Sensors and Materials, Vol. 33, No. 2 (2021) 775–788
MYU Tokyo

S & M 2494

Novel OneM2M Communication Mechanism Based on Labeling of IoT Devices

Dong-Meau Chang,[1] Yao-Hong Tsai,[2*] Tse-Chuan Hsu,[3] and Hsien-Wei Tseng[1**]

[1]School of Mathematics and Information Engineering, Longyan University, Fujian,
No. 1, North Dongxiao Rd., Longyan City, Fujian Province 364012, China
[2]Department of Information Management, Hsuan Chuang University, Hsinchu,
No. 48, Hsuan Chuang Rd., Xiangshan Dist., Hsinchu City 300, Taiwan
[3]Department of Healthcare Information & Management, Ming Chuan University/Hsuan Chuang University,
No. 48, Xuanzhuang Rd., Xiangshan Dist., Hsinchu City 300, Taiwan

(Received July 20, 2020; accepted January 20, 2021)

Keywords: OneM2M, IoT, communication, computing model

The increasing demand for food has raised the profile of the agricultural sector. The Internet of Things (IoT) is an important technology that is capable of solving many problems in modern agriculture. In traditional IoT communication technology, the operation technology of a single platform through remote or decentralized management is reasonably mature. In such an environment, all devices need to return the results of their execution tasks to the core server, and the server dispatches further script tasks to be performed. However, there is a problem in the mechanisms used for communication management of the devices at the same time when a large amount of data needs to be transmitted. The machine-to-machine (M2M) service platform has been standardized to enable the communication of devices, which is the basis for smart environments with IoT. Recently, the OneM2M standard communication mechanism has provided a way for devices to send messages to core servers and assign specific tasks to other devices, and a two-way communication mechanism capable of returning and sending instructions has been realized. In this study, an innovative label computing model combined with the M2M communication mechanism was proposed, and labels were used to record events in a standard way. By notifying all devices of the result of the labeling through the M2M mechanism, the cost of data transmission among the devices of a network can be reduced. The proposed model can also automatically determine how to assign and execute IoT operation tasks through the labeling mechanism. Finally, the proposed model was implemented in a smart agriculture environment to increase agricultural production. The execution time on a simulated smart agricultural system was reduced by about 30% compared with that on an existing system.

1. Introduction

The Internet of Things (IoT) and smart manufacturing have become very popular industrial and business concepts in recent years. Through the Internet, relevant equipment data can be

*Corresponding author: e-mail: yaohong.tsai@gmail.com
**Corresponding author: e-mail: hsienwei.tseng@gmail.com
https://doi.org/10.18494/SAM.2021.3045

ISSN 0914-4935 © MYU K.K.
https://myukk.org/

controlled from remote centers. By using embedded systems or software design methods, all equipment-operating systems are written into the chip in sensors to allow the equipment to help operators control a smart environment. The future IoT world is envisaged to contain a large number of smart sensors, actuators, controllers, and robots, which are connected and integrated into existing systems to maximize the efficiency and throughput of operations and satisfy human expectations.(1) The rapid increase in the number of devices has led to a new model of computing technology in which all the devices become interactive. In this new computing model, machines talk to one another to accomplish tasks automatically in different application scenarios.(2)

However, when IoT devices with sensors are connected through a network, some problems may arise when all devices receive a task through a centralized platform. Connectivity and communication are critical building blocks for machine collaboration and process optimization.(3) Studies in these fields have attracted much interest in both academia and industry.(4) Machine-to-machine (M2M) communication allows connected sensors to exchange information with each other autonomously, enabling tasks to be assigned and information to be collected.(5) A large volume of data is generated and stored by all the connected machines for observation, analysis, and further studies.(6) OneM2M is a global partnership project founded in 2012 whose goal is to create a global technical standard for M2M and IoT technologies to support a wide range of applications and services such as smart cities, smart grids, connected cars, home automation, public safety, and health.(7) Such a standard can be used to assist network communication packets to intelligently find routes for communication transmission. Collision detection technology for communication packets and traffic service quality management algorithms enable the smooth transmission of digital data in packets to the destination, ensuring that data transmission packets are not lost.

However, if a packet delay occurs, a system cannot determine whether a service is still being provided, so it will not be able to effectively grasp the status of the device service after a task is assigned. This situation limits the passive operation of devices, making it impossible to perform a two-way active response operation.(8) The data response operation between devices cannot be regularly obtained and devices cannot respond to the server for subsequent operations. There is a lack of good communication design service architecture for the present IoT technology.(9) In most IoT models, the active one-to-one operation of hardware device system services is mainly performed by humans. In particular, when service applications must be implemented through IoT, it is necessary to perform a large number of device operations, which cannot be performed at the same time.(10)

The OneM2M standard communication mechanism provides a way for devices to assign messages to core servers and specific tasks to other devices to perform operations. This communication mechanism implements bidirectional return and dispatch instructions.(11,12) In this study, an innovative label-computing model combined with the M2M communication mechanism was proposed. This model was designed to manage events in a standard way, and labels were used to record each event. By notifying all devices of the result of the labeling through the M2M mechanism, the cost of transmitting data among the devices of a network can be reduced. The proposed model can also automatically determine how to assign and execute

IoT operation tasks through the labeling mechanism. Experimental results showed the efficiency of the proposed model. The variation of the response time was limited to within a small interval as the number of devices was increased. The execution time on a simulated smart agricultural IoT system was reduced by about 30% compared with that on an existing system.

2. System for M2M Based on Fog Computing

2.1 Related work

Even though IoT systems can transmit data through the Internet, there is still no effective system to manage communication among devices for various applications. When cloud computing is applied, devices can be shared by different users by adopting dynamically configured calculations. If an IoT system can expand an I/O device to the sensing layer, the data collected by the end device can be integrated with the cloud.(12) This allows the sensing device at the "fog" end to gather information and perform real-time operations. Guinard(13) combined cloud computing technology with the IoT Electronic Product Code (EPC) global network technology architecture to build an EPC cloud and apply it to electronic goods security systems. Connected via the cloud platform, the IoT system attempts to systematically implement the SaaS system through the cloud technology of the IoT device and successfully overcomes the difficulties faced by remote management.

Hsu *et al.*(14) took advantage of the M2M communication structure developed by OSGi to construct a message server for task assignment. When a service was delivered from the message server to the end nodes, the devices activated data exchange and task assignment services by themselves. The information of each task and the result of the script were synchronized according to the order of task assignment to make sure the output was correct. When the device required a program update or data exchange, it was assigned by an M2M system to increase the scope of services. Hsu *et al.* explored an IoT technology that can be used to improve the communication management of a large number of devices. All terminal IoT devices were built with a fog platform computing model and provided a layer of "fog" between the terminal device and the cloud data center. This greatly reduced the computing and storage load of the cloud, improved the system efficiency, increased the transmission rate, and reduced the latency analysis time.

Hsu *et al.* designed new calculation methods and adjusted the devices to increase their performance. When the system detected that the service response time was too long, i.e., exceeding T s, the corresponding S1 event was defined in the model. In their experiments, if the waiting time exceeded T s, the system sent an S1 event requesting a communication response. In this scenario, the IoT platform sent a confirmation service status detection message and waited for the response. If the service message response time was still more than T s, the system recorded the service time in the database to show the abnormal signal of the device in the platform. It recorded the delay frequency as N times. When the delay event occurred for the second time, the system initiated a detection process multiple times to send the detection message and performed $(N - 1) \times T$ times for the calculation of synchronous message detection.

In practice, this method can cause serious problems if network congestion occurs. For example, if a packet delay occurs, the system cannot establish whether the terminal device is still providing services, so the service status of the device cannot be confirmed effectively after the event is assigned. Furthermore, when packet reception is delayed, the calculated value on the core server will vary largely. If the system only uses average values, the results of the difference in values will further affect the execution procedures of the devices.

2.2 Simulations on smart agricultural system

We built a smart agricultural IoT system for simulating the proposed method. Let the device calculated on the detection device side be based on the temperature change. When a temperature change occurs, the system increases the number of detections and returns the analysis results to the core server for further decision-making. Then the average temperature is calculated through the core server and compared with the current preset value. If it is not within the tolerance range, tasks and parameters are sent to the environmental control devices and the control program starts to increase or decrease the temperature.

For example, if our temperature sensor detects a current temperature of 26 °C, the sensing device will adjust the detection frequency flexibly based on the current temperature. Assuming the detection results are not within the tolerance range of 26–28 °C, the detection frequency is reduced. Otherwise, the detection frequency is increased when the detection result falls within the interval. The simulation results are shown in Table 1.

The results of this comparison of temperature changes are returned to the core server for further subsequent environmental control processing. When the core server receives the returned value, it sends the environmental control task to the fan to start it and adjust the temperature. However, if an event delay occurs at this time, the situations shown in Table 2 may occur.

It is observed from the above results that the sensor startup time is automatically delayed when the data from the sensing device cannot be returned to the core server. The third and

Table 1
Simulation results of the smart agricultural IoT system.

Event	1	2	3	4	5	6
Time	60	120	180	120	60	120
Current temperature	26	27	24	24	26	25
Tolerance range	26–28	26–28	26–28	26–28	26–28	26–28

Table 2
Simulation results when packet delay occurs.

Event	1	2	3	4	5	6
Time	60	120	180	120	180	120
Current temperature	26	27	24	24	23	24
Tolerance range	26–28	26–28	26–28	26–28	26–28	26–28
Packet status			Delay/Loss	Delay/Loss		

fourth events in Table 2 are either delayed or lost. However, at this time, the average temperature drops, and the fifth event still maintains the original detection frequency instead of increasing the detection frequency in the fourth and fifth events.

3. Labeling of IoT Sensors

3.1 Motivation

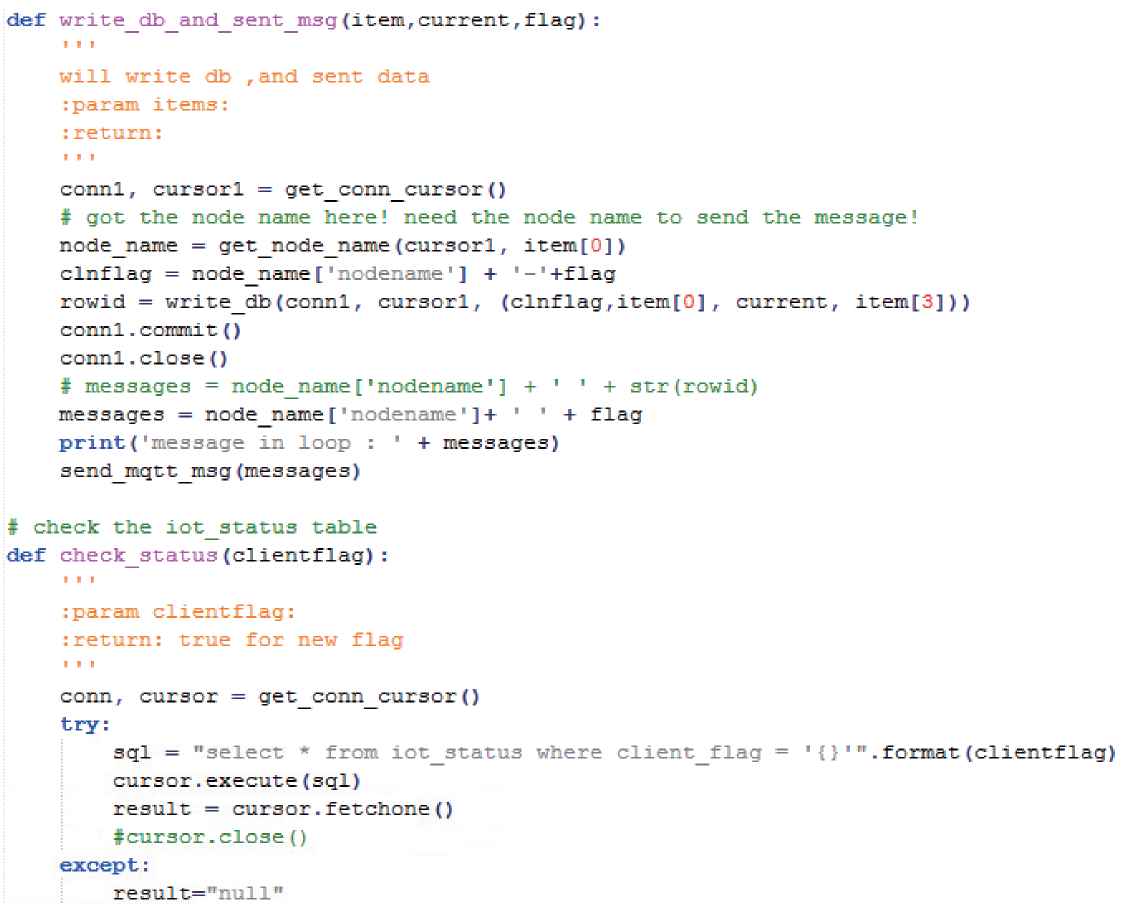
When discussing the situation caused by the delay, there are two possible states. The first is the devices start to perform analysis and calculation after they obtain the script, and the time cost required for the calculation and analysis exceeds the preset time. Because the total calculation time exceeds the original setting, the device will not be able to return data because the calculation is still in progress. The second state is that the device has calculated and analyzed the results, but the transmitted data cannot be returned to the server as expected owing to insufficient network performance or other problems. Regarding these two states, a fog decentralized calculation method(14) that can eliminate some delay events was previously proposed. However, the mechanism still needs to be adjusted to avoid repeatedly opening services due to a long analysis and calculation time or a delay in network status transmission.

Toward solving the above problems, a flag mode was added in this study to allow the core server to check and verify through the use of flags. Through the data flag return status, the core server can establish whether the sensing device and the environmental control device have the same flag status so that they can process the event after each data return and so that tasks can be sent between the end devices in the platform system. This method can greatly reduce the time difference between events after a miscalculation. At the same time, the sensing device directly sends the event status to the gateway and notifies the environmental control device of the execution status. This model can be used to solve the problem of a delayed startup or the multiple startup of environmental control devices after a frequency miscalculation or network transmission delay.

3.2 Labeling modules

In this study, a creative technology platform is established, so that the task script can be transmitted to all nodes through the platform. Nodes first install scripts and then perform tasks. After the end device starts the service, the back end of the system communicates with the creative service platform. The platform next waits to receive data online. If the device does not receive the data from the connected node, the platform agent resends the task script and asks the device to provide the data again.

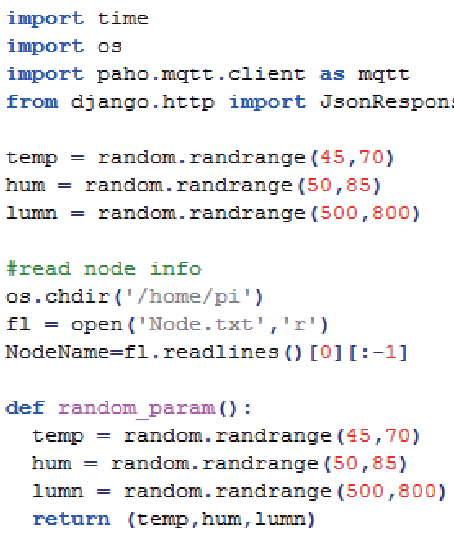
We designed two sets of modules on the server, **iot_subscribe_write_to_db** and **iot_server_sent_msg**. They are used to record flag events and respond to data events. On the terminal device, we established **pi_subscribe_mqtt_message** for flag synchronization. At the same time, we used Python event records to compare and verify event responses. The source codes were written in Python language as shown in Figs. 1–3.



```python
def write_db_and_sent_msg(item,current,flag):
    '''
    will write db ,and sent data
    :param items:
    :return:
    '''
    conn1, cursor1 = get_conn_cursor()
    # got the node name here! need the node name to send the message!
    node_name = get_node_name(cursor1, item[0])
    clnflag = node_name['nodename'] + '-'+flag
    rowid = write_db(conn1, cursor1, (clnflag,item[0], current, item[3]))
    conn1.commit()
    conn1.close()
    # messages = node_name['nodename'] + ' ' + str(rowid)
    messages = node_name['nodename']+ ' ' + flag
    print('message in loop : ' + messages)
    send_mqtt_msg(messages)

# check the iot_status table
def check_status(clientflag):
    '''
    :param clientflag:
    :return: true for new flag
    '''
    conn, cursor = get_conn_cursor()
    try:
        sql = "select * from iot_status where client_flag = '{}'".format(clientflag)
        cursor.execute(sql)
        result = cursor.fetchone()
        #cursor.close()
    except:
        result="null"
```

Fig. 1. (Color online) **iot_server_sent_msg** for flag setting.



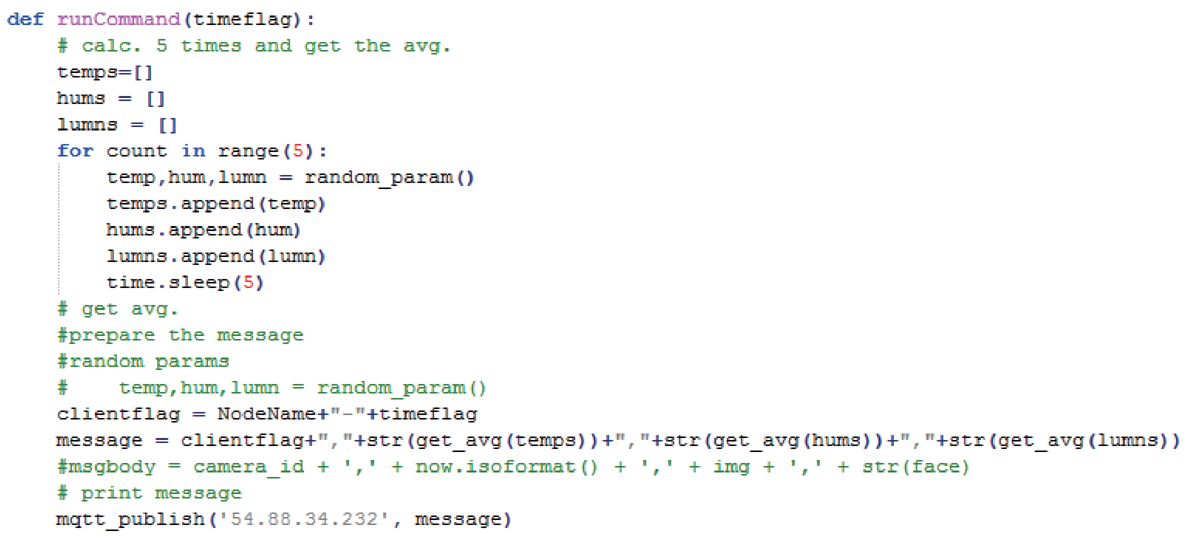
```python
import time
import os
import paho.mqtt.client as mqtt
from django.http import JsonResponse

temp = random.randrange(45,70)
hum = random.randrange(50,85)
lumn = random.randrange(500,800)

#read node info
os.chdir('/home/pi')
f1 = open('Node.txt','r')
NodeName=f1.readlines()[0][:-1]

def random_param():
  temp = random.randrange(45,70)
  hum = random.randrange(50,85)
  lumn = random.randrange(500,800)
  return (temp,hum,lumn)
```

Fig. 2. (Color online) **iot_subscribe_write_to_db** server setting in standard flag range.



```
def runCommand(timeflag):
    # calc. 5 times and get the avg.
    temps=[]
    hums = []
    lumns = []
    for count in range(5):
        temp,hum,lumn = random_param()
        temps.append(temp)
        hums.append(hum)
        lumns.append(lumn)
        time.sleep(5)
    # get avg.
    #prepare the message
    #random params
    #    temp,hum,lumn = random_param()
    clientflag = NodeName+"-"+timeflag
    message = clientflag+","+str(get_avg(temps))+","+str(get_avg(hums))+","+str(get_avg(lumns))
    #msgbody = camera_id + ',' + now.isoformat() + ',' + img + ',' + str(face)
    # print message
    mqtt_publish('54.88.34.232', message)
```

Fig. 3. (Color online) **pi_subscribe_mqtt_message** for flag synchronization.

4. Experimental Results

4.1 Parameter setting

The parameters of the creative service platform are preset through the core system. For example, the tomato planting temperature must be maintained at 25–30 °C. If it is lower than 5 °C or higher than 40 °C, the growth of plants stops. The light must be sufficient and the pH value of the soil should be between 6 and 6.5. The system platform sets parameters as variables under three scenarios to guarantee sweetness, multiple harvests, and normal settings. The parameter settings have the following three instructions: (A) start the circulating fan, (B) start the LED light, and (C) start the liquid fertilizer. The ranges of parameter settings for (A), (B), and (C) are 0–5, 0–5, and 0–5, respectively. After the initial setting, through the M2M mechanism, the core system sends parameters to the sensing device for monitoring. When the sensing device detects that the status value is in the parameter interval, it separately notifies devices A, B, and C to start their services, and returns the status results to the core server.

4.2 Improved transmission model

The transmission mechanism of each sensing device is innovatively adjusted. After the core server gives the parameter conditions, the sensing device performs the initial detection. In the initial state, the environmental information is detected every 60 min. In the following, all times are in minutes. After the detection, the environmental data are compared with the parameter conditions of the sensing device. If the state parameters are the same, the detection time is increased by the delay. Each detection time is corrected to $(N + 1) \times t$, $t = 60$, and $N = N + 1$, the next detection time is gradually increased, and the judgment cost is repeatedly written by increasing the time to reduce the interval after detection. If the detection result is not within the parameter conditions, the device notifies the other environmental control devices to start devices A, B, and C. The startup time of each device is 60 and the detection time is corrected to $(N - 1) \times t$, $t = 60$, if $N - 1 < 0$, $N = 1$, and the device is forced to gradually increase the detection range.

After the value of the sensing device is detected, all states are flagged and returned to the core server for storage. If the status has not changed, the value of the last code of the return flag is set to 0 and the serial numbers A, $A + 1$, and $A + 2$ are given. If the status has changed, the value of the last code of the return flag is set to 1 and the serial numbers B, $B + 1$, and $B + 2$ are given. In addition to notifying the environmental control device to start the program through the gateway, the status change results are stored in the core server, and the core server compares the status events and records the delayed transmission results through the serial number sequence. If the event occurs in the states of B, $B + 1$, $B + 2$, etc., a reload message is sent through the core server. The sensing device and the environmental control device must return the service status at the same time to reduce the problem of packet delay or loss.

4.3 Analysis of results

In this study, the previous research results in Sect. 2.2 are improved by employing a labeling mechanism. Through the initial parameter settings and M2M models for task dispatch, the mechanism provides communication and data exchange between sensing equipment and environmental control equipment. The improved transmission mechanism combines sending tasks by the core server with a flag-based returning mechanism to avoid packet delay or loss. At the same time, the detection of lost packets is used to force the reload task to the non-responding nodes to reduce the problem of delayed task execution. Under the same conditions, the sensing device and the environmental control device communicate directly through the gateway at the same time. The task execution results of both sides are transmitted back to the core server at the same time. When the detection status of the sensing state is the same as the initial parameter interval setting, the sensing device sends a record and flag to the core server (Table 3). When the result is outside the setting parameter interval, the environmental control device is notified and the status is written back to the database of the core server (Table 4).

Under normal conditions, the environmental control device does not receive any message and only synchronously responds to the server through the sensing device and performs flag recording. When the sensing value is not within the preset parameter range, the environmental

Table 3
Return form of execution procedure of sensing device.

Event	1	2	3	4	5	6
Time	60	120	180	120	180	240
Current temperature	26	27	24	24	26	25
Tolerance range	26–28	26–28	26–28	26–28	26–28	26–28
Flag	00 + 00 (A)	00 + 01 (A + 1)	11 + 00 (B)	11 + 01 (B + 1)	00 + 02 (A + 2)	—
Last code	0	0	1	1	0	—
Return code	00000	00010	11001	11011	00020	—

Table 4
Return form of execution procedure of environmental control device.

Event	1	2	3	4	5	6
Execution time	—	—	60	60	—	—
Fan status	—	—	1	1	—	—
Operation done	—	—	0	0	—	—
Flag	—	—	11 + 00 (*B*)	11 + 01 (*B* + 1)	—	—
Return code	—	—	11000	11010	—	—

control device is notified, the detection time is changed, and the data is transmitted back to the core server. The environmental control device usually waits for a notification from the sensing device. As shown in Table 1, a temperature change is detected at the third event when the data is received, indicating that the device needs to start the service. The fan device is started and operated for 60 s. When the service starts, the status is 1, otherwise the status is 0. The device sends the return form of the execution procedure of the environment control device: code 3, value 11000, code 4, value 11010. The core server performs a synchronous check to see whether the received value from the sensing device is 11001 (11011) for code 3 (code 4). A match indicates that the event handler is complete. If the comparison result does not match, it is first checked whether the flag 1100 is incorrectly connected to the code. If there is an error state, M2M reloading is started, and the whole system is resynchronized to confirm the code connection. When the code is correct but the sequence number is incorrect, the M2M unidirectional reloading mechanism is sent to restart the sensor or the environmental control device.

4.4 Performance improvement

Under the same environment setting, sensing devices were built to collect data from the smart agricultural IoT system, and the simulation results are shown in Table 5. First, we set the temperature range to 26–28 °C and set the flag. When the detected temperature falls within this range, the environment is in a stable temperature state. By using the delayed response algorithm, the frequency of the returned data can be dynamically adjusted. Therefore, it can be seen that when the temperature changes for the third event, the system automatically corrects the detection time for the fourth event. After the temperature is returned to the preset range for the fifth event, the detection time is extended for the sixth event. Using the same parameters, we let the packets of the third and fourth events be lost in the simulation. When the system does not receive the third and fourth data, it can be seen from Table 6 that the system does not have the third and fourth packets. Then a record is established and the frequency of the third time due to the detection, i.e., 180 s, is maintained. When the fifth packet is detected, the system continues to operate. However, this status will affect the total detection time. When no packet delay occurs, the time required to complete 10 events is 1140. Otherwise, the time required is 1560 after a delay occurs. Under this assumption of the simulation, if the average temperature is the same as that for the previous time, the system performance will markedly deteriorate.

Table 5
Simulation results for performance testing.

Event	1	2	3	4	5	6	7	8	9	10
Execution time	60	120	180	120	60	120	180	120	60	120
Package loss	—	—	—	—	—	—	—	—	—	—
Total time	60	180	360	480	540	660	840	960	1020	1140
Current temperature	26	27	24	24	26	26	25	25	26	26
Tolerance range	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28

Table 6
Simulation results when a packet delay occurs.

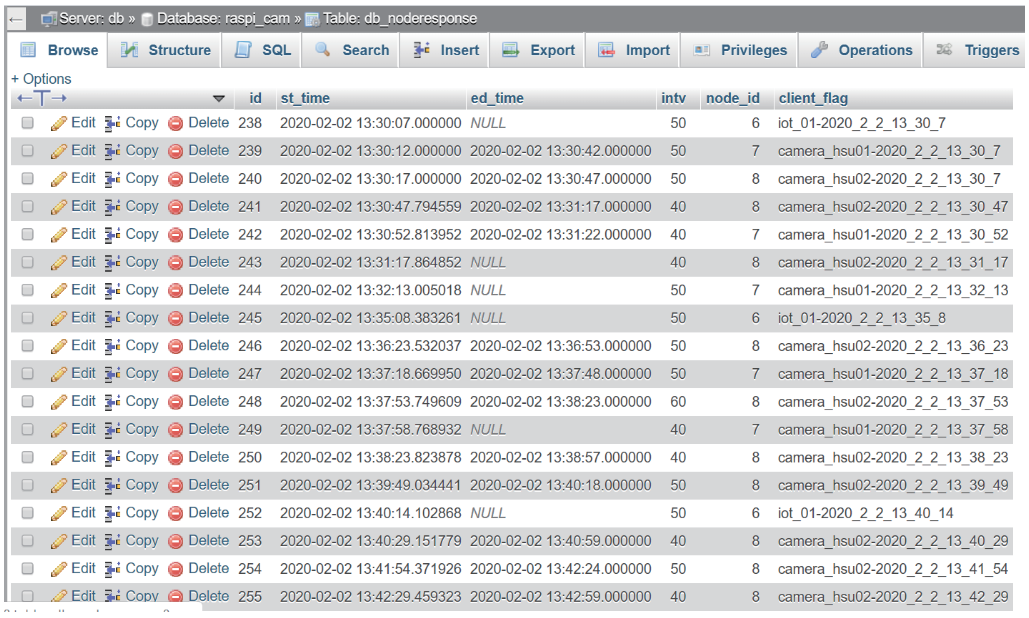
Event	1	2	3	4	5	6	7	8	9	10
Execution time	60	120	180	240	180	240	180	120	180	240
Package loss	—	—	180	180	—	—	—	—	—	—
Total time	60	180	360	540	720	840	1020	1140	1320	1560
Current temperature	26	27	24	24	26	26	25	25	26	26
Tolerance range	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28

Table 7
Simulation results when using automatic temperature sensor.

Event	1	2	3	4	5	6	7	8	9	10
Execution time	60	120	180	240	180	120	60	0	60	120
Package loss	—	—	180	180	—	—	—	—	—	—
Total time	60	180	360	540	720	840	900	900	960	1080
Current temperature	26	27	24	24	23	24	25	26	26	26
Tolerance range	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28	26–28

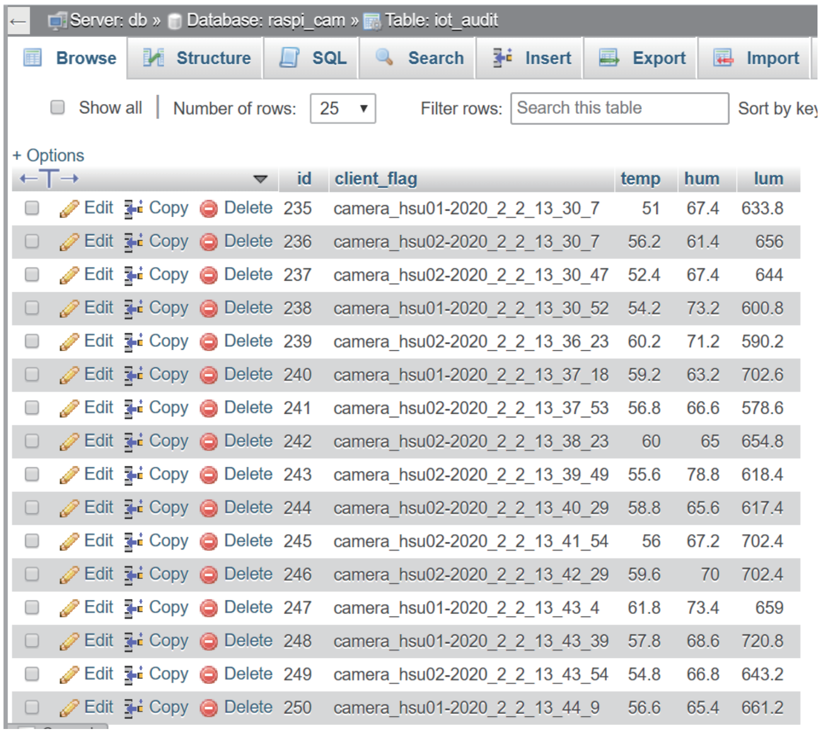
In the past, the system used the average temperature as the basis for event triggering. In this study with the dynamic adjustment of the detection frequency, event triggering should not be limited by the average temperature. Because the detection of events is dynamic, the temperature will also be adjusted after the sprinkler and air conditioner are started. We try to let the temperature sensor automatically start to perform the detection, which can markedly improve the system performance in the experiment. The simulation results are shown in Table 7.

In this simulation, two abnormal packet losses occurred, which caused the detection time to change twice. Since the temperature detection module starts according to the actual situation, the total time in this experiment is reduced from 1140 to 1080. From the experimental data, it is found that the flag detection combined with the delay detection module adopted in this study can reduce the average time of events occurring in the smart agricultural IoT system to the optimal value under the same detection frequency. After the flag setting is introduced, the end devices in the fog environment can act on their own. The actual temperature detected by the sensing device can provide the optimal frequency of event detection. At the same time, when a packet delay occurs, the fog end device can record and continue to operate in accordance with the previous script. This can also avoid the situation of devices stopping detection and processing after their disconnection. Results of the experimental simulation are shown in Figs. 4–6.



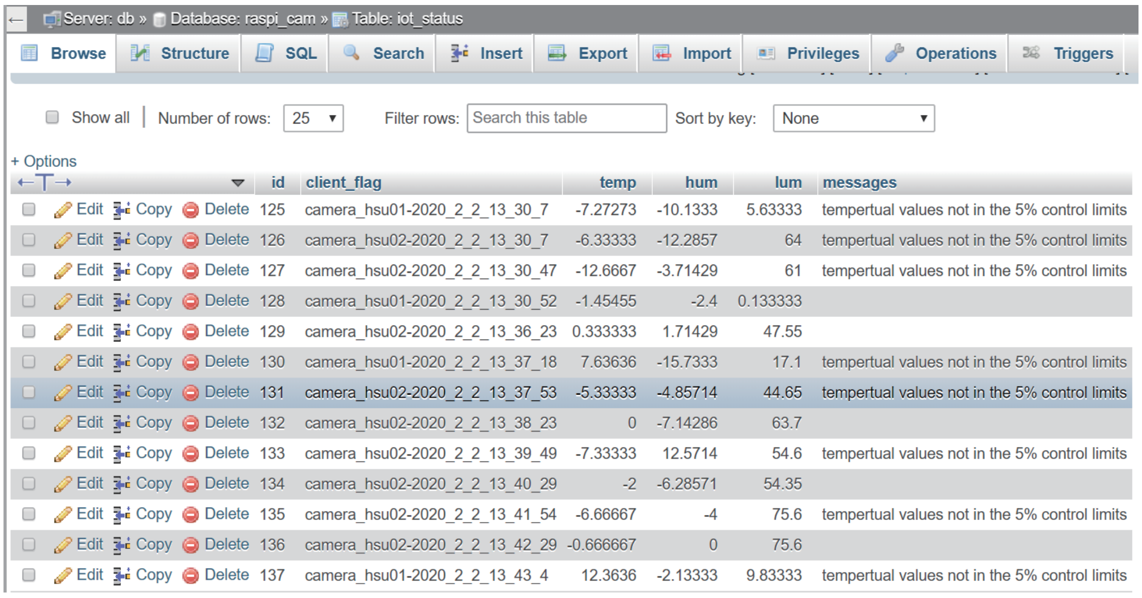
id	st_time	ed_time	intv	node_id	client_flag
238	2020-02-02 13:30:07.000000	*NULL*	50	6	iot_01-2020_2_2_13_30_7
239	2020-02-02 13:30:12.000000	2020-02-02 13:30:42.000000	50	7	camera_hsu01-2020_2_2_13_30_7
240	2020-02-02 13:30:17.000000	2020-02-02 13:30:47.000000	50	8	camera_hsu02-2020_2_2_13_30_7
241	2020-02-02 13:30:47.794559	2020-02-02 13:31:17.000000	40	8	camera_hsu02-2020_2_2_13_30_47
242	2020-02-02 13:30:52.813952	2020-02-02 13:31:22.000000	40	7	camera_hsu01-2020_2_2_13_30_52
243	2020-02-02 13:31:17.864852	*NULL*	40	8	camera_hsu02-2020_2_2_13_31_17
244	2020-02-02 13:32:13.005018	*NULL*	50	7	camera_hsu01-2020_2_2_13_32_13
245	2020-02-02 13:35:08.383261	*NULL*	50	6	iot_01-2020_2_2_13_35_8
246	2020-02-02 13:36:23.532037	2020-02-02 13:36:53.000000	50	8	camera_hsu02-2020_2_2_13_36_23
247	2020-02-02 13:37:18.669950	2020-02-02 13:37:48.000000	50	7	camera_hsu01-2020_2_2_13_37_18
248	2020-02-02 13:37:53.749609	2020-02-02 13:38:23.000000	60	8	camera_hsu02-2020_2_2_13_37_53
249	2020-02-02 13:37:58.768932	*NULL*	40	7	camera_hsu01-2020_2_2_13_37_58
250	2020-02-02 13:38:23.823878	2020-02-02 13:38:57.000000	40	8	camera_hsu02-2020_2_2_13_38_23
251	2020-02-02 13:39:49.034441	2020-02-02 13:40:18.000000	50	8	camera_hsu02-2020_2_2_13_39_49
252	2020-02-02 13:40:14.102868	*NULL*	50	6	iot_01-2020_2_2_13_40_14
253	2020-02-02 13:40:29.151779	2020-02-02 13:40:59.000000	40	8	camera_hsu02-2020_2_2_13_40_29
254	2020-02-02 13:41:54.371926	2020-02-02 13:42:24.000000	50	8	camera_hsu02-2020_2_2_13_41_54
255	2020-02-02 13:42:29.459323	2020-02-02 13:42:59.000000	40	8	camera_hsu02-2020_2_2_13_42_29

Fig. 4. (Color online) Results of the autonomous detection of data through three nodes in the experiment.



id	client_flag	temp	hum	lum
235	camera_hsu01-2020_2_2_13_30_7	51	67.4	633.8
236	camera_hsu02-2020_2_2_13_30_7	56.2	61.4	656
237	camera_hsu02-2020_2_2_13_30_47	52.4	67.4	644
238	camera_hsu01-2020_2_2_13_30_52	54.2	73.2	600.8
239	camera_hsu02-2020_2_2_13_36_23	60.2	71.2	590.2
240	camera_hsu01-2020_2_2_13_37_18	59.2	63.2	702.6
241	camera_hsu02-2020_2_2_13_37_53	56.8	66.6	578.6
242	camera_hsu02-2020_2_2_13_38_23	60	65	654.8
243	camera_hsu02-2020_2_2_13_39_49	55.6	78.8	618.4
244	camera_hsu02-2020_2_2_13_40_29	58.8	65.6	617.4
245	camera_hsu02-2020_2_2_13_41_54	56	67.2	702.4
246	camera_hsu02-2020_2_2_13_42_29	59.6	70	702.4
247	camera_hsu01-2020_2_2_13_43_4	61.8	73.4	659
248	camera_hsu01-2020_2_2_13_43_39	57.8	68.6	720.8
249	camera_hsu02-2020_2_2_13_43_54	54.8	66.8	643.2
250	camera_hsu01-2020_2_2_13_44_9	56.6	65.4	661.2

Fig. 5. (Color online) Temperature, humidity, and light brightness results returned by fog IoT device.



Server: db » Database: raspi_cam » Table: iot_status

Number of rows: 25 Filter rows: Search this table Sort by key: None

	id	client_flag	temp	hum	lum	messages
Edit Copy Delete	125	camera_hsu01-2020_2_2_13_30_7	-7.27273	-10.1333	5.63333	tempertual values not in the 5% control limits
Edit Copy Delete	126	camera_hsu02-2020_2_2_13_30_7	-6.33333	-12.2857	64	tempertual values not in the 5% control limits
Edit Copy Delete	127	camera_hsu02-2020_2_2_13_30_47	-12.6667	-3.71429	61	tempertual values not in the 5% control limits
Edit Copy Delete	128	camera_hsu01-2020_2_2_13_30_52	-1.45455	-2.4	0.133333	
Edit Copy Delete	129	camera_hsu02-2020_2_2_13_36_23	0.333333	1.71429	47.55	
Edit Copy Delete	130	camera_hsu01-2020_2_2_13_37_18	7.63636	-15.7333	17.1	tempertual values not in the 5% control limits
Edit Copy Delete	131	camera_hsu02-2020_2_2_13_37_53	-5.33333	-4.85714	44.65	tempertual values not in the 5% control limits
Edit Copy Delete	132	camera_hsu02-2020_2_2_13_38_23	0	-7.14286	63.7	
Edit Copy Delete	133	camera_hsu02-2020_2_2_13_39_49	-7.33333	12.5714	54.6	tempertual values not in the 5% control limits
Edit Copy Delete	134	camera_hsu02-2020_2_2_13_40_29	-2	-6.28571	54.35	
Edit Copy Delete	135	camera_hsu02-2020_2_2_13_41_54	-6.66667	-4	75.6	tempertual values not in the 5% control limits
Edit Copy Delete	136	camera_hsu02-2020_2_2_13_42_29	-0.666667	0	75.6	
Edit Copy Delete	137	camera_hsu01-2020_2_2_13_43_4	12.3636	-2.13333	9.83333	tempertual values not in the 5% control limits

Fig. 6. (Color online) Flag matches are automatically included in note information.

5. Conclusions and Future Work

In this study, the problem in the previous study(14) that the time axes of the sensing device and the environmental control device are inconsistent is corrected by a delayed feedback method when the IoT device execution time is longer than that of the previous study. Through the independent operation of different fog devices, the simulation system of the proposed model constructs flags corresponding to the results of execution events. The flags are used to compare and verify the results on the core server of the innovation service platform. The innovative service platform provides users with simple parameter-setting methods. The M2M standard protocol is used to send data and service codes to all end nodes.

According to our experimental results, the labeling method can assist fog devices to continue operations after network disconnection. When a device is offline, the difference between the calculation time and the actual value will not be too large, and the operations for events can be gradually corrected. After the device is reconnected to the network, the execution is resumed for event services, and the effect of the time difference between the performance of operations of the event will be minimized. By using the proposed method, each node can continue to operate independently and shorten the waiting response time when an abnormal situation occurs in the IoT communication system. The results of an experimental simulation in a real IoT environment showed a reduction in the computing time of each node's independent operation.

Acknowledgments

The works are supported by the Longyan University’s Qi Mai Science and Technology Innovation Fund Project of Longyan City (2017QM0201 and 2018LYQM0202), Longyan University’s Qi Mai Science and Technology Innovation Fund Project of Liancheng ([2018]132) and Shanghang (2019SHQM05) County, Longyan and Longyan University’s Research and Development Team Fund ((2018)8), Great Project of Production, Teaching, Research of Fujian Provincial Science and Technology Department (2019H6023).

References

1 M. A. Feki, F. Kawsar, M. Boussard, and L. Trappeniers: Computers **46** (2013) 24. https://doi.org/10.1109/MC.2013.63
2 W. Lumpkins: IEEE Consum. Electron. Mag. **2** (2013) 47. https://doi.org/10.1109/MCE.2013.2240615
3 M. Amadeo, C. Campolo, J. Quevedo, D. Corujo, A. Molinaro, A. Iera,, L. Aguiar, and V. Vasilakos: IEEE Network **30** (2016) 92. https://doi.org/10.1109/MNET.2016.7437030
4 Z. Meng, Z. Wu, C. Muvianto, and J. Gray: IEEE Internet Things **4** (2017) 236. https://doi.org/10.1109/JIOT.2016.2646375
5 J. N. Al-Karaki, K. C. Chen, G. Morabito, and J. De Oliveira: Ad Hoc Networks **18** (2014) 1. https://doi.org/10.1016/j.adhoc.2014.03.006
6 R. Schniederman: IEEE Signal Process. Mag. **4** (2013) 10. https://doi.org/10.1109/MSP.2013.2255940
7 A, Zanella, N. Bui, A. Castellani, L. Vangelista, and M Zorzi: IEEE Internet Things **1** (2014) 22. https://doi.org/10.1109/JIOT.2014.2306328
8 J. Latvakoski, M. Alaya, H. Ganem, B. Jubeh, B. A. Iivari, J. Leguay, J. Bosch, and N. Granqvist: Future Internet **6** (2014) 261. https://doi.org/10.3390/fi6020261
9 F. Tao, Y. Cheng, L. Xu, L. Zhang, and B. Li: IEEE Trans. Ind. Inf. **10** (2014) 1435. https://doi.org/10.1109/TII.2014.2306383
10 M. Glaab, W. Fuhrmann, J. Wietzke, and B. Ghita: IEEE Commun. Mag. **53** (2015) 42. https://doi.org/10.1109/MCOM.2015.7355583
11 S. Xu, C. H. Chen, and T.C. Chang: IEEE Internet Things **6** (2019) 9464. https://doi.org/10.1109/JIOT.2019.2929118
12 G. Wu, S. Talwar, K. Johnsson, N. Himayat, and K. D. Johnson: IEEE Commun. Mag. **49** (2011) 36. https://doi.org/10.1109/MCOM.2011.5741144
13 D. Guinard: A Web of Things Application Architecture—Integrating the Real-World into the Web, Ph.D. Dissertation (2011). https://webofthings.org/dom/thesis.pdf
14 T. C. Hsu, Y. H. Tsai, D. M. Chang, F. Y. Liu, and C. H. Chang: Sens. Mater. **31** (2019) 1815. https://doi.org/10.18494/SAM.2019.2254

About the Authors

Dong-Meau Chang received his B.S. degree in chemical engineering from Tatung University, Taiwan, in 1988 and his M.S. and Ph.D. degrees from National Taiwan University and Tatung University, Taiwan, in 1990 and 1996, respectively. Since 2018, he has been a professor at Longyan University. His research interests are in IoT and big data analysis. (morganch@me.com)

Yao-Hong Tsai received his M.S. and Ph.D. degrees in information management from the National Taiwan University of Science and Technology (NTUST), Taipei, Taiwan, R.O.C., in 1994 and 1998, respectively. He was a researcher at the Advanced Technology Center, Information and Communications Research Laboratories, Industrial Technology Research Institute (ITRI), Hsinchu. He is currently an associate professor at the Department of Information Management, Hsuan Chuang University, Hsinchu. His current research interests include image processing, pattern recognition, personal communication systems, and cloud computing. (yaohong.tsai@gmail.com)

Tse-Chuan Hsu received his M.S. degree in computer engineering from Tunghai University, Taichung, Taiwan. He is pursuing his Ph.D. degree at the Bath Spa University Centre for Creative Computing and he is also a lecturer at the Department of Information Management, Hsuan Chuang University, Hsinchu City, Taiwan. His research interests include the Internet of Things, software engineering, and cloud computing data analysis systems. (davidhsu@hcu.edu.tw)

Hsien-Wei Tseng received his B.S. degree in electrical engineering from National Kaohsiung University of Applied Sciences in 2002 and his Ph.D. degree in electrical engineering from Tamkang University in 2010. He joined the School of Information Engineering, Longyan University, Fujian, China, in Aug. 2017, where he now serves as a professor. His main research interests include system design and performance evaluation in wireless communication systems. https://sites.google.com/view/hsienwei-tseng (hsienwei.tseng@gmail.com)