Aggregating Small Packets in M2M Networks: An OM2M Implementation

Sheng-Chieh Lee†, *Kun-Ru Wu†, Ching-Kuo Hsu†, Po-Yen Chang†, Jia-Ming Liang‡, Jen-Jee Chen§, Yu-Chee Tseng†
†Department of Computer Science, National Chiao Tung University, Taiwan
‡Department of Computer Science and Information Engineering, Chang Gung University, Taiwan
§Department of Electrical Engineering, National University of Tainan, Taiwan
*Email: kunruwu@cs.nctu.edu.tw

***Abstract*---IoT (Internet of Things) and M2M (machine to machine) have attracted a lot of attention since more and more devices are expected to connect to the Internet for special purposes such as environment monitoring, home automation, industrial surveillance, and e-Health care. Currently, the *OM2M (Open source platform for M2M communication)* is a promising project which implements oneM2M and SmartM2M standards as an open-source platform for integrating various M2M services, applications, and devices. However, the individual data generated from those IoT/M2M devices is usually quite small, which incurs a lot of control overhead and thus decreases network performance significantly. Therefore, in this work we design and implement an OM2M 'plugin' that can aggregate small IoT data effectively. We will show how the plugin works and verify the effectiveness on the network bandwidth in this demonstration.**

***Index Terms*---Data aggregation, Internet of Things (IoT), machine-to-machine communication (M2M), open source platform for M2M communication (OM2M), oneM2M, smartM2M.**

I. Introduction

The emergence of *Internet of Things (IoT)* is promising issues in present. It can provide amount of applications and services with various types of devices. In order to connect these IoT devices, the *machine-to-machine (M2M)* communication is developed. The *OM2M (Open source platform for M2M communication)* [1] is an open source M2M project which implements oneM2M [2] and SmartM2M [3] standards for integrating heterogeneous devices. When the number of connected devices arises, the gateway has to collect amount of sensor data. In order to provide reliable transmission, TCP is widely used. However, during TCP connection establishment process, e.g., three-way handshake, the device has to exchange control messages for each transmission. The massive small data and control messages occupy the network bandwidth, and decrease network performance. This motivates us to study the overhead reduction for the network with massive small data and develop a data aggregation mechanism based on OM2M.

In an OM2M network, it consists of devices, gateways and a network domain, as Fig. 1. The *Service Capability Layer (SCL)* is used to provide with several services for M2M interaction. There are many kinds of applications on SCL, such as Device (as DSCL), Gateway (as GSCL) and Network (as NSCL) applications. The domains can communicate with each other by SCL which resides in the DSCL, GSCL and NSCL. In our system, the devices are used for monitoring environment, the gateway is responsible for collecting sensor data from devices, and the server in the network domain will pull data from the gateway. Based on the OM2M architecture, we develop a plugin, 'Data Aggregation', to merge multiple small data into one packet in the gateway domain. It can aggregate data based on the schedule, thus save bandwidth by reducing headers and the number of control messages.

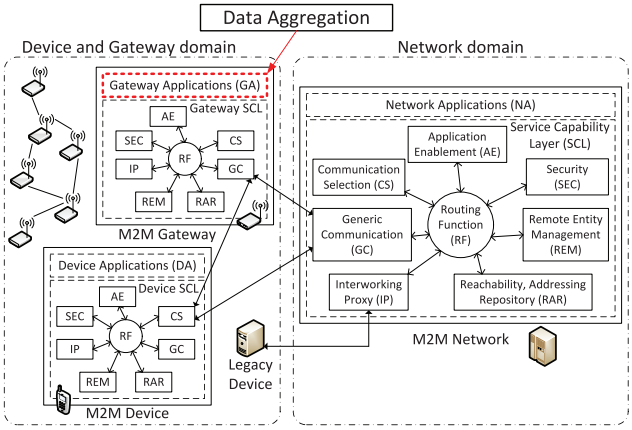

Fig. 1. OM2M Architecture.

In this paper, we consider a M2M network with N devices, M gateways and one network server, where each device D_i, $i = 1..N$ can communicate via gateway G_j, $j = 1..M$ through Wi-Fi, Ethernet or LTE, etc. In addition, each device D_i has the *periodic traffic* with admitted data rate R_i (bytes/sec). For QoS guarantee, all of the period traffic has its transmission delay constraint L_i (ms) to ensure the transmission instancy between the devices to the server. When transmitting traffic via OM2M platform, the header is the major portion of packets. In other words, the header size is larger than payload size. Our problem is to ask how to aggregate the OM2M packet for each device D_i, $i = 1..N$ such that the QoS of devices can be guaranteed (including the traffic delay L_i, data rate R_i, and request size Q_i) while the total number of devices can be served effectively by data aggregation.

II. Prototype Evaluation

In our prototype, we use a laptop as the M2M server and deploy several gateways and M2M devices by Raspberry PI 2 (Model B), as Fig. 3(a). The OM2M with version v0.8.0 is installed in the gateway and server to accomplish 'small data aggregation'. The devices equip with multiple sensor such as temperature and humidity sensors to take charge of monitoring the environment. When the number of connected device increases, the massive connections and small data are generated. In our system, the devices transmit sensing data to both gateway and cloud server based on RESTful API.

2155-6814/16 $31.00 © 2016 IEEE
DOI 10.1109/MASS.2016.31

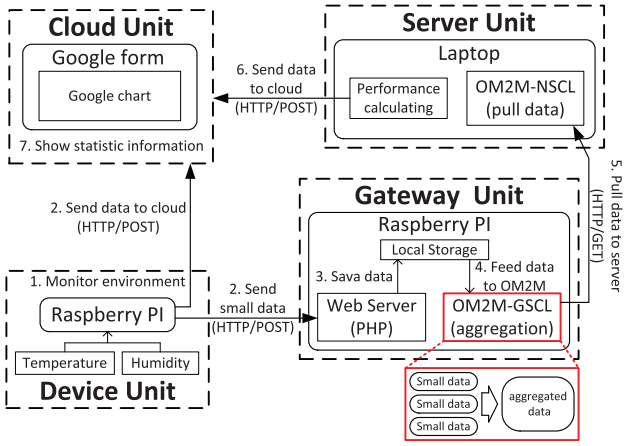

Fig. 2. Flowchart of our prototype.

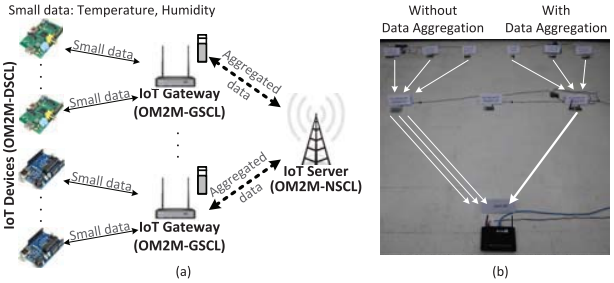

Fig. 3. Demonstration scenario.

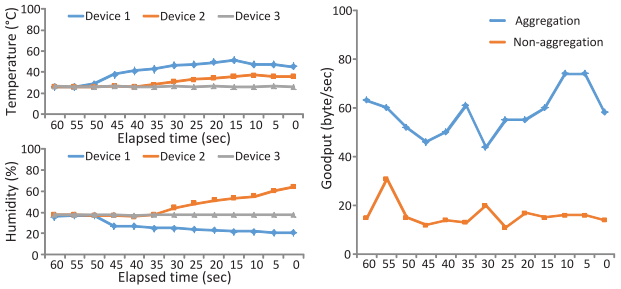

Fig. 4. Performance evaluation of our system.

The cloud server is responsible for recording the values of temperature and humidity. Whenever the gateway receives sensing data, it puts the data in a local database. Our 'Data Aggregation' plugin reads the data from the database automatically, and uses aggregation technique [4] to store the data in contentInstance on OM2M GSCL. The server pulls data from gateway with RESTful API according to its schedule, e.g., a customized periodic or a specific time. While pulling data, we also calculate the consumed bandwidth and goodput. These performance metrics are also transmitted to the cloud server for recording. Therefore, we can observe the environmental sensing data and network performance on the cloud server.

Fig. 3(a) is our demo scenario, there are six M2M devices (Raspberry PI with temperature and humidity sensors), two M2M gateways (Raspberry PI with OM2M) and a M2M server (Laptop with OM2M) in the prototype. The devices sense temperature and humidity in the environment. The sensing data is transmitted to the gateway via Wi-Fi. Our aggregation mechanism is a plugin in OM2M. When gateway executes aggregation, it calls 'Activator.java, Controller.java, Monitor.java and Mapper.java' to process data, and stores data in DATA/contentInstances under SCL resource tree. The server pulls data from gateway based on its schedule. The developed prototype of our system is shown in Fig. 3(b). In order to observe the performance of aggregation, we also place a non-aggregation gateway in the prototype to compare. The network traffic without data aggregation is plotted in orange lines. On the other hand, the blue lines are the network traffic with data aggregation.

Fig. 4 shows the experiment results. In order to vary the temperature and humidity, we use a hair dryer to blow hot air to Device 1 and place a cup of hot water to generate hot vapor under Device 2. Fig. 4 (left part) shows that the temperature increases and humidity decreases on Device 1. For Device 2, both the temperature and humidity increase because of hot water. The sensing data is transmitted to the gateway and then the server can pull the sensing data by using aggregation mechanism or not. Fig. 4 (right part) shows the goodput in the network. We can see that the data aggregation mechanism has higher goodput due to conserving sending the OM2M headers and reducing control messages. On the other hand, the system without aggregation generates large amounts of small data which consumes more bandwidth and has more control messages for TCP connections. In our aggregation approach, it can combine multiple small data into one packet. Therefore, we can send data efficiently by reducing the amount of headers and control messages to improve network efficiency.

To conclude, in this paper we develop a 'Data Aggregation' plugin to deal with massive small data on OM2M platform. It can efficiently aggregate data based on the schedule and thus save bandwidth by reducing the number of headers and control messages. We also implement it on OM2M platform to validate its performance. The experiment results show that when the network has massive connections, our mechanism can not only send data efficiently but also reduce the control messages and thus save bandwidth significantly.

ACKNOWLEDGMENTS

This research is co-sponsored by MOST 102-2218-E-182-008-MY3, 105-2221-E-182-051, 104-2221-E-024-005, 104-2221-E-009-113-MY3, 104-2218-E-009-009, 104-3115-E-009-002, 104-2745-8-182-001, CGU NERPD2C0763, EERPD2F0021, MoE ATU Plan, Delta Electronics, ITRI, and Institute for Information Industry. This work was also supported by NTU and Intel under Grants MOST 103-2911-I-002-001, NTUICRP-104R7501, and NTU-ICRP-104R7501-1.

REFERENCES

[1] M. Ben Alaya, Y. Banouar, T. Monteil, C. Chassot and K. Drira, ''OM2M: Extensible ETSI-compliant M2M service platform with self-regulation capability,'' *Procedia Computer Science*, vol. 32, pp. 1079--1086, 2014.
[2] ''oneM2M: Standards for M2M and the Internet of Things.'' http://www.onem2m.org/.
[3] ''SmartM2M - ETSI Portal,'' https://portal.etsi.org/tb.aspx?tbid=726&SubTB=726.
[4] P.-Y. Chang, J.-M. Liang, J.-J. Chen, K.-R. Wu, Y.-C. Tseng, Z. Ren, and M. Wu, ''Massive connectivity and small data issues for M2M/IoT communications,'' *IEEE Vehicular Technology Society Asia Pacific Wireless Communications Symposium (APWCS)*, pp. 1--5, 2016.