Department of Information Science and Technology

Reefer Container Monitoring System Based on WSN and Cloud Technology

Peiyao Tang

A Dissertation presented in partial fulfillment of the Requirements for the Degree of

Master in Telecommunications and Computer Engineering

Supervisor:

Dr. Octavian Adrian Postolache, Associate Professor

ISCTE-IUL

Co-supervisor

Dr. Bin Yang, Full Professor

Shanghai Maritime University

September 2019

Abstract

Reefer containers are the main transportation method for the import and export of food and medicine. For high-quality products is necessary to monitor the condition of the reefer containers in order to avoid affecting goods quality due to environmental variations. Monitoring the reefer containers which are used to transport fruits, vegetables, and dairy products is one of the examples. In this context appears the necessity to develop this work expressed by a distributed sensor system for monitoring reefer containers. With the support of the WSN (wireless sensor network) including a set of sensors, it is possible to obtain the information about the temperature, humidity and location data of the reefer container and upload those data to a cloud platform expressed in the case of the purposed system by The Things Network platform.

Based on LEACH (Low Energy Adaptive Clustering Hierarchy) routing algorithm, the embedded software was developed to guarantee a well-balanced distribution of the energy load among WSN end-nodes.

A web application and a mobile application has been developed to display the data coming from the WSN node. To check if the reefer container working in a good condition, an alarm software module has been developed to highlight abnormal data coming for the system.

The routing algorithm has been simulated and the effectiveness of the algorithm is verified by simulation results. The effectiveness of the proposed system was experimentally tested, and several results are included in this dissertation.

Keywords: Reefer Container Monitoring; Wireless Sensor Network; Internet of Things; ZigBee; LoRa

Resumo

Os contentores frigoríficos são o principal método de transporte para a importação e exportação de alimentos e medicamentos. Em produtos de alta qualidade, é necessário monitorizar as condições dos contentores frigoríficos, a fim de evitar a perda da qualidade das mercadorias devido a variações térmicas. Por exemplo, monitorarizando os contentores frigoríficos usados para transportar frutas, vegetais e laticínios. Neste contexto, aparece a necessidade do desenvolvimento deste projeto descrito por um sistema de sensores distribuídos para monitorizar contentores frigoríficos. Com o suporte da rede de sensores sem fios, incluindo um conjunto de sensores, é possível obter informações sobre os dados da temperatura, humidade e localização do contentor refrigerado e fazer upload desses dados numa plataforma em cloud expressa no caso do sistema proposto por plataforma de rede de coisas.

Com base no algoritmo de roteamento LEACH, o software incorporado foi desenvolvido para garantir uma distribuição equilibrada da carga de energia entre os nós de WSN.

Uma aplicação Web e uma aplicação móvel foram desenvolvidas para mostrar os dados provenientes do nó WSN. Para verificar a qualidade dos dados, um módulo de software de alarme foi também desenvolvido para destacar dados anormais que chegam ao sistema.

O algoritmo de roteamento foi simulado e a eficiência do algoritmo é verificada pelos resultados da simulação. A eficiência do sistema proposto foi testada experimentalmente e os vários resultados estão incluídos nesta dissertação.

Palavras-chave: Monitorização de contentores refrigerados; Rede de sensores sem fios; Internet das Coisas; ZigBee; LoRa

Acknowledgments

Firstly, I would like to express my sincere thanks to my advisor, Professor Octavian Postolache and to co-supervisor, for all the availability and support during the realization of this project.

I transmit my acknowledgements to the Telecommunications Institute IT-IUL and ISCTE-IUL, for all the conditions including the material and resources needed for this dissertation.

My biggest thanks go to my family for the unconditional support, especially to my parents, Shouchun Tang and Tingjie Dai.

A big thanks to my girlfriend, Lin Ma, for all the help, support.

A special thanks to my colleagues and friends, Dongchen Ni, Zeyu Ma, Joao Monge, Mariana Jacob Rodrigues, Yongshuang Wang and Yu Jin, who accompanied me on this journey, for all the encouragement and for, in some way, helping me complete this project.

Content

List of Figures

List of Tables

List of Acronyms

API	Application Programming Interface
ATM	Asynchronous Transfer Mode
BLE	Bluetooth Low Energy
CMOS-IC	Complementary Metal-Oxide-Semiconductor Integrated Circuit
CPU	Central Processing Unit
GPS	Global Positioning System
I2C	Inter-Integrated Circuit
ICSP	In-Circuit Serial Programming
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
FSK	frequency shift keying
IoT	Internet of Things
LEACH	Low Energy Adaptive Clustering Hierarchy
LPP	Low Power Payload
LPWAN	Low-Power Wide-Area Network
M2M	Machine-to-Machine
MEMS	Micro-Electro-Mechanical System
OSI	Open System Interconnection
OTAA	Over-The-Air-Activation
P2P	Peer-to-Peer
PCB	Printed Circuit Board
RAM	Random Access Memory
ROM	Read-Only Memory
RF	Radio Frequency
RH'T	Relative Humidity and Temperature
SD Card	Secure Digital Card
SoC	System on Chip
SPI	Serial Peripheral Interface
SNR	Signal-Noise Ratio
TTN	The Things Network
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Networks
WSN	Wireless Sensor Network

Chapter 1 - Introduction

1.1 Facts and Motivation

Since the 1990s, the global economy has changed and developed rapidly. Economic globalization and trade liberalization have become the trend of world economic development. This trend has effectively promoted the development of the international cargo transportation industry, which has led to a rapid increase in the volume of goods transported around the world. As one of the main modes for international cargo transportation, container transportation has also shown a good development trend. Almost 90% of the world trade is accomplished with the help of containers and they can be used in different means of transportation, including ships and trains [1].

With the progress of various technologies, people have higher and higher requirements for the quality of container transportation. In order to meet the various demands of international trade, the government, enterprises, universities and research institutes put a high value on container transportation research. All parties have got involved in the research and application of various technologies for container transportation. At present, the problems involved in container transportation mainly include:

- the overall optimization of container flow,
- resource allocation of port handling equipment,
- transportation route problem,
- space allocation research in the container yard,
- container monitoring technology.

One of the research hot topics in container monitoring is to study the reefer container monitoring system. Environmental factors such as temperature variations and humidity variations may affect the quality of transported goods during transportation. Sometimes the limit of the temperature and humidity can be quite strict, for example, as soon as the storage temperature is beyond 0 ℃, the strawberry has a risks growth of microbial. Thus, the reefer container should be used to maintain the quality of the food and medicine. At the same time, it is necessary to monitor the condition of the reefer container to ensure that the reefer container is working properly. For example, monitor

the reefer containers which are used to transport fruits, vegetables, dairy products to ensure the quality of the goods. Therefore, it is necessary to design and implement a temperature, humidity, and location monitoring system for reefer containers [2].

To monitor the reefer container, the MSc research work uses various technologies to increase performance and improve the practice of the reefer container monitoring system. One of these technologies is the WSN which uses WSN node placed on the reefer container to collect information and transfer data during transportation. Using a data information system, it is possible to store and manage the reefer container state information. All of the state information can be shown on a web application and a mobile application.

1.2 Objectives

In this dissertation, it is proposed to design and implement the hardware and software of a reefer container monitoring system, in which several sensors and RF wireless modules have been considered attached to the reefer container.

The goal is to transfer the state information of the reefer container from the WSN nodes to the data information system and display the state information.

In order to solve the problem that some radio communication links of the WSN node may be blocked by other containers, two different RF wireless modules were considered to design the network architecture of the WSN.

A routing algorithm was considered to extend the stability period (time interval before the death of the first WSN end node) of WSN which focuses on guarantee a well-balanced distribution of the energy load among nodes of the sensor network.

A web application and a mobile application was developed to present the state information coming from the WSN end node. An alarm system was developed to alarm for abnormal temperature and humidity information.

1.3 Structure of the dissertation

Chapter 2 includes the literature review on the Internet of Things, Wireless Sensor Networks and how these concepts can be used to monitor reefer containers. Chapter 3 describes the developed system as a whole and how everything is connected. The hardware structure of this project is described in Chapter 4. Chapter 5 includes a description of the routing algorithm and the embedded software. Chapter 6 is the

experimental validation and results. The conclusions and future work are presented in Chapter 7.

Chapter 2 – State of Art

2.1 Container

In the 1940s, containers were introduced in the United States. In the early 1950s, the United States adopted the boxback style and piggyback style to adapt to the rapid development of road and rail transportation, that is, the container semi-trailer or container was loaded onto the railway flatcar for transportation [3]. This approach has laid the foundation for container-mediated “door-to-door” transport and road-rail intermodal. Inspired by the container road-rail intermodal, in 1995, the United States began to use the shipping container. In 1990, container shipping accounted for at least 75% of all liner trade, accounting for almost all of the Asian and European trade. As the preferred method of bulk goods and international freight transportation, container shipping is an unprecedented revolution in the way goods are transported. Since the container came into being, the transportation industry has been keeping a close eye on it. Due to the economic downturn, the container market has experienced a period of recession. In 2005, with the recovery of the world economy, the container market began to expand rapidly.

2.2 Reefer Container

With the rapid development of international logistics, the demand for global cross-border cold chain transportation has been very strong in the past decade. With the rapid development of international logistics, the demand for global cross-border cold chain transportation has been very strong in the past decade. Since the trade of globalization is expanding and continuously enhancing, people's concepts of consumption are changing, global cold chain transportation is used in more and more situations. For example, southern hemisphere countries such as South Africa and Australia provide fresh agricultural and sideline products (fruits and vegetables, fish and dairy products) for the consumer market such as Europe, America and Asia [4]. Little girls in China can eat fresh bananas from South America and little boys in Brazilian have a chance to taste ice cream from the United States. In international logistics, these frozen goods are mainly transported by reefer containers [5].

At present, types of transportation equipment which are suitable for cold chain logistics include road reefer trucks, railway reefer trucks, reefer vessels and reefer containers. Reefer containers are main transportation equipment in international cold chain transportation. The reefer container is suitable for container intermodal transportation and it has the advantages of flexible operation, convenient transportation, sturdiness and durability. It also has a microcomputer controller and a refrigeration unit.

The reefer containers have gradually replaced traditional refrigerated transport vehicles such as reefer ships and reefer trucks. According to Danameier Maritime's report, in 2015, the total volume of international shipping containers was more than 20 million TEU, of which more than 2 million TEUs were reefer containers.

According to the market report released by Drewry, global seaborne reefer trade continues to expand, with a gain of over 5% in 2017 to 124 million tonnes, containerized reefer traffic expanded by 8% in 2017. Therefore, in the foreseeable future, the global cold chain transportation industry with reefer container transportation as the main mode of transportation will continue to expand [6].

Reefer container, shown in Figure 2.1, consists of a refrigeration unit, Microcomputer controller and a container.

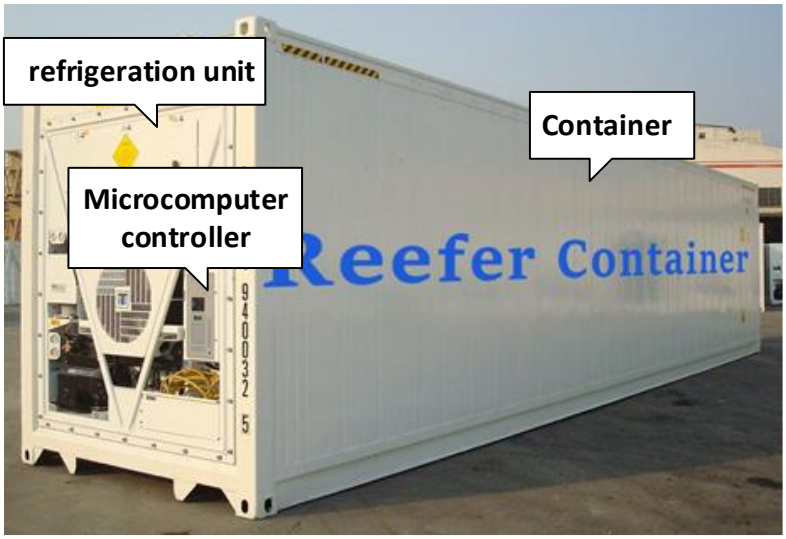


Figure 2.1 - Reefer container

In international shipping, the most common is the 40’ reefer container (as shown in Figure 2.2).

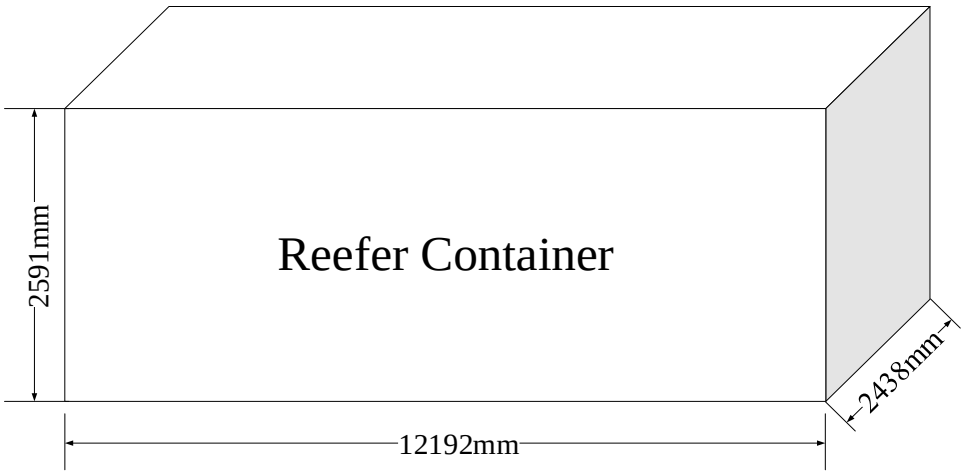


Figure 2.2 - Size of the standard reefer container

In order to measure the temperature and humidity in the reefer container, some sensors were considered. Some researchers selected the Sensirion SHT30 temperature and humidity sensor [7]. Sensirion SHT11 also be considered by other researchers [8]. Table 1 shows a comparison between different relative humidity and temperature sensors.

Table 1 Comparison between different relative humidity and temperature sensors

Manufacturer Part Number	SHT11	SHT30-DIS-B2.5KS	SI7021
Manufacturer	Sensirion AG	Sensirion AG	Silicon Labs
Description	SENSOR HUMID/TEMP 5V DTL 3% SMD	SENSOR HUMID/TEMP 5V I2C 3% SMD	SENSOR HUMID/TEMP 3.6V I2C 2% 6DFN
Quantity Available	18652	2971	7500
Unit Price (EUR)	10.17	3.52	2.67566
Minimum Quantity	1	1	2500
Packaging	Cut Tape (CT)	Cut Tape (CT)	Tape & Reel (TR)
Part Status	Obsolete	Active	Active
Sensor Type	Humidity, Temperature	Humidity, Temperature	Humidity, Temperature
Humidity Range	0 ~ 100% RH	0 ~ 100% RH	0 ~ 100% RH
Output Type	Digital	I²C	I²C
Output	12b	16b	12b

Accuracy(typical)	±3% RH,±0.4	±2% RH,±0.2	±2% RH,±0.3
Response Time	8s	8s	18s
Voltage – Supply	2.4V ~ 5.5V	2.15V ~ 5.5V	1.9V ~ 3.6V
Mounting Type	Surface Mount	Surface Mount	Surface Mount
Operating Temperature	-40°C ~ 123°C	-40°C ~ 125°C	-40°C ~ 85°C
Package / Case	10-SMD Module	8-VFDFN Exposed Pad	6-WDFN Exposed Pad
Supplier Device Package	10-LCC	8-DFN (2.5x2.5)	6-DFN (3x3)

Compared with SHT30 and SHT11, the operating temperature range of Si7021 is smaller, but the operating range (-40°C to 85°C) is enough to work in the reefer containers. The measurement accuracy of Si7021 is between SHT30 and SHT11, but the Si7021 is the cheapest. Thus, Si7021 is selected to be the RH'T sensor of the reefer container monitoring system in the presented work.

2.3 Internet of Things (IoT)

Internet of Things (IoT) is an extended network based on the Internet. It integrates various information sensing devices with the Internet to consist of a huge network to realize the interconnection of people, machines and objects [9]. The Internet of Things is the "Internet which connected by things", it is an important part of the new generation of information technology. The core and foundation of the Internet of Things is still the Internet and the IoT can be considered as an extended network based on the Internet. The client of the IoT extends to any item to connect the things and make things to communicate with each other. Therefore, the Internet of Things is a network that connects any sensing devices such as radio frequency identification, infrared sensors, global positioning systems, laser scanners. It's a network that can exchange information to achieve intelligent identification, location, tracking, monitoring and management of things in this network [10]. In this way, Machine to Machine interaction can be done (as shown in Figure 2.3).

Figure 2.3 - Internet of Things (IoT)

2.4 Embedded system

Embedded systems are based on applications. It is a special computer system in which the software and hardware system can be trimmed to meet the strict requirements of application system on functions, reliability, cost, size, power consumption and other comprehensive performance. It integrates software and hardware and could work independently. Unlike general-purpose computer systems such as personal computers, embedded systems usually perform predefined tasks with specific requirements. Because the embedded system is only for a specific task, the designer can optimize it to reduce the size and cost [11].

Embedded systems have high requirements for speed and reliability, so the software of the embedded system is usually stored in a ROM (Read-Only Memory) chip. Because the storage capacity of the ROM chip is limited, the embedded system requires that the length of the program code should be minimized while maintaining the executive function and speed. Unlike general-purpose computers that are capable of running user-selected software, software on embedded systems is usually temporarily unchanged, so it is often referred to as firmware. Some embedded systems also include an operating system, but most of the embedded systems implement full control from a single program.

Embedded systems control a lot of devices in daily use today. In fact, almost all devices with digital interfaces, such as watches, microwave ovens, video recorders, and automobiles, use embedded systems [12].

2.5 The Arduino

Arduino is an easy-to-use and open-source electronics prototyping platform that includes hardware (various models of Arduino board) and software (Arduino IDE). It was developed in the winter of 2005 by a European development team. Arduino board is built on the open-source simple I/O interface and has a Processing/Wiring development environment similar to Java and C.

Arduino can sense the environment variables through a variety of sensors and can feedback and influence the environment such as controlling lights, motors and other devices through actuators.

Just write the program code in the Arduino IDE and upload the program code to the Arduino board, the program code will be executed on the Arduino board. The microcontroller on the board can be programmed in Arduino's programming language, the program code will be compiled into a binary file and burned into the microcontroller [13]. Projects can be expanded through a lot of C and C++ libraries available online. Solutions in LabVIEW [14] and even nowadays in Python [15] also exist.

Within the Arduino, there are multiple choices to make projects. The choice of the model falls on the characteristics, dimensions, weight and price. Table 2 [16] shows the available Arduino models as well as the mentioned features.

Table 2 Comparison between different Arduino's types

Name	Processor	Operating /Input Voltage	CPU Speed	SRAM [kB]	Flash [kB]	USB	Price (€)
101	Intel® Curie	3.3 V/ 7-12V	32MHz	24	196	Regular	28.65
Gemma	ATtiny85	3.3 V / 4-16 V	8MHz	0.5	8	Micro	Discontinued
LilyPad	ATmega168V/ATmega328P	2.7-5.5 V /2.7-5.5 V	8MHz	1	16	-	17.95
LilyPad SimpleSnap	ATmega328P	2.7-5.5 V /2.7-5.5 V	8MHz	2	32	-	17.95

LilyPad USB	ATmega32U4	3.3 V / 3.8-5 V	8MHz	2.5	32	Micro	21.95
Mega 2560	ATmega2560	5 V / 7-12 V	16MHz	8	256	Regular	35
Micro	ATmega32U4	5 V / 7-12 V	16MHz	2.5	32	Micro	18
MKR1000	SAMD21 Cortex-M0+	3.3 V/ 5V	48MHz	32	256	Micro	30.99
Pro	ATmega168/ATmega328P	3.3 V / 3.35-12 V5 V / 5-12 V	8MHz/ 16MHz	1 2	16 32	-	Discontinued
Pro Mini	ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8MHz 16MHz	2	32	-	Discontinued
Uno	ATmega328P	5 V / 7-12 V	16 MHz	2	32	Regular	20
Zero	ATSAMD21G18	3.3 V / 7-12 V	48 MHz	32	256	2 Micro	39
Due	ATSAM3X8E	3.3 V / 7-12 V	84 MHz	96	512	2 Micro	34
Esplora	ATmega32U4	5 V / 7-12 V	16 MHz	2.5	32	Micro	Discontinued
Ethernet	ATmega328P	5 V / 7-12 V	16 MHz	2	32	Regular	39.99
Leonardo	ATmega32U4	5 V / 7-12 V	16 MHz	2.5	32	Micro	18
Mega ADK	ATmega2560	5 V / 7-12 V	16 MHz	8	256	Regular	43
Yùn	ATmega32U4 AR9331 Linux	5 V	16 MHz 400MHz	2.5 16MB	32 64MB	Micro	17.27

Compare with other board, Arduino Uno has a good characteristics and also has a lower price. Thus, in the present work, the Arduino UNO was used.

2.6 The raspberry Pi

Raspberry Pi is a credit-card-sized single-board computer made by the Raspberry Pi Foundation.

Raspberry Pi is an ARM-based microcomputer motherboard with SD/MicroSD card as the hard disk. There are 1/2/4 USB ports and a 10/100 Ethernet interface (type A without network port) around the motherboard. Keyboard, mouse, network cable, TV output interface with video analog signal and HDMI high-definition video output interface, all of the above components are also integrated on a motherboard that is only slightly larger than the credit card.

Raspberry Pi can run the operating system. The Raspberry Pi Foundation provides a distribution of ARM-based Debian and Arch Linux for public download. With most of the PC's basic functions, raspberry Pi can perform functions such as spreadsheets, word processing, playing games, playing HD videos. It supports Python as the main programming language and is also supports programming languages such as Java and C.

Raspberry Pi is widely used in IoT systems. For instance based on Raspberry Pi, a Wireless Sensor Network for automatic Irrigation and Security systems was implemented [17]. Raspberry pi with a Pi Camera also can be used to implement Open CV to create a mini-standalone station for counting people [18].

2.7 The Orange Pi

Orange Pi is an open-source [19] single-board computer, an ARM development boards. It can run Android4.4, Ubuntu, Debian and other operating systems and is also compatible with the Raspberry Pi. Orange pi uses All Winner H3 SoC (System on Chip) and has 1GB of DDR3 memory.

It can be used to build a computer, a wireless network server, recreational machines, music player, HD video player, speaker, Android and Scratch. The Orange Pi PC can handle 4K video which is not even possible with the Raspberry Pi. The Hardware Specification is shown in Table 3.

Table 3- Orange Pi Hardware Specific Sheet

Item	Descriptions
CPU	ARM® Cortex™-A7 Dual-Core
GPU	ARM® Mali400MP2, Complies with OpenGL ES 2.0/1.1
System Memory(SDRAM)	1GB DDR3 @960M

Onboard Storage	up to 64GB on TF slot, up to 2T on 2.5 SATA disk (The default is no Nand Flash)
Onboard Network	10/100/1000 ethernet, wifi 802.11 b/g/n
Video Output	CVBS and HDMI ,RGB/LVDS,VGA
Audio I/O	3.5 mm jack，PHOUT
USB	Four USB 2.0 HOST, one USB 2.0 OTG
Expansion	Extensible 26-pin headers, Camera connector, Display connector for LVDS and touch screen
Dimensions	112mm X 60 mm
Weight	60g

2.8 Wireless communication

Wireless communication refers to the communication between multiple nodes performed by radio without conductors or cables. Wireless communications include a variety of fixed, mobile and portable applications such as two-way radios, mobile phones, personal digital assistants and wireless networks. Examples of other radio wireless communications include GPS, garage door remotes, and wireless mouse. Most of the wireless communication technologies use radio, including short distance technologies such as Wi-Fi and also include deep space networks with a communication distance of more than several million kilometers. However, some wireless communication technologies do not use the radio but use other electromagnetic waves wireless technologies, such as light, magnetic fields, electric fields, and the like [20].

Wireless Network refers to any type of radio computer network and it is generally combined with the telecommunications network. Wireless telecommunication networks are generally used in remote information transmission systems that using electromagnetic waves (such as radio waves) as a carrier and physical layer network. Wireless Networks normally are Wireless Local Area Networks (WLAN), Wireless Personal Area Network (WPAN) and Wireless Sensor Networks (WSN).

2.9 Wireless sensor networks

In recent years, wireless sensor networks have become a research hotspot, especially the rapid development of MEMS (Micro-Electro-Mechanical System) has

promoted the birth of smart sensor nodes. A smart sensor is a digital sensor with an integrated microprocessor that includes some logic functions and/or can make certain types of decisions [21]. A smart sensor is defined as a sensor that can manipulate and calculate sensor-derived data and communicate data to the user via a communication interface (using a two-way digital bus) [22]. Compared to traditional sensor nodes, the smart sensor nodes are smaller.

Wireless sensor network is a network composed of smart sensor nodes integrated with a data processing unit and a communication module. A variety of types of sensors integrated by smart sensor nodes to sense environmental parameters, such as air temperature, soil moisture, relative humidity, object motion speed and pressure. Batteries are the main source of energy for these smart sensor nodes, and some nodes have secondary energy sources that obtain energy from the environment, such as solar panels.

2.9.1 Wireless Sensor Network Architecture

The common wireless sensor network consists of a large number of sensor nodes includes a few aggregation nodes and management nodes. The sensor nodes located in the monitoring area are responsible for collecting information and then transmitting the data to the aggregation node.

A sensor node is generally an embedded system with low processing power, storage capacity and communication capabilities. In most cases, sensor nodes are powered by batteries with limited energy. Each sensor node can implement the functions of terminals and routers that the same as traditional network nodes. In addition to local information collection and data processing, it also stores, manages, integrates data forwarded by other nodes, and works with other nodes to accomplish specific tasks. Compared with sensor nodes, the aggregation nodes have stronger processing, storage, and communication capabilities, but the basic structure is similar. The aggregation node is mainly responsible for connecting the sensor network with the external network, realizing the communication protocol conversion between different protocol stacks, and realizing the monitoring tasks of the management node. The architecture of the wireless sensor network is shown in Figure 2.4.

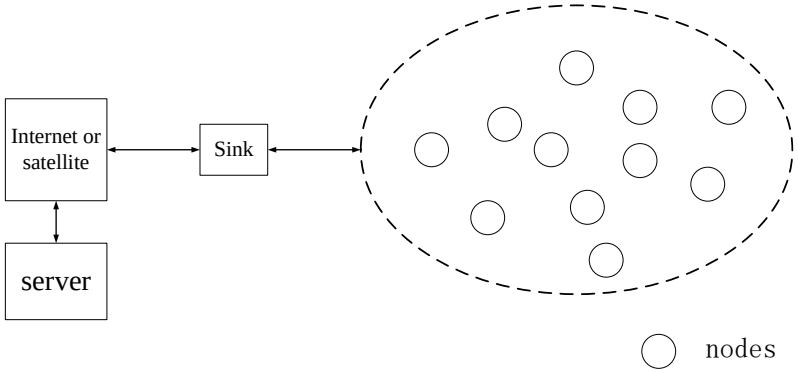


Figure 2.4 - The architecture of Wireless Sensor Network

In a different background, the architecture of WSN is different. But they have similar modules, include sensor module (sensor, signal conditioning and digital-to-analog), processing module (CPU and memory), wireless communication module and power supply module. The architecture of WSN sensor node is shown in Figure 2.5.

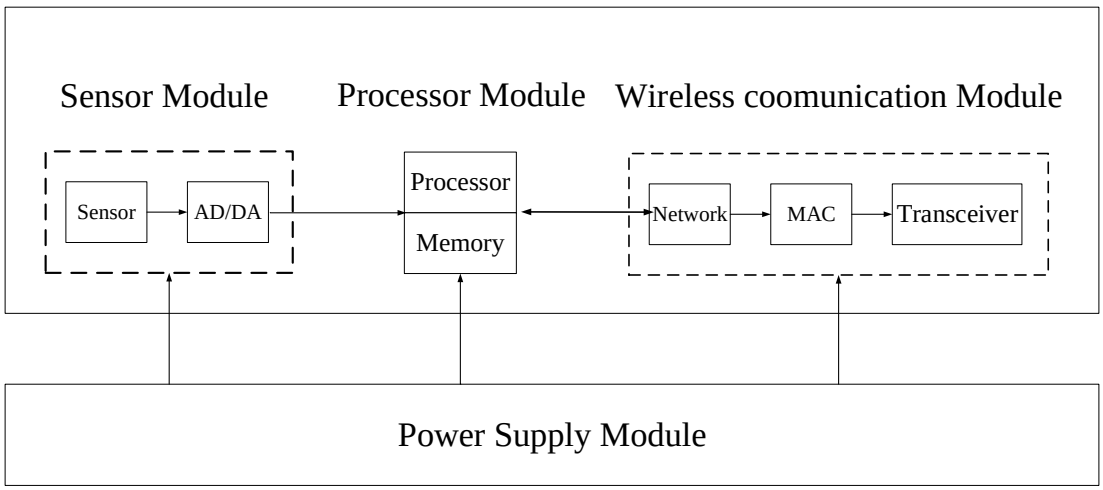


Figure 2.5 - The architecture of WSN sensor node

2.9.2 Protocol Stack for Wireless Sensor Networks

Most wireless sensor networks are designed for applications and have a simple network structure. As shown in Figure 2.6, the typical structure includes a physical layer, data link layer, network layer, transport layer, application layer, protocol stacks of energy management, and mobility management and task management which is similar to the 7-layer model of OSI (Open System Interconnection).

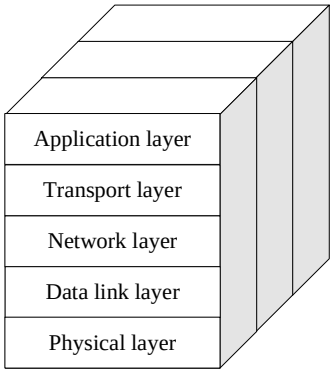


Figure 2.6 - The functions of the protocol layers

The functions of the protocol layers are as follows:

(1) The physical layer provides a simple and robust signal modulation and wireless transfer function for the entire network. Physical layer is used to collect data and simply process the data. The physical layer also completes the modulation and demodulation of the signal, the transmission and reception of the signal, and the power control;

(2) The data link layer is mainly responsible for communication data framing, frame detection, media access and error control. It also achieves access control and establishes a reliable communication link between nodes;

(3) The network layer is primarily responsible for route generation and routing. Nodes generally use multi-hop routing to connect to other nodes because of multi-hop routing suits for information integration and cooperative signal processing;

(4) Transport layer is mainly responsible for the transmission control of the data stream to ensure the quality of the communication service;

(5) The application layer includes various sensor network application software systems to provide an effective software development environment and tools for users to develop sensor network application software.

The functions of the protocol stack platforms are as follows:

(1) Energy management platform manages energy;

(2) The mobility management platform is responsible to discover and register mobile sensor nodes. In this way, nodes can track and identify their neighbors, maintain the route of the mobile node to the sink node, and ensure that the sensor node can dynamically track the location of its neighbor nodes;

(3) The task management platform schedules tasks and time plans for sensor nodes in a specific area.

There are two main types of wireless sensor networks: organized networks and unorganized networks. In an unorganized wireless sensor network, sensor nodes are

densely distributed in the monitoring area in a similar manner. Once deployed, these nodes implement monitoring tasks in an unorganized manner. In the case that a WSN has a large number of nodes, it is difficult for an unorganized network to effectively manage each node device in the network and eliminate the failed node. In an organized wireless sensor network, each node is preset to a defined role, which can effectively reduce network management and maintenance costs. The short-range wireless communication protocol have ZigBee, BLE, WI-FI, etc. The long-range communication wireless communication protocol have LoRa, SigFox, NB-IoT, etc.

2.9.3 ZigBee Network

ZigBee is a standard that defines a set of communication protocols for the low-data-rate short-range wireless networking [23]. ZigBee-based wireless devices operate in 868 MHz, 915 MHz, and 2.4 GHz frequency bands. The maximum data rate is 250 K bits per second. ZigBee is targeted at battery-powered applications that require low data rates, low cost and long battery life. In many ZigBee applications, the total time that a wireless device engages in any type of activity is very short. The device spends most of the time in power-saving mode (also known as sleep mode). As a result, ZigBee-enabled devices can run for a few years before they need battery replacement [24].

ZigBee's structure is divided into four layers: the physical layer, MAC layer, network/security layer and application/support layer. The application/support layer and the network/security layer are defined by the ZigBee Alliance, while the MAC layer and the physical layer are defined by the IEE802.15.4 protocol. The following is the role of each layer in the ZigBee structure:

Physical layer: As the underlying protocol of the ZigBee protocol structure, it provides the most basic services for the upper MAC layer, such as data interfaces. It also plays a role in interacting with the real (physical) world;

MAC layer: MAC layer is responsible for the establishment, maintenance, termination, and acknowledgment of data transmission and reception of wireless data links between different devices;

Network/security layer: It guarantees the transmission and integrity of data, and encrypts data at the same time;

Application/Support Layer: It communicates between multiple devices based on

design goals and needs.

AS shown in Figure 2.7, the star topology network is a very simple centralized communication scheme. The biggest feature of this topology is that the communication of any two nodes needs to rely on the auxiliary forwarding of the coordinator to complete the communication, even if the two nodes are very close.

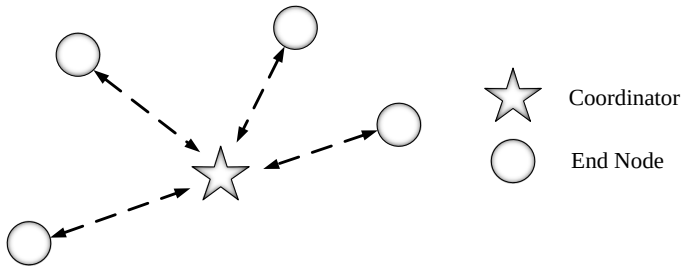


Figure 2.7 - ZigBee star topology

As shown in Figure 2.8, a typical feature of a tree topology is that an end node can only send data to its parent node, and the router can only continue to send data to its parent node when communicating with other external nodes (it is not a child node of the router). Until one of the parent nodes of the destination node is encountered.

Figure 2.8 also shows an example of how relaying messages can help extend the network or even bypass obstacles. For example, device A needs to send a message to devise D, but there is an obstacle between them and the signal is difficult to penetrate. The tree topology provides assistance by relaying messages near the barrier and reaching device D (from A to B, B to C and C to D). Sometimes this is referred to as multi-hop because a message jumps from one node to another until it reaches its destination. When devices working in this way, higher coverage comes at the cost of potentially high message latency.

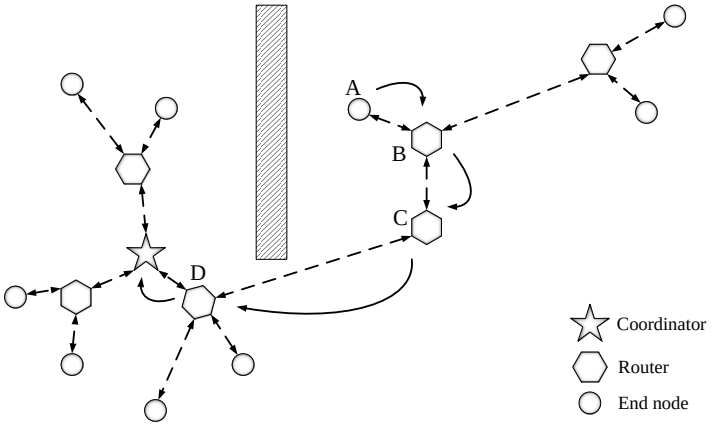


Figure 2.8 - ZigBee tree topology

In ZigBee Mesh topology (shown in Figure 2.9), the network is basically similar to the tree topology structure's association process. The biggest feature of the Mesh structure is that routers can communicate with each other without having to communicate through their parent nodes.

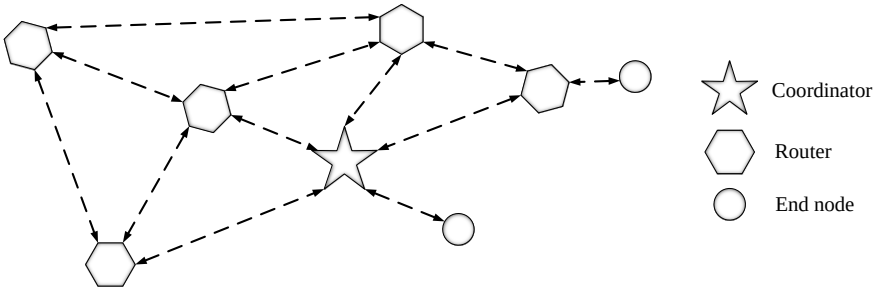


Figure 2.9 - ZigBee Mesh topology

In the present work, the ZigBee Mesh network was used.

2.9.4 LoRa and LoRaWAN

LoRa is the physical layer or wireless modulation used to create remote communication links [25]. Many conventional wireless systems use frequency shift keying (FSK) modulation as the physical layer because it is a very efficient modulation that achieves low power consumption. Based on linear frequency modulation spread spectrum modulation, LoRa maintains the same low power characteristics as FSK modulation but greatly increases the communication range.

LoRa has long range capabilities. The LoRa Gateway can cover hundreds of square kilometers (can cover a small port of shipment). The range is largely dependent on the environment or obstacles in a given location, but the link budget of LoRa is larger than any other standardized communication technology.

As shown in Figure 2.10, LoRaWAN defines the communication protocol and system architecture of the network, while the LoRa physical layer enables remote communication links [26].

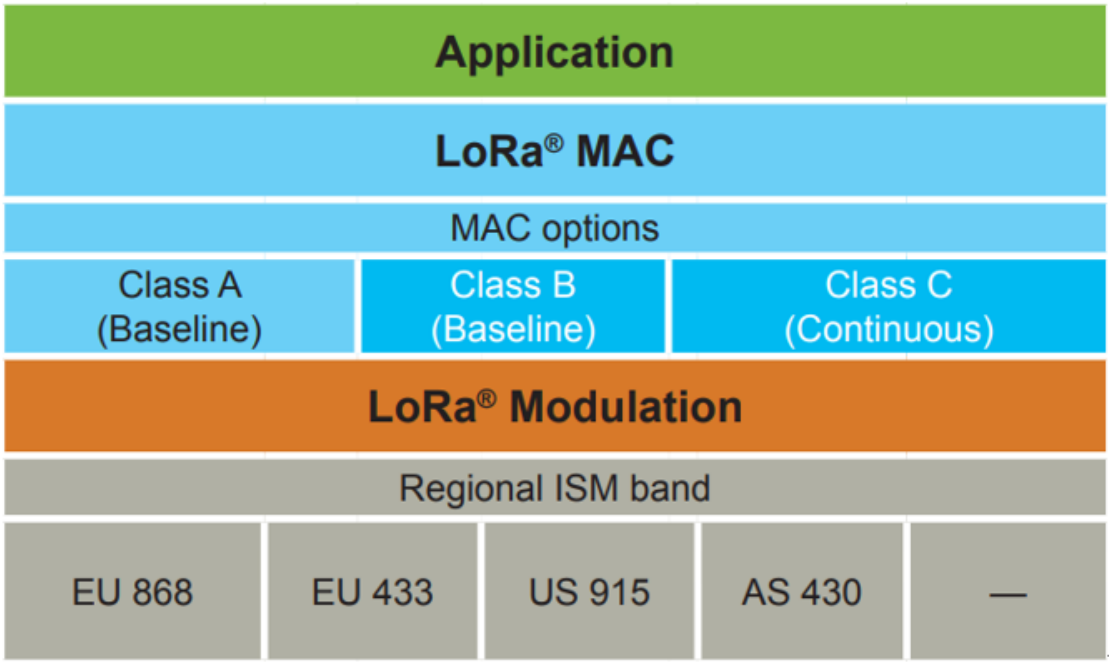


Figure 2.10 - System architecture for the network

Protocols and network architectures have the greatest impact on determining node battery life, network capacity, quality of service, security, and various applications for network services. Many existing deployed networks utilize a Mesh network architecture. In a Mesh network, each end node forwards information of other nodes to increase the communication range and the size of the network. Although it increases the range, it also increases complexity, reduces network capacity and shortens battery life because nodes receive and forward information from other nodes will consume energy. Thus, when realizing the long-distance connection, the most significant thing about long-distance star architecture ((shown in Figure 2.11)) is to protect battery life.

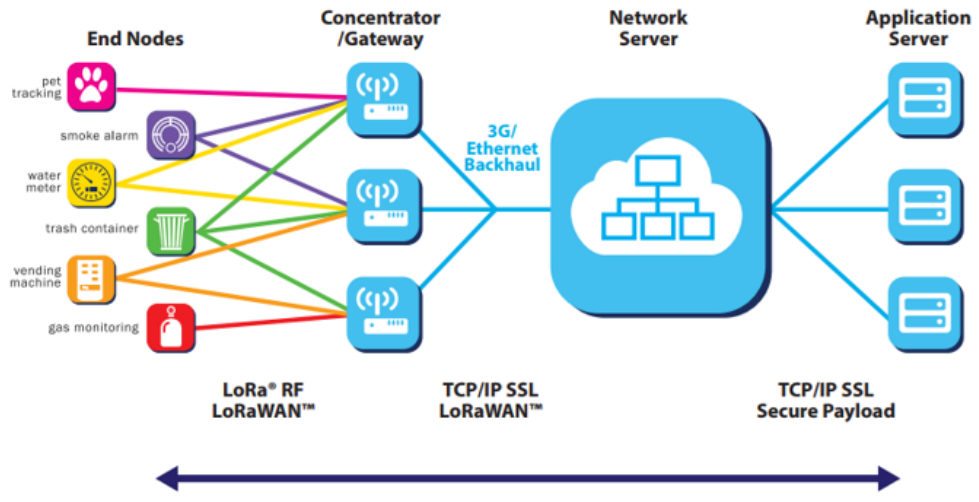


Figure 2.11 - The long range star architecture

2.9.5 The Things Network

The Things Network (TTN) is the LoRaWAN web server. Registering an end node or application with the server is free and the network traffic is also free. But services are based on best effort, so uptime or latency cannot be guaranteed. The Things Network encourages users to develop networks by making registration gateways extremely simple [26]. If the physical location and antenna location of the LoRa gateway is provided when registering the gateway, The Things Network can estimate the network coverage and map it to the user.

The back-end system of the TTN is used to routing data between applications and devices. Normally, a gateway is necessary to an IoT network, it can be a bridge between the Internet and a particular radio protocol. LoRaWAN is a kind of non-IP protocols, thus, the LoRaWAN cannot directly deliver the data to the application server. It is necessary to employ some routing and processing to forward data to the application server. The Things Network is located between the gateway and the application (as shown in Figure 2.12) [27].

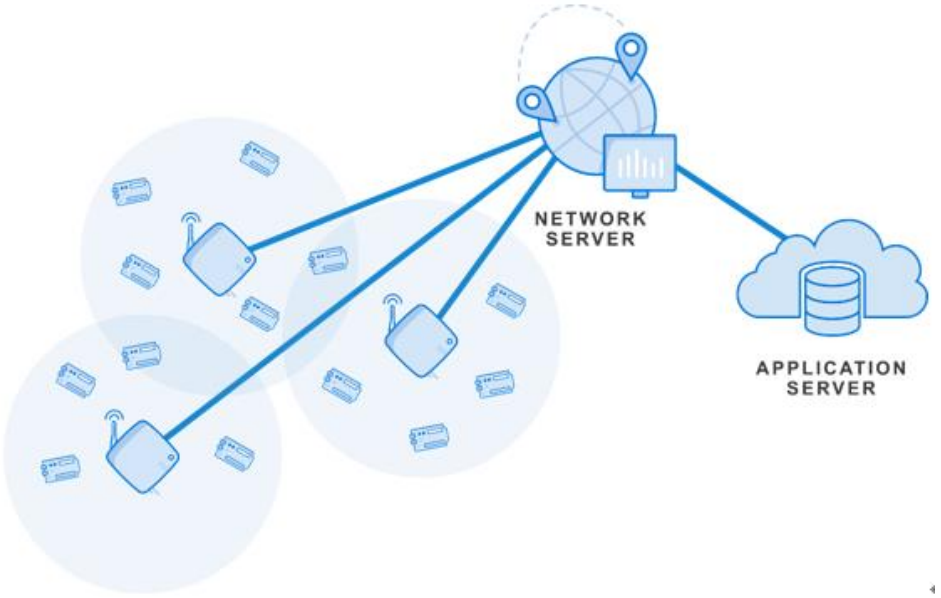


Figure 2.12 - Packet upload flow through TTN

2.9.6 Cayenne

Cayenne is designed to build a drag-and-drop IoT project. It helps developers, designers, and engineers to easily prototype and connect devices that be used to build

the projects. The project which is built under the help of the Cayenne also can be put into production. The platform mainly consists of two parts, they are:

Cayenne Mobile App – Remotely monitor and control the IoT project from an Android or iOS app;

Cayenne Online Dashboard – Use customizable widgets to visualize data, set rules and schedule events.

All of the historical data will be stored in Cayenne and meaningful data can be viewed or displayed. Thus, it's easy to check if all of the devices are working well or if the data has some problems.

From the data history chart, the Cayenne also can filter by different time ranges. Depending on the time range selected, the data display will be updated to show more or less detail. Cayenne can choose from any available preset range and the data be displayed also may depend on the data currently available to the device [28].

Cayenne has a tool called Cayenne Low Power Payload (LPP). With the help of the Cayenne Low Power Payload (LPP), it is an easy thing to send data over LoRaWAN. The Cayenne LPP meets the payload size limit, which can be reduced to 11 bytes and allows the device to send multiple sensor data at once. In addition, Cayenne LPP also allows devices to send different sensor data in different frames [29].

Chapter 3 – System Description

3.1 Overview

The architecture of the reefer container monitoring system has three parts (as shown in Figure 3.1). In first part, our system is expressed by a ZigBee network. Each WSN end node is expressed by smart sensors distributed on the level of the refer container. These nodes can acquire data and transfer data between nodes in the network.

The second part consists of a LoRa Gateway and the server. The LoRa available WSN end node transmits data to the LoRa Gateway through LoRa. The LoRa Gateway uploads data to the database by using the Internet.

The third part represents a web application that access the information in the cloud, interprets and correlates all container information. On the web page, the temperature and humidity values are displayed in the graph in real-time, the location of the container is displayed in real-time on the map and allows data visualization with a friendly graphical user interface.

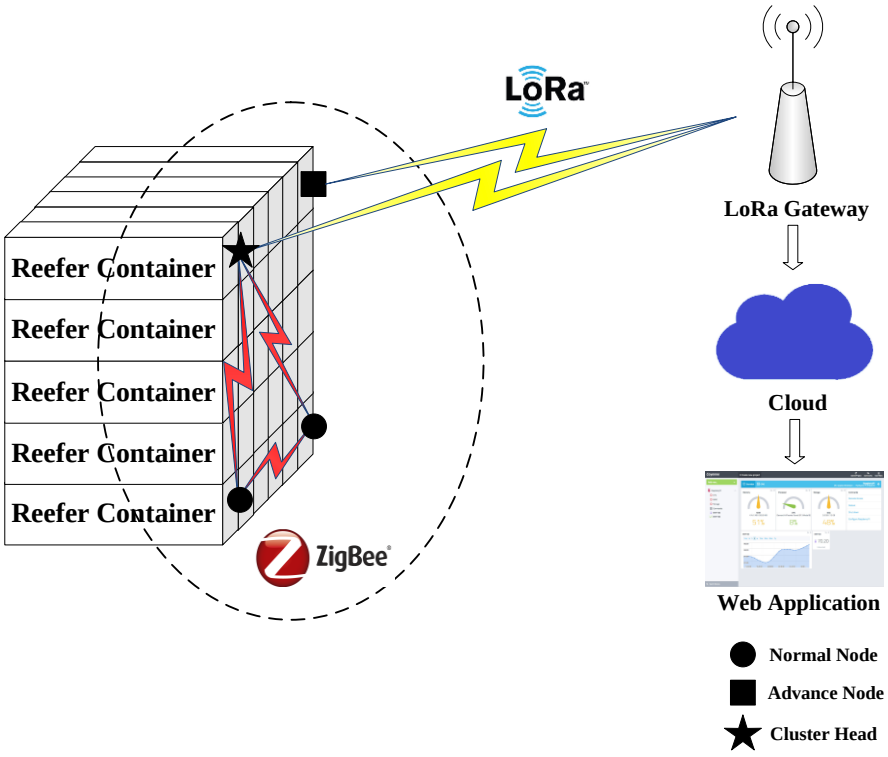


Figure 3.1 - The block diagram architecture of the container monitoring system

3.2 WSN end node design

Each WSN end node includes RH'T sensor, GPS module, radio modules and Microcontroller. If the temperature and humidity sensor is placed outside the container, it is difficult to acquire the temperature and humidity data in the container. Thus, the temperature and humidity sensor should be put inside the container. However, for the 20' and 40' container, a reliable intra-container link is not possible for 868 and 2400MHz [30]. If radio modules are placed inside the container, a reliable ZigBee network link between nodes is not possible. So, the radio module should be put outside the container.

If the GPS module is placed inside a container, the GPS module can't read the location information. According to the instructions of the GPS module, the GPS module only works outdoors [31]. Therefore, GPS undoubtedly needs to be put outside the container.

The end nodes are designed as shown in Figure 3.2. The sense node is fixed on the outside the container and includes the radio module, the GPS module and the Microcontroller are placed in the box, and the temperature and humidity sensor is placed inside the container. The temperature and humidity sensor is connected to the Microcontroller I2C (Inter-Integrated Circuit) port.

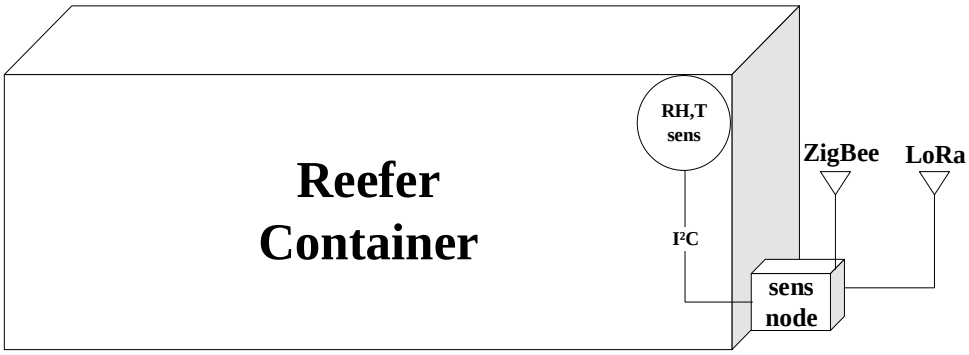


Figure 3.2 - The architecture of WSN end node

3.3 Design of communication architecture

3.3.1 Container Terminal Context

Containers in the container terminal generally have only two states: transportation

state or storage state.

After the reefer container arrives at the container terminal, it will be transported to the container yard and stored according to the container stacking rules. In Shanghai Yangshan Port, in order to facilitate the transport of containers into and out of the yard, the large yard is divided into small blocks by roads, as shown in Figure 3.3.

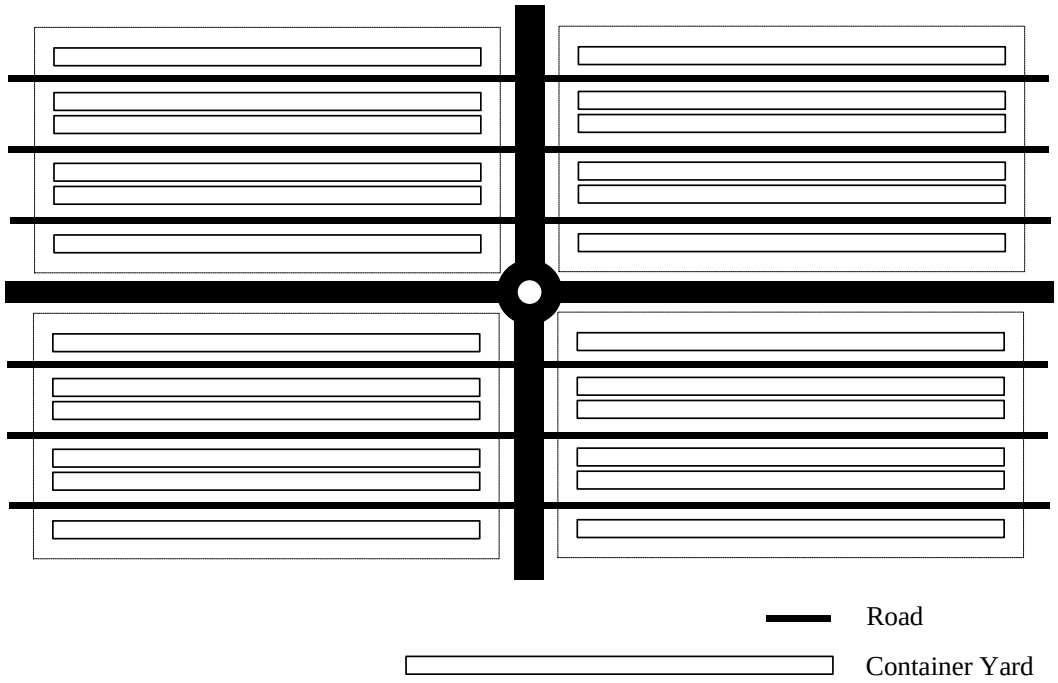


Figure 3.3 - The architecture of the container storage yard (top view)

In each storage block, the container is divided into multiple and orderly stacks. Generally, as shown in Figure 3.4, each stack is composed of 30 containers, 6 containers are placed horizontally and 5 containers are placed vertically.

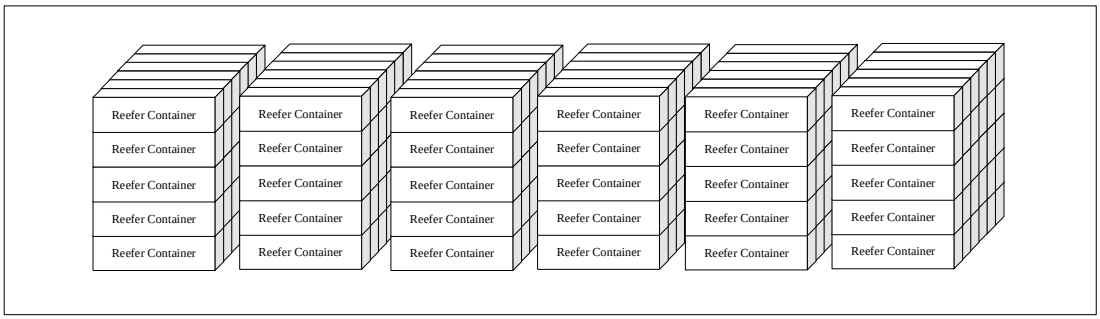


Figure 3.4 - The containers stored in the container yard

3.3.2 Design of the network topology

Data communication is mainly divided into two parts. In the first part, the end node acquires the data through the sensors and transmits them to the based station. Then, the based station transfers the data to the server and real-time displays it in on the web application.

Because the reefer containers often store the items that are easily deteriorated by the temperature and humidity and have a high value. Thus, the first principle of designing the transmission architecture is to ensure that all of the reefer containers are under monitoring. Due to the high mobility of the reefer containers, it is very difficult to frequently replace batteries that support power to monitoring equipment. Therefore, the second design principle of the communication architecture is to minimize the energy consumption of the monitoring equipment under the condition that each reefer container can be monitored.

LoRa and ZigBee can meet the requirements of low power consumption. LoRa can transmit data within more than 5km in outdoors and is more suitable for large areas context, such as a container terminal. LoRa systems are also consuming less energy than ZigBee [23]. When the reefer containers are in the state of being transported in the container terminal, the most ideal transmission method is to directly transfer data to the LoRa gateway through LoRa. When the containers are in the storage yard, as shown in Figure 3.5, the containers in the upper layer can transmit data to the LoRa gateway by using LoRa directly, but the containers in the lower layer will be blocked by other containers and cannot transmit data to the gateway through LoRa.

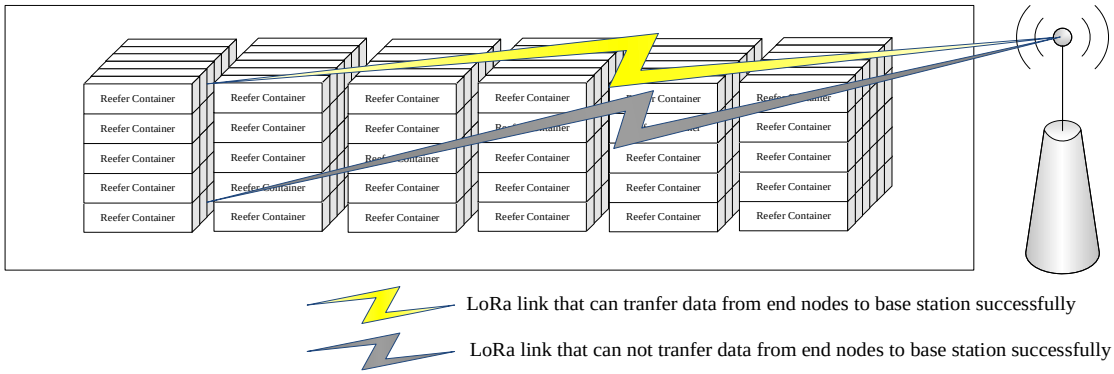


Figure 3.5 - The LoRa link to the Base Station

In the container terminal environment, LoRa is more suitable for transmitting data over long distances with low power consumption, but the containers in the lower layer

in the storage yard may be blocked. A Network architecture is proposed to solve those problems. Both LoRa and ZigBee communication modules are included on each monitoring terminal. When the WSN end node can use the LoRa module to transmit data to the LoRa gateway, the data is directly transmitted by using LoRa, and these devices are referred to as Advance Node (LoRa&ZigBee available). But some containers are in the lower layer of the storage yard and the WSN end node cannot directly use LoRa to transmit the data to the gateway, and these devices are referred to as Normal Node (only ZigBee available). As shown in Figure 3.6, the Normal Node transfer the data to the Advance Node which is in the same ZigBee network (Some containers are undoubtedly on the upper layer, thus, in each ZigBee network there are undoubtedly have some nodes can communicate with the gateway). Those WSN end nodes that receive data from Normal Nodes and forward data to the gateway can be referred to as Cluster Head. After the Cluster Head receives the data packets from the Normal Node, it will aggregate the packets and forward it to the gateway.

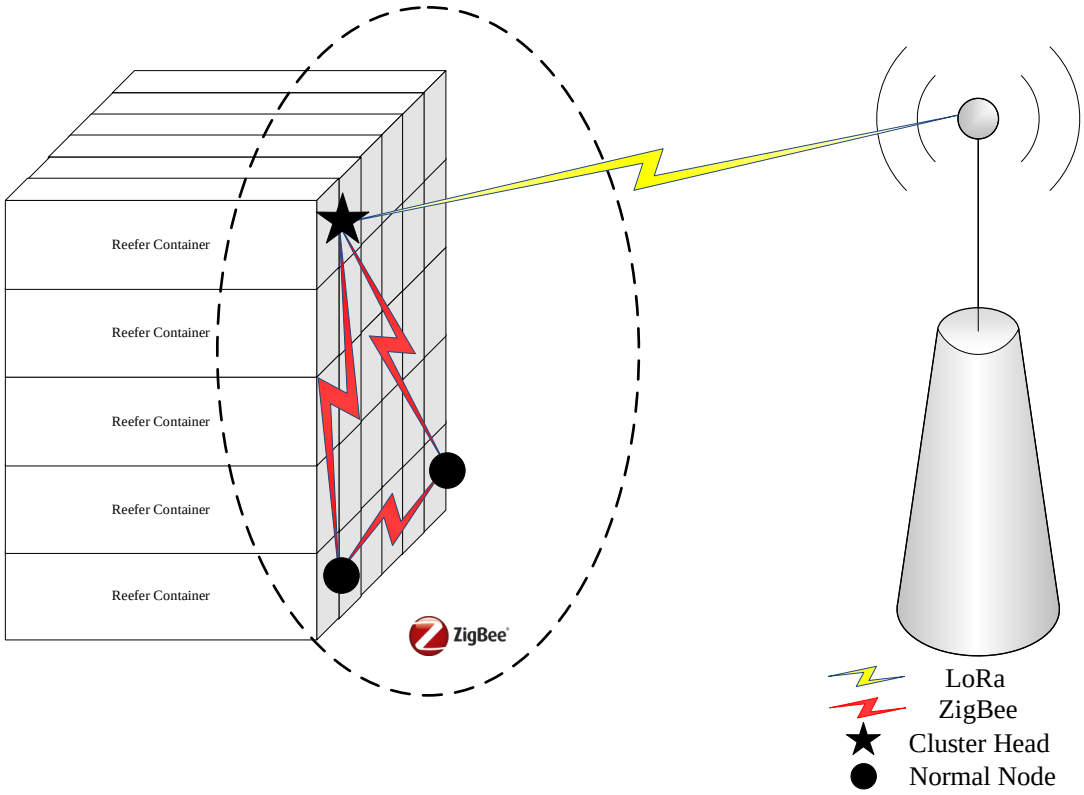


Figure 3.6 - The wireless communication architecture based on Lora and ZigBee protocols

According to the stacking principle of the port yard, we can know that each storage unit is composed of 30 containers, 6 containers are placed horizontally and 5 containers are placed vertically.

Therefore, a storage yard is composed of 30 containers, that is, a ZigBee network is also composed of 30 nodes. As shown in the figure 3.7, according to the size of the reefer containers, 30 nodes are distributed in an area of 14628 mm in length and 12955 mm in width.

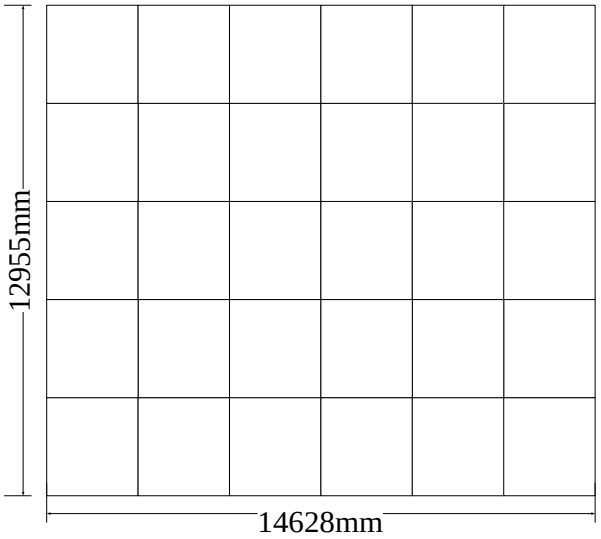


Figure 3.7 - The ZigBee network area for a storage unit

The 6 nodes in the upper of the area which can communicate with the LoRa gateway are Advance Nodes. The Advance Nodes has the possibility of becoming the Cluster Head, other nodes are Normal Nodes which can only transmit data to the Cluster Head.

Chapter 4 – System Hardware

4.1 Hardware Components of the WSN end node

Each of the end node has a microcontroller ATmega328p that will read the temperature, humidity and location of the reefer container from the sensors. The WSN end node is responsible for the acquisitions, processing and sending of data to the server. The hardware structure is shown in Figure 4.1.

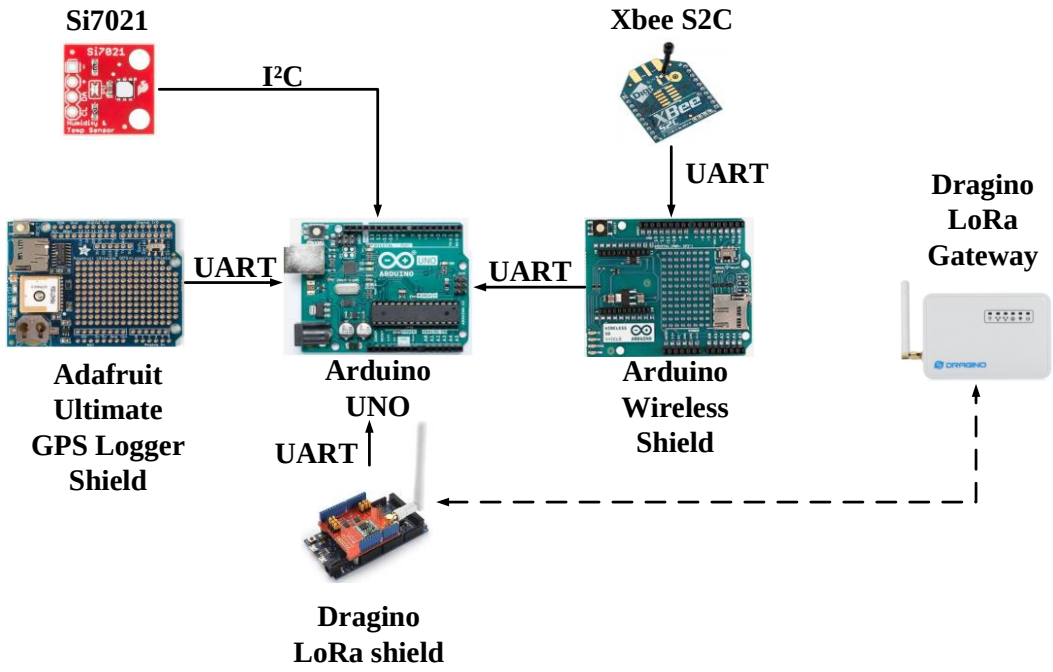


Figure 4. 1 - System Hardware: end node parts and gateway

It was developed embedded software for Arduino UNO, which is responsible of the acquisition, and processing of the data delivered by the sensors. All the microcontrollers are programmed using language C. Several Arduino libraries were used to deal with shields and the sensor boards.

4.2 Temperature and Humidity Sensor

SparkFun Humidity and Temperature sensors Si7021 (shown in Figure 4.2) are used to measure Humidity and Temperature.

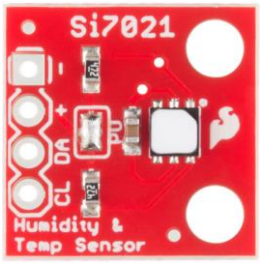


Figure 4. 2 - Si7021 humidity and temperature sensors

The Si7021 is a humidity and temperature sensor. It is a monolithic CMOS-IC (Complementary Metal-Oxide-Semiconductor Integrated Circuit) with a series of functions, such as to measure the humidity and temperature, convert analog signal to digital signal, calibrate data, process signal and it also has an I2C interface. This RH'T sensor uses industry-standard low-k polymer dielectrics to sense humidity. It has low power consumption, low drift, low hysteresis and also has long-time stability. All of the RH'T sensors are calibrated by the factory and sensor stores the calibration data in the memory. The calibration data guarantee that the accuracy of all the sensors is stable. Thus, one sensor can instead any sensors without doing any change in the hardware or software.

Si7021 has features such as accurate, low power consumption and be calibrated. The main features of the Si7021 are [32]:

- Typical Application Circuit for Relative Humidity and Temperature Measurement
- Precision Relative Humidity Sensor λ ± 3% RH (max), 0–80% RH
- High Accuracy Temperature Sensor λ ±0.4 °C (max), –10 to 85 °C
- 0 to 100% RH operating range
- Up to –40 to +125 °C operating range
- Wide operating voltage from 1.9 to 3.6 V
- Low Power Consumption. λ 150 µA when active current and λ 60 nA standby current

Thus, it is an ideal RH'T sensor for a lot of applications from asset tracking to industrial for measuring humidity and temperature. And it also an ideal RH'T sensor to measure the humidity and temperature inside of the reefer container.

The functional block diagram of Si7021 is shown in Figure 4.3.

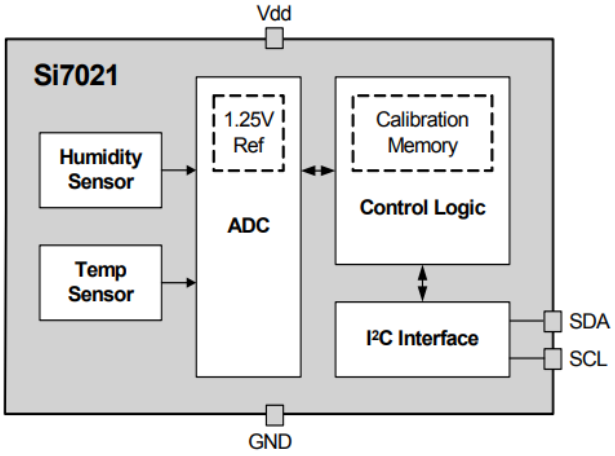


Figure 4.3 - The functional block diagram of Si7021

Figure 4.4 shows a typical application circuit that implements these functions.

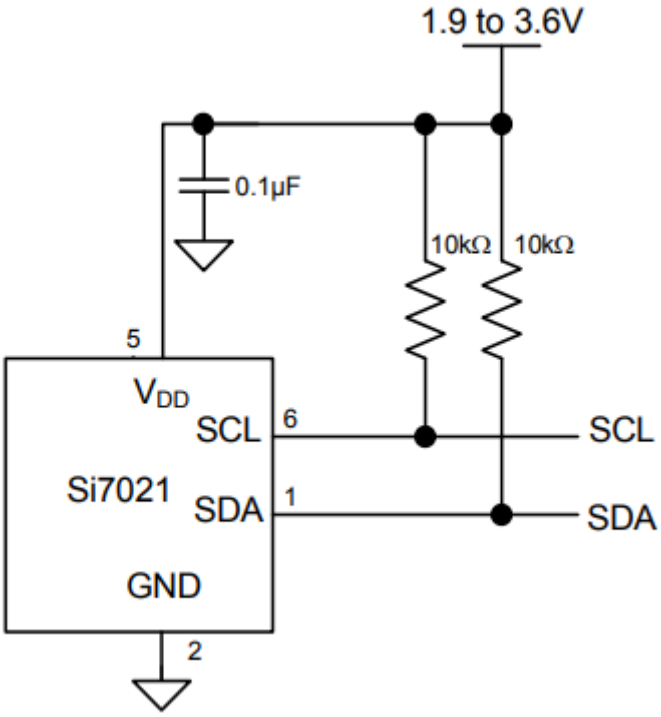


Figure 4.4 - The typical conditioning circuit of Si7021

Once the relative humidity measurement is made, the following expression can be used to convert the measurement to a relative humidity percentage:

$$\%RH = \frac{125 \cdot RH\_Code}{65536} - 6$$

Where $\%RH$ is the measured relative humidity value in $\%RH$, RH_Code is the 16-bit word returned by the Si7021.

The following expression can be used to convert temperature measurements to Celsius (°C):

$$Temperature(°C) = \frac{175.72 \cdot Temp\_Code}{65536} - 46.8 \quad (4.1)$$

Where temperature (℃) is the measured temperature value in ℃, $Temp\_Code$ is the 16-bit word returned by the Si7021.

The effective range to measure humidity is from 20% to 80% RH, and the effective range to measure temperature is from –10 ℃ to 60 ℃. If the temperature or humidity extend the effective range, the result acquired by SI7021 may change and the recovery times may longer.

Drift due to aging effects at typical room conditions of 30 ℃ and 30% to 50% RH may be affected by dust, vaporized solvents or other contaminants such as venting strips, adhesives, packaging materials, etc. The temperature and relative humidity accuracy are shown in Figure 4.5.

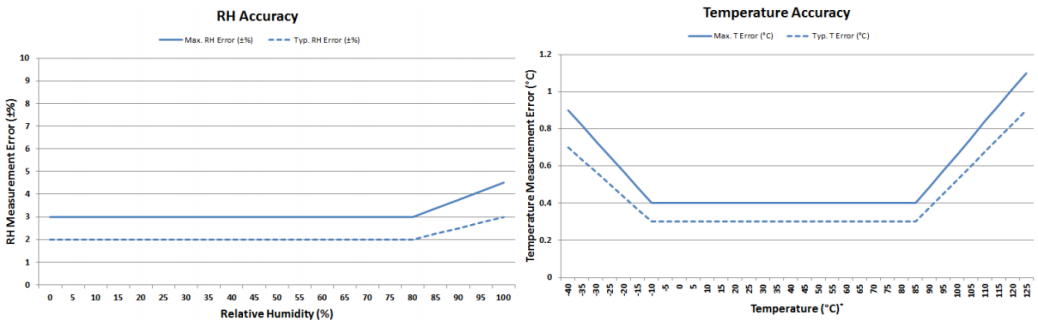


Figure 4.5 - The RH and temperature measurement errors from datasheet

Place the Si7021 in a temperature and humidity controlled room to measure the accuracy of the relative humidity sensor. Set the temperature to a convenient fixed value, such as 25–30 ° C, and change the relative humidity from 20% to 80%, then follow the steps below to rise back to 20%: 20% – 40% – 60% – 80% – 80% – 60% – 40% – 20%. At each humidity set-point, the humidity is stood for 60 minutes for the sensor to read the data. Figure 4.6 shows the results of the measurement [32].

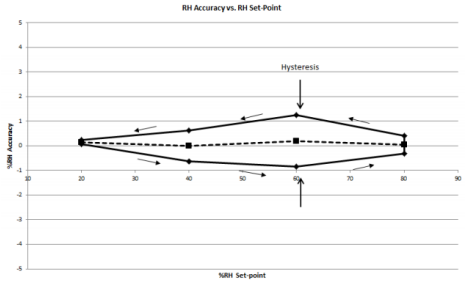


Figure 4. 6 - Measuring Sensor Accuracy Including Hysteresis

RH accuracy is the average of two data points at each relative humidity set point. In this case, the sensor displays an accuracy of 0.25% RH. Si7021 accuracy specifications include:

- Unit-to-unit and lot-to-lot variation
- The margin for shifts that can occur during solder reflow

The accuracy specification does not include:

- Hysteresis (typically ±1%)
- Effects from long term exposure to very humid conditions
- Contamine of the sensor by particulates, chemicals, etc.
- Other aging related shifts ("Long-term stability")
- RH readings will typically vary with temperature by less than ± 0.05% ° C.

4.3 GPS Module

In order to get the information of the reefer container's location, an Adafruit Ultimate GPS Logger Shield was included in the WSN end node. Adafruit Ultimate GPS Logger Shield（as shown in Figure 4.7) uses an embedded GPS module from GlobalTop PA6H with Mediatek MT3339 that achieves the industry's highest level of sensitivity (-165 dBm).

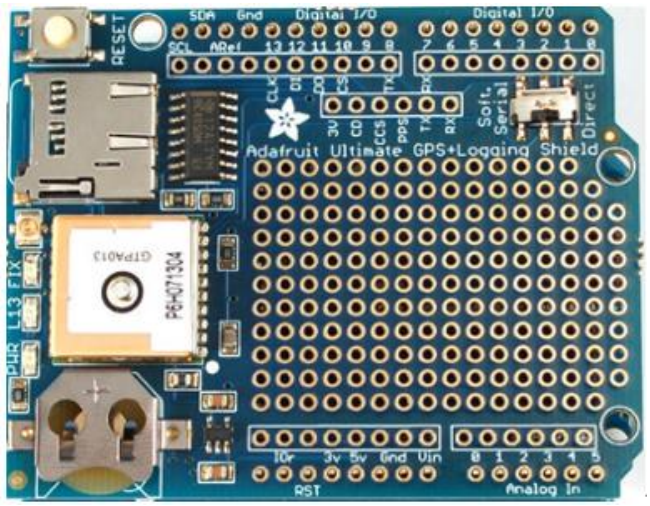


Figure 4. 7 - Adafruit Ultimate GPS Logger Shield

This GPS shield can supports an SD memory card and it also can work with Arduino UNO (as shown in Figure 4.8).

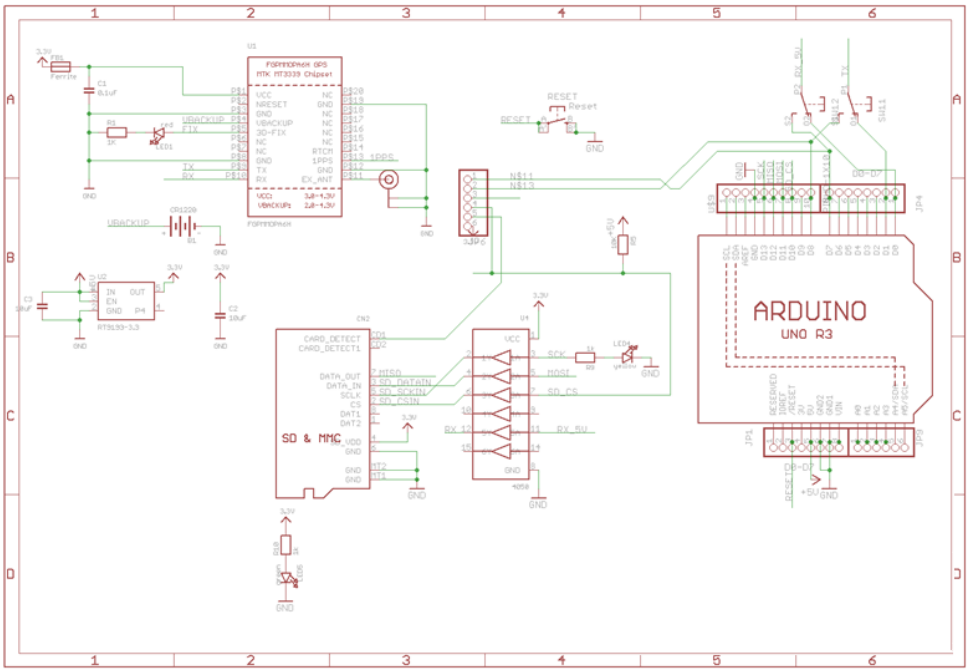


Figure 4.8 - The schematic of GPS shield connect with Arduino UNO

It also has the low power consumption for precise GPS signal processing to give the ultra-precise positioning under low receptive, high-velocity conditions (the shield is often accurate to 5-10 meters, but worse if the Shield is indoors or surrounded by tall buildings). GPS module characteristics [33]:

- -165 dBm sensitivity, 10 Hz updates, 66 channels
- Low power module - only 20mA current draw, half of most GPS's
- MicroSD card slot for data logging onto a removable card
- RTC battery included, for up to 7 years backup
- Built-in data logging to flash
- PPS output on fix
- Internal patch antenna + u.FL connector for external active antenna
- Power, Pin #13 and Fix status LED
- Big prototyping area

4.4 XBee S2C

XBee S2C 802.15.4 RF Modules (shown in Figure 4.9) are embedded solutions that can provide the wireless end-point connection to devices.

Figure 4.9 - XBee S2C transceiver

These devices provide network protocol for fast point-to-multipoint or peer-to-peer networks based on the IEEE 802.15.4. 15 General-purpose input/output (GPIO) ports in include in the XBee S2C ZigBee RF module. It can be used with the Arduino UNO and the schematic of the XBee S2C connect with the ATmega328P is shown in Figure 4.10.

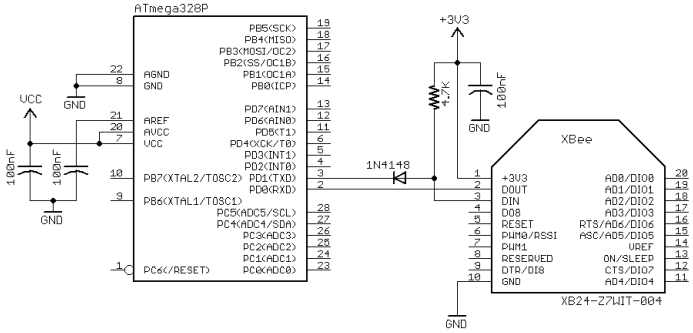


Figure 4.10 - The schematic of XBee S2C connected to ATmega328P

The XBee S2C can operate in the transparent operation mode and API (Application Programming Interface) operation mode in order to send or receive data by using the serial port. When the device is working in transparent mode, the device will act as the same function with the serial line.

In order to use RF transmission, the device received or send all of the UART (Universal Asynchronous Receiver/Transmitter) data queues through the DIN pin. After the RF data be received by the device, it will send data by using the DOUT pin. The configuration parameters can be set by using command mode.

The API mode of operation is an improved working mode from the transparent mode. It helps manage larger networks and is more suitable for tasks such as collecting data from multiple locations or remotely controlling multiple devices. API mode is a frame-based protocol that allows you to target data based on packets. This feature is

especially useful in large scale networks that need to control the operation of the radio network or need to know the packet comes from which node. The device transmits UART or SPI (Serial Peripheral Interface) data in the form of a packet (also known as an API frame). This mode allows for structured communication with serial devices [33].

The API mode provides a structured interface in which data is communicated through the serial interface in organized packets and in a determined order. This allows you to build complex communications between devices without having to define your own protocols. The API specifies how to send and receive commands, command responses, and device status messages from the device using the serial or SPI interface [34].

XCTU can be used to set XBee S2C to work in API mode. While XBee S2C is operating in API mode, all data is transferred according to the type of API frame. The API framework consists of multiple hexadecimal numbers. The API framework is shown in Figure 4.11.

Start delimiter	Length		Frame data								Checksum
			Frame type	Data							
1	2	3	4	5	6	7	8	9	...	n	n+1
0x7E	MSB	LSB	API frame type	Data							Single byte

Figure 4.11 - API Frame of XBee

The API Frame always begins with a start delimiter and the start delimiter always is 0x7E. The meaning of the start delimiter is that a new frame is beginning from the start delimiter. The start delimiter makes it easy for the devices to detect the new frames that will be received.

The total number of bytes of the API Frame Data field is indicated by the Length field. The total number of bytes does not include the start delimiter frame, length frame, and checksum frame [35]. This field contains the information that the device received or will send.

The structure of the data frame depends on the purpose of the API frame type. The frame type is an API structure identifier, it determines the type of API frame and indicates how the data field organizes data. This information and its order depend on the frame type defined by the Frame type field.

The checksum is the last byte of the frame and helps test data integrity. It is derived

by calculating the Hash value of all API frame bytes except the first three bytes (start delimiter and length). The device does not process frames sent over the serial interface with error checksums and ignores their data. The way to calculate the checksum of the API frame is [35]:

1. Add all bytes of the packet, except the start delimiter 0x7E and the length (the second and third bytes).

2. Keep only the lowest 8 bits from the result.

3. Subtract this quantity from 0xFF.

To verify the checksum of an API frame:

1. Add all bytes including the checksum; do not include the delimiter and length.

2. If the checksum is correct, the last two digits on the far right of the sum equal 0xFF.

4.5 Dragino LG01-N

The LG01-N is a LoRa gateway that is open-source and with single-channel made in Dragino. With the help of the LG01-N, bridge the LoRa wireless network to an IP network by using Wi-Fi, 3G or 4G is possible [36].

The system overview of LG4.1-N is shown in Figure 4.12. The LG01-N features the Dragino HE Linux module, which has a 400MHz ar9331 processor, 64MB RAM and 16MB Flash [37].

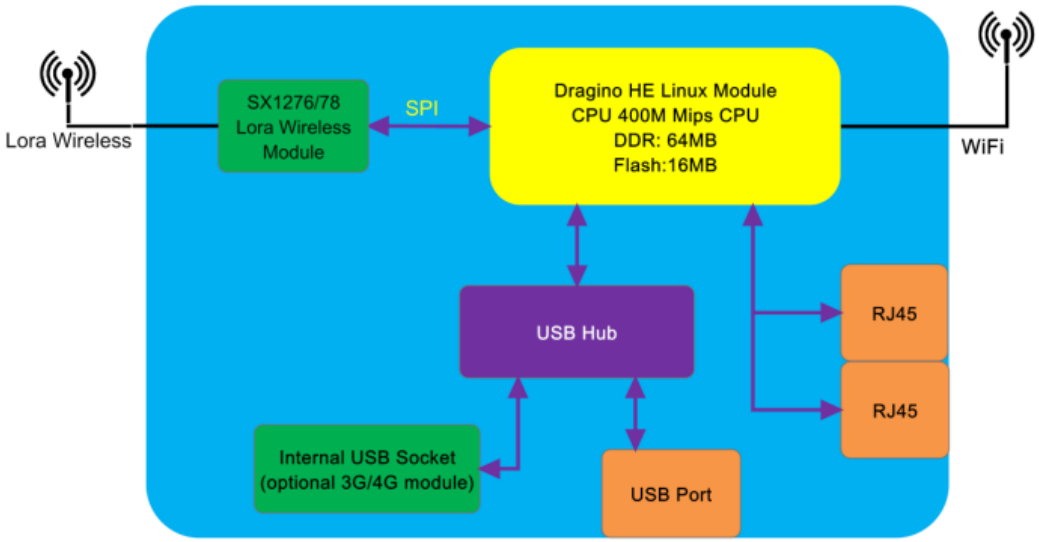


Figure 4. 12 - The LG01-N gateway system

After the LoRa End Node send or receive data, it will upload data to server or receive

data from server through LoRa gateway as shown in Figure 4.13.

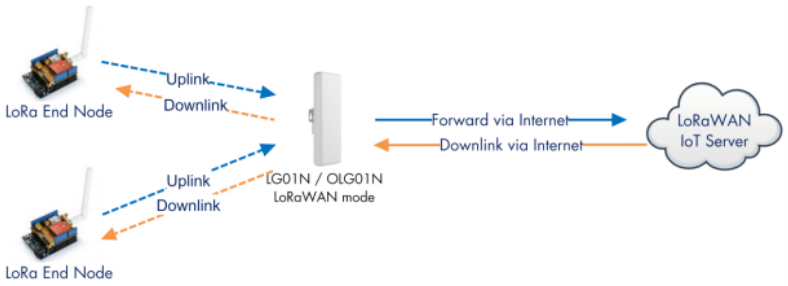


Figure 4.13 - The LoRa uplink and downlink of the end nodes

4.6 Arduino UNO

Arduino Uno (shown in Figure 4.14) is a microcontroller board based on the ATmega328P. The Arduino Uno includes 14 digital I/O pins (6 for Pulse Width Modulation output), 6 analog input pins, a 16 MHz quartz crystal oscillator, a USB connection, a power jack, an ICSP (In-Circuit Serial Programming) connector and a reset button [38].

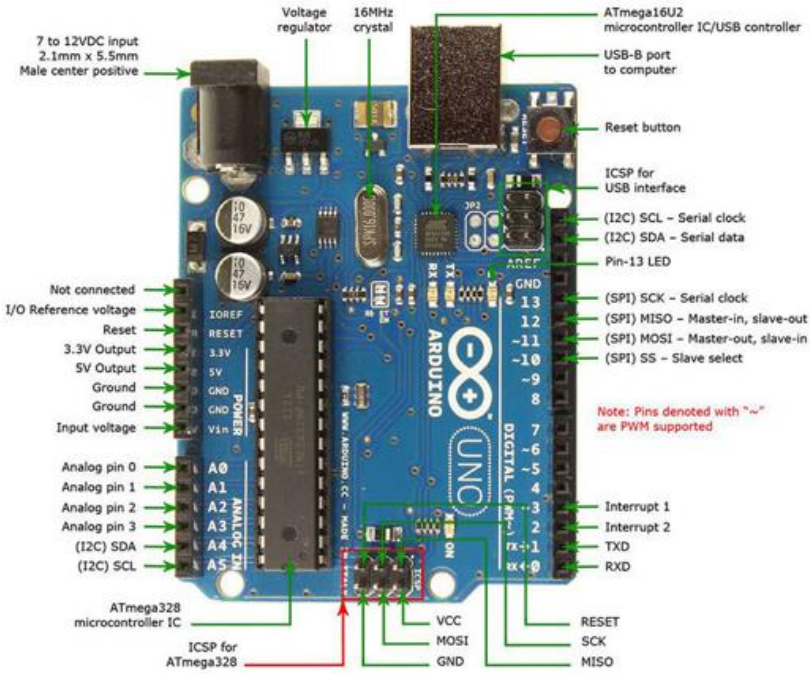


Figure 4.14 - Arduino Uno pinout / pin mapping

It contains almost everything which is necessary to support a microcontroller. Simply connect it to the computer with a USB cable or use the AC-DC adapter or battery is

enough to power it up [38]. The board can supply by an external power supply that can supply 6 to 20 volts. However, if the supply voltage from the external power supply is lower than 7V, the supply voltage of the Arduino Uno 5V pin may be lower than 5V, and the board may become unstable. If the voltage be used exceeds 12V, the board may damage [38]. Thus, the Arduino UNO can work well with a voltage range of 7 to 12 volts.

On this project, each node has an Arduino UNO. The gateway and nodes are connected as shown in figure 4.15. As shown in Figure 4.15, the sense node includes the Arduino UNO, Adafruit Ultimate GPS Logger Shield, XBee S2C and LoRa end node.

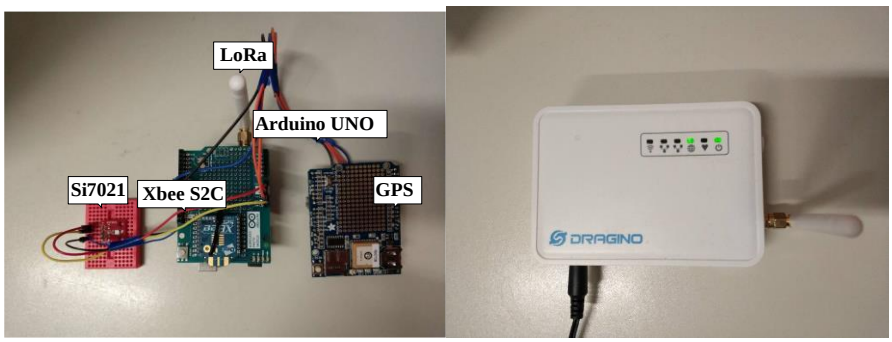


Figure 4.15 - The end node and Dragino gateway

The sense node is fixed on the outside the container and the Si7021 is put into the container (as shown in Figure 4.16).

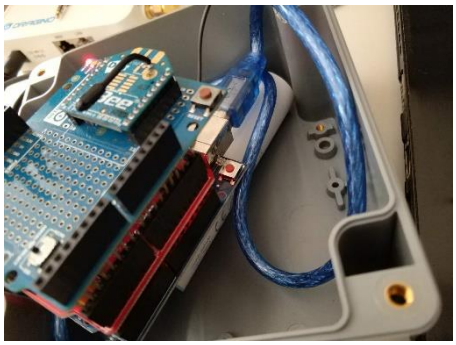


Figure 4.16 - The Sense node

Chapter 5 – Embedded Software

5.1 Design of the routing algorithm

Based on the proposed network architecture, a routing algorithm was designed which is suitable for this architecture.

The first purpose of the algorithm is to determine that in the ZigBee network, how many Advance Node (LoRa&ZigBee available) should become Cluster Head at the same time will extend the stability period (time interval before the death of the first sensor) of WSN. If the optimal number of Cluster Nodes is 2, in the ZigBee network there should be 2 Cluster Heads. Thus, the Normal Nodes should transfer data to the 2 Cluster Heads, and the Cluster Heads forward data from the Normal Nodes to LoRa gateway.

Under the condition the optimal number of Cluster Nodes is known, the second purpose of the algorithm is to determinate what rules can be used to select Cluster Nodes from Advance Nodes will extend the stability period of WSN. Thus, the algorithm decides which Advance Node should to be the Cluster Head and when it needs to change other nodes to be the Cluster Head.

5.1.1 The optimal number of Cluster Nodes

In order to calculate the optimal number of Cluster Nodes, it is necessary to use the radio energy dissipation model.

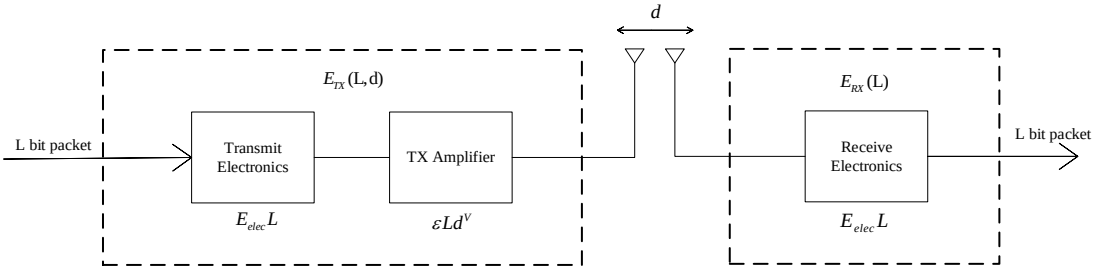


Figure 5.1 - Radio energy dissipation model

According to the radio energy dissipation model illustrated in Figure 5.1 [39], in order to achieve an acceptable SNR (Signal-Noise Ratio) in transmitting L bit packet

over a distance d , the energy consumed by the radio is given by:

$$E_{TX}(\mathrm{L},d)=\begin{cases}L\cdot E_{elec}+L\cdot\in_{fs}\cdot d^{2} & (\text{if } d<d_{0})\\ L\cdot E_{elec}+L\cdot\in_{mp}\cdot d^{4} & (\text{if } d\geq d_{0})\end{cases} \tag{5.1}$$

Where E_{elec} is the energy dissipated per bit to run the transmitter or the receiver circuit, $\in_{fs}$ and $\in_{mp}$ depend on the transmitter amplifier model, and d is the distance between the sender and the receiver.

Assume that the number of the WSN end node is n, the number of the Advance Nodes is m and the number of the Cluster Head is k .

Then the energy consumed by each Cluster Head in each round E_{CH} is:

$$E_{CH}=L\cdot E_{elec}\cdot\left(\frac{n-m}{k}-1\right)+L\cdot E_{DA}\cdot\left(\frac{n-m}{k}-1\right)+L\cdot E_{elec}\cdot\frac{n-m}{k}+L\cdot\in_{fs}\cdot d_{toGW}^{2} \tag{5.2}$$

Where E_{DA} is the energy dissipated of data aggregation, d_{toGW} is the distance from Cluster Head to LoRa gateway. The energy consumed by each Advanced Node during each round E_{AN} is

$$E_{AN}=L\cdot E_{elec}+L\cdot\in_{fs}\cdot d_{toGW}^{2} \tag{5.3}$$

The energy consumed by each Normal Node during each round E_{NN} is

$$E_{NN}=L\cdot E_{elec}+L\cdot\in_{fs}\cdot d_{toCH}^{2} \tag{5.4}$$

Where d_{toCH} is the distance from Normal Node to the Cluster Head.

The sum of the distances of all Normal Node to the Cluster Head $T\left[d_{toCH}\right]$ is

$$T\left[d_{toCH}\right]^{2}=\iint\left(x^{2}+y^{2}\right)\cdot\rho(x,y)\,dxdy \tag{5.5}$$

$$T\left[d_{toCH}\right]^{2}=\int_{0}^{\frac{6\cdot24}{k}}\int_{0}^{6\cdot26}x^{2}+y^{2}dydx=\frac{(26\cdot6)^{3}}{3}+\frac{(6\cdot24)^{3}}{3\cdot k^{3}} \tag{5.6}$$

Where $\rho(x,y)$ is the WSN end nodes distribution. Thus, the total energy consumed by each cluster in each transmission cycle $E_{cluster}$ is

$$E_{cluster}\approx E_{CH}+\left(\frac{n-m}{k}-1\right)\cdot E_{NN} \tag{5.7}$$

The total energy dissipated of the ZigBee network E_{tot} is:

$$E_{tot}\approx k\cdot E_{cluster}+(m-k)\cdot E_{AN} \tag{5.8}$$

By differentiating E_{tot} with respect to k and equating to zero, the optimal number K_{opt} of constructed clusters can be found.

$$k_{opt} = \sqrt[4]{\frac{(n-m)\left(6 \cdot 2.4\right)^3}{d_{toBS}}} \tag{5.9}$$

According to equation (9), k_{opt} is depend on the number of the Normal Nodes and the d_{toBS}. The MATLAB be used to calculate the k_{opt} in different d_{toBS}. Figure 5.2 shows the location of the nodes in ZigBee network.

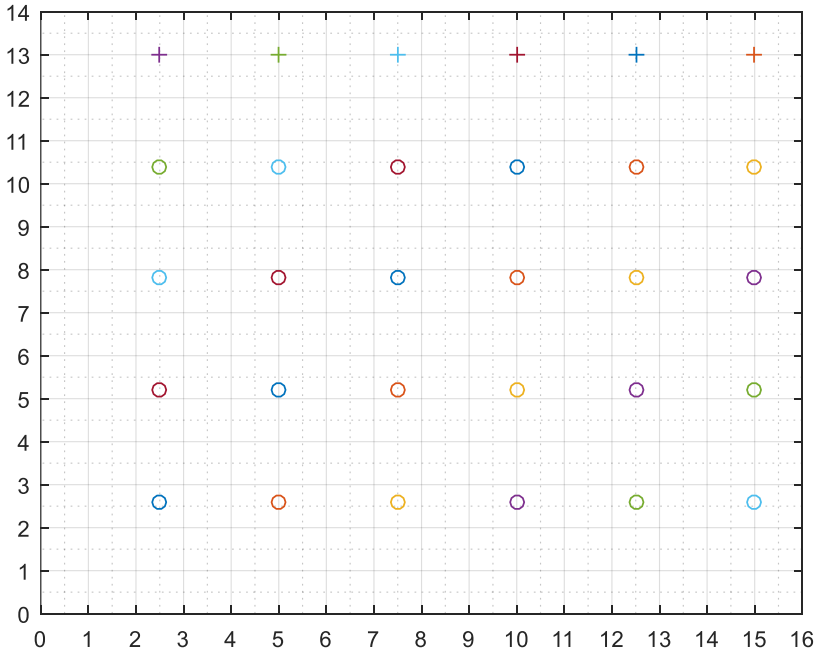


Figure 5.2 - The WSN nodes distributed on the container level

According to the Figure 3.7, the number of the containers in a storage unit is 30. So the number of the end node in a ZigBee network is 30 and the Advance node is 6. The $n-m$ in the equation (9) is 24. According to equation (9) the k_{opt} related with d_{toBS}. MATLAB was used to calculate the k_{opt} with different d_{toBS} and the code is shown in Figure 5.3.

```
for i=1:1:5000
K(i)=((54*(6*2.4)^3)/i)^0.25;
end
figure(1)
for i = 1:1:5000
plot(i,fix(K(i)));
grid on;
grid minor
set(gca,'YTick',[0:1:20])
set(gca,'XTick',[0:500:5000])
axis([0,5000,0,20])
xlabel('Distance to BS(m)');
ylabel('Kopt');
end
```

Figure 5.3 - The code be used to calculate k_{opt}

As shown in Figure 5.4, the k_{opt} depends to the distance from Advance Node and LoRa gateway.

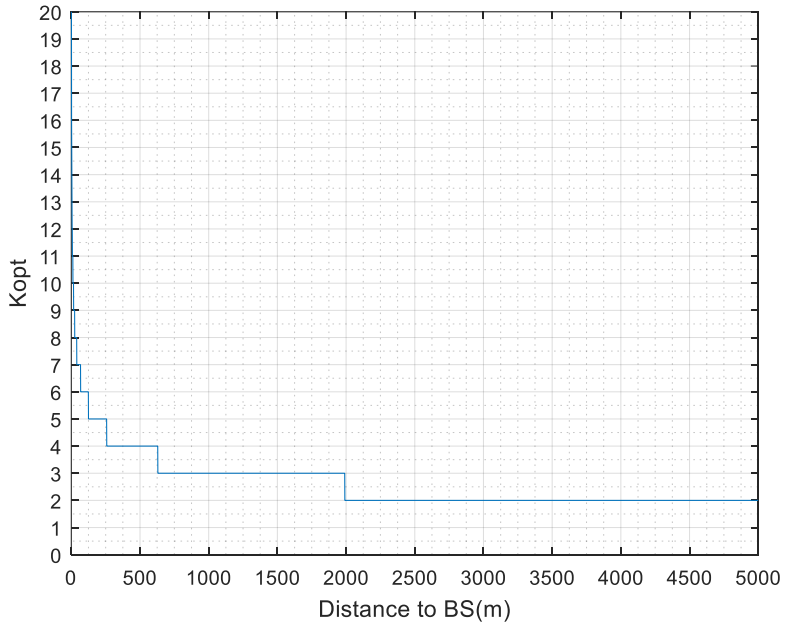


Figure 5.4 - The optimal number of cluster nodes

5.1.2 The routing algorithm – simulation and results

In order to extend the stability period of WSN, the routing algorithm focus on guarantee a well-balanced distribution of the energy load among WSN end nodes of the sensor network. Because if some of the Advance Node always to be the Cluster Node,

these Advance Nodes tend to die faster than other end nodes. So the routing algorithm should guarantee that the energy load is distributed by dynamically elected Cluster Head. Cluster Heads aggregate packets from their cluster Normal Nodes before forwarding them to the LoRa gateway. By rotating the Cluster Head uniformly among all nodes, each end node tends to expend the same energy over time.

Based on the LEACH, we design the routing algorithm which is suited for the containers in the container terminal storage yard. The operation of the routing algorithm is broken up into "rounds" and new Cluster Heads are elected and beginning with a set-up phase in each round. After the Cluster Heads are organized, the WSN entry a steady-state phase in that time Cluster Heads forward the packets to the LoRa gateway. In order to minimize overhead, the steady-state phase should be much longer compared to the set-up phase (the set-up phase consumes some energy without transmit any data).

Assume that the number of the Advance Nodes in a ZigBee network is m and the optimal number of Cluster Node is k_{opt} . Each Advance Nodes will become a cluster head exactly once every m / k_{opt} rounds. The m / k_{opt} rounds are called Network Epochs. In every Network Epoch, each Advance Node has a probability of k_{opt} / m becoming a cluster head. Initially, after the network was created, each advanced node had a probability of k_{opt} / m to decide whether to become a cluster head. The Advance Nodes that have already elected to be cluster heads in the current round can no longer become Cluster Heads in the same Network Epoch. This decision is made by the Advance Node n choosing a random number in [0, 1]. If the number is less than a threshold T (n), the Advance Node becomes a Cluster Head for the current round. The threshold is set as:

$$T(n)=\begin{cases}\dfrac{\frac{k_{opt}}{m}}{1-\frac{k_{opt}}{m}\cdot\left(r\cdot \bmod \frac{m}{k_{opt}}\right)} & if \quad n\in G \\ 0 & otherwise\end{cases} \tag{5.10}$$

Where r is the current round number, and G is the set of non-elected Advance Nodes in a Network Epoch. Using this threshold, each Advance Nodes will be a Cluster Head at one round within m / k_{opt} rounds. During round 0 (r = 0), each node has a probability m / k_{opt} of becoming a Cluster Head. The probability that the remaining nodes to be the Cluster Heads in the later round will increase because there are fewer nodes that are waiting to become Cluster Heads.

MATLAB is used to simulate this routing algorithm and compare with directly

transfer the data. We simulate the WSN in an area with dimensions 15m × 13(m). The population of the nodes *n* is equal to 30 and the Advance Nodes *m* is equal to 6. The way nodes are distributed over the field is illustrated in Figure 5.2. The distance from Cluster Head to LoRa gateway d_{toBS} is (instead of =)1(km). According to the results in Figure 5.4, the optimal number of Cluster Node k_{opt} = 2. We assume the radio characteristics as shown in Table 4.

Table 4- Radio characteristics

Transmitter Electronics	$E_{elec} = 50nJ / bit$
Receiver Electronics	$E_{elec} = 50nJ / bit$
Data Aggregation	$E_{DA} = 5nJ / bit / signal$
Transmit Amplifier	$\in_{fs} = 10pJ / bit / m^2$

I set the network to work 9000 round. Figure 5.5 (a) shows all of the nodes in the network.

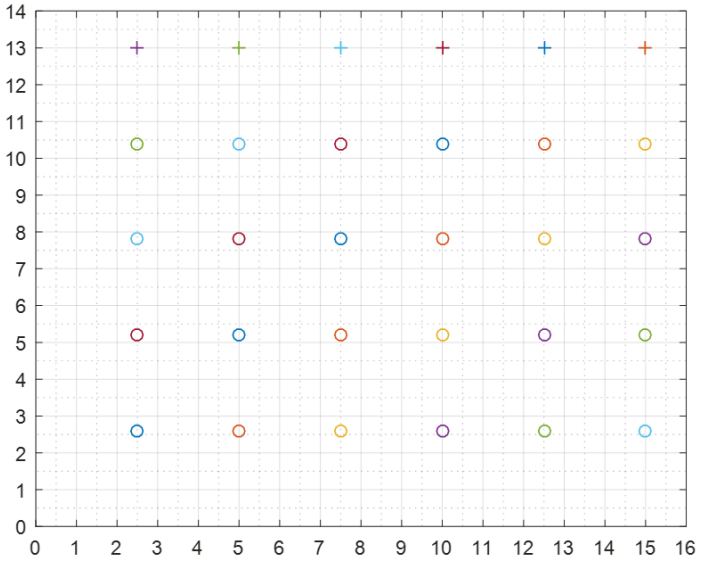


Figure 5.5 (a) - The nodes in the ZigBee network

Figure 5.5 (b) shows the throughput from ZigBee network to the Base Station Per round. This algorithm has the ability to transfer more data to Base Station.

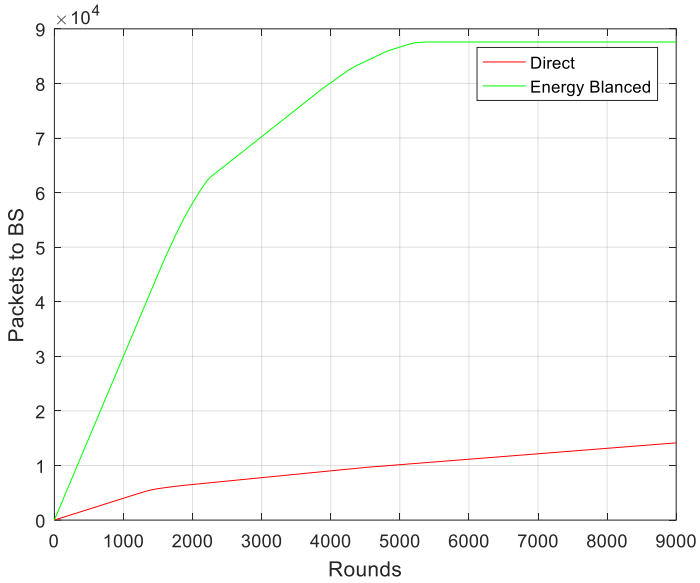


Figure 5.5 (b) - The throughput from cluster head to the LoRa gateway

As shown in Figure 5.5 (c), the blue line shows the number of the dead nodes when using direct transmission and the green line shows the number of the dead nodes when using energy is balanced. The death node will appear earlier when using direct transmission. When use the energy balanced algorithm, the stable period of the network is longer than the direct transmission.

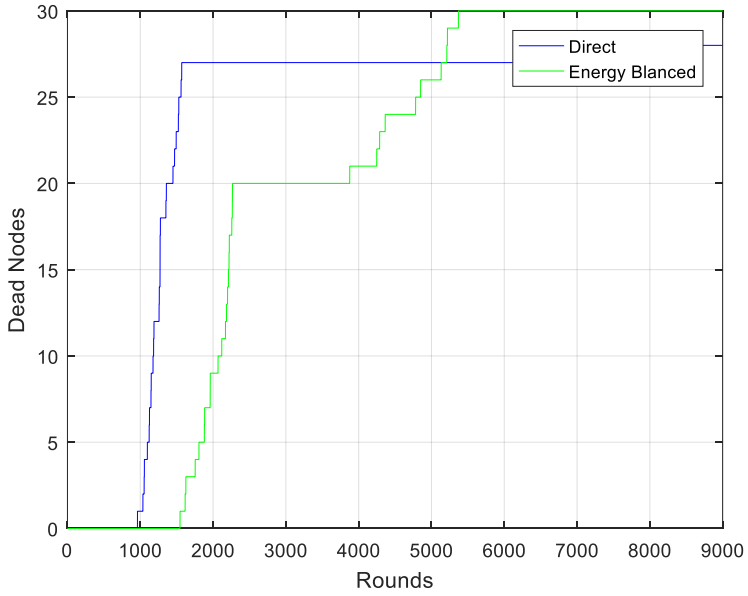


Figure 5.5 (c) - The number of dead nodes in the ZigBee network

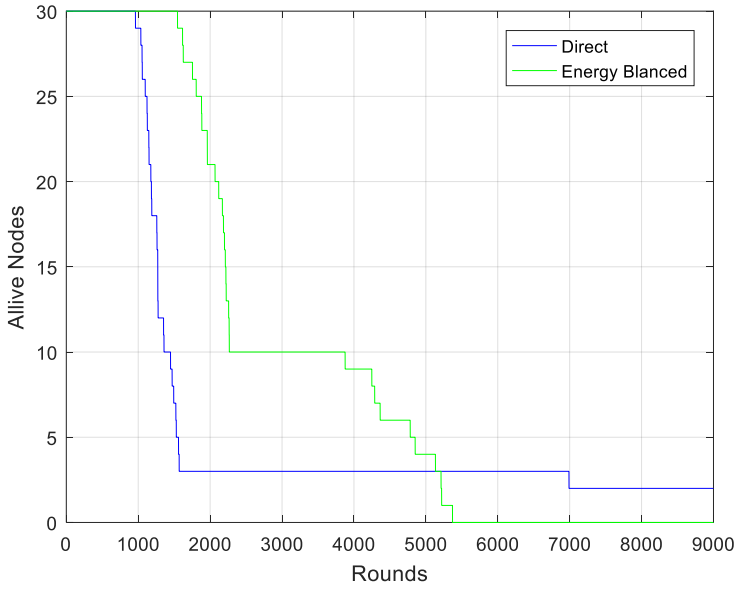


Figure 5.5 (d) - The number of alive in the ZigBee network

5.2 Embedded software

5.2.1 Sensor Data Acquisition

The acquisition and primary processing of sensor data are made on the level of the WSN end node by the ATmega328P microcontrollers. The program code is programmed using Arduino IDE ("Arduino - Software",) by language C and Arduino libraries.

The temperature and humidity data are acquired from Si7021. The connection between the Si7021 and the Arduino UNO is made through an I2C communication protocol. Adafruit_Si7021 library makes it simple to read temperature and humidity data from the Si7021. The result of acquiring RH'T data is shown in Figure 5.7, the red line is the temperature and the blue line is the relative humidity.

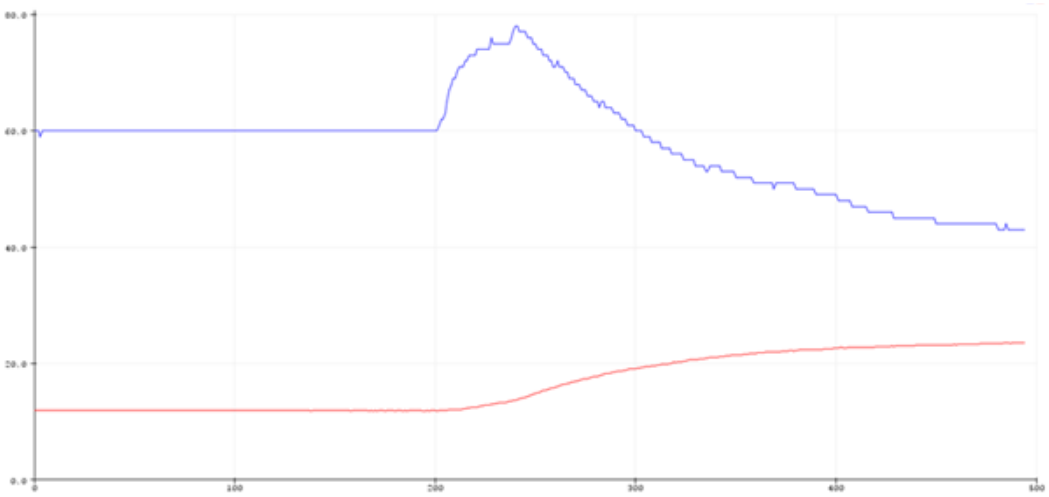


Figure 5.6 - The RH'T data

The location data is acquired from the Adafruit Ultimate GPS Logger Shield. The communication protocol between Arduino Uno and Adafruit Ultimate GPS Logger Shield is UART. Figure 5.8 shows the raw GPS "NMEA sentence" output from the module.

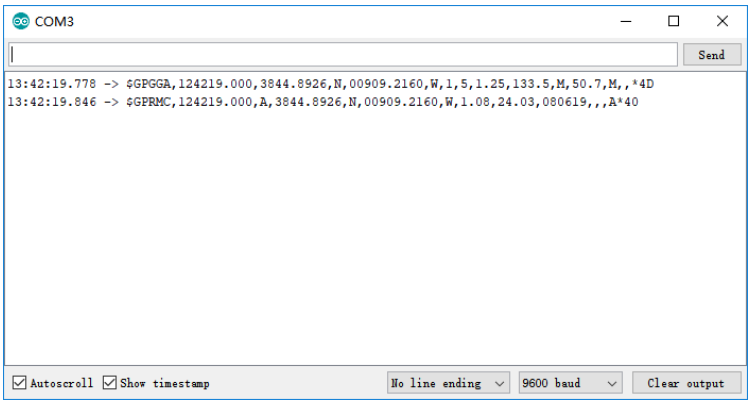


Figure 5.7 - The raw GPS data example

There are several NMEA sentences, the most commonly used is $GPRMC (the lowest coordinates recommended by Global Positioning or something similar) and the $PGGGA sentence. Both provide time, date, latitude, longitude, altitude, estimated land speed and positioning type. An example shows in "$GPRMC, 124219.000, A, 3844.8926, N, 00909.2160, W, 1.08, 24.03, 080619, A * 40".

- The first part '**124219.000'** is the current time GMT (Greenwich Mean Time). The first two numbers **12** indicate the hour the next two **42** are the minute, the next two **19** are the seconds and finally **000** is the milliseconds. So the time in this Figure is 12:42 and 19 seconds.

- The second part '**A**' is the status code, if it is a **V** that means the data is Void (invalid). If it is an **A** (as shown in Figure 5.8) that means its Active.
- The next 4 pieces of data '**3844.8926, N, 00909.2160, W**' are the geolocation data. According to the GPS, the location is 3844.8926, N (Latitude 38 degrees, 44.8926 decimal minutes North) & 00909.2160, W (Longitude 90 degrees, 9.2160 decimal minutes West). N and E are positive, S and W are negative.
- The data '**1.08**' is the ground speed in knots, the speed is 1.08 knots.
- The '**080619**' which is the current date (19th of June, 2019).
- Finally the '**A * 40**' is used as a data transfer checksum.

The "Adafruit_GPS" library was used to acquire the location data. The result is shown in Figure 5.9.

```
11:23:58.562 -> Time: 10:23:58.000
11:23:58.562 -> Date: 16/9/2019
11:23:58.562 -> Fix: 1 quality: 1
11:23:58.562 -> Location: 3844.8876N, 909.2115W
11:23:58.562 -> Speed (knots): 4.77
11:23:58.562 -> Angle: 133.73
11:23:58.562 -> Altitude: 144.70
11:23:58.562 -> Satellites: 5
```

Figure 5.8 - The data from Adafruit Ultimate GPS Logger Shield

5.2.2 XBee delivered data

Before the Normal node sends data to the Cluster head, it will encode the data according to the rule of the API Frame first. Then, it sends the API Frame to the Cluster head. Figure 5.10 shows an API Frame be received by Cluster head. The packet is "7E 00 12 90 00 7D 33 A2 00 41 4F 7F 88 3E 46 01 1B 2E 5B 3E 2E 08 83".

Receive Packet (API 2)

```
7E 00 12 90 00 7D 33 A2 00 41 4F 7F 8B 3E 46
01 1B 2E 5B 3E 2E 08 83
```

Figure 5.9 - API Frame

As shown in the Figure 5.10, the "7E" is the start delimiter, it means that this is a new API Frame.

Start delimiter

7E

Figure 5.10 - Start delimiter of the API Frame

Figure 5.11 shows that the “00 12” is the length, and it presents the length of the Frame data. By converting it to a decimal number we can know the length of the API frame is 18 bit.

Figure 5.11 - Length of the Frame data

“90” (as shown in Figure 5.12) is the Frame type. It means when a device configured with a standard API Rx Indicator (AO = 0) receives an RF data packet, it sends it out the serial interface using this message type.

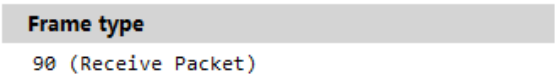


Figure 5.12 - Frame type

As shown in Figure 5.13, the 64-bit source address present the 64-bit address of the Normal node. The 16-bit source address present the 16-bit address of the Normal node.

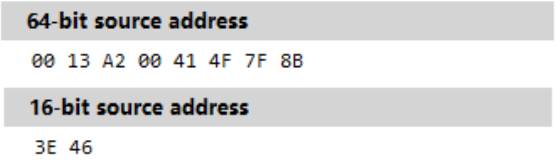


Figure 5.13 - Source address of the Normal node

The receive options (as shown in Figure 5.14) is “01” means that packet was a broadcast packet.

Figure 5.14 - Receive options

The RF data is “1B 2E 5B 3E 2E 08” (as shown in 5.15). They are hexadecimal numbers which present 27.94 and 62.08. they are the temparature data (27.94) and humidity data (62.08).

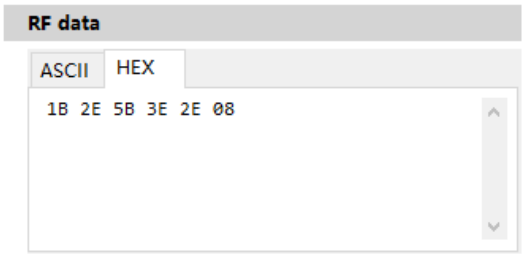


Figure 5.15 - RF data

As shown in Figure 5.16, the checksum is “83”.

Figure 5.16 Checksum

In Arduino IDE, the code shown in Figure 5.17. Put the temperature, humidity and location data in the ‘payload’ array in the form of a hexadecimal number. Add the 64-bit source address of the Cluster head, the ‘payload’ arrayand the size of the ‘payload’ into ‘zbTx’. The ‘zbTx’ is the Frame data. Send this packet to the Cluster head.

```
XBeeAddress64 addr64;
addr64 = XBeeAddress64(0x0013a200, 0x410A3F09);
payload[0] = thigh& 0xff;
payload[1] = tlow& 0xff;
payload[2] = hhigh& 0xff;
payload[3] = hlow& 0xff;
payload[4] = lath& 0xff;
payload[5] = latm& 0xff;
payload[6] = latl& 0xff;
payload[7] = longih& 0xff;
payload[8] = longim& 0xff;
payload[9] = longil& 0xff;
ZBTxRequest zbTx = ZBTxRequest(addr64, payload, sizeof(payload));
```

Figure 5.17 - The code to get the Frame data

As shown in Figure 5.18, after the Cluster head receive the data, it will check the start delimiter and the checksum. Then receive the Frame data and put them into the the ‘payload’ array.

```
xbee.readPacket();
if (xbee.getResponse().isAvailable()) {
  if (xbee.getResponse().getApiId() == ZB_RX_RESPONSE) {
    xbee.getResponse().getZBRxResponse(rx);
    if (rx.getOption() == ZB_PACKET_ACKNOWLEDGED) {

      nss.println(rx.getChecksum(), HEX);
      nss.println(rx.getPacketLength(), DEC);
      for (int i = 0; i < rx.getDataLength(); i++) {
      payload[i] = rx.getData()[i];}
```

Figure 5.18 - The code to receive the Frame data

The result of the Cluster head receive the Frame data is shown in Figure 5.19.

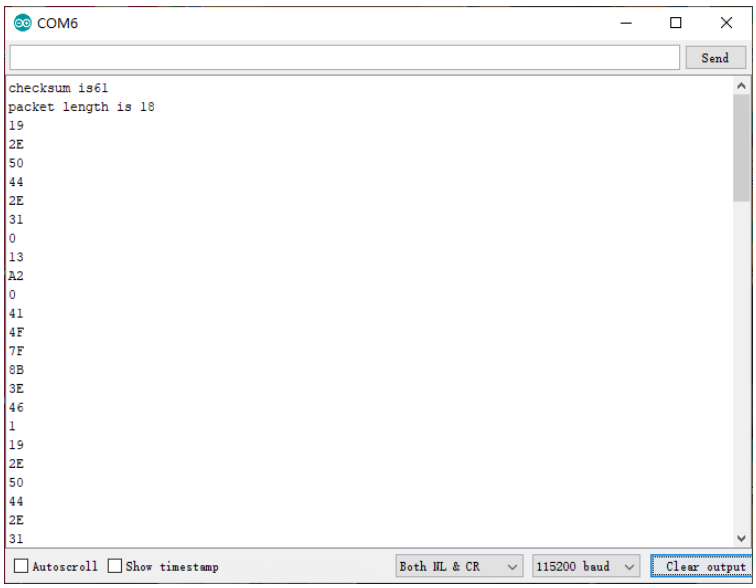


Figure 5.19 - The Frame data receive by Cluster head

5.2.3 LoRa delivered data

The LoRa device was set to work in OTAA (Over-The-Air-Activation) mode. OTAA devices are the most common type of device. When setting them up, each device working in OTAA mode will get three pieces of information: the application key, the application EUI, and the device EUI [40]. When the device is powered on, it initiates the join process, during which it negotiates a new set of keys based on the existing application key. If the device is powered off, it must be joined to the network again at startup. Not surprisingly, OTAA devices are most concerned with application keys. On the other hand, using OTAA devices makes it easy to manage a large number of devices because you don't have to manage many keys [40].

The main configurations made on the Dragino LG01-N module are LoRaWAN server settings and radio settings. These configurations are shown in

Figure 5.20. The IoT service is LoRaWAN/RAW forwarder and the debug level is little message output. The service provider is The Things Network.

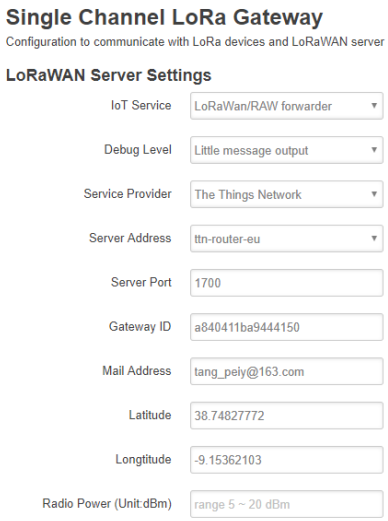


Figure 5.20 - LoRaWAN Server Settings

The radio settings (Figure 5.21) include frequency, spreading factor, coding rate and signal bandwidth. The LG01-N should work in the same frequency with the LoRa nodes and the frequency is 868.1MHz.

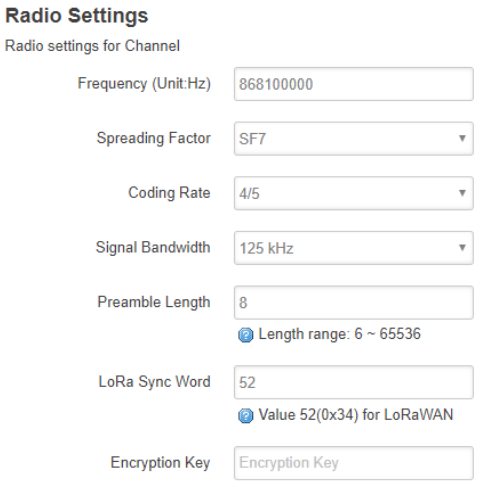


Figure 5.21 - Radio Settings

Set the application key, application EUI and device EUI in The Things Network server as shown in Figure 5.22 (a). Each of the devices working in OTAA mode will include those key to get the provision to update the data to the server as shown in Figure 5.22 (b).

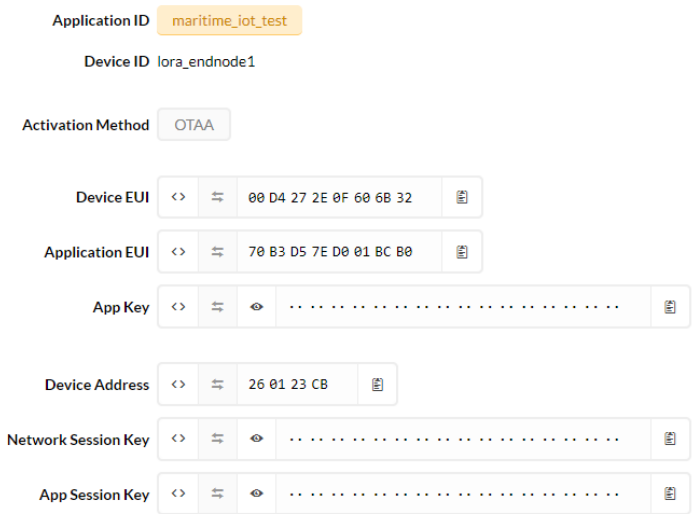


Figure 5.22 (a) - The key in The Thing Network

```
static const u1_t PROGMEM APPEUI[8]={ 0xB0, 0xBC, 0x01, 0xD0, 0x7E, 0xD5, 0xB3, 0x70 };
void os_getArtEui (u1_t* buf) { memcpy_P(buf, APPEUI, 8);}

static const u1_t PROGMEM DEVEUI[8]={ 0x32, 0x6B, 0x60, 0x0F, 0x2E, 0x27, 0xD4, 0x00 };
void os_getDevEui (u1_t* buf) { memcpy_P(buf, DEVEUI, 8);}

static const u1_t PROGMEM APPKEY[16] ={ 0xD3, 0x4D, 0xDB, 0xDE, 0xC0, 0xC6, 0x9B, 0x3D, 0xB2, 0x4E, 0x6D, 0x3B, 0xAB, 0x1E, 0x6E, 0xDD };
void os_getDevKey (u1_t* buf) {  memcpy_P(buf, APPKEY, 16);}
```

Figure 5.22 (b) - The key in end nodes

When the device work in OTAA mode, LoRa node will send a join request and get EV_JOINED which means OTAA joined successfully. As shown in the Figure, the LoRa node sends a join request and joins successfully. Then LoRa gateway forwards the temperature and humidity data that is acquired from Si7021 and uploads it to TTN.

The development is done by using the Arduino IDE. At first, the WSN end node sends a request to TTN and waits for a reply from the TTN. If the LoRa node doesn’t receive a reply, send the request again.

Next, the LoRa node receives the request from the TTN, acquires the data from the sensors, puts the data in a cayenne array, and then uploads the data to TTN.

The result of the LoRa node sending data to TTN is shown in Figure 5.23.

```
16:52:03.252 -> Si7021 test!
16:52:03.319 -> RXMODE_RSSI
16:52:03.352 -> ############    COUNT=1    ############
16:52:03.352 -> The temperature and humidity:
16:52:03.352 -> [26.18℃,66.50%]
16:52:03.352 -> 5633: engineUpdate, opmode=0x8
16:52:03.352 -> Packet queued
16:52:03.352 -> Packet queued
16:52:03.352 -> 5956: EV_JOINING
16:52:03.352 -> 6050: engineUpdate, opmode=0xc
16:52:09.517 -> 392066: engineUpdate, opmode=0xc
16:52:09.595 -> 392380: TXMODE, freq=868100000, len=23, SF=7, BW=125, CR=4/5, IH=0
16:52:14.479 -> 700781: RXMODE_SINGLE, freq=868100000, SF=7, BW=125, CR=4/5, IH=0, rxsyms=255
16:52:14.722 -> 715185: JaccRX1, dataLen=33
16:52:14.722 -> 715756: EV_JOINED
16:52:14.722 -> 715781: engineUpdate, opmode=0x808
16:52:14.722 -> 716180: TXMODE, freq=868100000, len=20, SF=7, BW=125, CR=4/5, IH=0
16:52:15.641 -> 774197: RXMODE_SINGLE, freq=868100000, SF=7, BW=125, CR=4/5, IH=0, rxsyms=255
16:52:16.264 -> 812985: RXMODE_SINGLE, freq=869525000, SF=9, BW=125, CR=4/5, IH=0, rxsyms=255
16:52:17.322 -> 878186: EV_TXCOMPLETE (includes waiting for RX windows)
16:52:17.322 -> 878230: engineUpdate, opmode=0x900
16:52:37.347 -> ############    COUNT=2    ############
16:52:37.347 -> The temperature and humidity:
16:52:37.347 -> [26.09℃,66.26%]
16:52:37.347 -> 2130364: engineUpdate, opmode=0x908
16:52:37.347 -> 2131003: TXMODE, freq=868100000, len=27, SF=7, BW=125, CR=4/5, IH=0
16:52:37.383 -> Packet queued
16:52:37.383 -> Packet queued
16:52:38.289 -> 2189660: RXMODE_SINGLE, freq=868100000, SF=7, BW=125, CR=4/5, IH=0, rxsyms=255
16:52:38.929 -> 2228448: RXMODE_SINGLE, freq=869525000, SF=9, BW=125, CR=4/5, IH=0, rxsyms=255
16:52:39.958 -> 2293650: EV_TXCOMPLETE (includes waiting for RX windows)
16:52:39.958 -> 2293697: engineUpdate, opmode=0x900
```

Figure 5.23 - The data was sent to TTN

As shown in Figure 2.12, after the packet is sent to the TTN, the TTN will forward it to the Cayenne using the Cayenne LPP. Cayenne LPP allows devices to send different sensor's data in different frames. To do this, each sensor data must be prefixed by two bytes [41]:

- Data Channel: Uniquely identifies each sensor in the device across frames, eg. "indoor sensor"
- Data Type: Identifies the data type in the frame, eg. "temperature".

The Cayenne LPP data types is shown in Table 5.

Table 5- Cayenne LPP data types

Type	IPSO	LPP	Hex	Data Size	Data Resolution per bit
Digital Input	3200	0	0	1	1
Analog Input	3202	2	2	2	0.01 Signed
Temperature Sensor	3303	103	67	2	0.1 °C Signed MSB
Humidity Sensor	3304	104	68	1	0.5 % Unsigned
GPS Location	3336	136	88	9	Latitude : 0.0001 ° Signed MSB
					Longitude : 0.0001 ° Signed MSB
					Altitude : 0.01 meter Signed MSB

The RH'T data's resolution in present work is 0.01 Signed and the location

data's resolution is 0.0001° Signed. Thus, the analog Input and GPS location Cayenne LPP data types was used to upload the data.

5.2.4 Software flow to select Advance node

The software flow is used to judge that a WSN end node is a Normal node (ZigBee enable) or an Advance node (ZigBee & LoRa enable). Every WSN end nodes have 10 chances to apply for joining the TTN and every time they wait 1 second for the request from TTN. As soon as the WSN end node receives the request from the TTN, this WSN end node will stop to send the apply to TTN and let the 'FlagAdvance' equal 1. The software flow is shown in Figure 5.25.

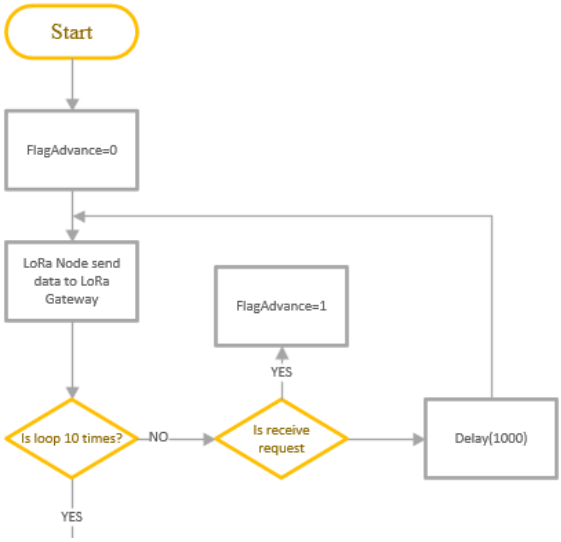


Figure 5.24 - The software flow to select Advance node

Chapter 6 – Experimental Results and Discussion

This chapter includes the experimental results of the developed system. Tests were carried out on the consumption of the system, and the system itself collecting real data from the sensors implemented. The test of the system consumption was made using a power supply with an ammeter. The accuracy of the values read by the temperature and humidity were also tested. Preliminary tests were carried out with only two sensors and then tests were performed on the complete system, with all sensors, connected to the database and mobile application.

6.1 Validation of the RH and temperature measurement capabilities

We used the AURIOL weather station that provides the reference values for temperature and humidity. The absolute errors were calculated under various temperature and humidity conditions. We used 5 sets of similar temperature and humidity data to check the accuracy of the data, and each set of data was measured twice. The results are shown in Table 6 [42].

Table 6 - Temperature and humidity measurements by SI7021 and AURIOL weather stations

Times	Si7021		Weather station	
	Temperature (°C)	Humidity (%)	Temperature (°C)	Humidity (%)
1(a)	25.15	52.49	25.1	52
1(b)	25.15	51.26	25.1	51
2(a)	13.76	48.69	13.7	49
2(b)	13.92	48.45	13.9	48
3(a)	8.37	68.75	8.4	69
3(b)	8.76	74.43	8.8	74
4(a)	9.85	78.39	9.8	78
4(b)	9.93	79.57	9.9	79
5(a)	12.43	73.89	12.4	73
5(b)	12.97	71.78	13.0	71

The absolute errors were calculated by using equation (6.1).

$$\text{error}=\frac{|value_{WS}-value_{SI}|}{value_{WS}}\times 100\% \quad (6.1)$$

Where $value_{ws}$ is the value acquired by AURIOL Weather station, $value_{ws}$

is the value acquired by Si7021.

Table 7 - The absolute error of the temperature and humidity

Times	Absolute error of the Temperature (%)	Absolute error of the Humidity (%)
1(a)	0.19	0.94
1(b)	0.19	0.50
2(a)	0.43	0.63
2(b)	0.14	0.93
3(a)	0.35	0.36
3(b)	0.45	0.58
4(a)	0.51	0.50
4(b)	0.30	0.72
5(a)	0.24	1.21
5(b)	0.23	1.09

The absolute error is shown in Table 6. The absolute error of each test was less than 5%. Therefore, it is possible to affirm that the temperature and humidity data obtained by Si7021 is credible [42].

6.2 Validation communication capabilities of the system

6.2.1 The maximum throughput

Throughput is the number of data packets received per second. Throughput can be referred to as the bandwidth in actual conditions. Bandwidth is more fixed, while the throughput is dynamic, depending on the current traffic. Throughput has a unit of bits per second. The throughput can be calculated by

$$Throughput_{kbps} = Packetsize * 8(bits/byte) * packets_{sec}$$

Where $Packetsize$ is the size of the packet and $packet_{sec}$ is the maximum number of packets be sent in per second.

Two XBee S2C working in the API model were used to do this test. The XCTU was used to measure the throughput for 100 seconds. The result (Figure 6.1) shows that the 351 packets, 89505 bytes were sent. The average transfer ratio is 7.15Kbps.

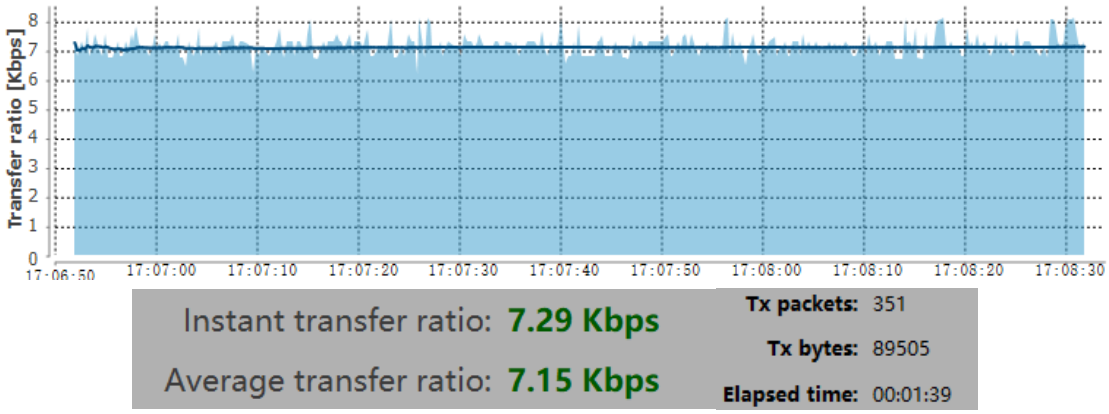


Figure 6.1- The maximum throughput of XBee S2C working in API model

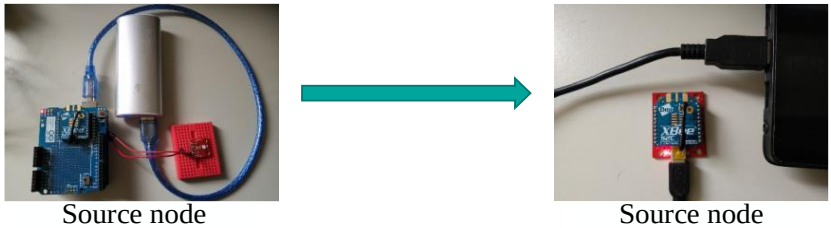
6.2.2 The packet loss rate

Packet loss is the number of packets lost during the transmission process from the transmitter to the receiver. Packets loss occurs when one or more data packets passing through a network fail to reach its destination [43]. The packets loss rate can be calculated based on formula 6.2.

$$Packet\ Loss\ Rate = \frac{N_{Loss}}{N_{total}} \tag{6.2}$$

Where N_{Loss} is the number of lost packet in transmission period, N_{total} refers to the total number of transmitted packets.

Four tests were completed to measure the packet loss rate by putting two sets of equipment in different transfer distance. As shown in Figure 6.2, the first test equipment was consisted of two XBee S2C which were working in the API mode. One of the XBee S2C is the source node, it work with a battery and Si7021. Another node is the destination node, it was connected with the PC. The packets were sent from the source node to the destination node.



Source node Source node

Figure 6.2 - Two XBee S2C node experiment diagram

As shown in Figure 6.3, the second test equipment was consisted of three

XBee S2C. In this case, the packets were sent from the source node to the forward node, and then the forward node forwarded the packets to the destination node.

Figure 6.3 - Three XBee S2C node experiment diagram

In the first set of tests, the distance between the source node and the destination node of two Xbee S2C equipment is 1m. The distance between the source node from the forward node and the forward node from the destination node are both 0.5m in three Xbee S2C equipment.

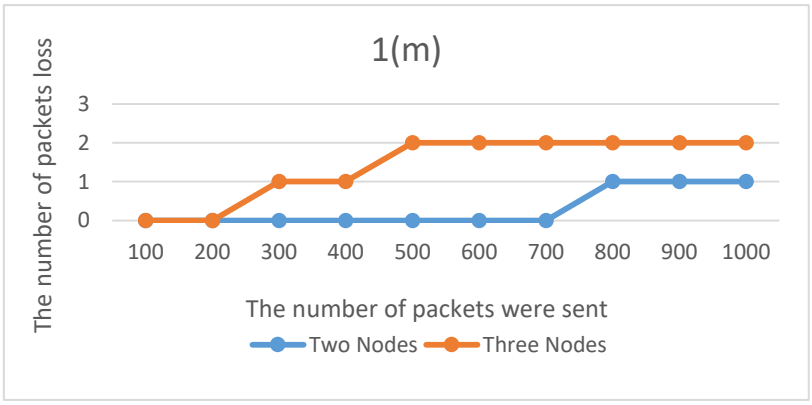


Figure 6.4- The number of packets loss when distance is 1(m)

Figure 6.4 shows the number of packets loss when transfer 1000 packets from the source node to the destination node. The two Xbee S2C equipment 1 loss 1 packets and three Xbee S2C equipment loss 2 packets.

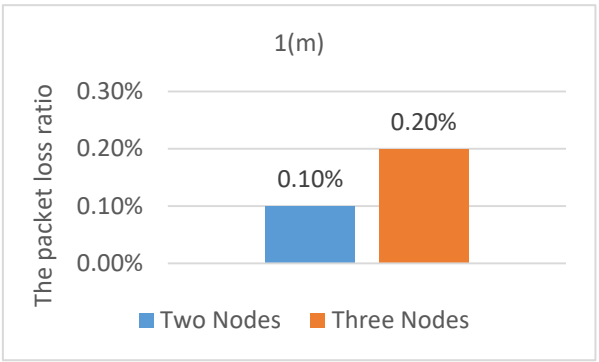


Figure 6.5 - The packets loss ratio when distance is 1(m)

The packets loss ratio of the two Xbee S2C equipment is 0.1% and packets loss ratio of three Xbee S2C equipment is 0.2% (as shown in Figure 6.5).

In the second set of tests, the distance between the source node and the destination node of two XBee S2C equipment is 10m. The distance between the source node from the forward node and the forward node from the destination node of three XBee S2C equipment are both 5m. Figure 6.4 shows the number of packets loss when transfer 1000 packets from the source node to the destination node.

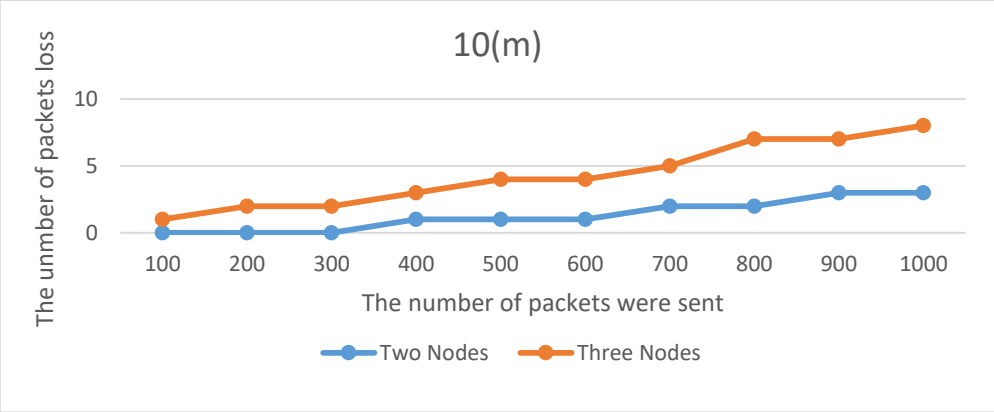


Figure 6.6- The number of packets loss when distance is 10(m)

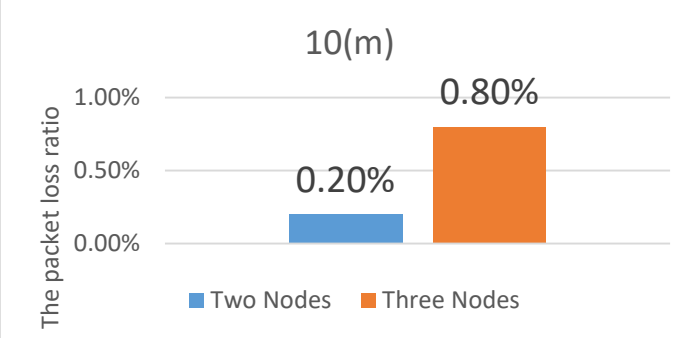


Figure 6.7- The packets loss ratio when distance is 10(m)

Figure 6.7 shows that the packets loss ratio of the two Xbee S2C equipment is 0.2% and packets loss ratio of three Xbee S2C equipment is 0.8%.

6.3 Indoor Communication Tests

Three end node (shown in Figure 6.8) was used to make the experiment.

- As shown in Figure 6.8 (a), the Advance node (LoRa & ZigBee available) consists of Arduino UNO, XBee S2C, LoRa node and a 5600mAh battery.

- As shown in Figure 6.8 (b), the Normal node 1 (only ZigBee available) consists of Arduino UNO, XBee S2C and a 5600mAh battery.
- As shown in Figure 6.8 (c), the Normal node 2 (only ZigBee available) consists of Arduino UNO, XBee S2C, Si7021, Adafruit Ultimate GPS Logger Shield and a 5600mAh battery.

(a) (b) (c)

Figure 6.8 - The location of the nodes

In this test, the data was acquired by Normal node 2 and was sent from Normal node 2 to Normal node 1 by using XBee S2C. Then the data was forwarded from Normal node 1 to Advance node by using XBee S2C. After that, the Advance node sent the data to the LoRa gateway and, in turn, to the TTN. The goal of this test is to prove that the WSN end node is possible to forward data by the ZigBee network when the LoRa module node can not work.

The localtion of those nodes is shown in Figure 6.9. As shown in Figure 6.9 (a), the Advance node was put on the first floor and Normal node 1 was put on the second floor. The distance between the Advance node and Normal node 1 is 6.4 (m).

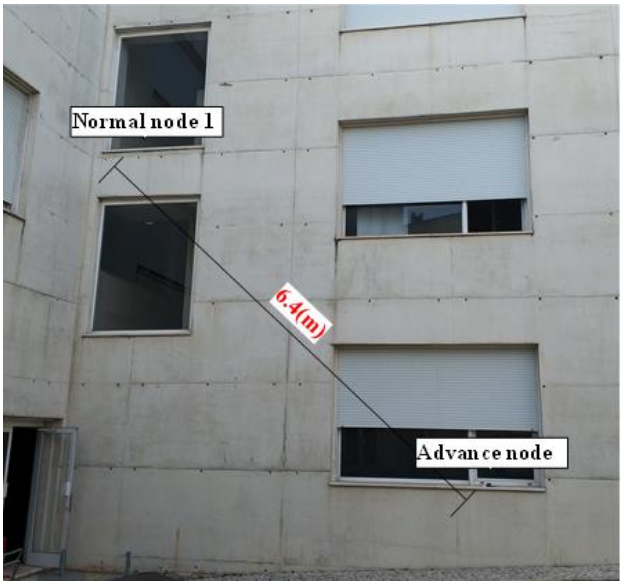


Figure 6.9 (a) - The location of the nodes in experiment

As shown in Figure 6.9 (b), the Normal node 2 was put on the third floor. The distance between Normal node 1 and Normal node 2 is 5.8(m).

Figure 6.9 (b) - The location of the nodes in experiment

As shown in Figure 6.10, TTN can receive the data acquired by Normal node 2. And the data include temperature, humidity, altitude, latitude and longitude. It proves that all of the data can transfer to the Advance node by the ZigBee network and can forward by LoRa available WSN end node.

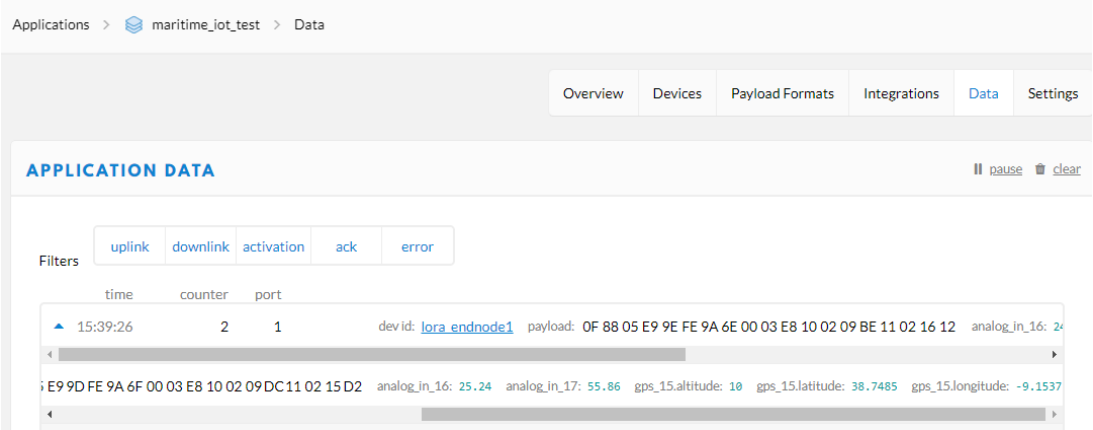


Figure 6.10 - The data acquired by Normal node 2

6.4 Field Tests

Tests were performed in a real environment with the objective of testing the system as a whole and the usability of the equipment. In this test, the WSN end node consists of Arduino UNO, XBee S2C, LoRa node, Si7021, Adafruit Ultimate GPS Logger Shield and a 5600mAh battery. The WSN node is fixed on the outside of the reefer container and the Si7021 is put into the container (as shown in Figure 6.11).

Figure 6.11 - The hardware of reefer container monitoring system

The reefer container was moved and the WSN end node acquired the data and transferred the data to the TTN. Cayenne received the data from the TTN, stored the data in the database and displayed the data on the web page.

Temperature and humidity data can be display in two chart in cayenne web page (as shown in Figure 6.12) and mobile APP (as shown in Figure 6.13).

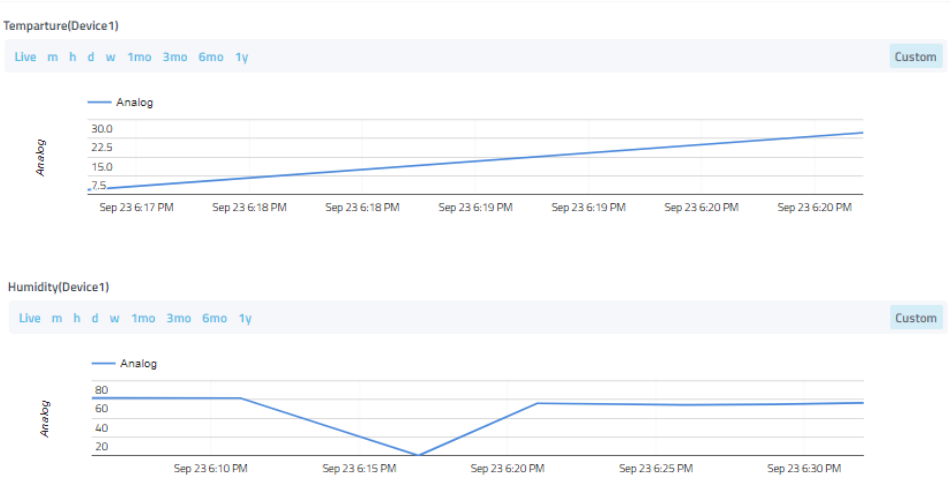


Figure 6.12 - The chart displayed RH'T in cayenne web page

The chart can display the temperature and humidity in real-time and it also can display the data for any time period.

Figure 6.13 - The chart displayed RH'T in cayenne mobile APP

The account and the password are necessary to log in to the mobile application. The chart also can display the temperature and humidity in real-time in the mobile application.

The location of the reefer container can be tracked on the cayenne web page and cayenne mobile APP. The orange line in the image is the path that the reefer container was moved. The green point is the location where this reefer container start to move. The small blue point is the location where this reefer container had been. And the big blue symbol is the real-time location of the reefer container.

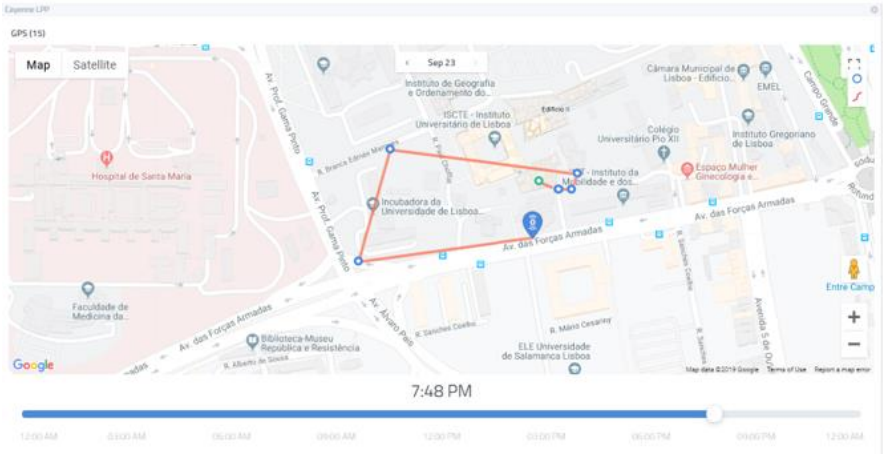


Figure 6.14 - The displayed the location in cayenne web page

An alarm system (as shown in Figure 6.15) is also included in the system with the help of cayenne. For example, when relative humidity was set to below 60 if the relative humidity above 60, the mobile application will send an email to alert this.

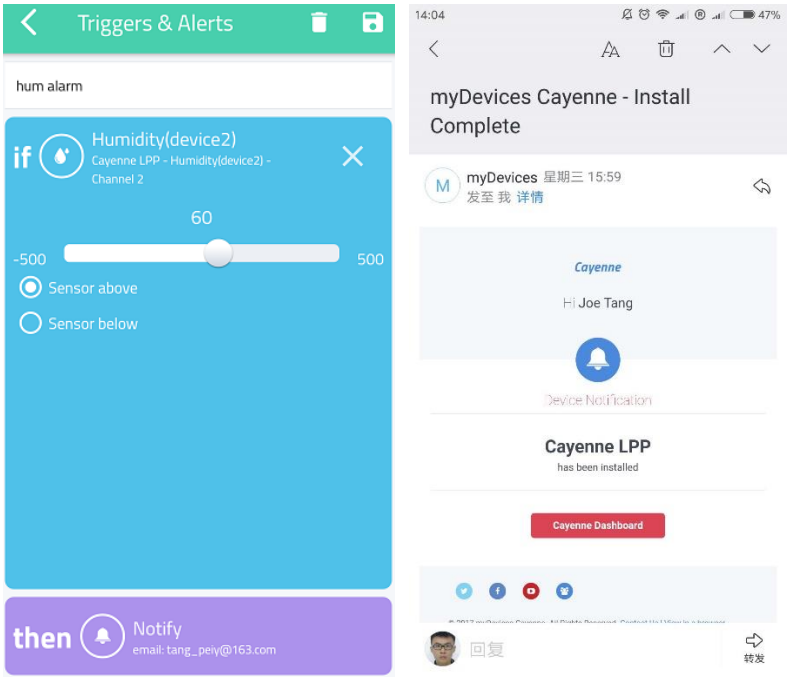


Figure 6.15 - The alarm email sent by cayenne

6.5 System power consumption

This section describes the consumption of the developed system. As shown in Figure 6.17, an ammeter supply power to the reefer container system and it also allowed measuring the individual current of each WSN end node.

Figure 6.17 - Power supply with an ammeter

The power consumption of each device is shown in Table 8.

Table 8 - The power consumption of each device

Device	Current consumption[mAh]
Arduino UNO	45
XBee S2C and Arduino Wireless Shield	35
Adafruit Ultimate GPS Logger Shield	28
Dragino LoRa end node	19
Si7021	2

Figure 6.18 shows the percentage of energy consumed by each device. The total power consumption of the reefer container is 129 [mAh].

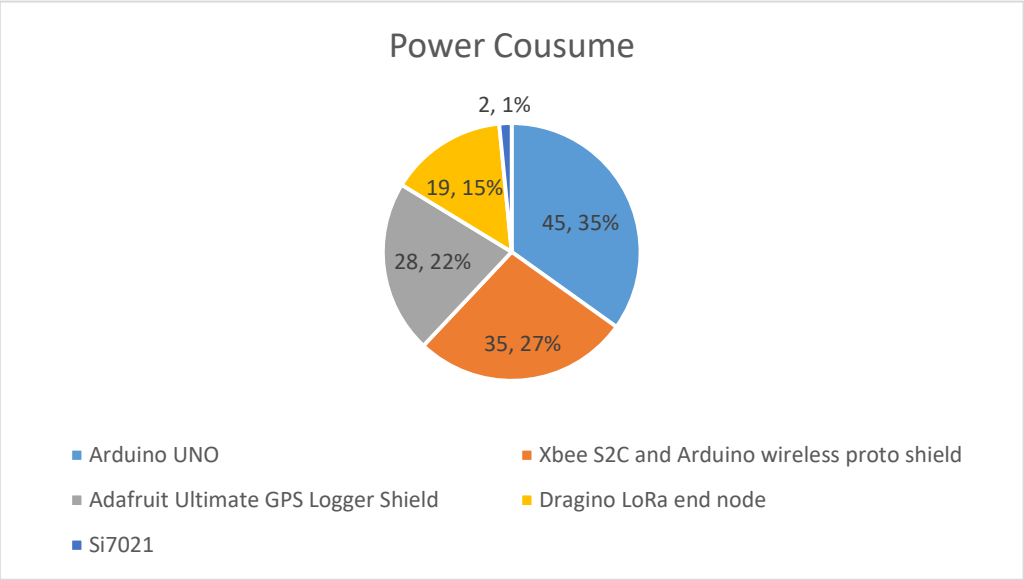


Figure 6.16 - The percentage of energy consumed by each device

Chapter 7 - Conclusions and Future Work

7.1 Conclusions

An IoT system for container monitoring was designed and implemented, which was the main goal of this dissertation.

The WSN end node architecture, WSN communication architecture and the system architecture were designed and implemented. With the help of WSN end node architecture, the WSN end node is possible to acquire the RH'T data inside of the reefer container and the location data of the reefer container and transmit the data by using ZigBee and LoRa. The WSN communication architecture provides the capability for the WSN end node to transfer the data to the LoRa gateway even if the LoRa communication link of the WSN end node is blocked by other containers. With the system architecture, every WSN end node has the capability to send the data to the TTN and display in both cayenne web application and mobile application.

Considering the container stacking rules, the radio energy dissipation model of the stored container was used to calculate the optimal number of Cluster Nodes which is an important parameter of the LEACH routing algorithm. The embedded software was developed by using the LEACH routing algorithm to guarantee a well-balanced distribution of the energy load among nodes of the sensor network.

The validation of the RH'T measurement capability and communication capabilities of WSN nodes were tested. The RH'T acquire by AURIOL weather station was compared with the result read by Si7021. The absolute error was calculated to affirm that the RH'T data obtained by Si7021 is credible. The maximum throughput and packet loss rate were measured to validate the WSN end node has the capability to forward the data in the ZigBee network.

Through the laboratory tests proved that WSN end nodes have the ability to forward data from Normal node (only ZigBee available) to Advance node (ZigBee & LoRa available) by using ZigBee network and Advance node has the ability to forward the data from Normal node to LoRa gateway. Through the field test, the RH'T and location of the reefer container can be displayed in real-time and it also can be displayed the data for any time period.

7.2 Contributions

This dissertation's main contributions are:

- A practical approach regarding communication protocols for WSN;
- Design and implementation of the WSN end node for the monitoring of reefer container including measurement of temperature, relative humidity and location;
- Development and implementation of a routing algorithm for extending the stability period;
- Development and implementation of web application and mobile application for data visualization.

The developed work and the initial results were included in the article and the poster in Appendix A, which were accepted and presented at the ATEE 2019 international conference, MARCH 28-30, 2019 Bucharest, Romania. The article will be published in IEEEXplorer ("https://ieeexplore.ieee.org/abstract/document/8724950").

Peiyao Tang, Octavian Adrian Postolache, Yangyang Hao, Meisu Zhong, "Reefer Container Monitoring System", ATEE 2019, Bucharest, Romania, 11th International Symposium on Advanced Topics in Electrical Engineering (ATEE).

7.3 Future Work

Despite being a fairly complete system, improvements can still be made, both software and hardware. Further testing of the system may be performed in a really container terminal.

Smaller PCB circuits can be created to improve the reliability of the WSN end node.

The routing algorithm can be improved to further extend the stability period.

The system can also be improved at the data processing, some inaccurate data can be eliminated.

References

[1] Mahlknecht, Stefan, and Sajjad A. Madani. "On architecture of low power wireless sensor networks for container tracking and monitoring applications." 2007 5th IEEE International Conference on Industrial Informatics. Vol. 1. IEEE, 2007.
[2] Rodríguez-Bermejo, Jose, et al. "Thermal study of a transport container." Journal of Food Engineering 80.2 (2007): 517-527.
[3] Li Jinlong. Container Logistics Practice [M]. Tsinghua University Press, 2010. (in Chinese)
[4] Ma Ting, Li Fang, Shan Daya. Research on Tracking and Traceability of Food Cold Chain Logistics Based on Internet of Things Technology[J]. Journal of University of Shanghai For Science and Technology, 2013, 35(6): 557-562. (in Chinese)
[5] GOU Junlai. Research on the Status Quo and Strategy of Cold Chain Logistics Development in China[J]. Modern Business Industry, 2010, 22(21): 13-14. (in Chinese)
[6] Container lines speed up their assault on reefer cargo. Drewry
[7] Zhang, Ning, and Yingjie Liu. "NB-IOT Drives Intelligent Cold Chain for Best Application." 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC). IEEE, 2019.
[8] Ruiz-Garcia, Luis, et al. "Testing ZigBee motes for monitoring refrigerated vegetable transportation under real conditions." Sensors 10.5 (2010): 4968-4982.
[9] Jia Yigang. Application of Internet of Things Technology in Environmental Monitoring and Early Warning[J]. Shanghai Construction Science & Technology, 2010(6): 65-67. (in Chinese)
[10] Chen Wei. Talk about those things in the Internet of Things [J]. Today, Yuanyuan, 2011 (20): 54-59. (in Chinese)
[11] He Limin. Definition and Development History of Embedded Systems[J]. Microcontrollers & Embedded Systems, 2004(1): 6-81
[12] Kaelbling L. Learning in embedded systems.[M]// Learning in embedded systems. 1993.
[13] McComb, Gordon. Arduino robot bonanza. McGraw Hill Professional, 2013.
[14] Asha, K. R., et al. "Real Time Speed Control of a DC Motor by Temperature Variation Using LabVIEW and Arduino." 2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT). IEEE, 2017.
[15] Teikari, Petteri, et al. "An inexpensive Arduino-based LED stimulator system for vision research." Journal of neuroscience methods 211.2 (2012): 227-236.
[16] “ Arduino Compare board specs.” [Online]. Available: https://www.arduino.cc/en/Products/Compare. [Accessed: 10-Sep-2019].
[17] Kumar, B. Harish. "WSN based Automatic Irrigation and Security System using Raspberry Pi Board." 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC). IEEE, 2017.
[18] Parthornratt, Tussanai, Natchaphon Burapanonte, and Wisarute Gunjarueg.

"People identification and counting system using raspberry Pi (AU-PiCC: Raspberry Pi customer counter)." 2016 International Conference on Electronics, Information, and Communications (ICEIC). IEEE, 2016.

[19] “ Orangepi Wiki.” [Online]. Available: http://www.orangepi.org/Docs/mainpage.html. [Accessed: 10-Sep-2019].

[20] Rappaport, Theodore S. "Wireless Communications--Principles and Practice, (The Book End)." Microwave Journal 45.12 (2002): 128-129.

[21] Powner, E. T., and F. Yalcinkaya. "From basic sensors to intelligent sensors: definitions and examples." Sensor Review 15.4 (1995): 19-22.

[22] Song, Eugene Y., Gerald J. FitzPatrick, and Kang B. Lee. "Smart sensors and standard-based interoperability in smart grids." IEEE sensors journal 17.23 (2017): 7723-7730.

[23] Farahani, Shahin. ZigBee wireless networks and transceivers. Newnes, 2011.

[24] IEEE 802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifi cations for Low-Rate Wireless Personal Area Networks (WPANs), Sept. 2006.

[25] “A technical overview of LoRa® and LoRaWAN™” [Online]. Available: https://lora-alliance.org/sites/default/files/2018-04/what-is-lorawan.pdf [Accessed: 10-Sep-2019].

[26] “ LoRaWAN with ProRF and The Things Network” [Online]. Available: https://learn.sparkfun.com/tutorials/lorawan-with-prorf-and-the-things-network/all#the-things-network [Accessed: 10-Sep-2019].

[27] “ Network Architecture” [Online]. Available: https://www.thethingsnetwork.org/docs/network/architecture.html#processing-flow-of-uplink-messages [Accessed: 10-Sep-2019].

[28] “Features of the Cayenne” [Online]. Available: https://developers.mydevices.com/cayenne/docs/features/ [Accessed: 10-Sep-2019].

[29] “Cayenne Low Power Payload” [Online]. Available: https://developers.mydevices.com/cayenne/docs/lora/ [Accessed: 10-Sep-2019].

[30] Tanghe, Emmeric, et al. "Intra-, inter-, and extra-container path loss for shipping container monitoring systems." IEEE Antennas and Wireless Propagation Letters 11 (2012): 889-892.

[31] “Adafruit Ultimate GPS Logger Shield” [Online]. Available: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-ultimate-gps-logger-shield.pdf [Accessed: 10-Sep-2019].

[32] “Si7021-A10” [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Weather/Si7021.pdf [Accessed: 10-Sep-2019]

[33] “Digi XBee3® 802.15.4 Radio Frequency (RF) Module” [Online]. Available: https://www.digi.com/resources/documentation/digidocs/PDFs/90002273.pdf [Accessed: 10-Sep-2019]

[34] “API mode overview” [Online]. Available: https://www.digi.com/resources/documentation/Digidocs/90001500/Concepts/c_a

pi_mode_overview.htm?TocPath=Operate%20in%20API%20mode%7C_____1 [Accessed: 10-Sep-2019]

[35] “API frame structure” [Online]. Available: https://www.digi.com/resources/documentation/Digidocs/90001942-13/concepts/c_api_frame_structure.htm?tocpath=XBee%20API%20mode%7C_____2 [Accessed: 10-Sep-2019]

[36] “LG01-N Single Channel LoRa IoT Gateway” [Online]. Available: https://www.dragino.com/products/lora/item/143-lg01n.html [Accessed: 10-Sep-2019]

[37] “LG01 LoRa Gateway User Manual” [Online]. Available: https://www.dragino.com/downloads/downloads/UserManual/LG01_LoRa_Gateway_User_Manual.pdf [Accessed: 10-Sep-2019]

[38] “Arduino Uno SMD” [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardUnoSMD [Accessed: 10-Sep-2019]

[39] Brahim, Elbhuru, et al. "Stochastic and Balanced Distributed Energy-Efficient Clustering (SBDEEC) for Heterogeneous Wireless Sensor Networks." INFOCOMP 8.3 (2009): 11-20.

[40] “APB vs OTAA devices” [Online]. Available: https://docs.exploratory.engineering/lora/lora_key_concepts/ [Accessed: 10-Sep-2019]

[41] “Cayenne Low Power Payload” [Online]. Available: https://developers.mydevices.com/cayenne/docs/lora/#lora-cayenne-low-power-payload [Accessed: 10-Sep-2019]

[42] Tang, Peiyao, et al. "Reefer Container Monitoring System." 2019 11th International Symposium on Advanced Topics in Electrical Engineering (ATEE). IEEE, 2019.

[43] Piyare, Rajeev, and Seong-ro Lee. "Performance analysis of XBee ZB module based wireless sensor networks." International Journal of Scientific & Engineering Research 4.4 (2013): 1615-1621.

Appendix A – Scientific Articles

Article: **Reefer Container Monitoring System**. This article has been accepted and presented at the ATEE 2019 11th International Symposium on Advanced Topics in Electrical Engineering (ATEE) and will be published in IEEEXplorer.

The 11th International Symposium on
ADVANCED TOPICS IN ELECTRICAL ENGINEERING

Organized by the The Faculty of Electrical Engineering The Doctoral School of Electrical Engineering University POLITEHNICA of Bucharest.

THE 11th INTERNATIONAL SYMPOSIUM ON ADVANCED TOPICS IN ELECTRICAL ENGINEERING
March 28-30, 2019
Bucharest, Romania

Reefer Container Monitoring System

Peiyao Tang[1, 2], Octavian Adrian Postolache[1], Yangyang Hao[2] and Meisu Zhong[2]
[1]ISCTE-University Institute of Lisbon, Instituto de Telecomunicacoes, Lisbon, Portugal
[2]Shanghai Maritime University, Shanghai, China
ptgoe@iscte-iul.pt, opostolache@lx.it.pt, yyhao@shmtu.edu.cn, zhongmeisu@163.com

***Abstract*- As the main logistics carrier of international transportation, the container is the main carrier of import and export goods. The role of containers in modern logistics systems has become increasingly prominent. The reefer container maintains temperature and humidity values within a specified range, which can ensure a safe transportation of food and medicines. Therefore, continuous monitoring and transmission of temperature and humidity in the reefer container is of great significance for ensuring the safety of transported products. In this paper, the WSN-IoT (Internet of Thing based on Wireless Sensor Network） monitoring system for reefer containers is presented. The system hardware and software and the experimental results are discussed.**

***Keywords*: reefer container monitoring, IoT, Firebase, ZigBee**

I. INTRODUCTION

The container is the main logistics carrier of international shipping, and containers mainly transport the imported and exported goods. Almost 90% of the world trade is accomplished with the help of containers and they can be used in different means of transportation, including ships and trains [1]. Container transportation is characterized by high efficiency, convenience, and safety, and has an important position and role in modern logistics systems. However, Environmental variations that are unfavorable to the quality may occur in the transported procedure. Therefore, it is important to monitor the status of the container to maintain the quality of transported goods.

The state of fruit, vegetables, dairy products and meat transported in the reefer container should be monitored to guaranty the quality maintenance. Temperature variations and humidity variations may affect the quality of those transported goods. The limit can be quite strict for the products with a storage temperature near 0 ℃, because it has a risk in the growth of microbial when the temperature beyond 0 ℃. The work focusses on distributed measurement of temperature and humidity for reefer containers to provide the information about the cargo during transportation [2].

Locally the wired sensors represent a good solution however the connectivity faults can appear during the loading and unloading of containers. Wireless sensor network may be considered a reliable solution. Thus, the wireless sensor nodes may continuously measure the temperature and humidity of the container as so as the container localization. The data delivered by WSN (Wireless Sensor Network) nodes are stored using a remote database that is part of a cloud service or a remote server. The data analysis results are presented on the user interface in real-time.

In this paper is presented an IoT (Internet of Thing) system for reefer container parameters monitoring based on WSN nodes that are attached to different containers distributed on the storage yard. The WSN nodes provide the inside temperature, humidity and provide real-time container tracking using GPS (Global Positioning System) receiver. The temperature and humidity together the time and localization values are delivered from sensors. Thus values of temperature and humidity out of recommended variation ranges are signaled, and time stamp and localization of container (GPS coordinates) are provided.

We experimented with a refrigerator instead of a reefer container that transported tomatoes. We monitored the temperature and humidity in the refrigerator. We also simulated the process of transporting containers from the ship to the yard and monitored the location of the refrigerator during the process.

II. RELATED WORK

In [3], the IoT system of cold chain container was researched. In this project a continuous monitoring of temperature was made, with multiple levels of alarm if the product had a problem. In [4], a container logistics monitoring system that combines WSN and GPRS was introduced. The location algorithm of shipborne wireless sensor network nodes based on the research of shipboard structure and shipborne wireless sensor network working environment are proposed in [5]. Authors in [6] proposed a solution for constructing a communication network for stacked containers by deploying communication nodes on the surface of containers to form a customized wireless sensor network. A WSN-based cargo monitoring system was introduced in [7]. An application of wireless sensor network that is dedicated to monitoring temperature-sensitive cold chain products was proposed in [8]. In [9], a wireless sensor module (WSM) and network, based on the IEEE 802.15.4 standard, for cold chain logistics reefer container temperature monitoring was proposed. Authors in [10] proposed a container cargo monitoring system based on wireless sensor networks, which use tilt/motion sensors to improve the network convergence time of the container. Authors in [11] explored the potential of wireless sensor technology to monitor fruit storage, using ZigBee technology with a special focus on two different business modules (Xbow

978-1-7281-0101-9/19/$31.00 ©2019 IEEE

and Xbee). In [12], a wireless sensor network node for shipborne dangerous goods container monitoring was designed. Authors in [13] tested wireless sensor nodes based on the ZigBee protocol during actual shipment. Authors in [14] demonstrated the compatibility and feedback of RFID and WSN to improve the refrigeration of perishable foods. Authors in [15] proposed a wireless network system that supports the integration of RFID, sensors, ZigBee, cellular mobile networks and satellite mobile networks to support local data collection and remote data transmission.

III. SYSTEM ARCHITECTURE

This section presents the architecture of the monitoring system, focusing on the main components of the system. Then, each of the systems' components are described, and it is introduced the software and the functions that can be implemented. The architecture of the container monitoring system is shown in Fig .1, and it can be divided in four parts:

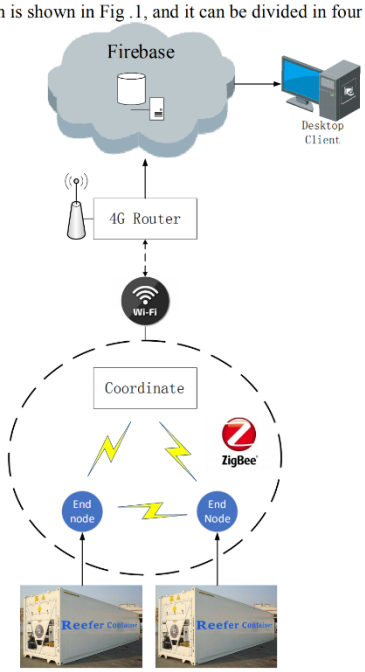


Fig. 1. General architecture of the system

The sensors are used to measure temperature, humidity, and location of the reefer container. The Arduino UNO is responsible for acquiring the temperature, humidity and location data and control the data transmission.

The temperature and humidity sensors are placed inside the container and the other components of the end node are placed in outside the container. (Fig. 2).

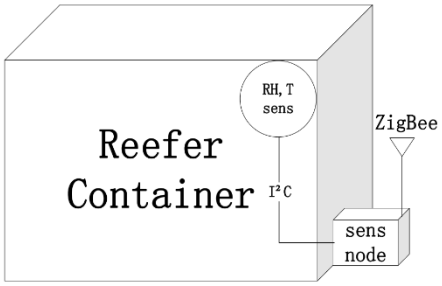


Fig. 2. Reefer Container – WSN sensor node for RH and temperature measurement

The Xbee is used to transfer local data in the ZigBee network. And the Raspberry Pi2 is responsible for sending the data to the Firebase to complete the task of remote data transfer.

Firebase is responsible for the storage of the data collected by the sensors in its Realtime Database. Firebase stores and synchronizes data with the NoSQL cloud database. The data is also synced to a Web page in real time.

The Web application reads the data that is stored in Firebase.

A. *The end node*

The end node consists of a Si7021, Arduino Uno, Xbee S2C and Adafruit Ultimate GPS Logger Shield. It is used to acquire data and send local data in ZigBee network. The structure of the end node is shown in Fig. 3.

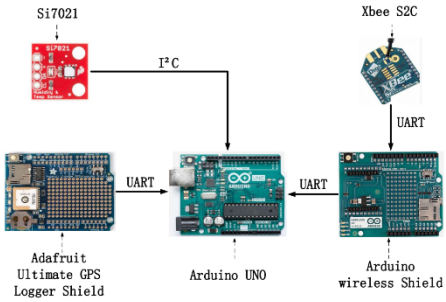


Fig. 3. The WSN end node modules

Our system uses Si7021 to measure temperature and humidity data and utilizes the Adafruit Ultimate GPS Logger Shield to measure the location data. To acquire data and control the transmission of the local data, we take advantage of

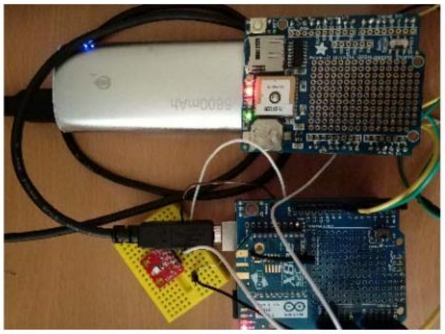


Fig. 5. WSN node: Arduino Uno with GPS shield and ZigBee Shields and power bank

We connected the Xbee S2C with Raspberry Pi 2 by using Exploring Xbee and by putting a Wi-Fi adapter on the Raspberry Pi 2. Then the Raspberry Pi 2 can receive the data from the end node through the ZigBee network, and upload the data to the Firebase over the Internet. It was used a battery with 10400mAh capacity to provide energy for the coordinator. The connected coordinator is shown in Fig. 6.

Fig. 6. The coordinator implementation: Raspberry PI 2 with WiFi, power bank, ZigBee Coordinator

We used the AURIOL weather station that provides the reference values for temperature and humidity. The absolute errors were calculated under various temperature and humidity conditions. We used 5 sets of similar temperature and humidity data to check the accuracy of the data, and each set of data was measured twice. The results are shown in TABLE I.

TABLE I
TEMPERATURE AND HUMIDITY MEASURE BY SI7021 AND AURIOL WEATHER STATION

Times	Si7021		Weather station	
	Temperature (°C)	Humidity (%)	Temperature (°C)	Humidity (%)
1(a)	25.15	52.49	25.1	52
1(b)	25.15	51.26	25.1	51
2(a)	13.76	48.69	13.7	49
2(b)	13.92	48.45	13.9	48
3(a)	8.37	68.75	8.4	69
3(b)	8.76	74.43	8.8	74
4(a)	9.85	78.39	9.8	78
4(b)	9.93	79.57	9.9	79
5(a)	12.43	73.89	12.4	73
5(b)	12.97	71.78	13.0	71

The absolute errors were calculated by using equation (1).

$$\text{error}=\frac{|value_{WS}-value_{SI}|}{value_{WS}}\times 100\% \qquad (1)$$

where:

$value_{WS}$ is the value acquired by AURIOL Weather station;

$value_{SI}$ is the value acquired by Si7021.

TABLE □
THE ABSOLUTE ERROR OF THE TEMPERATURE AND HUMIDITY

Times	Absolute error of the Temperature (%)	Absolute error of the Humidity (%)
1(a)	0.19	0.94
1(b)	0.19	0.50
2(a)	0.43	0.63
2(b)	0.14	0.93
3(a)	0.35	0.36
3(b)	0.45	0.58
4(a)	0.51	0.50
4(b)	0.30	0.72
5(a)	0.24	1.21
5(b)	0.23	1.09

The absolute error is shown in TABLE □. The absolute error of each test was less than 5%. Therefore, it is possible to affirm that the temperature and humidity data obtained by Si7021 is credible.

After ensuring the credibility of the temperature and humidity measurements, the code for the Arduino was written. Arduino was programmed to acquire the temperature, humidity, date, time, and location data (with a Baud rate of 115200bit /s) and to send the data to the Raspberry Pi 2 every 20 seconds. Then, in order to reduce the location data error caused by the poor GPS signal, we designed and implemented a data processing method. If the current sampling speed is more than twice of the speed obtained by the previous sampling, and the repair quality of the sampling is low, the sampling is considered invalid.

Then, we configured the Xbee S2C by using XCTU. We set the coordinator address to the destination address of the data transfer, so that the end device will transfer the data to the coordinator. Then, Baud rate was set to 115200bit/s. Finally, the data wireless transmission is tested.

As stated above, it was used a Firebase Realtime Database that is hosted in the cloud. The data is stored in JSON format and it is synchronized in real time to each connected client.

the Arduino UNO and Xbee S2C to send local data into the ZigBee network.

Si7021 is used to measure the temperature and humidity data. It has a precision relative humidity sensor with ± 3% RH (max), 0–80% RH, high accuracy temperature sensor with ±0.4 °C (max), –10 to 85 °C, 0 to 100% RH operating range, up to − 40 to +125 °C operating range. It also has a wide operating voltage (1.9 to 3.6 V), low power consumption with 150 µA active current with 60 nA standby current [16].

The Adafruit Ultimate GPS Logger Shield is used to get the location data. It has -165 dBm sensitivity, 10 Hz updates, 66 channels, low power module - only 20mA current draw, half of most GPS assembled & tested shield for Arduino Uno, MicroSD card slot for datalogging onto a removable card and RTC battery included, for up to 7 years backup [17].

Xbee S2C is an RF module designed for wireless communication and it works on ZigBee mesh communication protocols. Indoors, the effective transmission distance of Xbee S2C is around 60 meters and outdoors, and the effective transmission distance is around 1200 meters [18].

Arduino UNO is an ATMEL Microcontroller. It has an ATmega328P Microchip and the operating voltage is 5 Volt. Its input voltage is 7 to 20 Volts, it has 14 digital I/O (in which 6 provide PWM output) and 6 analog input pins. The DC current per I/O pin is 20 mA, and the DC current for the 3.3V pin is 50 mA [19]. The flash memory of the Arduino UNO is 32 KB, in which 0.5 KB is used by bootloader. SRAM is 2 KB, EEPROM is 1 KB and the Clock Speed is 16 MHz.

B. Coordinator

The coordinator consists of a Raspberry Pi 2 and Xbee S2C. The coordinator receives local data by using Xbee S2C and sends the data from the Xbee S2C to the Raspberry Pi 2. The structure of the coordinator is shown in Fig. 4.

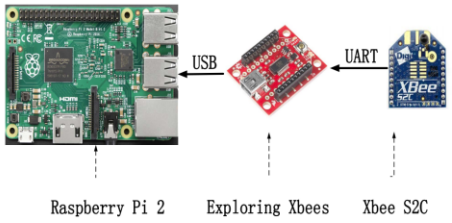


Fig. 4. The WSN coordinator architecture

Our system used the Raspberry Pi 2 to acquire data from the end node and used the Internet to complete the task of remote data transfer. The Raspberry Pi 2 is a small single-board computer. It has Broadcom BCM2836 ARMv7 Quad-Core Processor powered Single Board, running at 900MHz. It has a 1GB RAM, 40 pins extended GPIO, 4 USB ports, 4 poles stereo output and composite video port, full-size HDMI and CSI camera port for connecting the Raspberry Pi camera. It also has a DSI display port for connecting the Raspberry Pi touch screen display, Micro SD port for loading the operating system and storing data and Micro USB power source [20].

C. Software

The developed software consists of different parts. The first part focus on firmware. The second part is the storage of remote data processing based on Firebase and develop a Web application.

The Arduino Uno was programmed with the Arduino IDE to acquire the temperature and humidity data from the temperature sensor si7021, as well as the date, time and location from Adafruit Ultimate GPS Logger Shield. It was also used to control the data transmission frequency and speed. We also acquired the speed and fix quality from Adafruit Ultimate GPS Logger Shield. The speed can indicate the speed of the container and the fix quality can indicate whether the location information is accurate.

XCTU is an application designed to enable developers to interact with Xbee modules through a simple-to-use graphical interface. We used the XCTU platform to configure and test the Xbee S2C. First, a unique address for each Xbee S2C was set. Then, it was configured the type of the Xbee S2C: some devices are part of the end node, and the Xbee S2C that is connected to the Raspberry Pi 2 is the coordinator. At last, it was configured the address and the transfer speed of the Xbee S2C.

The Raspberry Pi 2 was programmed by using Python 3 to send the data to Firebase over the Internet and define the type and structure of the data. It was also programmed to control the data transmission frequency and transmission speed.

Firebase can be used to create a project and activate its Realtime Database. Changing the Firebase rules was necessary in order to give read and write permissions to the Web application.

It was used JSFiddle to develop the Web page. It was possible to design the shape of the page and show the location of containers based on Google Maps.

IV. Experimental Setup and Results

In this part, the multiple components of the end node were connected. It was used a wire to connect the si7021 and Adafruit Ultimate GPS Logger Shield to the Arduino UNO, and a Wireless Shield to connect the Xbee S2C with the Arduino UNO. Then it was used a battery with 5600mAh capacity to provide energy for the end node. The connected end node is shown in Fig. 5.

When building a cross-platform application using the JavaScript SDK, all the clients share a real-time database instance and automatically receive updates with the newest data. To use this Realtime database, we created a project and changed its rules to allow the reading and writing of data. Then, we connected the Raspberry Pi2 to the Realtime Database and started sending the temperature, humidity, date and time to the Firebase as float numbers, and the location data as strings. Using Python 3 to determine the storage structure of data, the data is stored in two kinds of structures at the same time. One structure is only to keep the newest set of data to the specified path; the others are to save each set of data to the specified path. Firebase automatically names each set of data with a specific file name. The Baud rate has been set to 115200bit/s. The storage structure of the data in Firebase is shown in Fig. 7.

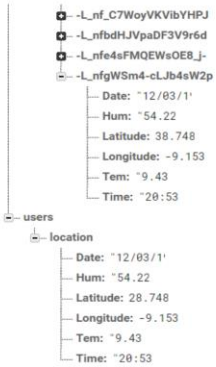


Fig. 7. Storage structure of the data in Firebase

The Web page of the reefer container monitoring was developed based on JavaScript. We added Firebase to this JavaScript project to associate with the Realtime Database in order to access and manipulate the data in this database. To set and draw a line chart to display the temperature and humidity in real-time, the Google chart tools was used.

Fig. 8. Temperature and Humidity; measurement on simulated reefer container: tomato transportation case

In the experiment, we used a refrigerator to simulate a reefer container that transported tomatoes. The best storage temperature for tomatoes is +7 to +15 ℃, and the best storage relative humidity is 44.6% to 59% [21]. We put the end node and tomatoes in the refrigerator and keep the temperature and humidity within the range that suitable for preserving tomatoes (as shown in Fig.8).

Continuous monitored of the temperature and humidity inside of the cooled box was carried out. The acquired and primary processed values were presented by a web APP (as shown in Fig .9). In this chart, the changing curves of temperature and humidity in the last 10 hours can be displayed. This chart also has a function that when a specific point of the curve is clicked, its value will display in a pop-up window.

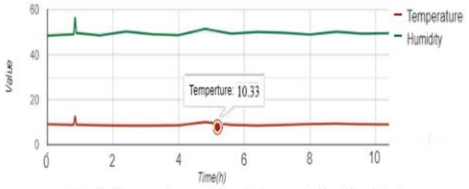


Fig. 9. Temperature and humidity monitoring line chart

A geographic information system component was developed as part of the web APP. The maps customization was performed using Google Maps JavaScript API. We use roadmap as the basic layer of the map (it is one of the basic map types supported by Google Maps API). Then, we add marks to represent the location of the container on the map. For each location mark are considered and can be displayed the information about the time, temperature and humidity.

Fig. 10. Tracking the simulated container during transportation from ship yard to storage yard: simulated scenario

Some low scale experiments were carried out. Thus a refrigerator including tomatoes with an attached WSN node (Fig.8) were used to simulate the reefer container that is transported from ship to the yard. The container historical and current locations are displayed on the map. (Fig. 10). According with the present study the location of the container

in the last 10 hours was presented. The black mark is the historical location of the refrigerator, and the red mark is the current location of the refrigerator. After clicking any mark on this map, the value of date, time, temperature and humidity of the container will display in the pop-up window.

If the temperature or humidity of a container in the yard exceeds the safe range in which the goods can be effectively stored, alarms are generated on the web page. The advantage of this monitoring system is that we can quickly find temperature and humidity changes and verifies if this value corresponds to required values for the stored goods (vegetables, medicine products). This monitoring system can help workers to find abnormities on container function in a short time using the web APP.

V. CONCLUSION

In this paper, we designed and implemented a monitoring system based on WSN ZigBee and Firebase to monitor several parameters that characterize the cooled containers. In container conditions such as temperature, humidity as so as the localization is in real-time on a Web page. This system uses a 4G router to transmit the data to remote server. Additional tests related temperature and humidity measurements were done considering the AURIOL weather station and the obtained values were compared with the values obtained with Si7021. The absolute errors were calculated, and the accuracy level was evaluated. We also process the location data acquired by the Adafruit Ultimate GPS Logger Shield to optimize the location data.

As future work, we intend to develop a routing algorithm to reduce energy consumption and extend the battery life of the end node.

ACKNOWLEDGMENTS

This work is funded by FCT/MEC through national funds and when applicable co-funded by FEDER – PT2020 partnership agreement under the project UID/EEA/50008/2019 and the Smart Sensors IT-IUL-SMU Joint Laboratory.

REFERENCES

[1] Mahlknecht, Stefan, and Sajjad A. Madani. "On architecture of low power wireless sensor networks for container tracking and monitoring applications." 2007 5th IEEE International Conference on Industrial Informatics. Vol. 1. IEEE, 2007.

[2] Rodríguez-Bermejo, Jose, et al. "Thermal study of a transport container." Journal of Food Engineering 80.2 (2007): 517-527.

[3] Astrid M, Karyono K. Communication from Smart Storage Container System Using Bluetooth, ZigBee, and XML-RPC. Advanced Science Letters, 2016.

[4] Wang K, Liang S, Xian X, et al. Design and implementation of a novel monitoring system for container logistics based on Wireless Sensor Networks and GPRS. Icic Express Letters, 2010, 4(5):1905-1911.

[5] Yang X, Zhang Y. Location Algorithm for Nodes of Ship-Borne Wireless Sensor Networks. International Journal of Distributed Sensor Networks, 2013,(2013-9-21), 2013, 2013(8):1172-1175.

[6] Liu J, Chen T, Bao J, et al. SCN: An Efficient Wireless Sensor Network Based Communication Scheme for Stacked Containers. International Journal of Hybrid Information Technology, 2013, 6.

[7] Amrutha V. A wireless sensor network for cargo Chain monitoring. International Journal of Applied Engineering Research, 2015, 10(3):6241-6256.

[8] Carullo A, Corbellini S, Parvis M, et al. A Wireless Sensor Network for Cold-Chain Monitoring. IEEE Transactions on Instrumentation & Measurement, 2009, 58(5):1405-1411.

[9] Lakshmil V R, Vijayakumar S. Wireless Sensor Network based Alert System for Cold Chain Management. Procedia Engineering, 2012, 38(4):537-543.

[10] Li C M, Nien C C, Liao J L, et al. Development of wireless sensor module and network for temperature monitoring in cold chain logistics, IEEE International Conference on Wireless Information Technology and Systems. IEEE, 2012:1-4.

[11] Ruiz-Garcia L, Barreiro P, Robla J I. Performance of ZigBee-Based wireless sensor nodes for real-time monitoring of fruit logistics. Journal of Food Engineering, 2008, 87(3):405-415.

[12] Zhang Linghan, Yan Xinxin, Yan Jian, et al. Design of WSN Node for Shipborne Dangerous Goods Container Monitoring. Instrument Technology and Sensors, 2014(1): 47-49.

[13] Ruiz-Garcia, Luis, et al. "Testing ZigBee motes for monitoring refrigerated vegetable transportation under real conditions." Sensors 10.5 (2010): 4968-4982.

[14] Badia-Melis, Ricardo, et al. "Refrigerated fruit storage monitoring combining two different wireless sensing technologies: RFID and WSN." Sensors 15.3 (2015): 4781-4795.

[15] Bai, Yong, Yonghui Zhang, and Chong Shen. "Remote container monitoring with wireless networking and Cyber-Physical System." Global Mobile Congress. IEEE, 2010 on-line at: https://ieeexplore.ieee.org/document/5634569?arnumber=5634569

[16] "Si7021-A20 I2C HUMIDITY AND TEMPERATURE SENSOR", https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf

[17] "Adafruit, Adafruit Ultimate GPS Logger Shield", https://cdnlearn.adafruit.com/downloads/pdf/adafruit-ultimate-gps-logger-shield.pdf

[18] DIGI, "XBee/XBee-PRO S2C Zigbee RF Module", https://www.digi.com/resources/documentation/digidocs/pdfs/90002002.pdf

[19] Arduino, "Arduino UNO", [Online]. https://store.arduino.cc/arduino-uno-rev3. [Accessed: 4-Jan2016].

[20] Raspberry Pi, "Raspberry Pi 2 Model B" https://www.raspberrypi.org/products/raspberry-pi-2-model-b.

[21] "Reefer Commodities & Temperatures" http://www.cncargocool.com/images/Reefer_Commodities_v2.pdf