Energy-efficient Internet of Things Monitoring with Low-Capacity Devices

Rodrigo J. Carbajales, Marco Zennaro, Ermanno Pietrosemoli
T/ICT4D, Wireless Communications Laboratory
International Centre for Theoretical Physics (ICTP)
Trieste, Italy
{*rcarbaja, mzennaro, ermanno*}*@ictp.it*

Felix Freitag
Department of Computer Architecture
Universitat Politècnica de Catalunya
Barcelona, Spain
{*felix*}*@ac.upc.edu*

***Abstract*—The Internet of Things (IoT) allows users to gather data from the physical environment. While sensors in public spaces are already widely used, users are reluctant to deploy sensors for shared data at their homes. The deployment of IoT nodes at the users premises presents privacy issues regarding who can access to their data once it is sent to the Cloud which the users cannot control. In this paper we present an energy-efficient and low cost solution for environmental monitoring at the users home. Our system is builds completely on open source components and is easy to reproduce. We leverage a community network to store and control the access to the monitored data. We tested our solution during several months on different low-capacity single board computers (SBC) and it showed to be stable. Our results suggest that this solution could become a permanently running service in SBCs at the users homes.**

***Keywords*-Internet of Things, Community Networks;**

I. Introduction

The Internet of Things (IoT) allows users to gather data from the physical environment. While sensors in public spaces are already widely used, users are reluctant to deploy sensors for shared data at their homes. Similar as public sensors have helped to optimize city resource management by reducing costs, there is a huge potential for services in the users' homes to assist in local decisions and ICT management.

Several reasons explain this lack of users' willingness to participate: We first can see that privacy issues play an important role. Many of today's IoT application foresee the sensors at the users home to send their data directly to the cloud service offered by the commercial provider (often, sensors and cloud service are offered by the same company), where users can then access and visualize their data. The users, however, are concerned about their lack of control on their data once in the provider's platform. Secondly, the cost of commercial solutions may be another obstacle that has so far hindered the massive take-up of the technically already existing and mature offers. Monthly fees for data storage and vendor-lockin are blocking further user engagement.

Technical solutions and application areas for local IoT services have been identified within the area of Fog Computing [9], where a small device close to the data obtaining sensor carries out initial processing. Concrete solutions for data transformations by devices at the users homes have been proposed in [10]. In their work data from community facilities was gathered and processed locally, resulting in important traffic savings. A community context for cloud computing has been shown in [3]. In this work, the trust that exists within a community of users helped gather local computing resources form participants on which distributed services were run. In [2], it was pointed out in that resources from many distributed nodes located on user premises cloud host local services while being energy-efficient.

In this paper, we propose and analyze a solution for energy-efficient and low cost solution environmental monitoring at the users homes. Our system is builds completely on open source components and is easy to reproduce. We leverage a community network to store and control the access to the monitored data. We tested our solution during several months on different low-capacity single board computers (SBC) and it showed to be stable. Deploying an open IoT infrastructure in a local context lets the users manage the stored information, adding privacy and flexibility to the users data management. From the obtained results, we see our solution as a suitable candidate to become a permanently running service in SBCs at the users homes.

II. Proposed System

A. Overall scenario

We consider wireless environment sensors that are installed by users in places of their interest, e.g. offices, houses, neighbourhood, with the purpose of assessing environmental parameters. These sensors are connected though Wifi with their LAN. They can thus transmit their data either to devices located in the same LAN or through the router to devices in other networks. On these devices, the data is stored and can be further processed.

While our solution can be applied in general, we illustrate in Figure 1 a concrete case of a community where we have deployed our system. In this case of a community network, the users build a network to interconnect with each other through wireless links. While the interconnected nodes form the backbone network, at the users' home, local access points (APs) are created to which the user's devices are connected. In our scenario, the environmental sensors (SCKs) connect through Wifi to the AP. In addition, the SBCs also at the

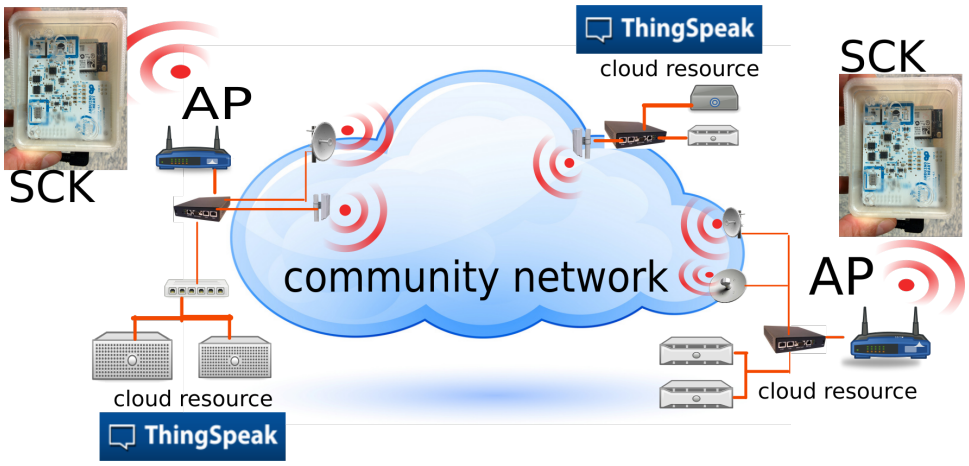

Figure 1. IoT monitoring with low-capacity devices in community network

users homes connect to the AP. In additional, at some locations more powerful resources like desktop PCs are connected. We foresee in the SBCs to host the data gathering platform, for instance ThingSpeak.

We note that an distinguishing feature of our scenario is that each node in the community network has a range of routable addresses (typically /27) available for local devices. As a consequence, devices connected to other nodes can be servers and can be reached. This is different to the typical situation in DSL Internet connection of home users where devices are behind NAT and no static public IP address is assigned.

B. Smart Citizen Kit

The wireless sensor kit from SmartCitizen [6] was chosen. It hosts with nine environmental sensors a variety of possibilities for measuring air quality. The Smart Citizen Kit (SCK) is a project which provides low-cost hardware and open source software. The embedded solution of the SCK is an Arduino AtHeart [7] which is easy to program and communicates with the computer over an USB interface. On the software side, Arduino provides a number of libraries to make programming the micro-controller easier. The simplest of these are functions to control and read the I/O pins rather than having to fiddle with the bus bit masks normally used to interface with the micro-controller.

The kit is composed of two boards, the ambient board with sensors and the Arduino data-processing board. The Environment board as seen on figure 2 is equipped with the following sensors:

- Amount of gases (CO & NO_2)
- Temperature
- Sound level
- Humidity
- Light intensity

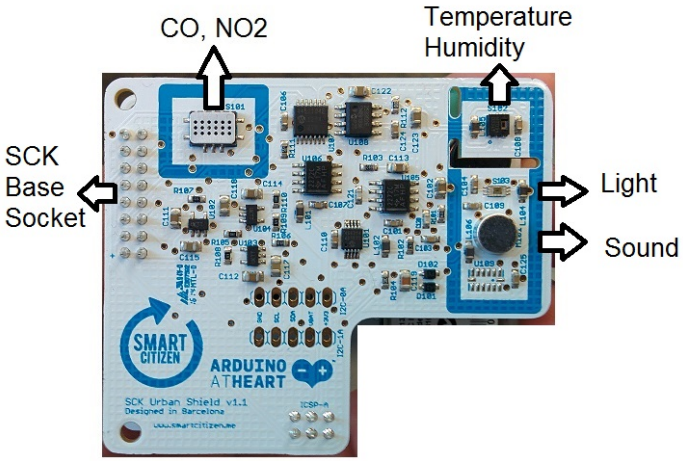

Figure 2. Smart Citizen Kit Environment Board

In addition, the Arduino data-processing board contains a voltage regulator that allows it to be fed by a photovoltaic panel, facilitating grid positioning. It is equipped with a WiFi radio as seen on figure 4 that allows to upload data from the sensors in real time to an on-line platform.

Once it is set up, the ambient board streams the measurement values over the WiFi module of the Arduino data-processing board. The devices low power consumption allows for placing it on balconies and windowsills. Power to the device can be provided by a battery, replenished by solar panel or other voltage source. The SCK device can be fitted with a 3D printed enclosure that makes it suitable to be placed on the open air.

C. Single board computer

For this work, several SBCs were used. We tested our solution to run on the Raspberry Pi (models A and B) and BeagleBone Black and Alix. The main characteristics of the

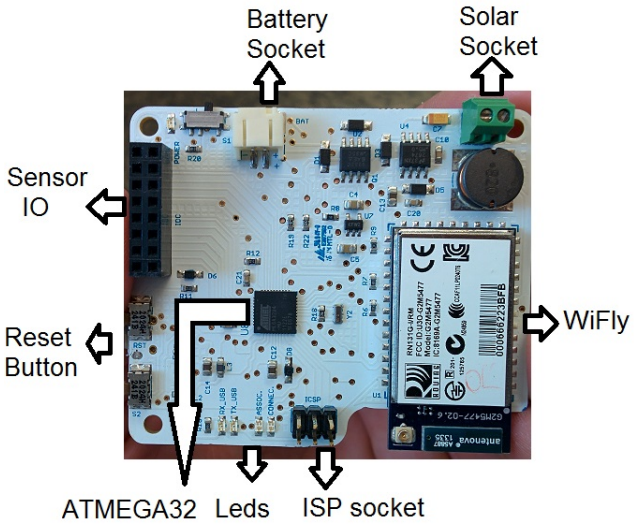

Figure 3. SCK Arduino data-processing Board top.

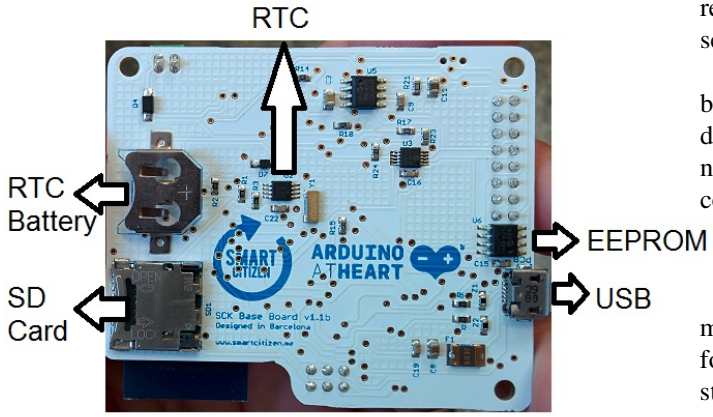

Figure 4. SCK Arduino data-processing Board bottom

SBCs are shown in Table 1.

This low cost and compact solution hosted both Cloudy and the TS server.

D. ThingSpeak data platform

The platform we use to gather the monitored data is ThingSpeak (TS) [5]. It is a free open source IoT application with an Application Programming Interface (API) designed to store and retrieve data from sensors using HTTP over the Internet or via a Local Area Network. Sensor logging applications, location tracking applications, and a social network of things with status updates can be created.

In addition to storing and retrieving numerical and alphanumerical data, the API allows for numerical data processing such as time-scaling, averaging, summing, and rounding. Each channel connected to a sensor supports data entries of up to 8 data fields, including latitude, longitude, elevation, and status. The channel feeds support JSON, XML, and CSV formats for integration in a variety of applications.

The TS on-line platform has some restrictions, like a minimum time of 15 seconds to update measurement data and it also requires a permanent Internet access, which is not always guaranteed in community networks often built with low cost and low reliability devices. Therefore, we installed our own version of the TS server that does not require an Internet connection, and also removing the 15 seconds limitation in upgrading time.

The TS source code is open-source and hosted on GitHub by iobridge. It is built using Ruby on Rails and MySQL database. TS can be installed on a powerful servers, a normal desktop computers and, as we show, on single board computer (SBC).

Table I
SBCs USED.MODEL, μPROCESSOR AND RAM MEMORY.

SBCs	Beaglebone	Raspberry	Alix
Model	Black	A+	3D2
μProc.	1GHz	700MHz	500MHz
	ARM	ARM	AMD
	Cortex A8	1176JZFS	LX800
RAM	512MB	512MB	256MB

III. EXPERIMENTS

Several experiments were carried out to assess the performance the proposed system. We first look at ease of usage for data visualisation and stability of the system. Then we study CPU usage of ThingSpeak in the SBC and finally we measure the energy consumption of the SBCs.

A. Monitoring SCK data with ThingSpeak

ThingSpeak allows the user to display graphically in channels via Web interface the gathered sensor data. The user can configure a public or privat access to these channels. It can be seen in Figure 5 that ThingSpeak shows correctly in its graphical display the data received from the SCK, corresponding to six different sensors. ThingSpeak in this case was running in a Raspberry Pi. While the data shown corresponds to zooming into a small time period, our experiment in fact was run during several weeks. In this time, the system showed to be stable and was permanently operational. When comparing the values measured by several SCK which were close to each other, we noticed however that there were deviations between the measured values of each. We attribute this fact to a lack of calibration of the sensors.

B. Use of CPU in SBC running ThingSpeak

To measure the CPU usage of RPi by the TS server, 6 different situations were created. which varied the send

Figure 5. Example of ThingSpeak channel with sensors values

Table II
SCKs AND DATA SENDING PERIOD

num. of SCKs	Sending period (sec)
1	20 s
2	20 s
10	20 s
1	1 s
2	1 s
10	1 s

period and the number of SCKs. Table II summarizes the experimental values.

Figure 6 shows the CPU consumption of the RPi with TS. It can be seen in a sending period of 20s, the CPU can cope and finish the processing of the data sent from a different number of SCKs. When the sending period was reduced to 1s, we can observe that the CPU is more heavily used and CPU usage increases with the number of SCKs sending data. It seems thus that for typical situations of 1 or 2 SCKs per home, the RPi works correctly, its resources may be too constrained for being able to gather data from many SCKs, as may be the situation of a deployment in a neighbourhood.

C. Power consumption running TS.

In this experiment we aim at studying the energy consumption of the RPi during the TS execution. Figure 7 shows the experimental setup where a Rapi is connected

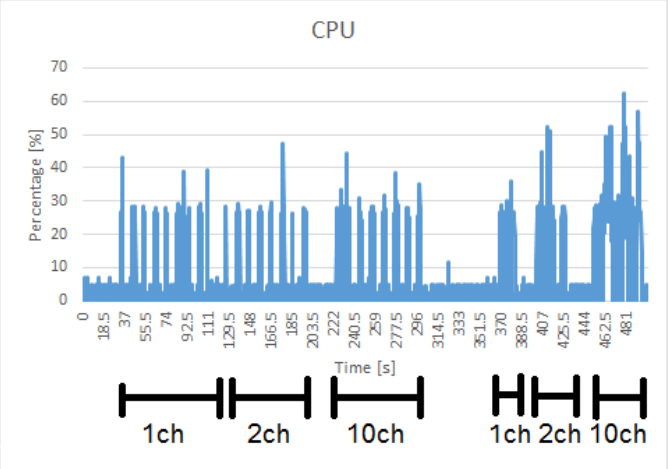

Figure 6. CPU percentage.

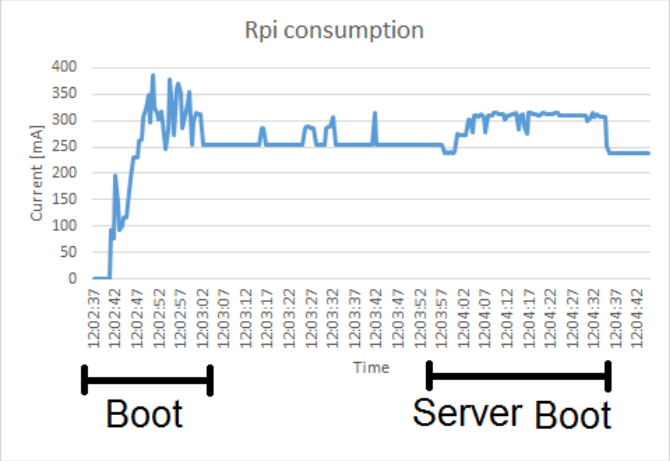

Figure 8. Boot consumption on RPi.

to power supply. A multimeter is used for measuring the current consumption.

Figure 8 shows the current used by the RPi during boot and when the TS server is started. It can be observed that the operations of the RPi are reflected in the current consumption.

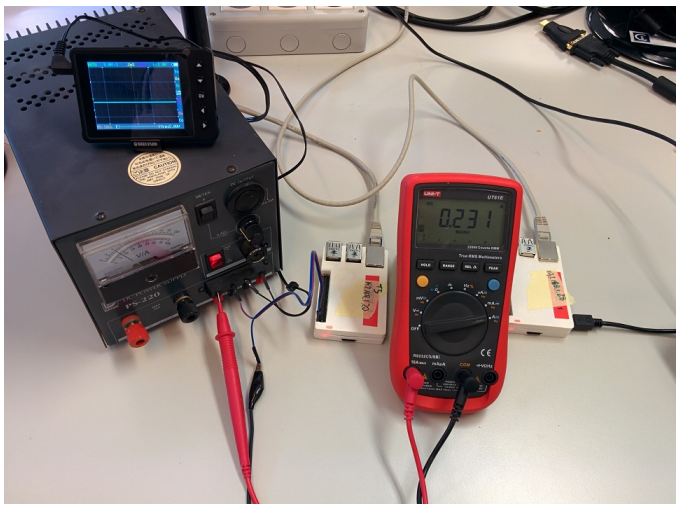

Figure 7. Measuring RPi power consumption and CPU percentage

The next experiments study the current consumption of the RPi when sending one channel and ten channels of data to the TS in the RPi every 20 seconds and every second. Figures 9, 10, 11, 12 show the RPi measured current consumption. It can be seen that the reception of data at each sending period leads to an increase of the current consumption. Comparing the scenario fo 1 and 10 channels, we can observe that 10 channels are reflected by a higher current consumption in the RPi.

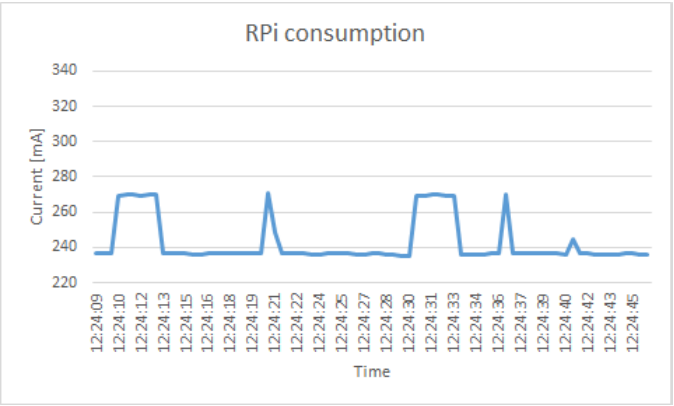

Figure 9. 1ch20s.

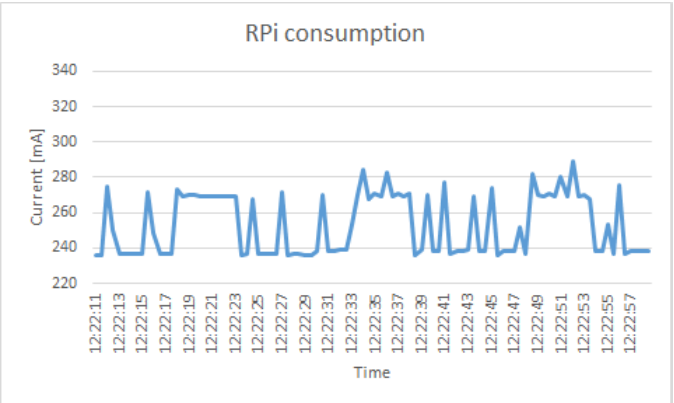

Figure 10. 1ch1s.

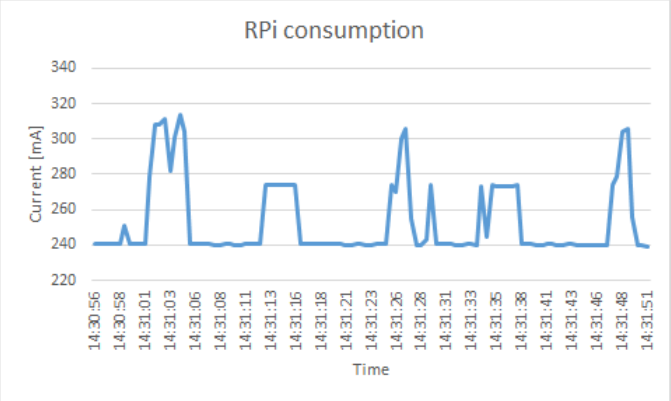

Figure 11. 10ch20s.

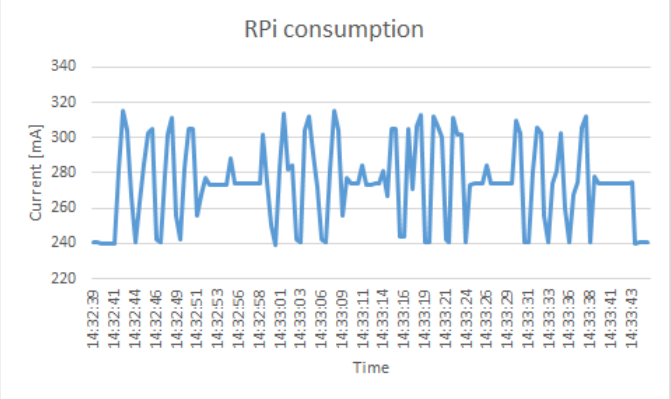

Figure 12. 10ch1s.

IV. Conclusion

A system was presented aiming at citizens to conduct IoT environmental monitoring with low-capacity devices in home environments, and for posterior sharing of the measured data among a community. The system was built with the Smart Citizen Kit (SCK) for measuring and sending the data, and the Raspbberry Pi SBC for hosting the ThingSpeak data platform.

The presented work showed advantages in energy consumption and cost, while being performant and uses-friednly. For this in the used SBCs a low electrical consumption was measured during the experiments. Thus the electricity bill of the users is kept low, allowing them to run this solution in a 24/7 mode. The system showed to be stable while it was run during several months in our experiments, and the measured values are easily available to the user through the ThinkSpeak Web interface. Regarding user friendliness, the configuration of the SCK is well documented and ThingSpeak channels can be configured with a few steps by an average user. The proposed solution therefor seems suitable for running as a permanent service in SBCs or home gateways deployed at users homes.

The proposed system could easily be extended to integrat and interact with more powerful resources for larger volumes of data. While currently the SCK data is of small size and send with moderate frequency, additional IoT sensors at homes may produce larger volumes of data also send to the SCB. Our next steps therefore will look at multiple data processing services running in the user homes SCB, and combining local storage with external cloud storage services.

Acknowledgements

This work is supported by European Community Framework Programme 7 FIRE Initiative project CLOMMUNITY, A Community Networking Cloud in a Box, FP7-317879.

References

[1] Mineraud J., et al. *"Contemporary Internet of Things platforms."* arXiv preprint arXiv:1501.07438 (2015).

[2] Freitag F., et al. *"A Look at Energy Efficient System Opportunities with Community Network Clouds"*, in Workshop on Energy Efficient Systems. EES 2014 at 2nd International Conference on ICT for Sustainability (ICT4S), Stockholm, Sweden, August 27, 2014.

[3] Jimenez J., et al. *"Supporting cloud deployment in the Guifi.net community network"* Global Information Infrastructure Symposium, 2013 , vol., no., pp.1,3, 28-31 Oct. 2013

[4] Telecommunications Network Open, Free and Neutral, http://guifi.net (2015).

[5] The open data platform for the Internet of Things. https://thingspeak.com (2015)

[6] Open source technology for citizens' political participation in smarter cities. https://smartcitizen.me (2015)

[7] Arduino AtHeart program based on Arduino technology. http://www.arduino.cc/en/ArduinoAtHeart (2015)

[8] SGX Sensortech Limited. Gas sensor MICS-4514 datasheet. http://www.cdiweb.com/datasheets/e2v/0278-Datasheet-MiCS-4514.pdf (2015)

[9] F. Bonomi et al. *"Fog Computing: A Platform for Internet of Things and Analytics"*, in Studies in Computational Intelligence: Big Data and Internet of Things: A Roadmap for Smart Environments, pp. 169-186, vol. 546, 2014.

[10] Kenji Yoi, Hirozumi Yamaguchi, Akihito Hiromori et al. *"Multi-dimensional Sensor Data Aggregator for Adaptive Network Management in M2M Communications"*. IFIP/IEEE International Symposium on Integrated Network Management, Ottawa, Canada, May 2015.