Research on Gateway Monitoring System of Internet of Things Based on Edge Computing*

*SHU Xuecheng, FANG Pingqing**

(*College of Mechanical and Automotive Engineering, Shanghai University Of Engineering Science, Shanghai* 201620, *China*)

Abstract: The monitoring system of Internet of Things gateway based on edge computing is designed for the problem of the diversity of communication protocols for Internet of Things devices and the shortage of cloud computing power. Internet of Things gateway adopt EdgeX Foundry edge computing platform framework, with Raspberry Pi 3B+as the core, adopt Docker virtualized container technology, and deploy the micro-service architecture in a distributed way. Through the combination of simulation and experiment, the transmission performance of Internet of Things gateway based on multiple communication protocols is tested and analyzed. A temperature and humidity alarm system based on Modbus communication protocol is constructed, and the average delay of the system is 376 ms. The experimental results show that the transmission performance of the Internet of Things gateway meets the needs of factories, and provides a new idea for data acquisition and control of equipment, and has high application value.

Key words: edge computing; Internet of Things; gateway monitoring system; Raspberry Pi 3B+

EEACC: 6150P **doi: 10.3969/j.issn.1005-9490.2019.06.043**

基于边缘计算的物联网网关监控系统的研究*

殳雪城,范平清*

(上海工程技术大学机械与汽车工程学院,上海 201620)

摘 要: 针对物联网设备通讯协议的多样性和云端计算力不足的问题,设计了一种基于边缘计算的物联网网关的监控系统。采用 EdgeX Foundry 边缘计算框架,以树莓派 3B+为核心,运用 Docker 虚拟化容器技术,分布式部署微服务架构。通过仿真与实验相结合的方法,测试分析了基于多种通讯协议的物联网网关的传输性能;构建了基于 Modbus 通讯协议的温湿度报警系统,得出系统的平均时延为 376 ms。实验结果表明,物联网网关的传输性能满足工厂需求,为设备的数据采集与控制提供了新的思路,具有很高的应用价值。

关键词: 边缘计算;物联网;网关监控系统;树莓派 3B+

中图分类号: TP277 **文献标识码: A** **文章编号: 1005-9490(2019)06-1569-05**

随着物联网(IOT)的快速发展,产业的升级,也对技术的演进提出了更高的要求。据 IDC(互联网数据中心)数据统计,到 2020 年将有超过 500 亿的终端与设备联网,这将导致终端数据出现井喷式增长,海量的数据连接需要计算能力更高的物联网网络架构实现数据的处理与分析[1]。与此同时,物联网的业务不断扩展,在智能交通,智能电网,智能家居以及智慧农业上的应用更加广泛,许多应用场景,如智能交通中实时路况的信息采集,自动驾驶中的传感器的数据采集与处理等等。在物联网快速发展的同时,越来越多的技术难点需要解决,例如:设备的通讯协议多种多样,数据传输性能低,网络带宽与延迟压力大等[2]。

边缘计算(edge computing)的诞生解决了物联网发展瓶颈的燃眉之急。边缘计算[3-4]是在靠近数据源头的网络边缘侧,融合网络、计算、存储及应用等核心能力的开放平台,就近提供边缘智能数据处理服务,以满足网络敏捷连接、实时业务、数据优化等应用需求。可以有效地缓解云计算平台的数据处理的负担,提高了数据处理的效率。同时,由于终端

项目来源:国家自然科学基金项目(51675324);国家青年自然科学基金项目(51505275)
收稿日期:2019-01-23 修改日期:2019-05-28

与边缘更贴近边缘侧，极大的降低了数据传输的延时性，解决了数据延时性的问题[5]。

本文提出了一种基于边缘计算的物联网网关的监控系统，系统主要包括感知传感器节点的部署，物联网网关硬件和软件平台。研究的目的是测试分析基于 EdgeX Foundry 边缘计算平台的物联网网关的数据传输性能；构建温湿度报警系统，添加边缘计算的规则，实现网关具有边缘侧处理数据的功能，降低数据传输的延时。

1 边缘计算平台

边缘计算要具备时效性、安全性、计算能力，并且部署在边缘侧，目前全球各大公司都加入了边缘计算平台系统的开发中。本文列举了一些已经开发了并部署的边缘计算开源系统。

Apache Edgent[6] 是一个可编程的模型，可以在本地上的边缘设备或网关上实时地处理数据流。数据由 Edgent 确定在边缘设备或后端系统上存储或分析。Edgent 使应用程序能够从发送连续的原始数据流转换为仅向服务器发送基本且有用的数据。因此，发送和存储到服务器的数据量可以显著减少。

OpenStack[7] 是一个云操作系统，它使用数据中心来控制计算、存储和网络资源，通过仪表板以及 web 界面向用户提供管理工具。OpenStack 的基础架构可以部署在边缘设备上，而 OpenStack 的分布式软件为虚拟机和容器技术提供支持，而虚拟机和容器技术是实现边缘计算的关键技术。

EdgeX Foundry[8] 是一系列松耦合、开源的微服务集合，位于网络的边缘，可以与设备、传感器、执行器和其他物联网对象的物理世界进行交互。EdgeX Foundry 旨在创造一个互操作性、即插即用、模块化的物联网边缘计算的生态系统。

除了一些边缘计算开源系统，还有一些商业的边缘计算系统，例如微软的 Azure IoT Edge、Google Cloud IoT 和亚马逊的 AWS Greengrass 等。这些商业边缘计算系统可以支持来自云服务提供的高级的数据分析以及网络边缘的人工智能等。

Azure IoT Edge[9] 将数据分析从云端迁移到边缘设备。Azure IoT Edge 有 IoT Edge 模块，IoT Edge 运行时和 IoT Edge 云界面交互。IoT Edge 运行时允许边缘设备使用自定义逻辑和云逻辑，IoT Edge 模块以 Docker 容器的形式在边缘设备上运行，执行 Azure 服务，第三方服务或自定义代码。IoT Edge 云界面允许用户分发和监控边缘设备上的工作负载。

Amazon AWS Greengrass[10] 是一款能够在边缘设备上实现本地计算、消息收发、数据缓存、同步和分析的软件。Greengrass 的核心是运行时，可以本地执行 AWS Lambda 代码，实现消息传递、数据缓存和安全策略的功能。

本文选取边缘计算平台 EdgeX Foundry 作为物联网网关的框架，其旨在为“物联网边缘计算”创建公共开放的框架。EdgeX Foundry 的边缘计算系统框架如图 1 所示。

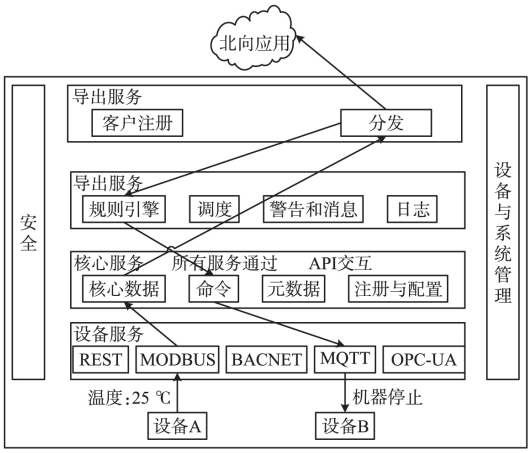


图 1 EdgeX Foundry 的边缘计算系统框架

由图 1 可看出 EdgeX Foundry 框架从南到北依次为：设备服务层、核心服务层、支持服务层、导出服务层。框架分为南北两侧。南侧：在物理领域内的所有物联网对象，以及与这些设备、传感器、执行器和其他物联网对象直接通信并从中收集数据的网络边缘，统称为“南侧”。北侧：将数据收集、存储、聚合、分析并转换为信息的云（或企业系统），以及与云通信的网络部分称为网络的“北侧”。EdgeX Foundry 可以使数据向北侧移动到云端，也可以横向移动到其他网关，或者返回到设备、传感器和执行器。

4 个服务层从南侧到北侧分别为：设备服务层、核心服务层、支持服务层、导出服务层。

两个增强的基础系统服务从分别为：系统管理，安全基础设施。

设备服务层（DS）：设备服务层负责与南向设备交互。

核心服务层（CS）：核心服务层分隔了边缘的北侧和南侧层。核心服务包括以下组件：核心数据，命令，元数据，注册和配置。。

支持服务层（SS）：支持服务层包含广泛的微服务，该层微服务主要提供边缘分析服务和智能分析服务。

导出服务层（ES）：在必要情况下，EdgeX Foundry 需要可以独立于其他系统运行。EdgeX Foundry 在不连接北向应用的情况下，是可以长时间

独立运行的。导出服务层还提供了一组微服务实现以下功能：北向应用可以在网关注册，并获取其想获得的南向设备的数据；设置数据发向的方向；设置数据传输的格式。

设备与系统管理：提供 EdgeX Foundry 微服务的安装、升级、启动、停止和监视，以及 BIOS 固件、操作系统和其他与网关相关的软件。

安全：EdgeX Foundry 内外的安全元件保护由 EdgeX Foundry 管理的设备、传感器和其他物联网对象的数据和命令。

2　基于多种通讯协议的数据传输性能测试

在物联网世界里，设备的通讯协议有各式各样的，协议标准很难统一，EdgeX Foundry 中的设备服务层的微服务可以通过每个物联网对象本身的协议与设备，传感器，执行器和其他物联网对象进行通信。设备服务层将由物联网对象生成和传递的数据转换为常见的 EdgeX Foundry 数据结构，并将转换后的数据发送到核心数据以及接收来自于命令微服务的命令。其框架如图 2 所示。针对不同的通讯协议，本文通过将模拟设备发送的数据包与接收到的数据包进行计算，得到丢包率；从发送命令给设备到客户端接收到数据为止，两者之间的时间差，得到时延，进而测试此框架的数据传输性能。

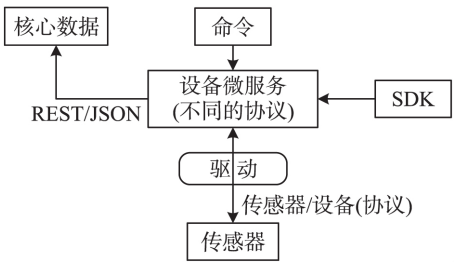
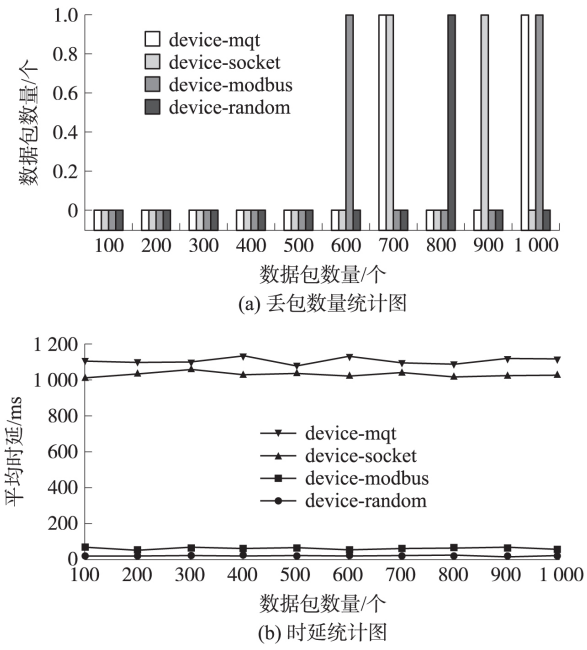


图 2　设备微服务框架图

本文的测试环境是混合编译，在树莓派 3B+上通过 Docker[11] 部署除了设备微服务以外的其他微服务，在 PC 端运行设备微服务。在导出端注册一个客户端，支持 MQTT 通讯协议，即在本地 PC 上运行一个 MQTT 服务器[12]。所测试的设备微服务为 device-modbus，device-random，device-mqtt 和 device-socket。其所仿真的模拟设备都具有主动发数据的能力，且传输的数据包较小。设定导出层接收 100 个数据包，便进行一次丢包率的统计，统计 10 次结果，其统计结果如图 3(a) 所示。设定导出层接收 100 个数据包，便进行一次延时的统计，统计 10 次结果，其统计结果如图 3(b) 所示。



(a) 丢包数量统计图

(b) 时延统计图

图 3　传输性能统计图

从图 3 可以看出，device-mqtt 的丢包率为 0.3%，device-socket 的丢包率为 0.2%，device-modbus 丢包率为 0.2%，device-random 丢包率为 0.1%，4 种不同的设备微服务的丢包率差别不大；device-mqtt 的平均时延为 1 026.5 ms，device-socket 的平均时延为 1 105.8 ms，device-modbus 的平均时延为 61.5 ms，device-random 平均时延为 21.1 ms，四者差距较大，主要原因有两点。第一点：device-mqtt 和 device-socket 是用 Java 语言编写，而 device-modbus 和 device-random 是用 Golang 语言编写的，具有高并发性的特点，相比较 Java 语言，Golang 语言有利于提高项目工程运行的效率；第二点：各种通讯协议的机制不一样，在一定的程度会有所影响。综上，数据的传输性能不仅与编程语言有关还与通讯协议有关，同时此框架可以支持不同通讯协议的设备微服务的扩展，综合考虑上述因素，本文在下文的物联网网关的应用上，采用 Golang 语言编写的框架，使用 device-modbus 的设备微服务。

3　物联网网关应用实例

工厂的设备间存放各种贵重、易燃易爆物品，需要实时监控设备间的温度，并即时提供高温报警。传统的读取温湿度数据需要人工读取，随着 WiFi 技术的产生，温湿度的读取通过 WiFi 模块，将温湿度的数据在网页上显示，上传到云端，在云端处理数据[13]。但是全国类似设备间有非常多，而且每秒要传输大量数据给云端，大量的数据运算给云端以及网络带宽造成巨大压力，并且云端响应的实时性较

差，而且还要面临远程网络中断问题，一旦出现网络中断，即有可能导致命令下发失败等问题。本文针对这一现实场景，将基于边缘计算的物联网网关应用到工厂的温湿度报警系统中，解决实际问题。其主要解决方案如表 1 所示。

表 1 解决方案

需求	解决方案
在采集数据不减少的情况下减轻云端压力	部署边缘计算网关，将计算能力下移
降低报警延时	在边缘计算网关中添加业务处理能力
解决与云端网络中断后，无法报警问题	在边缘计算网关中实现命令的发送

3.1 系统架构

系统的整体框架如图 4 所示，感知层由温湿度传感器，蜂鸣器，继电器电源控制开关等组成，传输层是整个系统的核心，采用 EdgeX Foundry 的边缘计算平台的框架，树莓派 3B+开发板充当物联网网关，实现数据的上传，存储，分析和命令的下发，应用层利用 ZUUL 的路由转发功能，设计了数据可视化界面，简化操作人员的操作。

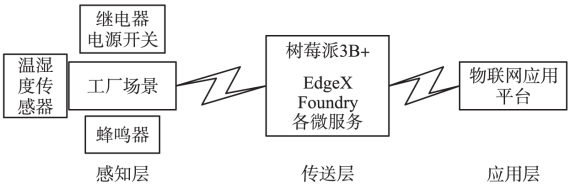


图 4 系统架构图

实验包括两部分，一是感知层温湿度传感器数据的上传，二是物联网应用平台添加设备，通过规则引擎微服务设定特定的规则。根据实验的目的，搭建了基于开源的边缘计算系统的物联网网关，采用基于 Modbus 的温湿度传感器采集温度和湿度，控制设备选用的是基于 Modbus 的继电器电源开关，控制蜂鸣器的开与关。

3.2 定时采集温湿度传感器

传统的温湿度数据是通过 AT 指令集来读取温湿度传感器的数据，或者通过上位机读取温湿度的数据，其效率比较低，本文基于边缘计算的任务调度微服务，将数据采集频率相等的设备归为一类，实现任务的统一调度，解决了定时采集温湿度传感器数据的问题，为大型工厂的任务调度提供新的思路。任务调度的机制是一个任务调度(schedule) 包含多个任务调度事件(scheduleevent) ，scheduleevent 通过 addressable 中 path 的地址发起 HTTP 请求。本文设定每 3 s 采集一次温湿度传感器的温度数据。

3.3 数据传输性能测试

本节使用的传感器设备是被动发数据即用户需要给传感器发送采集命令，传感器收到命令后，采集数据，上传数据，与上文模拟设备具有不同的特点。其延时为从命令下发到导出端导出数据为止。丢包率与上文定义无差别。经过测试，得出连接实际设备的丢包率和延时率如图 5 所示。

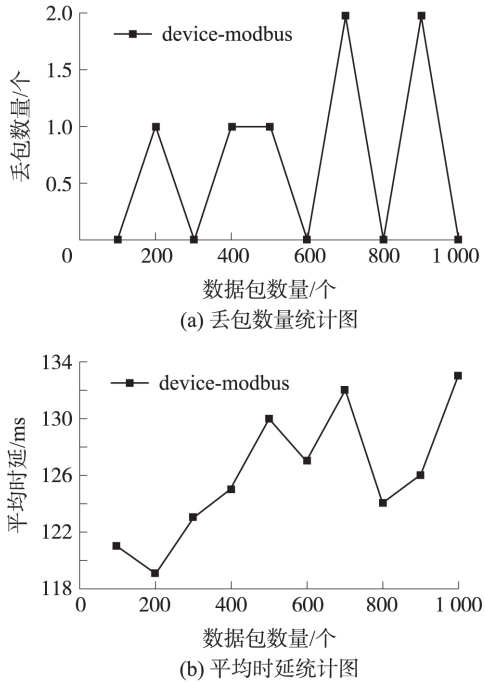


图 5 温度采集测试统计图

从图 5 可以得出，随着物联网网关接受的数据包的个数的增加，每 100 个数据包中丢包率在上升，延时也随之上升，这与设备运行时间的长短有一定的关联，整体的丢包率为 0.7%，平均时延为 126 ms，对于温湿度采集系统，是满足要求的。与上文的延时相比较，延时大大的提升，这与命令的下发的时间和设备内部的响应有关，属于可接受的范围，达到了实验预期目标。

3.4 温湿度报警系统测试

温度报警系统整体框架流程图如图 6 所示。温湿度报警系统的工作原理，物联网网关定时采集温湿度数据，并在物联网网关中通过规则引擎来设定

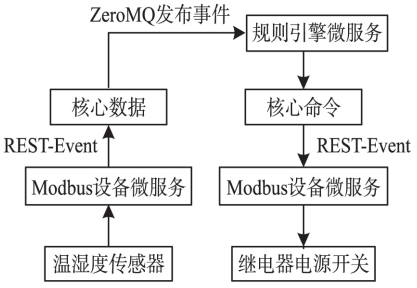


图 6 温湿度报警框架图

特定的规则，当温湿度传感器采集的温度大于某一特定的值，则会触发规则引擎，继电器电源开关切换到开的状态，蜂鸣器报警。

图 6 表示，核心数据接受到设备微服务发送的数据，并持久化储存，规则引擎订阅核心数据的数据，核心数据通过 ZeroMQ 消息队列发布数据，规则引擎接收到数据进行边缘计算，触发命令，核心命令微服务将命令通过设备微服务发送给控制设备，实现设备的控制。

搭建好温湿度报警系统后，编写继电器电源开关的设备文件，在物联网应用平台上添加设备，本文通过 Postman 的 post 的命令上传规则。传统的做法是设定某一特定的温度，即设置单门限来触发报警规则，这样的方式，会使继电器电源开关频繁的开闭，影响设备的使用的寿命。改进后的方式为设定上下限阀值，将单门限改为双门限，避免抖动，即当上传的温度数据达到 40 ℃时，温度继续上升至 42 ℃时，触发边缘计算规则，继电器电源开关设定为打开（即设置值为 1），当上传的温度数据达到 40 ℃时，温度继续下降至 38 ℃时，触发边缘计算规则，继电器电源开关设定为关闭（即设置值为 0），生成的 2 个规则文件，如图 7 所示。

本文通过吹风机给温湿度传感器加热，查看规则引擎的日志微服务，即可查看规则执行结果，本文做了 10 组实验，得出整个报警系统的平均时延为 376 ms，符合工厂生产要求。

```
package org.edgexfoundry.rules;
global org.edgexfoundry.engine.CommandExecutor executor;
global org.edgexfoundry.support.logging.client.EdgeXLogger logger;
import org.edgexfoundry.domain.core.Event;
import org.edgexfoundry.domain.core.Reading;
import java.util.Map;
rule "rule-on-02"
when
  $e:Event($rlist: readings && device=="test-temperature-01")
  $r0:Reading(name=="Temperature" && Float.parseFloat(value) > 42) from $rlist
then
executor.fireCommand("5c42ed119f8fc200017b619f","5c42ece89f8fc200017b619c",
"{\"Switch\":\"1\"}");
logger.info("Patlite warning triggered for temperature too high");
```

(a) 继电器开

```
package org.edgexfoundry.rules;
global org.edgexfoundry.engine.CommandExecutor executor;
global org.edgexfoundry.support.logging.client.EdgeXLogger logger;
import org.edgexfoundry.domain.core.Event;
import org.edgexfoundry.domain.core.Reading;
import java.util.Map;
rule "rule-off-02"
when
  $e:Event($rlist: readings && device=="test-temperature-01")
  $r0:Reading(name=="Temperature" && Float.parseFloat(value) < 38) from $rlist
then
executor.fireCommand("5c42ed119f8fc200017b619f","5c42ece89f8fc200017b619c",
"{\"Switch\":\"0\"}");
logger.info("Release alarm");
```

(b) 继电器关

图 7　规则文件

4　结语

本文针对物联网的痛点，设备通讯协议复杂多样，数据上传云端导致延时高的问题，设计了基于开源的边缘计算的物联网网关，运用开源的 EdgeX Foundry 的框架和 Docker 虚拟化容器技术，在树莓派 3B+上部署微服务，分布式部署微服务有利于开发人员的二次开发，增添特定的微服务，通过物联网应用平台实现设备的管理与控制，简化了用户的操作。

本文的创新点在于将 EdgeX Foundry 边缘计算的理论首次应用到物联网网关中，验证了该物联网网关在丢包率和延时性上都比较稳定；构建了基于边缘计算的温湿度报警系统，设定了定时采集数据任务，实现任务的统一调度，同时通过双门限避免控制抖动，触发边缘计算，达到将计算从云端迁移到边缘侧的目的，降低了数据传输与处理的延时，为工厂设备数据的采集与设备的控制提供了新的思路。

参考文献：

[1] Georgakopoulos D，Jayaraman P P，Fazia M，et al. Internet of Things and Edge Cloud Computing Roadmap for Manufacturing [J]. IEEE Cloud Computing，2016，3(4)：66-73.

[2] 赵梓铭，刘芳，蔡志平，等. 边缘计算：平台、应用与挑战 [J]. 计算机研究与发展，2018，55(2)：327-337.

[3] Satyanarayanan，Mahadev. The Emergence of Edge Computing [J]. Computer，2017，50(1)：30-39.

[4] Shi W，Cao J，Zhang Q，et al. Edge Computing：Vision and Challenges [J]. IEEE Internet of Things Journal，2016，3(5)：637-646.

[5] 施巍松，孙辉，曹杰，等. 边缘计算：万物互联时代新型计算模型 [J]. 计算机研究与发展，2017，54(5)：907-924.

[6] Belli L，Cirani S，Davoli L，et al. An Open-Source Cloud Architecture for Big Stream IoT Applications [M] //Interoperability and Open-Source Solutions for the Internet of Things. Springer International Publishing，2015：73-78.

[7] Lebre A，Pastor J，Simonet A，et al. Revising OpenStack to Operate Fog/Edge Computing Infrastructures [C] //2017 IEEE International Conference on Cloud Engineering(IC2E). IEEE，2017：138-148.

[8] Na W，Lee Y，Dao N N，et al. Directional Link Scheduling for Real-Time Data Processing in Smart Manufacturing System [J]. IEEE Internet of Things Journal，2018：1-1.

[9] Familiar B，Barnes J. Business in Real-Time Using Azure IoT and Cortana Intelligence Suite [M]. Apress，2017：291-350.

[10] Tärneberg，William，Chandrasekaran V，Humphrey M. Experiences Creating a Framework for Smart Traffic Control using AWS IOT [C] //IEEE/ACM International Conference on Utility & Cloud Computing. IEEE，2017：63-69.

（下转第 1577 页）

[9] 唐颖,张凡,郭勇. 移动机器人的超声波测距传感器设计[J]. 传感技术学报,2010,23(11):1699-1703.

[10] 冯志辉,刘恩海,岳永坚. 基于 FPGA 延迟线插入法的半导体激光测距[J]. 光电工程,2011.4:53-59.

杨保海(1978-),男,汉族,河南郑州人,硕士,副教授,主要研究方向为信号处理,图像处理和电子技术方向的教学与研究工作;

宋俊慷(1989-),男(汉族),陕西汉中人,硕士,工程师,主要研究方向为电子信息物联网应用技术;

黎运宇(1974-),男,土家族,湖南张家界人,硕士,高级实验师,研究方向为电子系统设计;

黄梅春(1976-),女,壮族,硕士,讲师,广西天等人,研究方向为电子系统设计。

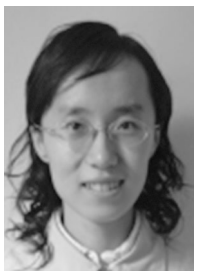
(上接第 1573 页)

[11] 蒋安国,王金泉,边晋炜. Docker 技术之 Docker-Compose 研究[J]. 现代信息科技,2018,2(10):75-77.

[12] 任亨,马跃,杨海波,等. 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.

[13] 康明星,匡炎. 基于 WiFi 的机房温湿度监测网络[J]. 物联网技术,2018,8(8):15-16,19.

殳雪城(1994-),男,汉族,江苏省如皋人,上海工程技术大学机械与汽车工程学院硕士在读,研究方向为边缘计算,物联网应用,1136610622@qq.com;



范平清(1980-),女,汉族,山东省泰安人,上海工程技术大学,副教授,硕士生导师,研究方向: 机械系统动力学和压电制动器,fanpingqing@163.com。