On the Integration of Heterogeneous Data Sources for the Collaborative Internet of Things

Federico Montori, Luca Bedogni
Department of Computer Science and Engineering (DISI)
University of Bologna, Italy
Email: {federico.montori2, luca.bedogni4}@unibo.it

***Abstract*—The Internet of things is foreseen as one of the next imminent Internet revolutions, as many devices will seamlessly communicate together to provide new and exciting services to the end users. One of the challenges that the IoT has to face is about both the heterogeneity of the data available and the heterogeneity of the communication. In this paper we focus on the former, by presenting an architecture able to integrate data coming from different sources, including custom made deployments and government data, and deliver new services made by the end users.**

I. Introduction

The Internet of Things (IoT) is one of the research and industrial fields that faced a rapid growth in the recent years, mostly thanks to the proliferation of new technologies associated with the easy of installatin and use. This developement has created ~~The approach to such ecosystem has been heterogeneous and sparse, leading to~~ a wide variety of standards and solutions at each layer of the network and application stack, leading to a heterogeneous environment both in terms of communication technologies as well as data storage, resulting in dosconncted network islands, in which it is easy to build networks with homogeneous devices, but it is hard to integrate data provided by other sources. ~~It has been a source of interest from many different points of view, in such a way that now we have dedicated infrastructures at physical, data link, network, transport and application layers.~~

Since the beginning of its diffusion, its potential has been explored in various fields of application and its major usefulness has been claimed to be in service composition and interoperability [1].

The requirements when designing collaborative IoT-related automation systems are varying due to the heterogeneity of the platforms and the hardware components as well as the network interfaces. This resulted in a sparse set of technologies and terminologies used in several scenarios determining a lack of interoperability among systems. The common approach to the problem of unifying entities within an ecosystem is typically architectural and leads to a difficult reuse of the components among different solutions [2]. To face these issues the European Commission supported initiatives like IoT-A [3], which aimed to release an architectural reference model and FI-WARE [4], which also helped architects in establishing an unified vision and nomenclature and now had become an implementation-driven open community. It also provided a sandbox in which partners could upload their open data, although it is not of broad use nowadays. However, such solutions did not solve the problem introduced by architectures, in fact different solutions tend to create separate islands which are hard to unify.

At the same time, makers worldwide build their own IoT in-home networks, which provide a low cost and customized environment that suit their needs. Platforms like Arduino and Raspberry Pi have demonstrated their easy-to-use and they fit most of the needs of citizens wanting to build their network. However, certain types of sensor can be expensive, or cannot be deployed, due to restrictions or for space limitations.

In this paper we aim to show the usefulness and the potential of open data exploitation for global cooperative IoT automation scenarios. As a preliminary approach, we focused our analyses on data coming from two of the main open data platforms for sensors, ThingSpeak and SparkFun, which provide open data coming from different types of sensors, along with themporal information, and possibly the location of the sensor. ~~Such platforms can be queried for data streams, supplied with a set of metadata, which gave us the motivation for such proposal. In fact, raw data streams come with temporal information, especially regarding the stream creation date and last update date.~~ Such parameters can tell us much about the general trend in the usage of these platforms throughout a time window of few years. Each stream in ThingSpeak comes with a creation date, which we reported, after a data extraction, in the diagram in Figure 1. Starting from such analysis it results a substantial growth in created channels starting from the end of 2014. A possible intuition behind them is the parallel innovation in simple hardware modules, for instance August 2014 corresponds to the launch of the first version of ESP8266 [5] on the market and in October 2014 was possible to flash its firmware though an SDK [6]. ~~Such period has not surprisingly been affected by a rapid growth according to the diagram. [TODO MORE EXAMPLES]~~ In the same figure we also report the time of the last update for all the found streams.~~The diagram reports also the last update date for each stream.~~ The oldest updates we found fall back to 2012, therefore we do not have nay information on streams that were last updateded before that date. ~~It is immediately clear that, since the oldest update date is in September 2012 while the oldest creation date is in 2010, all the data streams created before 2012 that did not~~

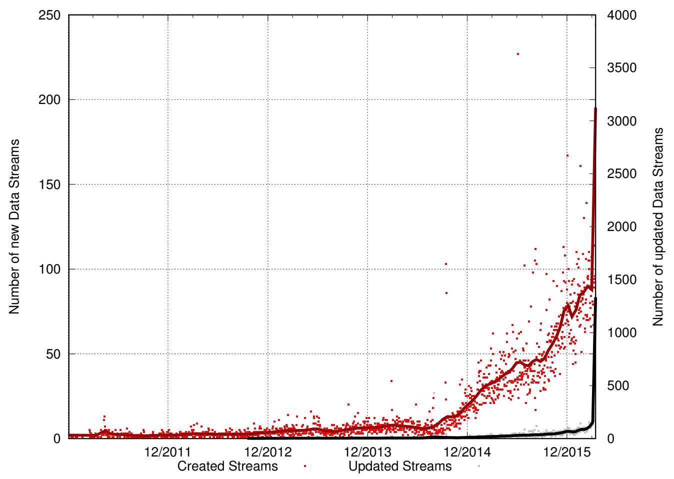

Fig. 1. Trend in creation of ThingSpeak channels.

~~perform an update after such date have been deleted due to a certain policy.~~ The steepness of the curve in the last days reveals that a significant amount of the data streams are still in use and updated daily or even hourly, and can therefore provide an updated data.

Such analyses shed some light on how rapidly the world of open data is growing and people are gaining interest in using a platform that takes away the burden of creating a local ecosystem. Thus, our work creates the basis for a solid global ecosystem with a strong impact on cooperative services, integrating data coming from custom made solutions with official data provided either by government agencies or other reliable data sources.

The rest of this paper is structured as follows: In Section II we present similar works from literature; Section III describes the data stream examples we considered for this analysis, and Section IV is devoted to the description on how it is possible to integrate heterogeneous data and the challenges on the topic; Section V describes our proposed architecture, and Section VI concludes this paper.

II. Related Work

The IoT is nowadays growing exponentially together with the number of solutions and architectures proposed to handle it. The total number of connected devices is expected to be 27 billions by 2024, while the total revenue opportunity is predicted to be up to $1.6 trillions [7] .

Regardless of the different perspectives, it is clear that the number of devices is growing, the data is becoming more and more heterogeneous, and one of the main challenges is how to handle such an amount of data and how to give a meaning to it. In the recent years there have been a huge number of attempts which, most of the times, are either self-contained since they require compliance to a specific framework, or commercial solutions.

Commercial solutions aim to constitute a living ecosystem in which entities are “plugged” and interoperable, participating for the benefit of the whole system and fully compliant with the other actors within the environment. Most of the times such frameworks, some of them depicted in [8], provide efficient software adapters for legacy systems. Such types of frameworks are often self contained and tend to create a cluster of devices which need to be framework-compatible in order to interoperate. An example is Cumulocity [9], a platform providing an unified service oriented HTTP REST interface to devices. Another project gaining interest in recent years is AllJoyn [10], developed by the Allseen Alliance. Such a framework again forces devices to either implement an attachment to a software bus between application, which is indeed the AllJoyn core, or connect to an AllJoyn router using a thin library. Either way, the communication introduces very low overhead and grants access to even constrained device. However, the protocol used is highly customized and makes AllJoyn a quite isolated ecosystem. Another example is Xively [11], which, again, allows devices to obtain interoperability even among different application protocols (CoAP, MQTT, HTTP, XMPP and others) offering an API that implements a custom message bus. Finally, another ecosystem to mention is the one implemented by Open Mobile Alliance, named OMA-LwM2M [12], which defines a custom layer over CoAP focused on exchanging instances called “objects” and operating upon them via the custom interfaces.

III. Open data as a source

As stated in the introduction, open data are the most powerful source of information when such data are not producible by the utilizers themselves. In this section we outline some of the well-known sources that we considered in order to achieve a homogeneous data store.

A. *ThingSpeak*

ThingSpeak [13], originally launched in 2010 by ioBridge, is an open source data platform and API for the IoT that enables the user to collect, store and analyze data. In more detail, it provides a personal cloud that users can deploy over their Local Area Network and easily display the data produced by sensors using ThingSpeak’s straightforward API. Data analysis and visualization have been made possible due to the close relationship between ThingSpeak and Mathworks, Inc. since such functionalities are driven by the integrated MatLab support. Furthermore, such a platform provides a global cloud hosting millions of open data records, called channels, which is useful both to users who cannot deploy their own cloud and to consumers who need to infer information coming from the stored data. Data is stored with an absolute freedom of expression, meaning that any data channel can have any name and does not need to stick to any format constraint. Data channels can be both private and public and provide also raw measurements encoded in XML, JSON or CSV and can be updated with new measurements every 15 seconds.

In the recent years, ThingSpeak had become very popular due to the rise of easily programmable IoT platforms such as Arduino [14], BeagleBone Black [15], ESP8266 [5] and many others. Such devices are becoming cheaper and cheaper and, on the other hand, it is easier to get started with them. Nowadays, for instance, an ESP8266 is able to manage a sensor, get connected through WiFi, be programmed through the simple, C-like Arduino SDK and still cost less than 5$ while its battery, if the duty cycle is low enough, is estimated to have a duration of around 7 years [16]. With a WiFi connection and an open platform such as ThingSpeak, a first home sensor network is very easy to boostrap, since the device controller does not need to have the control on the cloud and, furthermore, the data produced by the sensor is easily displayable on the end consumer's personal device, such as a Smartphone or similar.

B. Sparkfun

SparkFun Electronics, Inc. [17], founded in 2003 in Colorado, is a microcontroller seller and manufacturer, known for releasing all the circuits and products as open-source hardware. It also provides tutorials, examples and classes.

For the purpose of the present paper, SparkFun also hosts its own open source cloud of open data [18], on which the customers can test and upload the data collected by the embedded sensors. Users can push for free their data on such cloud in streams of 50 MB maximum size and with a maximum frequency of 100 pushes every 15 minutes. Unlike ThingSpeak, the location where the data comes from is specified at a coarse granularity since the name of the city is often obtainable, however the GPS coordinates are never given. On the other hand, data coming from SparkFun cannot be private and consumers can download stream contents encoded in JSON, XML, CSV, MySQL, Atom and PostgreSQL.

IV. Data Unification

In this section we point out the data streams' characteristics obtainable from our two open data clouds and how do we aim to unify them onto a single data cloud. We extracted the whole repositories and parsed the JSON files in order to give a first structure to such data. Since the data structure does not force strong constraints data is, as explained in detail in section IV, often incomplete.

Hereby are briefly presented the parameters that can be extracted from data streams:

- **Stream ID**: it is the data stream's unique identifier. In ThingSpeak it is represented by an incremental number, which is assigned when the stream is created. At the time of writing there are 28806 active streams with IDs spanning from 1 to 100172. In SparkFun the unique ID is given by a string of 20 random ASCII characters. At the time of writing we counted 3575 different SparkFun IDs.
- **Stream name**: it present in both platforms and it is decided by the user with no constraint. This means that it can even carry no useful information for its identification and categorization.
- **Geolocalization**: it is present in both platforms. In ThingSpeak not all the streams come with geolocalization data, however it is always given in GPS coordinates. Similarly, in SparkFun not all the streams are geolocalized, however, when they are, no GPS coordinates are given, but only the name of the city, or sometimes just the state or even the country, unless the user indicates the GPS coordinates in the data itself.
- **Tags**: are included in both platforms and often help to infer useful information about the data.
- **Creation Timestamp**: it is included in all ThingSpeak streams as a metadata, on the other hand, the creation date of a SparkFun stream is not included. Each SparkFun stream is limited to 50 MB, thus the first update of any SparkFun stream smaller than 50 MB corresponds to its creation date, however it is retrievable in a limited number of cases.
- **Last Update Timestamp**: it is included in all ThingSpeak streams as a metadata. In SparkFun is simply deducible from the timestamp of the last update in the stream, since the timestamp is included in each data update.
- **Description**: it is a ThingSpeak metadata and its characterization is fully assigned to the user (who can also decide not to include it).
- **Elevation**: it is a ThingSpeak metadata and not always indicated.
- **Last Entry ID**: it is a ThingSpeak metadata, which points to the last update record in the data, ordered using an incremental ID for each update.

From each data stream we extracted in particular the GPS position for a locational analysis, finding that such position is indicated, with different degree of precision, in 6665 data streams out of 32381. In 32% of such cases, only a macro area is given (the city, or even the state). These are the cases coming from SparkFun, for which we took the central position of the indicated entity using the Google Maps API. The result of the analysis is outlined in figure 2.

Given such results, the importance of information fusion from different sources is clear, since not only the sampling number of the sensing infrastructure is incremented, but also its coverage. Indeed, ThingSpeak appears to have much more utilization in the European region, whereas SparkFun seems to be more popular in North America. Furthermore, this consideration might be extended to different macro topic areas, meaning that some open data sources are specialized on a specific field of measuring. For instance, governmental sources providing open data such as EPA (United States Environmental Protection Agency) [19] are primarily focused on environmental data, whilst crowdsensing sources such as OpenSignal [20] regard measurements on cellular network signal strength and coverage.

Therefore, a basic unification counting on an essential set

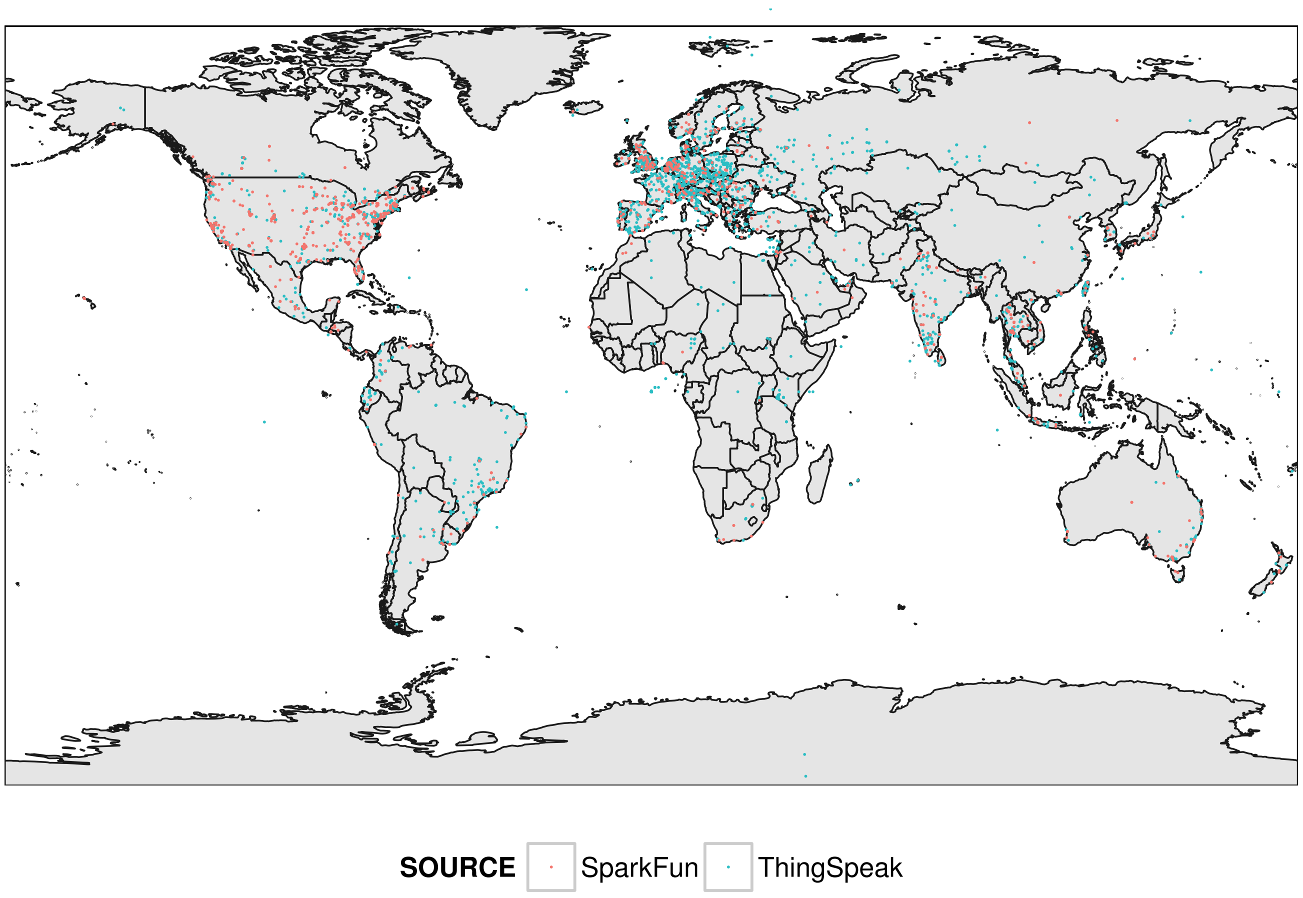

Fig. 2. Location of all ThingSpeak and SparkFun sensing sources.

of metadata is crucial, composing the minimum skeleton to which a data stream should be linked. For such purpose we aim to design an unique ID assignment policy, a geolocalization (with a precision error), a freshness of the information (given by the last update timestamp), a friendly name and an inferred measurement category (such as temperature, humidity and so on) together with an unit of measure.

V. Our Architectural Proposal

In this section we discuss the our proposal and how we intend to carry out the architectural design. We also give a glimpse on the case studies for this scenario in order to show why would someone use open data integration for measurement tasks.
Figure 3 shows an architecture depicting several use cases.

In our proposed architecture we assume to have a decision middleware, called Orchestrator, which is capable to return a service record, or a data stream, given a set of parameters determined by the user's choice. Such information is returned from one among the sources available, provided that the user specified his or her preference for "reliable" or "unreliable" data, which namely corresponds to official or user-defined respectively. We also aim to have our own data cloud for both data streams and services which are not intended to be published as open data onto one of the sources mentioned.

As a case study, an user can run an application making use of different measurements, for instance outdoor temperature and the amount of fine dust or pollen in the air, in order to infer an environmental condition or to trigger some action. For instance it would open automatically the window that is not facing the sun when it's too hot, but just if the pollen in the air is below a certain threshold, otherwise it turns on the air conditioning in order to avoid allergic reactions. Such a case study could happen in several situations. An user, for example, might be the owner of the temperature sensor, since it is cheap and easily configurable ("Sensor 3" in the figure). However, other sensors such as pollen sensors or fine dust sensors might be too expensive or rare to get, or simply not owned by the end user and therefore other users' measurements are needed, provided they are nearby enough (this is also why geolocalization is meant to be crucial).

Furthermore, different data sources might have a different update rate and a different "reliability", since they may

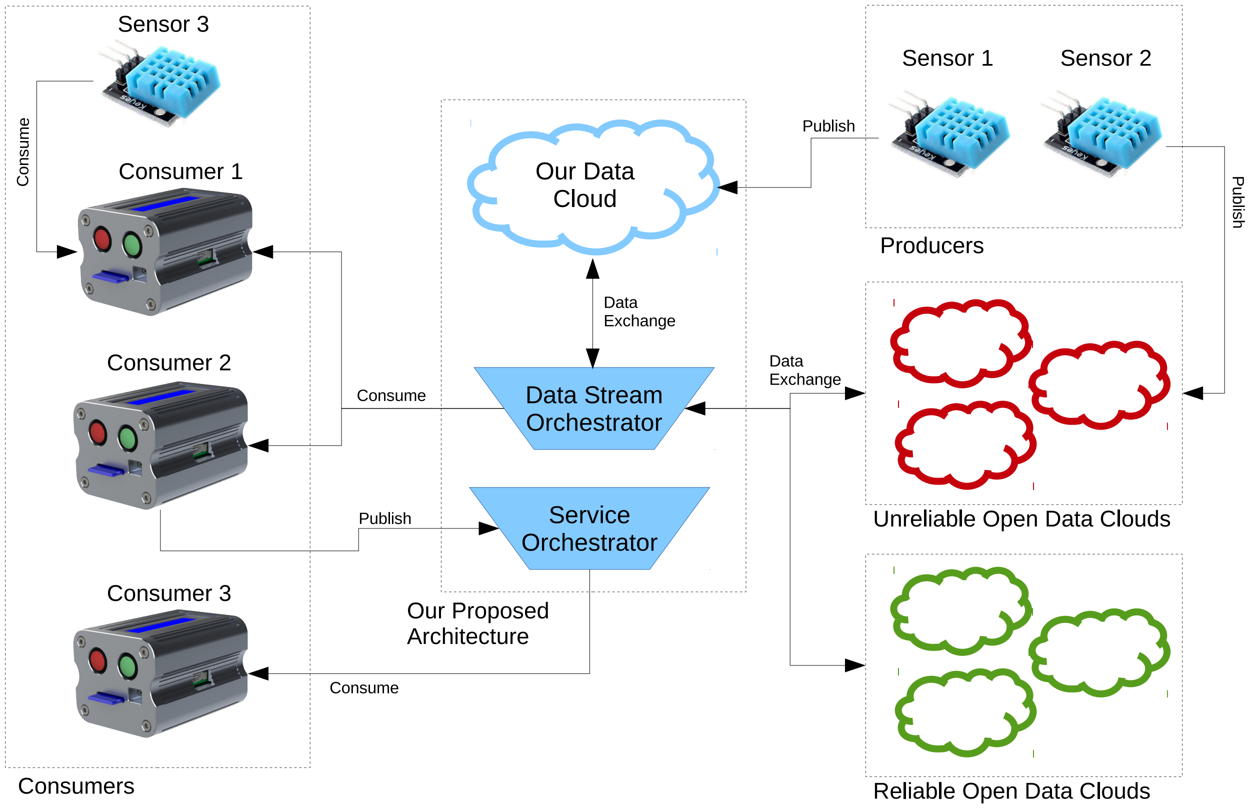

Fig. 3.

belong to providers that are recognized as trustworthy or not. This happens when such providers receive several positive feedbacks due to their estimated precision and update rate. Furthermore, in the integration presented we used two user-driven platforms, however, we aim to integrate our architecture with reliable and official data, such as EPA for the United States environmental measurements. Such measurements, since they are official, are considered to be reliable, however their slow update rate introduces a trade-off whenever an user must choose between a high reliability or a fast update rate. Some applications indeed might require information at a finer granularity over time, for example when they want to detect instantaneous condition changes. In such cases the user, choosing the update frequency at the expense of reliability, will necessarily use the data or the services provided by neighbors.

Our proposed architecture aims not only to unify raw data streams and make them universally available, but also to make users able to share their service endpoint and to provide additional capabilities derived from both data aggregation and personal computational capabilities ("Service 2" in the figure). As a simple example, an user receiving temperature and humidity data might calculate the heat index and expose it as a service interface.

Our proposal, given such a various set of use cases, provides the user with a wide variety of options regarding deployment and data retrieval. This is really important since the user is not forced to stick to a particular approach and gives a great advantage in an era where heterogeneity affects not only data and protocols, but also solutions.

VI. Conclusions

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.
[2] S. Krco, B. Pokric, and F. Carrez, "Designing iot architecture (s): A european perspective," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*. IEEE, 2014, pp. 79–84.
[3] "Iot-a european project," http://www.iot-a.eu/public/front-page, accessed: 2016-3-29.
[4] "Fi-ware open community," https://www.fiware.org, accessed: 2016-3-29.
[5] "Esp 8266," http://www.esp8266.com/, accessed: 2016-3-29.
[6] "Espressif sdk releases," http://bbs.espressif.com/viewforum.php?f=46, accessed: 2016-3-29.

[7] "Global m2m market to grow to 27 billion devices, generating usd1.6 trillion revenue in 2024," Machina Research, june 2015.

[8] H. Derhamy, J. Eliasson, J. Delsing, and P. Priller, "A survey of commercial frameworks for the internet of things," in *Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on*. IEEE, 2015, pp. 1–8.

[9] "Cumulocity framework," https://www.cumulocity.com/, accessed: 2015-11-30.

[10] "Alljoyn framework," https://allseenalliance.org/framework/documentation, accessed: 2015-11-30.

[11] "Xively," https://xively.com/, accessed: 2016-3-29.

[12] "Open mobile alliance lightweight machine-to-machine solution," ttp://openmobilealliance.org/about-oma/work-program/m2m-enablers/, accessed: 2016-3-29.

[13] "Thingspeak, the open data platform for the internet of things," https://thingspeak.com/, accessed: 2016-3-29.

[14] "Arduino," http://www.arduino.cc/, accessed: 2016-3-29.

[15] "Beaglebone black," http://beagleboard.org/black, accessed: 2016-3-29.

[16] A. Di Nisio, T. Di Noia, C. Carducci, and M. Spadavecchia, "Design of a low cost multipurpose wireless sensor network," in *Measurements & Networking (M&N), 2015 IEEE International Workshop on*. IEEE, 2015, pp. 1–6.

[17] "Sparkfun electronics," https://www.sparkfun.com/, accessed: 2016-3-29.

[18] "Sparkfun open data cloud," https://data.sparkfun.com/, accessed: 2016-3-29.

[19] "United states environmental protection agency," https://www3.epa.gov/, accessed: 2016-3-29.

[20] "Open signal," http://opensignal.com/, accessed: 2016-3-29.