Solarfertigation: Internet of Things architecture for Smart Agriculture

Giovanni Valecce(1,2,4), Sergio Strazzella(2), Antonio Radesca(3), Luigi Alfredo Grieco(1,4)
(1)Dep. of Electrical and Information Engineering (DEI), Politecnico di Bari, Italy, Email:name.surname@poliba.it
(2)Sf System srl, Carosino(TA), Italy, Email:n.surname@solarfertigation.com
(3)Next Generation Team, Chişinău, Moldova, Email:antonio@nextgenerationteam.com
(4) CNIT, Consorzio Nazionale Interuniversitario per le Telecomunicazioni, Politecnico di Bari, Bari, Italy.

***Abstract*—Smart agriculture is a promising Internet of Things (IoT) application domain in the Industry 4.0 framework. Climate changes affect the natural resources exploitation policies becoming a serious matter in the agricultural and food production context. Continuous monitoring of environmental parameters and agricultural processes automation can lead to resources optimization and production maximization. This work proposes both the conceptual model and the design of Solarfertigation, an IoT system, specifically designed for smart agriculture. In particular, the envisioned solution is able to detect some of the most meaningful terrain parameters to feed a decision-making process that drives automated fertilization and irrigation subsystems. In addition, to attain energy self-sustainability, Solarfertigation is powered by a photovoltaic plant. The key features of Solarfertigation are illustrated throughout this contribution together with its preliminary prototype implementation.**

***Index Terms*—Internet of Things; Smart Agriculture, Photovoltaic, Irrigation, Fertilization, Wireless Sensor Network (WSN), Cloud Computing.**

I. INTRODUCTION

The fourth industrial revolution, known as Industry 4.0, entails a capillary digital ecosystem embracing each single component of the factory from the field to the supervision (and beyond) levels [1]. The IoT is a key enabling technology for the Industry 4.0, because it can inherently provide communication capabilities to any device within a plant of interest [2]. Those devices can create a pervasive and networked environment, usually referred to as Cyber-Physical System (CPS), that can provide unforeseen services in most of the verticals known so far.

Smart agriculture is a promising field of application of Industry 4.0 concepts [3]. In fact, the integration of CPS, IoT, and cloud/edge/fog computing technologies with agricultural machineries can foster the set up of precision agriculture systems, automated irrigation planning tools, optimization of plant growth process, monitoring of agricultural land, and management of production, which bring significant advantages with respect to classic crop management [4].

Potential IoT applications in Smart Agriculture cover a wide range of scenarios: greenhouse management through sensors, energy efficiency enhancement, observation of phenological phases, detection of insects or crop diseases, and also production chain traceability [5].

Despite the clear advantages that IoT brings to these environments, setting up only Decision Support Tools (DST) softwares is a rather limited approach to the perspectives of fully automation that these technologies would allow. DSTs aid in decision making by offering useful insights into easily consumable bites, but it is still difficult to analyze intangible or indefinable data and although these softwares have become much simpler over the years, many farmers still find it difficult to use [6].

In order to bridge this gap, a fully automated fertilization system (namely Solarfertigation) is proposed hereby, alongwith a thorough description of its architecture, components, and real use cases from the field. Solarfertigation is designed to integrate the decision-making process and both the fertilization and irrigation automation. This system is designed to be energy self-sufficient thanks to the embedded Photovoltaic (PV) plant. Solarfertigation unit is designed not only to provide IoT network gateway functionalities, but to combine all the hardware and software components needed to run electrical actuations, software-services and proper network connections. With reference to the IoT network, open standard short-range telecommunications technologies for added value IoT services have been implemented. In particular, the network is based on low power protocol architecture, able to support also critical applications that need deterministic (or almost deterministic) service provisioning. These activities are grounded in the 802.15.4 protocol stack.

A prototype version of Solarfertigation has been also implemented and tested on the field: its logical and physical setup will be presented hereby along with a comprehensive picture of the proposed system architecture.

The remainder of this work is organized as follows: section II provides a detailed description of the system architecture. Section II proposes a thorough characterization of both hardware and software components, with a focus on logical connections and real use cases. The main achievements and future work perspectives are highlighted in section V.

II. SYSTEM OVERVIEW

A. Fertilization machine

Solarfertigation system is a combination of a fertilizer machine, PV plant and automation devices and software. Fertilization and irrigation subsystem is composed by the components used to manage fertilizer solution and the hydraulic parts made up of pipes, pump, solenoid valves, manifolds, taps and other accessories, to distribute the solution in the field. PV System is made up of several panels positioned on an appropriate support structure, batteries, and an inverter. The system can use both solid and liquid fertilizers. They are dosed respectively through cochlea and volumetric pump into a tank, where the solution is prepared. The machine is capable of changing the type and quantity of fertilizers in solution. For each irrigation cycle, the system can simultaneously manage multiple crops even with different growth phases in the same crop cycle. It also aims to detect the main environmental parameters defining the optimal amount of water and fertilizer to be used. Thus, it contributes significantly, to increased productivity of the land and electricity, water and fertilizers saving.

B. IoT architecture

The chosen communication architecture (shown in Figure 1) is grounded on a static multihop IoT network that senses the environment and reports collected data to the cloud. The IoT network is composed by several nodes and a gateway. IoT devices measure soil moisture whereas the gateway gather the reading of devices and measures (on its own) ambient temperature and humidity. The gateway is a central control unit for the system, communicating with cloud platform in order to receive commands, store data and interact with User Interface (UI). Raspberry Pi single-board computer has been chosen to integrate gateway functionalities and software services. The hardware specification of this platform will be provided in following sections. Amazon Web Services (AWS) [7] is the cloud platform provider for the system. This technology allows to have a virtual computer cluster available on the Internet. In the proposed use case, AWS provides an ActiveMQ message-oriented middleware for Message Queue Telemetry Transport (MQTT) broker services.

System central unit design and hardware foresee that it not only acts as an IoT network gateway but integrates all the hardware and software components able to run electrical actuations, software micro-services for event management, cloud connection, and database integration.

We adopted Institute of Electrical and Electronic Engineers (IEEE) 802.15.4, a standard for wireless communication issued by the IEEE (Institute for Electrical and Electronics Engineers). This protocol stack provides a low data rate solution with multi-month to multi-year battery life and very low complexity, operating in an unlicensed, international frequency band [8]. This standard allows for communication in a point-to-point or a point-to-multipoint configuration. A typical application involves a central coordinator with multiple remote nodes connecting back to this central host. IEEE 802.15.4 Physical Layer (PHY) manages the signal transmission over the physical medium [9]. Offset-Quadrature Phase-Shift Keying (O-QPSK) modulation with a 2 Mbps physical data rate [8] is used and works in the 2.4 GHz Industrial, Scientific and Medical (ISM) band, in the range between 2.405 and 2.48 GHz. At Medium Access Control (MAC) layer, the IEEE 802.15.4 [10] features Time Slotted Channel Hopping (TSCH) [11] [12],which is characterized by two peculiarities: (i) time division into slots, and (ii) frequency (or channel) hopping. At the Adaptation Layer, two solutions are proposed: (i) 6tisch Operation Sublayer (6top), also known as IPv6 over networks of resource-constrained nodes (6lo) [13], and (ii) IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) [14]. 6top provides link states, TSCH configuration, control procedures and scheduling policies. 6LoWPAN is the name of a working group having as an objective the adaptation of IPv6 on IEEE 802.15.4-based networks [15] [16] [17]. At network layer, the IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH) protocol stack features Routing Protocol for Low-power and Lossy networks (RPL) [18] [19] as routing protocol for 6LoWPAN networks, supporting multipoint-to-point, point-to-multipoint or point-to-point traffic. MQTT is used as application layer protocol. This publish-subscribe protocol is suitable for low-bandwidth and constrained devices interconnection.

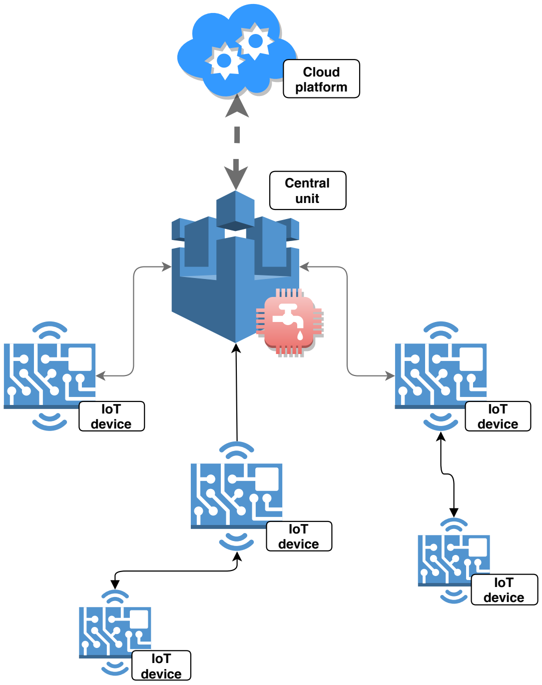


Fig. 1: IoT architecture

III. SYSTEM COMPONENTS

A. Hardware

This section provides an overview of the hardware needed for both IoT devices and central unit. The Raspberry Pi 3

Model B+[1] has been chosen as central unit.

This small single-board computer (Figure 2(a)) is the latest product in the Raspberry Pi 3 range, boasting a 64-bit quad core processor running at 1.4GHz, dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2/Bluetooth Low Energy (BLE), faster Ethernet and dual-band wireless LAN. Raspberry Pi provides a powerful feature for CPS interfacing: 40-pin General Purpose Input/Output (GPIO) pins. Any of the GPIO pins can be designated (in software) as an input or output pin and used for a wide range of purposes. The GPIO pins can be used with a variety of alternative functions, some are available on all pins, others on specific pins. In order to provide a simple weather station it has been integrated a DHT22 temperature-humidity sensor[2].

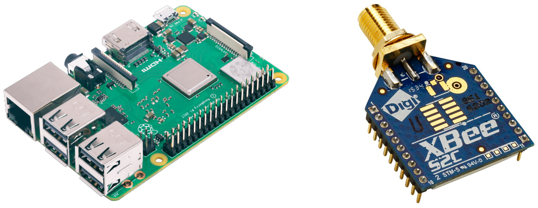

(a) Raspberry Pi 3 Model B+ (b) Xbee s2c

Fig. 2: Hardware equipment

The gateway also interacts with two external components:

- Analog-to-digital Converter (ADC);
- relay board.

The first is the Adafruit MCP3008[3] used for retrieving data like battery status and water level into the main tank, providing thus useful information to manage critical events such as low battery level and lack of water in the tank. The relay board is a LOW Level 5V 8-channel relay interface board[4] directly connected to the GPIO interface of raspberry pi, and powers on and off the specific output (water pumps, recycling motor and fertilizer mixers) which have separate power supply.

The components involved for each sensor node of the IoT are:

- XBee-PRO S2C RF module [5]
- DF Robot capacitive Soil Moisture Sensor [6]
- Nickel-Metal Hydride (NiMH) Battery
- PV panel

Leaving out of the scope power supply and recharge circuit, the Radio Frequency (RF) modules provided in this setup are the Digi International XBee®S2C 802.15.4 RF Modules (Figure 2(b)). It can provide quick, robust communication in point-to-point, peer-to-peer, and multipoint/star configurations[7].

B. Software

We based the software development on microservice architectural style instead of a monolithic structure. A monolithic application is structured with a single large codebase/repository that offer tens or hundreds of services using different interfaces such as HTML pages, Web services or/and REST services [20]. As can be seen in Figure 3, microservices architecture structures an application as a collection of loosely coupled services, enabling continuous delivery/deployment of large, complex applications.

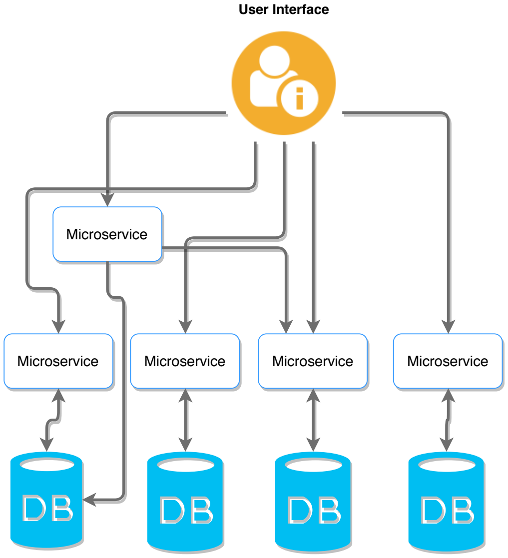


Fig. 3: Microservices and monolithic architectures

Scripts running on Raspberry pi driving CPS compose system backend and are developed using Node.js, an open-source, cross-platform JavaScript run-time environment that executes JavaScript code server-side. Frontend is deployed under Heroku Platform as a service (PaaS) provider and uses as data storage MongoDB, a document oriented database with JSON as data document format. This document store, is a computer program designed for storing, retrieving and managing document-oriented information, also known as semi-structured data [21].

The server side environment of the UI service is developed using Express and NodeJS. Adopted programming language for this service is Typescript, an open-source programming language defined as a strict syntactical superset of JavaScript (Figure 4).

C. Connections

Solarfertigation current release provides the following features:

- field data gathering, storage and visualization;
- real-time remote activation of irrigation and fertilization machines;
- date-based event scheduling;
- weather station.

---

[1] https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus

[2] https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf

[3] https://cdn-shop.adafruit.com/datasheets/MCP3008.pdf

[4] http://www.handsontec.com/dataspecs/module/8Ch-relay.pdf

[5] https://www.digi.com/resources/documentation/digidocs/pdfs/90001500.pdf

[6] https://www.dfrobot.com/wiki/index.php/Capacitive_Soil_Moisture_Sensor_SKU:SEN0193

[7] https://www.mouser.it/pdfdocs/ds_xbee-s2c-802-15-4.pdf

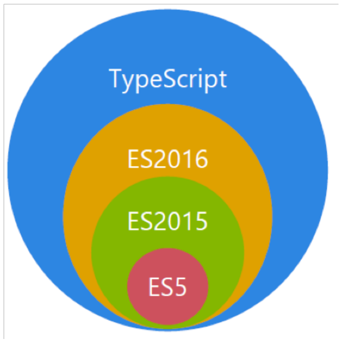


Fig. 4: Javascript layers

Every content is delivered to UI (Figure 6) using MQTT protocol [22] and ActiveMQ [23] message broker deployed on AWS as Software as a service (SaaS) (Figure 5).

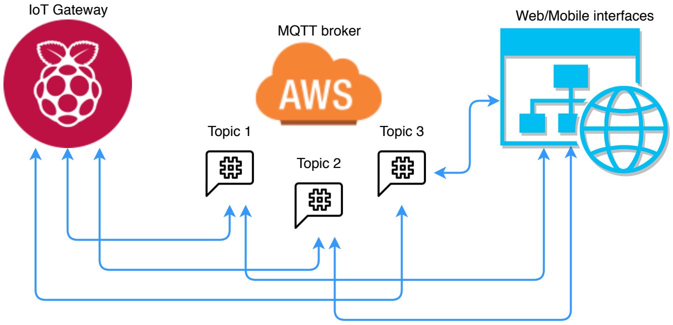


Fig. 5: MQTT connections

MQTT is a publish/subscribe, extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks [24]. The design principles are to minimize network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. Apache ActiveMQ is an open source message broker written in Java together with a full Java Message Service (JMS) client. It provides ”Enterprise Features” which in this case means fostering the communication from more than one client or server. Supported clients include Java via JMS 1.1 as well as several other ”cross language” clients.

MQTT with TLS client authentication broker is hosted on AWS cloud, a publish/subscribe broker service that enables the sending and receiving of messages to and from AWS IoT. Trough MQTT messaging structure it possible to deal with several system management aspects:

- data collection and storage into MongoDB;
- local database for scheduled events;
- real-time power-on and off commands.

Data provided by sensors are collected trough the IoT coordinator, processed by IoT gateway, sent to MQTT cloud broker and published into MongoDB. With these data, frontend side web and mobile graphical interfaces are built (6). On the other hand, local gateway listening to proper topics is able to collect messages containing informations about scheduled events, generating thus SQLite local database entries. A separate script is used to continuously read from local database changes and drive date-based events. Also remote actuation commands for real-time irrigation and fertilization devices control are sent via MQTT messages.

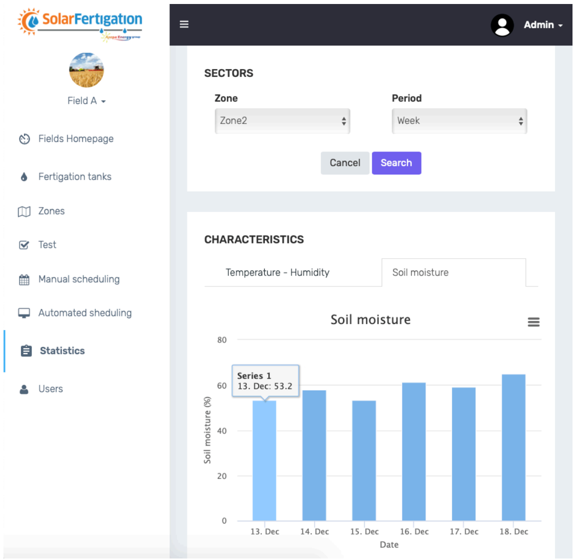


Fig. 6: Web interface for soil moisture section

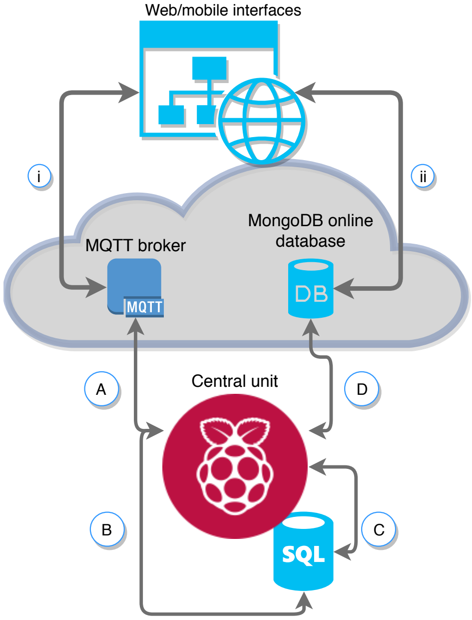


Fig. 7: System architecture

The system manages its functionality with the logical connections shown in Figure 7. (A) Mqtt broker and central unit exchange messages for real-time testing of electrovalves activation and events scheduling. MQTT message payload contains JSON data providing proper information fields. To prevent unpredictable behavior in case of internet connection failures, (B) connection represents the local storage of future events into local SQlite database. (C) The central unit continuously monitors events entry changes. Data gathered from the field sensors are stored into MongoDB Cloud Database (D). Web and mobile interfaces create MQTT messages (i) and read data to be displayed into graphs and interfaces from MongoDB online database(ii).

IV. EXPERIMENTAL SETUP

A. *Field operation*

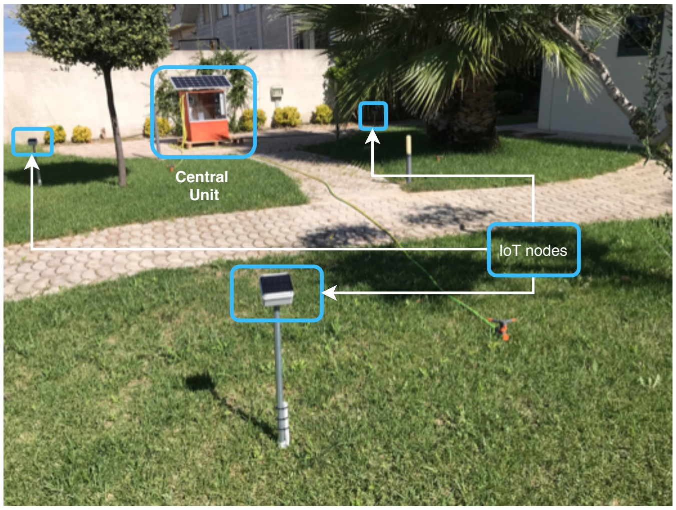
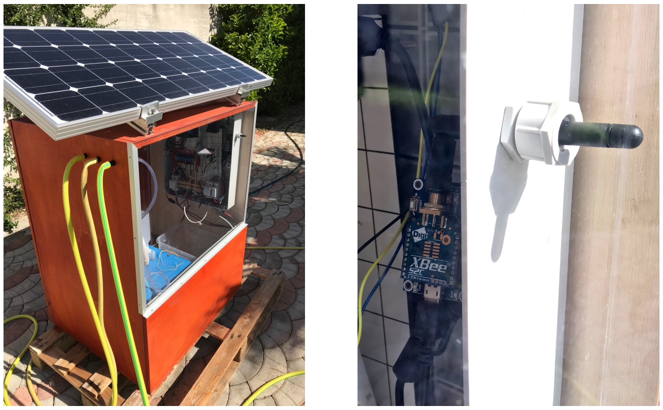


Fig. 8: Experimental setup with three IoT nodes

Figure 8 represents the experimental setup installed into a small garden. Here central unit and three remote sensor nodes are visible. In this setup the sensor nodes are placed at a distance of about 5 meters from each other due to spatial limitations.

As shown in Figure 9(a) the central unit is connected to the irrigation system through three outputs for three different field zones as well as the connection for water supply. The PV panel continuously charges two 12V batteries that supply power both to the irrigation system and to the electronic components.



(a) Solarfertigation central unit

(b) Xbee IoT coordintor

Fig. 9: Central unit and IoT Network coordinator

B. *Use cases*

Real-time activations For real-time testing purpose the systems creates MQTT messages containing information about the specific device to activate. The gateway listening to precise topic turns these information into electromechanical actuation.

Event scheduling and management Scheduled events messages provide information about quantity and type of fertilizers, irrigation duration and date/time details. Also event deletion, modification and logging of completed/aborted events are managed trough MQTT messaging. With these informations the electromechanical system is set to first create the required fertilizer mixture and then creating the water solution irrigating thus the area of interest.

Soil and ambient data publishing Data provided by sensors are collected by local gateway and published online on MongoDB as JSON objects. UI shows graphs and information from MongoDB.

In Table I some of the differences for use case requirements are listed.

Use case	**MQTT QoS**	**Database**
Real-time activations	2	No
Event scheduling and management	1	SQLite local
Environmental data collection	0	MongoDB

TABLE I: Use cases requirements

The Quality of Service (QoS) defines the guarantee of delivery for a specific message. There are three QoS levels in MQTT: at most once (0), at least once (1), exactly once (2). Indeed, for non-critical applications and redundant environmental data collection it is not necessary to ensure the highest level of service in MQTT. On one hand, building a local database for events storage, ensures the execution of scheduled irrigation and fertilization actions with no need of internet connection. On the other hand, NoSQL databases are more scalable and provide higher performance, and their data model befits the data collection use case of the proposed environment.

V. CONCLUSIONS AND FUTURE WORK

Solarfertigation system is still under development and the work will lead to scaling up the system with particular reference to the number of IoT nodes, different data sources interplay, integration of weather notifications, agronomic model-based algorithms, and machine learning integration. The current architecture has already been designed to support a large number of IoT nodes. Indeed, IEEE 802.15.4 supports heterogeneous traffic thanks to multi-hop routing and different traffic patterns among the IoT nodes. Soil moisture and weather data interplay will be evaluated from the agronomic point of view in order to improve farming algorithms and crop quality.

Although a weather station is already operating, a notification system for event management depending on weather forecasts is being integrated. Irrigation and fertilization events will be affected by environmental parameters following agronomic models for different crops and species in terms of nutrients and water needs, and various phenological stages. Some early experiments are in progress comparing well known studies such as Penman-Monteith [2] and Hargreaves-Samani [25]. The integration of agronomic models for different crops and the creation of a database including agronomic information for different species (nutrients, water needs, and phenological stages) is under development. Thanks to these first additions, algorithms for fertilization and irrigation management following the proposed models will be deployed. As data collection increases, it will be taken into account machine learning integration aiming to the full automation of farming processes. We will carry on extensive experimental tests to validate sensor node and machineries energy consumption as well as a comparison about water and fertilizer demand with a standard machine. Container images integration will be adopted to enhance portability and sharability of the software. About communication technologies some experimentation and performance evaluations will be conducted using Low Power Wide Area Network (LPWAN). In particular, Narrowband IoT (NB-IoT) performance evaluations are already underway and will be carried out both in laboratory set-up and on specific test sites. The research activity, will then provide a vision of a 5G-IoT technologies application into this Smart Agriculture environment (robotics and high bandwidth applications) considering a performance evaluation through experimental simulation tests. Then it will investigate the integration of the 5G technology within the industrial developed system.

ACKNOWLEDGMENT

This work was partially founded by Italian MIUR PON projects Pico&Pro (ARS01_01061), AGREED (ARS01_00254), FURTHER (ARS01_01283), RAFAEL (ARS01_00305) and by Apulia Region (Italy) Research Project E-SHELF (OSW3NO1).

REFERENCES

[1] Rainer Drath and Alexander Horch. Industrie 4.0: Hit or hype?[industry forum]. *IEEE industrial electronics magazine*, 8(2):56–58, 2014.

[2] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4):2347–2376, 2015.

[3] M. Hermann, T. Pentek, and B. Otto. Design principles for industrie 4.0 scenarios. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 3928–3937, Jan 2016.

[4] Partha Pratim Ray. Internet of things for smart agriculture: Technologies, practices and future direction. *Journal of Ambient Intelligence and Smart Environments*, 9(4):395–420, 2017.

[5] JM Barcelo-Ordinas, JP Chanet, K-M Hou, and J García-Vidal. A survey of wireless sensor technologies applied to precision agriculture. In *Precision agriculture13*, pages 801–808. Springer, 2013.

[6] Limitations & disadvantages of decision support systems. https://www.managementstudyguide.com/limitations-and-disadvantages-of-decision-support-systems.htm. (Accessed on 01/03/2019).

[7] Amazon web services (aws) - cloud computing services.

[8] Maria Rita Palattella, Nicola Accettura, Xavier Vilajosana, Thomas Watteyne, Luigi Alfredo Grieco, Gennaro Boggia, and Mischa Dohler. Standardized protocol stack for the internet of (important) things. *IEEE communications surveys & tutorials*, 15(3):1389–1406, 2013.

[9] LAN/MAN Standards Committee et al. Part 15.4: wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (lr-wpans). *IEEE Computer Society*, 2003.

[10] Approved ieee draft amendment to ieee standard for information technology-telecommunications and information exchange between systems-part 15.4:wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (lr-wpans): Amendment to add alternate phy (amendment of ieee std 802.15.4). *IEEE Approved Std P802.15.4a/D7, Jan 2007*, 2007.

[11] IEEE Standard for Low-Rate Wireless Networks –Amendment 2: Ultra-Low Power Physical Layer. *IEEE Std 802.15.4q-2016 (Amendment to IEEE Std 802.15.4-2015 as amended by IEEE Std 802.15.4n-2016)*, pages 1–52, 4 2016.

[12] Ieee standard for local and metropolitan area networks–part 15.4: Low-rate wireless personal area networks (lr-wpans) amendment 1: Mac sublayer. *IEEE Std 802.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011)*, pages 1–225, Apr. 2012.

[13] Qin Wang, Xavier Vilajosana, and Thomas Watteyne. 6tus layer specification. Internet-Draft draft-wang-6tsch-6tus-01, IETF Secretariat, May 2013. http://www.ietf.org/internet-drafts/draft-wang-6tsch-6tus-01.txt.

[14] Pascal Thubert, Erik Nordmark, Samita Chakrabarti, and Charles E. Perkins. Registration Extensions for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Neighbor Discovery. RFC 8505, November 2018.

[15] J. Hui and P. Thubert. Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks. *Internet Engineering Task Force (IETF) - Request for Comments: 6282*, 2011.

[16] Tim Winter, Pascal Thubert, Anders Brandt, J Hui, Richard Kelsey, Philip Levis, Kris Pister, Rene Struik, J Philippe Vasseur, and R Alexander. Rpl: Ipv6 routing protocol for low-power and lossy networks. Technical report, 2012.

[17] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. *Internet Engineering Task Force (IETF) - Request for Comments: 4944*, 2007.

[18] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, and R. Alexander. Rpl: Ipv6 routing protocol for low-power and lossy networks. RFC 6550, RFC Editor, March 2012. http://www.rfc-editor.org/rfc/rfc6550.txt.

[19] Tim Winter, Pascal Thubert, Anders Brandt, J Hui, Richard Kelsey, Philip Levis, Kris Pister, Rene Struik, J Philippe Vasseur, and R Alexander. Rpl: Ipv6 routing protocol for low-power and lossy networks. Technical report, 2012.

[20] Mario Villamizar, Oscar Garcés, Harold Castro, Mauricio Verano, Lorena Salamanca, Rubby Casallas, and Santiago Gil. Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud. In *Computing Colombian Conference (10CCC), 2015 10th*, pages 583–590. IEEE, 2015.

[21] What is mongodb? — mongodb. https://www.mongodb.com/what-is-mongodb. (Accessed on 08/01/2018).

[22] D Locke. Mq telemetry transport (mqtt) v3. 1 protocol specification. ibm developerworks technical library (2010), 2010.

[23] Apache activemq – index. https://activemq.apache.org/. (Accessed on 08/01/2018).

[24] Mqtt. http://mqtt.org/. (Accessed on 08/07/2018).

[25] George H Hargreaves and Zohrab A Samani. Reference crop evapotranspiration from temperature. *Applied engineering in agriculture*, 1(2):96–99, 1985.