

# Rapid Prototyping of An Automated Sensor-to-Server Environmental Data Acquisition System Adopting A FAIR-Oriented Approach

P. Celicourt<sup>1,2,\*</sup>, R. D. Sam<sup>2</sup>, and M. Piasecki<sup>3</sup>

<sup>1</sup>*Department of Soils and Agri-Food Engineering, Faculty of Agriculture and Food Sciences, Laval University, Quebec City G1V 0A6, Canada*

<sup>2</sup>*SENSAQ, LLC, Syosset 11791, USA*

<sup>3</sup>*Civil Engineering Department, The City College of New York, New York 10031, USA*

Received 04 October 2017; revised 08 April 2019; accepted 08 June 2021; published online 02 January 2023

**ABSTRACT.** Most existing environmental data acquisition systems are not designed to support automatic field data streaming to a data management system, but instead involve manual data exports therein. This paper introduces a FAIR-oriented (Findable, Accessible, Interoperable, and Reusable) approach and prototype of an automated sensor-to-web services and analytics wireless sensor network in which the aspects of data collection, transmission, and management as well as network organization are implemented automatically. The Python programming language was used to develop the necessary software components. The data and metadata supplied by custom-made stations are automatically stored in an extended instance of the Consortium of Universities for the Advancement of Hydrologic Sciences, Inc. (CUAHSI) Observations Data Model (ODM) to which a web interface is linked and makes the data available publicly in user's preferred units via Web Services and Data Analytics at a central station. The system has been initially tested in outdoor environments and the experiments demonstrate that it is effective in not only reducing the workload of the post-deployment phase, but also has potential to reduce human errors.

**Keywords:** data communications devices, microcomputer applications, rapid prototyping, software engineering, systems and software, wireless sensor networks, FAIR data

## 1. Introduction

Environmental monitoring is essential to capture past, current and potentially future conditions of the earth system and its biological communities. However, conventional approaches adopt networks of static and sparse measurement stations as it is prohibitively expensive to deploy and maintain dense and large networks of these stations to capture high-resolution spatio-temporal signals (Kumar et al., 2015; Lettenmaier, 2017; Solomatine et al., 2017; Tauro et al., 2018). In addition, hydrometeorological monitoring networks are reported to be in decline globally (Hannah et al., 2011; Whitfield et al., 2012; Walker et al., 2016) while recent advances in data mining have catalyzed the development of new models requiring more detailed data (Vogel et al., 2015). However, recent advances in sensor and computing technologies have enabled the development of both open-source, small and low-cost sensing platforms (e.g., Raspberry Pi and Arduino) operable by non-highly qualified personnel (Austen, 2015; Ali et al., 2016; Cressey, 2017; Turner et al., 2019; Jiang et al., 2019). They produce less accurate data, but at the

same time they have the potential to supplement traditional monitoring networks with additional higher-resolution observations (Hund et al., 2016; Jiang et al., 2016; Little et al., 2016). Even though robust, cheaper and lower maintenance sensing equipment are now available in the market (Buytaert et al., 2014), deploying a sensor network (front-end) and ultimately streaming the collected data to a data management system (back-end) remain a time-consuming, expensive and labor-intensive task (Hirafuji et al., 2011; Wong and Kerkez, 2016; Hanson et al., 2018). This presents a significant challenge to small-to-medium-scale research projects as often at such scale, the stations work in an interdependent fashion creating a meshed field-level network contrary to typical large-scale networks with sparse stations.

The front-end to back-end communication often requires not only the use of proprietary protocols but also the need to learn proprietary programming languages to configure and control the front-end devices. The front-end configuration and control tasks are even more complex when using sensors, sensor platforms and communication equipment from different vendors due to the need to integrate several different proprietary data formats and protocols. At the controller level, there also exist the complexities of hardware/firmware integration and associated calibration/electrical/signal compatibility issues, in addition to operating systems customization and installation using, for example, the Yocto project (<https://www.yoctoproject.org/>). Furthermore, the post-deployment data management and

---

\*Corresponding author. Tel.: +1 (418) 656-3173.  
E-mail address: paul.celicourt.1@ulaval.ca (P. Celicourt).

dissemination often involve substantial manual work along with a number of information technology solutions and idiosyncratic conventions to address data files naming, data transfer and parsing, in addition to additional software package installations. There are some representative examples of dataset documentation exhibiting such issues (see Fang et al., 2018; Longman et al., 2018; Ochoa-Tocachi et al., 2018; Rasouli et al., 2018; Spence and Hedstrom 2018, to cite only a few). In addition, it not only requires finding necessary expertise but also funds for medium to long term investment into data management infrastructure. These issues hinder the development of new sensor networks capable of performing to FAIR principles (findable, accessible, interoperable, reusable; Wilkinson et al., 2016), and remain a great challenge for large-scale aggregation (Hanson et al., 2018).

The need to develop efficient sensor-to-data dissemination workflow systems for environmental conditions monitoring has risen significantly over the past years as highlighted in Healey et al. (2014). This resulted in a plethora of approaches of various degrees of sophistication and complexity that address just what is needed at a certain moment rather than seeking to generalize workflow designs or make them adoptable to other situations. Our literature survey on data collection and management systems while yielding numerous documented approaches did not point to any workflow that offers automated and standards-based data archiving and/or access for small-to-medium-scale scientific applications. In addition, the use of Controlled Vocabularies and the incorporation of sufficient metadata to trace provenance of data and consequently their seamless interpretation, in most of the cases, were not addressed by the data management systems surveyed.

### **1.1. Review of Approaches to Sensor-to-Server Environmental Data Workflow**

Some environmental researchers have embraced a non-standard approach towards the development of sensor-to-data dissemination workflow systems. For instance, similar research efforts have been made to tether sensor networks and support data access via internet for applications that include monitoring of grid-connected photovoltaic system, precision agriculture, viticulture, aquaculture and structural health and environmental monitoring (Beutel et al., 2011; Colitti et al., 2011; Fernandes et al., 2013; Ferdoush and Li, 2014; Liao et al., 2014; Shariff et al., 2015; Li et al., 2016; Parra et al., 2017). Schneider et al. (2017) deploy a low-cost air quality sensing network in Oslo, Norway composed of 24 units connected to an online database server using General Packet Radio Service (GPRS) connections. Similarly, Gray et al. (2017) developed the prototype of an environmental data acquisition system adopting a sensor-to-cloud approach with limited provision of metadata. In all these case studies, the efforts did not consider any standard data model to support the data management process and the aspect of data interpretation and understanding was overlooked. In addition, the aforementioned work did not consider the implementation of standards-based metadata annotations, completeness of the annotations, the use of Controlled Vocabularies, and the ability

to search and retrieve the collected data in standard formats.

Other researchers have, instead, adopted a standard-based approach. As an example, Yang et al. (2009) introduced a system based on the Open Geographic Consortium (OGC) Sensor Web Enablement (SWE) standard (Botts et al., 2008). However, the set of metadata provided to describe the observations is limited due to the fact that the OGC-SWE, more specifically, its key component named SensorML (Botts, 2014), merely provides a general schema for describing sensor systems and processes and thus, fails to provide a detailed description of the hardware design (Villalonga et al., 2010). Furthermore, the SWE falls short in many other aspects as explained in Devaraju et al. (2015). However, the IEEE 1451 standards family (Lee, 2000), more specifically the IEEE 1451.0 (IEEE, 2007), bridges the gap in OGC-SWE to deal with sensor information using the Smart Transducer Web Services (STWS; Song and Lee, 2008). These open standards were developed in support of interoperability of transducers (sensors and actuators) and networked equipment and put forward the concept of Transducer Electronic Data Sheets (TEDS) in the form of an identification card attached to transducers. Despite its focus on high granularity information about transducers, the IEEE 1451 standards skip the (critically important) inclusion of a standards-based system for storing, managing, organizing, indexing, and documenting transducer data.

The aforementioned cases either did not use any metadata standards or the authors created their own idiosyncratic approach to metadata provision as noted by Tenopir et al. (2011) in their study of data management practices employed by environmental scientists. However, during the past decade, the Consortium of Universities for the Advancement of Hydrologic Sciences (CUAHSI) has developed the Hydrologic Information System (HIS) (Horsburgh et al., 2009) as a set of tools and data infrastructures to support data management and publication of field observations. While CUAHSI HIS, particularly the CUAHSI ODM (Horsburgh et al., 2008), has seen widespread popularity as an installation to share environmental data, like many existing systems, it is not a fully automated system for data input/output requiring considerable set up expertise when using some of the tools, such as the Streaming Data Loader (SDL). The authors' experiences using the SDL suggest that the manual mapping of sensor data files with metadata in the post-deployment phase of a sensor network is prone to be erroneous with significant impacts for large sensor networks. A reason to support this claim is that the mapping of every single column in a data file using the SDL is equivalent to traversing a multi-level tree structure where a choice of one element is to be made often between several elements displayed in a tabular format at every step. This process increases the user's cognitive workload as it requires that users keep track of the data column being mapped and its related metadata records at each step of the mapping process.

An attempt to circumvent the use of the SDL has been made by the Stroud Research Center which developed the Critical Zone Observatory (CZO) midStream system (Stroud Research Center, 2016). The midStream stack consists of a three-component (a receiver, a database and the midStream itself) Python-based package that automatically annotates streaming

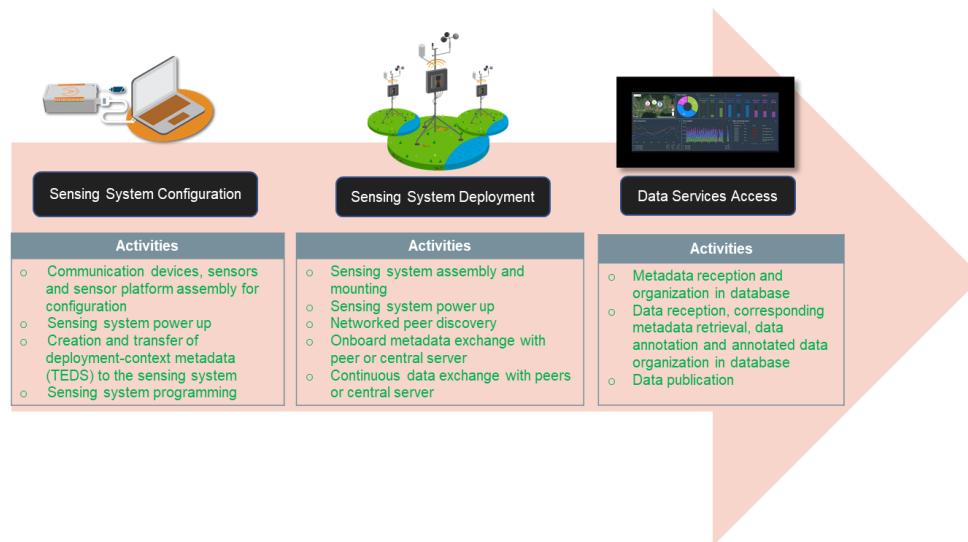
sensor data with corresponding metadata stored in an instance of the CUAHSI ODM hosted in a relational database server. A recent and generic software platform with a similar intention is the web-based “*ODM (version 2) Data Sharing Portal*” (Horsburgh et al., 2019) developed to support the collection and management of data from the Mayfly data logger in the Delaware River Watershed (Ensign et al., 2019). An instance of the portal is implemented to serve data collected by internet-connected low-cost electronics, including Arduino-based open-source electronics platforms, via Simple Object Access Protocol (SOAP) and/or Representational State Transfer (REST) Web Services. Another example with the same mechanics is the Cloud-Hosted Real-time Data Services for the Geosciences (CHORDS) project (Kerkez et al., 2016) intended to foster community discussion around the adoption of real-time data and the development of reference architecture for future implementations of real-time geosciences data systems. While these case studies adopt a sensor-to-server data-handling approach and represent a considerable step towards automation of data management, they still require manual configurations and organization of corresponding metadata to the ODM database and more importantly, the sensing platforms are not aware of any of their deployment-context metadata. This represents a serious issue that hampers inter-networks data analysis and reuse in real-world case studies such as Strachan and Daly (2017).

## 1.2. Contribution of This Paper

This paper contributes a) a new approach to the sensor-to-server data workflow implemented by environmental data acquisition systems in which the monitoring process (data generation, collection, storage, dissemination and usage; Zogheib et al., 2018) as well as network organization are implemented automatically with minimal human intervention and b) a small-scale and preliminary networked prototype system that implements the approach in a real-world deployment setting. The approach is a three-component process presented in Figure 1 with a

list of the different sequential activities carried out through each component. It promotes the development of a new generation of environmental data acquisition system deployable with deployment-context metadata stored onboard and can be updated and shared in a local network or over the internet as needed. Contrary to the existing approaches, the proposed one makes it mandatory to provide metadata upstream of the deployment stage. This is expected to play a significant role, for example, in tackling the two central issues (data interoperability and sustainability) in citizen science projects (Craglia and Granell, 2014; Wang et al., 2015) often associated to lack of standards and limited technical capacity (Buytaert et al., 2016; Paul et al., 2018). In addition, it guaranties that the metadata are preserved both in the front- and back-end of the monitoring process, laying the foundation for metadata synchronization between the front-end and back-end components provided that round-trip communication mechanisms are implemented. This feature is deemed critical as best practice for sensor networks and sensor data management by the Earth Science Information Partners (ESIP) EnviroSensing Cluster (ESIP EnviroSensing Cluster, 2016).

The case study that implements the approach comprises a chain of software and hardware components that collect environmental data and pipe it to a public or private repository where it is archived and made accessible via search and retrieval applications. To this end, the solution takes into account the need for semantic, schematic and syntactic interoperability both in a sensor network and sensor data management system. In doing so, the back-end demands on data management is incorporated, in the pre-deployment phase, in TEDS stored onboard of a custom-made sensor platform for automatic standard-based data storage, search and discovery purposes. The proposed approach, therefore, creates the foundation for the ability of the system to even keep track of sensor characteristics (accuracy, precision, range, etc.) being used by encoding the sensor characteristics in TEDS. This process, as it does not require (manual) organization of the metadata as in the SDL, proved to be an effective



**Figure 1.** High-level overview of the approach introduced in the paper.

tive way to make data management much less onerous both in terms of expertise and time requirements. The approach is equally applicable to large-scale and long-term deployment setups where each station works in an independent manner such as the Global Lake Ecological Observatory Network (Hanson et al., 2018), and the Trans-African Hydro-Meteorological Observatory (TAHMO; van de Giesen et al., 2014).

The solution presents the advantages of not only reducing the workload of the post-deployment phase, but also reducing human error in the data management system and has been effective in allowing data retrieval in user's preferred unit system via Web Services. To the best of the authors' knowledge, no existing environmental sensor monitoring system had a) featured the incorporation of metadata for data management purpose in the pre-deployment phase and b) supported automated standards-based data management and publication, although several have laid the groundwork for this (e.g., Jones et al., 2015; Sturtevant et al., 2016; Thorpe et al., 2016).

The paper is organized as follows: it first introduces the main criteria for the prototype system (software and hardware) design that implements the new approach; second, it presents the architecture of the prototype; subsequently, it discusses a preliminary real-world deployment case of the prototype system along with some results; and it concludes with a summary of the work along with a discussion of limitations of the system and our future work.

## 2. Requirements and System Design Criteria

The main objective of the research work was to develop the prototype of a field-deployable environmental monitoring unit that is capable of hosting its deployment-context metadata and automatically streaming both its metadata and collected data to its peers in a network or to a data management system. This latter must be able to automatically annotate the data with the corresponding metadata and save the result in a database. The data management system should ultimately output the data in both the Metric and English System of units and publish the results via web services with support for data analytics. Therefore, the scope of the system extends from the tools supporting the capture of the metadata to the development of a custom-made sensor platform with support for routing the captured metadata and collected data to the automated organization and publication of the data in standardized formats. Thus, the requirements for the prototype system include:

- a. The prototypes are cognitively and financially affordable. The sensor platform itself (not including the sensors) must be of the order of one to two hundred dollars as a variety of inexpensive (down to \$5 for a Raspberry Pi Zero or \$9 for a C.H.I.P microcomputer) computing platforms are now available.
- b. The prototypes, once configured for deployment, are capable of forming a local network (Base Node or Base Station) or joining a local network of peers (End Node or Router) using non-IP technologies making them suitable to work in remote areas.
- c. The prototypes, to form a network, must ideally be deployed in proximity to each other with line of sight. Here, the

term "proximity" is relative to the maximum distance the non-IP devices can transmit over. This distance or coverage is also constrained by the limitation on the cost of the sensor platform as more expensive wireless devices tend to be built with a larger coverage.

d. The software components must be portable to the extent possible to permit the testing of the reliability and robustness of the system using different computing platforms (e.g., Raspberry Pi, BeagleBone boards).

e. The system integrates a standard set of metadata along with controlled vocabularies to facilitate data publication, search, discovery, and interoperability from a web server through Application Programming Interfaces (APIs).

To meet these requirements, the major design criteria are outlined in the following sections.

### 2.1. Affordability and Ease of Deployment

The current commercially-available environmental sensing and information systems are often fraught with break-downs and omissions which results into an inevitable schism where on one end, a set of companies focus on the hardware for data collection which typically outputs a data file (e.g., Ochoa-Tocachi et al., 2018) that incurs the risk of being lost and on the other end, another set of companies or institutions focus on the software to standardize the data management and dissemination components using, for example, controlled vocabularies and data models. This breakup along the data path creates the conditions for inconsistencies and incompatibility regarding the choices of the semantics and syntaxes in the data files from each and every station in a same network. It also leads to heterogeneous data formats that are not compatible in their annotation extents using metadata which is a key impediment to data interoperability, integration and dissemination. These issues often result in total cost of ownership that is prohibitive due to the multidisciplinary technical expertise involved. These issues prompt the need for a new generation of affordable and easy-to-deploy environmental monitoring systems that bridge the various aspects of environmental data collection by deploying integrated solutions. Here, the term "easy-to-deploy" means an installation effort that requires minimal configuration steps both at the hardware level and at the web interface.

### 2.2. Cross-Platform and Reusable Software Components

The Python programming language was the language of choice for the rapid development of the built prototype for many reasons. First, it is open-source, thus, it offers the flexibility of developing (scientific) applications for private, public and commercial purposes without the need for extra permissions. Second, applications can be developed and deployed on multiple platforms (Windows, Linux, and Mac) and can interact with relational (SQLite, Microsoft SQL Server, PostgreSQL, and MySQL) and non-relational (e.g., MongoDB) Database Management Systems (DBMS). This versatility of Python to work with a diversity of database systems is important to the data management component of the system. Third, it supports the Object-

Oriented programming paradigm which we used to engineer the software applications. Fourth, it can be integrated with other programming languages such as C and C++ when higher processing performance is needed.

### 2.3. Software Quality and Modularity

Because the full set of requirements for the envisioned system was not known a priori, an iterative and incremental Test-Driven Development (TDD) approach along with an Object-Oriented approach was adopted. Using this strategy permitted developing reusable codes that sped up productivity. In particular, by automatically testing the software components, the next development steps were more successful compared to the partially-systematic functional testing practice that was used at the onset of the development.

Furthermore, time-tested solutions including Design Patterns (Gamma et al., 1995) was adopted to improve maintainability, modularity, scalability, inter-codes communication and reusability. For instance, the Observer, Command, Abstract Factory, and the Singleton patterns was implemented to improve the code development. Incidentally, Design Patterns played a more effective role in software engineering when used after the software system has been designed using TDD-based principles. For more on the software and hardware design experiences, the reader is invited to consult Celicourt et al. (2016).

### 2.4. Network Organization and Stations Authentication

The Base Node plays the role of the network coordinator and is, thus, responsible for executing a set of software-enabled procedures to identify each End Node attempting to join its network. These procedures include the transmission of several commands or requests for metadata to each End Node announcing itself to the Base Node. Thus, the implementation of a send-and-forget message exchange strategy to enable the Base station to simultaneously communicate with one or more nodes is critical. The advantage is that communication with the Base Station is not blocked and the latter does not have to wait for a reply to the last command sent before sending other commands to the same End Node or others. Ultimately, a candidate node can join the Base Station network and start collecting data if it has successfully replied to the received commands.

### 2.5. Data Discovery and Interoperability

Subsequent to deployment and activation of environmental monitoring stations, the data may need to be transferred to and stored on a server where not only it can be archived but also accessed by third parties. Often however, environmental data collection efforts overlook the critically important inclusion of a standards-based system for storing, managing, organizing, indexing, documenting and sharing sensor data. This is even more critical with the advent of the Internet of Things (IoT), a key component of the Future Internet Technologies for environmental applications (Granell et al., 2016), where data is being ubiquitously collected with a variety of heterogeneous devices and made publicly available via internet. Therefore, sup-

porting data discovery and interoperability through the provision of standards-based metadata to make collected data understandable to third-party users and track data provenance is a much-needed feature of the data management component of environmental monitoring systems. The data management and stewardship practices are beneficial to many different organizations (including private and public agencies, academics, etc.) and are being required by science funders and governmental agencies (Wilkinson et al., 2016). To support this criterion, the CUAHSI ODM, including its related controlled vocabularies, designed to provide a consistent format for the storage and retrieval of point environmental observations (Horsburgh et al., 2008) on the server side was adopted. The controlled vocabularies from the ODM standards are integrated as part of the TEDS or encoded metadata deployed on the edge devices. These data management aspects are introduced in more details in the next section.

## 3. Prototype Development and Testing

In line with the proposed approach, two types of hardware prototypes, each with a different suite of software components that enables their operation, were developed and described below. During the Sensing System Configuration process (Figure 1), the software packages are loaded to the corresponding prototype type (Base or End Node). Then, the necessary ancillary information (metadata) describing the overall deployment context including sensors and sensor platforms used are encoded and loaded onto the Node. Such information was encoded in the TEDS. However, an initial desktop tool named PyTEDS (Celicourt and Piasecki, 2015a), a Graphical User Interface (GUI) in Python that facilitates the entry and encoding of the information was developed. Upon creation, the TEDS are loaded into each End Node using a custom Secure Shell (SSH) client over Ethernet or WIFI from a laptop computer prior to deployment. This is an initial step towards the ultimate goal to deploy the CUAHSI ODM practices and standards on edge devices and enable them to output metadata-annotated data in standard formats like WaterML (OGC, 2012).

### 3.1. System Architecture

The Base Node features the ability to create a local network and gather, in a first step, metadata to identify or authenticate End Nodes and in a second step, data from authenticated End Nodes and organize the results in the ODM instance. The main software packages developed include the following:

a. Network Manager: The NM, linked to the communication device of the Base Station, is responsible for input/output exchanges between the Base Station and the rest of the network. It performs a preliminary processing of the incoming information and identifies the category of the message (TEDS or measurements/data or announcement). Depending on the status of an End Node (announcement received or not), it pushes the received message to the Message Manager or it auto-formulates requests/commands that are sent to the End Node. This feature is designed with concurrent communication capabilities and thus operates in such a way that the Base Station can initiate communication with a newly installed End Node

while it continues gathering and also processing data and meta-data for the rest of the network.

b. Network Interface: The NI, linked to the communication device of the End Node, is responsible for input/output exchanges between the End Node and its corresponding Base Station. It pre-processes incoming messages before they reach the Message Manager. This ensures that only messages from a known Base Station reached the Message Manager.

c. Message Manager: the MM plays the role of a dispatcher of messages (at the Base Station) or requests (at the End Node) received. At the Base Station, the MM routes the received message to either the TEDS Manager (if message contains TEDS) or the Data Manager (if the message contains measurements) following a pre-processing of the received message to determine its category (TEDS or measurements) and the status of the End Node (registered or not registered). Similarly, at the End Node, it routes it either to the TEDS Manager (if TEDS is requested) or to the Data manager (if the message is a confirmation message that the End Node was successful registered with the Base Station and it can start taking measurements). It keeps track of the End Node status (registered or not).

d. TEDS Manager: On the Base Station, the TM verifies the incoming TEDS from the Data Manager for integrity, then decodes them and extracts the relevant information (metadata). It then organizes the metadata as a JavaScript Object Notation (JSON) string and passes it onto the Data Loader. On the End Node, the TM retrieves the appropriate TEDS according to the requests received by the End Node and passes them onto the Message Manager.

e. Data Manager: The DM running on the Base Station decodes the received data/measurements from the NM. It then receives the corresponding metadata from the database instance using the Data Loader/Extractor to annotate the data. Furthermore, it organizes the annotated data as a JavaScript Object Notation (JSON) string and passes it onto the Data Loader/Extractor. On the End Node, the DM collects and encodes data from sensors and passes it onto the Message Manager.

f. Data Loader/Extractor: This application retrieves the fields/columns from the Django Application for the tables corresponding to the incoming data from the DM or metadata/TEDS from the TM. It also receives the content of the JSON string with the data that it subsequently saves into the database instance implemented using the SQLite database server.

g. WofPy (Texas Water Development Board, 2012): The data and accompanied metadata are made accessible by end-users' applications via the WofPy (WaterOneFlow Web Services in Python) open-source web services developed by the Texas Water Development Board (<http://www.twdb.texas.gov>). The Web Services support a variety of requests ranging from getting a simple description of the stations in the network to retrieval of time-window based data slices.

h. HydroUnits (Celicourt and Piasecki, 2015b): This is a tool developed specifically to perform unit transformation, time series conversion and dimensional analysis in hydrologic information systems. It creates contents of the *IEEE1451-dot0UnitsRepresentation* Table mentioned below. In addition,

it has been successfully integrated with WofPy's data-access object (DAO) software module to allow data retrieval in the user's preferred physical unit system.

i. Data Visualization: This is a customized data analytics application built on top of the Django Application using the Leaflet ([www.leafletjs.com](http://www.leafletjs.com)) and D3 ([www.d3js.org](http://www.d3js.org)) JavaScript frameworks that support online data visualization. This application extends the Template component (HTML pages with or without embedded JavaScript codes) of the Django Application.

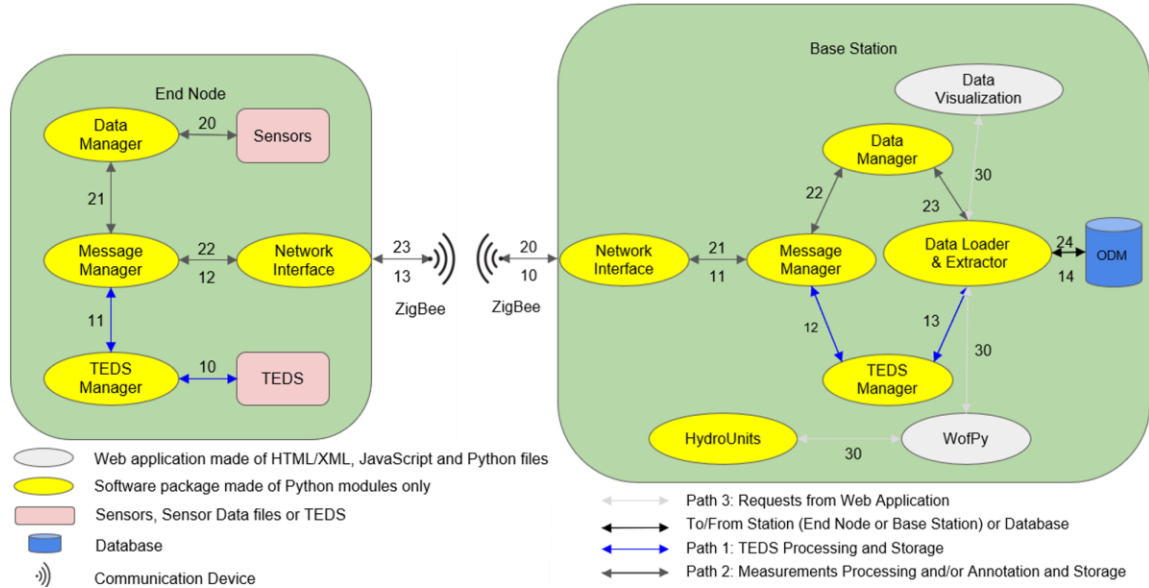
It must be noted that the software tools running on the Base Station are built around a Django (Holovaty and KaplanMoss, 2009) Web Application which plays a prominent role in the data storage, discovery and interoperability. The Django web framework follows a Model-View-Template design pattern with built-in Object-Relational Mapping capabilities and designed to support rapid development of web applications using Python. The Model consists in an augmented instance of the ODM built from the ODM SQL Queries translated to Python codes and comprises two extra tables:

- *IEEE1451dot0UnitsRepresentation* is used to store the IEEE 1451.0 representation of the units defined in the Units table of the original ODM. This table helped in identifying the unit for a sensor following the decoding of its TEDS.
- *Metadata Configuration* is a table that stores configuration information about each station (ID of the communication module, Sensor Channel number, for example) to assist the Base Station in identifying messages origin and annotation of the incoming data.

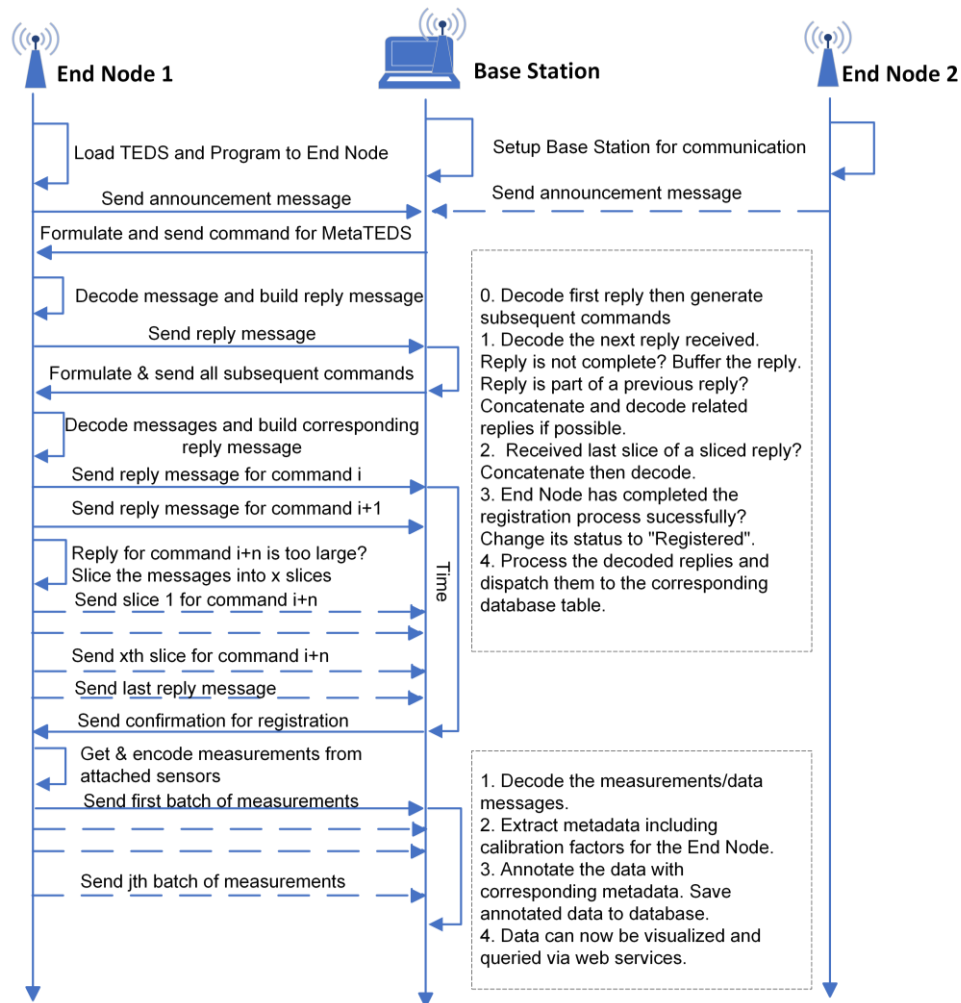
Once loaded with the necessary software packages and configurations, the End Nodes and the Base Station are ready to be deployed (the *Sensing System Deployment* process depicted in Figure 1 and the communication follows a two-stage pattern as described below and depicted in Figure 3). Here each stage is related to its corresponding multi-step process as presented in Figure 2. Here each step/link is labelled "KN" (e.g., 10, 24) in Figure 2 where K indicates the  $K^{\text{th}}$  stage and N indicates the  $N^{\text{th}}$  step information (TEDS and measurements) embedded in a message to be saved in the database, travels either on the End Node side or the Base Station side in the  $K^{\text{th}}$  stage to reach the database.

a. The first stage ( $K = 1$ ) of the process, called Authentication Stage, is detailed in the top dashed box in Figure 3. It starts with a first step where the End Node sends an announcement message to the Base Station (see top arrow from left to right in Figure 3) which, in a second step, sends a command/request for TEDS to the announcing End Node (see second arrow from right to left in Figure 3). This strategy has the benefit of reducing the energy consumption footprint of the Base Station which otherwise would be regularly scanning its coverage area to discover new End Nodes. In addition, a newly installed End Node has the advantage of joining the network and starts collecting data immediately as it does not have to idle and wait until the Base Station initiates contact. A third advantage of making the End Nodes event generators is that the computing overhead of the Base Station is reduced.

The first step of this stage corresponds in Figure 2 to Step



**Figure 2.** Architectural diagram depicting the components of the system prototype.



**Figure 3.** Schematic of messages processing sequences within the system during the deployment phase.



12 followed by Step 13 where the announcement message travels from the End Node Message Manager to its Network Interface which sends it out to the Base Station through the End Node communication device. Received by the communication device on the Base Station side, the announcement message followed Step 10 to reach the Network Manager. This latter auto-formulates a series of commands (requests for TEDS) that are sent back to the announcing End Node again in Step 10 of the Base Station. In turn, the Network Interface of the End Node passes the received request onto the Message Manager which then decodes it. The requested TEDS is searched and returned by the TEDS Manager to the Message Manager which forwards it to the Base Station. Thus, the TEDS embedded into a response-message travels from Steps 10 to 13 on the End Node side.

The response-message from the End Node arriving at the Base Station follows Steps 10 to 14 to reach the database. Upon receiving a message that contains a TEDS, the Message Manager of the Base Station routes the message to the TEDS Manager in Step 12 and the TEDS Manager itself decodes the TEDS, extracts and passes the relevant content in the form a JSON string to the Data Loader through Step 13. The content is finally stored in the ODM instance in Step 14.

The process continues as explained above and in a Ping-Pong pattern until the End Node successfully replies to all requests from the Base Station which subsequently registers the End Node for the second stage explained below and the Network Manager sends a registration confirmation message that follows the same path as a request for TEDS to the End Node.

b. The second stage ( $K = 2$ ), named Measuring Stage, is detailed in the bottom dashed box in Figure 3. During this stage, the End Node periodically forwards measurements from the attached sensors and associated timestamps using a coin-cell-battery-powered real-time clock (RTC) module to the Base Station following the reception of the confirmation message.

The measurements forwarding process starts with the End Node Message Manager instructing its Data Manager to start collecting periodic measurements from the sensors. In turn, the Data Manager encodes the measurements gathered from the sensors in Step 20 and sends them back to the Message Manager which in turn sends them out to the Base Station via the Network Manager and the communication device. Thus, a batch of measurements takes Steps 20 to 23 before it leaves the End Node to reach the Base Station.

Once the message with measurements has been received, the Network Manager of the Base Station determines the status of the sender before it is passed onto the Message Manager which in turn routes it to the Data Manager instead of the TEDS Manager because the sender is at this point registered with the Base Station network. The Data Manager package then decodes it and connects to the Data Loader/Extractor to get the corresponding metadata from the ODM instance. Upon obtaining the metadata, the Data Manager annotates the measurements and sends them back to the Data Loader in the form of a JSON string. The Data Loader then stores the content of the JSON string into the ODM instance. Thus, during the sec-

ond stage, a batch of measurements arriving at the Base Station travels from Steps 20 to 24 to be stored with its corresponding metadata in the ODM instance.

Once the annotated data is saved in the ODM instance, it is ready to be harvested by the WofPy and the Data Visualization packages in Step 30. HydroUnits aids WofPy in outputting the requested data in the user's preferred physical units.

In the hardware prototypes, the Digi International's Xbee Series 2 modules that implement the ZigBee protocol with a communication range of 400 feet were used to establish communication (see Figure 2). While this is admittedly somewhat limiting and thus less useful for many applications it worked well for our initial testing. For deployment settings that are more demanding in terms of communication range, more powerful communication devices, such as the Digi International's Digi XLR Pro unit that can transmit data over a distance of up to 100 miles, can effortlessly replace the Xbee Series 2. The Xbee Series 2 devices implement the ZigBee ([www.zigbee.org](http://www.zigbee.org)) communication protocol which has been used in various energy-constrained applications including healthcare (Yuce, 2010; Chang et al., 2011), food logistics (Jedermann et al., 2006), environmental monitoring (Ferdoush and Li, 2014; Somov et al., 2014) and energy system monitoring (Shariff et al., 2015). Although it is a mature and popular protocol (Baronti et al., 2007; Lee et al., 2007), an emerging and more promising alternative technology that becomes available to meet the need for long-range communication, low power and low data rate of our application is LoRa (Long Range; Bembe, 2019; Raza et al., 2017). LoRa devices implement serial communication interfaces such as Inter-Integrated Circuit (I2C), Serial Peripheral Interface (SPI), Universal Asynchronous Receiver-Transmitter (UART). Thus, the prototypes which also implement these interfaces (see Figure 4) can with minimal effort be converted to a LoRa end node and take the benefits of the emergence of low power wide area (LPWAN) technologies. Because the Python programming language was used, this conversion will further be facilitated by the availability of the official python library named PyLoRa (<https://pypi.org/project/pyLoRa/>) for communication with LoRa devices.

Even though the Xbee Series 2 devices (depicted with a wave signal emission in Figure 2) used can work in one of two modes (Application Programming Interface (API) vs Transparent Mode (AT)), for the purpose of this work, the Xbee modules were configured in API mode using specific software, XCTU (Digi International Inc., 2016). The API mode offers several advantages for our automated solution compared to the AT mode. Among the benefits of the API mode are: data management and transmission to multiple destinations, reception of success/failure status of each transmitted Radio Frequency packet, identification of the source address of each received packet. This last item is of great importance as it constitutes the key feature we used in the End Node announcement and discovery phase where the base station performs messages exchange with every End Node.

During a typical operation cycle, End Nodes sample the data and send them periodically to the Base Station. In order to reduce energy consumption, a mechanism to control the Sleep/



Wake cycle of the Xbee trying to put it into sleep mode for most of the sampling time interval was needed. To this end, the Xbee modules (End Nodes only) were configured for Pin Hibernation Mode during their configuration with the XCTU software which allows the nodes to automatically wake up the Xbee when data is available for transmission and put it back to sleep mode until the next data packet would be available for transmission; a technique called bit-banging. Unlike the End Nodes Xbee module configured as Slave, the Xbee module at the Base Station was configured as Coordinator and was not allowed to attain sleep mode so it can continuously listen for incoming packets including announcement messages from newly deployed End nodes.

### 3.2. Preliminary Field Testing

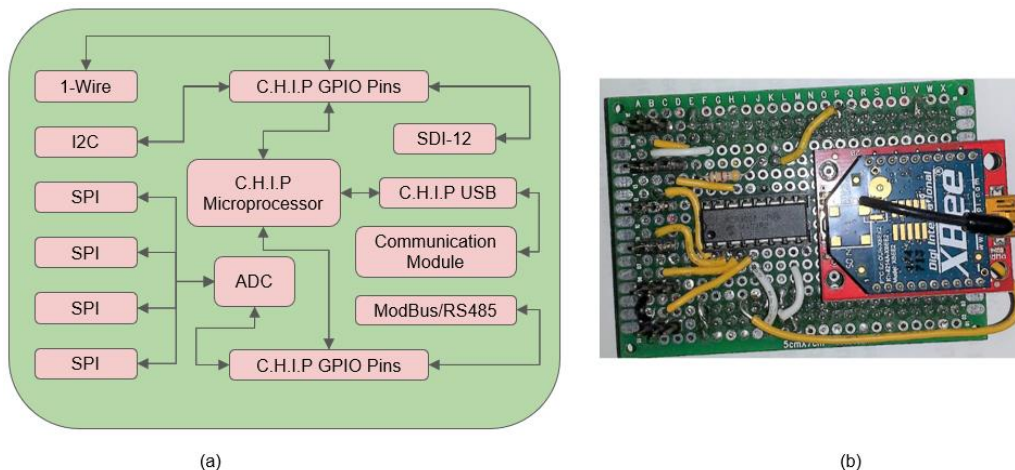
The system, more specifically the End Node hardware, as developed in an indoor environment has been intensively tested during the development phase in indoor environments. This initial testing steps helped avoiding hurdles that might have surfaced during field testing while not eliminating them completely. The system was also in an outdoor environment using an incremental and modular testing approach. At this point, the Raspberry Pi-based End Node hardware development has been discontinued and the efforts were redirected to the C.H.I.P microcomputer ([www.getchip.com](http://www.getchip.com)) developed by Next Thing Co. The decision was made based on limitations of the Raspberry Pi microcomputer (lack of support for the various serial configurations) and the existing desire to further minimize cost, footprint, and energy consumption of the hardware component. The transition to C.H.I.P allowed to test the portability of the software components.

#### 3.2.1. Custom-Made End Nodes Hardware Prototype Used in Field Testing

The hardware development effort culminated into the design of a compact, hand-soldered and double-sided circuit board (Figure 4b) mounted on the C.H.I.P microcomputer with inter-

faces for the most popular sensors communication protocols. The hardware component provides support for the following sensors physical interfaces depicted in Figure 4a: One-Wire, SPI, I2C, RS485/Modbus, USB, and Serial-Digital Interface at 1200 baud (SDI-12). For the latter, the logic and voltage levels on the serial data line at both ends (sensor and datalogger sides) of the interface have been tested using the bit-banging technique and visualized using an Oscilloscope. With these interfaces, the hardware component becomes more flexible in its ability to accommodate different types of sensors available on the market.

The total cost of hardware components used in the End Node prototype development is approximately USD 113 not including the cost of the sensors used. While a manufacturing-ready product could not be developed, the cost of the prototype demonstrates that a datalogging device with advanced capabilities can be developed at a cost that is about 10 times cheaper than current commercial dataloggers with similar performance such as dataloggers from Onset Computer Corporation. The battery used to power the End Node represents about 36% of the overall cost. Despite the relatively high cost of the prototype, there are several opportunities to reduce the cost. There are a few components (e.g., Xbee Shield and USB hub) that can be removed or replaced (e.g., USB-to-RS485 adapter) by custom-made components. For instance, the C.H.I.P has 4 UART ports that can be used for serial communication. Therefore, if the Max485 transceiver from Maxim Integrated ([www.maximintegrated.com](http://www.maximintegrated.com)) or the DS3985N transceiver from Texas Instruments ([www.ti.com](http://www.ti.com)) was used as the RS-485 interface on a UART port, the overall cost could be reduced by up to 5%. The cost can be further reduced by another 15% if the Xbee Shield is removed and the Xbee is mounted directly on the circuit board and connected to a UART port instead of the C.H.I.P USB block (Figure 5). An additional significant cost reduction can be achieved if many or all of the through-hole components used, which are more convenient to use during hand-soldered prototyping, are replaced by their surface-mounted equivalents which are generally more cost-effective. This strategy would reduce the overall footprint of the custom Printable Circuit Board built



**Figure 4.** (a) Block components of the circuit board and (b) the resulting custom-made circuit board mounted on the C.H.I.P.

and the weight and energy consumption of the End Node.

### 3.2.2. FAIR-Approach to the System Deployment

The system deployment process follows the three-step approach depicted in Figure 1. After assembling the End Nodes prototypes, as explained above (Section 3), they are loaded with the necessary software components to enable their operation in the field. As highlighted above, before the field deployment, the prototypes must be configured or loaded with the TEDS which contains the necessary deployment-context information. In addition, they must be programmed to control the sensors. This program is written also using the Python programming language becomes, when loaded to the End Node, part of the Data Manager component of the End Nodes in Figure 2.

The PyTEDS GUI introduced in Celicourt and Piasecki (2015a) imposed a heavy cognitive workload on the authors as initial users, therefore an extra effort was made to develop a more user-friendly version of it that insulates the user from the technical details related to, for example, sensor characteristics among others, of the original version. This tool was, therefore, converted into a more user-friendly desktop tool using the Electron framework (<https://www.electronjs.org/>) with support for Mac and Windows that simplifies the End Node configuration process and programming. Thus, the FAIR-oriented system deployment process works as follows (Figure 5):

a. A preliminary Sensor Information System in the form of Database Tables that host relevant characteristics of the sensors used in our experiments was developed. These characteristics include, for example, the variables measured by the sensors and their corresponding units that are Controlled Vocabularies from the original ODM. In addition, manufacturer and sensor model information was also stored to facilitate identification by the user when these information are displayed in the Desktop tool.

b. Based on the user's selections and inputs in the sequences of interfaces of the GUI, the TEDS generation engine generates the TEDS that include Controlled Vocabularies as part of the metadata. The TEDS are, then, transmitted to the End Node under configuration.

c. In addition, the code base was extended to support automated program generation using the Sensor Control Program Generation engine based on the choices made by the user regarding the sensors and the variables of interest from a sensor. This is important as a sensor such as the HydraProbe II supports more than one variable. The automated program generation process is made possible using the Abstract Syntax Trees (AST) concept in Python. The program generation is the final step to get the End Node ready for deployment. At this point, the End Nodes can communicate with the Base Node as demonstrated in Section 3.2.3 (below) and form networks to exchange their metadata (TEDS content) and their timestamped data.

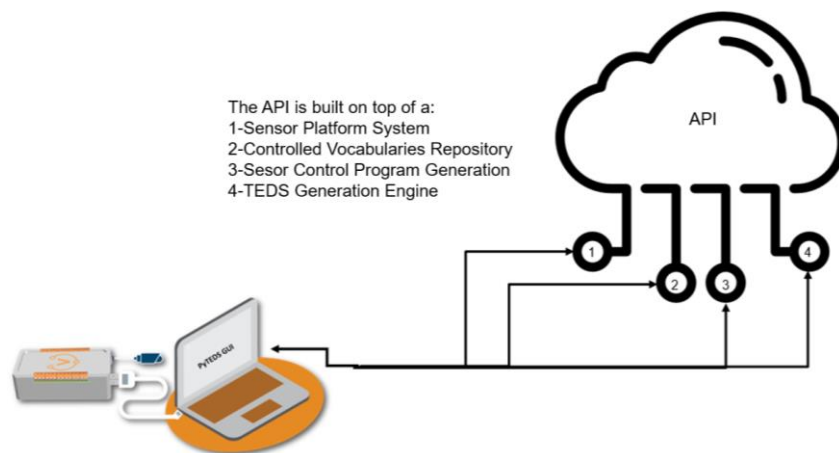
d. These tools were migrated to the Heroku (<https://www.heroku.com/>) cloud platform and therein, an API was built to permit the Desktop tool to place HTTP requests for the four main computing services (sensor metadata (1), Controlled Vocabularies (2), TEDS creation (3), sensor control program generation (4)) depicted in Figure 5. The objective here is to use this setup as a stepping stone to develop a multi-user system where the devices can be configured and the collected data and metadata can be made publicly available through high granularity filters (e.g., down to a specific variable).

e. The deployment in the cloud offered the opportunity to permit other users to experiment the introduced FAIR-approach without the real hardware. Accordingly, a virtual version of the hardware configuration process was developed and it is available for testing using the link: <https://bit.ly/pyteds>.

### 3.2.3. Field Testing Setup

Before conducting the field system testing, intermediate field tests were conducted to verify the range of the communication devices (Xbee Series 2 with 400 feet range), to ensure that the sensors used work correctly, and to see if the batteries used were able to effectively meet the power required by each End Node. Several tests were successfully completed with two End Nodes and a Base Station having the following physical configurations:

a. Base Station: Dell XPS 13 Laptop Computer and Xbee



**Figure 5.** API-based End Nodes configuration and programming processes.

Series 2 (Figure 6a).

b. End Node 1: Adafruit's 10200 mAh USB Battery Pack, Stevens' Hydra Probe II, Xbee Series 2 and the developed Data-logger (Figure 6b).

c. End Node 2: Adafruit's 2500 mAh lithiumion battery, Adafruit's waterproof DS18B20 digital temperature sensor, Xbee Series 2 and the developed Datalogger (Figure 6c). Each End Node (placed at about 44 feet from the Base Station) was programmed to collect data over a 1-minute time interval for a total duration of about 1 hour during which the data was transmitted immediately to the Base (Node) Station.

#### 3.2.4. Field Testing Results

The results, presented below in Figures 7 to 10, are for illustration of the end-to-end functionality of the system only. Figure 7 shows the two End Nodes displayed after being registered with the Base Station and the preliminary Data Analytics portal with selected Variables (Soil Temperature, Soil Water Electrical Conductivity and Volumetric Water Content) measured End Node 2 and the Air Temperature collected by End Node 1. Figure 7 also shows that the Air Temperature unit recorded the expected decrease in temperature in the afternoon. Figure 8 shows the modified REST-based Web Services portal for data retrieval using WoFPy. Furthermore, with the support of HydroUnits performing dimensional analysis, the data can also be retrieved in user's preferred equivalent unit. For instance, Figure 10 shows the converted time series in degree Celsius from the original time series in degree Kelvin presented in Figure 9.

### 4. Summary and Future Work

This paper documents initial efforts to develop the prototype of an affordable and easy-to-deploy environmental condition monitoring system in which sensing equipment deployed in the field automatically streams the collected data to a standards-based data management system. Additional software applications were developed to support data analytics and the publication of the collected data in the user's preferred units using Web Services. A custom-made hardware component to which a HydraProbe II soil moisture sensor from Stevens Water and the DS18B20 temperature sensor from Maxim Integrated are attached for field data acquisition was further developed. Finally, the system was tested in a real-world environment. The results demonstrate that it has the potential to not only make the deployment of hydro-meteorological stations much less onerous, but also to lay the foundation of cost effectiveness and thus affordability. However, the developed system in its current state suffers a variety of limitations compared to current in-use environmental data acquisition systems that we address below.

First and foremost, both proven (communication devices) and unproven (computing devices and accessories) technologies were combined to develop the prototypes. In addition, due to the limited resources available and the project started initially in an academic setting, a long-term deployment to assess the robustness and reliability of the system in comparison with

proven environmental monitoring equipment were not performed. Furthermore, the ability of the computing device (C.H.I.P) to communicate with a large number of sensors in addition to sampling under high frequency was not tested.

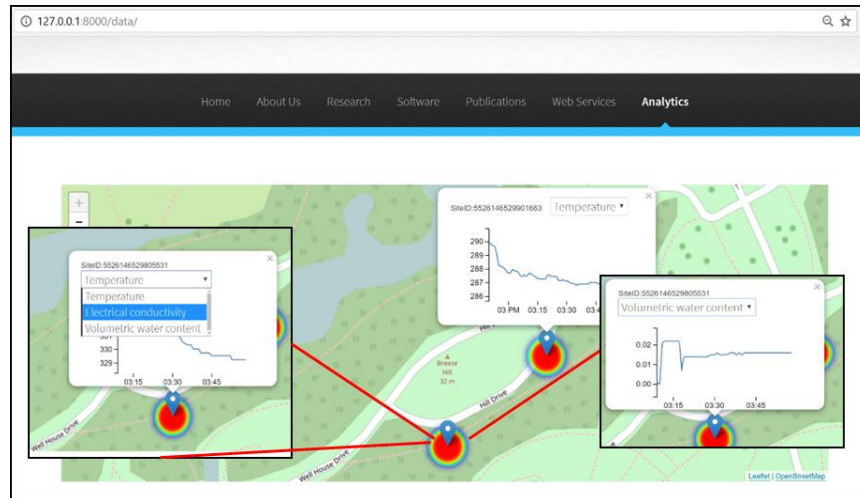
The adopted deployment topology was restricted to a point-to-multipoint with two End Nodes. This constituted a small network configuration that was not suitable to assess performance and scalability of the system of a much larger network. Furthermore, the capability of the prototype system to automatically inject the field data to the standards-based data management system was tested by deploying a local web server on a laptop in the field; this is somewhat of a simplification. However, the ability of a micro-computer such as the Raspberry Pi to serve as a network manager application was tested. Thus, the necessary software components can be deployed on a micro-computer that serves as a gateway to pipe the received information from its network to a more powerful server housed in a secure place running the software applications that were running on the laptop during the field experiments. This new deployment configuration will serve as a more general case of the setting presented in this paper. In addition, the performance of the data management system to process and respond to multiple parallel requests from external users including those who are not familiar with the system was not assessed.

The experiments were conducted with ZigBee-based communication devices only having a range of 400 feet with line-of-sight. However, it is believed that the system is capable of working with a variety of other communication protocols such as GPRS, Global System for Mobile communications (GSM) and Satellite links because the devices implementing these protocols often support a serial communication interface. Thus, the custom-made hardware developed should be able to work with a module implementing either of these protocols through a serial port. Consequently, software updates will be required to format the data and permit the data exchange between the hardware and the communication module. Last but not least, no temperature and relative humidity testing was performed with the custom-made hardware to evaluate its ability to withstand harsh environmental conditions and thus, its ability to work without supervision.

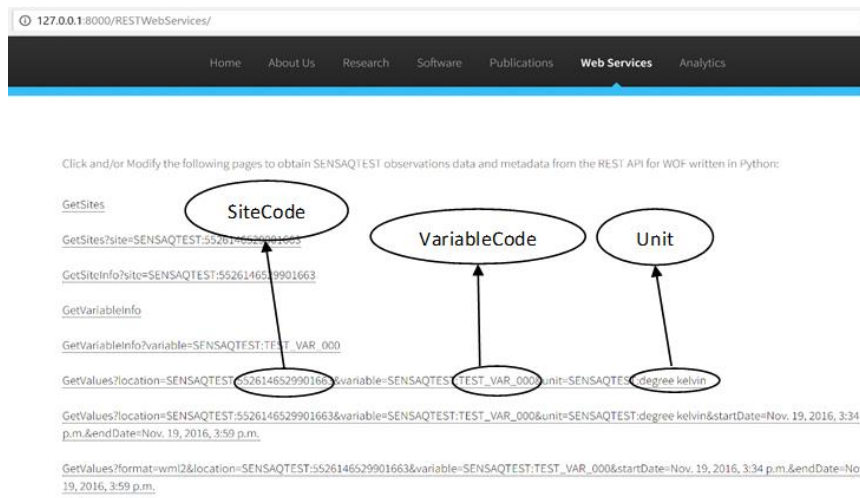
Future work aims at expanding the experiments to include a more general case where a gateway will play the role of the network coordinator and also routes the collected raw data and metadata (TEDS) to a remote server running the data management and publication applications. Furthermore, the usability and scalability of the overall system will be addressed. In addition, the future work plan also includes the evaluation of the robustness of the custom-made hardware to work in harsh environmental conditions and its ability to integrate with off-the-shelf sensors in a plug&play fashion. Beyond these objectives, the current system is envisioned to be transformed into a multi-user system that is deployed fully in the cloud to permit data sharing among users. It will provide functionalities for users to share their data with very high granularity, down to specific variable from an End Node. They will be able to search and integrate data made publicly available by other users of the plat-



**Figure 6.** (a) Base Station (Top) setup, (b) End Node 1 (Left) setup and (c) End Node 2 (Right) setup.

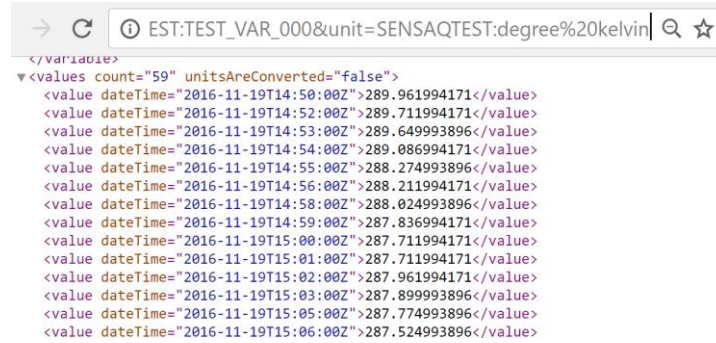


**Figure 7.** The preliminary data visualization interface of the system.



**Figure 8.** The WoFPy-based Web Services to retrieve the collected data.



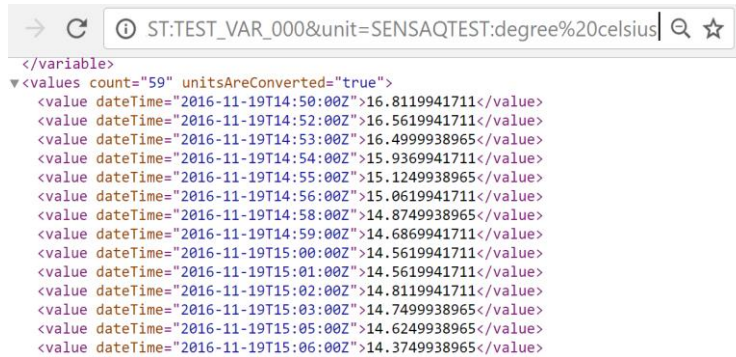


```

</variable>
{
  "values": [
    {
      "dateTime": "2016-11-19T14:50:00Z",
      "value": 289.961994171
    },
    {
      "dateTime": "2016-11-19T14:52:00Z",
      "value": 289.711994171
    },
    {
      "dateTime": "2016-11-19T14:53:00Z",
      "value": 289.649993896
    },
    {
      "dateTime": "2016-11-19T14:54:00Z",
      "value": 289.086994171
    },
    {
      "dateTime": "2016-11-19T14:55:00Z",
      "value": 288.274993896
    },
    {
      "dateTime": "2016-11-19T14:56:00Z",
      "value": 288.211994171
    },
    {
      "dateTime": "2016-11-19T14:58:00Z",
      "value": 288.024993896
    },
    {
      "dateTime": "2016-11-19T14:59:00Z",
      "value": 287.836994171
    },
    {
      "dateTime": "2016-11-19T15:00:00Z",
      "value": 287.711994171
    },
    {
      "dateTime": "2016-11-19T15:01:00Z",
      "value": 287.711994171
    },
    {
      "dateTime": "2016-11-19T15:02:00Z",
      "value": 287.961994171
    },
    {
      "dateTime": "2016-11-19T15:03:00Z",
      "value": 287.899993896
    },
    {
      "dateTime": "2016-11-19T15:05:00Z",
      "value": 287.774993896
    },
    {
      "dateTime": "2016-11-19T15:06:00Z",
      "value": 287.524993896
    }
  ]
}

```

Figure 9. Air temperature data collected by End Node 1 converted to degree Kelvin.



```

</variable>
{
  "values": [
    {
      "dateTime": "2016-11-19T14:50:00Z",
      "value": 16.811994171
    },
    {
      "dateTime": "2016-11-19T14:52:00Z",
      "value": 16.561994171
    },
    {
      "dateTime": "2016-11-19T14:53:00Z",
      "value": 16.499993896
    },
    {
      "dateTime": "2016-11-19T14:54:00Z",
      "value": 15.936994171
    },
    {
      "dateTime": "2016-11-19T14:55:00Z",
      "value": 15.124993896
    },
    {
      "dateTime": "2016-11-19T14:56:00Z",
      "value": 15.061994171
    },
    {
      "dateTime": "2016-11-19T14:58:00Z",
      "value": 14.874993896
    },
    {
      "dateTime": "2016-11-19T14:59:00Z",
      "value": 14.686994171
    },
    {
      "dateTime": "2016-11-19T15:00:00Z",
      "value": 14.561994171
    },
    {
      "dateTime": "2016-11-19T15:01:00Z",
      "value": 14.561994171
    },
    {
      "dateTime": "2016-11-19T15:02:00Z",
      "value": 14.811994171
    },
    {
      "dateTime": "2016-11-19T15:03:00Z",
      "value": 14.749993896
    },
    {
      "dateTime": "2016-11-19T15:05:00Z",
      "value": 14.624993896
    },
    {
      "dateTime": "2016-11-19T15:06:00Z",
      "value": 14.374993896
    }
  ]
}

```

Figure 10. Air temperature data collected by End Node 1 converted to degree Celsius.

form. This will require quite more effort to enable the End Nodes to, for example, automatically submit their metadata and data to the platform without using a “back-haul” system and the back-end system must dispatch and organize the received information to the corresponding user’s database.

It is hoped that the resulting system will find its applications in a variety of areas including meteorology, water resources assessment, environmental conditions studies (e.g., weather, air and water quality in citizen science projects), and water/waste-water utilities.

**Acknowledgments.** This work is supported by the Grove School of Engineering at The City College of New York, the City University of New York’s Environmental CrossRoads Initiative and a grant from the IEEE Foundation.

## References

- Ali, A.S., Zanzinger, Z., Debose, D., and Stephens, B. (2016). Open Source Building Science Sensors (OSBSS): A low-cost Arduino-based platform for long-term indoor environmental data collection. *Build Environ.*, 100, 114-126. <https://doi.org/10.1016/j.buildenv.2016.02.010>
- Austen, K. (2015). Environmental science: Pollution patrol. *Nat. News*, 517(7533), 136. <https://doi.org/10.1038/517136a>
- Baronti, P., Pillai, P., Chook, V., Chessa, S., Gotta, A. and Hu, Y. (2007). Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards. *Comput Commun.*, 30(7), 1655-1695. <http://dx.doi.org/10.1016/j.comcom.2006.12.020>
- Beutel, J., Buchli, B., Ferrari, F., Keller, M., Zimmerling, M. and Thiele, L. (2011). *X-Sense: Sensing in extreme environments*. In Design, Automation and Test in Europe Conference and Exhibition, 1-6. IEEE. <https://doi.org/10.1109/DATe.2011.5763236>
- Botts, M. (2014). *OGC® SensorML: Model and XML Encoding Standard*. Open Geospatial Consortium: Wayland, MA, USA.
- Botts, M., Percivall, G., Reed, C. and Davidson, J. (2008). *OGC® sensor web enablement: Overview and high-level architecture*. *GeoSensor networks* Springer Berlin Heidelberg, pp 175-190. [https://doi.org/10.1007/978-3-540-79996-2\\_10](https://doi.org/10.1007/978-3-540-79996-2_10)
- Buytaert, W., Zulkafli, Z., Grainger, S., Acosta, L., Alemie, T.C., Bastiaensen, J., De Bièvre, B., Bhusal, J., Clark, J., Dewulf, A. and Foggin, M. (2014). Citizen science in hydrology and water resources: opportunities for knowledge generation, ecosystem service management, and sustainable development. *Front. Earth Sci.*, 2(26). <https://doi.org/10.3389/feart.2014.00026>
- Buytaert, W., Dewulf, A., De Bièvre, B., Clark, J. and Hannah, D.M. (2016). Citizen science for water resources management: toward polycentric monitoring and governance? *J. Water Res. Pl. Manage.* 142(4). [https://doi.org/10.1061/\(ASCE\)WR.1943-5452.0000641](https://doi.org/10.1061/(ASCE)WR.1943-5452.0000641)
- Celicourt, P. and Piasecki, M. (2015a). An IEEE 1451.0-based Platform-Independent TEDS Creator using Open Source Software Components. *Int. J. Sens. Netw.* 3(1), 1-11. <https://doi.org/10.11648/j.ijssn.20150301.11>
- Celicourt, P. and Piasecki, M. (2015b). HydroUnits: Supporting Dimensional Analysis in Hydrologic Computing Systems using Sensor-based Standards. *J. Hydroinform.* 18(2), 168-184. <http://dx.doi.org/10.2166/hydro.2015.075>
- Celicourt, P., Sam, R. and Piasecki, M. (2016). Development of a Wireless Environmental Data Acquisition Prototype Adopting Agile Practices: An Experience Report, *J. Softw. Eng. Appl.*, 9, 479-490. <http://dx.doi.org/10.4236/jsea.2016.910031>
- Chang, W., Sung, T., Huang, H., Hsu, W., Kuo, C., Chang, J., Hou, Y., Lan, Y.C., Kuo, W.C., Lin, Y.Y. and Yang, Y.J. (2011). A smart medication system using wireless sensor network technologies. *Sens. Actuator A Phys.*, 172(1), 315-321. <http://dx.doi.org/10.1016/j.sna.2011.03.022>

- Colitti, W., Steenhaut, K. and De Caro, N. (2011). *Integrating wireless sensor networks with the web*. Extending the Internet to Low power and Lossy Networks.
- Craglia, M. and Granell, C. (2014). Citizen Science and smart cities-Report of summit, Ispra 5-7 February 2014, JRC Technical Report, [http://publications.jrc.ec.europa.eu/repository/bitstream/JRC90374/lbn\\_a26652enn.pdf](http://publications.jrc.ec.europa.eu/repository/bitstream/JRC90374/lbn_a26652enn.pdf), accessed June 1, 2019.
- Cressey, D. (2017). Age of the arduino. *Nat.*, 544(7648), 125-126. <https://doi.org/10.1038/544125a>
- Devaraju, A., Jirka, S., Kunkel, R. and Sorg, J. (2015). Q-SOS-A Sensor Observation Service for Accessing Quality Descriptions of Environmental Data. *ISPRS Int. J. Geoinf.*, 4(3), 1346-1365. <http://dx.doi.org/10.3390/ijgi4031346>
- Digi International Inc. (2016). XCTU Configuration and Test Utility Software: User Guide. DIGI. <http://www.digi.com/resources/documentation/digidocs/PDFs/90001458-13.pdf>
- Ensign, S., Arscott, D., Hicks, S., Aufdenkampe, A., Muenz, T., Jackson, J. and D. Bressler (2019). A digital Mayfly swarm is emerging, *Eos.*, 100, <https://doi.org/10.1029/2019EO116611>
- ESIP EnviroSensing Cluster (2016). Community Wiki Document on Best practices for sensor networks and sensor data management. Federation of Earth Science Information Partners. [http://wiki.esipfed.org/index.php/EnviroSensing\\_Cluster](http://wiki.esipfed.org/index.php/EnviroSensing_Cluster), 2016.
- Fang, X., Pomeroy, J., DeBeer, C., Harder, P., and Siemens, E. (2018): *Hydrometeorological data from Marmot Creek Research Basin, Canadian Rockies*, Federated Research Data Repository (FRDR), <https://doi.org/10.20383/101.09>
- Ferdoush, S. and Li, X. (2014). Wireless Sensor Network System Design Using Raspberry Pi and Arduino for Environmental Monitoring Applications. *Procedia Comput. Sci.*, 34, 103-110. <http://dx.doi.org/10.1016/j.procs.2014.07.059>
- Fernandes, M., Matos, S., Peres, E., Cunha, C., Lopez, J., Ferreira, P. Reis, M. and Morais, R. (2013). A framework for wireless sensor networks management for precision viticulture and agriculture based on IEEE 1451 standard. *Comput. Electron. Agr.*, 95, 19-30. <http://dx.doi.org/10.1016/j.compag.2013.04.001>
- Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
- Granell, C., Havlik, D., Schade, S., Sabeur, Z., Delaney, C., Pielorz, J. and Havlik, F. (2016). Future Internet technologies for environmental applications. *Environ. Modell. Softw.*, 78, 1-15. <https://doi.org/10.1016/j.envsoft.2015.12.015>
- Gray, J., Banhazi, T.M. and Kist, A.A. (2017). Wireless data management system for environmental monitoring in livestock buildings. *Info. Process. Agr.*, 4(1), 1-17. <https://doi.org/10.1016/j.inpa.2016.12.001>
- Hannah, D.M., Demuth, S., van Lanen, H.A., Looser, U., Prudhomme, C., Rees, G., Stahl, K. and Tallaksen, L. M. (2011). Large-scale river flow archives: importance, current status and future needs. *Hydrol. Process.*, 25(7), 1191-1200. <https://doi.org/10.1002/hyp.7794>
- Hanson, P.C., Weathers, K.C., Dugan, H.A. and Gries, C. (2018). The Global Lake Ecological Observatory Network. *Ecol. Inform.*, 415-433. [https://doi.org/10.1007/978-3-319-59928-1\\_19](https://doi.org/10.1007/978-3-319-59928-1_19)
- Healey, N.C., Oberbauer, S.F., Ahrends, H.E., Dierick, D., Welker, J.M., Leffler, A.J. and Tweedie, C.E. (2014). A Mobile Instrumented Sensor Platform for Long-Term Terrestrial Ecosystem Analysis: An Example Application in an Arctic Tundra Ecosystem. *J. Environ. Inform.*, 24(1). <https://doi.org/10.3808/jei.201400278>
- Hirafuji, M., Yoichi, H., Kiura, T., Matsumoto, K., Fukatsu, T., Tanaka, K., Shibuya, Y., Itoh, A., Nesumi, H., Hoshi, N. Seisi, N.N., Adinarayana, J., Sudharsan, D., Saito, Y., Kobayashi, K. and Suzuki, T. (2011). Creating high-performance/low-cost ambient sensor cloud system using openfs (open field server) for high-throughput phenotyping. *IEEE. Proc. of the 2011 SICE Annual Conference (SICE)*, Tokyo, 2090- 2092.
- Holovaty, A. and Kaplan-Moss, J. (2009). *The definitive guide to Django: Web development done right*. Apress. <https://doi.org/10.1007/978-1-4302-1937-8>
- Horsburgh, J.S., Caraballo, J., Ramírez, M., Aufdenkampe, A.K., Arscott, D.B. and Damiano, S.G. (2019). Low-cost, open-source, and low-power: but what to do with the data? *Front. Earth Sci.*, 7(67). <https://doi.org/10.3389/feart.2019.00067>
- Horsburgh, J., Tarboton, D., Piasecki, M., Maidment, D., Zaslavsky, I., Valentine, D. and Whitenack, T. (2009). An integrated system for publishing environmental observations data. *Environ. Modell. Softw.*, 24(8), 879-888. <http://dx.doi.org/10.1016/j.envsoft.2009.01.002>
- Horsburgh, J.S. Tarboton, D.G., Maidment, D.R. and Zaslavsky, I. (2008). A relational model for environmental and water resources data. *Water Resour. Res.*, 44 (2008). <https://doi.org/10.1029/2007WR006392>
- Hund, S.V., Johnson, M.S. and Keddle, T. (2016). Developing a hydrologic monitoring network in data-scarce regions using open-source Arduino dataloggers. *Agri. Environ.Lett.*, 1(1). <https://doi.org/10.2134/aer2016.02.0011>
- IEEE (2007). *IEEE Standard for a Smart Transducer Interface for Sensors and Actuators-Common Functions, Communication Protocols, and Transducer Electronic Data Sheet (TEDS) Formats*, IEEE Standard 1451.0-2007.
- Jedermann, R., Behrens, C., Westphal, D. and Lang, W. (2006). Applying autonomous sensor systems in logistics-Combining sensor networks, RFIDs and software agents. *Sens. Actuator A Phys.*, 132(1), 370-375. <http://dx.doi.org/10.1016/j.sna.2006.02.008>
- Jiang, S., Babovic, V., Zheng, Y. and Xiong, J. (2019). Advancing opportunistic sensing in hydrology: A novel approach to measuring rainfall with ordinary surveillance cameras. *Water Resour. Res.*, 55(4), 3004-3027. <https://doi.org/10.1029/2018WR024480>
- Jiang, Q., Kresin, F., Bregt, A.K., Kooistra, L., Pareschi, E., Van Putten, E., Volten, H. and Wesseling, J. (2016). Citizen sensing for improved urban environmental monitoring. *J. Sensors*, <https://doi.org/10.1155/2016/5656245>
- Jones, A.S., Horsburgh, J.S., Reeder, S.L., Ramírez, M. and Caraballo, J. (2015). A data management and publication workflow for a large-scale, heterogeneous sensor network. *Environ. Monit. Assess.*, 187(6), 348. <https://doi.org/10.1007/s10661-015-4594-3>
- Kerkez, B., Daniels, M., Graves, S., Chandrasekar, V., Keiser, K., Martin, C., Dye, M., Maskey, M. and Vernon, F. (2016). Cloud Hosted Real-time Data Services for the Geosciences (CHORDS). *Geosci. Data J.*, 3(1), 4-8. <https://doi.org/10.1002/gdj3.36>
- Kumar, P., Morawska, L., Martani, C., Biskos, G., Neophytou, M., Di Sabatino, S., Bell, M., Norford, L. and Britter, R. (2015). The rise of low-cost sensing for managing air pollution in cities. *Environ. Int.*, 75, 199-205. <https://doi.org/10.1016/j.envint.2014.11.019>
- Lee, J.S., Su, Y.W. and Shen, C.C. (2007). A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi. In *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE* 46-51. IEEE. <https://doi.org/10.1109/IECON.2007.4460126>
- Lee, K. (2000). IEEE 1451: A standard in support of smart transducer networking. In *Instrumentation and Measurement Technology Conference, 2000. IMTC 2000. Proceedings of the 17th IEEE* 2, 525-528. IEEE.
- Lettenmaier, D.P. (2017). Observational breakthroughs lead the way to improved hydrological predictions, *Water Resour. Res.*, 53, 2591-2597. <https://doi.org/10.1002/2017WR020896>
- Li, X., Liu, Q., Yang, R., Wen, J., Zhang, J., Cai, E. and Zhang, H. (2016). The combination of ground-sensing network and satellite remote sensing in Huailai county. *IEEE Sens. J.*, 16(10), 3819-3826. <https://doi.org/10.1109/JSEN.2016.2535350>
- Liao, Y., Mollineaux, M., Hsu, R., Bartlett, R., Singla, A., Raja, A., Bajwa, R. and Rajagopal, R. (2014). Snowfort: An open source wireless sensor network for data analytics in infrastructure and environmental monitoring. *IEEE Sens. J.*, 14(12), 4253-4263. <https://doi.org/10.1109/JSEN.2014.2358253>
- Little, K.E., Hayashi, M. and Liang, S. (2016). Community-based ground-water monitoring network using a citizen-science approach. *Groundw.*, 54(3), 317-324. <https://doi.org/10.1111/gwat.12336>
- Longman, R.J., Giambelluca, T.W., Nullet, M.A., Frazier, A.G., Kodama,

- K., Crausbay, S.D., Krushelnycky, P.D., Cordell, S., Clark, M.P., Newman, A.J. and Arnold, J.R. (2018). Compilation of climate data from heterogeneous networks across the Hawaiian Islands. *Sci. data*, 5, 180012. <https://doi.org/10.1038/sdata.2018.12>
- Ochoa-Tocachi, B.F., Buytaert, W., Antiporta, J., Acosta, L., Bardales, J.D., Céleri, R., Crespo P., Fuentes, P., Gil-Ríos, J., Gualpa, M., Llerena C., Olaya D., Pardo P., Rojas G., Villacís, M., Villazón, M., Viñas P. and Bièvre, B. (2018). High-resolution hydrometeorological data from a network of headwater catchments in the tropical Andes. *Sci. data*, 5, 180080. <https://doi.org/10.1038/sdata.2018.80>
- OGC (2012). Open Geospatial Consortium WaterML 2.0 Part 1-Timeseries. Document # 10-126r4. Open Geospatial Consortium Implementation Standard. <http://www.opengeospatial.org/standards/waterml> (accessed February, 2020)
- Parra, L., Sendra, S., Lloret, J. and Rodrigues, J.J. (2017). Design and deployment of a smart system for data gathering in aquaculture tanks using wireless sensor networks. *Int. J. Commun. Syst.*, 30(16), e3335. <https://doi.org/10.1002/dac.3335>
- Paul, J.D. and Buytaert, W. (2018). Citizen science and low-cost sensors for integrated water resources management. *Adv. Chem. Pollu. Environ. Manage. Prot.*, 3, 1-34. <https://doi.org/10.1016/bs.apmp.2018.07.001>
- Rasouli, K., Pomeroy, J.W., Janowicz, J.R., Williams, T.J. and Carey, S.K. (2018). *Hydrometeorological data collected at Wolf Creek Research Basin, Yukon Territory, Canada over 1993-2014*, Federated Research Data Repository, <https://doi.org/10.20383/101.0113>
- Schneider, P., Castell, N., Vogt, M., Dauge, F.R., Lahoz, W.A. and Bartonova, A. (2017). Mapping urban air quality in near real-time using observations from low-cost sensors and model information. *Environ. Int.*, 106, 234-247. <https://doi.org/10.1016/j.envint.2017.05.005>
- Shariff, F., Rahim, N. and Hew, W. (2015). Zigbee-based data acquisition system for online monitoring of grid-connected photovoltaic system. *Expert. Syst. Appl.*, 42(3), 1730-1742. <http://dx.doi.org/10.1016/j.eswa.2014.10.007>
- Solomatine, D., Mazzoleni, M., Alfonso, L. and Chacon Hurtado, J.C. (2017). *Towards socio-hydroinformatics: optimal design and integration of citizen-based information in water-system models*. In EGU General Assembly Conference Abstracts (Vol. 19, p. 12370).
- Somov, A., Baranov, A. and Spirjak, D. (2014). A wireless sensor-actuator system for hazardous gases detection and control. *Sens. Actuator A Phys.*, 210, 157-164. <http://dx.doi.org/10.1016/j.sna.2014.02.025>
- Song, E.Y. and Lee, K.B. (2008). STWS: A unified web service for IEEE 1451 smart transducers. *IEEE Trans. Instrum. Meas.*, 57(8), 1749-1756. <https://doi.org/10.1109/TIM.2008.925732>
- Spence, C. and Hedstrom, N. (2018) *Baker Creek Research Catchment Hydrometeorological and Hydrological Data*, <https://doi.org/10.20383/101.026>, 2018
- Strachan, S. and Daly, C. (2017). Testing the daily PRISM air temperature model on semiarid mountain slopes. *J. Geophys. Res. Atmos.*, 122(11), 5697-5715. <https://doi.org/10.1002/2016JD025920>
- Stroud Research Center (2016). Critical Zone Observatory (CZO) mid-Stream Manual. Avondale, PA, USA: Stroud Research Center. [https://github.com/StroudCenter/midStream/blob/master/Stroud\\_CZO\\_midStream\\_Manual.docx](https://github.com/StroudCenter/midStream/blob/master/Stroud_CZO_midStream_Manual.docx). Accessed October 31, 2016.
- Sturtevant, C., Hackley, S., Meehan, T., Roberti, J.A., Holling, G. and Bonarrigo, S. (2016). *From field notes to data portal-An operational QA/QC framework for tower networks*. AGU Fall Meeting Abstracts, abstract #B41B-0402.
- Tauro, F., Selker, J., Van De Giesen, N., Abrate, T., Uijlenhoet, R., Porfiri, M., Manfreda, S., Caylor, K., Moramarco, T., Benveniste, J. and Ciruolo, G. (2018). Measurements and observations in the XXI century (MOXXI): Innovation and multi-disciplinarity to sense the hydrological cycle. *Hydrol. Sci. J.*, 63(2), 169-196. <https://doi.org/10.1080/02626667.2017.1420191>
- Tenopir, C., Allard, S., Douglass, K., Aydinoglu, A.U., Wu, L., Read, E., Manoff, M. and Frame, M. (2011). Data sharing by scientists: practices and perceptions. *PLoS One*, 6, e21101. <http://dx.doi.org/10.1371/journal.pone.0021101>
- Texas Water Development Board. WOFpy - a python wrapper for Water-OneFlow services. <https://pythonhosted.org/WOFpy/> (accessed March 3, 2015)
- Thorpe, A.S., Barnett, D.T., Elmendorf, S.C., Hinckley, E.L.S., Hoekman, D., Jones, K.D., LeVan, K.E., Meier, C.L., Stanish, L.F. and Thibault, K.M. (2016). Introduction to the sampling designs of the national ecological observatory network terrestrial observation system. *Ecosphere*, 7(12). <https://doi.org/10.1002/ecs2.1627>
- Turner, B., Hill, D.J., Carlyle-Moses, D.E. and Rahman, M. (2019). Low-cost, high-resolution stemflow sensing. *J. Hydrol.*, 570, 62-68. <https://doi.org/10.1016/j.jhydrol.2018.12.072>
- Van de Giesen, N., Hut, R. and Selker, J. (2014). The Trans - African Hydro-Meteorological Observatory (TAHMO). *Wiley Interdisciplinary Reviews: Water*, 1(4), 341-348. <https://doi.org/10.1002/wat2.1034>
- Villalonga, C., Bauer, M., Aguilar, F.L., Huang, V.A. and Strobbach, M. (2010). *A resource model for the real-world internet*. In: Lukowicz P., Kunze K., Kortuem G. (eds) Smart Sensing and Context. EuroSSC 2010. Lecture Notes in Computer Science, vol 6446. Springer, Berlin, Heidelberg. [https://doi.org/10.1007/978-3-642-16982-3\\_13](https://doi.org/10.1007/978-3-642-16982-3_13)
- Vogel, R.M., Lall, U., Cai, X., Rajagopalan, B., Weiskel, P.K., Hooper, R.P. and Matalas, N.C. (2015). Hydrology: The interdisciplinary science of water. *Water Resour. Res.*, 51(6), 4409-4430. <https://doi.org/10.1002/2015WR017049>
- Walker, D., Forsythe, N., Parkin, G. and Gowing, J. (2016). Filling the observational void: Scientific value and quantitative validation of hydrometeorological data from a community-based monitoring programme. *J. Hydro.*, 538, 713-725. <https://doi.org/10.1016/j.jhydrol.2016.04.062>
- Wang Y., Kaplan N., Newman G., and Scarpino R. (2015) CitSci.org: A New Model for Managing, Documenting, and Sharing Citizen Science Data. *PLoS Biol.*, 13(10), e1002280. <https://doi.org/10.1371/journal.pbio.1002280>
- Whitfield, P.H., Burn, D.H., Hannaford, J., Higgins, H., Hodgkins, G.A., Marsh, T. and Looser, U. (2012). Reference hydrologic networks I. The status and potential future directions of national reference hydrologic networks for detecting trends. *Hydro. Sci. J.*, 57(8), 1562-1579. <https://doi.org/10.1080/02626667.2012.728706>
- Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A. and Bouwman, J. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Sci. data*, 3. <https://doi.org/10.1038/sdata.2016.18>
- Wong, B.P. and Kerkez, B. (2016). Real-time environmental sensor data: An application to water quality using web services, *Environ. Model. Softw.*, 84, 505-517. <http://dx.doi.org/10.1016/j.envsoft.2016.07.020>
- Yang, J., Zhang, C., Li, X., Huang, Y., Fu, S. and Acevedo, M. (2009). Integration of wireless sensor networks in environmental monitoring cyber infrastructure. *Wirel.*, 16(4), 1091-1108. <http://dx.doi.org/10.1007/s11276-009-0190-1>
- Yuce, M. (2010). Implementation of wireless body area networks for healthcare systems. *Sens. Actuator A Phys.*, 162(1), 116-129. <http://dx.doi.org/10.1016/j.sna.2010.06.004>
- Zogheib, C., Ochoa-Tocachi, B.F., Paul, J.D., Hannah, D.M., Clark, J. and Buytaert, W. (2018). Exploring a water data, evidence, and governance theory. *Water Secur.*, 4, 19-25. <https://doi.org/10.1016/j.wasec.2018.11.004>