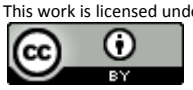
Available online at: https://ijact.in

Date of Submission	11/10/2018
Date of Acceptance	24/10/2018
Date of Publication	31/10/2018
Page numbers	2831-2837 (7 Pages)

This work is licensed under Creative Commons Attribution 4.0 International License.



ISSN:2320-0790

An International Journal of Advanced Computer Technology

IMPROVEMENT, OPTIMIZATION AND COMPARISON OF A REMOTE MONITORING ARCHITECTURE BASED ON WSN APPLICATION TO THE CROSSBOW PLATFORM

[1]Ayoub Marzak, [2]Mohamed Hamraoui
[1, 2]RITM Laboratory, CED Engineering Sciences, Ecole Supérieure de Technologie, Hassan II University of Casablanca, Morocco
[1]ay.marzak@gmail.com, [2]hamraoui@hotmail.com

Abstract: Recently, wireless sensor networks have moved to the concept of hybrid networks. This new concept which merged into Machine To Machine systems, has allowed the wireless sensor networks, to integrate common platforms and exploitable integrals in several types of monitoring and information gathering applications.
In this article, we suggest, in an experimental setting, a remote monitoring architecture with high availability and resilience. It allows one to sense, to process and provide real-time data via hybrid communications technologies.
Several scenarios for data processing and routing of sensed data by the wireless sensor network (ZigBee Technology) were tested and compared in real time in two different environments.

Keywords: Wireless Sensor Networks, Hybrid Architecture, Supervision, Fault Tolerance, Availability, Reliability, Raspberry Pi 3.

I. INTRODUCTION

A Wireless Sensor Network "WSN" is an ad hoc network of many micro-sensor nodes. One can extend their use to hostile or unreachable regions. They can detect, evaluate and connect to different devices in order to collect environment's data. The collected information will be used and taken into account in making decision strategies on an under-surveillance environment. A structural and dynamic model of sensor nodes via UML was approached in [1]. More globally, in [2] the authors were interested in the whole WSN model.
The WSN guarantees an innovative services' set and a better representation of a specific environment. As an example, the evolution of the remote monitoring architecture QoS offers a strategy of acquisition, and continuous sending of measurements to a data processing machine. This is occurring while guaranteeing security by restricting access to authorized users [3].
Among others, various studies have been dedicated to all the questions related to hybrid networks [3], [4], [5], [6], [7], [8], [9], [10]. According to this, the authors of [9] and [10] have implemented a hybrid remote monitoring architecture offering a highly acceptable QoS in terms of availability and resilience. This scheme is based on an architecture model dedicated to Distributed Hypermedia Systems (REST: Representational State Transfer), created by Roy Fielding [11]. REST is a hybrid architecture style based on several models and network concepts, combined with additional specifications.
In this work, firstly, we have improved and optimized a hybrid remote monitoring architecture [9], based on the WSN and various information technologies. In a second

step, we compared the results obtained from its implementation by a standard computer and a Nano-computer (Raspberry Pi 3).
In this paper, the work is organized as follows. After a general introduction, Section II illustrates the improvement of the hybrid remote monitoring architecture. Section III details the implementation of the hybrid architecture in two different environments, and the experimental results followed by a conclusion.

II. IMPROVEMENT OF THE HYBRID REMOTE MONITORING ARCHITECTURE

The objective of this work is to improve the robustness of some existing architectures [9], [10]. We implemented a new version, both in a computer and a nano-computer, to compare the behavior of our algorithms over time (validation of data routing in real time on different systems).

A. Proposed Hybrid Architecture

In this work, we are interested in Zigbee technology [12], [13] (also known as IEEE 802.15.4 [14], [15]), which provides wireless links with low energy consumption.

Let's review the main elements constituting our platform:

- Zigbee [12], [13]: it is a wireless network protocol, like Wi-Fi or Bluetooth, which is suitable for control-and-command´s devices over networks, and other applications requiring low debit but high reliability.
- MQTT [16]: (Message Queuing Telemetry Transport) it is a publish-subscribe messaging service based on the simple and extremely light TCP / IP protocol. It works on the client / server principle. The server, named as the broker, collects the information transmitted by the publishers (Communicating objects).
- HTTPS [17]: (Hypertext Transfer Protocol Secure) it is an Internet communication protocol that protects the integrity and confidentiality of data while transferring information between the client and the server.
- Sockets [18]: it is a model for inter-process communication (IPC) that allows various processes to communicate on the same machine through a TCP / IP network. These sockets will allow managing incoming and outgoing flows to ensure communication between the client and the server.

The architecture proposed by the authors of [9] (Fig. 1) is composed mainly of WSN, a processing server and the users (clients). The processing server is composed of two parts. The first part, reserved for the configuration (web services), allows the platform to receive the hardware and protocol configuration of each connected client in order to generate a polling program. The second part deals with the polling, real-time monitoring of the environment and access to the database in order to visualize the archive of the collected data. This last part is done between the various protocols (MQTT [16], HTTP / HTTPS [17], Sockets [18]), and the communication channels (Wifi [19], Ethernet [20], Bluetooth with low consumption [21], the Internet [22], GSM / GPRS [23]).

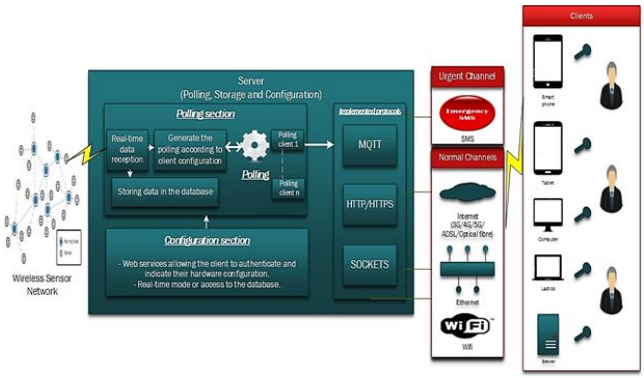


Fig. 1. Hybrid Remote Monitoring Architecture

B. Improvement of the proposed hybrid architecture

The proposed architecture consists of ensuring the local or remote connection of several users to sensors´ network upon different interconnection technologies. It must meet an acceptable level of requirements in terms of availability, reliability and security according to the study case considered.
We approached several issues related to the design of this architecture. We can first mention the adaptation and identification of data from the sensor network before storing it in a data server. The fault tolerance due to a random error is handled in case of emergency (critical values, abnormal overshooting, failures…). The information is routed according, to a priority procedure (passing through an emergency channel), to the appropriate users. Finally, to strengthen the security of the data sent, we are implementing solutions in this regard.

Based on the points cited above, we previously developed an architecture under JAVA, supporting, on the one hand, multiple types of clients (Smartphone, Tablet, Computer, Data Server…) and, on the other hand, several WSN. In this work, we have improved, optimized and evaluated this approach by calculating the packets transmission time as well as its implementation in a Nano-computer (Raspberry Pi 3 Model B [24]). This will give rise to a comparison of our approach on different systems (powerful or limited in terms of the machine resources).

The improved architecture (Fig. 2) is based on:

(i) a modification of fault tolerance strategies, and therefore of the polling algorithm;

(ii) the decomposition of the platform into three distinct servers (processing, storage and configuration).

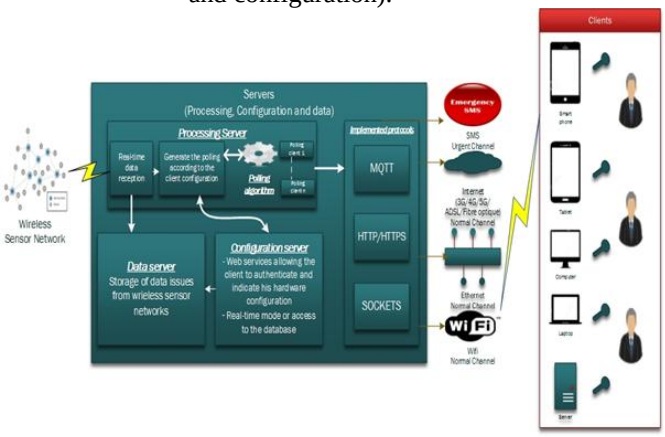


Fig. 2. Enhanced Remote Monitoring Architecture

The configuration server implements the configuration part, the processing server deals with both the data acquisition and the polling part, while the data server is dedicated to storing the data. Our new architecture has been implemented in Raspberry pi 3 Model B with the integration of the low energy Bluetooth channel [21].

The configuration algorithm shown in algorithm 1, describes our configuration sending technique which is done by the connected client, via an interface, and send his physical and protocol configurations. This allows him to generate a program related to our polling strategy (algorithm 2). Thus, the client retrieves, through another interface, the physical values from the WSN.

The polling algorithm (algorithm 2) below, is established in order to better represent our method for the tolerance to the technical breakdowns.

ALGORITHM 1: CONFIGURATION ALGORITHM

```
Type    MQTT = Structure
        URL : String
        TOPIC : String
        End

Type    RES = Structure
        URL : String
        PORT : String
        End

Data    M [3] : MQTT
        R [6] : RES
        GSM1, GSM2 : Integer
        Choice : Boolean
Begin
        For i from 1 to 3
                Read (Choice)
                If Choice = True then
                        Read (M [i] . URL)
                        Read (M [i] . TOPIC)
                End If
        End For

        For i from 1 to 6
                Read (Choice)
                If Choice = True Then
                        Read (R [i] . URL)
                        Read (R [i] . PORT)
                End If
        End For

        Read (GSM1)
        Read (GSM2)
        Send (M, R, GSM1, GSM2)
End
```

ALGORITHM 2: POLLING ALGORITHM

```
Data    Packet : Digit
        Data [ ] : Digital Vector
        Channel : Digital Couple
        Channel_Ref : Digital Couple
Begin
        Packet ← Collect ( )
        Data [ ] ← Process (Packet)
        For i from 1 to 6
        If Urgent (Data [i]) Then
                Send on UrgentChannel (Data [i])
        Else
                Canal ← InitiationSendCycle ( )
                Channel_Ref ← Channel
                SendCycle ← 0
        While (Send (Data [i], Channel) = False) Do
        If TestChannels ( ) = True Then
                Channel ← SearchChannel ( )
        Else
                SendCycle ← SendCycle + 1
        If SendCycle > 5 Then
                Send UrgentChannel (Data [i])
                Send ErrorLog ( )
        Else
                Canal ← Channel_Ref
        End If
        End If
        End While
        End IF
        End For
End
```

WSN allows the collection of physical quantities (Temperature, Pressure…) from the sensor nodes deployed on the field to the base station. These data packets are sent to the registered clients according to the polling algorithm. When collecting a new packet, the system automatically triggers its transmission. This packet is usually processed by a processing interval of priority packets that we set according to the application. If, for example, the temperature detected exceeds the maximum set, the system automatically sends it to the relevant users via the emergency channel (SMS messaging). Otherwise, it initializes and resumes the normal sending cycle via the last reference channel which is always used as long as the data arrives successfully at its destination. In the case of a failure, the channel test procedure is triggered in order to select a new functional channel. If after five attempts, no channel is detected, the system uses the emergency channel which handles both the sending of the data and the report on the failures that have occurred. All packets are transmitted with receipt acknowledgment.

III. IMPLEMENTATION, EXPERIMENTAL TEST RESULTS AND COMPARISON

Our hybrid architecture based on WSN and interconnection means was realized under JAVA.

A. Implemetation

We have implemented a ZigBee WSN platform (CrossbowMicaZ) [25] consisting of 40 sensor nodes based on the microcontroller « MPR2400 » based on « Atmel ATmega128L » from ZigBee-Alliances [12]. The sensor nodes (« MDA 100 » & « MTS 420 ») are connected to the base station in a mesh topology.

First, we collect physical quantities, including temperature, humidity, geo-position of the node relative to the base station and the energy consumed by each node. These data are stored in a server. Then, we perform various tests by a computer and Raspberry Pi 3 with interfaces designed under JAVA (Figs. 3, 4, 5) and a smartphone (Android application). Sending SMS is handled by a GSM / GPRS module [23].

One can access the various services offered by our system (Supervision, History…) by means of set up interfaces. The «Client-Authentication» interface (Fig. 3) allows the user to authenticate and select the desired service. On the other hand, the «Client-Configuration» interface (Fig. 4) makes it possible to insert the desired configuration detailed in algorithm 1. Thus, the polling server generates a program related to the client's request and consequently the data is displayed in the protocol's interfaces. Figures 5 and 6 show the client interface specific to the MQTT protocol and the visualization of the history and statistics of our application.

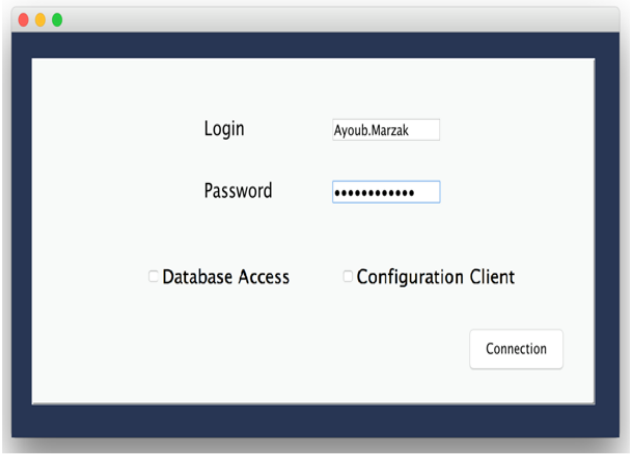


Fig. 3. Client-Authentication Interface

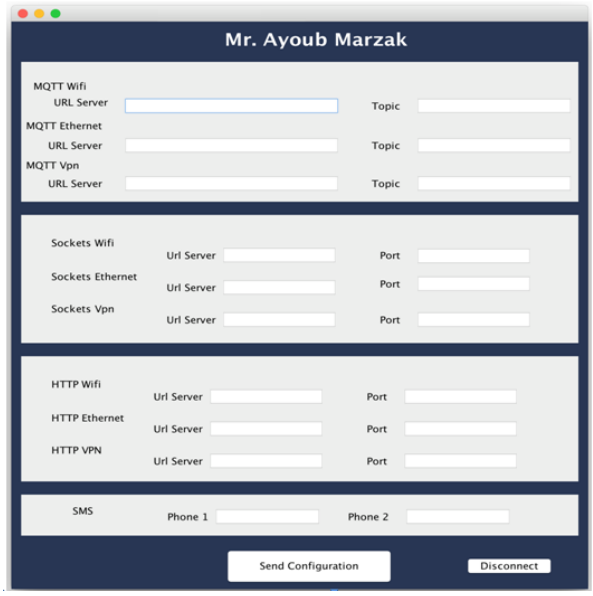


Fig. 4. Client-Configuration Interface

Fig. 5. Client-Protocol MQTT Interface

Fig. 6. Access Interface to the Database

B. Results of experimental tests

In this work, we evaluate and validate our algorithm by the time calculation T of information processing (T = packet sending time + acknowledgment time) under both environments (Computer and Raspberry pi 3). For each device, we calculated the information processing time, the average, the minimum and the maximum time of the packet transmission. These results are based on actual measurements of the experimental tests that we performed on our platform.

The results of the calculation of T are recorded in our database as well as the sensors measurements and their archiving. This allowed us to study and establish graphs (Figs. 6 to 11) for different possible cases of our information processing strategy (algorithm 2).

For the calculation of T we experimentally tested our system on three possible scenarios:

1) 1st Scenario: Normal transmission of packets using the reference channel

Figures 7 and 8 illustrate the evolution of T calculated according to the transmitted packets. This case study corresponds to the processing of information through the reference channel.

On a set of packages:
For the computer environment: the time T varies between 22 ms and 96 ms and the TAverage is 44.35 ms;
For the Raspberry environment: the time T varies between 50 ms and 256 ms the TAverage is 101.62 ms.

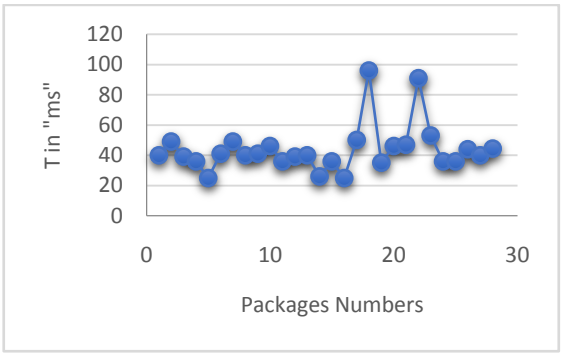


Fig. 7: T Calculated For the Computer Environment

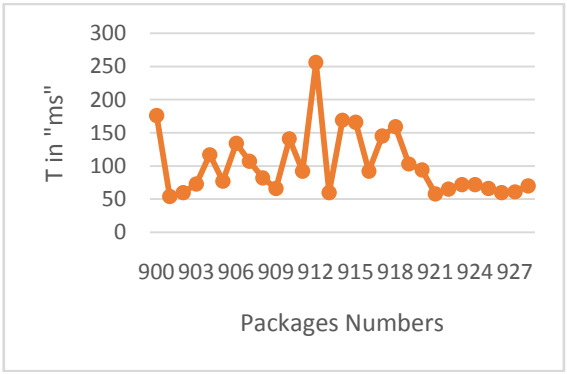


Fig. 8: T Calculated For The Raspberry Pi 3 Environment

2) 2nd Scenario: Transmission with polling to another channel

This scenario addresses the case of polling to another channel following a failure in the reference channel. The evolution of the transmission time T according to the transmitted data is illustrated in figures 9 and 10. We observe that, in this case, T is less than or equal to 1000 ms on the Computer environment and between 1132 ms and 1800 ms on the Raspberry Pi 3.

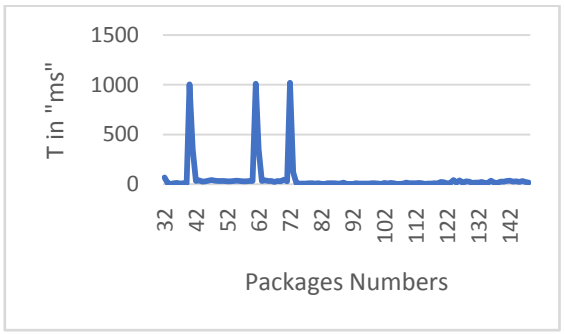


Fig. 9: T Calculating For The Computer Environment

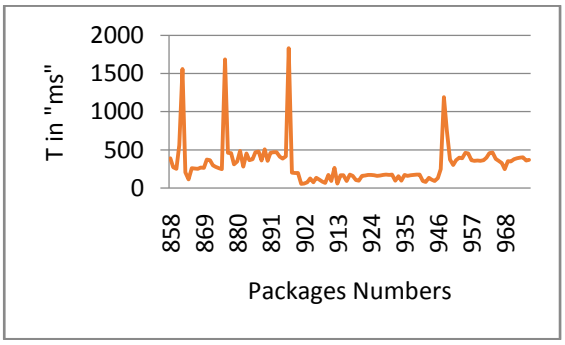


Fig. 10: T Calculating for the Raspberry Pi 3 Environment

3) 3rd Scenario: Transmission of packets using the emergency channel

The figures 11 and 12 represent the variation of the time T according to the packets transmitted in the case of using the emergency channel. The latter is used for:

- Cases of urgent data transmission;
- Cases of data transmission following the failure of all channels.

In the Computer environment: T is around 5000 ms;

In the Raspberry environment: T is between 3000 ms and 5000 ms.

Fig. 11: T calculated for the computer environment

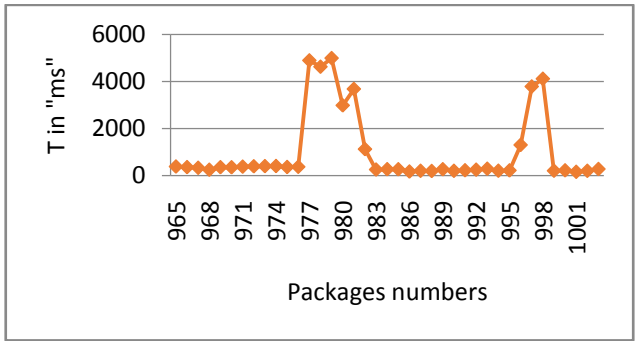


Fig. 12: T Calculating for the Raspberry Pi 3 Environment

C. Comparison of the experimental results in both environments

The experimental tests were performed under both environments (Computer and a Raspberry Pi 3 Nano-computer). For our platform, the results of T were calculated according to 3 possible scenarios (Table 1). We notice that for the 1st and 2nd scenarios (normal case) the times are acceptable and the environment Computer responds quickly to our solution compared to Raspberry Pi 3. However, for the 3rd scenario, where we handle urgent information, Raspberry Pi 3 is faster than the computer.

TABLE 1: COMPARISON OF T ACCORDING TO THE 3 SCENARIOS BETWEEN THE 2 ENVIRONMENTS.

	Computer Environnement			Raspberry Pi 3 Environnement		
	T_{min}	T_{max}	T_{moy}	T_{min}	T_{max}	T_{moy}
1st scenario	22 ms	96 ms	44,35 ms	50 ms	256 ms	101,62 ms
2nd scenario	997 ms	1000 ms	1003 ms	1132 ms	1800 ms	1444,73 ms
3rd scenario		5000 ms		3000 ms	5000 ms	4000 ms

IV. CONCLUSION

In this paper, we have proposed a remote monitoring architecture based on the crossbow WSN that meets certain requirements. It uses a WSN and three hosted servers in a standard computer and a Nano-computer for the acquisition, storage and dissemination of data, taking into account our information processing strategies.

To validate its feasibility, we tested our prototype for the two equipment according to three case studies: the transmission of data by the reference channel, the polling towards other channels and the prevention of failures by using the emergency channel. Following these tests, we compared the results obtained by the two environments.

V. REFERENCES

[1] A. Sahu, E. B. Fernandez, M. Cardei, et M. Vanhilst, "A Pattern for a Sensor Node", in Proceedings of the 17th Conference on Pattern Languages of Programs, New York, NY, USA, 2010, p. 7:1–7:7.

[2] M. Cardei, E. B. Fernandez, A. Sahu, et I. Cardei, "A Pattern for Sensor Network Architectures", in Proceedings of the 2Nd Asian Conference on Pattern Languages of Programs, New York, NY, USA, 2011, p. 10:1–10:8.

[3] A. Marzak, M. Hamraoui, H. Belhadaoui, "Conception et réalisation d'une architecture hybride intégrant des réseaux de capteurs sans fil et technologies d'information et de communication" Mediterranean Telecommunications Journal « MTJ », Vol. 6, N° 2, June 2016.

[4] M. Cardei, A. Marcus, I. Cardei, et T. Tavtilov, "Web-based heterogeneous WSN integration using pervasive communication", in Performance Computing and Communications Conference (IPCCC), 2011 IEEE 30th International, 2011, p. 1‑6.

[5] D. Barata, G. Louzada, A. Carreiro, et A. Damasceno, "System of Acquisition, Transmission, Storage and Visualization of Pulse Oximeter and ECG Data Using Android and MQTT", Procedia Technol., vol. 9, p. 1265‑1272, 2013.

[6] H. Huang, S. Xiao, X. Meng, et Y. Xiong, "A Remote Home Security System Based on Wireless Sensor Network and GSM Technology", in 2010 Second International Conference on Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010, vol. 1, p. 535‑538.

[7] S. Nadir, A. Marzak, K. Lahma, H. Belhadaoui, et M. Hamraoui, "Design and complexity analysis of algorithm treating the credibility of the information: Application to WSN", NNGT Int J Netw. Comput., vol. 2, févr. 2015.

[8] A. Marcus, M. Cardei, I. Cardei, E. Fernandez, F. Frati, et E. Damiani, "A Pattern for Web-based WSN Monitoring (Invited Paper)", J. Commun., vol. 6, no 5, août 2011.

[9] A. Marzak, M. Hamraoui, H. Belhadaoui, "Heterogeneous Networks of Remote Monitoring with High Availability and Resilience Application to Wireless Sensor Networks" The International Journal of Computer Science and Information Security "IJCSIS", Vol. 15 No. 8 Aug. 2017.

[10] A. Marzak, M. Hamraoui, "Architecture de télésurveillance basée sur les réseaux de capteurs sans fils. Application à la plateforme de capteurs CrossBow" Mediterranean Telecommunications Journal "MTJ", Vol. 8, N° 2, August 2018.

[11] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures", University of California, Irvine, 2000.

[12] « The ZigBee Alliance | Control your World ». [En ligne]. Disponible sur: http://www.zigbee.org/.

[13] F. Shariff, N. A. Rahim, et W. P. Hew, "Zigbee-based data acquisition system for online monitoring of grid-connected photovoltaic system", Expert Syst. Appl., vol. 42, no 3, p. 1730‑1742, févr. 2015.

[14] "IEEE-SA - The IEEE Standards Association – Home". [En ligne]. Disponible sur: http://standards.ieee.org/.

[15] C. Suh, Z. H. Mir, et Y.-B. Ko, "Design and implementation of enhanced IEEE 802.15.4 for supporting multimedia service in Wireless Sensor Networks", Comput. Netw., vol. 52, no 13, p. 2568‑2581, sept. 2008.

[16] "IEEE Xplore Abstract - MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks". [En ligne]. Disponible sur: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=4554519&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D4554519.

[17] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, et T. Berners-Lee, "Hypertext transfer protocol--HTTP/1.1. RFC 2616", June, 1999.

[18] K. L. Calvert et M. J. Donahoo, "TCP/IP sockets in Java : practical guide for programmers." Morgan Kaufmann, 2011.

[19] P. Mühlethaler et O. Salvatori, "802.11 et les réseaux sans fil". Eyrolles, 2002.

[20] R. M. Metcalfe et D. R. Boggs, "Ethernet : distributed packet switching for local computer networks", Commun. ACM, vol. 19, no 7, p. 395‑404, 1976.

[21] J. Lee, Y. Su, C. Shen, "A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi", IEEE IECONN, Nov. 2007, pp. 46-51.

[22] J. Postel, "Internet Protocol", 1981.

[23] T. Halonen, J. Romero, et J. Melero, "GSM, GPRS and EDGE performance : evolution towards 3G/UMTS". John Wiley & Sons, 2004.

[24] Jain, S., Vaibhav, A., & Goyal, L. (2014, February). "Raspberry Pi based interactive home automation system through E-mail". In Optimization, Reliabilty, and Information Technology (ICROIT), 2014 International Conference on (pp. 277-280). IEEE.

[25] Crossbow, W. S. N. Professional Kit: http://www. xbow. com. Products/productdetails. aspx.