Bridging UPnP and ZigBee with CoAP

Protocol and its Performance Evaluation

Jin Mitsugi
Auto-ID Laboratory Japan
Keio University
5322 Endo Fujisawa 252-0882, Japan
mitsugi@sfc.wide.ad.jp

Shigeru Yonemura
Auto-ID Laboratory Japan
Keio University
5322 Endo Fujisawa 252-0882, Japan
raccoony@sfc.wide.ad.jp

Hisakazu Hada
Auto-ID Laboratory Japan
Keio University
5322 Endo Fujisawa 252-0882, Japan
hada@sfc.wide.ad.jp

Tatsuya Inaba
Auto-ID Laboratory Japan
Kanagawa Institute of Technology
1030 Shimo-ogino Atsugi 243-0292, Japan
tinaba@ic.kanagawa-it.ac.jp

ABSTRACT

Incorporation of heterogeneous wireless sensor and actuator networks (WS&AN) is an essential challenge of web based Internet of Things (IoT) architectures. We propose to use UPnP and end-to-end HTTP communication using CoAP to bridge WS&AN and IoT system. UPnP enables automatic discovery of sensor devices which directly connect to a WS&AN via a gateway. Instead of translating WS&AN and UPnP protocols at the gateway, we propose to use CoAP in WS&AN. This provides flexible communications between sensor devices and applications. Drawback of this end-to-end Web based IoT information system is vulnerability to excessive traffics from sensor devices or from the applications because there is no authority to monitor and control traffics in the architecture. We examined the performance of our implementations to find that the transmit performance of a single sensor device could be limited by the serial communications of embedded transceiver. Excessive data requests from applications might also result in the packet loss and wasteful WS&AN congestion. If the traffic is confined within the performance limits, the implemented UPnP and ZigBee bridging using CoAP shows satisfactory performance. We can subscribe up to 16 sensor devices data with 500 msec using simple HTTP POST requests.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: General

ACM CoNEXT 2011 IoTSP, December 6, 2011, Tokyo, Japan.

General Terms

Design

Keywords

UPnP, ZigBee, HTTP

1. INTRODUCTION

Internet of Things (IoT) enables us to monitor and process physical objects and real-world incidents in information system. To bridge physical objects and information system, wireless sensors and actuator network (WS&AN) is essential. Existing WS&AN standards define a set of protocol and data structure. A typical WS&AN protocol, ZigBee[1] defines networking and application profiles on top of IEEE 802.15.4 PHY and MAC standard[2]. Since a WS&AN protocol terminates its protocol at a sensor sink, the incorporation of a WS&AN into an Web based IoT architectures, such as SENSEI [3], Sensor Web Enablement (SWE) [4] and IoT-A[5], requires additional mechanism to discover and control of WS&AN devices from the IP network. We also need to consider that a WS&AN usually comprises a constrained network and constrained devices —- the network entails long latency and a device has the minimum computational power and storage.

UPnP (Universal Plug and Play) [6] provides a mechanism to interconnect intelligent appliances, sensors and PCs of all form factors. UPnP uses HTTP over Ethernet to discover and control devices. A device is registered to a control point as a UPnP device with a unique identifier (ID) called UUID. We can include a WS&AN, such as ZigBee, to a UPnP network via a gateway. There have been a number of researches to transparently connect UPnP and ZigBee by bi-directionaly translating the UPnP profile and ZigBee profile at a

gateway[7, 8, 9]. Combination of UPnP with a constrained network provides a significant benefit in device discovery such that the control point in UPnP automatically collects devices' information. In addition, authors previously reported[10, 11] a small extension of UPnP to include EPC (Electric Product Code), a globally unique identifier system in a form of URN, we can further discover services related to the device by using established discovery service, ONS (Object Naming Service)[12].

We have two design choices when we incorporate a WS&AN in an IoT architecture which extensively uses HTTP and Web technology, such as SENSEI, SWE and IoT-A. One approach is to use a gateway as a protocol translator as mentioned earlier. When we have a new service or new device or an update of application protocol, however, we need to update the gateway to incorporate the change. Another approach is to establish an end-to-end HTTP connection between applications and WS&AN devices. This significantly simplifies the requirements toward the gateway. Since a direct application of HTTP over a WS&AN, which is usually a constrained network and involves constrained devices, should be impractical, we use CoAP (Constrained Application Protocol)[13] being developing in IETF in our system.

CoAP provides a mechanism to terminate the HTTP communications from application and converted to a compact form of HTTP for a constrained network. HTTP header information such as method and content type are represented by a binary data. CoAP originally presumes an end to end IP connection, typically UDP over 6LoWPAN, for its networking layer. Although implementations of CoAP over an IP network are available for selected platform such as linux, Contiki and Tiny OS, it is usually straightforward and practical to directly use CoAP on top of ZigBee. This way, a sensor device only needs to process CoAP as an application sublayer data (APS). Otherwise, we need IP processing function in the sensor device. In addition, since most of the off-the-shelf ZigBee coordinator are designed to connect to a PC with serial interface, we may also need to establish a PPP link over the serial interface. The performance of CoAP is reported in [14], only in the cases where CoAP is used in a linux PC rather than constrained devices. Incorporation with IoT architecture is also not covered.

Figure 1 shows how applications are connected to WS&AN device through UPnP in our system. Application, in this context, could be considered as an end-user application or a sensor service such as sensor observation service (SOS) in SWE. As shown in the figure, bridging UPnP and ZigBee with CoAP enables flexible communications between applications and sensor devices. One application can subscribe many sensor devices by sending identical HTTP messages to each device whether or not the device is connected to a constrained network. CoAP communications in a constrained network is invisible from subscribing applications. This, on the other hand, may result in overload of a sensor device or congestion in ZigBee network because there is no authority to control the traffic toward ZigBee devices. Many applications subscribe to a single device as shown in Fig.1. Evaluation of permissible subscribing and polling loadings to a single device and the whole network are essential to prevent end-device collapse and constrained network congestion.

In this paper, we introduce a protocol to bridge UPnP and ZigBee by CoAP and report the outcome of performance evaluation with up to 20 sensor devices. Section 2 introduces the protocol featuring the mechanism to bridge UPnP and ZigBee, which our previous publications[10, 11] did not cover. Bridging UPnP and ZigBee requires new functions to handle SSDP (Simple Service Discover Protocol) of UPnP in CoAP. In Section 3, throughput performances of a single constrained device and of 20 devices are examined to clarify the performance limit of a single device and the whole system.

2. COAP EXTENSION AND IMPLEMENTATION

2.1 CoAP extension to bridge UPnP and ZigBee

In this paper, a sensor device is supposed to be composed of a sensor MCU and a ZigBee transceiver and associated sensors and actuators. The sensor MCU is connected to the ZigBee transceiver usually with a serial interface such as UART or I^2C. The MCU handles CoAP processing in addition to the sensors and actuators managements. There is a single chip transceiver chip in the market that can handle the role of MCU too. We treat such a single chip solution as a sensor device.

Figure 2 shows a sensor device association and UPnP Join procedure. Underlined commands and responses are of UPnP, which resides in APS (Application sublayer) of ZigBee. APS payload is described using CoAP as shown in Fig. 3. Upon the completion of ZigBee network association, the sensor MCU send a Join to the gateway. A CoAP packet from a sensor device to the coordinator is extracted from ZigBee APS by the coordinator and is transferred to gateway. In addition to the CoAP defined Code (GET, POST, PUT, DELETE), we extend CoAP Code to cover UPnP commands and response as in Table 1. This way, the gateway only needs to check "Code" field, which is a fixed length from the beginning of a CoAP packet, to create a virtual UPnP device for the ZigBee device. The gateway, then, forwards the CoAP packet to the control point after converting it to HTTP. A traffic CoAP packet, involves Code less than 5, is transferred to re-

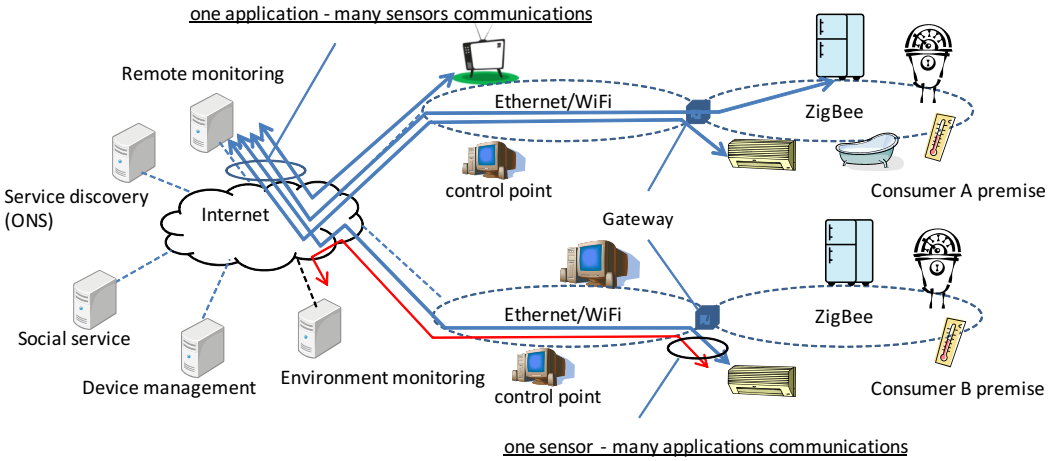

Figure 1: Our web based Internet of Things information system where we can monitor and control sensor devices by sending HTTP request. Since the publishing control is fundamentally done in end-to-end manner, a gateway connecting a constrained network only needs to transfer commands and responses. This facilitate us to add new devices and new services.

```
POST http://gw00.home00.racow.net/urn:epc:id:sgtin:457122707.0100.1
<?xml version=""1.0"" encoding=""UTF-8""?>
<message>
<command>poll</command>
<requestID>12345</requestID>
<responseURI>http://saveenergy.navi.ranking.racow.net/response.php</responseURI>
<query target="sink"><![CDATA[{"set":{"lighton":1}}]]></query>
</message>
```

Figure 5: A message from an application comprises two blocks. A block is for the gateway, another is to the destination ZigBee device.

Table 1: Extended CoAP Codes to incorporate UPnP controls. Leave is issued by a sensor device when it leaves UPnP. Alive, Ping and CollectAll are for management of UPnP.

Method	Code	Note
GET	1	native coap code
POST	2	native coap code
PUT	3	native coap code
DELETE	4	native coap code
Join	11	our extension
Leave	12	our extension
Alive	13	our extension
Ping	14	our extension
CollectAll	15	our extension

questing applications after converting the header to that of normal HTTP. The gateway creates a virtual UPnP device upon receiving the Join request and notify a control point according to UPnP protocol. Figure 4 shows the sequence for an application to subscribe or to request a sensor data. The binding mechanism of a report and its destination address is similar to the subscription controls in EPCIS query interface [15] and in UPnP[1]. An example subscription message from an application is shown in Fig.5, which specifies request ID and response URI. The request ID is bound to a transaction ID in CoAP and recorded in a database by the gateway. When the gateway receives a report from a sensor device, the gateway retrieves the corresponding request ID and response URI from its database and send the report to subscribing application.

2.2 Implementation

[1]In UPnP term, subscription and eventing

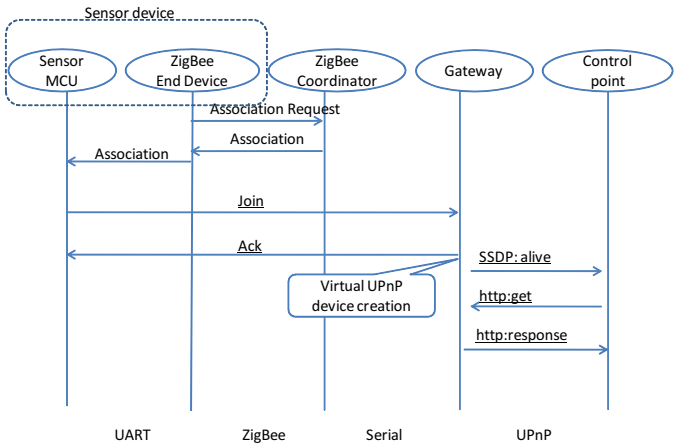

Figure 2: After ZigBee network association, a sensor device send a UPnP Join message to gateway through the coordinator. The message is encoded in a CoAP format. The gateway can immediately filter UPnP related packets from traffic packets by checking CoAP Code.

0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
Ver	T	OC		Code				Transaction ID							
Options															
Payload															

ver: Coap version. must be 1.
T: Transaction type. 0=Non-confirmable, 1=Acknowledgement, 3=reset
OC: Option Count
Code: HTTP method represented by 8-bit integer
Transaction ID: A unique ID to distinguish packet.

Figure 3: Coap header involves fixed length header. UPnP packets can be selected just by inspecting the fixed length 32-bit header

Figure 6 shows the hardware implementation of our sensor device. We use connectport X4 as the ZigBee coordinator with a custom software to use CoAP. The gateway and control point functions of UPnP with the CoAP extension are implemented in a board computer Alix 2D13 (Fig.7) with Java. The gateway also has HTTP server (lighttpd) to process subscription controls. We implemented 49 sensor devices part of which are used as environmental sensors and the rests are to monitor and control consumer electronics (Fig.8) through IPv6 network[10].

3. PERFORMANCE EVALUATIONS

As we state in Section 1, bridging UPnP and ZigBee with CoAP may result in a sensor device collapse or a constrained network congestion. We evaluated our system performance subjected to subscription and polling

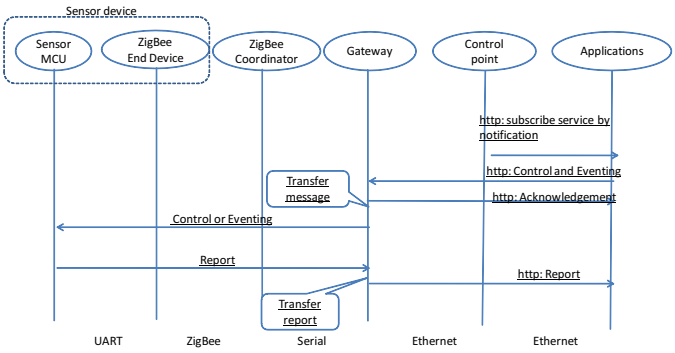

Figure 4: Sensor data subscription and control sequence

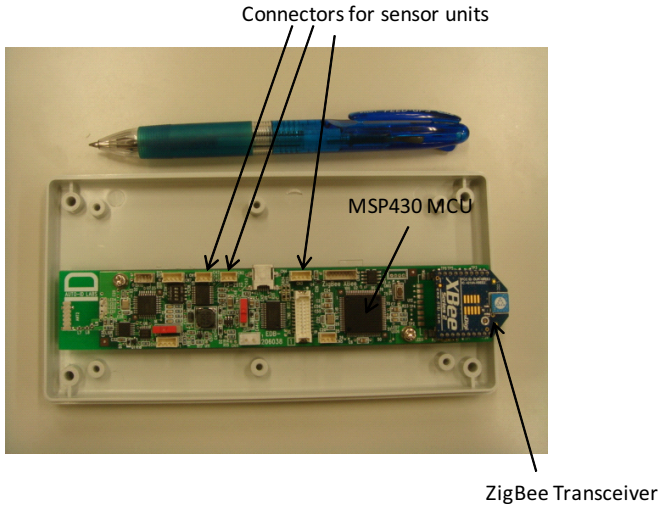

Figure 6: A hardware implementation of sensor device. The MSP430 acquires data from and controls sensor units and then transfers to ZigBee transceiver.

traffics. In this paper, the subscription traffic originates from an application to place a request to the destination sensor device. After the registration, the traffic is generated by the sensor device whenever it experiences a predefined time interval or an event (Fig.9). The polling traffic also originates from an application by requesting or sending command data to remote device. Upon receiving a polling traffic, a sensor device returns data or an acknowledgement by request. Subscription traffic may overload a sensor transmission performance or results in constrained network congestion while polling traffic may overload the gateway or CoAP processing of sensor MCU.

3.1 Performance under Subscription traffic

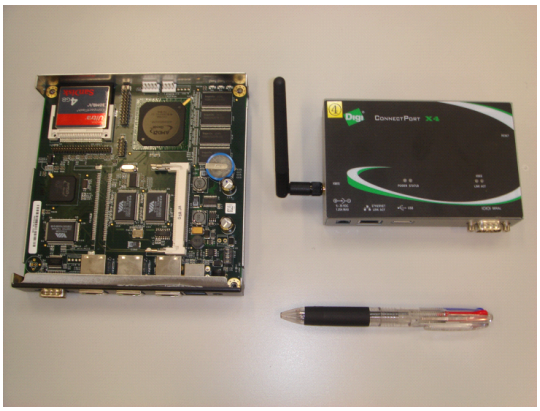

Figure 7: We use connectport X4 as ZigBee coordinator (right). Gateway and control point of UPnP with CoAP extension are implemented in Alix 2D13 with CentOS 5.0 (left).

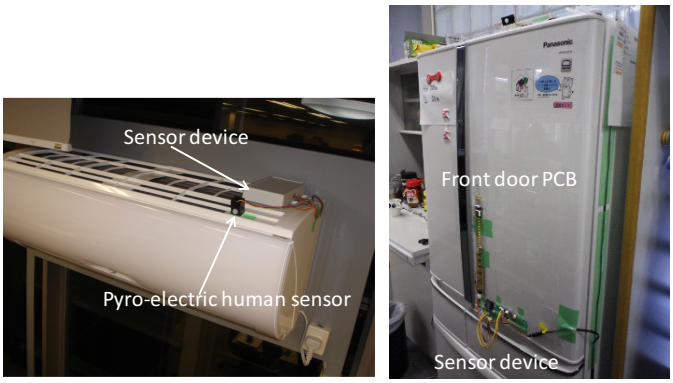

Figure 8: Sensor devices monitor and control commercial consumer electronics. Example sensor data in the air conditioner implementation (left), are sensed and designated room temperatures, wind strength and power consumption, while in refrigerator implementation (right), we monitor freezer and refrigerator temperatures and door opening status.

3.2 Single sensor device transmission

We first examined APS round trip time of ZigBee transceiver driven by sensor MCU. We formed two hop network composed of an sensor device, a router and a coordinator, in an environment where no ZigBee device shares the frequency channel but there are interference from background WiFi. The sensor device continuously transmitted fixed length data to the coordinator with APS acknowledgement (APS_ACK). We observed the transmission timings in the air with Daintree Sensor Network Analyzer (SNA).

Figure 10 shows the result. The processing timings in the router and the router to transfer and to acknowledge are approximately 7 msec and 11 msec, respectively. We measured actually transmitted time intervals by changing the transmit interval time from 300 msec to 1000 msec of the sensor device. We sent 100 packets from the sensor device to the coordinator. The measured time intervals in the air are collected using SNA and computed average as in Fig.11. It is shown in Fig.11 that a sensor device cannot transmit consecutive two packets in less than 300 msec. When we forced to send packets with less than the time interval, the packets starts accumulating in the transmission queue inside the transceiver. Close debugging of transmission control firmware in MSP430 revealed that the minimum interval 300 msec matches the whole processing time to send an APS data with XBee through UART with 9600 bps in our implementation. The ideal shortest time duration to send 100 byte data over the UART takes 104 msec (= 100 × 10/9600) including one stop and one start bits. Since the UART communications is virtually flow controlled[2] it is reasonable to take about 300 msec for UART communications. Since we identify the minimum time interval of consecutive two packets is 300 msec, we define the minimum transmission rate from a single end node is 500 msec with a safety margin of 200 msec. It should be noted that this minimum time interval between two consecutive APS packets dominates the APS throughput of a single sensor device. Suppose APS maximum payload is 100 Byte [3], the maximum APS throughput from a single sensor device is 1.6 kbps (100×8/0.5).

We also measured a transit traffic at a ZigBee router. It takes 7 msec to route a packet meaning that the transit packets is transferred at 115 kbps (100 Byte × 8 bit/ 7 msec) much faster than that of the originating data (Fig.12).

3.3 Multiple sensor devices throughput performance

We set the transmitting interval of each sensor device to be 500 msec not to overload the transceiver and then increase the number of transmitting sensor device one by one with 5 minutes interval until we observe network congestion. A sensor device is either a sensor device or a router depending on the route discovery outcome. The subscription of each sensor device were performed by an application by "POST"ing a HTTP message similar to Fig.5. Upon receiving the message from an application, the sensor device reports its sensor

[2]Even though the communication does not explicitly RTS/CTS flow controlled, the UART communication in MCU checks if the transmitting register is cleared or not.

[3]ZigBee MAC payload is 127 Byte, subtracting NWK and AUX layer headers yield about 100 Byte for APS

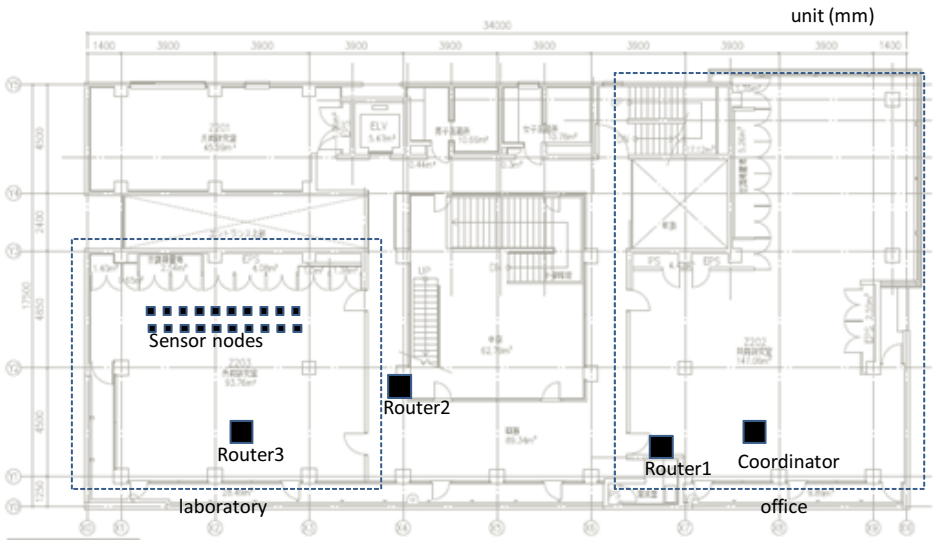

Figure 13: We examined the performance of system in our office. Sensor device were collected from consumer electronics in remote installations and are placed in the laboratory. We have average 3 hops from a sensor device to the coordinator.

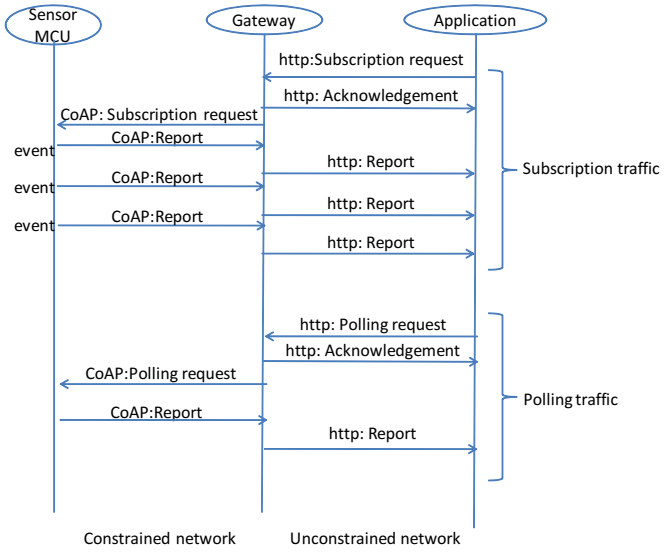

Figure 9: Two types of traffics, polling and subscription traffics, are considered in the performance evaluation.

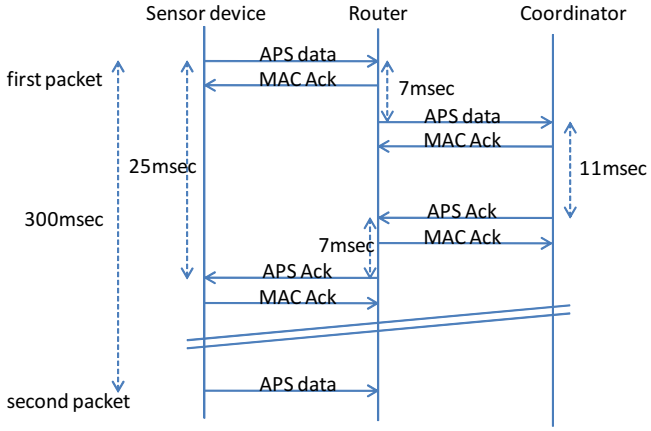

Figure 10: Timing chart of APS_ACK enabled two hops network. Compared to the round trip APS time 25 msec, the consecutive APS data transmission takes much longer time 300msec.

data to the gateway in a form of CoAP. The gateway translates the CoAP message to an HTTP message and send it to the destination URL specified by the application. In the experiment, all the report data is stored in a data base (Postgres) with Apache front end. The stored data is analyzed later to evaluate throughput, packet loss and transmission delay. The experiment was done in our office and laboratory. We located the coordinator in our office and all the sensor devices were in laboratory and 4 routers in the middle to form a multihop network (Fig.13) in a $612\text{m}^2 = 34\text{m} \times 18\text{m}$ working place with steel doors and thick walls. An APS throughput of a sensor device for a specified network loading is computed by counting the number of packet received by Postgres data base within the specified time duration. The aggregated APS throughput is computed by summing all APS throughputs considering the number of hops. We can subscribe up to 20 sensor device

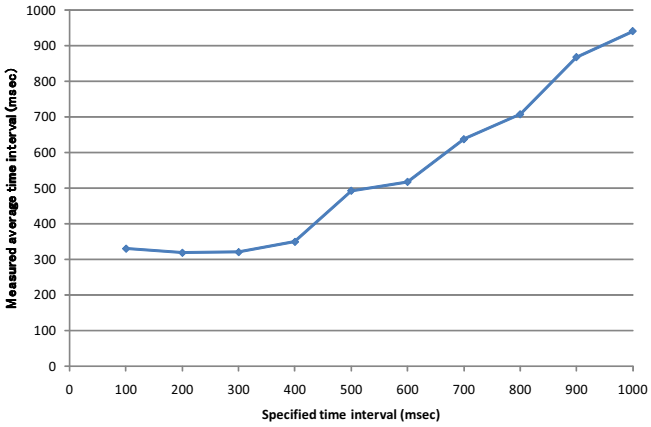

Figure 11: When we send packets with short time intervals, observed transmission rate in the radio saturates at 300 msec. The limit stems from the serial interface of transceiver.

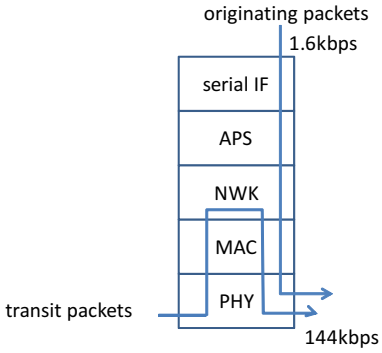

Figure 12: Transit packets pass a router faster than that of the originating packets. Bottleneck of sensor data publishing depends on the workload of the sensor device which originates sensor data

with 55 aggregated hops achieving about 60 kbps aggregated throughput. Up to 14 sensor devices, the aggregated throughput is linear to the number of sensor device, which means there is no traffic congestion in ZigBee network. Since the maximum throughput of a sensor device is 1600 bps, we compute the ideal aggregated through by multiplying 1600 bps and the aggregated number of hops and compared with the measured throughput as shown in Fig.14 It is shown that our implementation is quite close to the ideal throughput up to 16 sensor devices. Since the ideal throughput is dictated by the serial communications between the sensor MCU and transceiver, it is shown that the end-to-end communications are not impeded by UPnP and ZigBee bridge using CoAP.

3.4 Performance under Polling traffic

Polling traffic is generated by a multithread Java program and is transmitted to a sensor device via a gateway. We change the polling time interval from 100 msec to 1000 msec and generated 100 polling requests in each time interval. The performance is measured by counting the number of successfully responded requests, the number of requests rejected by sensor device and the number of request rejected by the gateway. The result is shown in Fig.15. When the polling interval is less than 400 msec, polling requests could be rejected by the gateway. A gateway rejection was counted by observing the response against a polling request. Less than 800 msec time-interval polling requests may be rejected by a sensor device depending on its working load. ZigBee network reveals no problem against this polling traffic because the traffic (1.1 kbps = 100byte × 8 bit/0.7 sec) is far lighter than the maximum aggregated bandwidth (60 kbps).

4. CONCLUSIONS

An Internet of Things architecture needs to incorporate heterogeneous constrained networks and devices to capture physical objects and real-world incidents. End-to-end web information system is a good solution to leverage the recent development of web technology. HTTP, however, may incur excessive traffics to constrained network and device. CoAP being developed in IETF could be a good solution to establish end-to-end HTTP communications. Since UPnP provides an efficient protocol to discover newly associated devices even if the device is directly associated to a constrained network such as ZigBee, we propose to extend CoAP to include UPnP related commands as methods. This provides flexible communications between applications and sensor devices and relaxes the performance requirements on the gateway. This may, on the hand, overload the gateway and sensor devices. We examined the performance of our implementation to find that the serial interface between sensor MCU and sensor transceiver could be a bottleneck for sensor device to application traffic. On the other hand, too many subscription from applications may result in overload in the gateway and sensor devices. Once we clarify the traffic limitations and confine the traffic within the limits, our web based Internet of Things shows satisfactory performance. We can subscribe up to 16 sensor device with 500 msec interval by submitting simple HTTP POST data to sensor devices.

5. ACKNOWLEDGEMENT

This research is supported by Japan Ministry of Internal Affairs and Communications under a contract "Net-

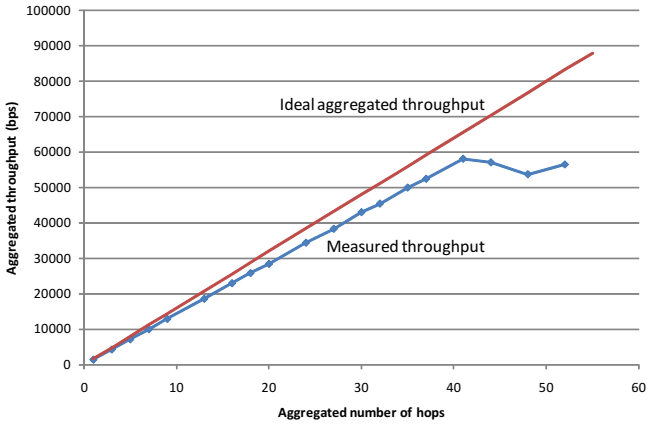

Figure 14: We can subscribe up to 16 sensor device at the maximum throughput of a single sensor device. Each square in Measured throughput indicate the increment of number of sensor device.

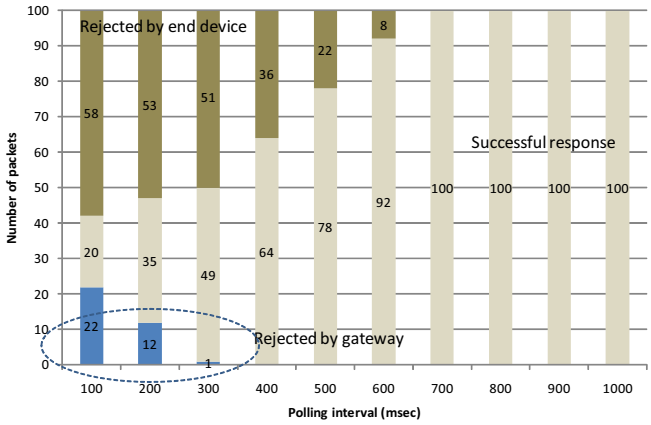

Figure 15: Polling traffic need to be control to prevent less than 700 msec time interval polling requests.

work controlled systems standardization, WiMAX data collection" in Fiscal 2010. The authors appreciate the technical supports from Toppan Printing, Co., LTD. and Alpha systems Inc.

6. REFERENCES

[1] Shahin Farahani, "ZigBee Wireless Networks and Transceivers", Newnes, (2008).

[2] "IEEE Standard for Information technology. Telecommunications and information exchange between systems. Local and metropolitan area networks. Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)", (2006)

[3] Global and pluggable sensor and actuator networking framework, EC FP7 SENSEI document D.3.2, (2009).

[4] Mik Botts, George Percivall, Carl Reed, John Davidson, Editors, "OGC Sensor Web Enablement: Overview And High Level Architecture", OCG 07-165, (2007).

[5] Joachim W. Walewski, Editor, "Initial Architectural Reference Model for IoT", EC FP7 IoT-A (257521) D1.2, (2011).

[6] UPnP Device Architecture 1.1, (2008)

[7] Kawamoto, R. Emori, T. Sakata, S. Furuhata, K. Yuasa, K. Hara, S. "DLNA-ZigBee Gateway Architecture and Energy Efficient Sensor Control for Home Networks," 16th IST Mobile and Wireless Communications Summit, 1-5 July 2007.

[8] Kuk-Se Kim; Chanmo Park; Kyung-Sik Seo; Il-Yong Chung; Joon Lee, "ZigBee and The UPnP Expansion for Home Network Electrical Appliance Control on the Internet", The 9th International Conference on Advanced Communication Technology,vol. 3, (2007), pp.1857-1860.

[9] Seong Hoon Kim, Jeong Seok Kang, Hong Seong Park, Daeyoung Kim, Young-joo Kim,"UPnP-ZigBee internetworking architecture mirroring a multi-hop ZigBee network topology", IEEE Transactions on Consumer Electronics, vol. 55, Issue 3, (2009), pp.1286-1294.

[10] H., Hada, J.,Mitsugi, "EPC based Internet of Things Architecture", IEEE RFID-TA, September, (2011).

[11] J.Mitsugi, H., Hada, T.,Inaba, K.,Ihara, G.,Kojima, T.,Kondo, "Enabling globally unique Sensor ID with dual-interface RF tag", IEEE Sensors, November, (2011).

[12] "EPCglobal Object Name Service (ONS) 1.0.1", (2008).

[13] Shelby,Z., Frank,B., and Sturek,D.,"Constrained Application Protocol", Internet Draft draft-ietf-core-coap, (2010).

[14] Koojana Kuladinithi, Olaf Bergmann, Thomas Potsch, Marjus Becker, Carmelita Gorg, "Implementation of CoAP and its Application in Transport Logistics", Extending the Internet to Low Power and Lossy Networks", IP+SN 2011, (2011).

[15] EPCglobal, "EPC information services version 1.0.1 specification", (2007).