VOL. 16, NO. 7, APRIL 2021 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

GEOLOCATION SYSTEM OF FIRE MONITORING STATION THROUGH LORAWAN

Pedro Luna[1], Sebastián Gutiérrez[1] and Hiram Ponce[2]
[1]Facultad de Ingeniería, Universidad Panamericana, Aguascalientes, México
[2]Facultad de Ingeniería, Universidad Panamericana, Ciudad de México, México
E-Mail: pedro.luna@up.edu.mx

ABSTRACT

The evolution of the Internet of Things (IoT) over the last few years has a growing outcome, where LoRaWAN™ based solutions have a remarkable increase in its quality and innovation. This evolvement is pushing the integration of new capabilities, such as real-time responsiveness, resilient systems, massive management of devices, geolocation, tracking, and smarter notification systems. The current work presents a design and an implementation proposal of a LoRaWAN™ solution to monitor the probability of existing fire in a given location. This application is capable of integrating the above-mentioned features by exploiting the benefits of LoRaWAN™ protocol and its capability to interact with cloud services. The system makes use of The Thing Network (TTN) as LoRa™ Network Server (LNS), LoRa™ cloud Geolocation Services for tracking, Amazon Web Service Elastic Cloud Computing (AWS EC2) as a container of the application, and advanced features of Go programming language for the rest of features such as notifications, dashboards and data operations as extraction, transformation and load (ETL).

Keywords: cloud computing, fire monitoring, IoT, LoRaWAN™, reactive systems, real-time monitoring.

1. INTRODUCTION

Software requirements have drastically transformed over the last decade, the sprout of new technologies and the improvement of the existing ones have lead tech-users to change their consuming habits [1]. The notorious improvement in worldwide connectivity has delivered significant increases in speed, coverage, and access to the internet, granting the opportunity to developers to rethink classical solutions such as computer programs, network infrastructure, and even complete information systems. Within this context, the Internet of Things (IoT) and cloud computing gains relevance and interest among developers and academy. Both topics, with their own issues and peculiarities, face common challenges due to its internet-based and distributed nature, where availability, reliability, and appropriate use of resources are critical matters.

Conveniently, knowledge areas as reactive system design, have already addressed a proposal to cope with these scenarios. Although IoT systems can be designed and developed under classical object-oriented programming paradigms, who has a great variety of well-tested design patterns such as Group of Four (GoF) or Model view controller (MVC), the reactive design has a particular focus on systems that expect to handle billions of clients [2], a requirement that is imperative to comply in any IoT system due to the number of devices currently connected to the Internet [3]. Given the diversity in IoT application domains [4], a concrete topic is required to test the before mentioned statements. In this case, fire monitoring has been selected, because according to existing literature [5], to create a solution for early detection and its opportune warning, requires an affordable amount of resources, and the prevention of fire means a relevant economic impact. Under the circumstances described in [6-10], many fire incidents take place in wide-open areas, where electricity and telecommunications are not always available.

LoRa™ is a long-range protocol with low energy consumption that has been tested in previous cases, delivering satisfactory results that prove its proficiency in suburban and rural areas [11-13]. As a complement, LoRaWAN™ is another protocol that implements an extra layer of security for data privacy, integrity, and availability, and concurrently, links LoRa™ with classical network infrastructure, which enables the possibility to use the collected data with other services hosted in the cloud [14][15]. For these reasons, LoRa™ and LoRaWAN™ have been selected as communication protocols for this project.

Another relevant topic for this solution is the visualization of results. Given the importance to locate the position of an early fire alarm to take actions in real-time and the power consumption constraints before mentioned, Global Navigation Satellite System (GNSS) based solutions are substituted by innovative methods for obtaining the position of monitors. In [16], this is possible through massive data algorithms that classify data to generate a matrix with frequencies that works along with a matrix of territorial adjacency and then loads that data to train machine learning systems. Other interesting approaches are proposed in [17], which uses the Time Difference of Arrival data to calculate the coordinates of the stations, in [18], where Time of arrival TOA is combined with TDOA to improve precision, or [19], where Angle of Arrival (AoA), TDA and TDOA is used to refine precision. For this work, Semtech's API for geolocation [20] is used, this method computes the position by using the Received Signal Strength Indicator (RSSI), Signal Noise Ratio (SNR), TDOA (optional value), and the coordinates (Latitude, Longitude) from LoRa™ gateways in range.

In this work, we present a design and an implementation proposal of a LoRaWAN™ solution to monitor the probability of existing fire in a given location. This application is capable of integrating the above-mentioned features by exploiting the benefits of LoRaWAN™ protocol and its capability to interact with cloud services. The system makes use of The Thing Network (TTN) as LoRa™ Network Server (LNS), LoRa™ cloud Geo-location Services for tracking, Amazon Web Service Elastic Cloud Computing (AWS EC2) as a container of the application, and advanced features of Go programming language for the rest of features such as notifications, dashboards and data operations as extraction, transformation, and load (ETL).

This document contributes to previous proposals presenting an end-to-end solution that integrates data gathering, processing, geo-location, and visualization throughout a minimal application composed of a dashboard and a map that displays the status and location of fire monitoring stations nodes through LoRaWAN™.

The rest of the document is organized as follows: Section II describes the material and methods. Section III illustrates the results. Section IV presents the conclusions. Finally, future work is presented in section V.

2. MATERIALS AND METHODS

All LoRa™ related hardware has been selected, configured, and provisioned according to RP002-1.0.1 LoRaWAN™ Regional Parameters [22], the architecture for the proposed IoT system has been designed using the LoRaWAN™ 1.1 specification [21], and integrated as described in Figure-1.

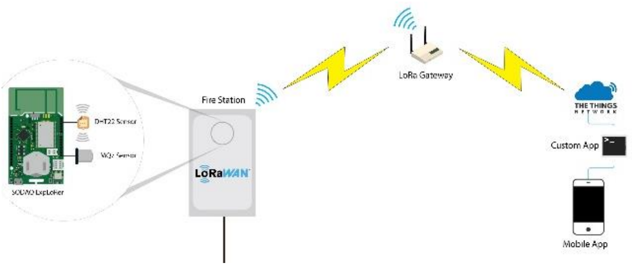


Figure-1. Architecture of IoT solution.

The application has been programmed in Go language using the requirements established by Reactive design patterns [2]. The integration of services and processing of metadata has been possible using the information contained in LoRaWAN™ Backend Interfaces 1.0 specification [23]. Table-1 contains a summary of the materials used to deploy and test the solution in Aguascalientes, central Mexico (21.9 °N, -102.3 °E), using the 915 MHz frequency plan.

Table-1. Required material for the solution.

Domain	Component	Model	Manufacturer
End Node	**Prototype Board**	**SODAQ Explorer**	**Microchip**
	Temperature, humidity sensor.	DHT22	----
	CO Sensor	MQ7	----
Concentrator	LoRa™ Gateway	Wirnet iFemtoCell 915	Kerlink
Network / Cloud Services	LNS	Public community network	TTN
	Compute resources	AWS EC2	Amazon
	Map render Service	Google Maps API	Google
	Geolocation solver	Geolocation LoRa™ Cloud Services	Semtech
Application	Custom Application	----	---

From a workflow perspective, this solution uses the data pipeline described in Figure-2. Where:

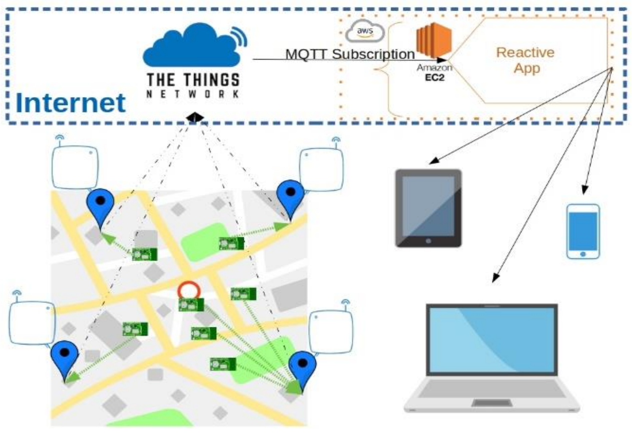


Figure-2. Data pipeline.

a) The end nodes gather information from the environment and send the data to the gateways using the LoRa™ protocol. The end nodes consist of a SODAQ ExpLoRer board equipped with a DHT22 sensor to read levels of temperature and humidity, and an MQ7 sensor to measure CO levels. They join the LoRa™ Network using the OTAA method and operates under Class A mode, which allows for bi-directional communications whereby each end devices uplink transmission is followed by two short downlink receive windows. The transmission slot scheduled by the end-device is based on its own communication needs with a small variation based on a random time basis (ALOHA-type of protocol). This Class A operation is the lowest power end-device system for applications that only require downlink communication from the server shortly after the end-device has sent an uplink transmission. Downlink communications from the server at any other time will have to wait until the next scheduled uplink.

b) The gateways collect data from every end node in range and forward the packages via TCP-UDP to a specific LNS. This process is illustrated on the left side of Figure-3.

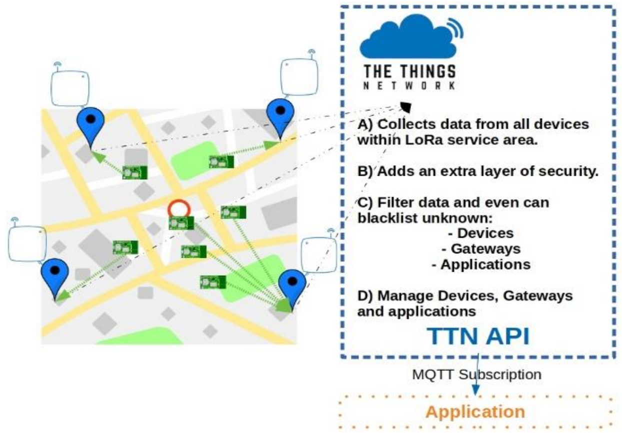

THE THINGS NETWORK
A) Collects data from all devices within LoRa service area.
B) Adds an extra layer of security.
C) Filter data and even can blacklist unknown:
- Devices
- Gateways
- Applications
D) Manage Devices, Gateways and applications
TTN API
MQTT Subscription
Application

Figure-3. The Things Network (TTN) connection and role of the solution.

During the setup, the iFemtoCell gateways have to be reconfigured to point at the closest TTN endpoint which in this case is ttn-router-us-west.

c) The LNS receives the information sent from several gateways, discard unauthorized packages, and associates the data collected with the corresponding application as described on the right side of Figure-3. For this reason, all end nodes and gateways have to be previously provisioned in the TTN platform. Note that at this point the data is not stored anywhere, TTN lacks a mechanism to do this. Although they have integrations with third-party platforms that allow doing so, in this case, the data is not required for further analysis, hence, is more effective to consume the data in real-time, this task is performed by the custom application using the TTN Data API which allows creating a bridge between TTN MQTT broker service and our application.

d) A an MQTT subscription notifies the application every time new data is available, so it can fetch a JSON formatted stream of a LoRa™ uplink that is structured as defined by [21] and [23]. Once the data has been acquired, the application triggers two ETL operations simultaneously. These operations are detailed in 5) and 6). This application was developed entirely in Go, taking advantage of its concurrent and multi-paradigm nature. Unfortunately, there is no implementation of the actor model in Go, which has to be solved by creating a singleton for every subscription and then an observer for any client that requires to get information of a given monitor. However, the architecture of application follows the reactive principles which are to be Responsive, resilient, elastic, and message-driven. This allows the application to handle massive connections, guarantee its own availability and reliability, and dynamically scale up and down according to the demand of service (Figure-4).

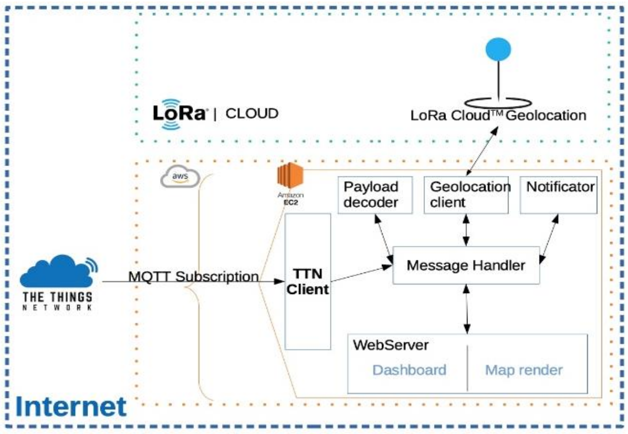
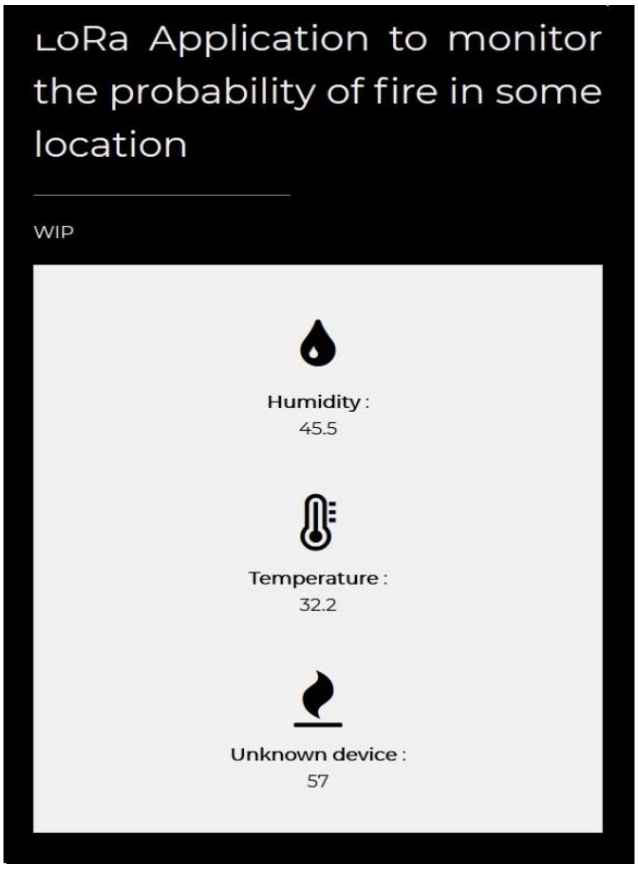


Figure-4. Application architecture.

e) The set of ETL operations begin with the payload decoder, which consists of extracting the payload from the LoRa™ uplink and then transforms its content using a decoder. This decoder converts a sequence of hexadecimal values into different types of values. The payload also contains an identifier that allows knowing the type of sensor that has generated the value. This information is organized and loaded into the dashboard service as shown in Figure-5.


LoRa Application to monitor the probability of fire in some location

WIP

Humidity :
45.5

Temperature :
32.2

Unknown device :
57

Figure-5. Temperature, humidity and CO monitoring dashboard.

f) ETL operations continue by using the uplink metadata and extract the required values by the LoRa™ Cloud services to compute geolocation, those values are transformed into a JSON object and then loaded into Geolocation service as described in Figure-6.

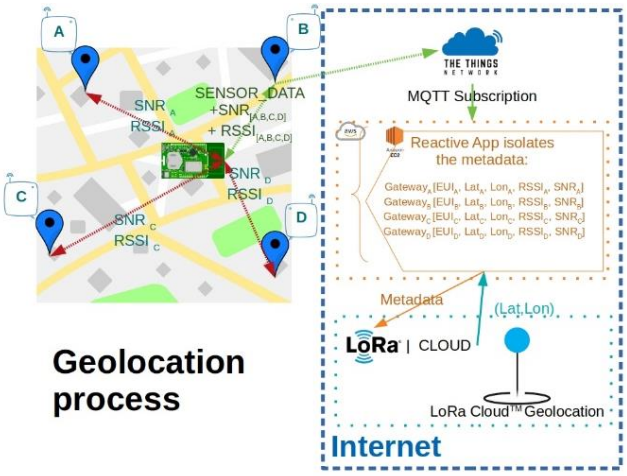
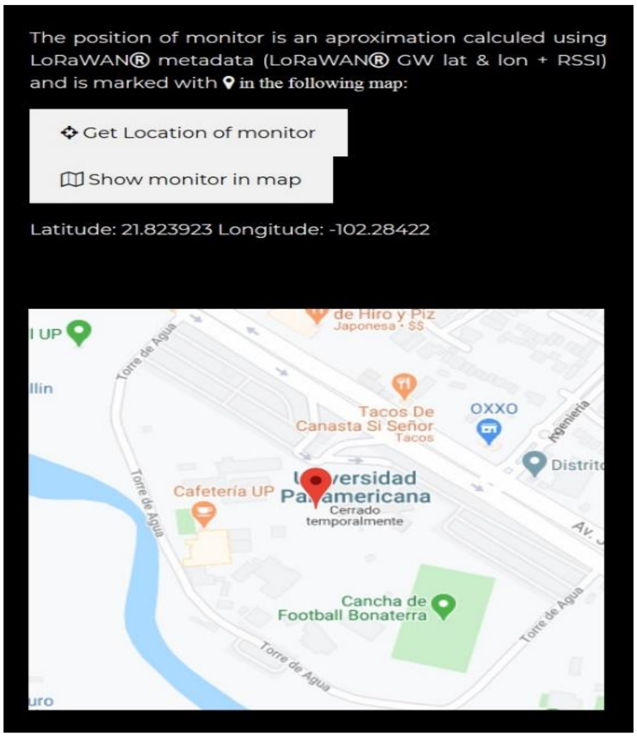

A
B
C
D
SNR A
RSSI A
SENSOR_DATA
+SNR [A,B,C,D]
+ RSSI [A,B,C,D]
SNR D
RSSI D
SNR C
RSSI C
THE THINGS NETWORK
MQTT Subscription
aws
Reactive App isolates the metadata:
Gateway A [EUI A, Lat A, Lon A, RSSI A, SNR A]
Gateway B [EUI B, Lat B, Lon B, RSSI B, SNR B]
Gateway C [EUI C, Lat C, Lon C, RSSI C, SNR C]
Gateway D [EUI D, Lat D, Lon D, RSSI D, SNR D]
Metadata
(Lat,Lon)
LoRa | CLOUD
LoRa Cloud™ Geolocation
Internet
Geolocation process

Figure-6. Geolocation process.

g) Once the results are ready, the data is evaluated by the 30-30-30 rule that is an algorithm used on large forest fire prevention cases and has proven to be effective in [10].

h) Finally, when a possible fire incident is detected, the application sends an email alert containing a warning message and the coordinates of the monitor that has reported the incident. Otherwise, information is temporarily stored and discarded once new data arrives. In the Dashboard, a button has been added to manually retrieve the position of a given monitor, a second button has been added to call Google maps API and render a map to have a visual reference of the monitor position, as shown in Figure-7.


The position of monitor is an aproximation calculed using LoRaWAN® metadata (LoRaWAN® GW lat & lon + RSSI) and is marked with in the following map:

Get Location of monitor

Show monitor in map

Latitude: 21.823923 Longitude: -102.28422

Figure-7. Geolocation board of fire monitoring station.

3. RESULTS

Throughout this application, previous work related to fire monitoring using LoRa™ was easily replicated, confirming that this protocol is suitable for solving this kind of issue.

The objective of integrating services to add value to previous solutions has been accomplished by the integration of geolocation, notifications, and a descriptive interface that properly displays real-time data in a meaningful and self-descriptive interface.

The most challenging issues during the development of this application were related to implementing the elements suggested by the reactive design patterns, however, an acceptable behavior has been obtained following the recommendations given by [2]. Obtained values during testing demonstrate the temperature and CO sensor values increase, while the relative humidity values decrease as presented in Figure-8.

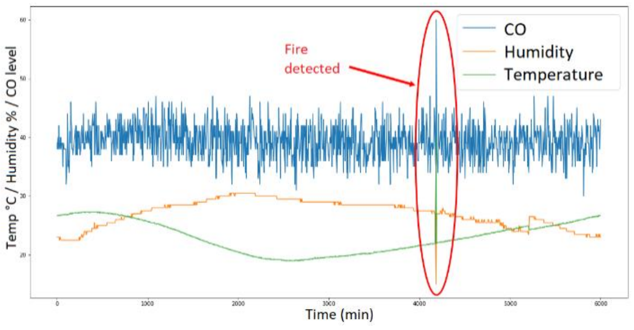
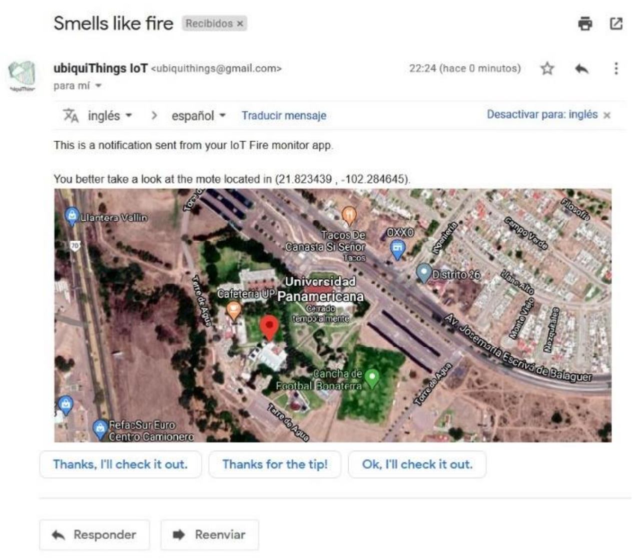


Figure-8. Temperature, Humidity and CO values during a detected fire incident.

During the testing phase, the application was able to notify an induced incident of fire as illustrated in Figure-9, where we can observe the email notification eceived by the user.


Smells like fire Recibidos ×

ubiquiThings IoT <ubiquithings@gmail.com> 22:24 (hace 0 minutos)
para mí

inglés > español Traducir mensaje Desactivar para: inglés ×

This is a notification sent from your IoT Fire monitor app.

You better take a look at the mote located in (21.823439 , -102.284645).

Llantera Vallin
Tacos De Canasta Sí Señor
OXXO
Distrito 56
Universidad Panamericana
Cafetería UP
Cancha de Fútbol Bonaterra
Av. Josemaría Escrivá de Balaguer
Tierra de Agua
Fefa Sur Euro Central Camionero
Thanks, I'll check it out.
Thanks for the tip!
Ok, I'll check it out.
Responder
Reenviar

Figure-9. Email notification of fire.

4. CONCLUSIONS

In this paper, we proposed a proof-of-concept design and implementation of a LoRaWAN-based system to monitor the probability of existing fire in a given location. This application was capable of exploiting the benefits of LoRaWAN™ protocol and its capability to interact with cloud services. Features like notifications and dashboards were implemented properly.

To cope with an end-to-end solution that meets the requirements demanded by IoT, our work still needs to integrate more elements. A significant improvement to comply with reactive systems can be achieved by using resources as Scala, which is a programming language created to develop reactive systems and already has implementations of the required components for this kind of solution. Another resource that could be used to improve this work are containers to create micro-services.
The novelty of the present study is the demonstration of capable of integrating a LoRaWAN™ solution to monitor the probability of existing fire in a given location by exploiting the benefits of protocol and its capability to interact with cloud services.

5. FUTURE WORK

To further extend this work, we will include redesigning the geo-location system of the fire monitoring station to improve its performance and meet standards of reactive systems.

ACKNOWLEDGEMENT

Special thanks to Universidad Panamericana for allowing us to use their LoRa™ infrastructure, to all the members of Semtech's R\&D team in Rapperswil, CH. for sharing their knowledge and expertise, and to all the people that supported the creation of this App.

REFERENCES

[1] J. Bonér, D. Farley, R. Kuhn and M. Thompson. 2014. The Reactive Manifesto. Reactivemanifesto.Org.

[2] R. Kuhn, B. Hanafee and J. Allen. 2017. Reactive design patterns. Manning Publications Company.

[3] Gartner Says 5.8 Billion Enterprise and Automotive IoT End points Will Be in Use in 2020, Online: https://www.gartner.com, Last access: May 30, 2020

[4] O. Vermesan *et al.* 2011. Internet of Things: Strategic Research Roadmap. in Internet of Things-Global Technological and Societal Trends.

[5] G. Saldamli, Et Al. 2019. Wildfire Detection using Wireless Mesh Network, Fourth International Conference on Fog and Mobile Edge Computing.

[6] E. Surya, M. Rosmiati, F. Rizal. 2018. Integrating Forest Fire Detection with Wireless Sensor Network Based on Long Range Radio. International Conference on Control, Electronics, Renewable Energy and Communications.

[7] S. Rizanov, A. Stoynova, D. Todorov. 2019. System For Early Warning And Monitoring Of Wildfires. Proceedings of the XXVIII International Scientific Conference Electronics.

[8] M. Antunes, *et al*. 2019. Low-Cost System for Early Detection and Deployment of Countermeasures against Wild Fires. IEEE 5th World Forum on Internet of Things..

[9] E. Abdul, *et al.* 2019. Modelling of wireless sensor networks for detection land and forest fire hotspot, TELKOMNIKA. 17(6): 2772-2781.

[10] R. Vega-Rodríguez, Et Al. 2019. Low Cost LoRa based Network for Forest Fire Detection. Sixth International Conference on Internet of Things: Systems, Management and Security.

[11] B. Reynders and S. Pollin. 2016. Chirp spread spectrum as a modulation technique for long range communication. 2016 IEEE Symp. Commun. Veh. Technol. Benelux, SCVT 2016, no. 2, pp. 0-4, doi: 10.1109/SCVT.2016.7797659.

[12] M. Saari, A. Muzaffar Bin Baharudin, P. Sillberg, S. Hyrynsalmi and W. Yan. 2018. LoRa - A survey of recent research trends. 2018 41st Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2018 - Proc., pp. 872-877, doi: 10.23919/MIPRO.2018.8400161

[13] L. Vangelista. 2017. Frequency Shift Chirp Modulation: The LoRa Modulation. IEEE Signal Process. Lett. 24(12): 1818-1821, doi: 10.1109/LSP.2017.2762960.

[14] J. Haxhibeqiri, E. De Poorter, I. Moerman and J. Hoebeke. 2018. A Survey of LoRaWAN for IoT: From Technology to Application. Sensors (Basel)., 18(11), doi: 10.3390/s18113995.

[15] J. De Carvalho Silva, J. J. P. C. Rodrigues, A. M. Alberti, P. Solic, and A. L. L. Aquino. 2017. LoRaWAN - A low power WAN protocol for Internet of Things: A review and opportunities. in 2017 2nd International Multidisciplinary Conference on Computer and Energy Science, SpliTech 2017.

[16] E. Estrada, M. P. M. Vargas, J. Gómez, A. P. P. Negron, G. L. López and R. Maciel. 2019. Smart cities big data algorithms for sensors location. Appl. Sci. doi: 10.3390/app9194196.

[17] B. C. Fargas and M. N. Petersen. 2017. GPS-free geolocation using LoRa in low-power WANs. in GIoTS 2017 - Global Internet of Things Summit, Proceedings, doi: 10.1109/GIOTS.2017.8016251.

[18] A. A. A. Elsabaa, M. Ward and W. Wu. 2019. Hybrid localization techniques in lora-based WSN. in ICAC 2019 - 2019 25th IEEE International Conference on Automation and Computing, doi: 10.23919/IConAC.2019.8895196.

[19] C. Rus, *et al.* 2020. LoRa communication and geolocation system for sensors network. MATEC Web of Conferences 305, 00043.

[20] API v3 (LoRa\textregistered TAO\/RSSI), Online: https://www.loracloud.com/documentation/geolocation?url=v3.html, Last access: June 11th 2020.

[21] N. Sornin, A. Yegin. 2017. LoRaWANTM 1.1 Specification. LoRa Alliance, Inc.

[22] LoRaWAN® Regional Parameters RP002-1.0.0, Online: https://lora-alliance.org/resource-hub/lorawanr-regional-parameters-rp002-100, June 11th 2020.

[23] N. Sornin, A. Yegin. 2017. LoRaWAN Backend Interfaces 1.0 Specification. LoRa Alliance, Inc.