Integrating Wireless Sensor Network into Cloud services for real-time data collection

Conference Paper · October 2013

DOI: 10.1109/ICTC.2013.6675470

8 authors, including:

Rajeev Piyare
Fondazione Bruno Kessler

Sun Park
North West Institute of Engineering & Technology

Sang hyeok Park
Catholic University of Korea

Integrating Wireless Sensor Network into Cloud Services for Real-time Data Collection

Rajeev Piyare*, Sun Park, Se Yeong Maeng, Sang Hyeok Park, Seung Chan Oh, Sang Gil Choi, Ho Su Choi and Seong Ro Lee
Department of Information Electronics Engineering
Mokpo National University
534-729, Korea South
rajeev.piyare@hotmail.com

***Abstract*— This paper presents an extensible and flexible architecture for integrating Wireless Sensor Networks with the Cloud. REST based Web services is used as an interoperable application layer that can be directly integrated into other application domains for remote monitoring such as e-health care services and smart environments. For proof of concept, we have set up a REST based Web services on an IP based low power WSN test bed, which enables data accessibility from anywhere. The alert feature has also been implemented to notify users via email or tweets for monitoring data when they exceed values and events of interest.**

***Keywords*— Internet of Things, Wireless Sensor Networks, Cloud computing, Web service, Open.Sen.se server.**

I. Introduction

The Internet of Things (IoTs) can be described as connecting everyday objects like smart-phones, Internet TVs, sensors and actuators to the World Wide Web where the devices are intelligently linked together enabling new forms of communication between things and people, and between things themselves. Building IoTs has advanced significantly in the last couple of years since it has added a new dimension to the world of information and communication technologies. According to [1], in 2008, the number of connected devices surpassed connected people and it has been estimated by Cisco that by 2020 there will be 50 billion connected devices which is seven times the world population. Now anyone, from anytime and anywhere can have connectivity for anything and it is expected that these connections will extend and create an entirely advanced dynamic network of IoTs.

In fact, one of the most important elements in the Internet of Things paradigm is wireless sensor networks (WSNs). WSNs consist of smart sensing nodes with embedded CPUs, low power radios and sensors which are used to monitor environmental conditions such as temperature, pressure, humidity, vibration and energy consumption [2]. Since, the number of users of the Internet is increasing therefore; it is wise to provide WSN services to this ever growing online community.

Cloud services are a flexible, powerful and cost-effective framework in providing real-time data to users at any time with vast coverage and quality. The Cloud consists of hardware, networks, services, storage, and interfaces that enable the delivery of computing as a service [3]. In addition, it's also possible to upload the data obtained from the wireless sensor nodes to the Web services based on Simple Object Access Protocol (SOAP) and Representational State Transfer (REST), using messaging mechanisms such as emails and SMS or social networks and blogs [4]. By bridging, evaluating and linking these sensor networks, data conclusions can be made in real-time, trends can be predicted and hazardous situations can be averted.

In this paper, we present the design, development and integration of an extensible architecture for WSN with the Cloud based sensor data platform, Open.Sen.se [5] where info-graphic of different data streams can be displayed, accessed and shared from anywhere with Internet connectivity. The collected data from the sensor nodes are processed, stored and analyzed on Open.Sen.se server via an Application Programming Interface (API). For proof of concept in a smart environment, we have implemented a REST based Web services on an IP based low power WSN test bed, which enables data access from anywhere for the smart environment.

The remaining of the paper is organized as follows. In Section II, we briefly discuss related work. Section III describes the proposed architecture while Section IV discusses the middleware implementation of our approach. Section V presents the implementation results and discussions and finally, some conclusions are presented.

II. Related Literature

Wireless sensor platforms have been widely deployed in a number of applications ranging from medical such as Alarm-Net [6], or CodeBlue [7] to environmental monitoring [8, 9]. The architecture of these systems has been designed in a very ad hoc fashion and is not flexible to adapt to other applications or scenarios while the core problem is the same, remote monitoring using sensor networks. During the last few years, many researchers have investigated on ways to connect wireless sensor networks to the Cloud. Authors in [10-14] have presented Internet protocols for connecting wireless sensor networks to the Internet but no real implementations have been shown. Much of the previous work has been on theoretical aspects of system architecture rather than actual deployment. Use of Web services to connect sensor networks with external networks have also been suggested by researchers in [15]. However, their work was mainly focused on the feasibility of

SOAP based Web services in terms of energy and bandwidth overheads.

SenseWeb [16] is one of the first architectures being presented on integrating WSN to the Internet for sharing sensor data. Users were able to register and publish their own sensor data using the SenseWeb API. The main drawback of SenseWeb is that all the decision making process is executed at a single central point called the Coordinator. That is, all the intelligence to control and to make a decision is located at this central point and if the Coordinator fails, the entire network is disrupted.

It is therefore suggested that the various decision levels can be implemented onto different architectural layers. The upper level known as Supervision Layer will be used for all sensor data storage, analysis and for decision making, while the Sensor Layer where the sensors are deployed can be used to partially analyze data and the determination of reactive response. The Coordinator still exists as a point of control for analysis of data and remote monitoring as well as acting as a gateway between sensors and the Cloud service.

In order to address the above mentioned issues of flexibility and centralized decision making process, we designed and implemented a more flexible architecture for integrating WSN to Cloud. The architecture presented in this work can be customized in different ways in order to accommodate different application scenarios with minimum recoding and redesign. Due to low power, simple network deployment, reliable data transmission and low installation costs, the ZigBee wireless standard has been preferred for this study over Wi-Fi and Bluetooth. In addition, to reduce the overall cost of implementation and network latency, each End Device is only equipped with an XBee ZB module with sensors.

III. PROPOSED ARCHITECTURE

The architecture of the proposed system is divided into three layers (Fig. 1): Sensor Layer, the Coordinator Layer and the Supervision Layer.

The Sensor Layer consists of sensors that interact with the environment. Every sensor was integrated with wireless nodes using an XBee ZB platform called End Devices. These End Devices form a Mesh network and send the information gathered by the sensors to the Coordinator Layer through the sink node called the base station. Messages are routed from one End Device to another until they reach this base station.

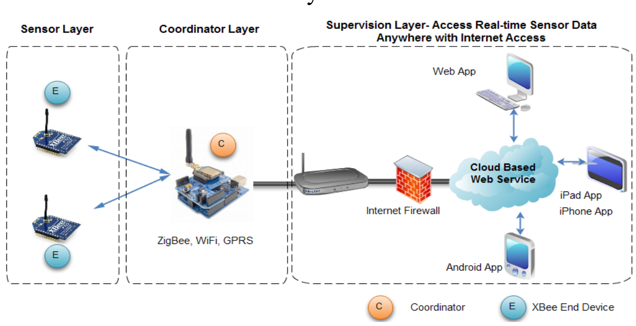

Figure 1. Proposed Architecture

For our prototype system, we have utilized XBee module from Digi International, Inc. Each XBee ZB module has the capability to directly gather sensor data and transmit it without the use of an external microcontroller, a capability known as XBee direct [17]. This offers many advantages. By excluding the external microcontroller, the overall size of the project can be reduced. By using XBee alone, it can minimize weight which is an important factor for systems such as Body Sensor Networks or wearable's. Omitting an external microcontroller also reduces power consumption which is a critical advantage for wireless systems that run on batteries and save money.

The Coordination Layer is responsible for the management of the data received from the sensor network. It temporarily stores the gathered data into buffer and sends it to the Supervision layer at predefined intervals. Base station which comprises of Arduino UNO, Ethernet shield and XBee is connected to the Internet using RJ45 cable and is powered using an AC adaptor. It serves as a mobile mini application server between the wireless sensors and the dedicated network and has more advanced computational resources compared to the End Devices found in Sensor Layer. At the base station, the sink node gathers data from wireless sensors using the ZigBee protocol and sends this data to Cloud based sensor data platform.

Finally, the Supervision Layer accommodates the base station with a Web server to connect and publish the sensor data on the Internet. This layer stores the sensor data in a database and also offers a Web interface for the end users to manage the sensor data and generate statistics. For the Supervision Layer, we have used Open.sen.se [5] HTTP Service which provides a REST based API to publish and access the sensor data. Thus, allowing existing networks to be connected into other applications with minimal changes.

IV. CONNECTING SENSOR NETWORK TO CLOUD

The access to Cloud services has to be easy, direct, open and interoperable. That is, the provided communication means and programming interfaces (APIs) shall be easy to implement on every platform and developing environment [18]. The most open and interoperable way to provide access to remote services or to enable applications to communicate with each other is to utilize Web services. There are two classes of Web services: Simple Object Access Protocol (SOAP) and Representational State Transfer (REST). REST is a much more lightweight mechanism than SOAP offering functionality similar to SOAP based Web services.

Open.Sen.se is an open source "Internet of Things" application and API to store and retrieve data from things and sensors using Hypertext Transfer Protocol (HTTP) over the internet or via a Local Area Network (LAN). In addition to storing and retrieving numeric and alphanumeric data, Open.Sen.se API allows for numeric data processing such as time scaling, averaging, median and summing. The channel feeds supports JavaScript Object Notation (JSON), Extensible Markup Language (XML), and comma-separated values (CSV) formats for integration into applications.

Therefore, in our approach we have used REST based Web service utilizing standard operation such as GET and POST requests that return (JSON) responses to communicate between the base station and the Open.Sen.se server. JSON is a lightweight data-interchange format. It is easy for human beings to read and write. It is also simpler for machines to parse and generate messages than using XML. For example, to read the current sensor value, an HTTP GET request is sent to the resource of the sensor. The response includes a textual representation of the current sensor value. A soon as the Coordinator decodes the received data packets from the End Devices, an HTTP POST request is sent from the base station to a pre-specified URL, containing the updated value as illustrated in Fig. 2. To access the Open.Sen.se API, the following base URL is used: *http://api.sen.se*. Each data entry is stored with a date and time stamp and is assigned a unique Entry ID. In terms of authentication, every communication between the connected Device and Open.Sen.se server is protected with a Sen.se key which is specific and unique to each user.

```
117.17.80.199
117.17.80.1
Init done
Received I/O Sample from: 13A20040902679
Sample contains analog data
POST value for feed indoor-temperature
Posting in feed indoor-temperature
In postSense
POST in feed indoor-temperature succeeded
POST value for feed node-voltage
Posting in feed node-voltage
In postSense
POST in feed node-voltage succeeded
```

Figure 2. Notification indicating succesful POST for data values

V. IMPLEMENTATION RESULTS AND DISCUSSION

In order to evaluate and demonstrate the proposed model, we implement it by using the technical approach which is described in the above sub-sections. A WSN was created to collect temperature and battery voltage readings. Preliminary experiments were performed to evaluate the system in terms of sensor data accessibility, alert notification time, and battery consumption. Furthermore, Senseboard was created on Open.sen.se server to present the collected data to the user in an easy and meaningful way.

A. Senseboard

Open.Sen.se server offers graphical interface called 'Senseboard' where different apps can be added. This allows info graphic data streams to be displayed and viewed in real-time anywhere and on any website. It also offers critical multiviz functionality to combine data from multiple sensors into one graph. Fig. 3 shows the real-time acquisition curve with measurements showing environment temperature (Red Line) and End Device battery voltage (Green Line). The Senseboard created for this implementation is supported by Internet Explorer, Safari, Firefox, Opera browsers and can also be accessed at http://open.sen.se/sensemeters/tab/3114/.

B. Event Notification

An event notification system is also implemented on Open.Sen.se server based on measurements from sensors and

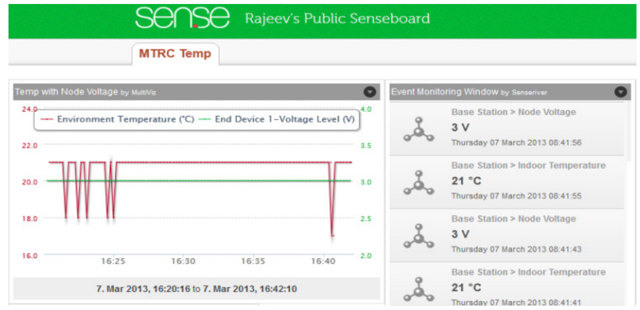

Figure 3. Senseboard displaying real-time Node Voltage and Environmental Temperature

predefined If-conditions. This allows monitoring End Devices supply voltage. If the voltage is too low, the End Device will enter sleep mode automatically. When Open.Sen.se server receives the voltage data for each remote Node through the base station, it compares it with a predefined threshold of 2.1V. If the measurement is equivalent to the threshold, it triggers the predefined actions. For instance, it can send a notification alert to the user via a push email or tweets. Fig. 4 demonstrates the notification email received by the user as soon as the threshold value is reached.

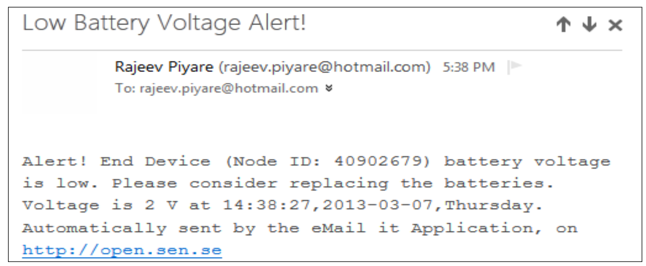

Figure 4. Notification Email to alert the user on low battery voltage

The time taken to notify the user from the time the event has occurred, in this case low battery voltage, was also measured. Using a variable DC power supply, the voltage for the End Device was manually reduced to 2.1V and the time it takes to receive the alert notification via an email was noted. Ten trials were conducted and it is observed that it takes about 8-13s and an average of 11s for the notification email to be auto generated and delivered to the user on their specified email account from the Open.Sen.se server (Fig. 5). For event notification, we consider this value to be acceptable as the required time to notify the user.

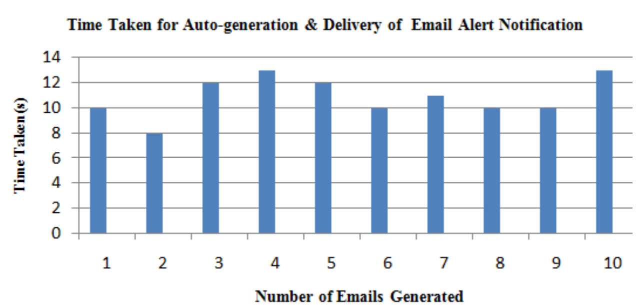

Figure 5. Time taken for auto generation and delivery of Email Alert Notification (10 attempts)

C. Battery Lifetime of the End Devices

For wireless sensor networks, energy efficiency is one of the important functional indexes since it directly affects the life cycle of the system. Replacing batteries regularly for failed sensor nodes in huge wireless networks is not convenient due to terrain and space limitations and also due to hazardous environments in which they are placed in. Therefore, the best method to save energy is setting sleep mechanism. The power consumption measurement is only carried out for the End Devices as the Coordinator is mains powered at the base station. To provide for an energy-efficient operation mode, End Devices are configured to be in a cyclic sleep mode. After transmission has completed, the End Device will return to sleep mode for another sleep cycle. The following Table shows the average power consumption during different modes of an End Device. The measured average power consumption is not considering the power consumed by the XBee module only, but also includes the voltage regulation component and its peripheral circuits.

TABLE I. CURRENT MEASUREMENT OF AN END DEVICE

Parameters	**End Device**
Activate and Deactivate current (I_{onoff})	8.1mA
Listen current (I_{listen})	40mA
Transmitter current (I_{trans})	38mA
Sleep current (I_{sleep})	0.6mA
Battery Capacity	2000mAh
Battery Voltage	9V

a. Sample of a Table footnote. (Table footnote)

The transmitted data from the End Device consisted of 2 bytes (one to encode the sensed temperature and the other for the supply voltage). With this data, MATLAB® simulations were conducted to estimate the lifetime of XBee ZB wireless sensor nodes with variable data packet size and different values of consecutive transmission time (update period) as shown in Fig. 6.

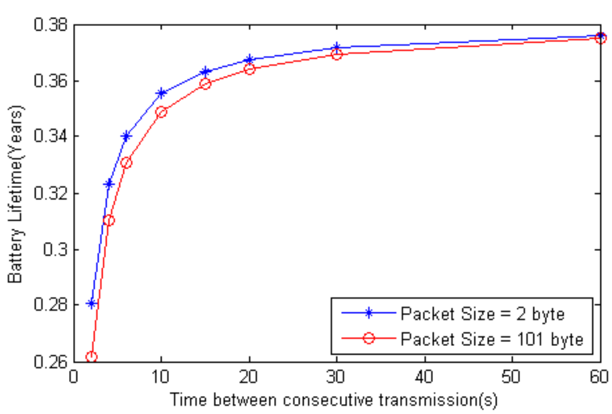

Figure 6. Wireless Sensor Network Node lifetime with different packet size and update period

The figures include two extreme cases for the value of data size: 2 bytes and 102 bytes which is the maximum admissible value of the ZigBee/802.15.4 MAC payload. From the results obtained, the figure shows that ZigBee technology provides a typical maximum battery lifetime of up to several years for many typical scenarios of mote networks. It was also observed that the lifetime of the node decreases as the packet size increases. Hence, it is also possible to achieve longer lifetime for battery powered sensor nodes using high current capacity lithium batteries.

VI. CONCLUSIONS

This paper proposed a flexible architecture for integration of Wireless Sensor Networks to the Cloud for sensor data collection and sharing using REST based Web services. To avoid loss of data and network disruption due to failure of Coordinator, we embedded intelligence at different architectural layers to accommodate for the diverse requirements of possible application scenarios with minimum redesign and recoding. The evaluation results illustrate that the sensor data can be accessed by the users anywhere and on any mobile device with internet access. The results also demonstrated that it takes an average of 11s for the alert notification email to be auto generated and delivered to the user on their specified email account from the Open.Sen.se server. In addition, using the sleep mechanism for low power XBee ZB transceiver modules provided an energy efficient approach to increase the lifetime of sensor nodes.

ACKNOWLEDGMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2009-0093828) and MSIP(Ministry of Science, ICT&Future Planning), Korea, under the ITRC(Information Technology Research Center) support program (NIPA-2013-H0301-13-2005) supervised by the NIPA(National IT Industry Promotion Agency).

REFERENCES

[1] M. Swan, "Sensor Mania! The Internet of Things, Wearable Computing, Objective Metrics, and the Quantified Self 2.0," Journal of Sensor and Actuator Networks, vol. 1, pp. 217-253, 2012.

[2] W. Wei, K. Lee, and D. Murray, "Integrating sensors with the cloud using dynamic proxies," in Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on,, 2012, pp. 1466-1471.

[3] Perumal.B, P. Rajasekaran.M, and Ramalingam.H.M, "WSN Integrated Cloud for Automated Telemedicine (atm) based e-healthcare applications," in 4th International Conference on Bioinformatics and Biomedical Technology, Singapore, 2012, pp. 166-170.

[4] C. Alcaraz, P. Najera, J. Lopez, and R. Roman, "Wireless Sensor Networks and the Internet of Things: Do We Need a Complete Integration?," presented at the 1st International Workshop on the Security of the Internet of Things (SecIoT'10), Tokyo ,Japan, 2010.

[5] (2012, 20th December, 2012). Open.Sen.se. Available: http://open.sen.se/sensemeters/tab/3114/

[6] A. Wood, G. Virone, T. Doan, Q. Cao, L. Selavo, Y. Wu, et al., "ALARM-NET: Wireless sensor networks for assisted-living and residential monitoring," University of Virginia Computer Science Department Technical Report, 2006.

[7] D. Malan, T. Fulford-Jones, M. Welsh, and S. Moulton, "Codeblue: An ad hoc sensor network infrastructure for emergency medical care," in International workshop on wearable and implantable body sensor networks, 2004.
[8] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, et al., "Deploying a wireless sensor network on an active volcano," Internet Computing, IEEE, vol. 10, pp. 18-25, 2006.
[9] J. Tooker, X. Dong, M. C. Vuran, and S. Irmak, "Connecting soil to the cloud: A wireless underground sensor network testbed," in Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012 9th Annual IEEE Communications Society Conference on, 2012, pp. 79-81.
[10] P. A. C. d. S. Neves and J. J. P. C. Rodrigues, "Internet Protocol over Wireless Sensor Networks, from Myth to Reality," JOURNAL OF COMMUNICATIONS, vol. 5, pp. 189-195, 2010.
[11] M. R. Kosanović and M. K. Stojčev, "CONNECTING WIRELESS SENSOR NETWORKS TO INTERNET," FACTA UNIVERSITATIS, Mechanical Engineering, vol. 9, pp. 169-182, 2011.
[12] A. E. Kouche, "Towards a wireless sensor network platform for the Internet of Things: Sprouts WSN platform," in Communications (ICC), 2012 IEEE International Conference on,, 2012, pp. 632-636.
[13] B. Li and J. Yu, "Research and Application on the Smart Home Based on Component Technologies and Internet of Things," Procedia Engineering, vol. 15, pp. 2087-2092, // 2011.
[14] N. Mitton, S. Papavassiliou, A. Puliafito, and K. S. Trivedi, "Combining Cloud and sensors in a smart city environment," EURASIP Journal on Wireless Communications and Networking, vol. 2012, p. 247, 2012.
[15] D. Guinard and V. Trifa, "Towards the web of things: Web mashups for embedded devices," in Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain, 2009.
[16] A. Kansal, S. Nath, J. Liu, and F. Zhao, "SenseWeb: An Infrastructure for Shared Sensing," IEEE MultiMedia, vol. 14, pp. 8-13, 2007.
[17] C. Chiu-Chiao, H. Ching Yuan, W. Shiau-Chin, and L. Cheng-Min, "Bluetooth-Based Android Interactive Applications for Smart Living," in Innovations in Bio-inspired Computing and Applications (IBICA), 2011 Second International Conference on, 2011, pp. 309-312.
[18] C. Doukas, Building Internet of Things with the Arduino vol. 1, 2012.