Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications

Mihai T. Lazarescu
Dipartimento di Elettronica, Politecnico di Torino
mihai.lazarescu@polito.it

***Abstract*—The Internet of Things (IoT) provides a virtual view, via the Internet Protocol, to a huge variety of real life objects, ranging from a car, to a teacup, to a building, to trees in a forest. Its appeal is the ubiquitous generalized access to the status and location of any "thing" we may be interested in.**

Wireless sensor networks (WSN) are well suited for long-term environmental data acquisition for IoT representation. This paper presents the functional design and implementation of a complete WSN platform that can be used for a range of long-term environmental monitoring IoT applications. The application requirements for low cost, high number of sensors, fast deployment, long lifetime, low maintenance, and high quality of service are considered in the specification and design of the platform and of all its components. Low-effort platform reuse is also considered starting from the specifications and at all design levels for a wide array of related monitoring applications.

I. Introduction

More than a decade ago, the Internet of Things (IoT) paradigm was coined in which computers were able to access data about objects and environment without human interaction. It was aimed to complement human-entered data that was seen as a limiting factor to acquisition accuracy, pervasiveness and cost.

Two technologies were traditionally considered key enablers for the IoT paradigm: the radio-frequency identification (RFID) and the wireless sensor networks (WSN). While the former is well established for low-cost identification and tracking, WSNs bring IoT applications richer capabilities for both sensing and actuation. In fact, WSN solutions already cover a very broad range of applications, and research and technology advances continuously expand their application field. This trend also increases their use in IoT applications for versatile low-cost data acquisition and actuation.

However, the sheer diversity of WSN applications makes increasingly difficult to define "typical" requirements for their hardware and software [1]. In fact, the generic WSN components often need to be adapted to specific application requirements and environmental conditions [2]. These ad hoc changes tend to adversely impact the overall solution complexity, cost, reliability, and maintenance that in turn effectively curtail WSN adoption, including their use in IoT applications [3].

To address these issues, the reusable WSN platforms receive a growing interest. These platforms are typically optimized by leveraging knowledge of the target class of applications (e.g., domain, WSN devices, phenomena of interest) to improve key WSN application parameters, such as cost, productivity, reliability, interoperability, maintenance.

Among the IoT application domains, the environmental/earth monitoring receives a growing interest as environmental technology becomes a key field of sustainable growth worldwide. Of these, the open nature environmental monitoring is especially challenging because of, e.g., the typically harsh operating conditions and difficulty and cost of physical access to the field for deployment and maintenance.

The generic WSN platforms can be used with good results in a broad class of IoT environmental monitoring applications. However, many IoT applications (e.g., those in open nature) may have stringent requirements, such as very low cost, large number of nodes, long unattended service time, ease of deployment, low maintenance, which make these generic WSN platforms less suited.

This paper presents the application requirements, the exploration of possible solutions, and the practical realization of a full-custom, reusable WSN platform suitable for use in low-cost long-term IoT environmental monitoring applications. For a consistent design, the main application requirements for low-cost, fast-deployment of large number of sensors, and reliable and long unattended service are considered at all design levels. Various trade-offs between platform features and specifications are identified, analyzed, and used to guide the design decisions. The development methodology presented can be reused for platform design for other application domains, or evolutions of this platform.

Also, the platform requirements of flexibility and reusability for a broad range of related applications was considered from the start. A real-life application, representative for this application domain, was selected and used as reference throughout the design process. Finally, the experimental results show that the platform implementation satisfies the specifications.

The rest of the paper is organized as follows. Section II reviews published works addressing similar topics. Section III defines a comprehensive specification set for WSNs for IoT environmental monitoring applications. Section IV summarizes the structure and the main functions of a WSN platform for these applications. Section V details the specifications and design solutions for the WSN nodes and field deployment devices. Section VI presents the practical realization and operation of the WSN nodes and the field deployment devices. Section VII presents the application server design, structure,

and operation. Section VIII presents an effective sensor node field deployment procedure. Section IX concludes the paper.

II. Related Work

WSN environmental monitoring includes both indoor and outdoor applications. The later can fall in the city deployment category (e.g., for traffic, lighting or pollution monitoring) or the open nature category (e.g., chemical hazard, earthquake and flooding detection, volcano and habitat monitoring, weather forecasting, precision agriculture). The reliability of any outdoor deployment can be challenged by extreme climatic conditions, but for the open nature the maintenance can be also very difficult and costly.

These considerations make the open nature one of the toughest application fields for large scale WSN environmental monitoring, and the IoT applications requirements for low cost, high service availability and low maintenance further increase their design challenges.

To be cost-effective, the sensor nodes often operate on very restricted energy reserves. Premature energy depletion can severely limit the network service [4]–[7] and needs to be addressed considering the IoT application requirements for cost, deployment, maintenance, and service availability. These become even more important for monitoring applications in extreme climatic environments, such as glaciers, permafrosts or volcanoes [2], [8]–[12]. The understanding of such environments can considerably benefit from continuous long-term monitoring, but their conditions emphasize the issues of node energy management, mechanical and communication hardening, size, weight, and deployment procedures.

Open nature deployments [13]–[17] and communication protocol developments and experiments [7], [18] show that WSN optimization for reliable operation is time-consuming and costly. It hardly satisfies the IoT applications requirements for long-term, low-cost and reliable service, unless reusable hardware and software platforms [19]–[24] are available, including flexible Internet-enabled servers [25]–[27] to collect and process the field data for IoT applications.

This paper contributions of interest for researchers in the WSN field can be summarized as: 1) detailed specifications for a demanding WSN application for long-term environmental monitoring that can be used to analyze the optimality of novel WSN solutions, 2) specifications, design considerations, and experimental results for platform components that suit the typical IoT application requirements of low cost, high reliability, and long service time, 3) specifications and design considerations for platform reusability for a wide range of distributed event-based environmental monitoring applications, and 4) a fast and configuration-free field deployment procedure suitable for large scale IoT application deployments.

III. IoT Environmental Monitoring Requirements

WSN data acquisition for IoT environmental monitoring applications is challenging, especially for open nature fields. These may require large sensor numbers, low cost, high reliability, and long maintenance-free operation. At the same time, the nodes can be exposed to variable and extreme climatic conditions, the deployment field may be costly and difficult to reach, and the field devices weight, size, and ruggedness can matter, e.g., if they are transported in backpacks.

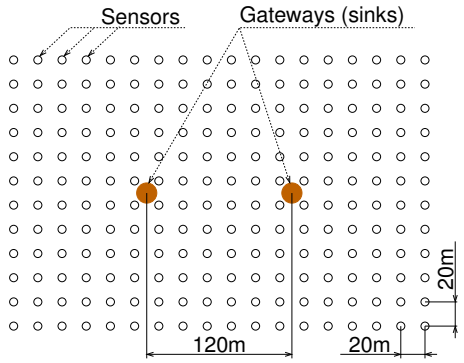


Fig. 1: Example of an ideal WSN deployment for in situ wildfire detection applications.

Most of these requirements and conditions can be found in the well-known application of wildfire monitoring using in situ distributed temperature sensors and on-board data processing. In its simplest event-driven form, each sensor node performs periodic measurements of the surrounding air temperature and sends alerts to surveillance personnel if they exceed a threshold. Fig. 1 shows a typical deployment pattern of the sensor nodes that achieves a good field coverage [28]. For a fast response time, the coverage of even small areas requires a large number of sensor nodes, making this application representative for cost, networking and deployment issues of the event-driven high-density IoT application class.

In the simplest star topology, the sensor nodes connect directly to the gateways, and each gateway autonomously connects to the server. Ideally, the field deployment procedure ensures that each sensor node is received by more than one gateway to avoid single points of failure of the network.

This application can be part of all three WSN categories [20]: event-driven (as we have seen), time-driven (e.g., if the sensor nodes periodically send the air temperature) and query-driven (e.g., if the current temperature can be requested by the operator). This means that the infrastructure that supports the operation of this application can be reused for a wide class of similar long-term environmental monitoring applications like:

- *water level* for lakes, streams, sewages;
- *gas concentration in air* for cities, laboratories, deposits;
- *soil humidity* and other characteristics;
- *inclination* for static structures (e.g., bridges, dams);
- *position changes* for, e.g., land slides;
- *lighting conditions* either as part of a combined sensing or standalone, e.g., to detect intrusions in dark places;
- *infrared radiation* for heat (fire) or animal detection.

Since these and many related applications typically use fewer sensor nodes, they are less demanding on the communication channels (both in-field and with the server), and for sensor node energy and cost. Consequently, the in situ wildfire detection application can be used as reference for the design of a WSN platform optimized for IoT environmental monitoring and the platform should be easily reusable for a broad class

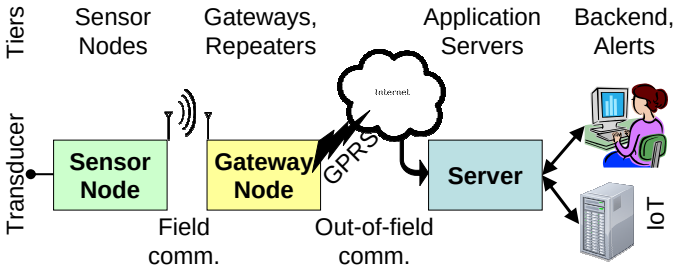


Fig. 2: Tiered structure of the WSN platform.

of related applications.

Thus, the requirements of a WSN platform for IoT long-term environmental monitoring can be defined as follows:

- low-cost, small sensor nodes with on-board processing, self-testing and error recovery capabilities;
- low-cost, small gateways (sinks) with self-testing, error recovery and remote update capabilities, and supporting several types of long-range communication;
- sufficient gateway hardware and software resources to support specific application needs (e.g., local transducers, and data storage and processing);
- detection of field events on-board the gateway to reduce network traffic and energy consumption;
- field communication protocol efficiently supporting:
  - from few sparse to a very large number of nodes;
  - low data traffic in small packets;
- fast and reliable field node deployment procedure;
- remote configuration and update of field nodes;
- high availability of service of field nodes and servers, reliable data communication and storage at all levels;
- node ruggedization for long-term environment exposure;
- extensible server architecture for easy adaptation to different IoT application requirements;
- multiple-access channels to server data for both human operators and automated processing;
- programmable multichannel alerts;
- automatic detection and report of WSN platform faults (e.g., faulty sensor nodes) within hours, up to a day;
- 3–10 years of maintenance-free service.

These requirements will be used in the following sections to define the requirements and analyze the design alternatives for the nodes, networking, deployment procedure and operation of a WSN platform suitable to support a significant set of IoT applications for environmental monitoring.

IV. Platform Structure

The main purpose of the WSN platform is to provide the users of the IoT application (human operators or computer systems) an updated view of the events of interest in the field.

The tiered structure of the used platform (see Fig. 2) was introduced by one of the first long-term outdoor WSN experiments [13] and allows:

- a good functional separation of platform components for optimization according to application requirements;
- a cloud-based field data access to bridge the latency-energy trade-offs of the low power communication segments and the ubiquitous and fast access to field data for end users (either humans or IoT applications).

The *sensor nodes* are optimized for field data acquisition using on-board transducers, processing, and communication to gateways using short-range RF communications, either directly or through other nodes. The *gateways* process, store, and periodically send the field data to the application server using long-range communication channels. The *application server* provides long-term data storage, and interfaces for data access and process by end users (either human or other applications).

The platform should be flexible to allow the removal of any of its tiers to satisfy specific application needs. For instance, the transducers may me installed on the gateways for stream water level monitoring since the measurement points may be spaced too far apart for the sensor node short-range communications. In the case of seismic reflection geological surveys, for example, the sensor nodes may be required to connect directly to an on-site processing server, bypassing the gateways. And when the gateways can communicate directly with the end user, e.g., by an audible alarm, an application server may not be needed.

In addition to the elements described above, the platform can include an *installer device* to assist the field operators to find a suitable installation place for the platform nodes, reducing the deployment cost and errors.

V. WSN Node Design

In this section will be presented the use of the specifications defined in Section III to derive the specifications of the WSN platform nodes, design space exploration, analysis of the possible solutions, and most important design decisions.

A. Sensor Node Design

Since IoT applications may require large numbers of sensor nodes, their specifications are very important for application performance, e.g., the in situ distributed wildfire detection selected as reference for the reusable WSN platform design.

One of the most important requirements is the sensor node cost reduction. Also, for low application cost the sensor nodes should have a long, maintenance-free service time and support a simple and reliable deployment procedure. Their physical size and weight is also important, especially if they are transported in backpacks for deployment.

Node energy source can influence several of its characteristics. Batteries can provide a steady energy flow but limited in time and may require costly maintenance operations for replacement. Energy harvesting sources can provide potentially endless energy but unpredictable in time, which may impact node operation. Also, the requirements of these sources may increase node, packaging and deployment costs.

Considering all these, the battery powered nodes may improve application cost and reliability if their energy consumption can be satisfied using a small battery that does not require replacement during node lifetime.

The sensor node energy consumption can be divided into:

- *RF communication*, for data and network maintenance;
- *processing*, e.g., transducer data, self-checks, RTC;
- *sensing*, e.g., transducer supply, calibration;
- *safety devices*, e.g., watchdog timer, brown-out detector;
- *power down* energy required by the node components in their lowest power consumption mode.

The last three (sensing, safety, and power down) depend mostly on component selection, while the energy for RF communication and processing depend mostly on the communication protocol and the application, respectively.

For the selection of the communication protocol it should be considered that the sensor nodes of event-driven environmental monitoring applications typically need to periodically report their health status and to notify alters of field events right after their detection. Thus, the simplest form of sensor node communication requirements can be implemented energy- and cost-effective using a transmit-only radio device.

However, having no radio receive capabilities the sensor nodes cannot form mesh networks and need to communicate directly with the receiver (gateway), in a star topology. They also cannot receive acknowledgments or prevent packet conflicts. Consequently, the protocol has to include redundant retransmissions at the source to keep the message loss acceptable for the application.

Note the distinction between *packet* and *message*. A *message* is the data to be conveyed and fits into a packet. Due to packet loss, the *same message* can be repeatedly sent using *different packets*. The message is received if *at least* one packet carrying it is properly received and the receiver should discard message duplicates.

To examine the redundancy necessary for such a network, Fig. 3 shows the simulation results for the number of lost "alive" *messages* by a gateway monitoring up to 5,000 sensor nodes in a star topology over one year time. Each node sends one heartbeat *packet* per hour and the gateway considers that a node is missing if no heartbeat *message* is received within a given timeout (the graph includes plots for timeouts from 30 minutes to almost 6 hours).

It can be seen that with enough transmission redundancy the false node missing reports due to message loss can be kept very low. For instance, if the application can accept a 6 h detection delay of missing (faulty) nodes, then the false report rate (report normally operating nodes as missing) due to message loss can be around 1/year for a field with about 1,000 sensor nodes.

More generally, a message loss rate of about 1/year can be assumed for an RF space utilization, due to concurrent transmissions, of about 8% ($\approx 1000\,\text{packets} \times 0.28\,\text{s}/\text{packet}/3600\,\text{s}$) if the message is sent with a redundancy of 6 (the number of packets that repeat the same message).

In this scenario, the duty cycle of the radio is very low, less than 0.01% (1 packet of 0.28 s sent every hour). Note that this assumes a long packet duration for a few bytes payload to minimize the receiver error rate (e.g., using 24 bytes preamble, 4 bytes sync word, 4 data bytes, 1200 Baud symbol rate, FEC and CRC for the packet structure of the widely used Texas Instruments CC1150 device, see Fig. 4).

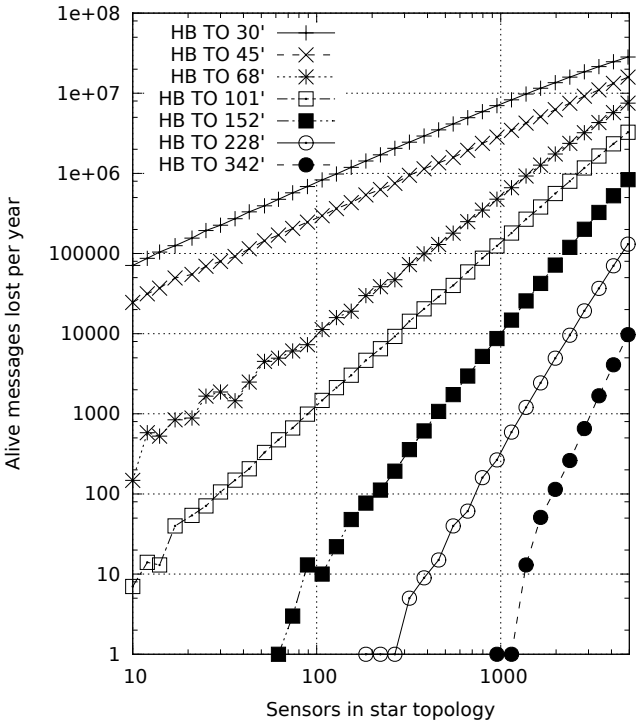


Fig. 3: Lost alive messages per year as function of node density and heartbeat timeout (280 ms packets, transmission rate 1/h).

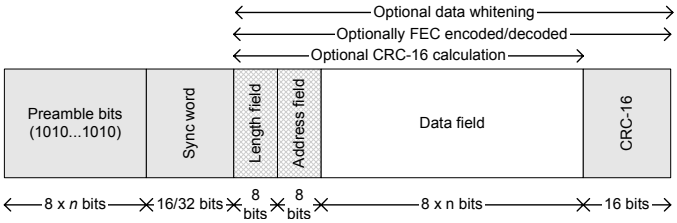


Fig. 4: TI CC1150 packet format (source: device data sheets).

If the application requires bidirectional sensor node communications, these need a transceiver. This increases their cost and energy consumption, which may translate to higher application cost, reduced service time, and higher maintenance.

The receiver operates with a low duty cycle to reduce its energy consumption. A widely used technique is low power listening (LPL, that can achieve duty cycles of 1-3% [29]). Better duty cycles and channel utilization can be achieved using synchronized networks [24], [30]. However, these require the nodes to keep track of the time using a constantly running accurate oscillator, which adds to node cost and energy consumption. Moreover, the oscillator drift in extreme environmental conditions may require more frequent synchronization messages or repeated energy-intensive network re-registrations if time synchronization is lost [9], [24]. Besides, the increased protocol complexity requires more processor resources (e.g., program and data memory, and processing), which further increase node cost and energy consumption.

However, other mechanisms can be used to reduce the radio energy consumption. For instance, the transmission power can be lowered if the communication range decreases by using a

mesh topology instead of a star topology. It should also be considered the additional traffic per node due to peer message forwarding. Also, message routing in large mesh networks can lead to undesired dynamic effects, some difficult to foresee and avoid, like bottlenecks or instabilities [17], [18]. These may reduce the network service reliability and require additional analysis and maintenance.

A receiver can also reduce packet collisions (e.g., using clear channel assessment (CCA) techniques), thus lowering the wasted energy due to message retransmissions. In this case, higher reception error rates can also be accepted if they can be compensated by an automatic retransmission of lost messages. This also allows to shorten the messages (e.g., by increasing the symbol rate, reducing the preamble, removing the FEC), effectively reducing the energy spent per message.

B. Gateway Node Design

The main role of the gateways in a WSN application is to collect, process, and forward to an Internet-connected server the field data received from the sensor nodes. They are fit with an in-field communication interface for sensor nodes and an out-of-field communication channel for the application server.

The long range communication, e.g., a GPRS modem (see Fig. 2) can be responsible for most energy consumption of the gateway. Although carefully optimized gateways can reach several years of operation using large battery packs [2], they are usually fit with energy harvesting devices (or mains power supply) to reduce their cost and maintenance requirements.

The gateway node shapes and sizes are typically less constrained by the application compared to sensor nodes. However, a small form factor generally helps gateway integration in a variety of applications. Moreover, specific applications may require to connect transducers directly to gateways.

The field communication protocol is typically selected to optimize the sensor nodes, which are especially important for applications with large numbers of sensor nodes. The impact of the field protocol on gateway resources (e.g., as processing, code size, data memory) is usually less important. Also, the communication with the peer gateways may use a different channel to prevent the congestion of the sensor node channel. This is especially useful when the MAC of the sensor nodes cannot improve the communication reliability by resending the lost messages (e.g., they have no receiving capabilities).

For star topology deployments in difficult propagation conditions it may be necessary to extend the reachability of the gateways using *repeater* nodes. Their basic operation is to forward to the gateways in range all sensor node messages they receive, so that the gateways can process them as if they were received directly from the sensor nodes.

In an event-driven application, the gateways reduce the communication with the server by processing the field data on-board to detect significant events. For instance, they can autonomously detect faulty nodes or analyze the data from several sensor nodes for events that cannot be detected at single sensor node level.

The gateways usually allow to remotely change their parameters, processing flows, or update their entire program [31]–[34] to improve their fit to (changing) application requirements. For a high quality of service over long periods of time it is also important to perform frequent self-checks and trigger error recovery mechanisms in case of anomalies. At the same time, they should also preserve as much as possible the data collected from the field across the error recovery procedures.

C. Deployment Device Design

This is typically an interactive handheld device used by the field personnel to ensure a suitable WSN node deployment in the field and to automatize the most time-consuming and error-prone parts of the deployment procedure. This is particularly important for applications with a large number of nodes (such as the reference application), which are particularly sensitive to deployment quality, time and cost.

Basically, it assists the operator in assessing if a node is operational and can properly operate in an installation spot. For a sensor node, this usually means assessing the suitability of its connections with peer nodes (for mesh networks) or the gateways/repeaters. For a gateway node it means assessing its connectivity with peers, the long range connection coverage (e.g., GPRS coverage), and the energy harvesting suitability (e.g., the proper orientation of the solar panels). The deployment device should be able to connect with the sensor nodes and gateways in range and display the data easy to see and understand in open nature conditions and by operators that may have just basic training in WSN operation.

Besides these basic functionalities, the deployment device can also have localization capabilities (e.g., GPS) that can guide the field operators to the predefined node deployment positions and record the actual node deployment locations. It may also have long range communication capabilities to upload the deployment data to the application server.

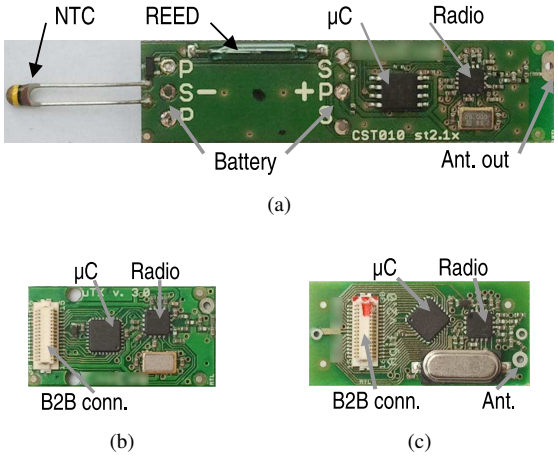
VI. Device Implementation

In the following are presented the most important implementation choices for the platform devices that are based on the requirements in Sections III and V and are suitable for long-term environmental monitoring IoT applications.

A. Sensor Node Implementation

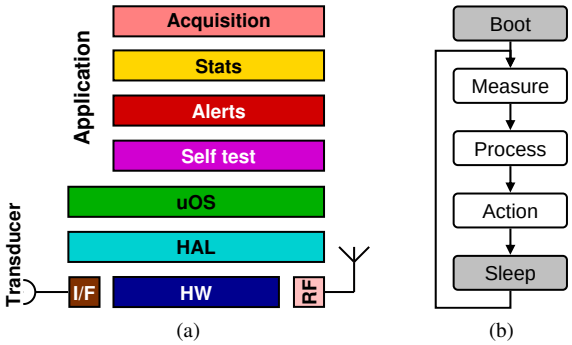
Fig. 5 shows several sensor nodes designed for long-term environmental monitoring applications. The node for in situ wildfire monitoring (PCB in Fig. 5a) is optimized for cost since the reference application typically requires a high number of nodes (up to tens of thousands). The communication protocol is unidirectional, not synchronized, as the node has no RF receive capability. As discussed in Section V-A, this solution minimizes node cost and energy requirements, and can also simplify network operation and maintenance.

The node microcontroller is an 8 bit ATMEL AVR ATtiny25 with 2 KB program and 128 bytes data memory, clocked by its internal 8 MHz RC oscillator (to reduce the costs and energy consumption, since it does not need accurate timings). The full custom 2 KB program has the structure in Fig. 6a. A minimal operating system supports the operation of the main program



(a) (b) (c)

Fig. 5: PCB of sensor nodes for environmental monitoring: (a) for in situ wildfire detection, (b) for mostly analog and (c) for mostly digital applications (scale ≈1:1).



(a) (b)

Fig. 6: Sensor node: (a) firmware structure for reference application and (b) operation state flow diagram.

loop shown in Fig. 6b and provides the necessary interface with the node hardware, support for node self-tests, and the communication protocol.

The NTC transducer is connected as a voltage divider and the main loop supplies it every second with a 0.02% duty cycle to acquire a temperature reading using the microcontroller ADC. Each sample updates the stored values of the instantaneous and average temperature, and its variation speed and sign. All values are then compared with specific patterns that are closely correlated to wildfires. A specific alert message is sent if any of these combinations is matched. To improve the continuity of service and lower the maintenance requirements, the sensor program periodically performs a suite of self-checks for hardware, software and configuration errors. Any anomaly is signalled to gateways, if possible.

The node average current consumption is about 4.7 μA, largely due to the microcontroller watchdog timer. Both the microcontroller and the transducer are active with duty cycles below 0.05%. The microcontroller consumes about 600 μA in active state and the transducer a few tens of μA, depending on the temperature. The current consumption during packet transmission does not exceed 30 mA.

The normal sensor node activity consists of sampling the temperature every second, processing the sample, and sending one 0.28 s packet/h. In these conditions, its theoretical service time exceeds 16 years on a ½ AA-size 1 Ah lithium battery. Sensors used in deployments for wildfire monitoring applications with up to 1,000 sensor nodes, some since 2008, continue to operate regularly without battery replacements.

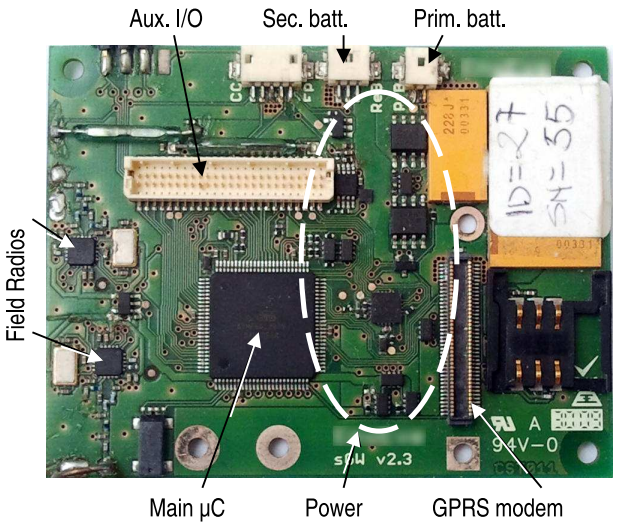
In alert conditions (e.g., if a wildfire is detected in the reference application), the sensor node priority switches from low energy consumption to propagating the alert quickly and reliably. The alert messages are repeated for the whole duration of the alert condition at random intervals between 1–3 s.

The sensor nodes use a channel in the 433 MHz ISM band to achieve better propagation in forest environment and longer range for a given RF power [35], [36], which is necessary to ensure a good gateway reachability in a star topology. A normal mode helical antenna (NMHA, approx. 2.4×1.1 cm) emerged from field tests as a good compromise between cost, size, and RF efficiency. It achieves a gain of about -9 dBi operating in close proximity to the tree trunk and inside the node plastic package.

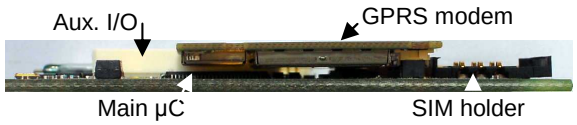
The PCB is a double sided FR-4 with components mounted on one side to reduce the costs. It is finished with conformal coating to withstand long environment exposure. The sample in Fig. 5a requires only the battery and the NMHA (co-linear to the PCB, on the right side) to be operational. Fig. 12a shows the node deployed on a tree in an application housed in a custom, low-cost plastic case. The NTC is in good thermal contact with the surrounding air through an aperture at the lower end of the package, to prevent water infiltration.

Fig. 5b and 5c show derivative sensor node platforms designed for fast development of various environmental monitoring applications. Their structure is similar to the temperature sensor in Fig. 5a. The NMHA is mounted on the right, while they can be plugged in application-specific boards using the B2B connector on the left. The application board typically hosts the application transducers, their interface circuitry, and the power supply. Most microcontroller pins and the power supply lines are routed to the connector so that it can be used to provide processing power, I/O, and networking to the application board, from which it requires only power supply.

These nodes use ATMEL AVR ATtiny261 microcontrollers (Fig. 5b, best suited for analog applications) and ATtiny48 (Fig. 5c, with more digital interfaces). Both have up to 8 KB program and 1 KB data memory. The protocol stack code (largely the same as for the temperature sensor) takes less than 1 KB. Without the application board, the idle current consumption is 4.8 μA average and up to 30 mA during message transmission. As for the temperature sensor in Fig. 5a, these are consistent with the long service duration required by the IoT applications. Various sensor nodes (for monitoring applications for dam stability, water level, chemical gases, etc.) deployed since 2009 either operate regularly on their original battery or had lifetimes consistent with theoretical calculations similar to the one above, adjusted for application-specific sensor energy requirements and operating conditions (e.g., the temperature, which can influence the battery capacity).



(a)



(b)

Fig. 7: PCB of the gateway node for environmental monitoring: (a) top view and (b) side view with the GPRS modem mounted on top (scale ≈1:1).

B. Gateway Node Implementation

Fig. 7a shows the gateway node double side FR-4 PCB with components mounted on one side for lower costs.

It uses an ATMEL AVR ATmega1281 microcontroller with 128 KB program and 8 KB data memory on-board. In-field communications are handled by two transceivers operating in the 433 MHz band, one for the sensor nodes (receive-only) and one for the peer gateways, to reduce the congestion on both channels as discussed in Section V-A and V-B. For long range communications are provided connectors for a Cinterion TC63i GPRS modem and its SIM. The power supply module on-board the gateway can power the modem and includes three connectors: one for an external 2 Ah Li-ion rechargeable battery, one for an external 3.6 V lithium primary battery (for backup power), and one for an optional external unregulated energy supply (e.g., a solar panel) that is used to charge the rechargeable battery (if connected). For proper operation, the gateway PCB in Fig. 7b needs: one of the batteries, the modem and a SIM card, and the antennas for the in-field and long-range communications.

The gateway continuously monitors the field channel for incoming sensor node messages using LPL to reduce the consumption of the radio receiver. Inter-gateway communications use an unscheduled protocol similar to that of the sensor nodes, since the traffic on this channel largely mirrors the sensor node traffic (the field messages are mirrored to the neighbour gateways to avoid data loss in case of failures).

The small dimensions of the gateway ($7.5 \times 5.7 \times 7$ mm excluding batteries and antennas) help its integration with

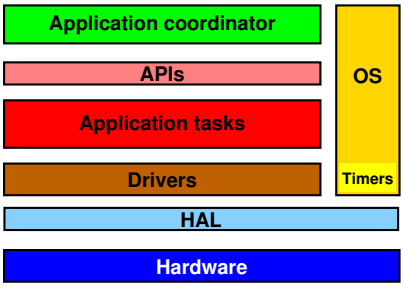


Fig. 8: Gateway firmware block diagram.

application-specific cases or boards, and the field deployment. However, the application may embed the gateway in larger structures, e.g., at the bottom of a birdhouse as shown in Fig. 12b, with solar panels for energy harvesting on the roof.

Several mechanisms contribute to gateway high availability of service required for IoT applications. The power manager switches automatically to the primary battery whenever the rechargeable battery is depleted, e.g., after extended periods of low energy harvesting. The gateway software also implements several run-time self- and peer-assisted checks and error recovery mechanisms for its most important functions. Recovery procedures attempt to limit service disruptions and avoid field maintenance operations, e.g., by using run-time reconfigurations, by auto-resets that preserve the data collected from field, or by automatically falling back to boot loader mode for remote control or firmware update. Moreover, most configuration parameters can also be remotely changed during normal operation through remote procedure calls (RPC).

Fig. 8 shows the layers of the full custom software structure of the gateway. The top-level operation is controlled by an *application coordinator*. On the one hand, it accepts service requests from various gateway tasks (e.g., as reaction to internal or external events, such as message queue nearly full or alert message received from field sensors, respectively). On the other hand, the coordinator triggers the execution of the tasks needed to satisfy the service request currently served. Also, the coordinator implements a priority-based service preemption allowing higher priority service requests to interrupt and take over the gateway control from any lower priority service requests currently being served. This improves the gateway forwarding time of alert messages, for instance.

The *application tasks* implement specific functionalities for the application, such as the message queue, field message handling, sensor node status, field message postprocessing, RPC, etc. They are implemented as round-robin scheduled co-routines to spare data memory (to save space and costs the gateway uses only the microcontroller internal RAM).

Manual configuration during sensor node deployment is not necessary because the field node IDs are mapped to the state structure using a memory-efficient associative array. The node IDs are added as they become active in gateway range up to 1,000 sensor nodes and 10 peer gateways, while obsolete or old entries are automatically reused when needed.

The gateway average current consumption in normal operation is 1.6–1.8 mA, depending on sensor node and peer traffic. It can rise to almost 500 mA during GPRS traffic in

worst connection conditions. Nevertheless, the gateway can operate for about one year on a D-size lithium battery in some applications, e.g., when it receives field data from only one sensor node and without peer gateways. In such cases, the average current decreases to 1.2 mA with the peer radio turned off, which amounts to $0.0012\,\mathrm{A} \times 24\,\mathrm{h} \times 365\,\mathrm{days} \approx 10.5\,\mathrm{Ah}$ in a year time. From the 19 Ah charge of a D-size lithium battery (e.g., Tadiran TL-5930) this leaves $19\,\mathrm{Ah} - 10.5\,\mathrm{Ah} = 8.5\,\mathrm{Ah}$ for GPRS traffic per year. At the maximum rate of 480 mA this means more than 2'50" of GPRS data transfer daily, more than enough to upload the data collected from one sensor.

In fact, gateways deployed inside sewages for level monitoring applications receiving data from one sensor node and no peers operate for one year on 19 Ah batteries, which is consistent with the theoretical calculations above. It is also worth noting that the gateway average current can be further reduced by using the hardware SPI port to interface with the radio devices and by programming the latter to autonomously scan for incoming packets instead of the software-controlled LPL over a software SPI port emulation used currently.

The gateways perform field data aggregation in a buffer (up to about 400 messages in the microcontroller internal data memory) to save energy and connect to the server either periodically (time-driven behaviour), or when the buffer becomes full or as soon and as long alert messages are received from field (event-driven behaviour). Since the sensor nodes transmit mostly heartbeat messages that update their state on the gateway, the message buffer fills gradually unless the gateway is configured to collect variable field data (such as temperature readings) that the sensor nodes piggyback on the heartbeat messages.

The repeater node uses the gateway design with unused hardware and software components removed.

C. Field Deployment Device Implementation

The deployment device is made by a gateway device connected through a serial port to an Openmoko smartphone platform[1] that runs a Linux operating system (see Fig. 9). The gateway interfaces with the field nodes, while the field data processing and the user interface (UI) are handled by an application running on Openmoko Linux OS.

The Openmoko GPS can be used to assist field orientation of the operator and node localization during deployment.

VII. Application Server

The main purpose of a WSN application server is to receive, store, and provide access to field data. It bridges the low power communication segments, with latency-energy trade-offs, and the fast and ubiquitous end user field data access (by humans or IoT applications).

The full custom server software has the structure shown in Fig. 10. It provides interfaces for:

- field nodes (gateways);
- the operators and supervisors for each field;
- various alert channels;

[1] http://wiki.openmoko.org/

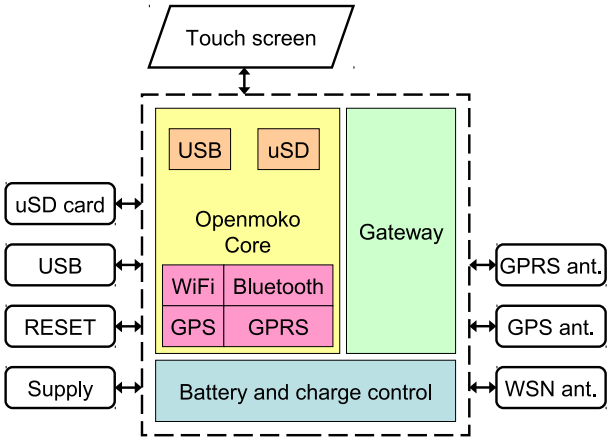


Fig. 9: Block structure of the deployment device.

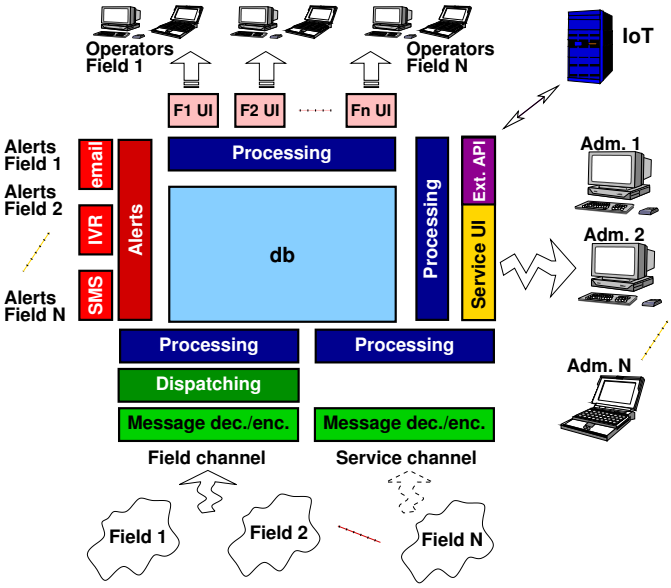


Fig. 10: Application server interfaces.

- external access for other IoT systems.

Each interface has a processing unit that includes, e.g., the protocol drivers. A central engine controls the server operation and the access to the main database. It is written in Java, uses a MySQL database and runs on a Linux operating system.

Two protocols are used to interface with the field nodes (gateways) for an energy-efficient communication over unreliable connections: normal and service (boot loader) operation.

The normal operation protocol acknowledges each event upon reception for an incremental release of gateway memory even for prematurely interrupted communications. Messages and acknowledges can be sent asynchronously to improve the utilization of high latency communication channels.

Time synchronization overhead is avoided at every communication level. The gateways timestamp the field messages and events using their relative time and the server converts it to real-world time using an offset calculated at the begin of the gateway communication session.

The protocol for the boot loader mode is stateless, optimized for large data block transfers and does not use acknowledges.

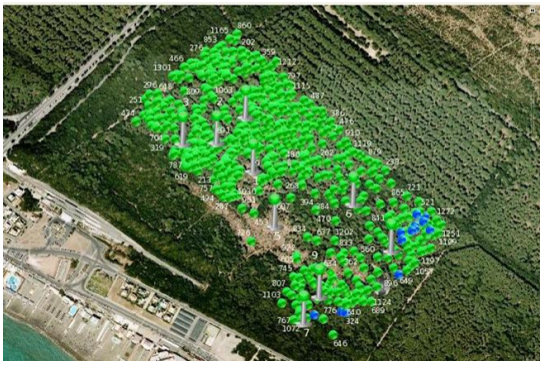


Fig. 11: Display of the status of a 1,000-sensor node field.

(a) (b)

Fig. 12: Typical deployment for the reference application nodes: (a) sensor and (b) gateway at the bottom of a birdhouse.

The gateway maintains the transfer state and incrementally checks and builds the firmware image. An interrupted transfer can also be resumed with minimal overhead.

IoT applications often produce large amounts of data that are typically synthesized in synoptic views by the servers [25], [26] (see Fig. 11). The server also uses high-availability techniques for a high quality of service, e.g., a shadow server automatically takes over the service if the main server fails.

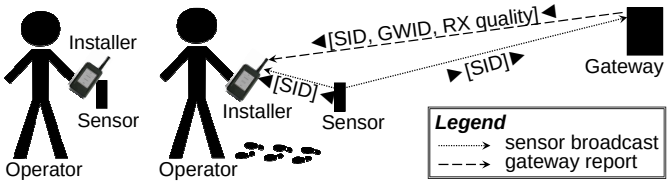
The integration with IoT applications is supported by providing remote access to historical and real-time field data.

VIII. Field Deployment Procedure

The node deployment procedure of the WSN platform aims to install each node in a field location both close to the application-defined position and that ensures a good operation over its lifetime. For example, Fig. 12 shows some typical deployments for the reference application nodes.

Node deployment can be a complex, time-consuming, error-prone, and manpower-intensive operation, especially for applications with a large number of nodes. Thus, it needs to be guided by automatic checks, to provide quick and easy to understand feedback to field operators, and to avoid deployment-time sensor or gateway node configuration.

The check of node connectivity with the network is important for star topologies and especially for transmit-only nodes (like the reference application sensor nodes). These nodes cannot use alternative message routing if the direct link with the gateway is lost or becomes unstable.



(a) (b)

Fig. 13: Field deployment of sensor nodes: (a) use deployment device magnet to set to deployment state, (b) display position suitability.

The deployment procedure of the sensor node of the reusable WSN platform takes into account the unidirectional communication capabilities of the sensor nodes. It is also designed to avoid user input and deployment-time configurations on the one hand, and a fast automatic assessment of the deployment position and reliable concurrent neighbour node deployment on the other hand.

The sensor nodes are temporarily switched to deployment operation by activating their on-board REED switch (see Fig. 5a) using a permanent magnet in the deployment device, as shown in Fig. 13a. This one-bit near field communication (NFC) ensures a fast, reliable, input-free node selectivity. Its device ID is collected by the deployment device that listens only for strong deployment messages. These correspond to nodes within just a few meters providing an effective insulation from collecting IDs of nearby concurrent node deployments.

The gateways that receive the sensor node deployment messages report the link quality with the node (see Fig. 13b). The deployment device collects all the data, and computes and displays an assessment of deployment position suitability. No gateway or node configuration is required and the procedure can be repeated until a suitable deployment position is found.

IX. Conclusion

WSNs are traditionally considered key enablers for the IoT paradigm. However, due to the widening variety of applications, it is increasingly difficult to define common requirements for the WSN nodes and platforms.

This paper addresses all phases of the practical development from scratch of a full custom WSN platform for environmental monitoring IoT applications. It starts by analysing the application requirements and defining a set of specifications for the platform. A real-life, demanding application is selected as reference to guide most of node and platform solution exploration and the implementation decisions.

All aspects of the WSN platform are considered: platform structure, flexibility and reusability, optimization of the sensor and gateway nodes, optimization of the communication protocols for both in-field and long range, error recovery from communications and node operation, high availability of service at all levels, application server reliability and the interfacing with IoT applications. Of particular importance are IoT requirements for low cost, fast deployment, and long unattended service time.

All platform components are implemented and support the operation of a broad range of indoor and outdoor field

deployments with several types of nodes built using the generic node platforms presented. This demonstrates the flexibility of the platform and of the solutions proposed.

The flow presented in this paper can be used to guide the specification, optimization and development of WSN platforms for other IoT application domains.

ACKNOWLEDGMENT

Most of the work was done in collaboration with and included in the product lines of Minteos s.r.l.[2], which contributed to functional specification, experimentation and production.

REFERENCES

[1] K. Romer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Comm.*, vol. 11, no. 6, pp. 54–61, Dec. 2004.

[2] I. Talzi, A. Hasler, S. Gruber, and C. Tschudin, "PermaSense: investigating permafrost with a WSN in the Swiss Alps," in *Proceedings of the 4th workshop on Embedded networked sensors*, ser. EmNets '07. New York, NY, USA: ACM, 2007, pp. 8–12.

[3] P. Harrop and R. Das, "Wireless Sensor Networks 2010–2020," IDTechEx Ltd, Downing Park, Swaffham Bulbeck, Cambridge, CB25 0NW, United Kingdom, Report, 2010.

[4] N. Burri, P. von Rickenbach, and R. Wattenhofer, "Dozer: Ultra-Low Power Data Gathering in Sensor Networks," in *Information Processing in Sensor Networks*, Apr. 2007, pp. 450–459.

[5] I. Dietrich and F. Dressler, "On the lifetime of wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 5, no. 1, pp. 5:1–5:39, Feb. 2009.

[6] B. Yahya and J. Ben-Othman, "Towards a classification of energy aware MAC protocols for wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 9, no. 12, pp. 1572–1607, 2009.

[7] J. Yang and X. Li, "Design and implementation of low-power wireless sensor networks for environmental monitoring," in *Wireless Communications, Networking and Information Security*, Jun. 2010, pp. 593–597.

[8] K. Martinez, P. Padhy, A. Elsaify, G. Zou, A. Riddoch, J. Hart, and H. Ong, "Deploying a sensor network in an extreme environment," in *Sensor Networks, Ubiquitous, and Trustworthy Computing*, vol. 1, Jun. 2006, p. 8.

[9] A. Hasler, I. Talzi, C. Tschudin, and S. Gruber, "Wireless sensor networks in permafrost research — concept, requirements, implementation and challenges," in *Proc. of 9th International Conference on Permafrost*, vol. 1, Jun. 2008, pp. 669–674.

[10] J. Beutel, S. Gruber, A. Hasler, R. Lim, A. Meier, C. Plessl, I. Talzi, L. Thiele, C. Tschudin, M. Woehrle, and M. Yuecel, "PermaDAQ: A scientific instrument for precision sensing and data recovery in environmental extremes," in *Information Processing in Sensor Networks*, Apr. 2009, pp. 265–276.

[11] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *Proceedings of the 7th symposium on Operating systems design and implementation*, ser. OSDI '06. Berkeley, CA, USA: USENIX Association, 2006, pp. 381–396.

[12] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli, "The hitchhiker's guide to successful wireless sensor network deployments," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, ser. SenSys '08. New York, NY, USA: ACM, 2008, pp. 43–56.

[13] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," *Wireless Sensor Networks*, pp. 307–322, 2004.

[14] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, ser. SenSys '05. New York, NY, USA: ACM, 2005, pp. 51–63.

[15] L. Bencini, F. Chiti, G. Collodi, D. Di Palma, R. Fantacci, A. Manes, and G. Manes, "Agricultural Monitoring Based on Wireless Sensor Network Technology: Real Long Life Deployments for Physiology and Pathogens Control," in *Sensor Technologies and Applications*, Jun. 2009, pp. 372–377.

[16] S. Verma, N. Chug, and D. Gadre, "Wireless Sensor Network for Crop Field Monitoring," in *Recent Trends in Information, Telecommunication and Computing*, Mar. 2010, pp. 207–211.

[17] Y. Liu, Y. He, M. Li, J. Wang, K. Liu, L. Mo, W. Dong, Z. Yang, M. Xi, J. Zhao, and X.-Y. Li, "Does wireless sensor network scale? A measurement study on GreenOrbs," in *INFOCOM, 2011 Proceedings IEEE*, Apr. 2011, pp. 873–881.

[18] M. Kuorilehto, M. Kohvakka, J. Suhonen, P. Hmlinen, M. Hnnikinen, and T. D. Hmlinen, *Ultra-Low Energy Wireless Sensor Networks in Practice*. John Wiley & Sons, Ltd, 2007.

[19] C. Hartung, R. Han, C. Seielstad, and S. Holbrook, "FireWxNet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments," in *Proceedings of the 4th international conference on Mobile systems, applications and services*, ser. MobiSys '06. New York, NY, USA: ACM, 2006, pp. 28–41.

[20] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "SensorScope: Out-of-the-Box Environmental Monitoring," in *Information Processing in Sensor Networks*, Apr. 2008, pp. 332–343.

[21] N. Kotamäki, S. Thessler, J. Koskiaho, A. Hannukkala, H. Huitu, T. Huttula, J. Havento, and M. Järvenpää, "Wireless in-situ sensor network for agriculture and water monitoring on a river basin scale in southern Finland: Evaluation from a data user's perspective," *Sensors*, vol. 9, no. 4, pp. 2862–2883, 2009.

[22] S. Burgess, M. Kranz, N. Turner, R. Cardell-Oliver, and T. Dawson, "Harnessing wireless sensor technologies to advance forest ecology and agricultural research," *Agricultural and Forest Meteorology*, vol. 150, no. 1, pp. 30–37, 2010.

[23] S. Tennina, M. Bouroche, P. Braga, R. Gomes, M. Alves, F. Mirza, V. Ciriello, G. Carrozza, P. Oliveira, and V. Cahill, "EMMON: A WSN System Architecture for Large Scale and Dense Real-Time Embedded Monitoring," in *Embedded and Ubiquitous Computing*, Oct. 2011, pp. 150–157.

[24] T. Watteyne, X. Vilajosana, B. Kerkez, F. Chraim, K. Weekly, Q. Wang, S. Glaser, and K. Pister, "OpenWSN: a standards-based low-power wireless development environment," *Transactions on Emerging Telecommunications Technologies*, vol. 23, no. 5, pp. 480–493, 2012.

[25] K. Aberer, M. Hauswirth, and A. Salehi, "Global Sensor Networks," Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland, Technical report LSIR-2006-001, 2006.

[26] M. Corra, L. Zuech, C. Torghele, P. Pivato, D. Macii, and D. Petri, "WSNAP: a Flexible Platform for Wireless Sensor Networks data collection and management," in *Environmental, Energy, and Structural Monitoring Systems*, Sep. 2009, pp. 1–7.

[27] C. Jardak, K. Rerkrai, A. Kovacevic, and J. Riihijarvi, "Design of large-scale agricultural wireless sensor networks: email from the vineyard," *International Journal of Sensor Networks*, vol. 8, no. 2, pp. 77–88, 2010.

[28] P. I. Fierens, "Number of wireless sensors needed to detect a wildfire," *International Journal of Wildland Fire*, no. 18(7), pp. 625–629, 2009.

[29] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, ser. SenSys '04. New York, NY, USA: ACM, 2004, pp. 95–107.

[30] C. Cano, B. Bellalta, A. Sfairopoulou, and J. Barcelo, "A low power listening MAC with scheduled wake up after transmissions for WSNs," *Communications Letters*, vol. 13, no. 4, pp. 221–223, Apr. 2009.

[31] T. May, S. Dunning, and J. Hallstrom, "An RPC design for wireless sensor networks," in *Mobile Adhoc and Sensor Systems Conference*, Nov. 2005, p. 138.

[32] M. Cohen, T. Ponte, S. Rossetto, and N. Rodriguez, "Using Coroutines for RPC in Sensor Networks," in *Parallel and Distributed Processing Symposium*, Mar. 2007, pp. 1–8.

[33] K. Whitehouse, G. Tolle, J. Taneja, C. Sharp, S. Kim, J. Jeong, J. Hui, P. Dutta, and D. Culler, "Marionette: using RPC for interactive development and debugging of wireless embedded networks," in *Information Processing in Sensor Networks*, 2006, pp. 416–423.

[34] F. Yuan, W.-Z. Song, N. Peterson, Y. Peng, L. Wang, B. Shirazit, and R. LaHusen, "A Lightweight Sensor Network Management System Design," in *Pervasive Computing and Communications*, Mar. 2008, pp. 288–293.

[35] J. Boan, "Radio Experiments With Fire," *IEEE Antennas and Wireless Propagation Letters*, vol. 6, pp. 411–414, 2007.

[36] C. Figueiredo, E. Nakamura, A. Ribas, T. de Souza, and R. Barreto, "Assessing the communication performance of wireless sensor networks in rainforests," in *Wireless Days*, Dec. 2009, pp. 1–6.

[2] http://www.minteos.com/