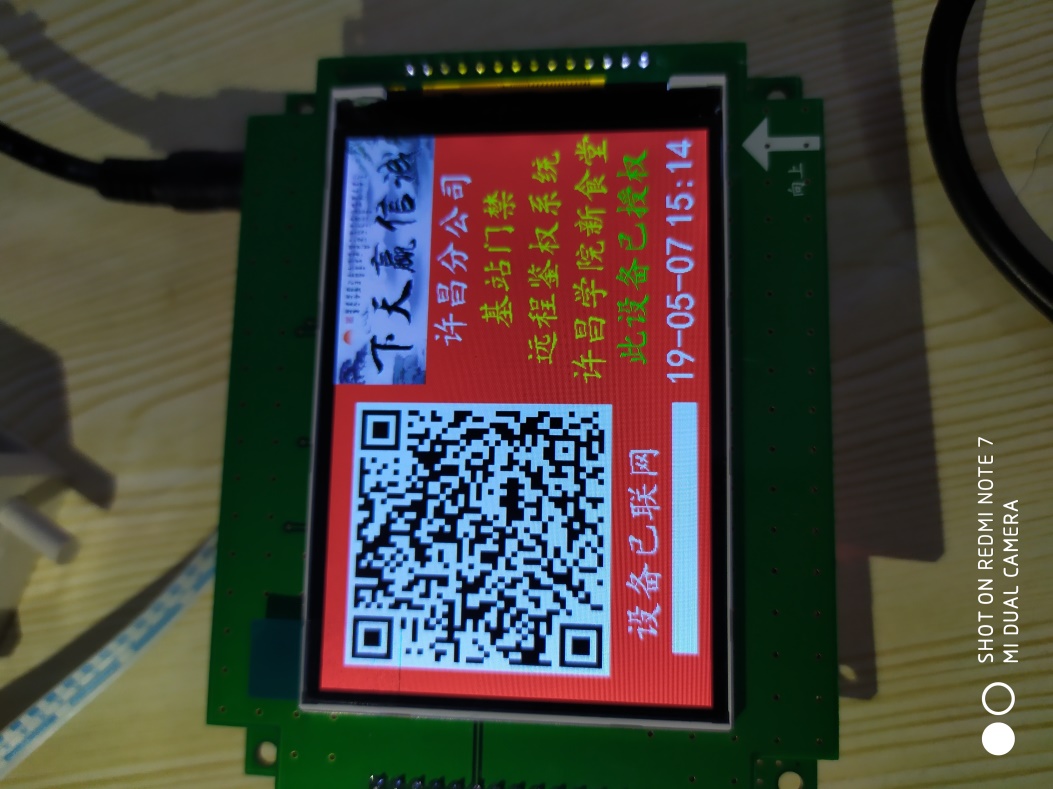


图5-8 设备联网图

（3） 准备工作。配置服务器环境，将项目放到服务器上；将硬件编写的hex文件烧写进主板中；硬件数据主控模块生成二维码；借助扫描二维码的微信公众号，可以识别二维码中服务器地址和设备ID，发送至服务器。

（4） 请求开门。微信扫描二维码后，在工单中选择安全巡检，发送开门请求，服务器程序进行审核处理开门请求。审核成功后，将指令发送给数据传输单元，数据传输单元判断不是自己需要的指令则将其转发给主控模块，主控模块执行开锁，开锁成功后将状态返回给服务器。服务器程序再给微信用户反馈发送成功的消息。微信公众号则根据收到的消息，在用户界面上显示。开门申请和开门成功反馈结果如图5-9、图5-10所示。

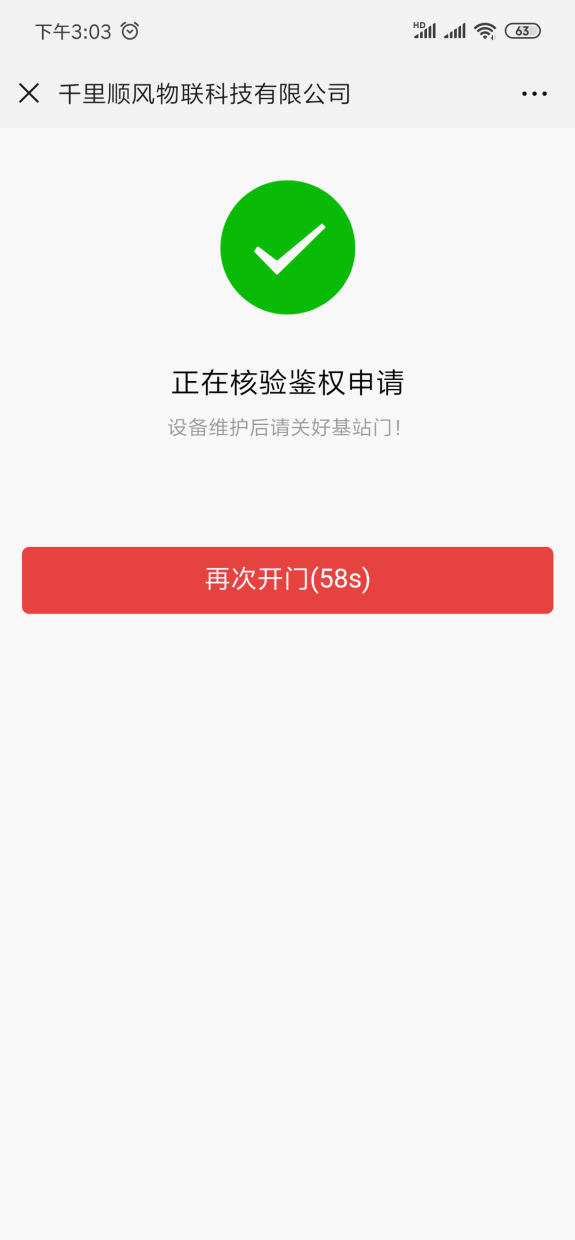


图5-9 开门申请界面 图5-10 开门成功界面

6. 总结

本系统采用Java语言进行服务器程序的编写，使服务器与用户和数据传输单元进行数据传输。在数据传输模块初次上电之后，在服务器端对装置进行初始化和注册信息的操作。在装置和数据传输过程中，主要定义指令是由数据传输模块还是主控模块的的指令以及指令所表示的含义。在服务器和数据传输单元之间进行数据传输中，数据传输模块向服务器发送指令主要包含初始化请求报文、认证请求报文、设置IP地址报文、设置秘钥报文、设置keepalive报文、停止数据连发报文、加密秘钥报文、设置身份识别码报文。

在服务器程序使用mina框架，mina是一个高可用性的网络框架，使开发者不必关注数据是如何发送和接收的，把重点专注在逻辑功能的实现和数据的编解码，在此服务器程序编写的过程中，只需要使用实现mina框架的服务器接口。传输网络协议使用websocket和TCP协议进行数据传输。

用户只需要扫描主控显示屏上的的二维码，关注公众号后，选择基站门禁，再次扫描二维码，选择安全巡检工单，申请开门后，电磁锁打开。基于手持终端的门禁远程鉴权装置主要用于基站，取消了传统的钥匙。此装置在即将到来的5G时代，将会发挥不可限量的作用。

参考文献

1. 景光铮,王若桐,李轩,等.基于微信平台二维码身份识别的门禁系统[J].电子世界, 2018(6),1-5.
2. 周杨,张光照.基于微信与二维码的图书馆门禁系统设计——以贵州财经大学图书馆为例[J].江苏科技信息, 2018, 35(11): 35-37.
3. 吴洁明.软件工程实践教程[M].北京:清华大学出版社,2004,10-35.
4. 刘剑,刘奇穗.51单片机开发与应用基础教程（C语言版）[M].北京:中国电力出版社,2014,30-63.
5. 贾燕枫.Eclipse开发入门与项目实践[M].北京:人民邮电出版社,2006,78-87.
6. 姜承尧.高性能网站MySQL数据库实践[M].维普中文科技期刊数据库,2013.1-3.
7. 金玉军,姜晓彤,况迎辉.基于高性能NIO的MINA框架的应用[J].中国科技信息, 2009(16):93-94.
8. 陆惠恩.软件工程[M].上海:上海交通大学出版社,2016,54-86.
9. 陈明月.浅谈基于嵌入式GPRS无线数据传输系统的实现及应用[J].计算机产品与流通 2019(02):127,2-6.
10. Xiangyang Q I, Lin S, Cui H, et al. Research on the Remote Data Collection Based SQL Server[J]. 武汉理工大学学报, 2006, 28(s3):828-832.
11. 孙卫琴.Tomcat与Java Web 开发技术详解（第2版）[M].北京:电子工业出版社, 2009,98-112.
12. 谢希仁.计算机网络[M].北京:电子工业出版社,2016,60-75.
13. 陈更力,张青.基于JavaSocket网络编程的一种新实现[M].电脑开发与应用,2006,35-64.
14. 杨铁军,徐和飞,黄琳.一种基于MINA框架的数据通信平台设计[J].微计算机信息,2009(33):22-24.
15. 传智播客高教产品研发部.Java基础入门[M].北京:清华大学出版社,2014,66-92.

附 录

CabinDoorDao.java

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import Data.Cabin;

import Data.Order;

import DataBase.DBHelper;

public class CabinDoorDao {

private static final Logger LOG = Logger.getLogger(CabinDao.class);

public static int getDriveSize(long id) {

try {

Connection conn = DBHelper.getConn();

PreparedStatement psmt = conn.prepareStatement("select count(id) from cabindoor where cabinId=?");

psmt.setLong(1, id);

ResultSet rs = psmt.executeQuery();

if(rs.next())

{

return rs.getInt(1);

}

} catch (SQLException e) {

e.printStackTrace();

}

return -1;

}

public static int updateDriveInfo(List<RtuDriveParam> rtuDrivers, longid, int size) {

Connection conn = DBHelper.getConn();

try {

PreparedStatement psmt2 = conn.prepareStatement("insert into cabindoor(cabinId,driveId,seq,state) values(?,?,?,?,?)");

PreparedStatement psmt1 = conn.prepareStatement("delete from cabindoor where cabinId=?");

int newSize=0;

psmt1.setLong(1, id);

psmt1.execute();

for (int i = 0; i < rtuDrivers.size(); i++) {

RtuDriveParam param = rtuDrivers.get(i);

int seq=param.getFirstSeq();

psmt2.setLong(1, id);

psmt2.setInt(2, param.getId());

psmt2.setInt(3, seq);

psmt2.setInt(5, 0);

psmt2.addBatch();

seq++;

}

psmt2.executeBatch();

return newSize;

} catch (SQLException e) {

e.printStackTrace();

}

return -1;

}

public static int updateDriveInfo(short s,long cabinId) {

Connection conn = DBHelper.getConn();

try {

PreparedStatement psmt1 = conn.prepareStatement("delete from cabindoor where cabinId=?");

PreparedStatement psmt3 = conn.prepareStatement("insert into cabindoor(cabinId,seq,state，time) values(?,?,?,?)");

int newSize=0;

psmt1.setLong(1, cabinId);

psmt1.execute();

int seq=s;

psmt3.executeBatch();

return newSize;

} catch (SQLException e) {

e.printStackTrace();

}

return -1;

}

public static void updateDoorState(Long id,byte[] seqs,int state) {

try {

Connection conn = DBHelper.getConn();

PreparedStatement psmt = conn.prepareStatement("update cabindoor set state=?,openId=? where cabinId=? and seq=?");

for (int i = 0; i < seqs.length; i++) {

psmt.setInt(1, state);

psmt.setString(2, "");

psmt.setLong(3, id);

psmt.setInt(4, seqs[i]);

psmt.addBatch();

}

psmt.executeBatch();

} catch (SQLException e) {

e.printStackTrace();

}

}

public static void updateDoorState(long id,int state) {

try {

Connection conn = DBHelper.getConn();

PreparedStatement psmt = conn.prepareStatement("update cabindoor set state=? where id=?");

psmt.setInt(1, state);

psmt.setLong(2, id);

psmt.executeUpdate();

} catch (SQLException e) {

e.printStackTrace();

}

}

public static void clearState(long id) {

try {

Connection conn = DBHelper.getConn();

PreparedStatement psmt = conn.prepareStatement("update cabindoor set state=?,openId=? where cabinId=?");

psmt.setInt(1, 0);

psmt.setString(2, "");

psmt.setLong(3, id);

psmt.executeUpdate();

} catch (SQLException e) {

e.printStackTrace();

}

}

public static void updateDoorState(long id, byte[] doorState,int start) {

System.out.println("start="+start);

try {

Connection conn = DBHelper.getConn();

PreparedStatement psmt = conn.prepareStatement("update cabindoor set state=? where cabinId=? and seq=?");

int seq=start;

for (int i = 0; i < doorState.length; i++) {

psmt.setInt(1, doorState[i]);

psmt.setLong(2, id);

psmt.setInt(3, seq);

psmt.addBatch();

seq++;

}

psmt.executeBatch();

} catch (SQLException e) {

e.printStackTrace();

}

}

public static int queryState(Long doorId, String openId, String password) {

try {

Connection conn = DBHelper.getConn();

PreparedStatement psmt1 = conn.prepareStatement("select state,time,openId from cabindoor where id=?");

PreparedStatement psmt2 = conn.prepareStatement("update cabindoor set state=5,openId=?,time=? where id=?");

psmt1.setLong(1, doorId);

ResultSet rs = psmt1.executeQuery();

if (rs.next()) {

int state=rs.getInt("state");

long time=rs.getLong("time");

String ropenId=rs.getString("openId");

if (state==0) {

psmt2.setString(1, openId);

psmt2.setLong(2, System.currentTimeMillis());

psmt2.setLong(3, doorId);

psmt2.executeUpdate();

return 0;

}else if (state==5) {

long curTime=System.currentTimeMillis();

if (curTime-time>90\*1000) {

psmt2.setString(1, openId);

psmt2.setLong(2, curTime);

psmt2.setLong(3, doorId);

psmt2.executeUpdate();

return 0;

}else if (openId.equals(ropenId)) {

return 0;

}else {

return 5;

}

}else {

return state;

}

}else {

return -1;

}

} catch (SQLException e) {

e.printStackTrace();

}

return -1;

}

public static List<Order> queryIsUse() {

List<Order> orders=new ArrayList<>();

try {

Connection conn = DBHelper.getConn();

PreparedStatement psmt = conn.prepareStatement("select a.id as cabinId"

+ "b.seq as seq,b.openId as openId from cabindoor as b inner join cabin as a "

+ "on a.id=b.cabinId where a.customId=1 and b.state=1 order by openId");

ResultSet rs = psmt.executeQuery();

while (rs.next()) {

Order order=new Order();

order.setCabinId(rs.getLong("cabinId"));

order.setDoorId(rs.getInt("seq"));

order.setOpenId(rs.getString("openId"));

orders.add(order);

}

} catch (SQLException e) {

e.printStackTrace();

}

return orders;

}

}

**CabinDoorDao.java**

致 谢

首先，我要感谢导师王奎甫和宋运隆老师。在设计期间，王老师和宋老师的给予我许多帮助，我深深的被他们渊博的学识和超强的动手实践能力所折服。宋老师也教了我许多为人处世的道理，并在我考研复试期间给我进行辅导。

其次，感谢我的任课老师，我去找他问问题的时候他很热心的帮我解答，并教我学习的思路，在得到老师的指点后，我豁然开朗，工作效率也变得高了起来。我相信这个经验以后一定能在我的学习生活中发挥重要作用。

此外，感谢在设计过程中给与我帮助的同学。这次有网络部分的传输，对所学的计算机网络课程有了更深的理解。感谢热心的网友，第一次使用MySQL和mina框架，在初次使用时连Eclipse的环境搭建都搭建错了，在遇到不会的问题时，就从网上找，虽然网上的信息很多，但在锲而不舍的寻找之后就像扒开层层迷雾，自己的思路也越来越清晰，再遇到问题解决解决起来也有了思考的思路，这对于我的软件开发之路有里程碑式的意义。

最后，再次感谢王奎甫老师和宋运隆老师在这段时间对我的悉心指导！