Hyperlocal Monitoring of Air Pollutants at Street-level using a Long-Range Wireless Sensor Network

Amritpal Kaur

A thesis submitted to
Auckland University of Technology
in fulfilment for the degree of
Doctor of Philosophy

November 2022

School of Engineering

Abstract

Air pollution monitoring is a crucial practice aimed at systematically measuring, assessing, and analyzing the air quality we breathe. It involves the collection of data related to the presence and concentration of various pollutants, such as particulate matter, ground-level ozone, carbon monoxide, sulphur dioxide, and nitrogen dioxide, among others, in the atmosphere. The air pollution monitoring is essential due to the far-reaching impact of air pollution on human health, the environment, and overall well-being. It serves several critical purposes such as protecting public health, environmental preservations, regulatory compliance, research and policy development, early warning systems, and public awareness. Air pollution monitoring is a fundamental tool in safeguarding public health, preserving the environment, and facilitating evidence-based policymaking. Its significance continues to grow as urbanization and industrialization contribute to increased pollution levels, making it essential for creating a cleaner, healthier, and more sustainable future. A Wireless Sensor Network pollution monitoring system provides valuable analysis of air pollution monitoring pollution at varying heights at street level. The development of the network uses Long-Range (LoRa), a proprietary low-power wide-area network modulation technique, to measure particular matter, carbon dioxide, temperature, and humidity.

The research was conducted with different scenarios. In the first scenario, the proposed LoRa network was simulated using the ns3 simulator. Before the sensor placement at the street-level, the network simulation is required to know the estimation of the pollutant levels and the network performance. The second scenario was to develop and implement the LoRa network for real-time air pollution monitoring using pollution sensors (particulate matter, carbon dioxide, and temperature and humidity sensor). Sensor placement was decided to be implemented at the car park at Auckland University of Technology, North Shore Campus, Auckland, New Zealand. Sensor units were placed on two poles using a 2x3 matrix with 0.5m of distance. The matrix with sensor units is used to monitor pollution levels at different street elevations. The data collected is transmitted to the server using the LoRa gateway. Results and findings were obtained using the same scenario and future recommendations for Air Quality Indexing and Air Quality Health Indexing for New Zealand. Collected data was validated with a flow meter, temperature and humidity meter, and smart air quality meter.

Air pollution monitoring at street-level is important to localised data for different locations in Auckland city. The research will support the Auckland Council and policymakers in making decisions on the air pollution monitoring systems and the different variations for low-cost air pollution monitoring systems. The research addresses the health impacts of air pollution on urban and rural areas. Particulate matter and other pollutant concentration levels collected in results and findings have a high effect on respiratory and cardiovascular health. It empowers policymakers, researchers, and citizens with valuable data to make informed decisions, implement effective pollution control measures, and work towards creating healthier and sustainable cities.

Table of Contents

Abstract i

Table of Contents iii

List of Figures x

List of Tables xix

Glossary / Abbreviations xx

Attestation of Authorship xxv

Acknowledgements xxvi

Chapter 1 Introduction 1

1.1 Overview of Air Pollution Using Wireless Sensor Networks 1

1.1.1 Different Air Pollutants 5

1.1.2 Pollution Risks and Threats 6

1.2 History of Wireless Sensor Networks 8

1.2.1 Early Research on Sensor Networks 9

1.2.2 Distributed Sensor Networks program at the Defence Advanced Research Projects Agency 9

1.2.3 Sensor Networks in the 1980s and 1990s 10

1.2.4 Sensor Network Research in the 21st century 10

1.3 History of Air Pollution Monitoring 10

1.3.1 Air Pollution History in New Zealand 13

1.4 History of air quality measurement and legislation 14

1.5 Objectives 14

1.6 Significance of Street Level Air Pollution Monitoring 15

1.7 Research Methodology 16

1.8 Structure of Thesis 17

Chapter 2 Air Pollution Monitoring Using Wireless Sensor Networks 19
2.1 Introduction 19
2.2 Literature Review 23
2.2.1 Simulation and Network Deployment 24
2.2.2 Cellular Networks 74
2.2.3 Vehicular Networks 76
2.3 Summary and Contribution 80
Chapter 3 Network Performance of Proposed Air Monitoring Network 83
3.1 Introduction 83
3.2 Comparison Between Simulators 85
3.2.1 Optimised Network Engineering Tool (OPNET) 86
3.2.2 TOSSIM 87
3.2.3 JavaSim (J-Sim) 87
3.2.4 Objective Modular Network Testbed in C++ (OMNET++) 87
3.2.5 Network Simulator 2 (ns-2) 88
3.2.6 Network Simulator 3 (ns-3) 89
3.3 Simulation Process 90
3.4 LoRa Module 94
3.4.1 PeriodicSender 94
3.4.2 LoRaMAC 95
3.4.3 LoRaPhy 95
3.4.4 LoRaChannel 96
3.4.5 LoRaNetDevice 97
3.4.6 Helpers and Tests 97
3.5 Performance Evaluation 98
3.5.1 Variables and Metrics 99

3.6 Results 100
3.6.1 System Model 100
3.6.2 LoRa Sensor Network Setup 101
3.6.3 LoRa Sensor Network Implementation in ns-3 103
3.6.3.1 Physical Layer 103
3.6.3.2 MAC Layer 103
3.6.4 Lora Sensor Network System Model 104
3.6.4.1 Network Throughput 105
3.6.4.2 End-to-End Delay 106
3.6.4.3 Packet Delivery Ratio (PDR) 107
3.7 Compared with Wireless Sensor Networks 109
3.8 Estimated Pollution in the Specified Simulated Area 111
3.9 Summary and Contribution 116
Chapter 4 . Low-Cost Monitoring Wireless Sensor Network Design and Development 119
4.1 Classification of Wireless Communication Protocols 119
4.1.1 SigFox 119
4.1.2 Long-Range Wide Area Network 121
4.1.3 Narrow-Band Internet of Things 122
4.2 Long-Range and Long-Range Wide Area Network 124
4.2.1 Long-Range 125
4.2.2 LoRa- Chirp Spread Spectrum 125
4.2.3 Long-Range Wide Area Network 126
4.2.4 Architecture 126
4.2.5 Classes 127
4.2.5.1 Class A operational mode 127
4.2.5.2 Class B operational mode 128

4.2.5.3 Class C operational mode 129
4.3 The Things Network 130
4.3.1 Network Server 130
4.3.2 Features 130
4.3.3 Architecture 132
4.4 Radio Communication Theory 133
4.4.1 Decibel and Decibel-milliwatt 133
4.4.2 Link budget and Link margin 134
4.4.3 RSSI and SNR 135
4.4.3.1 Received Signal Strength Indicator 135
4.4.3.2 Signal to Noise ratio 136
4.5 Air Pollution Monitoring Scope 136
4.5.1 Star Topology 137
4.5.2 Mesh Topology 138
4.5.3 Clustered Tree Topology 139
4.6 Hardware 140
4.6.1 Gateway 140
4.6.1.1 SparkFun LoRa Channel-1 Gateway 140
4.6.1.2 LPS8 LoRa Gateway 142
4.6.1.3 Hardware system 143
4.6.1.4 Interface 144
4.6.1.5 WIFI Spec: 144
4.6.1.6 LoRa Spec: 144
4.6.1.7 LPS8 connectivity 147
4.6.2 End Nodes 155
4.6.3 Sensors 158

4.6.3.1 CO_2 sensors 158
4.6.3.1.1 Working of CO2 sensor 159
4.6.3.2 DHT22 160
4.6.3.2.1 Working of DHT22 161
4.6.3.3 Dust Sensor 161
4.6.3.3.1 Dust sensor working 162
4.6.3.4 PM Sensor 163
4.6.4 Impact of Hardware 165
4.7 Software 165
4.7.1 The Things Stack 165
4.7.1.1 Integration Workflow 167
4.7.1.2 Payload Formatter 168
4.7.2 ubidots 168
4.7.2.1 Setup Ubidots 168
4.7.2.2 Payload Formatter 171
4.7.2.3 Webhook Integration 172
4.7.2.4 Monitoring Data 173
4.7.3 Impact of Software 174
4.8 Limitations 175
4.8.1 Impact of limitations 176
4.9 Issues 177
4.9.1 Gateway 177
4.9.2 Upgraded with The Things Stack 178
4.10 Experimental Setup 179
4.10.1 Front-end Layer 180
4.10.2 Middleware Layer 181

4.10.3 Back-end layer....181
4.10.4 Gateway Position....182
4.10.4.1 Gateway Location at University....182
4.11 Summary and Contribution....183
Chapter 5 Results and Discussion....186
5.1 Measurement Scenario....186
5.2 Pilot testing stage 1 with Dust sensor....189
5.2.1 Results and Discussion....189
5.2.1.1 Received Signal Strength Indicator (RSSI) of Node 1....192
5.2.1.2 Signal-to-Noise Ratio (SNR) of Node 1....192
5.3 Pilot testing stage 2 with four sensors....193
5.3.1 Results and Discussion....195
5.4 Final testing with Street Elevations....201
5.4.1 Results....205
5.4.1.1 CO_2 concentration levels....205
5.4.1.2 Received Signal Strength Indicator....211
5.5 Summary of Results....212
Chapter 6 Sensor Data Validation....216
6.1 Flow 1 and Flow 2....216
6.1.1 Working of Flow....217
6.1.2 Principles....217
6.1.3 Setting Connections....218
6.2 Plume Lab AQI....222
6.3 Results....223
6.4 Smart Air Quality Monitor....226
6.4.1 Features....227

6.5 Summary and contribution 228
Chapter 7 Conclusion and Future Work 230
7.1 Reviews of Motivations and Objectives 230
7.2 Reviews of Methods 230
7.3 Contributions 231
7.4 Conclusion 232
7.5 Future Work 237
7.5.1 Proposed Mathematical Modelling. 238
7.5.1.1 AQI for $PM_{2.5}$ 240
7.5.1.2 Air Quality Health Indexing (AQHI) 242
Appendix A ns-3 Coding Explanation 246
Appendix B Schematic Diagrams 261
Appendix C Sensors – Technical Information 263
Appendix D Results 292
Appendix E Conferences and Poster/Paper Publications 300
References 336

List of Figures

Figure 1.1 Wireless Sensor Networks....2
Figure 1.2 Air Pollutants Revised....6
Figure 1.3 Sizes of Particulate Matters....7
Figure 2.1 Performance Parameters of WSNs....21
Figure 2.2 Architecture Diagram of WAMPS....24
Figure 2.3 Nodes Deployment Strategy....25
Figure 2.4 AQI for the selected area....25
Figure 2.5 Hardware Architecture....26
Figure 2.6 Smart gas sensor network deployment used in the COTNAM simulation....27
Figure 2.7 Sensor node used for gas monitoring. (a) Block architecture (left) (b) Top and bottom sides (right)....27
Figure 2.8 Triangular, square, and hexagonal grid deployment....28
Figure 2.9 System Architecture....29
Figure 2.10 The correspondence between the selections of SF values vs Distance....29
Figure 2.11 Air quality data models and their associations....30
Figure 2.12 Deployment of stationery monitors in Hong Kong....32
Figure 2.13 Ambient Wireless Sensor....32
Figure 2.14 Wireless Sensor Hardware Architecture....33
Figure 2.15 Sensors in wireless sensor networks....34
Figure 2.16 Data visualization using web applications....34
Figure 2.17 Geographical locations of the EPAV sensors. The horizontal (east-west) size of the domain is approximately 55 km, and the vertical (north-south) dimension is approximately 40km....35
Figure 2.18 LabVIEW front panel results for indoor cases....36
Figure 2.19 Block diagram of monitoring system....37
Figure 2.20 Wireless sensor network model....38
Figure 2.21 5G Edge Architecture....39
Figure 2.22 General architecture of the Pollution-Spots system....40
Figure 2.23 Receptors....41
Figure 2.24 Module NodeMCU architecture....41
Figure 2.25 System Overview....42

Figure 2.26 Mobile Sensing Box 43
Figure 2.27 Personal Sensing Device 43
Figure 2.28 Overall wireless sensor network system architecture for indoor air quality monitoring 44
Figure 2.29 (a) Functional block diagram of the sensor node, (b) Arduino Uno microcontroller board, and (c) Digi XBee module 44
Figure 2.30 Air Quality Monitoring System Architecture 45
Figure 2.31 Block diagram of the RF sensor unit 46
Figure 2.32 High-level system architecture 47
Figure 2.33 Experimental setup for cylinder-based testing 48
Figure 2.34 Block Diagram of the Raspberry Pi module and other devices 48
Figure 2.35 Three-tier architecture of IoT 49
Figure 2.36 Proposed System Architecture 50
Figure 2.37 Methodology for air pollution 51
Figure 2.38 Hardware Architecture 53
Figure 2.39 Fiber-optic sensor 54
Figure 2.40 Block Diagram of Proposed Model 55
Figure 2.41 Components of a CO-WSN unit 56
Figure 2.42 Cloud Server 57
Figure 2.43 Two types of air quality monitoring IoT sensor modules 59
Figure 2.44 System architecture 62
Figure 2.45 Overall designed architecture of large-scale air pollution remote alarm system 63
Figure 2.46 GPRS/SMS Communication Module 64
Figure 2.47 LPWA-based air quality monitoring system 65
Figure 2.48 Deployed network at campus 66
Figure 2.49 System architecture 67
Figure 2.50 Architecture of the monitoring system 68
Figure 2.51 Proposed automatic microscale air quality monitoring system 69
Figure 2.52 Proposed participatory urban sensing framework 70
Figure 2.53 LTE-M-based outdoor network architecture 71
Figure 2.54 Experimental setup 72
Figure 2.55 Block diagram of low-cost environmental monitoring system based on LoRa for WSN 73

Figure 2.56 System Architecture 74
Figure 2.57 HazeWatch System Architecture 74
Figure 2.58 GPRS Unit 75
Figure 2.59 HBase service architecture 76
Figure 2.60 Overview of the proposed system 77
Figure 2.61 System Architecture of MSS sensor nodes 78
Figure 2.62 Global System 79
Figure 2.63 Architecture of WSN-based indoor air quality monitoring system. 79
Figure 3.1 OMNET++ interface 88
Figure 3.2 ns-2 interface 89
Figure 3.3 ns-3 interface 90
Figure 3.4 Structure for Dummy Sensor Nodes 92
Figure 3.5 Creating Functions for Dummy Sensor Nodes 93
Figure 3.6 Transmission Process of Dummy Nodes 94
Figure 3.7 Simulation Model 101
Figure 3.8 Structure of LoRaWAN network 102
Figure 3.9 Network Topology GUI from ns-3 simulator 104
Figure 3.10 LoRa Sensor Network 105
Figure 3.11 Chart showing Network Throughput results. 106
Figure 3.12 End-to-End Delay 107
Figure 3.13 Packet Delivery Ratio 108
Figure 3.14 Network Throughput 109
Figure 3.15 End-to-End Delay 110
Figure 3.16 Packet Delivery Ratio 111
Figure 3.17 Temperature Readings 113
Figure 3.18 Humidity Readings 113
Figure 3.19 CO_2 Readings 114
Figure 3.20 PM_1 Readings 114
Figure 3.21 $PM_{2.5}$ Readings 115
Figure 3.22 PM_4 Readings 115
Figure 3.23 PM_{10} Readings 116
Figure 4.1 Sigfox Architecture 120
Figure 4.2 LoRaWAN Architecture 121
Figure 4.3 NB-IoT System Architecture 123

Figure 4.4 LoRaWAN technology stack 126
Figure 4.5 Top-level LoRaWAN Architecture 127
Figure 4.6 Class A (default) 128
Figure 4.7 Class B (beacon) 129
Figure 4.8 Class C (continuous) 129
Figure 4.9 Worldwide 131
Figure 4.10 Auckland, New Zealand 131
Figure 4.11 The Things Network Architecture 133
Figure 4.12 Components of a basic communication system to calculate the link budget. 134
Figure 4.13 Received Signal Strength Indicator 135
Figure 4.14 Signal to Noise Ratio 136
Figure 4.15 Star Topology 138
Figure 4.16 Mesh Topology 139
Figure 4.17 Tree Topology 140
Figure 4.18 LoRa Sparkfun Channel-1 Gateway 141
Figure 4.19 Architecture of LPS8 LoRa Gateway 143
Figure 4.20 LPS8 System Overview 146
Figure 4.21 LPS8 Applications 146
Figure 4.22 Connecting using Wi-Fi 147
Figure 4.23 Showing Network after First Boot 147
Figure 4.24 Connect using Ethernet with DHCP IP from the router. 148
Figure 4.25 Connect to WiFi with DHCP IP from the router. 148
Figure 4.26 LPS8 Login Interface 149
Figure 4.27 LPS8 LoRa Gateway System Overview 149
Figure 4.28 WAN Client Setting in LPS8 150
Figure 4.29 Successful Network Connection 150
Figure 4.30 LoRaWAN Configuration 151
Figure 4.31 TTN Server Web Browser 151
Figure 4.32 Cluster Selection 152
Figure 4.33 Creating Gateway 152
Figure 4.34 Gateway ID and Server address 153
Figure 4.35 Frequency Plan Selection 153
Figure 4.36 Service Provider Selection 154
Figure 4.37 LoRa Configuration 154

Figure 4.38 LoRa Node 155
Figure 4.39 LoRaWAN Network Diagram..... 156
Figure 4.40 Pin Configuration of LoRa ESP32 TTGO Board..... 157
Figure 4.41 Pin Connectivity of sensors. 157
Figure 4.42 CO_2 sensors 158
Figure 4.43 CO_2 sensor working..... 159
Figure 4.44 DHT22 sensors 160
Figure 4.45 Temperature and Humidity sensor working. 161
Figure 4.46 Dust sensors..... 162
Figure 4.47 Dust Sensor Working 163
Figure 4.48 SPS30 Sensor for measuring Particulate Matter..... 164
Figure 4.49 SPS30 sensor working. 164
Figure 4.50 Things Stack Console. 167
Figure 4.51 Payload matter. 168
Figure 4.52 Demo Dahboard to select plugins. 169
Figure 4.53 Things Stack Plugin..... 169
Figure 4.54 Token Selection 170
Figure 4.55 Endpoint URL 171
Figure 4.56 Payload Decoder..... 172
Figure 4.57 Decoding Function 172
Figure 4.58 Webhook Integration 173
Figure 4.59 Devices on Ubidots 173
Figure 4.60 Different Variables on Device..... 174
Figure 4.61 Network Block Diagram 179
Figure 4.62 Network design and Working..... 180
Figure 4.63 Gateway Position..... 182
Figure 4.64 Gateway Position Map 183
Figure 5.1 Sensing Modules 187
Figure 5.2 LoRa Modules with Sensors..... 187
Figure 5.3 Sensor Placement 188
Figure 5.4 Sensor Placement 188
Figure 5.5 Temperature Readings..... 190
Figure 5.6 Humidity Readings..... 190
Figure 5.7 Dust Readings..... 191

Figure 5.8 CO_2 Readings....191
Figure 5.9 RSSI of Node 1....192
Figure 5.10 SNR of Node 1....193
Figure 5.11 LoRa Sensing Module....194
Figure 5.12 LoRa Sensing Modules with Sensors....194
Figure 5.13 $PM_{2.5}$ Concentration Levels....195
Figure 5.14 PM_{10} Concentration Levels....196
Figure 5.15 PM_1 Concentration Levels....196
Figure 5.16 PM_4 Concentration Levels....197
Figure 5.17 CO_2 Concentration Levels....198
Figure 5.18 Temperature Levels....198
Figure 5.19 Humidity Levels....199
Figure 5.20 Received Signal Strength Indicator....200
Figure 5.21 Signal to Noise Ratio....201
Figure 5.22 Pole 1 with sensor units at different distances....203
Figure 5.23 Pole 2 with sensor units at different distances....204
Figure 5.24 CO_2 concentration level at 0.5m (node 1)....206
Figure 5.25 Temperature at 0.5m (node 1)....207
Figure 5.26 Humidity at 0.5m (node 1)....208
Figure 5.27 $PM_{2.5}$ at 0.5m (node 1)....208
Figure 5.28 PM_{10} at 0.5m (node 1)....209
Figure 5.29 PM_1 at 0.5m (node 1)....210
Figure 5.30 PM_4 at 0.5m (node 1)....211
Figure 5.31 RSSI Node 1....212
Figure 6.1 Flow 1....216
Figure 6.2 Signup with application....219
Figure 6.3 Privacy Policy....219
Figure 6.4 Creating Profie....220
Figure 6.5 Step to Start Flow....220
Figure 6.6 Flow Detected....221
Figure 6.7 Selecting Flow....221
Figure 6.8 Updating Flow.....222
Figure 6.9 Plumes AQI....223
Figure 6.10 PM_1 Concentration Levels....224

Figure 6.11 $PM_{2.5}$ Concentration Levels....224
Figure 6.12 PM_{10} Concentration Levels....225
Figure 6.13 VOC Levels....225
Figure 6.14 NO_2 Concentration Levels....226
Figure 6.15 AQI by Plumes....226
Figure 6.16 Smart Air Quality Monitor....227
Figure 7.1 Network Design....236
Figure 7.2 Proposed Mathematical Modelling....238
Figure A.1 ns3 code for Data Storage....246
Figure A.2 ns3 code for Declaring Input Parameters....247
Figure A.3 ns3 code for Creating Random variables....248
Figure A.4 ns3 code for Transmitting Packets to nodes....248
Figure A.5 ns3 code for Receiving Packets....249
Figure A.6 ns3 code for Spreading factor....249
Figure A.7 ns3 code for Print Statement....250
Figure A.8 ns3 code for Commented code....250
Figure A.9 ns3 code for Main Function Code....251
Figure A.10 ns3 code for creating a single LoRa server....251
Figure A.11 ns3 code for Assigning Position to nodes....251
Figure A.12 ns3 code for the Base position for sensor nodes....252
Figure A.13 ns3 code for Assigning position for dummy sensor nodes....252
Figure A.14 ns3 code for Creating Channel....253
Figure A.15 ns3 code for Offset Beacon....253
Figure A.16 ns3 code for Assigning gateway to nodes....253
Figure A.17 ns3 code for Enable communication process....254
Figure A.18 ns3 code for Monitoring energy....254
Figure A.19 ns3 code for Connection between gateway and server....255
Figure A.20 ns3 code for Addresses for gateway and server....255
Figure A.21 ns3 code for Trace values transmitted....256
Figure A.22 ns3 code for Enable interference....256
Figure A.23 ns3 code for Data transmission between LoRa and dummy nodes....257
Figure A.24 ns3 code for Communication between LoRa and nodes....257
Figure A.25 ns3 code for Gateway to server....258
Figure A.26 ns3 code for Server to LoRa Nodes....258

Figure A.27 ns3 code for Generating NetAnim file. 259
Figure A.28 ns3 code for Simulation Process 260
Figure A.29 ns3 code for Calculation of performance metrics 260
Figure B.1 Schematic Layout of ESP32 Board 261
Figure B.2 Schematic for connector pins 262
Figure C.1 Dust sensors 263
Figure C.2 Circuit Diagram 264
Figure C.3 Different Parameters 264
Figure C.4 Sampling Time Pulse 264
Figure C.5 LoRa Node 265
Figure C.6 SPS30 Sensor for measuring Particulate Matter 269
Figure C.7 SPS30 Working 270
Figure C.8 PM_{10} readings 276
Figure C.9 PM_{1} readings 276
Figure C.10 $PM_{2.5}$ readings 277
Figure C.11 PM_{4} Readings 277
Figure C.12 DHT22 sensors 278
Figure C.13 Pinout and Dimensions of DHT22 279
Figure C.14 Block Diagram of DHT22 280
Figure C.15 Signal transmission 281
Figure C.16 Signal Diagram 281
Figure C.17 Structure Diagram of DHT22 working with The Things Stack 282
Figure C.18 Humidity Readings 282
Figure C.19 Temperature Readings 283
Figure C.20 CO_2 sensors 287
Figure C.21 CO_2 Reading 291
Figure D.1 Pole 1 CO_2 Readings 292
Figure D.2 Pole 2 CO_2 Readings 292
Figure D.3 Pole 1 Temperature Readings 293
Figure D.4 Pole 2 Temperature Readings 293
Figure D.5 Pole 1 Humidity Readings 294
Figure D.6 Pole 2 Humidity Readings 294
Figure D.7 Pole 1 PM_{10} Readings 295
Figure D.8 Pole 2 PM_{10} Readings 295

Figure D.9 Pole 1 PM_1 Readings 296
Figure D.10 Pole 2 PM_1 Readings 296
Figure D.11 Pole 1 $PM_{2.5}$ Readings 297
Figure D.12 Pole 2 $PM_{2.5}$ Readings 297
Figure D.13 Pole 1 PM_4 Readings 298
Figure D.14 Pole 2 PM_4 Readings 298
Figure D.15 Pole 1 RSSI Readings 299
Figure D.16 Pole 2 RSSI Readings 299

List of Tables

Table 1.1 Wireless Sensor Networks....1

Table 1.2 Air Pollution History....12

Table 2.1 Yearly Distribution from 2005 to 2021 of the Articles Published Relevant to Air Pollution Monitoring Using Wireless Sensor Networks....23

Table 3.1 Comparative Analysis of Different Network Simulators....86

Table 3.2 Network Parameters....91

Table 3.3 Dummy Sensor Range....112

Table 4.1 Sigfox features....120

Table 4.2 Comparison between different technologies....124

Table 7.1 Calculate the AQI for $PM_{2.5}$:....241

Table 7.2 Different Categories of AQI....242

Table 7.3 AQHI Calculation for $PM_{2.5}$....244

Table C.1 Specification of the PM sensor....270

Table C.2 Technical specifications....278

Table C.3 Data and Signal format....279

Table C.4 Pinout Configuration....287

Glossary / Abbreviations

AAPRC:	Auckland Air Pollution Research Committee
ABP:	Activation by Personalization
ADC:	Analog-to-digital converter
ADF:	Anomaly Detection Framework
ADR:	Adaptive Data Rate
AERONET:	Aerosol robotic network
AI:	Artificial Intelligence
ANN:	Artificial Neural Network
AOD:	Aerosol optical depth
API:	Application Interface
APPs:	Applications
AppSKey:	Application Key
AQI:	Air Quality Index
AQM:	Air Quality Monitor
AWACS:	Airborne Warning andControl System
AWS IoT:	Amazon Web Services Internet of Things
BPSK:	Binary Phase Shift Keying
BS:	Base Station
CH:	Cluster Head
CLI:	Command-Line Interface
CMOS:	Complementary metal-oxide-semiconductor
CNN:	Conventional Neural Networks
CO:	Carbon Monoxide
COAP:	Constrained application Protocol
CR:	Code Rate
CSN:	Community sensor network
CSS:	Chirp Spread Spectrum
CTMs:	Chemistry Transport models
CTT:	Clustered Tree Topology
D2D:	Device to Device
DARPA:	Defence Advanced Research projects Agency

dB:	Decibel
dBm:	Decibel Milliwatts
DeepCM:	Depth calibration Method
DHCP:	Dynamic Host Configuration Protocol
DOAS:	Differential optical absorption spectroscopy
DSN:	Distributed Sensor Networks
EAQI:	Event-driven Air quality Inspection
ECS:	Emergency Communication system
EDGE:	Efficient data gathering and estimation
Eds:	End-nodes
EPAV:	Environment Protection Authority Victoria
ESP32:	Espressif system
EUI:	End-device Unique identifier
FET:	Field Effect Transistor
GOME:	Global Ozone Monitoring Experiment
GPIO:	General Purpose Input/Output
GPRS:	General Packet radio services
GPS:	Global Positioning System
GSM:	Global system for mobile communication
GUI:	Graphical user interface
GWs:	Gateways
HTTP:	Hypertext Transfer Protocol
I/O:	Input/Output
I2C:	Inter-integrated circuit
IAQD:	Indoor Air Quality Detector
IARC:	International Agency for Research on Cancer
iDEMS:	Indoor Environment Monitoring System
IEEE:	Institute of Electrical and Electronics
ILSCE:	Extended short-term context encoder
INET:	Internet Networking
IoT:	Internet of Things
IP:	Internet Protocol
IPTO:	Information Processing Techniques Office
IR:	Infrared

ISC3:	Industrial Source Complex
ISFET:	Ion-sensitive Field Effect Transistor
ISM:	Industrial, Scientific and Medical
ISVT:	Interpolation Singular Value Thresholding
Jist:	Java Simulator
JSON:	JavaScript Object notation
KNN:	k-nearest neighbours
LCD:	Liquid Crystal Display
LED:	Light Emitting Diode
LFGs:	Landfill Gases
LMIC:	LoRaWAN MAC in C
LN:	LoRa Node
LoRa:	Long-Range
LoRaWAN:	Long Range Wireless Area Networks
LPG:	Liquid Petroleum gas
LPWA:	Low-power vast area network
LPWANs:	Low-Power Wide Area networks
LSM:	Least Square Estimation Method
LSTM:	Long Short-Term memory
LTE-M:	Long-Term Evolution for Machines
MAC:	Media Access Control
MDPI:	Multidisciplinary Digital Publishing Institute
MEMS:	Microelectromechanical system
MIC:	Message Integrity Code
MODIS:	Moderate Resolution Imaging Spectroradiometer aerosol retrieval algorithm
MOPITT:	Measurement of Pollution in the Troposphere
MSB:	Mobile Sensing Box
MSS:	Modular Sensor System
mW:	Milliwatts
NASA:	National Aeronautics and Space Administration
NB-IoT:	Narrow band Internet of Things
NDIR:	Non-dispersive infrared gas detector
NED:	Network Description
NO2:	Nitrogen Dioxide

NOAA:	National Oceanographic and Atmospheric Administration
Ns-2:	Network simulator 2
Ns-3:	Network Simulator 3
NwkSKey:	Network Key
O3:	Ozone
OMI:	Ozone Monitoring Instrument
OMNET++:	Objective Modular Network Testbed in C++
OPNET:	Optimized Network Engineering Tool
OTAA:	Over the Air Activation
OTcL:	Object oriented extension of TCL
PAHs:	Polycyclic aromatic hydrocarbons
PARM:	P-tree based association rule mining
PDR:	Packet Delivery ratio
PEM:	Personal Environment Monitor
PHY:	Physical Layer
PID:	Photoionization Detector
PLR:	Packet Loss Ratio
PM:	Particulate Matters
PPM:	Parts per million
QPSK:	Quadrature Phase Shift Keying
RCQ:	Converging recursive Quartiles
RSSI:	Received Signal Strength Indication
Rx:	Receiver
SCFDMA:	Single carrier frequency Division Multiple Access
SDA:	Send and receive data
SensIT:	Sensor Information Technology
SF:	Spreading Factor
SNR:	Signal to Noise Ratio
SO_2:	Sulphur Dioxide
SOSUS:	Sound Surveillance System
SQL:	Structured Query Language
SSA:	Single Scattering Albedo
SSID:	Service Set Identifier
SSL:	Secure Socket Layer

SSN:	Static sensor network
STEL:	Short Term Exposure List
SUMO:	Simulation of Urban Mobility
SVR:	Support Vector Regression
SVT:	Singular Value Thresholding
SWANS:	Scalable Wireless Adhoc Network Simulator
TCP/IP:	Transmission Control Protocol/Internet Protocol
TNGAPMS:	The next-generation Air Pollution Monitoring System
TOMS:	Total Ozone Mapping Spectrometer
TtL:	Transistor-transistor Logic
TTN:	The Things Network
TW:	Time Windows
TWA:	Time Weighted Average
Tx Power:	Transmitting Power
UART:	Universal asynchronous receiver-transmitter
UDP:	User Datagram Protocol
UNB:	Ultra Narrow Band
USB:	Universal Serial Bus
USI:	Universal Sensor Interface
VARMA:	Vector auto-aggressive moving average
VOC:	Volatile Organic Compounds
VSN:	Vehicle sensor networks
WAPMS:	Wireless Air Pollution Monitoring System
WHO:	World Health Organization
WPAN:	Wireless Personal Area Network
WSNs:	Wireless Sensor Networks

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person except where explicitly defined in the acknowledgements, nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Amritpal Kaur
November 2022

Acknowledgements

First and foremost, I would like to say my deepest appreciation to GOD, for his blessing, strength, patience, and knowledge he has given me throughout this long, challenging, exciting and inspiring PhD journey. This journey wouldn't be possible without extreme guidance and assistance of many individuals who contributed and extended their helpful.

The research wouldn't be possible without professional guidance and support from my supervisors. Their continuous support, encouraging to keep me on track. My greatest gratefulness to my primary supervisor Dr Jeff Kilby. His constant support, valuable mentoring, and motivation helped me to work smoothly through easy and tough times. The weekly progress meetings helped me to keep me focused on my research study. His professional way of guidance and proof reading of my papers and thesis drafts with quick responses helps me to improve and get better during my research.

I also would like to thank my secondary supervisor Dr Hakilo Sabit whose valuable guidance and suggestions helped to improve my research h skills and thesis documentation during different phases of study.

I would like to dedicate this thesis to my parents, Heera Lal, and Rupinder Kaur, who have been always there to support me during my hard times. They always encouraged me to move forward with my disability. Without their encouragement and blessing, this thesis wouldn't be possible.

Especially thanks and acknowledgement to my loving husband Lakhvinder Singh Gill. Without his love, patience and motivation during my research, this thesis would have not been completed. Thank you for being a part of my life and making pleasant PhD journey.

The funding for living expenses during my research was provided by Dr Jeff, from his research fund. I would like to thank you for supporting me with Stipend. I also would like to thank Prof. Peter Chong for his assistance as a mentor and supporting me for the fee's payments during my hard times in PhD Journey.

I would also like to acknowledge the postgraduate office at AUT School of Engineering for their support and assistance during PhD journey with doctoral forms. Specially acknowledge to the technicians- Stephen Hartley and Daniel, for their technical support and assistance.
Finally, thanks to my PhD colleague at AUT; Adnan Ghaffar, who always motivated me to complete this thesis and gave me suggestions whenever I am in seek of help.

Chapter 1
Introduction

The chapter introduces the history of wireless sensor networks (WSNs) and how they are used in air pollution monitoring. It also covers the research objectives and methodology. At the end of the chapter, it covers the structure of this research.

1.1 Overview of Air Pollution Using Wireless Sensor Networks

Wireless Sensor Networks (WSNs) can be defined as self-configured and infrastructural. WSN monitors physical or environmental conditions, such as temperature, sound, vibration, pressure, motion, or pollutants [1-4]. Different attributes of wireless sensor networks are explained in Table 1.1.

Table 1.1 Wireless Sensor Networks

Attribute	Description
Sensors	Active and passive with sparse coverage for small or large, heterogeneous sensors, homogeneous, homogeneous, dense, and sparse range, and fixedand planned deployment.
Sensing entities of interest	Distributed, localized extent, static and dynamic mobility, cooperative and non-cooperative nature.
Operating environment	Benign and adverse.
Communication	Wired and wireless communication
Processing architecture	Centralized, distributed, hybrid.[5]
Energy availability	Constrained and unconstrained.

Sensor nodes transmit their data to the network using the central position of the network, known as the sink node, where the sensor data are analyzed and observed. A sink node acts as an

intermediator between users and the network, where the node receives data or information from the network server and collects data from the sink nodes. Communication among the sensor nodes can be performed using radio signals.

In WSNs, sensor nodes are interconnected by several sensing devices, transceivers, and power components. A single sensor node in a WSN is resource-constrained with high speed, large storage capacity, and network bandwidth. Sensor nodes are responsible for the self-organisation of the network infrastructure using multi-hop communication. The working model of sensor nodes may be either continuous or event-driven. Wireless sensor devices can have actuators to "act" upon certain conditions. Figure 1.1 shows new WSN applications and requires a new protocol design for different constraints. Over the past few decades, sensor networks have advanced in terms of semiconductors, networking, and new emerging technologies to drive diverse deployment technologies for large-scale WSNs. The new WSN technologies have enabled new generations of different applications and were used for deployment from 5 to 10 years ago. Currently, state-of-the-art WSNs have numerous applications in homes, workplaces, and professional lives.

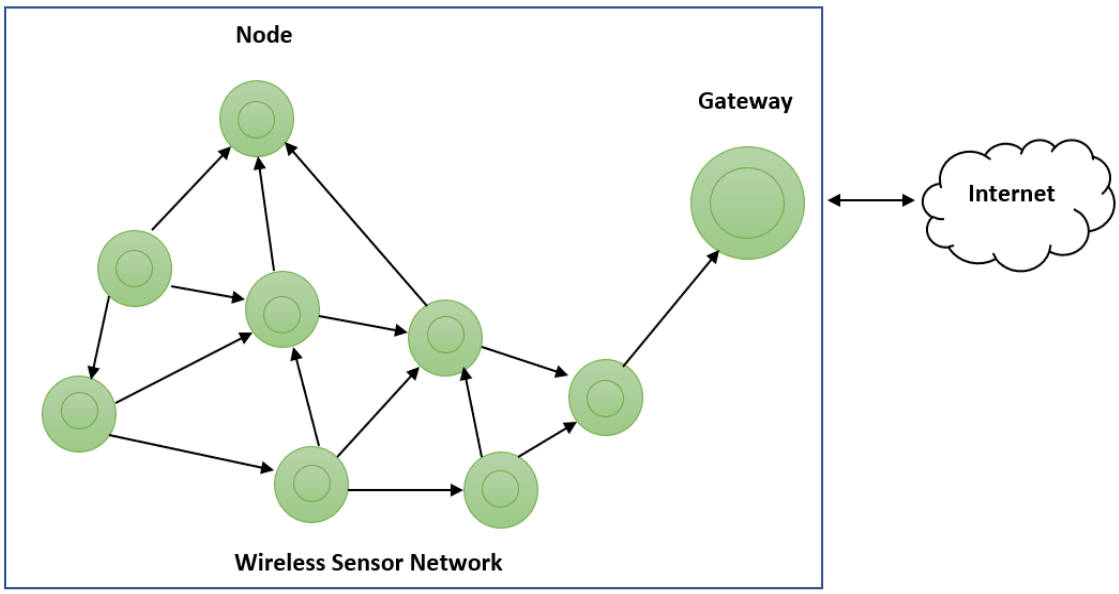


Figure 1.1 Wireless Sensor Networks.

In September 1990, for the 21st century, business weeks were an essential technology [6]. It has low-cost multiple sensor devices using wired or wireless links with Internet connections to

deploy many sensor units, providing opportunities for developing and managing houses, towns, and the environment. In addition, in the networking system, micro-sensors offer new technology systems for developing defence arenas, generating new capabilities for exploration, surveillance, and other tactical applications deployed in water, on-body, indoor, and outdoor locations.

Network sensor units can detect and track threats (e.g., airborne and moving vehicles, personnel, chemical and biological agents). Each sensor node has embedded processing capability and multiple onboard sensors operating in the acoustic, seismic, infrared (IR), magnetic modes, imagers, and micro radars. The sensor units will have storage capacity, wireless links for other nodes, location, and positioning using the global positioning system (GPS) or local positioning algorithms. Interconnected microsensors are appropriate for data collection for multiple distributed sensors and explain the different possible ranges for general sensor networks and include other applications, such as military sensing, physical security, air traffic control, traffic surveillance, video surveillance, industrial and manufacturing automation, distributed robotics, environment monitoring, and building and structure monitoring. The sensors in these applications may be small or large, and the networks may be wired or wireless [5].

For future applications and opportunities, WSNs will have well-established systems using a combination of hardware and software. However, wireless systems and network designers are currently faced with making sense of and understanding the complex trade-offs among many application variables, including deployment costs, hardware and software, system reliability, security, and performance. It needs to work on energy saving, network protection and security, semiconductor design systems, and new sensor technologies to break down the network deployment features. WSNs are considered distributed networks developed by small and lightweight sensor nodes [7]. These sensor nodes are effectively used for measuring different parameters such as particles in air, toxic gases, temperature, pressure, and motions [7].

The development of WSNs was motivated by military applications, such as battle surveillance, disaster management, pollution monitoring, offshore exploration, aircraft control, traffic control, marine environment, and process automation. WSNs have been deployed for air pollution monitoring, including collecting observed data over time across ample space [8]. The sensor data monitoring system receives measured data from the sensor network and provides

helpful information to users by understanding the condition of the remote location. Controlling the sensor network and monitoring air pollution requires a sensor network control system and an air pollution monitoring system.

Rapid urbanization and industrialization have led to the degradation of mental quality parameters. It is essential to monitor the environment and record the diversity of air pollution indices to develop policies and procedures for the public [2]. If traditional methods are used for air pollution monitoring, WSNs will not be feasible in broader areas. In new ways, WSNs are used with low-cost sensors to create opportunities to collect real-time data from different locations and provide a pollution map. Wireless sensors are used for monitoring pollution because of their accuracy, calibration between sensor nodes, and fault-tolerant topology control [2].

A sensor network consists of an array of thousands of sensors of diverse types interconnected by a wireless communication network [1]. A single sensor network may include interconnected subnets of different topologies, such as stars, clusters, meshes, fully connected meshes, clustered trees, and hybrids [1].

They used different applications from the military to the medical field using wireless sensor networks (WSNs) [9]. Advances have facilitated electronic miniaturization, performance growth, wireless technologies, energy efficiency, and protocols [10]. The sensor collects data from the surroundings and provides high performance using advancements in wireless sensor technology and the falling price of wireless sensor network devices. Over the past few decades, WSN protocols have been developed and supported by the Internet of Things (IoT). These communication protocol processes are energy efficient and present the traits of WSNs [11], but WSNs have a limited communication range in urban coverage. Many systems use sensor deployments with limited resources to deliver reliable data with complex applications [10].

Devices operate at relatively low data rates to achieve their goals. Low-power wide-area networks (LPWANs) have attracted interest worldwide. Specifically, the Netherlands deployed an LPWAN [12]. Several technologies compete with LPWAN, such as SigFox, narrowband (IoT), and long-range (LoRa) [13]. LoRa is an LPWAN with low power consumption and a long-range communication network with a low data rate. LPWAN focused on different technologies and was recommended as a fundamental IoT application [14].

Among the other network technologies, long-range (LoRa) or LoRaWAN is one of the most common methods for long-range communication [15]. LoRa is a small wireless network that works based on chirp-spread-spectrum modulation with different spreading factors and bandwidths according to the requirements of the location and network [16]. LoRa uses ISM bands for communication, for example, 433 MHz, 868 MHz, and 915 MHz, or depending on the bands divided into different channels [17] and can operate with a non-licensed band for long-range communication, less than 1 GHz [18].

1.1.1 Different Air Pollutants

To track the effects of air pollution on the environment and health, it is necessary to monitor the pollution level in urban and suburban areas [19]. Primary particles are emitted directly from sources, whereas secondary particles are formed in the atmosphere from gaseous emissions missions [19]. Pollutants can be solid particles, liquid droplets, or gases, as shown in Figure 1.2. In air pollution, common pollutants draw intense concerns, including particulate matter (PM) of different sizes, such as $PM_{2.5}$, PM_4, PM_1, and PM_{10}. Particulate matter is a highly complex mixture of particles and toxic gases. Particulate matter is known as atmospheric aerosol. PMs are composed of solid and liquid particles of different sizes that affect the environment and human health owing to their chemical effects. Particulate matter consists of primary and secondary elements [19]. Primary elements are directly released from sources to the environment, and secondary elements are formed in the environment, including chemical reactions. There are two types of particulate matter, with more extensive and smaller particles. These can be considered coarse (PM_{10}) and fine ($PM_{2.5}$) particles. Toxic gases such as ozone (O_3), carbon monoxide (CO), sulphur dioxide (SO_2), and nitrogen oxide (NO_2), significant sources of air pollution, are road traffic emissions that emit 97 per cent of carbon monoxide (CO) and 75 per cent of nitrogen oxide (NO). Volatile organic compounds (VOC) and polycyclic aromatic hydrocarbons (PAHs) [19]. VOCs are emitted as gases by solids and liquids. VOCs include various chemicals, some of which can cause short- and long-term adverse health effects [20].

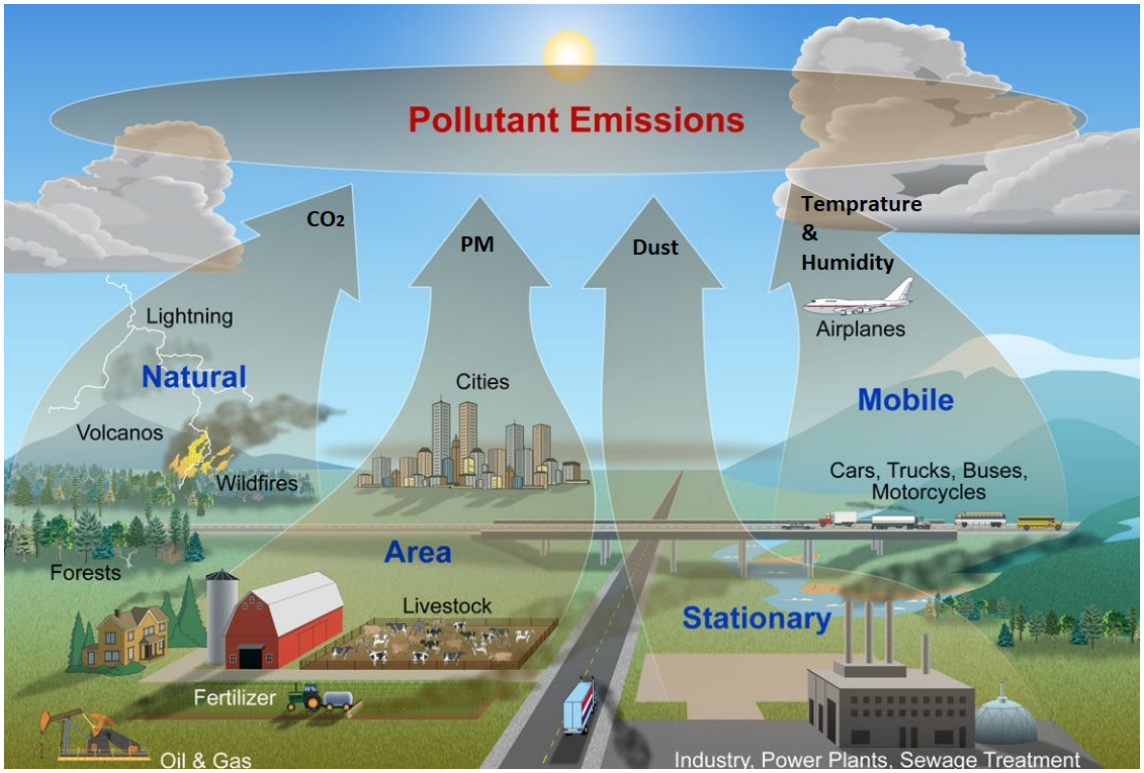


Figure 1.2 Air Pollutants Revised [21].

1.1.2 Pollution Risks and Threats

Particulate matter (PM) consists of solid and liquid particles that are sufficiently small to inhale when suspended in the air. PM values fluctuate significantly in structure and chemical composition depending on the source of the material. PM is derived from human activities, and natural resources are classified into different sizes. PM_{10} has a diameter of 10 micrometres (μm) or less, and PM2.5 has a diameter of 2.5 micrometres (μm), a subset of smaller particles within the PM_{10} range shown in Figure 1.3.

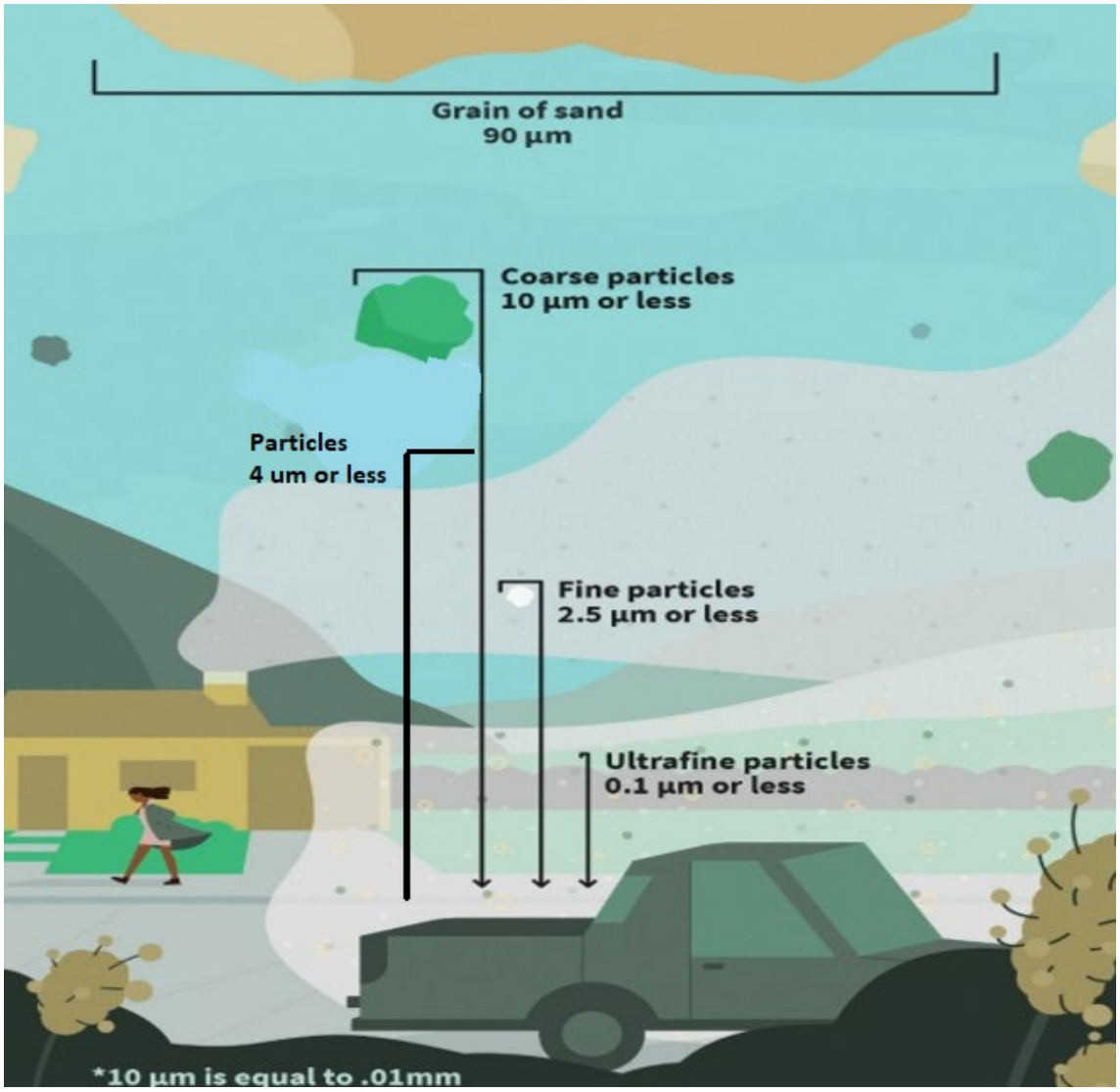


Figure 1.3 Sizes of Particulate Matters [22]

PM_{10} and $PM_{2.5}$ are causing health issues such as respiratory system and lung disease. PM_{10} contains large particles and is less harmful than $PM_{2.5}$. $PM_{2.5}$ travels deeply in the human body and causes severe problems such as cancer [23]. The PM sensors can measure with different methods such as filter-based gravimetric and attenuation and optical techniques such as filter-based gravimetric and attenuation optical.

There are some low-cost sensors for monitoring $PM_{2.5}$ and PM_{10}. Their typical cost is approximately Dollar 10-100. PMS3003 was developed by Plantower, which is the third-generation evolution of sensor PMS1003. Its monitoring range was 0.3 to 1.0, 1.0 to 2.5, 2.5 to 10 (mm), and its response time was less than 10 seconds. Some sensors, Sharp

GP2Y1010AU0F, Samyog DSM501A, Shinyei PPD60PV, and Shinyei PPD42NS, cannot differentiate between the particle sizes and typically report the concentration of $PM_{2.5}$ and PM_{10} particles with dimensions greater than 0.5 μm. PMS5003 and PMS7003 are the fifth and seventh generation of sensors developed by the Plantower Company. HK-A5 is the digital particle density sensor for monitoring the concentration of $PM_{2.5}$ and PM_{10} within the 0.3-10 microns range.

1.2 History of Wireless Sensor Networks

To understand the trade-offs in today's WSNs, it is helpful to examine their history [24]. Like many other new advanced technologies, WSNs have their origins in military and heavy industrial settings, which are very different from the light industrial and consumer WSN applications that are used often today. The Sound Surveillance System (SOSUS), created by the United States Military in the 1950s to find and track Soviet submarines, was the first wireless network to closely resemble a modern WSN. This network used hydrophones and submerged acoustic sensors dispersed over the Atlantic and Pacific oceans. Even though it now only serves to monitor underwater fauna and volcanic activity more peacefully, this sensing device is still in use. Governments and universities eventually began using WSNs in air quality monitoring, forest fire detection, natural disaster prevention, weather stations, and structural monitoring. Although there was a high demand for WSNs in the market, expanding beyond these constrained applications proved difficult. Previous research, technology, and heavy industry applications have relied on large, expensive sensors and exclusive networking protocols. These WSNs prioritize performance and functionality over other aspects, such as hardware and deployment costs, networking standards, power consumption, and scalability. The high cost and low-volume combination have hindered the widespread acceptance and deployment of WSNs in broader applications.

Technologies from the three research fields of sensing, communication, and computation are required for creating sensor networks (including hardware, software, and algorithms). Therefore, both individual and combined developments have promoted sensor network research. Radar networks used in air traffic control are examples of early sensor networks. A national power grid can be viewed as an extensive sensor network [24]. These systems were

developed with specialized computers and communication capabilities before “sensor networks" emerged.

1.2.1 Early Research on Sensor Networks

Defense applications, like many other technologies, have fuelled sensor-network research and development. To identify and track covert Soviet submarine surveillance during the Cold War, the SOSUS system of underwater hydrophones was established at key points [5]. The National Oceanographic and Atmospheric Administration (NOAA) now uses SOSUS to monitor oceanic occurrences such as seismic activity and animal behaviour. Aerostats such as sensors and Airborne Warning and Control System (AWACS) aircraft are now a part of this air defence system [5].

These sensor networks generally adopt a hierarchical processing structure, where processing occurs at consecutive levels until the information about events of interest reaches the user. Although the research focused on satisfying mission needs, for example, acoustic signal processing and interpretation, tracking, and fusion, [5] it provided some key processing technologies for modern sensor networks.

1.2.2 Distributed Sensor Networks program at the Defence Advanced Research Projects Agency

Modern research on sensor networks began in 1980 with the Distributed Sensor Networks (DSN) program at the Defence Advanced Research Projects Agency (DARPA) [25]. R. Kahn served as the Information Processing Techniques Office's director and is credited with co-inventing the TCP/IP protocols and being a key figure in creating the Internet (IPTO). In 1978, a distributed sensor network workshop identified the technological elements of a DSN, including sensors, communication, processing methods, algorithms, and distributed software. Researchers at Pitt's Burgh and Pennsylvania's Carnegie Mellon University (CMU) concentrated on developing a networking architecture that enables flexible and transparent access [25]. They developed a communication-oriented system called Accent, the primitives of which support transparent networking, system reconfiguration, and rebinding.

1.2.3 Sensor Networks in the 1980s and 1990s

Although the technology for small sensors was not completely developed, early researchers on sensor networks considered a vast number of tiny sensors [25]. Platforms "own" weapons in platform-centric warfare, and those weapons "own" sensors in a rigid architecture. In other words, sensors and weaponry are mounted on and controlled by distinct autonomous platforms. Sensors are not always associated with platforms or weaponry [25]. Instead, they work together via a communications network, relaying information to the proper "shooters." Through multiple observations, geometric and phenomenological diversity, an expanded detection range, and a quicker response time, sensor networks can enhance detection and tracking performance. Utilizing commercial network technologies and common network interfaces also helps reduce development costs [25].

1.2.4 Sensor Network Research in the 21st century

The direction of sensor network research has significantly changed because of recent developments in computers and communication, and it is now getting closer to realizing its original aim. Wireless networking, low-power CPUs, and small, low-cost sensors based on micro-electromechanical system (MEMS) technology enable wireless ad hoc networks for various applications [26]. The DARPA's recently ended Sensor Information Technology (SensIT) program pursued two crucial directions for research and development. It began with the creation of fresh networking strategies. These sensor nodes must be prepared for ad hoc deployment [26]. Today's networking methods are based on a fixed infrastructure created for voice and data. The second push involves networked information processing, obtaining pertinent, accurate, and timely data from the installed network [26].

1.3 History of Air Pollution Monitoring

Since approximately 400 BC, air pollution has been acknowledged as a concern for both human health and ecosystems. Following two millennia, several nations made written records available until measurements began in the eighteenth century [27]. This demonstrates the poor air quality in populated areas and near factories. The history of air pollution caused by human

activity was the main topic of this section, which also identified the key problems, their causes, and regional and worldwide trends. The previous 150 years were the only ones with high-quality measurements of air contaminants, and the last 40 years were the only ones with numerical modelling.

The history explains the time between the seventeenth and nineteenth centuries when substances found in the air were first identified and direct measurements were made [27]. Robert Angus undertook multi-site examinations in the UK to determine the measures of multi-pollutant analyses of the chemical climatology of the contaminated air. Distributed locations to measure atmospheric composition later emerged in the mid-twentieth century, and acid rain research and conclusions shifted focus to the late 1960s. Furthermore, measurements by Europe and North America to track particular types of pollutants have concentrated on local pollution issues in the industrial sectors [27].

Regional and international air chemistry requirements were added to ground-based air pollution monitoring networks in 2000. The third source of time-series data to access the chronology of air pollution is chemistry transport models (CTMs) [27]. The most recent source of pollution data is satellite remote sensing technology, developed over the past three decades and provides information on significant pollutant concentrations worldwide, including SO_2, NO_2, NH_3, CO, and O_3. The related sources of data on air pollution give an overview of the concerns and measures of the development of air pollution in the late nineteenth and early twentieth centuries [27]. The selection of different informational resources is given in Table 1.2 [27].

Table 1.2 Air Pollution History

Date	Air Pollution Event
400 BCE[1]	The relationship between air and health developed as an essential Air, waters and places attributed to Hippocrates.
First Century AD	Writers from imperial Rome, e.g., Seneca and Frontinus, refer to the probable health impacts of smoke.
947-1279	Smoke and gaseous pollutants from coal burning were identified as problems in Central Asia by Al-Masud (947) and China during the Song Dynasty (960–1279).
1273	The Smoke Abatement Act, the earliest legislation in England, prohibits using coal as it is 'prejudicial to health'.
1610	The Law of Nuisance (UK):William Aldred's pig farm case.
1661	John Evelyn published Fumifugium or The Inconvenience of the Aer and Smoak of London.
Seventeenth-century	Harmful effects of air are ascribed to various components, e.g. Kenelme Digby (acids), Nehemiah Grew (lead), John Evelyn (sulfur) and John Hall (antimony or mercury).
Eighteenth-century	Guillaume François Rouelle detects SO_2 by absorbing the gas in solid alkalis; Carl Wilhelm Scheele detects NH_3 via acid absorption.
1872	Robert Angus Smith publishes Air and Rain: The Beginnings of Chemical Climatology, having undertaken the first multi-site, multi-pollutant measurements.
1905	Smoke Nuisance Acts in Bengal.
1956	The UK Clean Air Act.
1960	Extensive local ecological damage by smelters.
1972	United Nations Stockholm Conference confirms acid rain as an essential international European issue.
1977	The USA has established its National Acid Deposition Program (NADP).
1979	UNECE Convention on Long-Range Transport of Air Pollution (LRTAP) was established.

[1] Before the Common Era

Table 1.2 (continued)

Date	Air Pollution Event
1985	Helsinki Protocol: Commitment to reduce SO_2 emissions by 30% (The 30% club).
The 1980s–1990s	Eutrophication of ecosystems by nitrogen deposition recognized
1993	The 'Six Cities' research in North America focuses on the human health effects of air pollution $PM_{2.5}$.
1995	Launch of the first satellite for passive remote sensing atmospheric composition (GOME) for global ozone monitoring.
1999	The UNECE Gothenburg Protocol was adopted to tackle multipollutant multi-effects (acidity, ozone and eutrophication).
The 2000s	Emissions of SO_2 and NO_x in Asia increasingly dominate global emissions and adverse effects.
2010	There is widespread evidence of recovery from the effects of acid deposition in Europe and North America with the decline in emissions of SO_2 and NO_x.
2012	Beijing smog, 13th January, with concentrations of PM and SO_2 like London 1952.
2015	Global SO_2 emissions reduced by 15% from the 1990 peak, while all other air pollutants increased.
2018	Peak global NOx emission? Global emissions of NH_3 and VOC continue to rise.
2020 & COVID-19	The global pandemic dramatically reduces industrial- and transport-related emissions of SO_2, NOx, VOC, and primary PM.

1.3.1 Air Pollution History in New Zealand

This section focuses on the history of air pollution monitoring in New Zealand. This section covers common air pollutants and the methods used for their measurements [28]. Air pollution monitoring in New Zealand before 1968 was reviewed by Sparrow [28]. This review was the first published report of the last century using measurements of dissolved solids in rainwater covering 1884-88 and 1907-1909. In the 1950s, similar work was carried out on issues in

Auckland and Christchurch. Both cities were set up for air quality studies. In Christchurch, winter air pollution has been monitored owing to high smoke and sulphur dioxide from domestic fires and other sources [28]. The Dominion Laboratory observed the first air pollution issue in 1954, and Wilkinson published the results.

In Auckland, the initial concern for air pollution was the emission of fumes from Manukau Mudflats and problems with local industry [28]. The dangerous situation in Auckland was addressed in 1955, and measurements of pollution levels were taken in 1956 and 1956. The Auckland Air Pollution Research Committee carried out this work (AAPRC), and Sparrow published reports. Since then, several methods and techniques have been used to measure pollution levels in Auckland and New Zealand [28].

1.4 History of air quality measurement and legislation

The first legislative controls for air pollution monitoring in New Zealand were introduced in 1956, allowing for managing specific air pollutants and provisions for industrial smoke and odour. This approach was effective for air pollution control for industrial pollution but did not include any sources, such as domestic fires or motor vehicles [28]. These deficiencies were addressed in a report by the Board of Health published in 1970. This report includes new recommendations for the Clean Air Act passed in 1972. The Department of Health administered the Clean Air Act of 1972.

1.5 Objectives

This research focuses on the development and implementation of hyperlocal air pollution monitoring via a LoRa Wireless Sensor Network (WSN) at the street level, measuring several pollution factors such as particulate matter ($PM_{2.5}$, PM_4, PM_1, and PM_{10}), carbon dioxide, temperature, and humidity. The Auckland Council is currently conducting limited studies to monitor ($PM_{2.5}$).

The sole purpose of this research was to find a method to improve pollution monitoring and data collection. Different pollutants in the air are monitored using low-power, long-range WSN, such as a network system proposed using the LoRa network. This research focuses on monitoring pollution under covered walkways in the city, where there might be higher pollutant levels because of different pollutants.

The main research objectives were as follows:

1. How, can the estimated pollutant values in a specified area be checked using an ns-3 simulation?
2. How do air pollutants vary at different street-level elevations and the sensor placement optimization? Does monitoring CO_2 and Particulate Matter Size ($PM_{2.5}$) at the street level enhance existing air pollution monitoring in New Zealand?

1.6 Significance of Street Level Air Pollution Monitoring

The significance of street-level air pollution monitoring using LoRa networks lies in its ability to provide real-time and granular data on air quality in urban areas. Here are some key points highlighting its importance:

- Localized Data: Street-level air pollution monitoring allows for the collection of data at specific locations within a Auckland city. This localized data helps identify pollution hotspots, enabling targeted interventions and policies to address air quality issues in areas with high pollution levels to Auckland Council.
- High Resolution Monitoring: LoRa networks can support a large number of low-cost sensors, enabling high-resolution monitoring of air pollution. Auckland Council and policymakers can access detailed data at different points in a city, providing a more accurate representation of air quality variations.
- Public Health Impact: Street-level air pollution monitoring is essential for understanding the potential health impacts of air pollution on residents living and working in urban areas. Fine particulate matter ($PM_{2.5}$) and other pollutants can have adverse effects on respiratory and cardiovascular health, and timely data helps inform

the public about pollution levels.

- Urban Planning and Traffic Management: Real-time air quality data can aid urban planners and traffic management authorities in designing better transportation systems and traffic flow strategies to reduce emissions and mitigate pollution. It enables data-driven decision-making for sustainable urban development.
- Citizen Engagement: LoRa networks can facilitate citizen engagement in environmental monitoring. Local residents, schools, and community groups can deploy low-cost sensors and actively participate in data collection, raising awareness about air quality issues and promoting community-led initiatives for cleaner air.
- Data-Driven Research: Researchers can use street-level air pollution data to study air quality trends, correlations with health outcomes, and the effectiveness of pollution control measures. This data-driven research contributes to a better understanding of the complex relationship between air pollution and human health.
- Scalability and Cost-Effectiveness: LoRa networks offer a scalable and cost-effective solution for deploying air quality monitoring systems across a city. The low-power and long-range capabilities of LoRa technology make it feasible to deploy a dense network of sensors without excessive infrastructure costs.

In conclusion, street-level air pollution monitoring using LoRa networks plays a vital role in addressing air quality challenges in urban areas. It empowers policymakers, researchers, and citizens with valuable data to make informed decisions, implement effective pollution control measures, and work towards creating healthier and sustainable cities.

1.7 Research Methodology

The proposed WSN monitors the development of hyperlocal air pollution monitoring at street level down St. Paul Street in Auckland for ($PM_{2.5}$) plus other pollutants. A proprietary radio communication technology called LoRa (Long Range), which uses license-free frequency bands, will be used for network development. LoRa enables low-rate information exchange over long distances with low power consumption. Low power usage and long-distance data

transmission are the two benefits of using LoRa nodes and gateways. The sensors, wirelessly connected directly to the LoRa nodes to a LoRa gateway, measure particulate matter, carbon dioxide, temperature, and humidity. The data gathered from the network gateway are presented and saved to create new cloud-based apps on a "The Things Network" server, a free public cloud. Hopefully, the findings of this research will demonstrate the necessity of monitoring the pollution levels of the chosen pollutants, such as particulate matter ($PM_{2.5}$), at the street level in Auckland and deploying a simulated network, which is the same scenario. Four LoRa modules coupled to ($PM_{2.5}$), temperature, humidity, and dust sensors connected to a LoRa gateway communicate with the TTN server and transmit the network data. LoRa Gateway sends packets and receives packets, packet size, packet drop, packet loss, total delay, delay per packet, and throughput.

1.8 Structure of Thesis

This thesis is organized into seven chapters.

Chapter 1 describes the background of the sensor networks and pollution monitoring. It states the issues caused by pollution and makes people aware of their health effects. The last section of this chapter covers the research methodology used for pollution monitoring and research contributions to simulation and network deployment.

Chapter 2 presents a state-of-the-art literature review on available air pollution monitoring methods using sensor networks and different communication techniques for data transmission over network techniques. This chapter focuses on the history, several WSN protocols, methods used for pollution monitoring, and forecasting algorithms used in pollution monitoring. The end of this chapter summarizes the research gap, and the rest of the thesis is based on that gap.

Chapter 3 presents a simulation of the proposed LoRa network to monitor its performance in terms of packet delivery ratio, network throughput, and total delay during transmission in both networks. This section is based on the estimation process of deploying the sensor nodes to understand network efficiency better. This chapter compares simulators, simulation setup, and simulation process and explains how the LoRa modules work in the ns-3 simulation setup. The

last part of this section describes the results obtained from the LoRa network setup. The end of this chapter summarizes the simulation process for both networks and explains their efficiency.

Chapter 4 focuses on the network deployment techniques used in network deployment in Chapter 6. This chapter is based on LoRa and LoRaWAN communication techniques for long-range deployments. This chapter also compares communication theories like Sigfox, LoRa, and NB-IoT. This section also explains the server used for data storage as a cloud, The Things Network, but later works on The Things Stack. The last section of this chapter summarizes the different technologies and the LoRa network.

Chapter 5 presents the deployment of the proposed network. First, it begins with an air pollution monitoring scope using three topologies. The network is based on a tree network topology. Also the hardware and software used in the network deployment based on their features is covered. This chapter also discusses the issues faced during deployment and covers a section on the experimental setup. This chapter also focuses on the results and discussion of the three different scenarios. The last section of this chapter summarizes the results obtained from the three scenarios.

Chapter 6 is proposed based on the validation of the data obtained in Chapter 5. Flow by Plumes Labs monitored the PM levels in the air for data validation. This section explains the flow with its principles and connection settings. Plumes Labs has its own Air Quality Indexing (AQI) framework covered in this section. The results and discussion from the Flow are also covered in this chapter. The Last section of this chapter summarizes these findings.

Chapter 7 the last chapter, describes the conclusion and future work based on the research and findings presented in the above chapters. The following section concludes based on the results, and the last section proposes the mathematical modelling approach that can be implemented in future work to be performed.

Chapter 2
Air Pollution Monitoring Using Wireless Sensor Networks

This chapter reviews the literature on air pollution monitoring using different networks and technologies and aims to find the gap between air pollution methodologies to conduct further research.

Air pollution is a significant issue worldwide, causing several million deaths annually. Monitoring and awareness of air pollution, its effects using existing technologies, and how different researchers have worked on air pollution monitoring are required.

2.1 Introduction

Over the last few decades, air pollution monitoring has attracted considerable attention. Wireless sensor networks use various pollution sensors in a typical air pollution monitoring system to sense air quality parameters such as sampling, collection, and communication. Sensor nodes connect to a gateway that relies on the cloud storage. This chapter focuses on pollution monitoring methods that additional researchers have used to determine air concentration levels, such as other particles and toxic gases.

Air pollution is the introduction of biological, chemical, particle, or hazardous gas pollutants into the atmosphere, which can harm individuals by causing discomfort, disease, or even death, as well as harm other living things such as food crops or the ecosystem [29]. Small, lightweight wireless sensor nodes constitute the distributed networks in WSNs [30]. Sensor nodes were placed in various locations to track environmental factors such as particles, poisonous gases, temperature, pressure, humidity, sound, objects' properties, and motion [30].

According to a World Health Organization (WHO) report, one of the top ten risk factors for air pollution is the estimated amount of indoor smoke from solid fuels. Intense fuel smoke is a risk factor for cataracts, TB, asthma, allergies, and medically unexplained illnesses such as building syndrome, cancer, and unfavourable pregnancy outcomes. Based on ample evidence of carcinogenicity in humans and test animals and solid mechanistic evidence, the International

Agency for Research on Cancer (IARC) Working Group classified outdoor air pollution and particulate matter from outdoor air pollution as harmful to humans.

Monitoring air pollution is crucial for ecosystems, animals, and people. A person inhales 14,000 litres of air daily, and airborne pollutants harm people's health [31]. Researchers exposed to particulate matter have higher morbidity and death rates from lung and cardiorespiratory diseases. Little consideration has been given to the potential involvement of contaminants produced from motor vehicle exhausts in inducing allergic disorders despite the adverse impacts of urban air pollution on the respiratory system.

The development of WSNs was motivated by military applications, including battle surveillance, disaster management, pollution monitoring, offshore exploration, aviation control, traffic control, maritime environment, and process automation, which served as the impetus for their development. WSNs were placed to track data over time and across a large area to monitor air pollution.

By knowing the state of the remote location, the monitoring system can provide users with useful information by gathering data from a sensor network. A sensor network control system and an air pollution monitoring system are necessary for the air pollution monitoring system. The degradation of air quality parameters is caused by rapid urbanization and industrialization [2]. Maintaining a record of several air pollutants requires public policies and procedures. With technological advancements, low-cost sensors have been used in WSNs for real-time data collection and pollution maps. Owing to various aspects such as accuracy, calibration between the sensor nodes, and fault tolerance topology control [1], WSNs are used for pollution monitoring. The sensor network consists of thousands of interconnected sensors connected via a wireless communication network.

A network may hierarchically consist of several interconnected subnets with different topologies [32]. The performance of WSNs is shown in Figure 2.1.

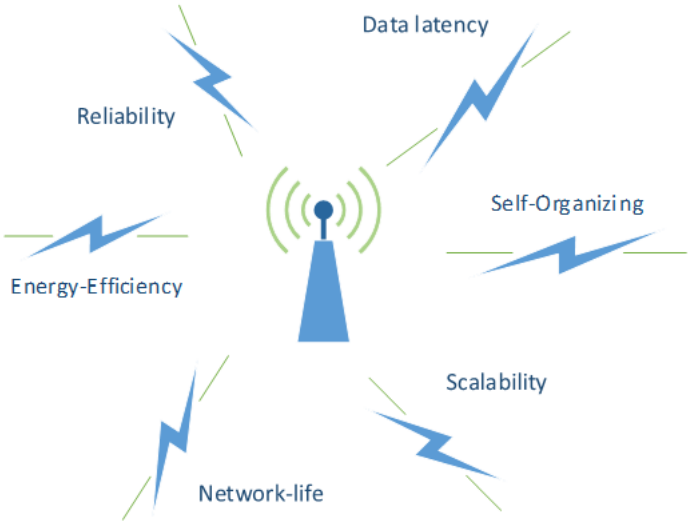


Figure 2.1 Performance Parameters of WSNs [32].

Reliability

This analysis is essential for the successful operation of WSNs. These three data delivery scopes are referred to as infrastructure communication. Users can express interest in a single sensor node, send requests to groups of sensor nodes in a particular area, distribute those requests to all other sensor nodes, or express interest in the entire network and send the message to all sensor nodes [33]. The accuracy of a sensor network depends on a realistic network estimation.

Energy Efficiency

Due to its constrained energy resources, energy efficiency is the primary constraint and performance criterion for a WSN. The actions of the sensor nodes are unsupervised and inaccessible. Traditional networks strive to provide services with a high calibre. To increase the lifetime of a WSN, hardware and software must be designed and operated with an awareness of energy consumption.

Network Life

The battery capacity of the node has long been a factor in the evaluation metric *"lifetime of the network"*. The network lifetime is a crucial issue in WSN research. Several definitions of network lifetime are available for various scenarios and protocols, although many energy-efficient protocols have been proposed to extend the network's life. The sensor network lifetime was defined as the time to the first sensor node failure.

Scalability

Depending on the application, the number of sensor nodes deployed in a sensor field can range from a few to several hundred or thousands. The deployment of sensor nodes can vary in density from sparse to extremely dense nodes. The effectiveness of a WSN with accuracy, a sizeable aggregate amount of energy, and the potential for a better-connected network would all result from a dense network. By designating the cluster head (CH) to oversee the local neighbourhood of the sensor nodes, clustered networks increase the scalability of the mesh networks.

Self-organising

Self-organisation increases network longevity and should consider energy costs and speed. There are three phases: defect localization, identification, and recovery. Mesh networks require much more time for self-configuration than other networks do. The star network requires minimal time for self-configuration, but in the clustered hierarchical network, the CH is responsible for initiating localized reconfiguration.

Data latency

When there is data latency, WSNs are predicted to have low data rates over a long period; however, they may experience burst traffic during certain occurrences. Star networks have the lowest data latency because there is no delay during the buffering stage of routers. However, as network density increases, star networks become less scalable and suffer more significant damage from collisions. Mesh networks have lower data loss but higher latency than star networks.

2.2 Literature Review

A systematic literature review was conducted to explore air pollution monitoring using WSNs by identifying focused research papers, such as (a) IEEE sensor journals, (b) IEEE IoT Things, (c) IEEE articles, (d) IEEE Access, and (e) IEEE Conference Proceedings. They used the following key terms: air pollution, smog, toxic gases, particulate matter, air quality, PM_{10}, and $PM_{2.5}$, from 2005 to 2021. After screening, 100 papers were found, with 70 articles relevant to this research, as shown in Table 2.1.

The literature review was designed as follows:

- Simulation and Network Deployment
- Cellular Networks
- Vehicular Networks

Table 2.1 Yearly Distribution from 2005 to 2021 of the Articles Published Relevant to Air Pollution Monitoring Using Wireless Sensor Networks

Journal name	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	Total
IEEE Journal	0	0	0	1	0	1	0	4	1	2	4	3	1	1	1	0	0	19
IEEE Article	1	1	1	0	0	1	0	2	1	0	1	1	2	1	1	0	0	13
IEEE Access	0	0	0	0	0	1	0	0	0	0	0	4	1	0	0	3	2	11
IEEE IoT Things	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
IEEE Transaction	1	0	0	0	0	0	0	0	0	2	0	0	0	1	0	0	0	4
Conference Proceedings	1	0	1	0	0	0	0	0	2	0	0	2	1	3	2	2	8	22
Total	3	1	2	1	0	3	0	6	4	4	5	10	5	6	4	5	11	70

2.2.1 Simulation and Network Deployment

Khedo et al. [34] proposed the wireless sensor network air pollution monitoring system (WAPMS), shown in Figure 2.2, deployed in Mauritius with many sensor units. Converging recursive quartiles (RCQ), a data aggregation technique is used in this system. The Java in Simulation Time (Jist)/scalable wireless ad hoc network simulator was used to model the WAPMS network system (swans). Jist Simulator transforms an existing virtual machine into a simulation platform by incorporating simulation time semantics at the byte-code level. A scalable wireless network simulator called SWANS was constructed on the Jist Framework.

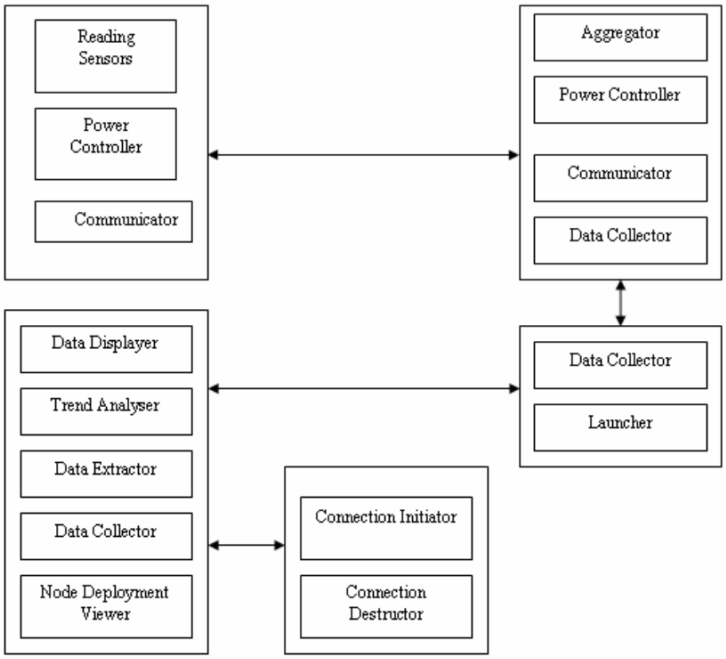


Figure 2.2 Architecture Diagram of WAMPS [34].

The network was prototyped in a small area, as shown in Figure 2.3, and expanded to cover the entire network. With only one sink node, the network was simulated and simplified by employing the gateway.

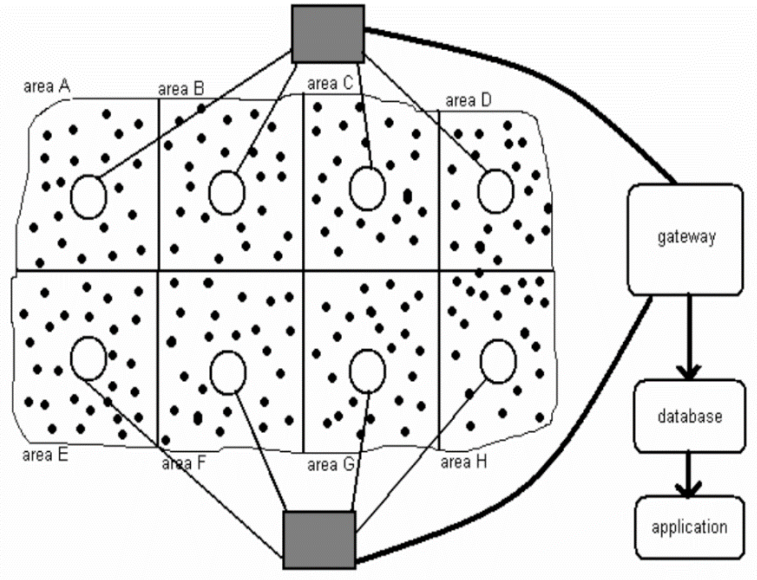


Figure 2.3 Nodes Deployment Strategy [34].

Figure 2.4 shows the air quality indexing (AQI) values used by the WAPMS network system to measure air pollutants that negatively impact human health and the environment. The researcher was motivated to develop an indexing system to categorize air pollution.

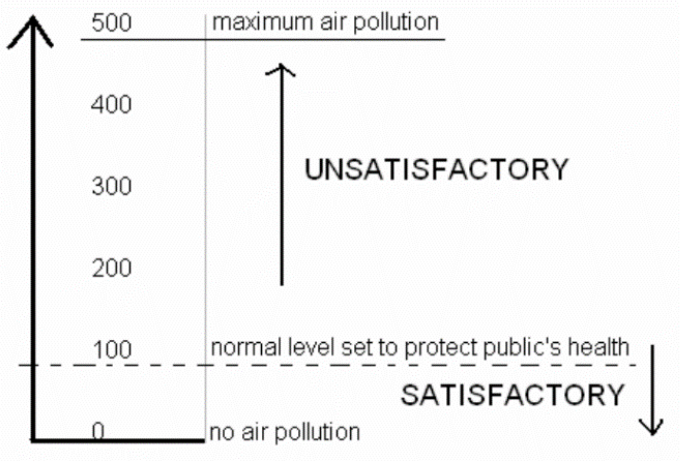


Figure 2.4 AQI for the selected area [34].

Liu et al. [35] proposed an urban air quality monitoring system, as shown in Figure 2.5, and incorporated the GSM hardware architecture and global system for mobile communication (GSM). The proposed method comprises sensor nodes, a gateway, and a control centre.

LabVIEW software was used to simulate the proposed network. In Taipei City, a network design has been implemented to track the CO concentration caused by vehicle emissions. This allows for providing high-resolution meteorological data and attention to air pollutants, such as precipitation, wind direction, speed, temperature, humidity, and CO concentration. The CO concentration was monitored using a Mics-5525 sensor. Using the WSN technology, the proposed network system is appropriate for microscale real-time air quality monitoring. The CO concentration was monitored using a Mics-5525 sensor [35].

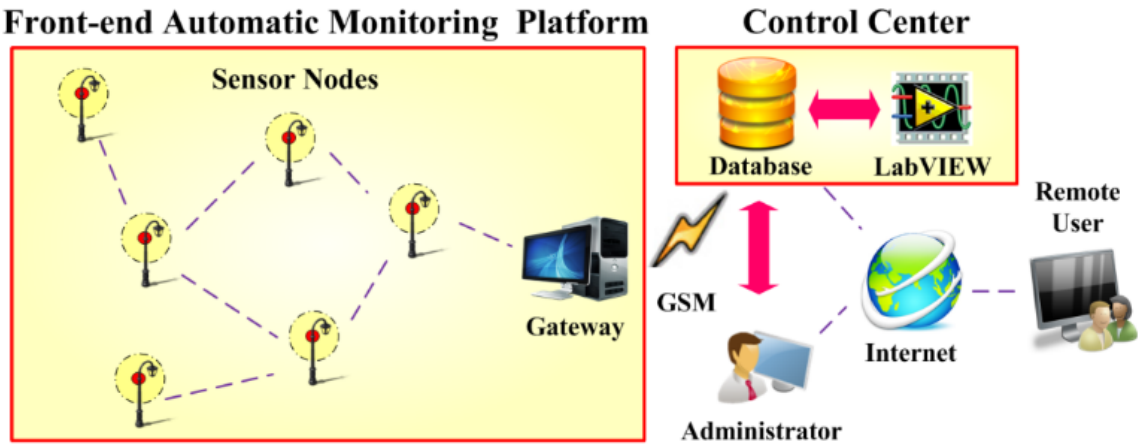


Figure 2.5 Hardware Architecture [35].

Kang et al. [36] developed a remote sensing system to monitor the emission of on-road vehicles using a practical location strategy. The main goal of this study is to formulate a novel challenge in which the bare minimum subset of roads needed to track traffic emissions is identified. This challenge was solved by turning it into a graph-theoretic problem and considering various factors such as traffic laws and restrictions. This depth technique was initially used to create hypergraph-based directed circuits. Later, an approximation method was proposed to quantify the affecting elements, including location and traffic congestion conditions.

Jelicic et al. [37] designed the wireless sensor network architecture for indoor pollution monitoring to resolve issues such as power consumption during network data transmission. The simulation of gas diffusion in COTNAM is shown in Figure 2.6. The sets of sample rates can be selected based on the number of people in the room owing to simulated gas flow diffusion.

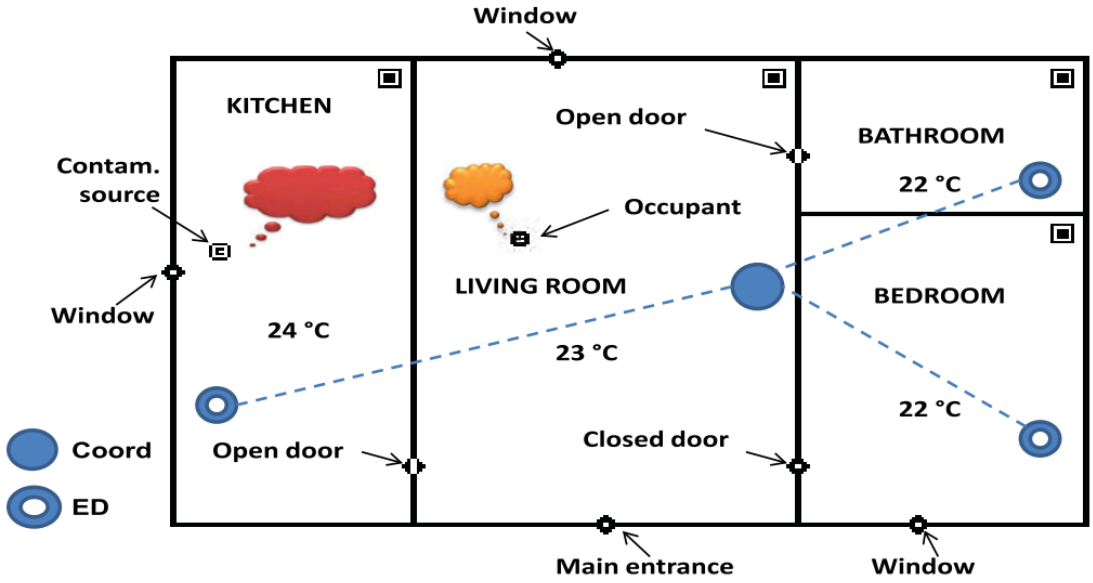


Figure 2.6 Smart gas sensor network deployment used in the COTNAM simulation [37].

At the first building level, 36 nodes were included in the network design. Temperature, humidity, CO, accelerometers for volatile organic compounds (VOCs), and particulate matter were monitored using various sensor nodes, as shown in Figure 2.7. To ensure the power consumption of the sensor node, the authors lowered the activity of the gas sensor.

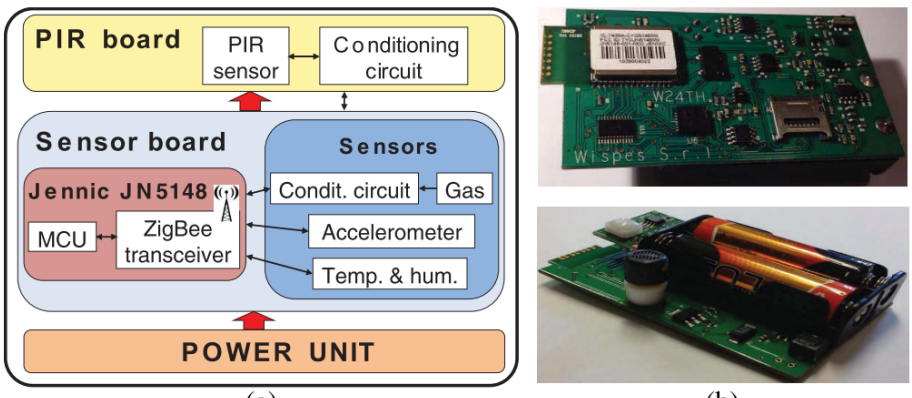


Figure 2.7 Sensor node used for gas monitoring. (a) Block architecture (left) (b) Top and bottom sides (right) [37].

Agarwal et al. [38], this network is based on real-time monitoring and consists of a few sensor nodes placed in a designated area to detect CO_2, CO, and SO_2. There is no network traffic when data are exchanged across the network. Sensor nodes can consume energy in a balanced manner. The proposed network deploys sensor nodes in a deterministic way. A cluster-tree topology was used to minimize the transmission time and reduce the likelihood of collisions. A cluster-tree topology was implemented to avoid collisions and minimize transmission delays. The simulation was performed in MATLAB, where nodes were deployed in 12×14 dimensions in triangular, square, and hexagonal grids, as shown in Figure 2.8. The designed network system works effectively with the cluster head selection algorithm to increase the network lifetime and consumes less energy for the pollution monitoring system.

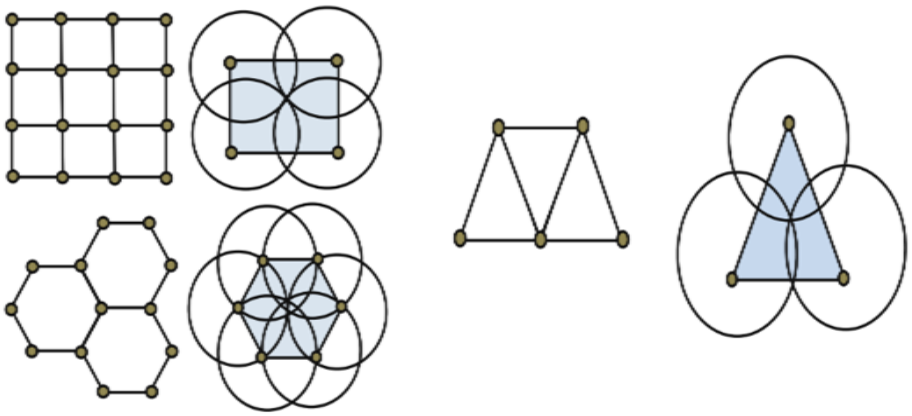


Figure 2.8 Triangular, square, and hexagonal grid deployment [38].

Rahim et al. [39] proposed an alternative data collection method based on the LoRa and LoRaWAN protocols to obtain data on air pollutants. The planned network was simulated at various circular locations with a radius of 6 km using an ns-3 simulator. The network performance can be increased by adding more devices while maintaining the network performance, owing to the system architecture of the LoRa and LoRaWAN protocols, as shown in Figure 2.9. The designed network used a Class A stationary battery to power the devices. A Class A fixed battery was employed in the network design to power the devices. The number of sub-regions of the region of interest and the length of the gateway sleep were used to divide the network into fixed time windows (TW) of 600s during the monitoring period.

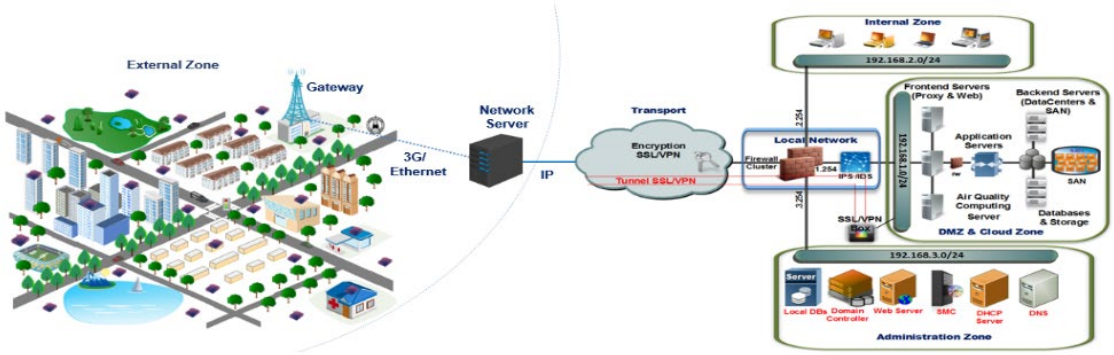


Figure 2.9 System Architecture [39]

According to the results, the worst effects were obtained for an area of 3000m, where all the end devices used the same spreading factor of 7, which does not favour receivers implementing the chirp spread spectrum modulation shown in Figure 2.10.

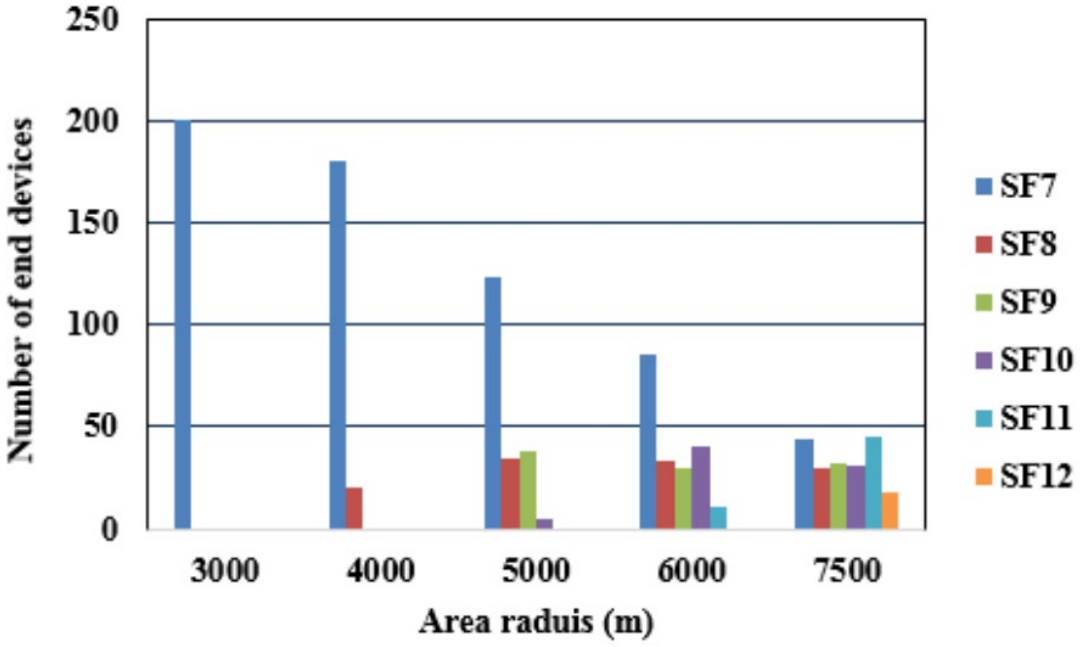


Figure 2.10 The correspondence between the selections of SF values vs Distance [39]

The data transmitted using the LoRa technology were saved to the cloud using a central monitoring unit. Range tests in an outdoor area showed that LoRa can reach approximately 2 km. The transmitting power (TX) is only about 110mA, lower than other wireless technologies.

LoRa technology has the advantages of long-range communication and low power consumption compared to other technologies. The experimental results show that our system can accurately monitor and measure gas and PM concentrations.

Shakkov et al. [40] developed an air pollution monitoring system by installing sensors on vehicles. The designed network uses a complementary approach with comprehensive calibration models to achieve accuracy for low-cost sensors and long-term sensor deployment. To resolve the problems of optimization and performance analysis of air pollution detection systems, a pseudo-random number generation algorithm was used. The implemented method is not limited to air pollution sensors and can be used in other applications. The research results will enable the optimization of various air pollution networks.

Simon et al. developed the real-time event-driven air quality inspection framework (EAQI) to assess air quality specifically for scalability, customizability, and data analysis methods. The designed framework provides public access to air quality information and data models, as shown in Figure 2.11. The framework is based on a distribution sensor network for the data collection and analysis of pollution measurements. Pollution measurements are based on an event stream that stores raw data as an event and provides a data history. The designed framework has a reusable template for developing new applications that require design scalability and an inherently data-centric nature.

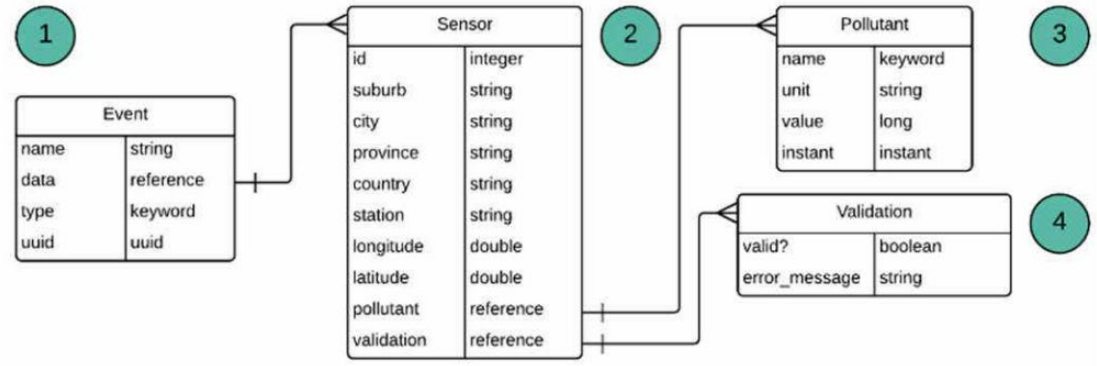


Figure 2.11 Air quality data models and their associations [41].

The network framework's fault tolerance, data validation, and responsiveness were examined using a web-based application. The test findings were encouraging, with 69% of the data

requests completed in 800ms. The robustness of the application was tested, and after 500 concurrent users, request timeouts began to occur. Future work for this project will include:

1. Improvements to the framework to handle meteorological elements and data sources, Processing plugins.
2. Moving beyond real-time air quality data collection, including automatic fault identification and predictive models.

Karapistoli et al. [42] investigate the real-life environmental monitoring application based on wireless sensor networks to prove the actual network deployment, and lab testing is an essential phenomenon for researchers. This research categorizes monitoring agricultural, air, and water pollution and destruction phenomena. This study explored different topologies; however, the focus was on the mesh topology to organize clusters to reduce power consumption.

Yi et al. [43] explored the three types of networks, including static sensor networks (SSN), community sensor networks (CSN), and vehicle sensor networks, were examined. Traditional air pollution monitoring techniques have been used to reduce the effects of air pollution on human health, environment, and economy. The drawbacks of conventional technologies include size, weight, and astronomical costs. These result in the monitoring stations needing to be deployed more sparsely, as seen in Figure 2.12. Researchers have developed the next-generation Air Pollution Monitoring System (TNGAPMS) to overcome these limitations. They still need to address a few problems and difficulties with their current technology, such as the inability to obtain 3D data, the impossibility of active monitoring, and uncontrolled or partially managed carriers.

Figure 2.12 Deployment of stationery monitors in Hong Kong [43].

Folea et al. [44] proposed a compact battery-powered system to monitor carbon dioxide level, temperature, relative humidity, absolute pressure, and intensity of indoor spaces based on wireless infrastructure IEEE 802.11 b/g. The designed network used the ZigBee technology combined with Wi-Fi connections and ambient sensors, as shown in Figure 2.13, and the design's power consumption was tested.

Figure 2.13 Ambient Wireless Sensor [44].

The hardware architecture of the network, in which all sensors are connected to collect sensor data, is shown in Figure 2.14.

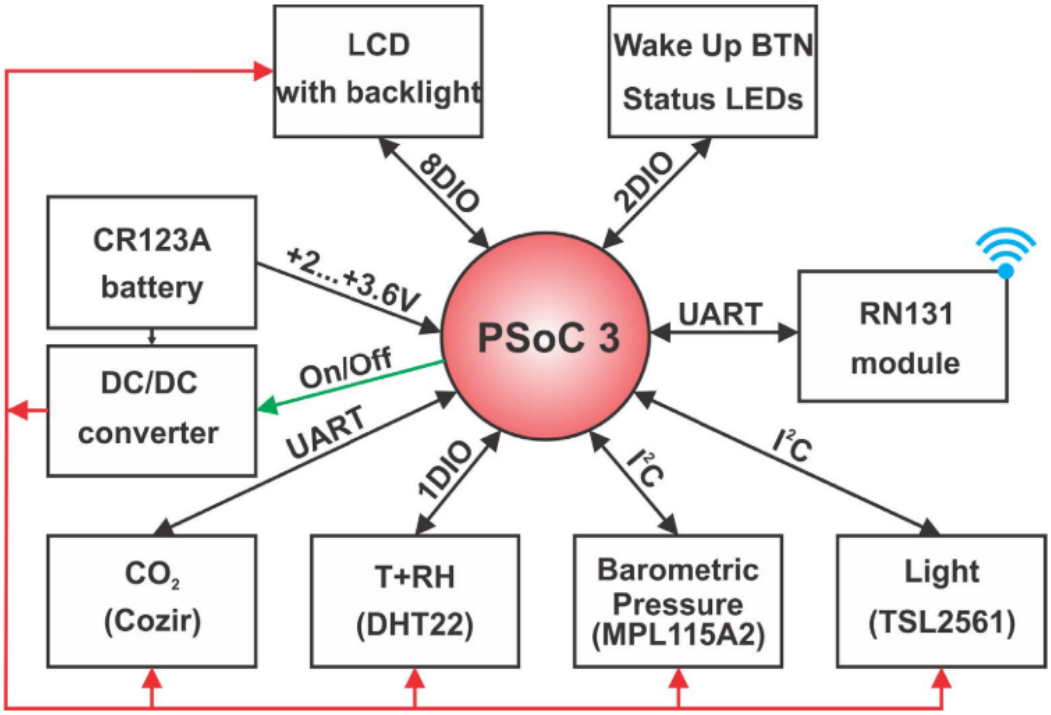


Figure 2.14 Wireless Sensor Hardware Architecture [44].

The network uses the user datagram (UDP) protocol to send data. The Wi-Fi module automatically connects with a particular access point and functions as a pipe for serial data transmitted via the UDP. To meet the requirements of WSNs, namely, low cost, low power consumption, multifunctionality, small dimensions, and wireless communication capabilities, a wireless sensor node is presented in this study and shown in Figure 2.15.

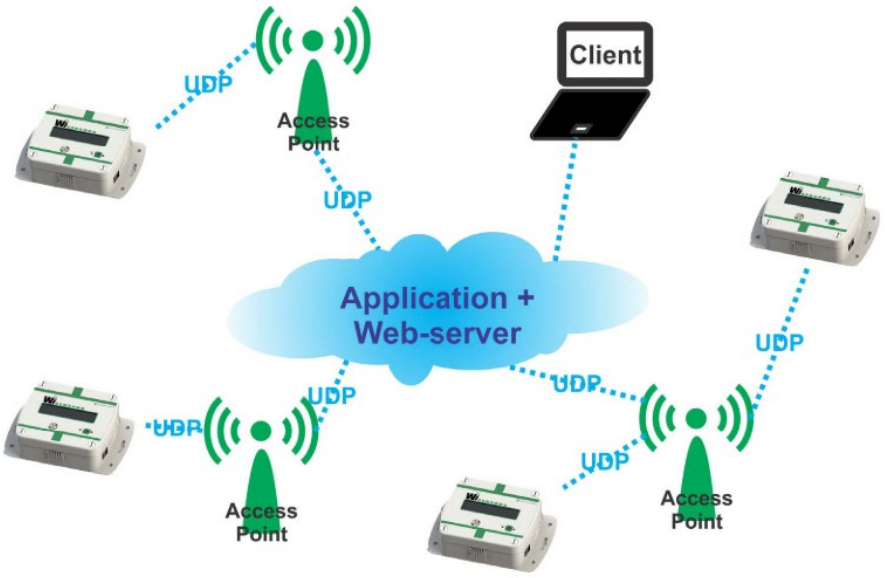


Figure 2.15 Sensors in wireless sensor networks [44].

Figure 2.16 shows the web application utilized for data visualization and the front panel and block diagram of the program created in LabVIEW.

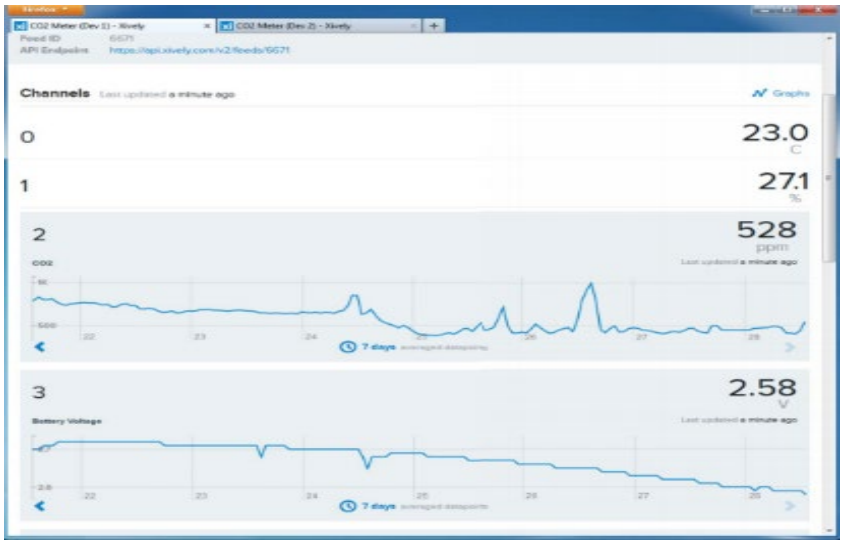


Figure 2.16 Data visualization using web applications [44].

Rajasegara et al. [45] proposed an improved high-precision system for estimating PM concentration. These analyses showed that increasing the concentration of low-precision

sensor nodes led to increased PM estimation accuracy. The sensor stations used to verify the proposed model with simulated data at the regularized rectangular grid nodes are shown in Figure 2.17. This demonstrates that a time series can be used to safeguard the environment. The analysis of this research shows that using low-precision and inexpensive sensors can increase the estimation accuracy in the monitored area. The purpose of this study's future deployment in the Victorian region was to place WSNs with PM10 sensors. The author proposed an improved high-precision system to estimate PM concentration.

Figure 2.17 Geographical locations of the EPAV sensors. The horizontal (east-west) size of the domain is approximately 55 km, and the vertical (north-south) dimension is approximately 40km [45].

Alassi et al. [46] proposed a comprehensive wireless solution in Qatar City (two indoor and one outdoor location) to monitor the concentration levels of CO2, NO2, and CH4, as well as other environmental parameters such as temperature, humidity, solar irradiance, wind direction, and wind speed. Data are collected wirelessly with sensor nodes using the IEEE 802.15.4 protocol sent signals by the WSN-3202 nodes. The NI-97,92 gateway was directly connected to a computer, where the National Instruments LabView 2012 was installed as part of the programming and GUI design software. Nodes receive the data and convert it into helpful information, as shown in Figure 2.18. The developed system functioned as appropriate. Nonetheless, a crucial issue requiring further research is the sensor calibration and the provision and mixing of gases.

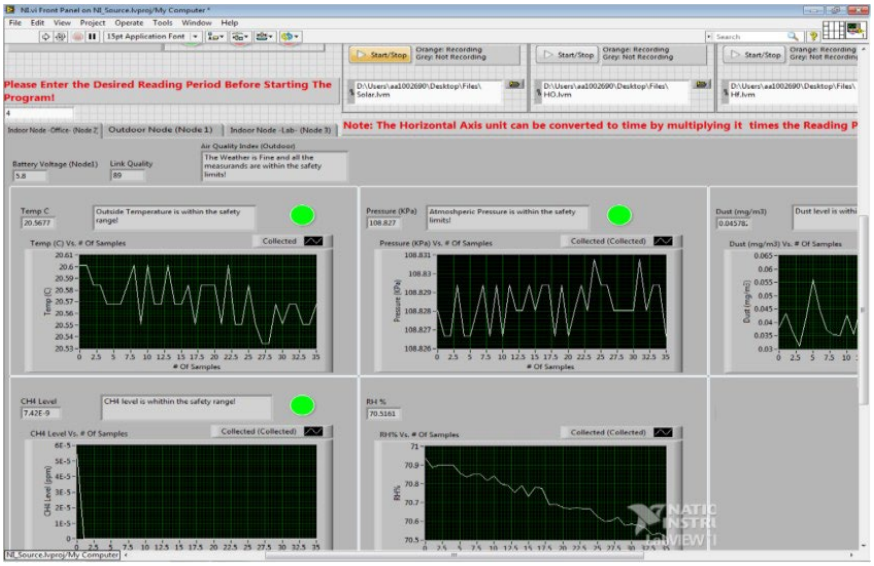


Figure 2.18 LabVIEW front panel results for indoor cases [46].

Shubhajit et al. [47] developed an air pollution monitoring system based on an Arduino microcontroller using sensors to measure several atmospheric parameters such as carbon dioxide, temperature, humidity, and methane. PPM metrics, or parts per million, were used to measure the pollutants. An LCD/serial display was used to collect sensor data and analyze the findings. This network system consists of hardware and software, and each sensor programming interface with Arduino requires Arduino IDE software. A block diagram of the planned air pollution monitoring system is shown in Figure 2.19. The intended network system sensors collected data from the surrounding environment. It is possible to take necessary steps to eliminate pollution.

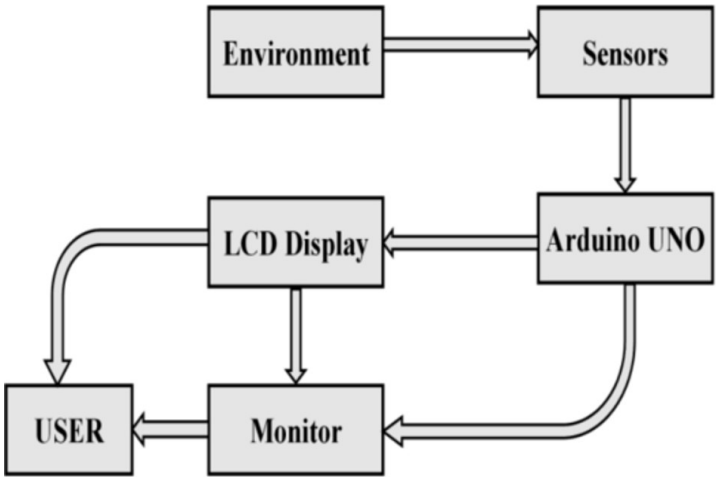


Figure 2.19 Block diagram of monitoring system [47].

Girish et al. [48] proposed an indoor air quality monitoring network using ZigBee wireless technology, shown in Figure 2.20. The proposed system focuses on monitoring CO_2, temperature, and relative humidity. The sensor data are transmitted to a coordinator node through a ZigBee wireless channel. The coordinator node connects to a PC using universal asynchronous receiver-transmitter (UART) connectivity. It receives sensor data from several sensor nodes registered and tracked through a GUI created with Java NetBeans. The combined controller and receiver of the sensor node, an SOC, result in a smaller and less expensive sensor node.

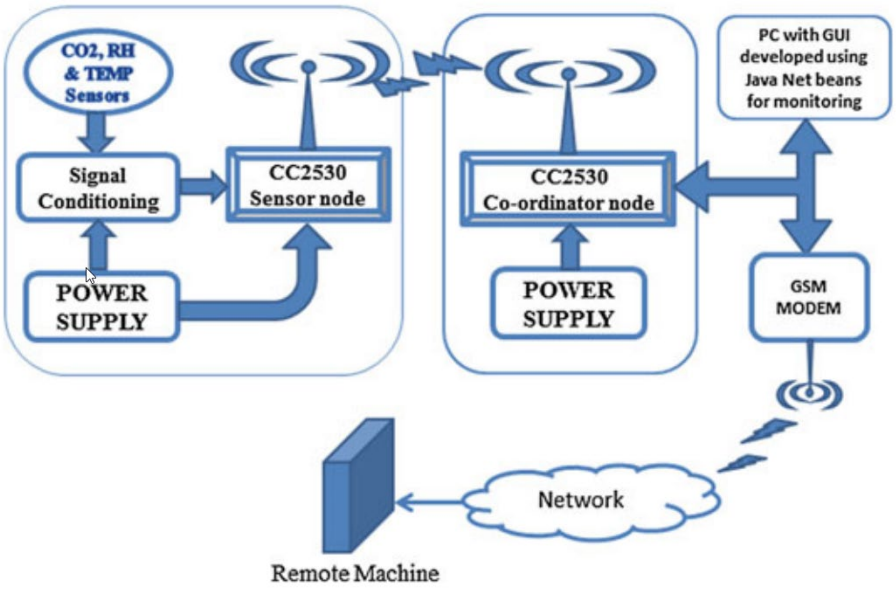


Figure 2.20 Wireless sensor network model [48].

Su et al. [49] conceptualised an Artificial intelligence (AI) powered air pollution monitoring system. In Figure 2.21, they showed two additional systems for low-cost calibration of air quality sensors and image processing of photographs taken by hyperspectral cameras to identify better air quality. The authors created and implemented various AI algorithms on a 5G edge testbed and carried out in-depth analytics on the effectiveness of the AI algorithms and necessary communication resources. Deploy many low-cost sensors and work on sensor calibration for high accuracy of reference stations using different AI techniques.

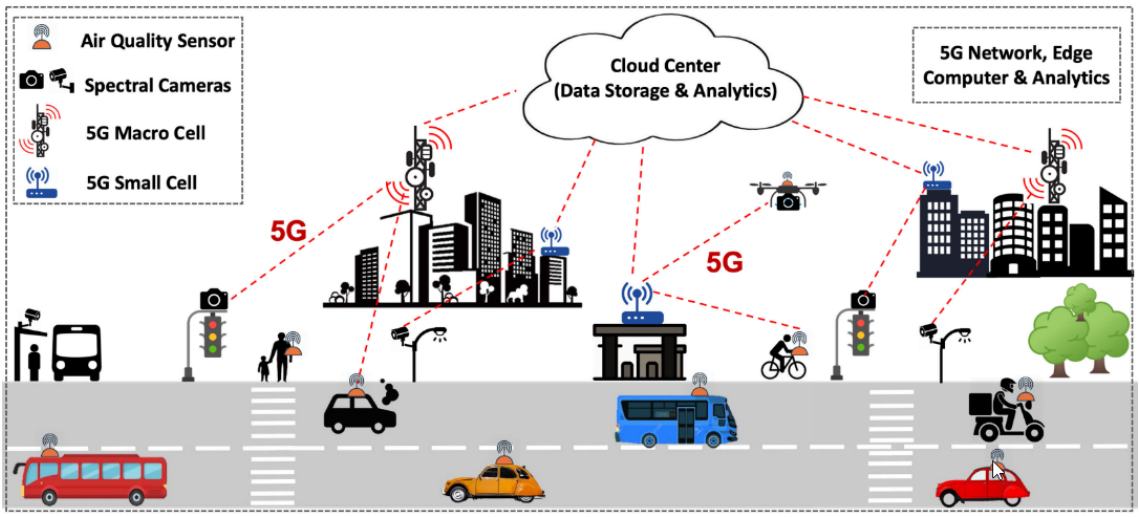


Figure 2.21 5G Edge Architecture [49].

The authors covered three areas: scalable air pollution monitoring architecture, low-cost sensor calibration and hyperspectral image processing on 5G, and communication and computation resources on edge servers. To monitor air pollution, this research must examine the runtime calibration of hierarchical sensors and a method for real-time image processing and video data. Further research is necessary to develop more advanced AI algorithms for deployment, sensor calibration, transfer learning, and data analyses. This requires local knowledge of edge servers.

Mendez et al. [50] presented a solution for an air pollution monitoring system based on a participatory sensing approach; the network structure is shown in Figure 2.22. The intended network system measures the environmental concentrations of CO, CO_2, VOCs, H_2, temperature, and humidity. Smartphones are connected through Bluetooth to transmit data to the server while collecting it. The necessary gamification strategies generate incentives for users.

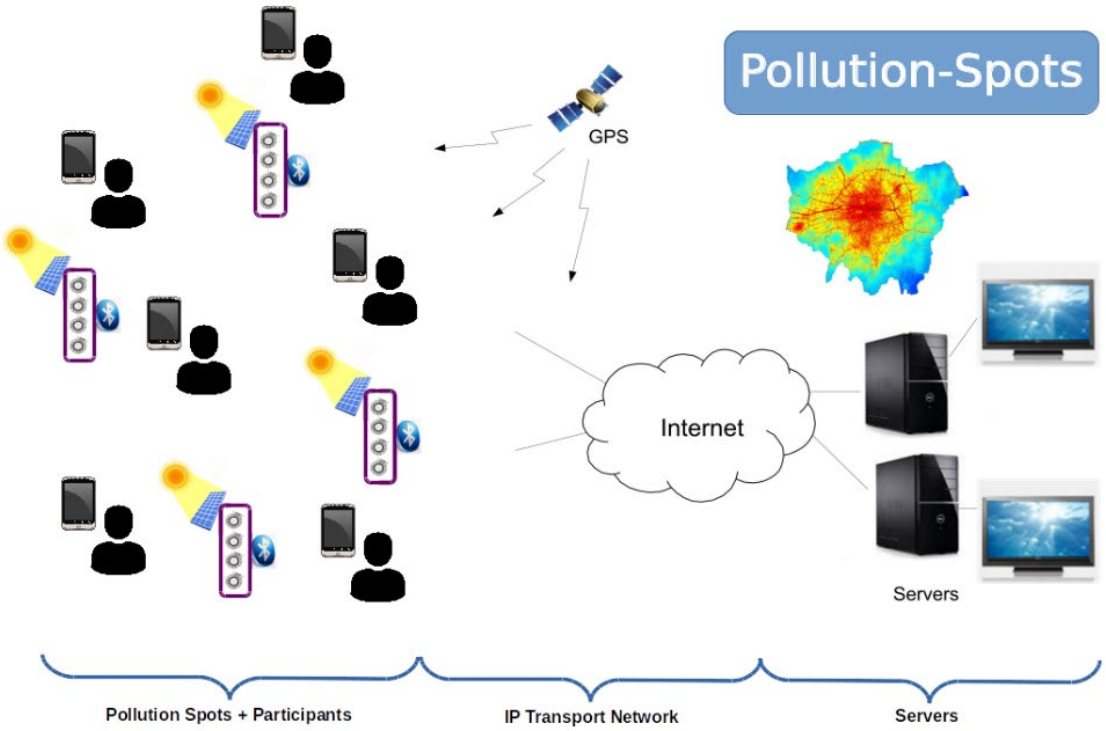


Figure 2.22 General architecture of the Pollution-Spots system [50]

Mocerino et al. [51] show the traffic data of passenger ships in the port of Naples used to estimate pollutant emissions starting from the European Environment Agency/European Monitoring and Evaluation Programme (EMEP/EEA) methodology. Ships are considered producers, and their receptors are shown in Figure 2.23. Data from different campaigns were collected several times inside the Naples port area, considering primary pollutants, such as SO_2, NO_2, PM_{10}, and Benzene. CALPUFF was used for simulations to monitor SO_2 concentration levels at different heights (0-60m) within the port area. The results were compared with the data measured at ground level.

Figure 2.23 Receptors [51]

Siagian et al. [52] presented a network system Using the MQ-7 sensor to detect carbon monoxide (CO) and the CCS811 sensor and MH-Z19 sensor to measure carbon dioxide (CO_2). In Figure 2.24, all these sensors are linked to NodeMCU, also known as the ESP8266 node MCU. Three NodeMCUs were used for the experiments, which involved monitoring the air quality in three different homerooms and sending data to a server using the nodes.

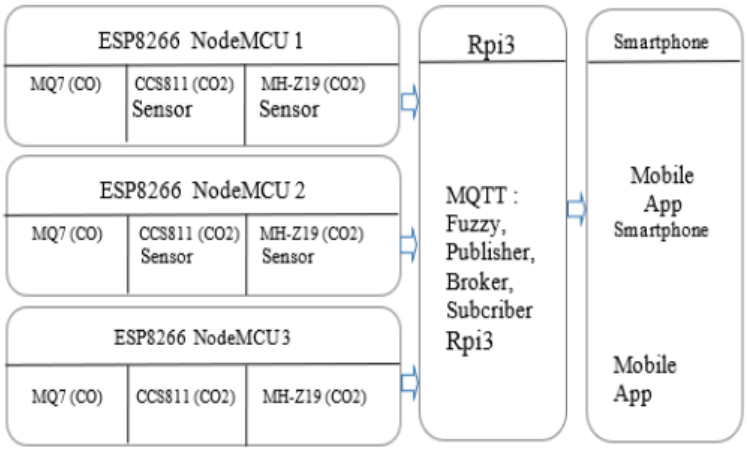


Figure 2.24 Module NodeMCU architecture [52]

Raspberry Pi3 has functioned as a server that collects data and receives informative Data from the sensors in the network system. Data from the CCS811 (TCOV) sensor and MH-Z19 were

processed using fuzzy logic rules for the decision-making process for CO_2. The results of this process were correct, moderate, and pleasant. Sensor values and measurements can be accessed using smartphones to provide good air quality.

Devarakonda et al. [53] illustrated a real-time vehicle-based mobile strategy for assessing fine-grained air quality, as shown in Figure 2.25.

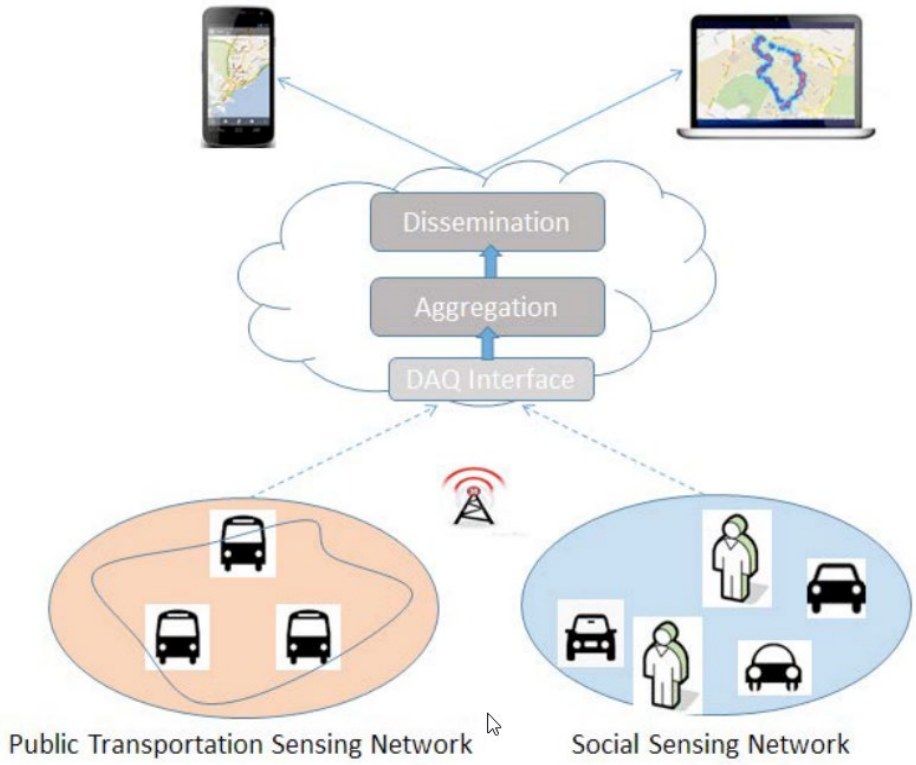


Figure 2.25 System Overview [53].

The sensing model was used for fixed routes and public transport such as buses. To gather fine-grained characteristics of air pollution, this model's Mobile Sensing Box (MSB) is shown in Figure 2.26. consists of a microcontroller board with add-on sensors, a peripheral GPS receiver, a cellular modem, and several individual sensing devices.

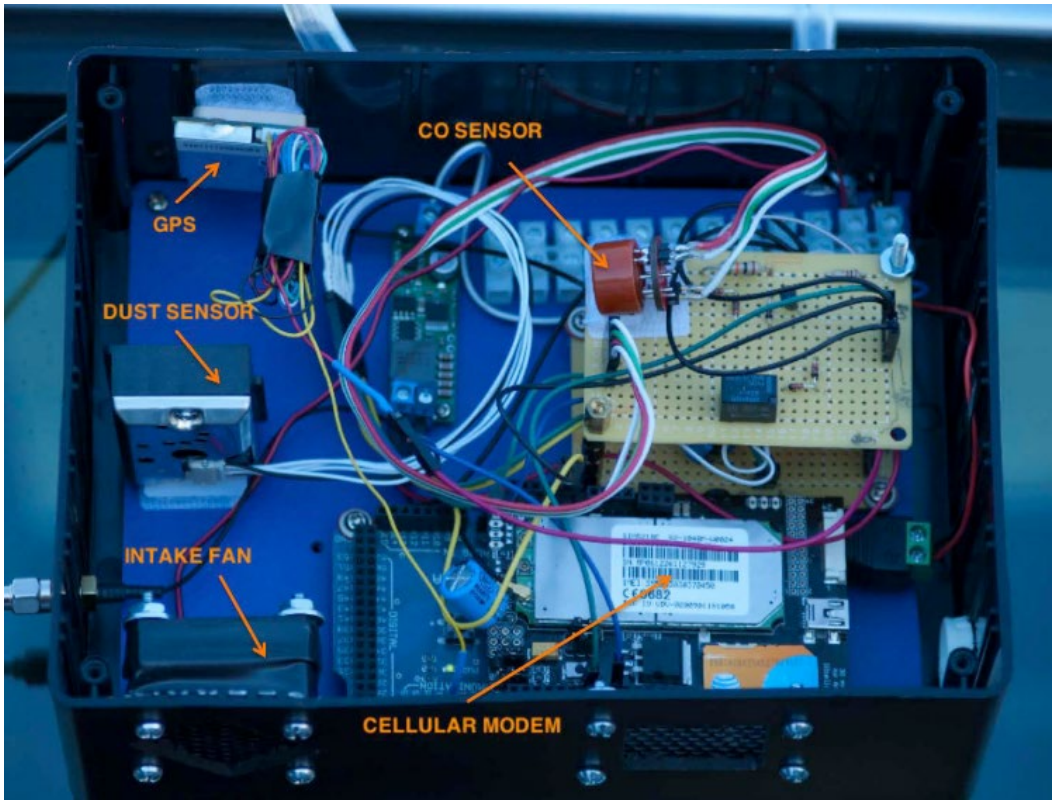


Figure 2.26 Mobile Sensing Box [53].

Figure 2.27 shows a PSD that can be placed in a vehicle and connected to a smartphone via Bluetooth. This experiment measured the CO and CO_2 levels in the atmosphere with MSB. The experiments presented several advantages and challenges. The data were transmitted over a cloud network using mobile applications.

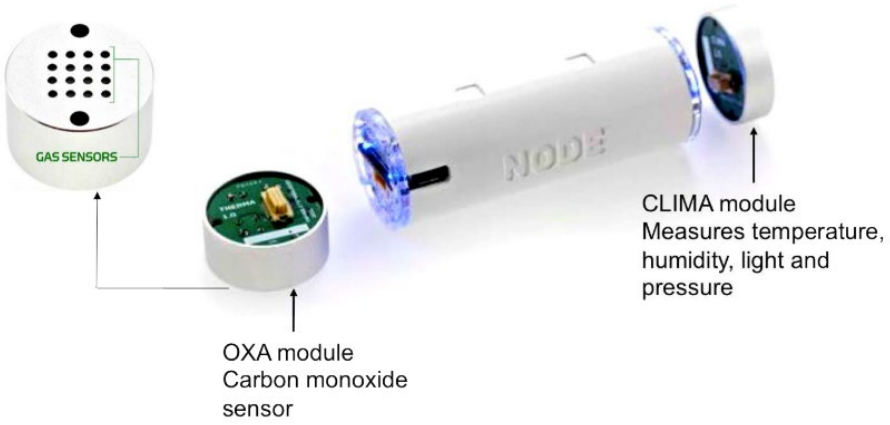


Figure 2.27 Personal Sensing Device [53]

Abraham et al. [54] presented a wireless sensor network-based low-cost indoor air quality monitoring device with a micro gas sensor, Arduino board and XBee modules. The designed network system could simultaneously collect the concentration levels of six pollutants from different locations. Researchers have also developed a least-squares estimation method (LSM) for sensor calibration and measurement of data conversion. The overall architecture of the wireless sensor network system for indoor air pollution monitoring is shown in Figure 2.28, and the sensor nodes are shown in Figure 2.29. This method has several advantages and challenges. The data were transmitted over a cloud network using mobile applications.

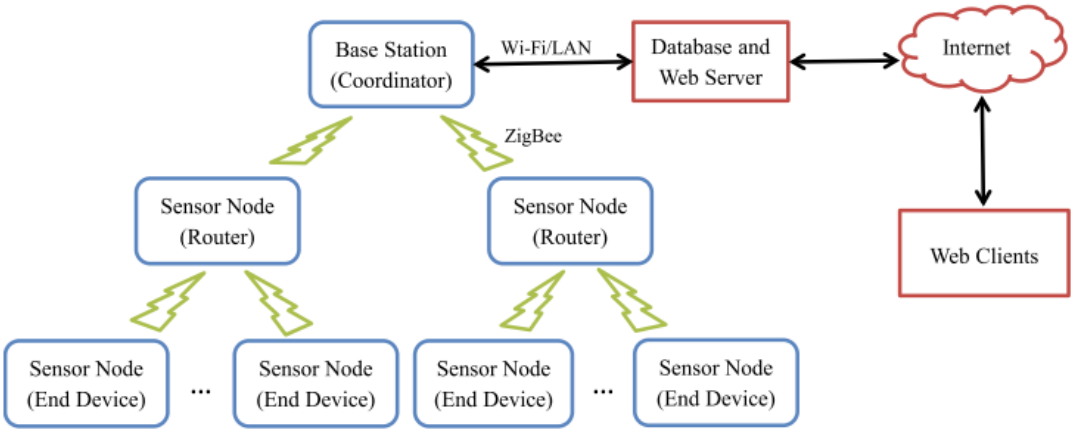


Figure 2.28 Overall wireless sensor network system architecture for indoorair quality monitoring [54].

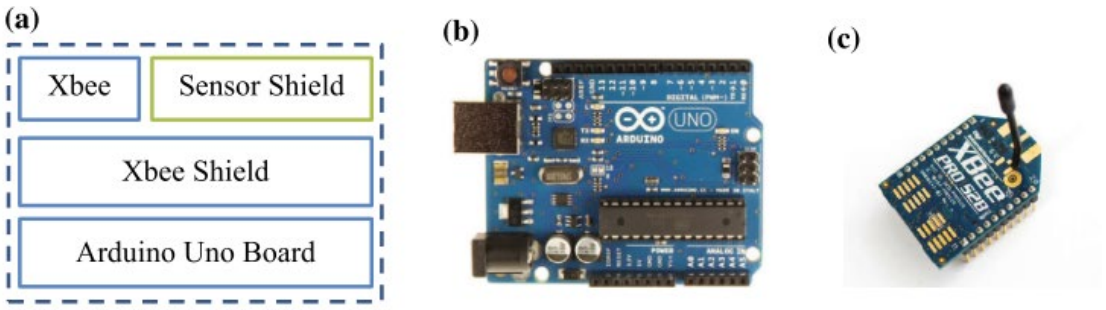


Figure 2.29 (a) Functional block diagram of the sensor node, (b) Arduino Uno microcontroller board, and (c) Digi XBee module [54].

Kadri et al. [55] describe an ambient real-time air quality monitoring system that uses machine-to-machine learning to connect scattered monitoring stations to the backend server wirelessly. The planned network was operated in Doha, Qatar, as shown in Figure 2.30. The system's foundation is using multi-gas (MG) monitoring stations that connect to a platform using M2M communication. Data filtering and cleaning were conducted on the backend server of the platform. The open-air package and R software were used in the network system for data collection. Accurate measurement data were also obtained while the research was conducted.

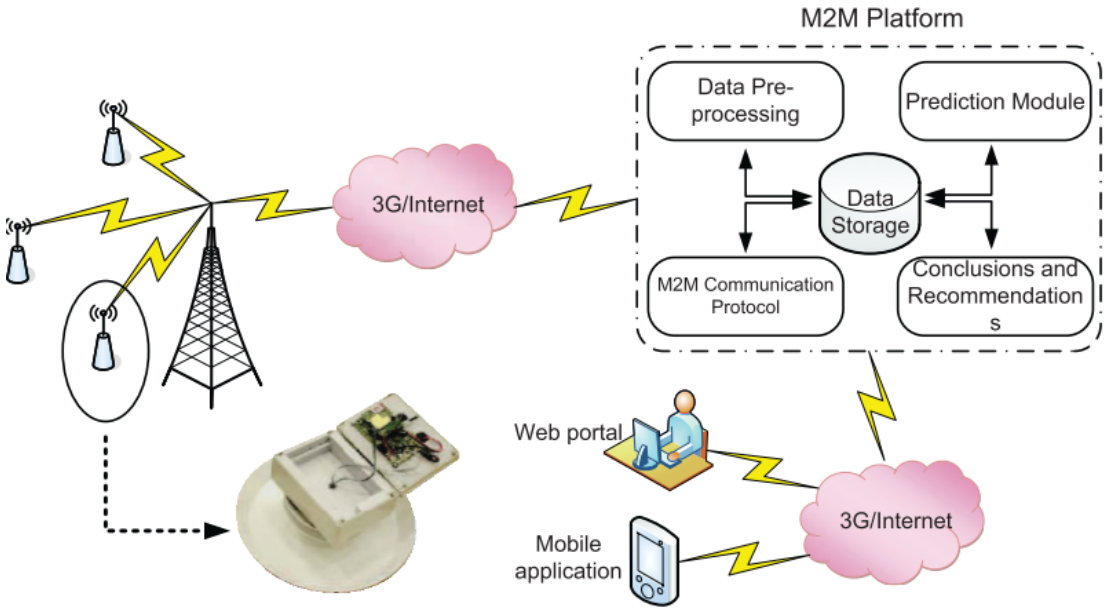


Figure 2.30 Air Quality Monitoring System Architecture [55].

Kang et al. [56] proposed a low-cost, portable RF-based sensor network using a 400 MHz transceiver to monitor air and water quality, including VOC, O_3, NO_2, and dust particles. The designed system was implemented in Incheon, South Korea, and effectively measured the pH and DO of the water. Figure 2.31 shows a block diagram that includes sensor units, GPS, and central processing units with network performance and efficiency to measure O_3 and NO_2. Therefore, developing a portable monitoring system to collect air and water quality information is essential.

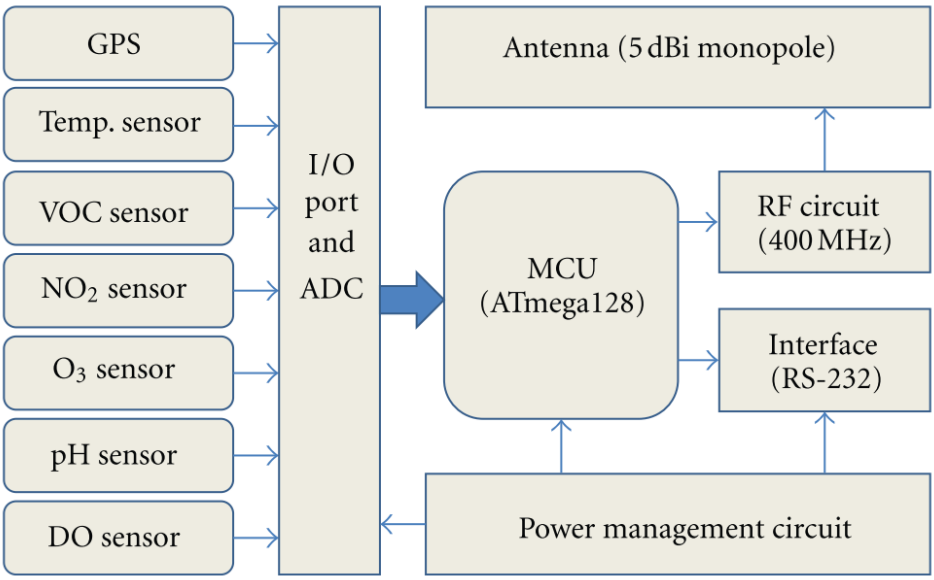


Figure 2.31 Block diagram of the RF sensor unit [56].

Park et al. [57] presented the LoRaWAN-based low-cost fine-grained air monitoring system, shown in Figure 2.32, where extensive simulations examined the network. For Internet of Things (IoT) solutions, one of the most well-liked Low-Power Wide Area Network (LPWAN) strategies is created to lessen the restrictions in current network systems. The network system can track detailed information regarding temperature, humidity, and PM levels in the air. The data were updated every hour at Seoul air monitoring sites, typically 5.5 km apart. Simulations were run in the LoRaSim simulator, which supports large-scale deployments, to verify the viability of the LoRaWAN. However, even with a 15-minute data update period, the system's sensor nodes may be quickly, cheaply, and densely deployed using little energy. Multiple gateways can be added to the network design to increase coverage and fidelity.

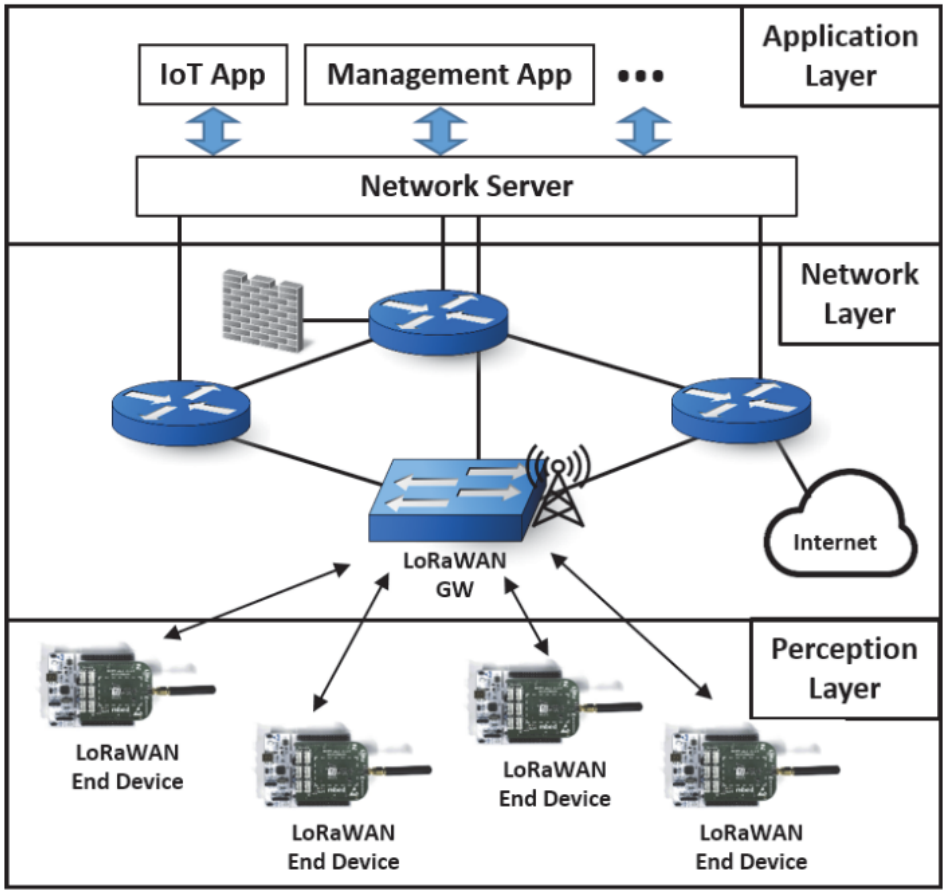


Figure 2.32 High-level system architecture [57].

Dooly et al. [58] developed the ultra-violet differential optical absorption spectroscopy fibre optic sensor to monitor vehicular emission. This paper shows the experimental results of the sensor with the concentrations of NO_2, SO_2, and NO, shown in Figure 2.33. During the experiment, the sensor was found to have a low susceptibility. The gas concentrations were calculated using the LabView program. There was an issue with cross-sensitivity between the gases caused by absorption. This study focused on implementing this project in a car in the future for installation and testing.

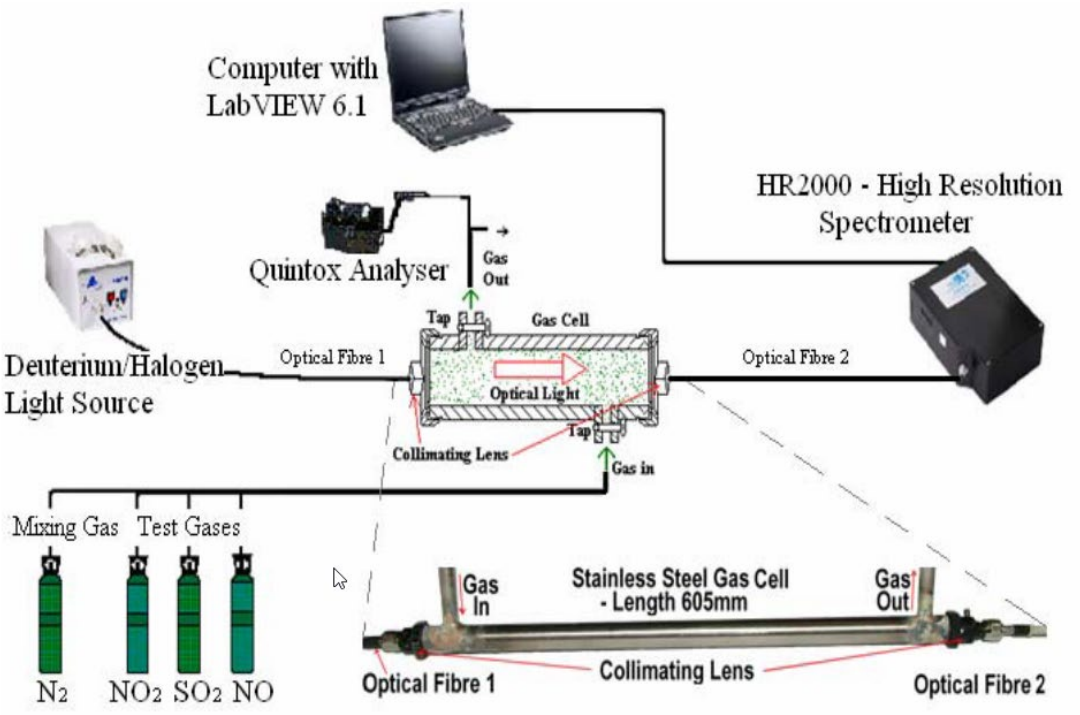


Figure 2.33 Experimental setup for cylinder-based testing [58].

Sajjan et al. [59], proposed IoT-based air contamination observing scheme to track the pollution levels and air quality through an internet-based web server. Sensors can collect data and are deployed at different locations, as shown in Figure 2.34. The proposed equipment includes an air quality control module that exposes the sector to the resonant intensity, cloud observation trace and an anomaly alarm module.

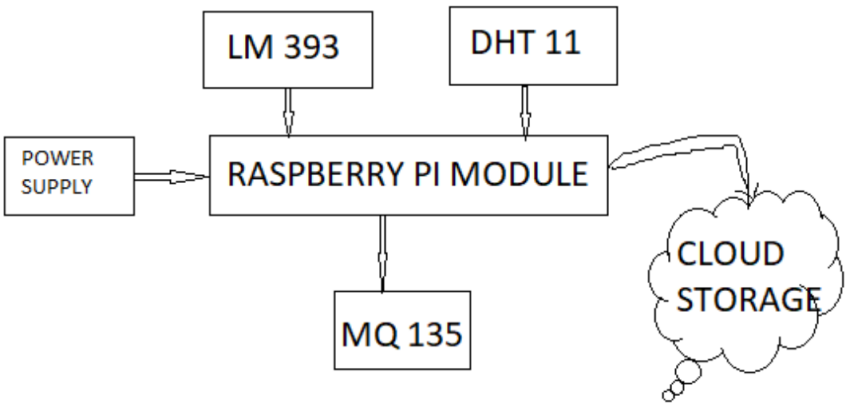


Figure 2.34 Block Diagram of the Raspberry Pi module and other devices [59].

The Raspberry Pi Wi-Fi module was used for regular data analysis, and the cloud-based monitoring module ensured the data were retrieved. Finally, in the event of unexpected circumstances, the Abnormal Alert Module notifies users. The suggested gadget uses sensors to detect toxic chemicals detrimental to human lifestyles. Send sensor input to monitor. Let us assume that the smoke parameter is above the acceptable range. In this example, the sensor updates the values every 30 seconds, whereas the algorithm generates analogue values. This is relevant from social and economic standpoints. Asthma and other respiratory conditions are caused by surface ozone, carbon dioxide, and particulate matter (PM). One of the most challenging tasks in today's society is monitoring the air quality in important regions. Therefore, this study investigated a clever Raspberry Pi method for monitoring environmental parameters. IoT principles raise the temperature, humidity, noise levels, and air quality.

Twahirwa et al. [60] implemented an IoT framework to create a customized LoRa-based air-pollution monitoring system. Two sensors, CO2 and PM2.5, were identified in the network system as the main pollutants affecting air quality and were crucial for assessing the compensated weather monitoring capabilities. The University of Rwanda's College of Science and Technology cafeteria, kitchen, and laboratory had sensor nodes installed, and the monitored values were relayed to the cloud using a gateway that supported the LoRaWAN protocol. Using a web-based user interface, the end user can query the system and retrieve analytical data to collect data for 11 months. The findings indicate that the kitchen environment had high parts per million for CO2 above 800ppm and PM2.5 concentration over 100ppm. Furthermore, the concentration of CO2 was 500 ppm, and PM2.5 was zero ppm in the laboratory area. Figure 2.35 shows the architecture of an air pollution monitoring system based on three-tier IoT, and Figure 2.36 shows the proposed system architecture.

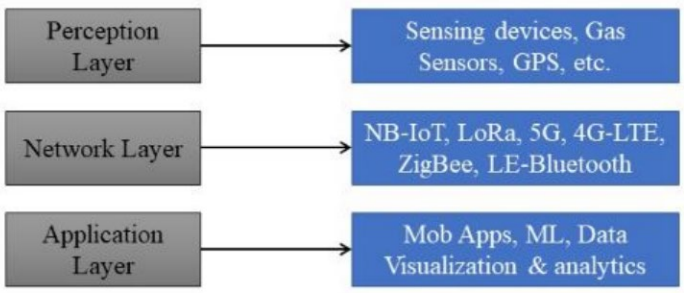


Figure 2.35 Three-tier architecture of IoT [60].

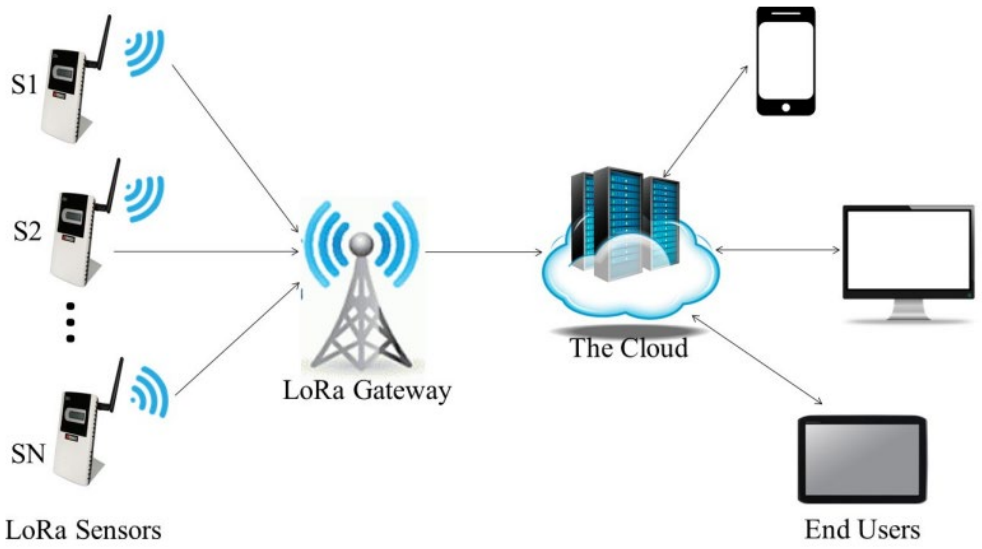


Figure 2.36 Proposed System Architecture [60].

LoRaWAN complained that the sensors were responsible for detecting the presence of air pollutants. The deployed sensors can operate in a non-condensed temperature range of 10±60ºC and a humidity range of 0–85% RH. They can also survive in extreme temperature and humidity conditions. The measured parameter readings are transmitted to the cloud through a gateway that connects the sensor and cloud components of the network and supports the LoRaWAN protocol. This study can be expanded to support the government's initiative to make Kigali a smart city. Other automated solutions are required to complete the system, owing to the high quantities of contaminants in the environment. The solutions consist of automatic Internet of Things-based cooling systems that activate wherever pollutants exceed the recommended limits.

Reyna et al. [61] demonstrated the outcomes of experiments performed on a locally created prototype of a personal environmental monitor. This prototype can measure the temperature, relative humidity, barometric pressure, altitude, worldwide location, and particulate matter. These devices are affordable, adaptable, and lightweight (PM). Additionally, it offers Bluetooth, Wi-Fi, and cellular data access. As these stations record the concentration of particulate matter at various distances or heights, the validity of the results of these investigations is sometimes questioned. A personal environment monitor (PEM) can be worn 24 hours a day as an alternative to these limitations to record time-series PM concentrations in various microenvironments where people do their daily business. In addition, dose-response

studies can be conducted to calculate the impact of PM on the person being watched, provided their physiological parameters (such as heart rate, lung function, and ECG). The prototype can be set up so the user can choose a USB or Bluetooth connection type or cloud to upload the captured data (Wi-Fi or GPRS). The user can also set recorded variables and sampling intervals (PM, temperature, relative humidity, barometric pressure, altitude, and global position). The sensor was connected to an Arduino Mega2560 R3. The prototype and two certified commercial PEMs This prototype can measure temperature, relative humidity, barometric pressure, altitude, worldwide location, and particulate matter. (Thermo Scientific ™ DataRAM ™ pDR1500) were placed in an isolated room. Room air was first cleaned with the HEPA Bionaire® air purifier model BAP1225 and then infused with wood smoke.

Perumal et al. [62] propose a framework to track air pollution, as shown in Figure 2.37, and an Arduino mini controls the proposed model. Air pollution monitoring systems aim to track and assess air quality using the knowledge acquired from external servers and stored online. Air quality is measured using millions of metrics (PPM) and depends on the components analyzed using Microsoft Excel.

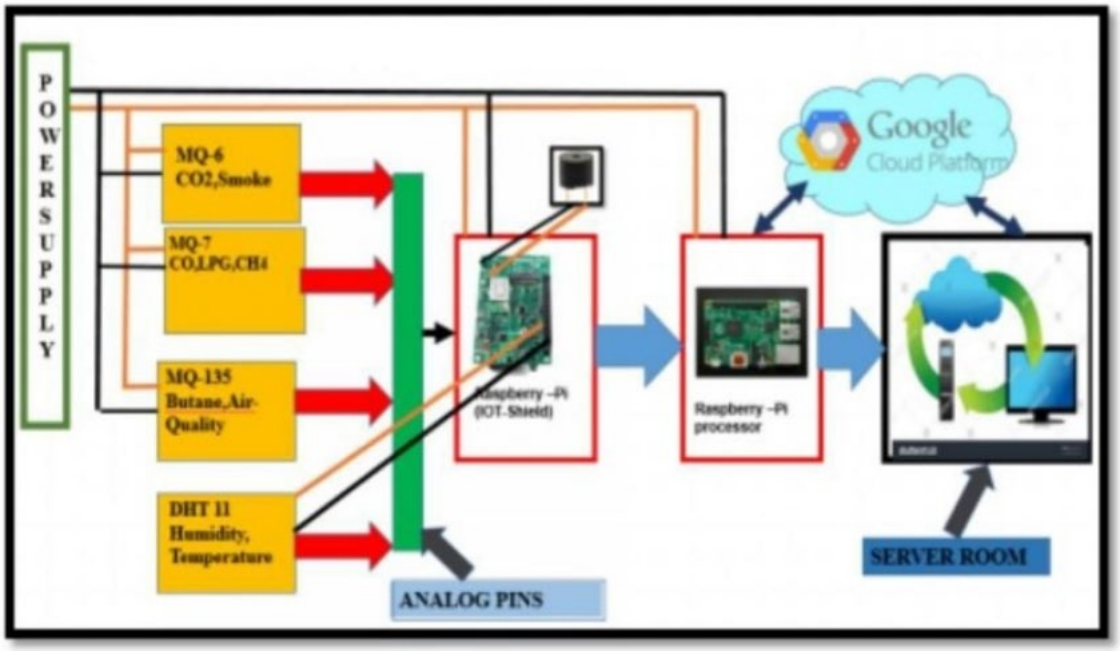


Figure 2.37 Methodology for air pollution [62].

Air quality measurements of the system were also precise. It is simple to monitor because the air quality in the PPM is presented on both the web page and the LCD screen. LPG sensors, typically used in homes, were added to the system. The system shows the humidity and temperature. Air pollution escalates because industries, factories, and vehicles are detrimental to human health.

Consequently, the authors created a tool or system to monitor and regulate surrounding air quality. When the air quality falls below a certain threshold, pollution levels may also serve as an alert. This system can identify NH_3, NOx, alcohols, benzene, flue gas, CO_2, and other hazardous gases. If a company's authorities do not reduce pollution, the system does. The computer shuts down if the carbon dioxide level exceeds a certain threshold. Similarly, in the event of a fire, the exhaust fan will operate, and if LPG and liquid colour changes are detected, appropriate measures will be taken to minimize the situation. The status of the sensors in a particular industry can be accessed from different locations.

Od et al. [63] proposed the Internet of Things (IoT) network, a real-time air quality monitoring system based on several LoRa sensor nodes, to implement an IoT network. Environmental characteristics were recorded using numerous sensor nodes, with the Arduino as the primary control board. The integrated data are then sent to the cloud via a single device. Users can easily synchronize data from numerous sensor nodes throughout the system in the cloud by connecting their gateway device to Wi-Fi, as shown in Figure 2.38.

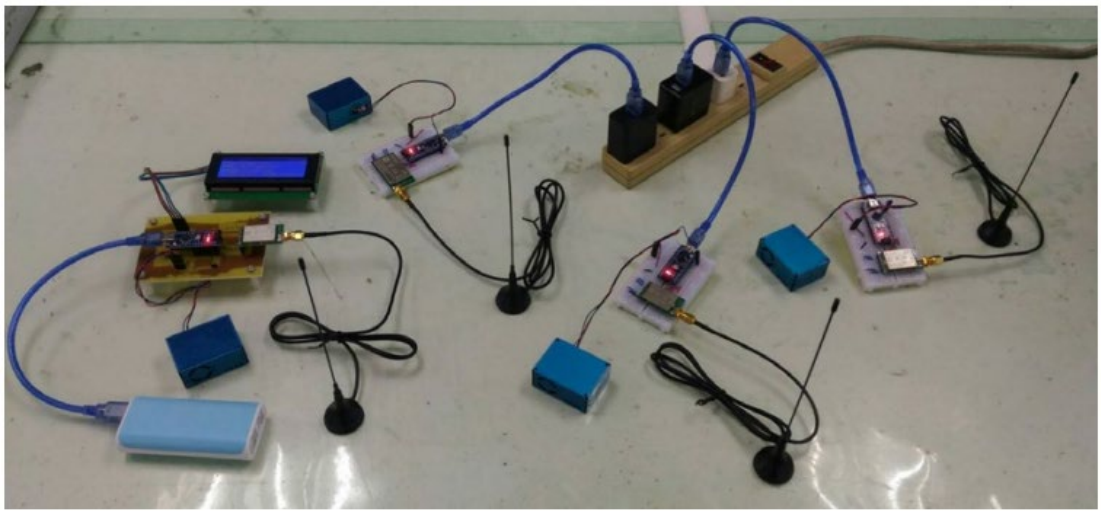


Figure 2.38 Hardware Architecture [63].

Users can access system data through mobile APPs and linked cloud websites, and the gadget provides air quality data for broad areas such as campuses. This information includes several characteristics such as temperature, humidity, PM_1, $PM_{2.5}$, and PM_{10} levels. This study leveraged the low-long-range (LoRa) technology in conjunction with air quality sensors and cloud data services to create a dependable and stable air quality monitoring and data analysis system. The system has a mission success rate of up to 99% and a data transmission range of up to 99% indoors. In addition, the device is inexpensive and has a long battery life. A PMS5003T sensor that can identify 12 different air quality characteristics was employed in this investigation. The wireless LoRa network in this study has the advantage of having a range of up to 15 km and consuming less power. The total hardware cost was less than USD $ 200 for the four sets of wireless sensor nodes. Therefore, it is a low-cost, high-quality air quality monitoring system.

Tao et al. [64] proposed a study for air pollution monitoring using the optical fibre chemical sensor to detect ammonia concentration in the high-temperature gas stream, as shown in Figure 2.39. This sensor uses a copper (II) chloride (CuCl2)-doped porous silica optical fibre, and the ammonia sensor calculates the ammonia concentration at an elevated temperature of 450°C. This study developed a complex inside an optical fibre made of porous silica. The ammonia content of the gas samples was at equilibrium. This innovative sensor can monitor NH_3 in high-temperature gases important for energy production [64].

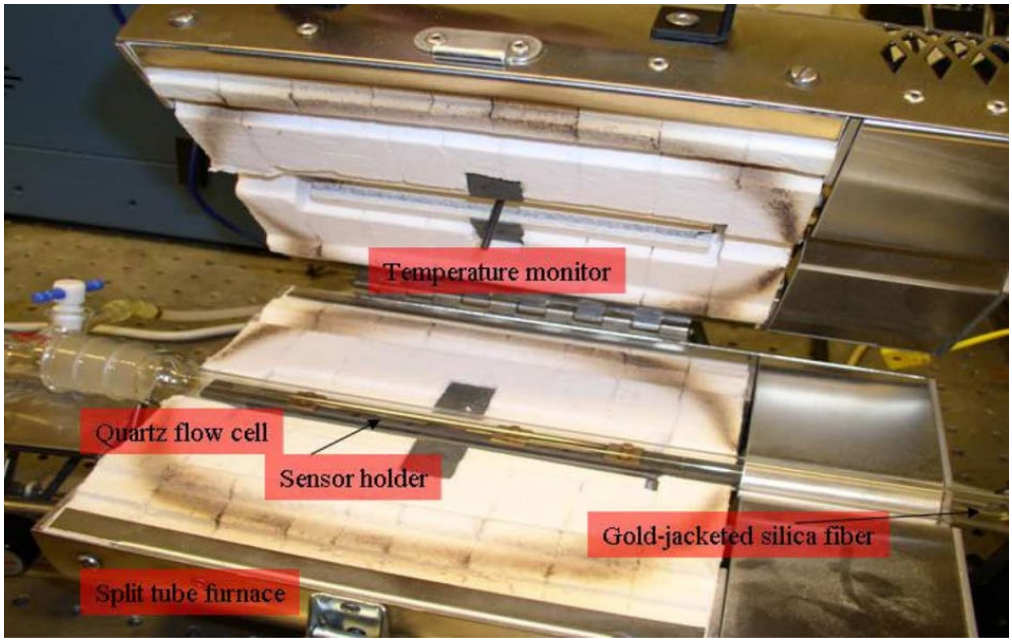


Figure 2.39 Fiber-optic sensor [64].

Nandanwar et al. [65] designed this study project for the well-known and established science of real-time air quality monitoring, which started in the 1980s. Since then, this method has not been widely used, and the solution has been employed to quantify the intricate and expensive air pollution. Interestingly, the most advanced technology is currently being utilized to measure air quality solutions quickly and correctly. These devices are more affordable and more compact than ever before. The display device uses one of the 'DSM501A'Sam's newest and least-priced dust sensors. This sensor can detect $PM_{2.5}$ and PM_{10}, and the proposed block diagram is shown in Figure 2.40.

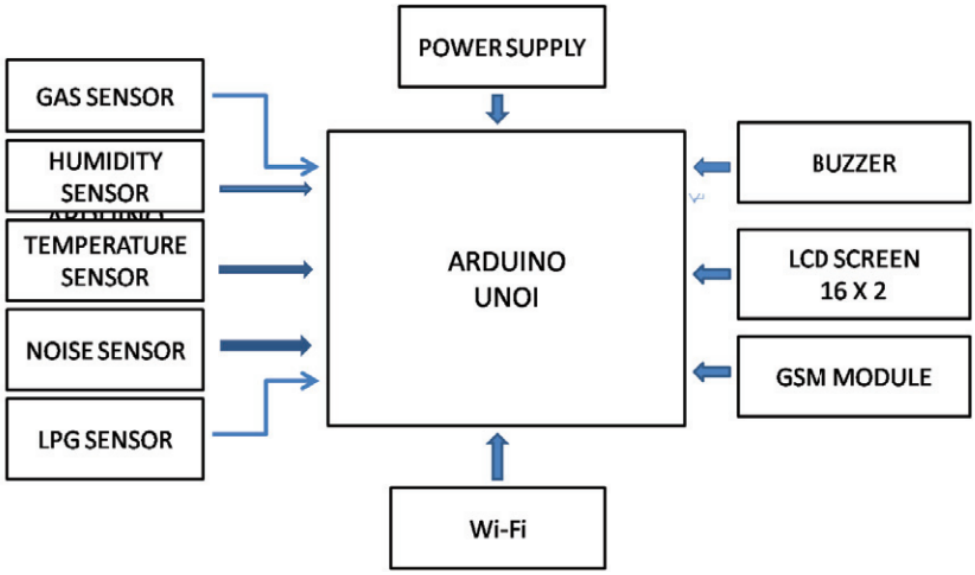


Figure 2.40 Block Diagram of Proposed Model [65].

This study proposed and evaluated a sound monitoring and air pollution control system. This method is ideal for reducing pollution and is entirely based on the Internet of Things (IoT). This technique can occasionally detect pollution levels (air or sound). This article examines several topics, including the hardware and software used in the proposed system, the fundamental elements of the Internet of Things (IoT), how it works, the microcontroller (Arduino Uno R3), and its architecture, function, and gas sensor. Modelled and used in the proposed system: Wi-Fi model ESP8266 and a sound sensor. The MQ135 and MQ6 gas sensors provide pollutant levels for different gas types. Wi-Fi connects all Internet operations, and an LCD is used for the visual displays. Automated air quality and noise control systems represent a step towards achieving maximum connectivity. Acoustic monitoring systems can solve and address critical problems in highly polluted environments.

Wang et al. [66] developed the Carbon-Monoxide sensor-based wireless sensor network to monitor air pollution. Carbon Monoxide directly affects health because of its affinity for blood hemoglobin. It is involved in both acute and chronic human health problems. The authors focused on CO detection and used the principles of non-dispersive infrared absorption. The CO-based network implementation uses analogue sensor units, semiconductors, and electrochemical sensors, as shown in Figure 2.41. Data acquisition methods were used to obtain the output signal of CO concentration.

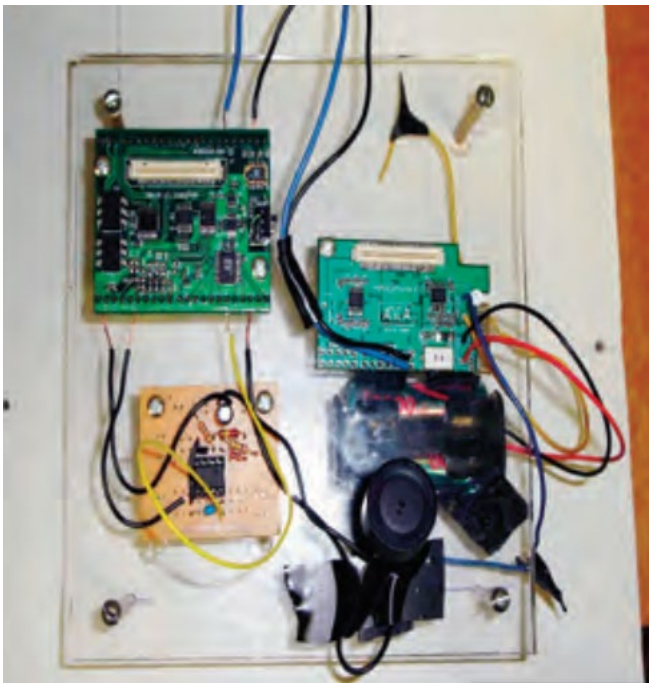


Figure 2.41 Components of a CO-WSN unit [66].

Liu et al. [67], Based on the Internet of Things, a CO_2, $PM_{2.5}$, temperature, and humidity-measuring indoor air quality detector (IAQD) was created and tested in households (IoT). The hardware and software designs of IAQD were thoroughly explained and are shown in Figure 2.42. Throughout the winter, seven IAQDs equipped with Zigbee radio modules were regularly installed in the target building. At 2-minute intervals, the gateway sequentially gathers sensor data from each IAQD and transmits them via GPRS or 4G to the cloud servers. The cloud platform is accessible to authorized users through web browsers or mobile apps. The highest $PM_{2.5}$ concentrations were ten times higher than typical during the cooking period, and CO_2 concentrations with the door closed increased to 2500 ppm compared to the door opened at night, according to monitoring data analysis results, which poses a hazard to the inhabitants' health.

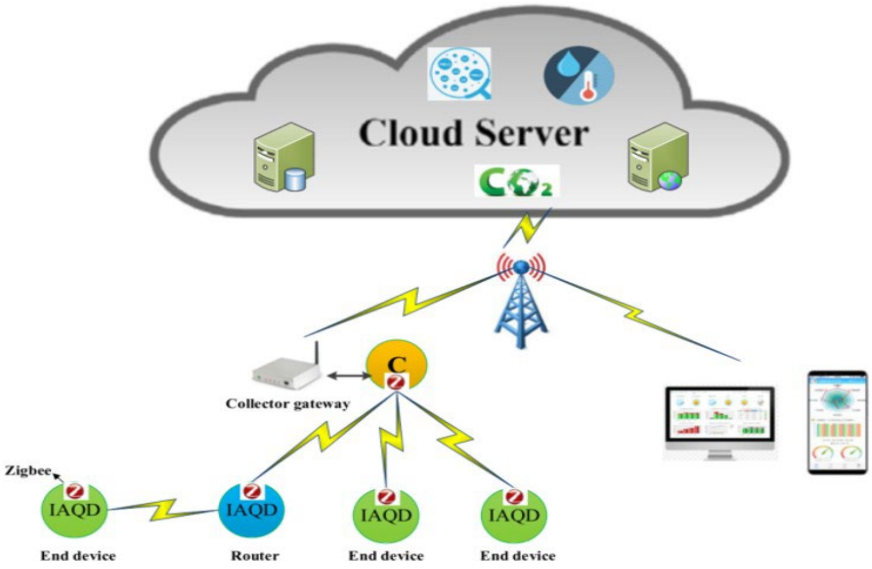


Figure 2.42 Cloud Server [67].

By analysing the continuous measurement data of the system for one month in winter, the following conclusions can be drawn:

- Concrete walls significantly affect the wireless signal quality of ZigBee networks. The packet loss ratio exceeds 8% after passing through the two barriers. The concentration of $PM_{2.5}$, which would have a major negative impact on human health, would be ten times higher than usual when cooking.

- Due to infrequent window openings in winter, the CO_2 concentration in the bedroom is higher than 1000 ppm, especially when sleeping at night, resulting in a higher CO_2 concentration. At night, the CO_2 concentration reaches 2500ppm when the door is closed. It should also be considered whether outdoor $PM_{2.5}$ concentrations are suitable for daytime ventilation. The average temperature in the bedrooms in the north was approximately 3°C lower than that in the south. The humidity in all rooms is dry; therefore, special attention should be paid to developing colds and respiratory illnesses.

- To increase the penetration of interior wireless signals and decrease PLR, the Zigbee wireless sensor network will eventually be upgraded and converted to LoRa communication. Additionally, it was discovered during the experiment that the trajectory of human action was strongly correlated with changes in the CO2 concentration. Through the CO2 concentration, the fluctuations can be investigated to determine the interior occupancy and follow the personnel's activity path. On the other

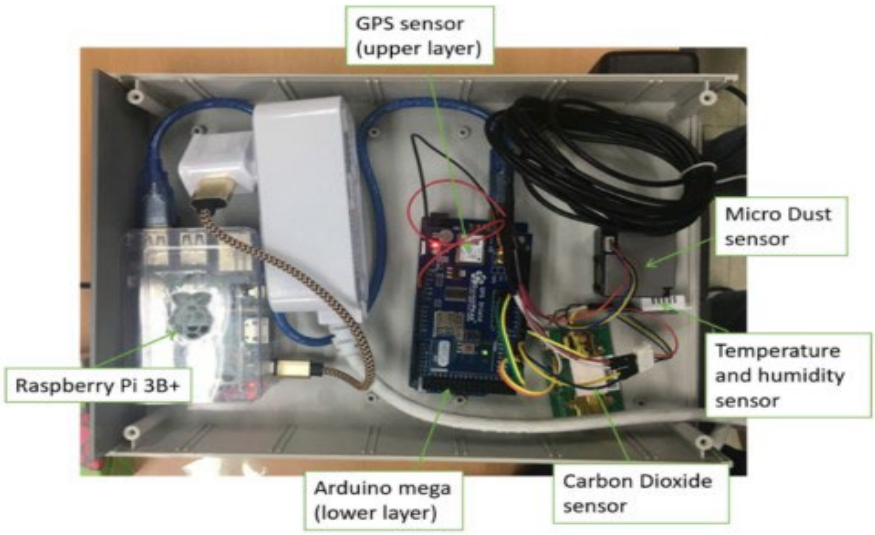
hand, continuous observation of various structures allows for a more thorough investigation of indoor air quality.

Zhao et al. established a network based on the complex network theory for disseminating air quality data, analyzed the primary stations in the area, and defined the interaction process between various sites. First, the authors analyzed the pollution monitoring data for the Beijing Tianjin Hebei region, abstracted the monitoring station as a network node, and created a boundary set based on the pollution path determined by reachability calculations over space and time. Network weights were assigned during propagation to indicate the level of station interaction by aggregating the influence of the propagation network path at various timestamps. Second, to identify the significant sources of contamination, the PageRank algorithm assesses the significance of the nodes in the distribution network.

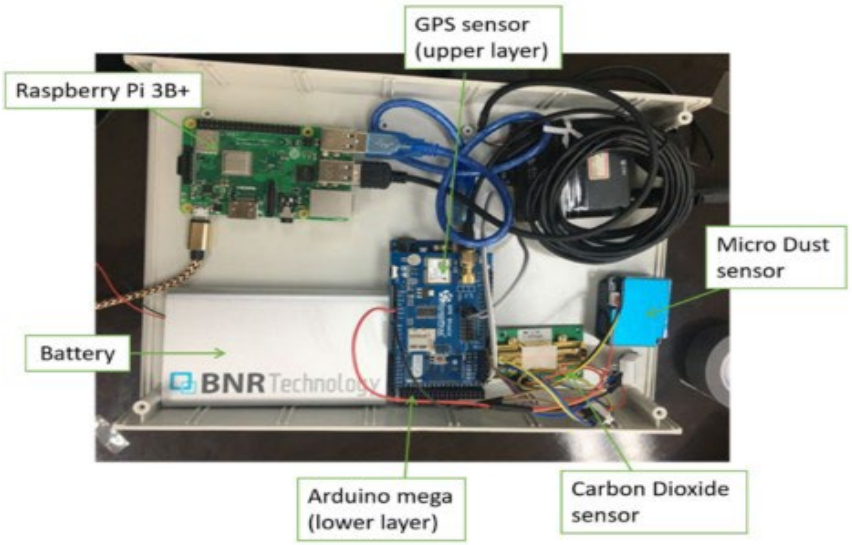
The main station considers both the quantity and quality of the experimental results to verify the network's accuracy and the node ranking algorithm and show how the node rank distribution and pollution conditions are relevant. This study provides a solid theoretical foundation for the timely prevention and control of air pollution and a rationale for choosing the locations of monitoring stations. Regarding the theoretical contribution, this study performed regional influence measurements in the air quality system, disclosed the regional pollutant propagation, and suggested a more logical and scientific analysis approach. As a result, our work offers technical assistance and theoretical guidance for additional air quality analyses, such as predictions, in accordance with realistic scenarios. In practical applications, key nodes of air quality contribute to effectively controlling pollution by taking unique prevention and control measures and benefiting overall air quality.

Zhang et al. [68] research focuses on using stationary and mobile IoT sensors mounted on vehicles that patrol the area to simulate the air quality pattern in each area. The proposed method can examine the range of air quality variations in the surrounding areas. The evaluation showed promising results for effective air quality monitoring and prediction in smart city applications. According to our analysis, a smart city application can benefit from accurate air-quality monitoring and forecasting. The authors suggested using a hybrid strategy to deploy numerous mobile IoT and static sensors to monitor air quality, as shown in Figure 2.43. Static sensors can provide holistic information continuously. However, mobile sensors can reduce

static sensor errors by providing more accurate data for specific areas. In this study, the collected data is used to build predictive models that provide quick insights into the air quality of people.



(a) Fixed Sensing IoT Device.



(b) Mobile Sensing IoT Device.

Figure 2.43 Two types of air quality monitoring IoT sensor modules [68].

This research has also developed visualization tools to analyze better, predict air quality, and inform professional researchers and general users. The main results of their study are as follows:

- A hybrid approach integrates stationary and mobile IoT sensors to measure and predict air quality data. The prediction results were analyzed using various machine models to demonstrate the validity and effectiveness of the proposed approach.
- A visualization tool that shows the relative distribution of air pollutants, focusing on PM_{10} and $PM_{2.5}$.
- It provides an intuitive understanding of air quality around people. In addition, the proposed system can be implemented using public transportation systems such as buses and taxis equipped with IoT sensor devices to measure various areas.
- The predicted air quality data were used in multiple scenarios when planning the outdoor activities. Environmental protection departments control environmental problems.
- This can significantly reduce pollution and pave the way for future studies to monitor urban pollution. This research suggests studying and testing a smart monitoring platform for urban pollution in rural areas based on big data in the context of big data paired with smart monitoring technologies. Experiments show that the platform's performance described in this article has been examined and has a certain level of dependability. Specific reference values exist for environmental monitoring.

A hybrid approach integrates stationary and mobile IoT sensors tomeasure and predict air quality data [68]. A hybrid approach that integrates stationary and mobile IoT sensors tomeasure and forecast air quality data [68].

Wies et al. [69] discussed the economic analysis and environmental impacts of integrating a photovoltaic (PV) array into a diesel-electric power system for small changes. The simulation was performed using MATLAB. This environmental model was developed to monitor the CO_2, PM, and NOx emissions. This network system was developed for three cases: diesel only, a diesel battery, and PV with a diesel battery. A hybrid optimization model was implemented for electric renewables (HOMER), which determines the air emission results.

Zeng et al. [70] suggest a study on comprehensive data-based urban pollution monitoring intelligent platforms in rural areas to improve urban pollution monitoring. To develop a smart monitoring platform for urban pollution in rural areas based on big data, this study integrated big data with smart monitoring technology. The dependability of the platform was tested using simulation tests that monitored air pollution in the metropolitan and rural areas. This study demonstrates that the proposed big-data-based smart urban pollution monitoring platform for rural areas has an average reaction time of 2.142 seconds. One hundred per cent of transactions are successful, and an appropriate platform performance test was conducted. This platform allows real-time monitoring and remote control to record the dynamic changes in the environment and enable early problem detection. This study examines urban pollution intelligent monitoring platforms in rural areas based on big data and background testing in the context of big data integrated with smart monitoring technologies. The tests show that the platform's performance is reliable, as suggested in this study. This has a specific reference value for environmental monitoring studies.

Yong et al. [71] developed the system to continuously improve Particulate emissions from industrial chimneys and flue play. Measuring and monitoring gas-stream particles is a technical challenge. This study focuses on legislative requirements covering various industrial processes for monitoring particulate emissions. According to the Environmental Protection Act of 1990, the automatic and continuous monitoring of particle emissions from stacks can be accomplished with a practical and affordable solution using electrostatic sensor technology and digital signal processing algorithms. Electrostatic charges are produced using these methods. It is anticipated that the electrified particles will encounter another surface.

Yu et al. [72] proposed a WSN based on an indoor air quality network structure, as shown in Figure 2.44. It features remote parameters and a firm updating mechanism to increase the adaptability and convenience of the network system. Plug-and-play air sensing devices, IEEE 1451.4 standards, real-time remote firmware upgrades, etc., are the main factors considered by this network. Dissemination time, energy usage, packet count, and network longevity were the four parameters used in the experiment to compare the proposed method (PARM). This network may vary the monitoring frequency and efficiently lower the error percentage, which will aid management in enhancing the air quality.

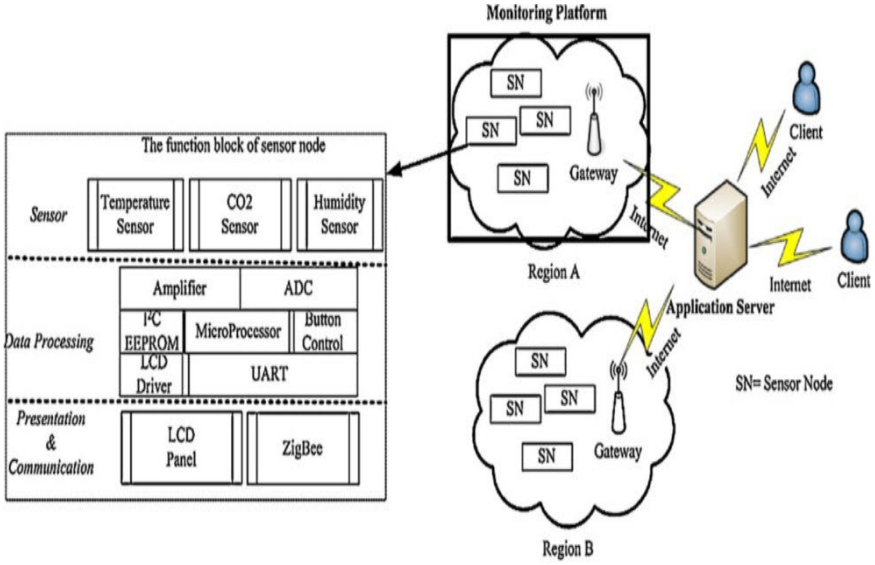


Figure 2.44 System architecture [72]

Xiao et al. [73] proposed a decentralized, centralized remote air quality monitoring system to realize remote monitoring of $PM_{2.5}$ and PM_{10} concentration changes, as shown in Figure 2.45. The system uses multiple control quality monitoring stations to detect the concentrations of $PM_{2.5}$, PM_{10}, and S0S011 sensors and a monitoring center server using the SIM900AGPRS module. The Control Center Server receives processes and statistical analyses, displays data from various control centers, and stores them in an SQL 2008 database. This information is useful for creating hourly, daily, monthly, and annual reports.

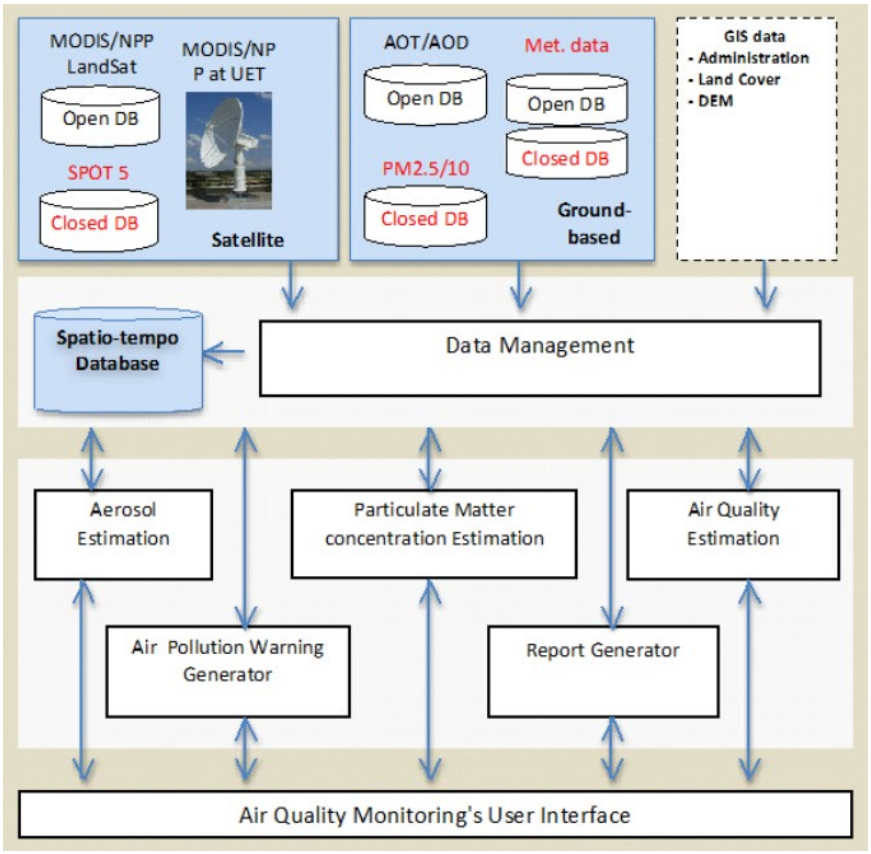


Figure 2.45 Overall designed architecture of large-scale air pollution remote alarm system [73].

The implementation results show that the GPRS/SMS-based remote monitoring system, shown in Figure 2.46, for gas emissions from construction and mining companies was analyzed in detail and provided with a program code for gas emission monitoring from construction and mining companies. The proposed system was highly flexible, efficient, and accurate. The SIM900B quad-band wireless GPRS/SMS module achieves low power consumption for voice, short message services, and data completion. In addition to sending facsimile information, an ADSP2183 high-speed data processor was used as a central data acquisition module to collect real-time parameters and monitor the system's operational status. The results of system tests show that the proposed method exhibits high flexibility, efficiency, and accuracy.

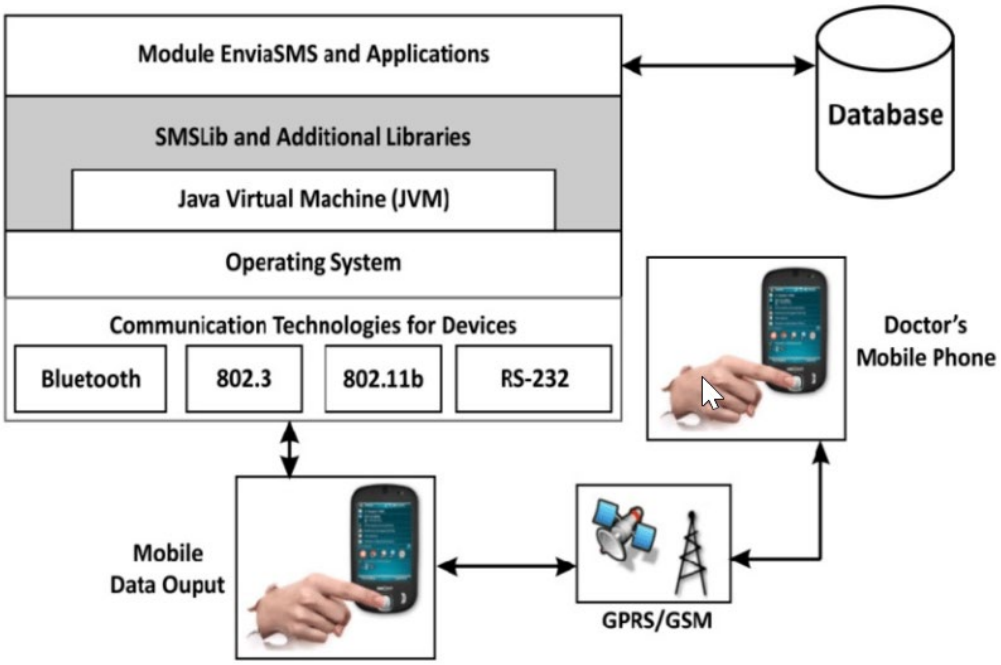


Figure 2.46 GPRS/SMS Communication Module [73].

Zheng et al. [74] developed a new approach to monitoring air quality based on cutting-edge Internet-of-things. Air quality data was collected using a portable sensor. Low-Power Wide Area Network (LPWA) technology has been combined with M2M learning techniques to solve air quality difficulties. An Internet of Things (IoT) cloud was used to construct the LPWA network architecture over a significant coverage area, as shown in Figure 2.47.

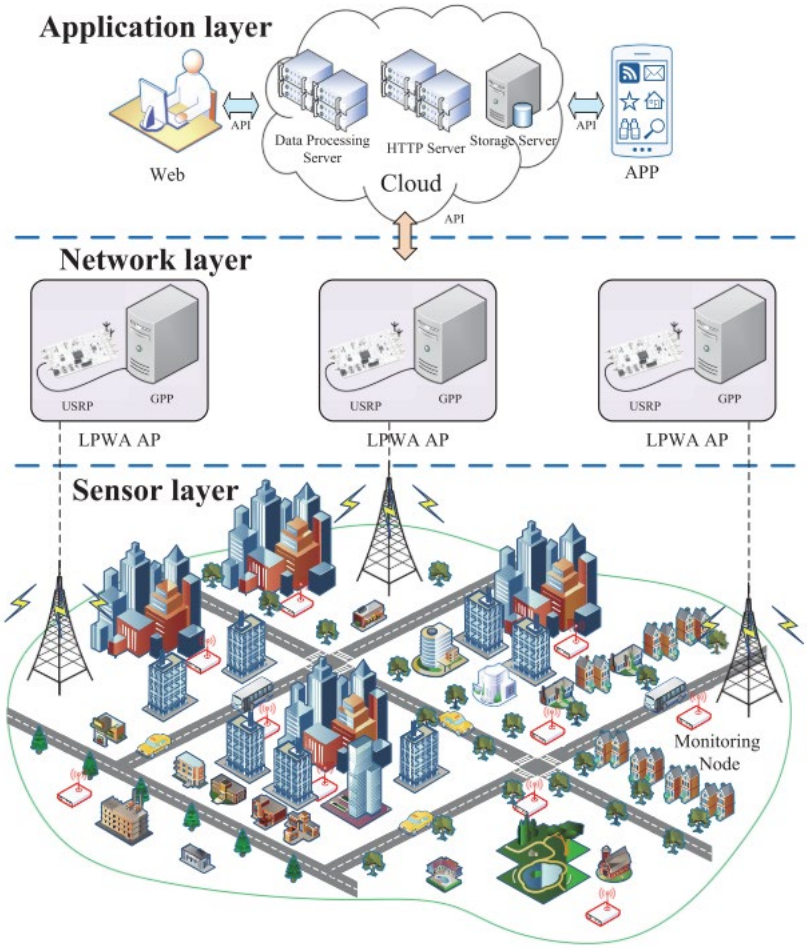


Figure 2.47 LPWA-based air quality monitoring system [74].

Portable monitoring sensor nodes can operate all day on a battery or solar panel and are designed for quick deployment. A GPP-based SDR platform was used to implement all the AP features. The sensed data were analyzed in an IoT cloud and saved in a database. Numerous tests have been performed in metropolitan settings to verify the dependability of the proposed system. Some intriguing insights emerged when comparing air quality trends with other comparable data. It is believed that extensive, long-term air monitoring can considerably advance our understanding of air pollution and lead to the discovery of partial solutions to this issue. Portal monitoring nodes can run all day on a battery or solar panel and are designed for quick deployment.

Aicardi et al. [75] developed a new approach based on the movable system to house imaging and air quality sensors for the environment. A cargo bicycle was used to implement proper sensors. The participatory network system, in which gadgets are directly connected to cell phones, was also addressed in this study. The planned network efficiently identifies problems

with the precision of inexpensive sensors. The low pollution output and minor changes in the collected environmental data are the advantages of the proposed network. However, this method has disadvantages, including lack of accuracy and subjecting the gadgets to physical shocks.

Peng et al. [20] established a method to reduce the power consumption of WSNs and enhance gas sensor accuracy. Modern technology and commercial off-the-shelf photoionization (PID) gas sensors have been used in this implementation. A small number of sensor nodes are positioned in the rooms of the monitoring system depicted in Figure 2.48 to measure the VOC concentrations in a small number of places.

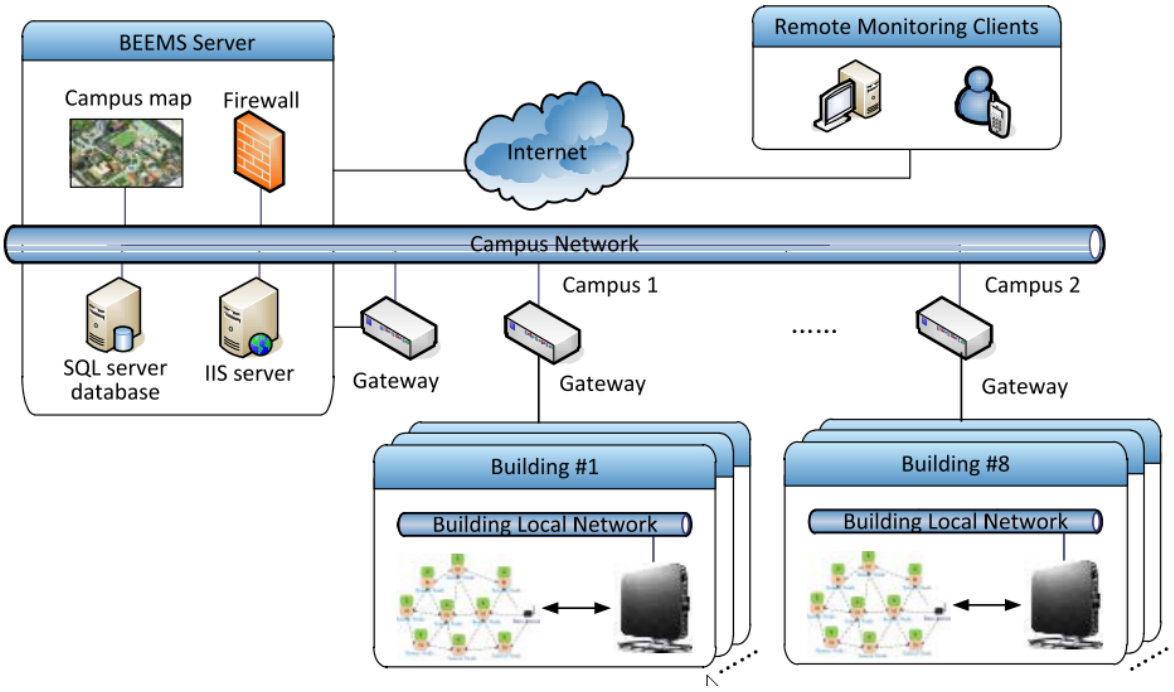


Figure 2.48 Deployed network at campus [20].

Thus, the network comprises photoionization detectors on end-device sensors, routers extending the network across large distances, and a coordinator interacting with a computer. The Atmel RF230 ZigBee module and ATmega16 microcontroller were utilized to process the communication data while consuming little power efficiently. Power usage and sensing effectiveness are prioritised, considering different smart tasking and power management algorithms. The graphical user interface used by the computer displayed the measured data on

the screen. The structure of the multilayer system permits the incorporation of sensor nodes scattered at vast scales in one or more buildings, and the measurement nodes are compact and straightforward to use. The configurable browser/server multilayer structure of the application software also allows for integrating other types of monitoring data, including power usage, temperature, and humidity. IAQ monitoring and building energy efficiency are also possible.

Brienza et al. [76] presented a low-cost monitoring device that enables real-time knowledge of pollutant gas concentrations in various cities. uSense, a low-cooperative monitoring tool that measures the number of harmful gases in the air, is shown in Figure 2.49. Using uSense, data collection through the network was attempted for in-field experimentation in various parts of the city. The main goal of this study was to examine whether a low-cost monitoring system can offer accurate air quality predictions while commenting on the advancements made to uSense. According to experiments, low-cost sensors are less accurate than the official data.

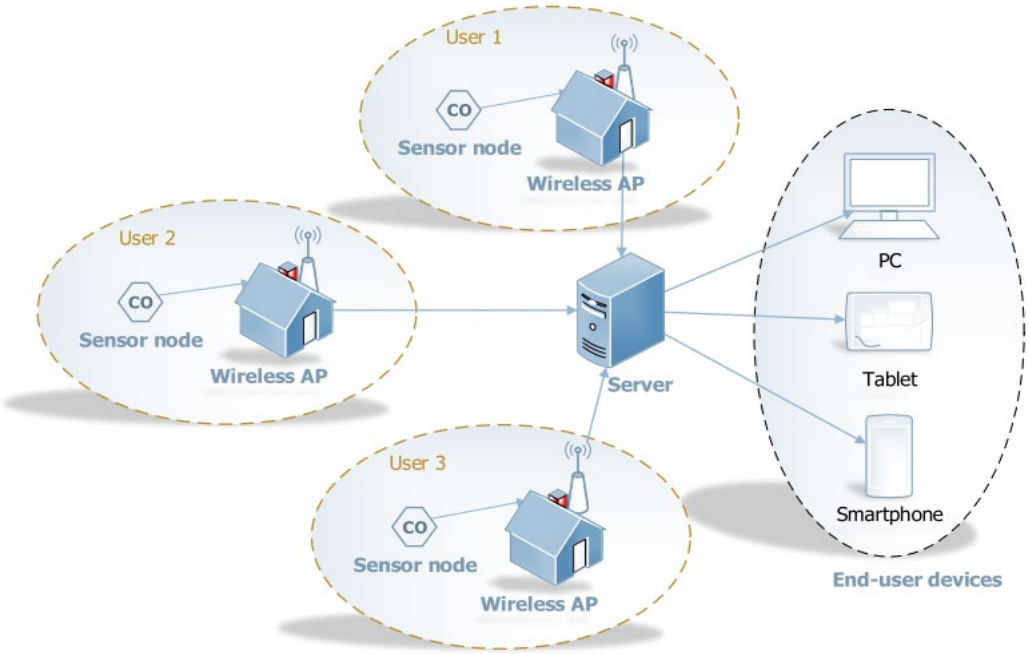


Figure 2.49 System architecture [76].

Polychronidou et al. [77] provide a distributed air pollution monitoring system to prevent the effects of air pollution on the respiratory system. The biomonitoring module of the system was designed as an air pollution-monitoring system. The network depends on cutting-edge technology and software (monitoring artificial intelligence, visual analytics, and clinical

decision-making systems) (smart sensors and wireless personal bio-monitoring systems). The monitoring system is shown in Figure 2.50, which presents the objective of the Take-A-Breath project to protect respiratory health. A bio-monitoring system is a key element of the overall architecture. Its features guarantee data-gathering and sensor communication.

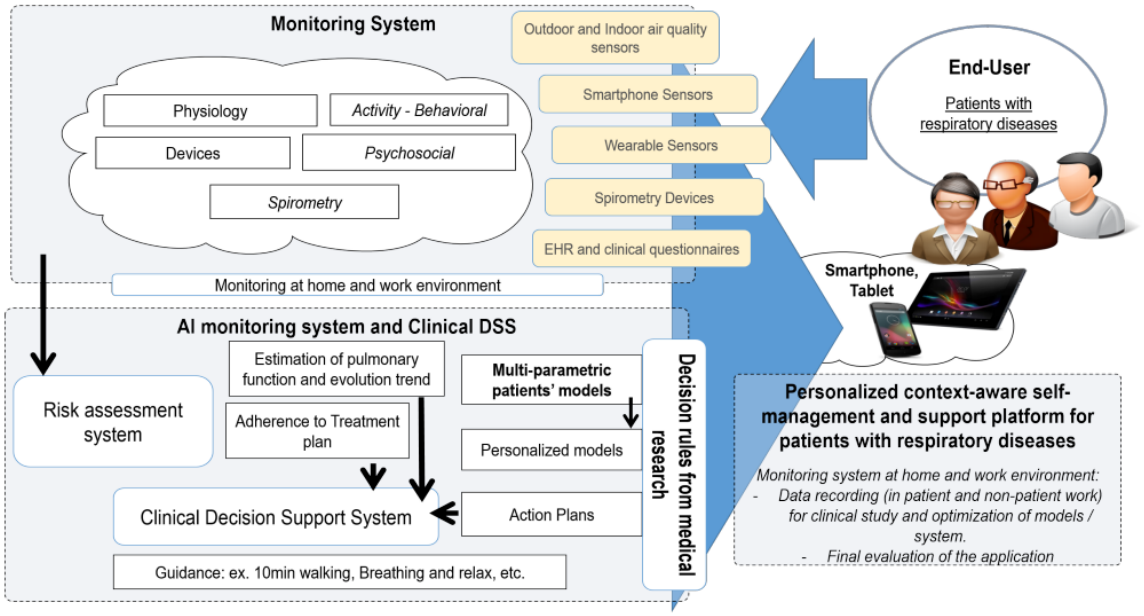


Figure 2.50 Architecture of the monitoring system [77].

Wen et al. [78] paper show the spatial-temporal variations in carbon monoxide at the street level and develop a pilot framework for the wireless sensor based on a real-time pollution monitoring system. The two main parts of the system are shown in Figure 2.51. In this study, the first component was the deployment of wireless sensors, which included 44 sensor nodes, 40 transmitter nodes, and four gateway nodes. A CO sensor, wireless connection module, and signal processing module were included in each sensor node. To capture realistic human exposure to road pollution, all sensors were placed at a height of 1.5 m on lampposts and traffic signs. 1.5 kilometres of Taipei City's Keelung Road constitutes the study area. The other component is a monitoring platform with a map-based interface for manipulating and visualizing the sensor data over time and space. Spatiotemporal air pollution trends were compared using a comprehensive real-time street-level monitoring infrastructure.

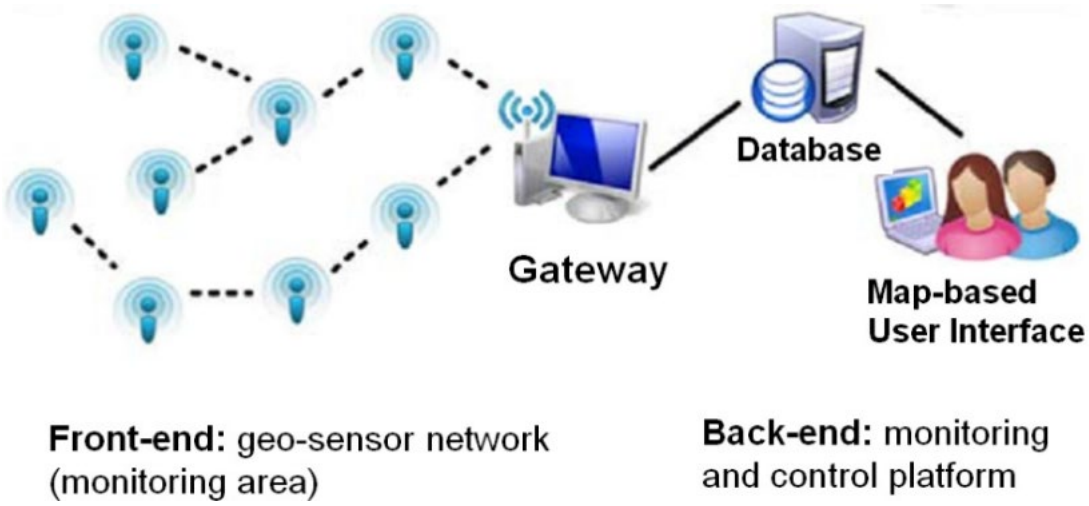


Figure 2.51 Proposed automatic microscale air quality monitoring system [78].

Knoll et al. [79] built a sensor network with a higher sensor density to increase the air pollution measurement density, and the data were then put into dispersion models, which can only provide rough estimates of the results. This study compared emerging network technologies such as LoRa, Sigfox, and NB-IoT. Air pollution values can be immediately monitored using a denser sensor network. Measurements were performed in the city of Graz using an STM32 Nucleo board with an expansion board equipped with a LoRa SX1272 transceiver. Each communication payload was 16 bytes long and included a message counter for verification. On the university campus, the sink node was placed outside a window on the third floor, and additional nodes for data gathering were positioned at a height of 1 m. Three distinct technologies were used to perform these measurements. The required characteristics, such as extended range and low power consumption, are brought about by LoRa and Sigfox. In contrast, the most recent technology, NB-IoT, is heavily influenced by mobile communication.

Chen et al. [80] proposed an open framework for participatory $PM_{2.5}$ monitoring in smart cities, shown in Figure 2.52, deploying more than 2,500 devices across Taiwan and 29 other nations. This study also examined problems with low-cost particle accuracy to determine the most reliable sensors.

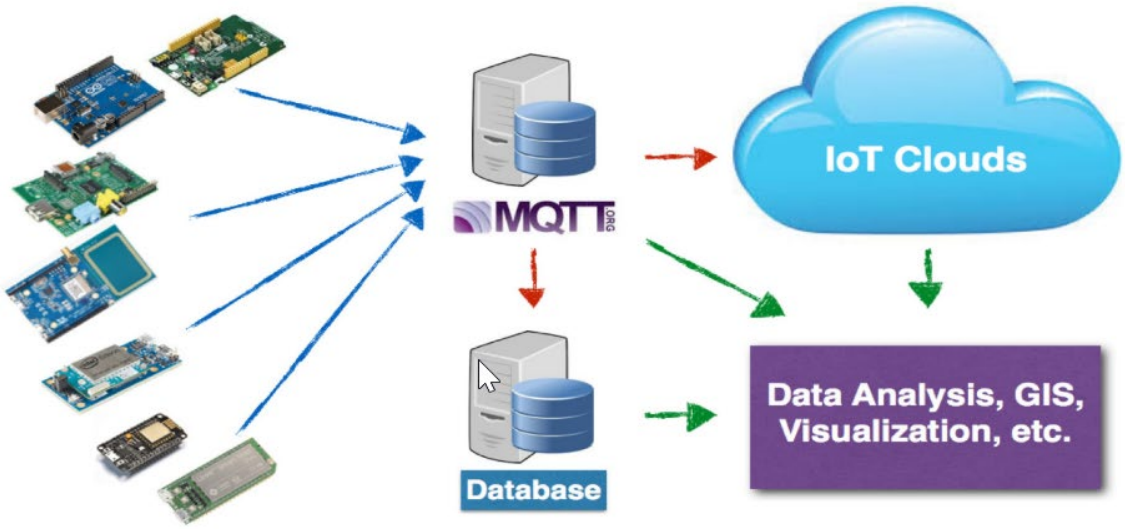


Figure 2.52 Proposed participatory urban sensing framework [80].

Until May 2017, it was one of the most significant deployment initiatives for monitoring $PM_{2.5}$. The framework's open system architecture, founded on the ideas of open hardware, open-source software, and open data, is its unique feature. The framework explores the low-cost particle sensor accuracy problem with a thorough set of comparison assessments to identify the most dependable sensor. It can create a successful ecosystem for participatory urban sensing of $PM_{2.5}$ particles by collaborating closely with governmental agencies, industry partners, and maker groups. Based on the success of its implementation, the network offers data services to enhance environmental awareness, prompt on-demand solutions, and support future government decisions. Researchers have provided several data services to improve environmental awareness, fast on-demand solutions, and support future government legislation according to deployment achievements. The proposed network is highly expandable and resilient [80].

Jamil et al. [81] proposed an innovative mesh network for air pollution monitoring in Smart Cities to deploy the sensor nodes based on the movement of cars and buses, as shown in Figure 2.53. Public transportation buses' sensors capture air pollution particles, including gases, smoke, and other contaminants. A routing algorithm was developed after passing through fixed nodes scattered across the city. The information collected regarding air pollution is the integration of mobile sensor nodes into a wireless sensor network. LTE-M is a future-oriented

technology for low deployment in public transport. The proposed network is based on power consumption.

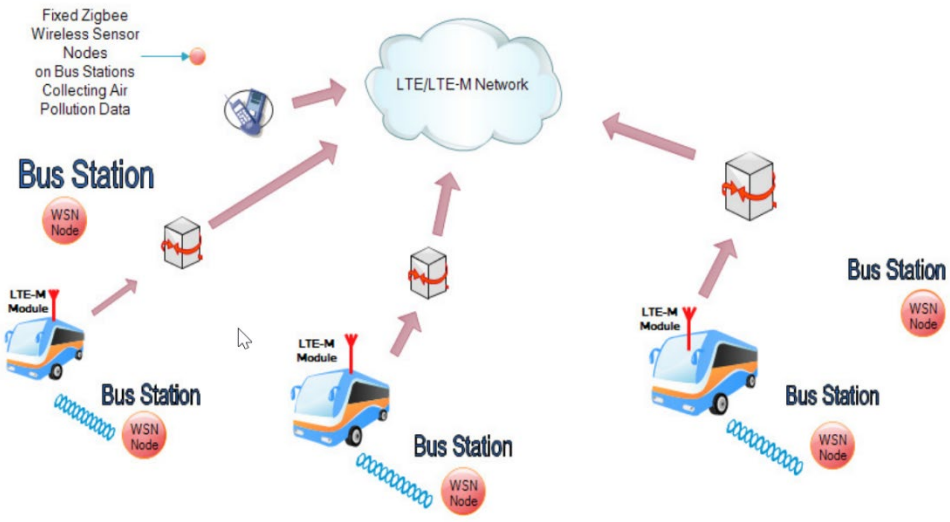


Figure 2.53 LTE-M-based outdoor network architecture [81].

Chowdhury et al. [82] proposed an energy-efficient machine learning-based sensor duty-cycling method for sensor hubs to receive data from air pollution sensors. A support vector algorithm was implemented to predict the missing samples. This study also focuses on the energy-efficient optimum duty cycle methodology and duty-cycling methods. Figure 2.54 shows the mobile sensing nodes that spend energy on sensing, processing, and communication. The primary approach of the research is to sense $PM_{2.5}$ concentrations using different sensors and an Arduino kit. Researchers have presented an optimum duty-cycling method for energy-intensive sensors to improve energy efficiency. The results showed a temporal correlation between $PM_{2.5}$.

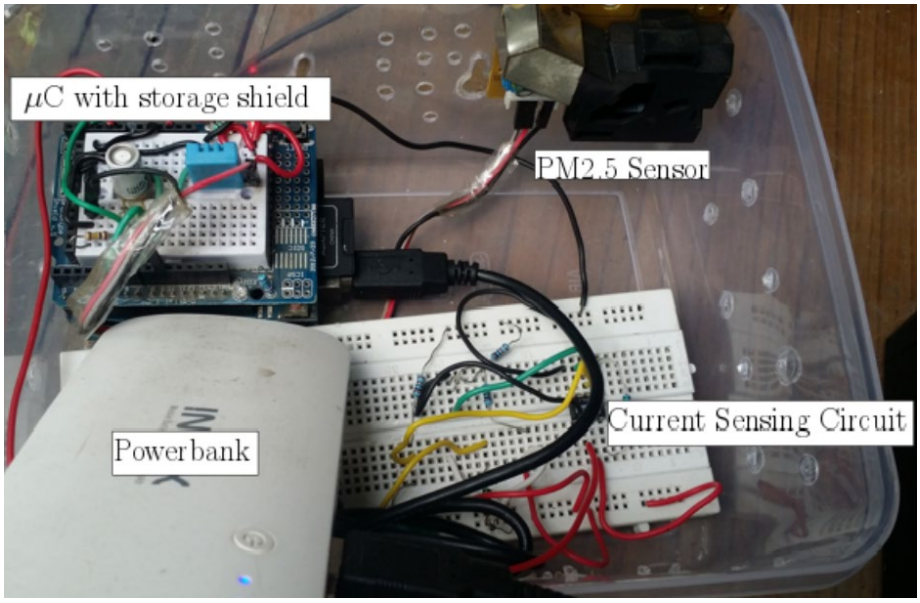


Figure 2.54 Experimental setup [82]

Botero et al. [83] provided a method for data reduction using data fusion from several sensors for the same variables and dynamic sub-sampling of the measured variables. Data scaling with a range of variables in mind. Data reduction is used to conserve energy, shorten transmission times, maintain channel availability, and provide a free storage space. Before transferring the data to the gateway node, they were processed using a low-power microprocessor at the sensor node. A low-cost monitoring station with a network diagram that includes environmental, PM, gas, electromagnetic radiation, and inertial sensors was used to validate the method, as shown in Figure 2.55. With a strong Gaussian noise component, the proposed approach was used for PM measurements for online down sampling. The creation of low-cost sensors and development systems with high processing capacity are combined with new communication architectures specifically geared towards LoRa.

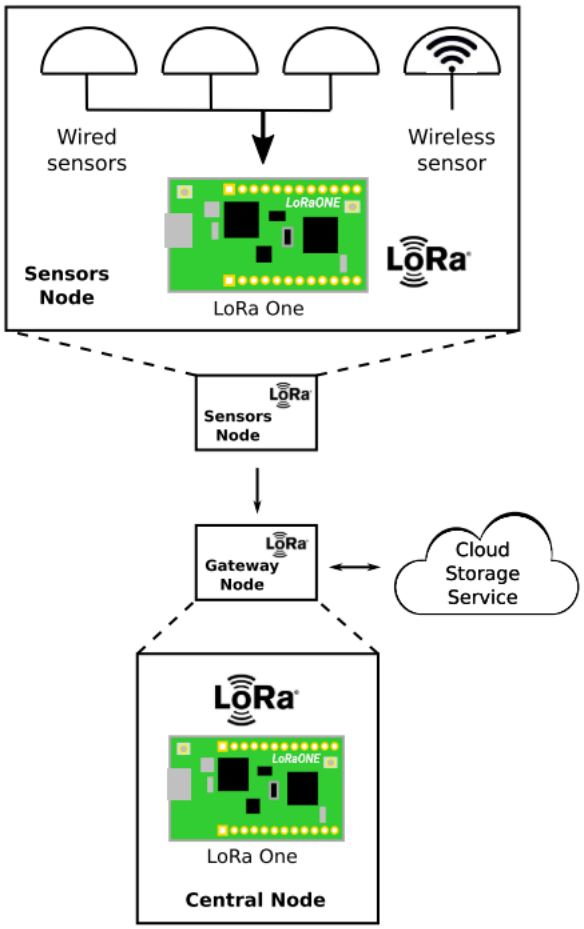


Figure 2.55 Block diagram of low-cost environmental monitoring system based on LoRa for WSN [83].

Tzortzakis et al. [84] Describe the network architecture and LoRa protocol implementation of a wireless sensor network that monitors temperature, humidity, brightness, carbon monoxide, methane, alcohol, and smoke sensors and transmits the collected data to a base station. The base station uses the GPRS to gather and upload data to the cloud. The designed system covers the entire city by using a few base stations. The data collected from the sensor nodes are translated into analogue readings. This network architecture, shown in Figure 2.56, was successful, and there was no packet loss between the connectivity's of the system. The base station uses GPRS to gather and upload data to the cloud.

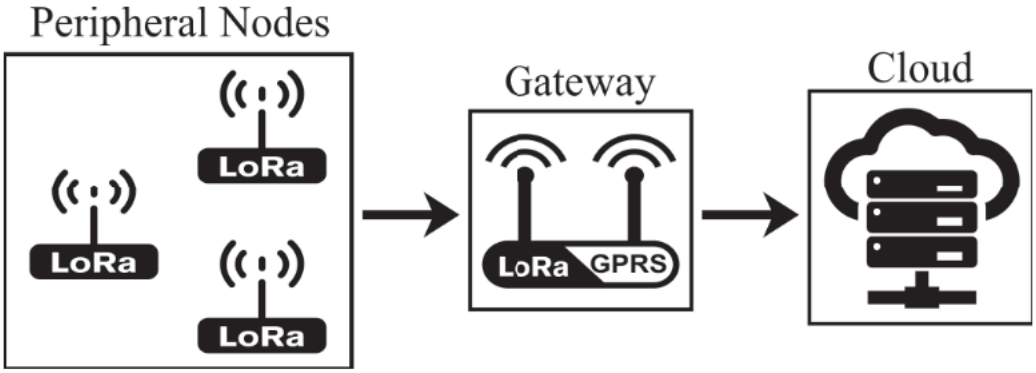


Figure 2.56 System Architecture [84].

2.2.2 Cellular Networks

Hu et al. [85] designed and evaluated the low-cost participatory sensing system combined with sensor units, smartphones, cloud computing, and mobile applications called HazeWatch, Figure 2.57. This mobile application helps visualize and estimate air pollution exposure based on mobility patterns.

Figure 2.57 HazeWatch System Architecture [85].

This paper makes three contributions: Develop web-based tools and mobile apps for the visualization and estimation of air pollution exposure tailored to individuals; 1) architect,

prototype, and compare multiple hardware devices and software applications for collecting urban air pollution data with high spatial density in real-time; and 2) conduct field trials to validate the system and show that it produces exposure estimates that are much more accurate than those of current systems. This information can be used globally, especially in nations with high pollution levels, to understand better the connection between urban air pollution and its effects on health.

Al-Ali et al. [86] The General Service Packet Ratio (GPRS) sensor system was created by fusing a fixed internet unit (which contains a microcontroller chip, air pollution sensor array, and GPRS modem) with a mobile data collection system, as shown in Figure 2.58. This device measured the number of pollutants such as CO, NO_2, and SO_2 in Sharjah, United Arab Emirates [86].

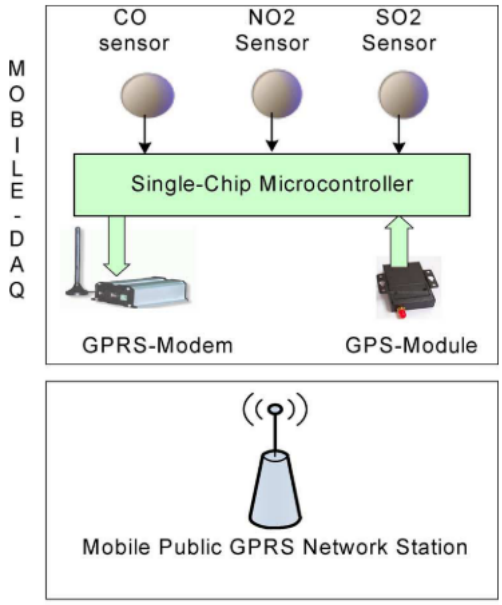


Figure 2.58 GPRS Unit [86]

The frame is then uploaded to the GPRS modem and sent to the pollution server across the open mobile network. The Pollution Server is connected to a database server that stores the pollutant level for later use by numerous clients, including environmental protection organizations, authorities in charge of car registration, and travel and insurance businesses. In real-time, Google Maps and the Pollution Server are connected to show the locations and amounts of pollutants in major cities.

Sun et al. [87] proposed that Environmental sensors and ZigBee wireless sensor networks are combined to create an intelligent indoor environment monitoring system (iDEMS), which uses the cloud to store and process environmental data in HBase. Hadoop MapReduce can calculate iDEMS for the HBase database for distributed computation. The architecture of the HBase database is shown in Figure 2.59. iDEMS was integrated with an intelligent control socket to improve indoor air quality and reduce concentrations [87].

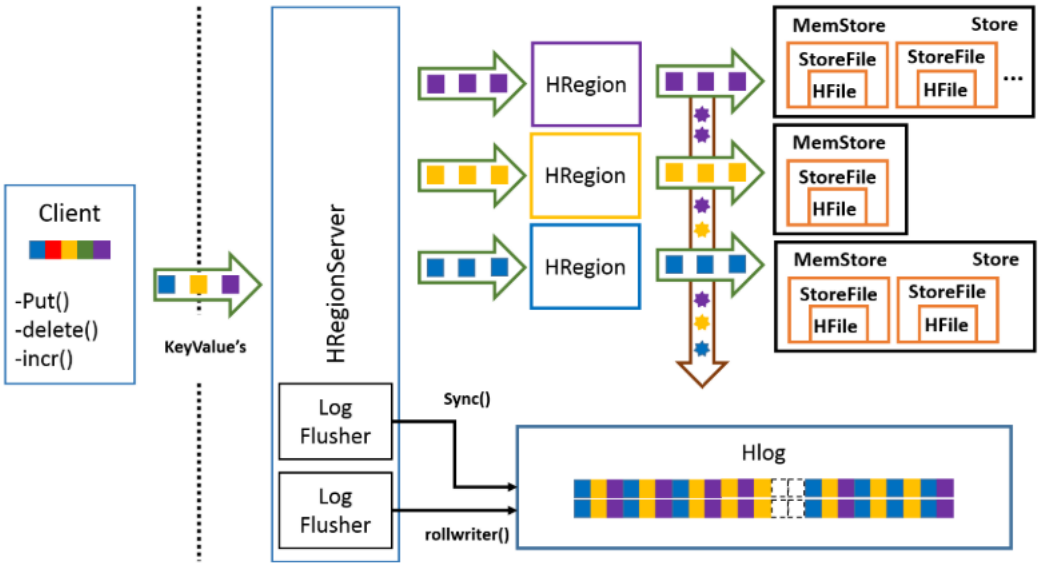


Figure 2.59 HBase service architecture [87].

Additionally, iDEMS includes an intelligent control socket to efficiently enhance the air quality of the indoor environment and lower the concentration of pollutants so that people are kept away from contaminants. It reduces the batch processing time in the converter portion by 60% without compromising services. Users can view real-time air pollution statistics; store data for days, months, and years; and track all changes in air pollution parameters using historical data.

2.2.3 Vehicular Networks

Cordova et al. [88] proposed a network for monitoring vehicular air pollution in the UK, shown in Figure 2.60. Technological advances such as wireless sensor networks and complexboard-based tools allow for capturing and analysing air pollution data [88]. This study provides a rationale for the novel approach and the empiricalinstruments used. This study investigates the

fuel composition and the combustion conditions that show the emission of CO_2 water vapor due to combustion. This study also focused on greenhouse gas emissions. Vehicular air pollution is monitored with a crossbow MicaZ platform and a quintox chemical gas analyzer.

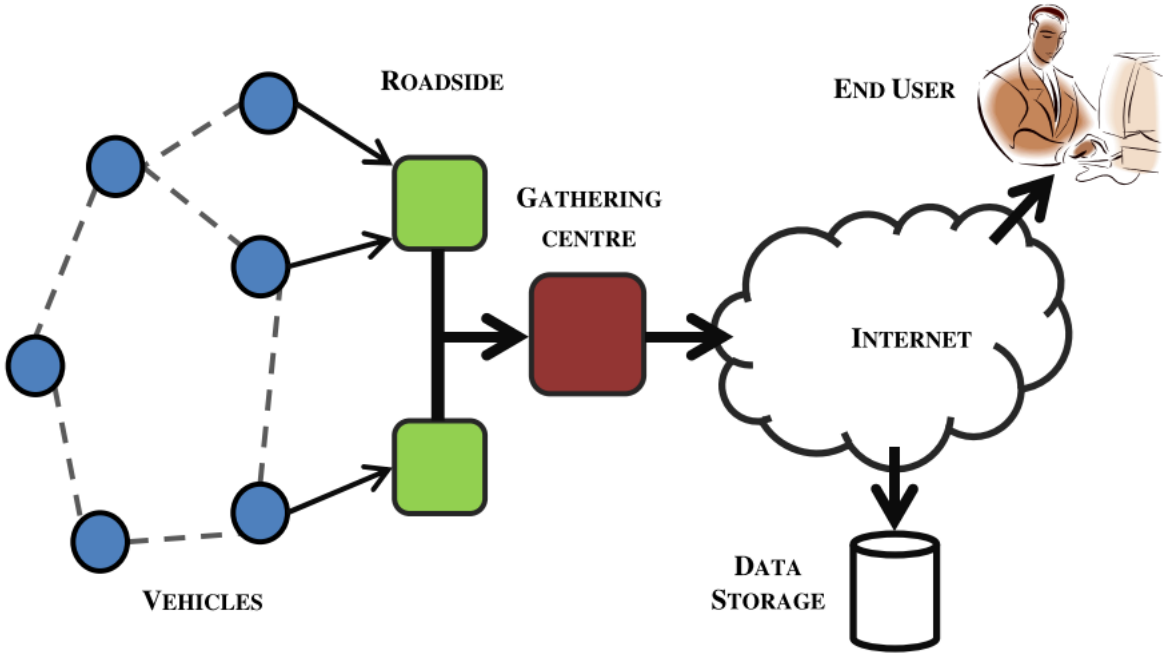


Figure 2.60 Overview of the proposed system [88].

Yi et al. [89] addressed the problems with earlier approaches, a contemporary methodology that gives statistics on air pollution was devised. The high spatial and temporal resolution presents management, configuration, and expansion challenges. By implementing the proposed Modular Sensor System (MSS) architecture shown in Figure 2.61 System Architecture of MSS sensor nodes
[89]. Universal Sensor Interface (USI) and modular design in a sensor node to monitor CO and NO_2, this work seeks to address these concerns. A small MSS sensor node was developed and assessed. It is compatible with different Wireless Sensor Networks and offers extendable sensor modules with plug-and-play capabilities (WSNs). According to the evaluation results, the MSS sensor nodes can detect low-concentration air pollution with a high energy economy and good data accuracy. They can also quickly adjust to dynamic reconfiguration.

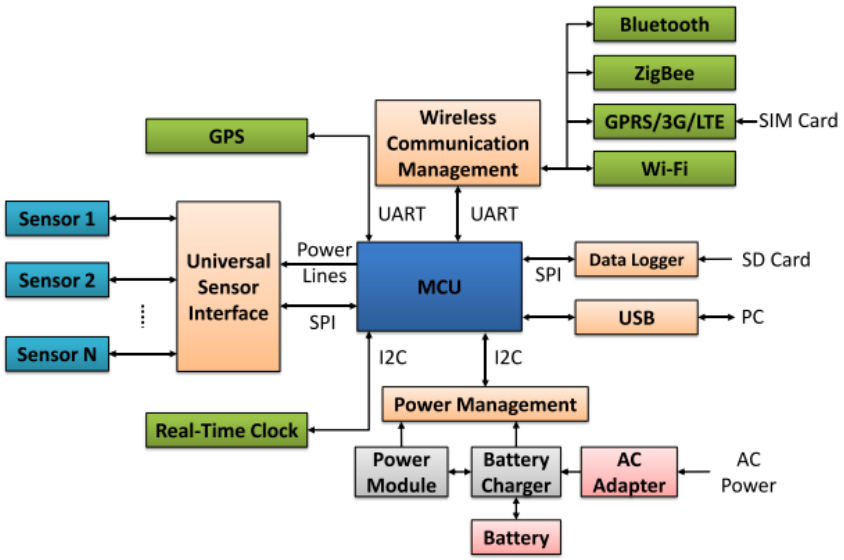


Figure 2.61 System Architecture of MSS sensor nodes [89].

Ngom et al. [90] offered a low-cost approach based on wireless sensor networks for monitoring Dakar's air quality in real-time. This system incorporated CO, CO_2, PM_{10}, $PM_{2.5}$, and PM_1 sensors. This study aims to develop a hybrid measurement kit that can be fixed as landmarks or mobile to analyze air quality using more precise measurements. Mobile technology is built into vehicles that move throughout the city. The implemented kit can send collected data to a web application for visualization using the LoRa protocol or cellular network. Figure 2.62 shows the acquisition architecture of detection, power supply, location, processing, and transmission units. The proposed measurement kit can be integrated into each moving vehicle or fixed as a landmark. Different applications can exploit other datasets.

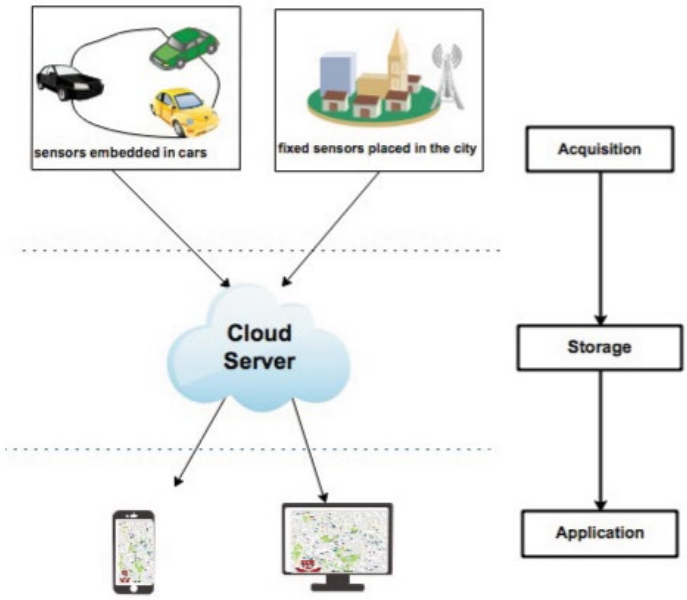


Figure 2.62 Global System [90].

Pavani et al. [91] comprehensively review urban air pollution monitoring using wireless sensor networks. This paper focuses on the advancements in microelectromechanical sensors (MEMS) and wireless sensor networks (WSN) to monitor primary pollutants such as CO_2, CO, O_3, So_2, VOC, and particulate matter (PM) [91]. This study covers indoor and outdoor pollution using WSN, as shown in Figure 2.63 Architecture of WSN-based indoor air quality monitoring system.. Researchers have developed routing algorithms and explained the comparison of these algorithms.

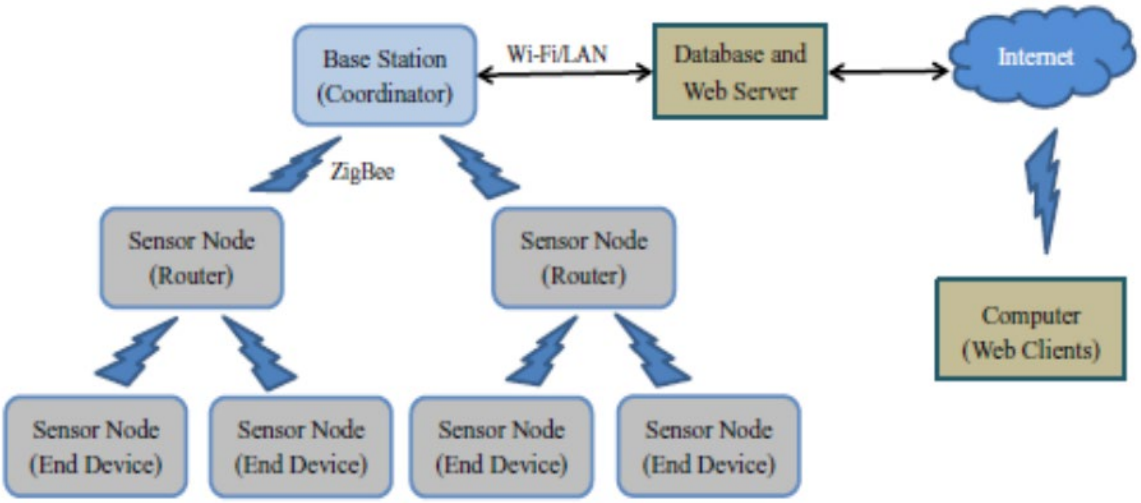


Figure 2.63 Architecture of WSN-based indoor air quality monitoring system.

Suriano et al. [92] developed a portable sensor system based on solid-state gas sensors that was created and put into use to track air pollution [92]. This system used commercial gas and carbon nanotubes, p-type, and nanotechnology sensors stacked in an array for multiple sensing and parameter devices. It measures the concentrations of NO_2, SO_2, and CO using real-time monitoring methods and the continuous monitoring of hazardous air contaminants. According to the results, the proposed system can be used as a new gas-sensing paradigm for environmental measurements and provides good accuracy.

2.3 Summary and Contribution

This chapter presents an overview of air pollution monitoring systems using different network technologies, specifically the currently used systems. Air pollution monitoring has been developed based on research and findings. It continues to evolve, where the needs are always ahead of the technologies, and new air pollution monitoring technologies and methods emerge quickly.

This chapter discusses Wireless Sensor Networks (WSNs) and their applications in monitoring air pollution. WSNs consist of small, battery-powered sensor nodes equipped with sensors, microprocessors, and communication radios. These nodes are used for various purposes, including pollution monitoring. The text highlights the global issue of air pollution and its adverse effects on health and the environment. City councils have started using pollution measurement units to address this problem, but these systems are often costly and require significant maintenance.

The chapter proposes a low-power-consumption network system using LoRa technology to address these issues. This network includes four sensors connected to LoRa modules, allowing for street-level data collection of particle matter, temperature, humidity, and CO_2 levels. The aim is to create a more efficient and cost-effective solution for monitoring air pollution in urban areas.

In addition to its focus on low-power LoRa-based sensor networks for air pollution monitoring, the chapter underscores the critical importance of addressing air pollution as a global concern.

The impact of air pollution on public health, agricultural production, urban development, and overall quality of life is emphasized. Here are some further points to consider:

- **Health and Environmental Consequences:** As noted by the World Health Organization, air pollution poses a severe threat to human health. High levels of pollutants in the air can lead to respiratory illnesses, cardiovascular problems, and even premature death. Furthermore, pollution negatively affects ecosystems, causing harm to plant life and wildlife.

- **Current Monitoring Solutions:** The chapter mentions that city councils have already addressed air pollution by implementing monitoring systems. These systems are likely essential for tracking pollution levels, understanding trends, and developing strategies to mitigate pollution's impact. However, they are large, power-hungry, and expensive to maintain.

- **The Promise of LoRa Technology:** LoRa (Long Range) technology is highlighted as a promising solution due to its ability to transmit data over long distances while consuming minimal power. This makes it particularly suitable for creating an efficient, cost-effective air pollution monitoring network.

- **Sensor Data:** The chapter specifies that the proposed network comprises four sensors capable of measuring particle matter, temperature, humidity, and CO_2 levels at various elevations. This diverse range of data points allows for a comprehensive understanding of air quality in urban areas.

- **Sustainability and Cost-Efficiency:** By adopting low-power LoRa technology and reducing the need for extensive maintenance, the proposed network aims to make air pollution monitoring more sustainable and cost-efficient. This approach aligns with the broader goal of addressing environmental issues while optimizing resource allocation.

- **Future Implications**: The deployment of such networks not only benefits current pollution management efforts but also has the potential to provide valuable data for future urban planning, environmental policies, and public health initiatives. It can help authorities make informed decisions to improve air quality.

In summary, the chapter highlights the urgency of addressing air pollution, the limitations of current monitoring systems, and the potential of LoRa-based sensor networks to offer a more efficient and sustainable solution for monitoring and mitigating air pollution in urban areas.

Chapter 3
Network Performance of Proposed Air Monitoring Network

This chapter focuses on the proposed network's simulation process and performance. Before network deployment, the designed network requires a simulation to estimate the network performance to reduce issues and save money. A LoRa sensor network was designed for this study, and the network performance was monitored. Network simulation is essential for better deployment results and is adequate for determining network efficiency.

3.1 Introduction

New technologies are being proposed using wireless technology, which is developing rapidly. New, unproven protocols cannot be introduced into computer networks on a significant scale because of the need for certainty in their likelihood of success. Therefore, analytical modelling or simulation tools are used to test new protocols and schemes. The results of analytical modelling are only sometimes accurate in terms of energy, memory, and processing power. Experimentation and mathematical modelling are used in communication network research to demonstrate viability and set limits on the anticipated performance of computer networks [93].

ns-3 is a research- and education-based simulator used for the research community [94] and is based on the ongoing contribution to developing the new model, debugging, and maintaining existing projects. ns-3 has the following policies that encourage people to contribute to ns-3, as they have for ns-2:

- Open-source licensing based on GNU GPLv2 compatibility.
- Wiki [94]
- Contributed Code page, similar to ns-2's popular Contributed Code page.
- Open bug tracker

ns-3 documents are provided from several resources, but detailed information is available at http://www.nsnam.org [94]. ns-3 also has Wikipedia links related to the ns-3 system architecture at http://www.nsnam.org/wiki/ covering troubleshooting guides, third-party contributed code, and papers. Several complex software managing the organizations and

changes to the code and documentation are available for ns-3 and can be used with different software in one system [94]. Most users are aware of the concurrent version system (CVS). The ns-3 programs or projects use mercurial as their source code management system.

Once the source code is downloaded to a local system, it requires compiling the source to produce reusable programs [94] using source code management tools. The most effective tool is MAKE[2], which is challenging to use in extensive and highly configurable sources but can be used with different alternatives [94]. Python has recently entered the system to build codes for practical applications. The operating system uses the waf command to create ns-3 projects or source code. Python-based systems are an effective new-generation method but are optional to learn and use in an existing process [94].

ns-3 uses C++ and Python as the scripting languages. Most of the ns-3 API[3] is available in Python, but the models in ns-3 can be written in C++ or Python [94]. If the user knows the C++ and object-oriented concepts, it will be effective and suitable for building projects correctly. It will take little time compared to the new users of C++. Several sources are available online to learn ns-3 coding, and various examples are available for multiple applications [94].

For example, in the ns-3 tutorial, various tools are available for development in the GNU "toolchain" [94]. A software toolchain is a set of programming tools used to build a user environment, and users can use the waf function. In ns-3, users can work in Linux- or Linux-like environments [94]. It can be run in a Windows environment using a virtual machine or the Cygwin software.

Before network deployment and installation of sensors in the real world, the ns-3 simulator is influential in determining how the designed network will perform using various parameters. It does not require the installation of computers or routers to communicate to create different topologies. A network simulator was designed for experimental purposes. The ns-3 discrete event network simulator was used to overcome these drawbacks. ns-3 helps to create virtual nodes (for example, dummy sensor nodes) using Helper classes that allow the installation of

[2] MAKE - command used to make directory in Ubuntu
[3] API - Application Programming Interface

devices, internet stacks, and applications. Using ns-3 makes creating PointToPoint, wireless, and CSMA connections between nodes easier. As in the designed network, wireless connections were made between the sensor nodes and gateway for the data transmission and communication process. There are some reasons for using ns-3 in the designed network are explained below:

- **Tracing of the nodes**–In ns-3, the routes of the sensor nodes can be traced to help the users check how much data is sent or received. Trace files were generated to monitor the transmission activities.
- **NetAnim** stands for the Network Animator, which is known as the version of how the network will look in reality and how it will transmit data from one device to another.
- **Pcap files**- In ns-3, Pcap files are generated to know the information about all the packets sent and received. This can be seen using Wireshark software.
- **GnuPlot**- GnuPlot is used for drawing plots from the received data in the trace file. It provides more accurate graphs than other graph-making tools do.

3.2 Comparison Between Simulators

This section introduces numerous network simulators considered for WSN's performance [95], such as ns-2, ns-3, OMNET++, OPNET, Tossim, and J-Sim simulators. Only ns-3 has built-in LoRa wireless communication methods. Different network simulators are discussed, but ns3 is used for this research to perform network capability analysis. All the network simulators are used for networking, but the NS3 simulator has built-in Lora modules. To install Lora modules, ns3 has a *.nsrc file that needs to be implemented in ns3. In ns3, coding can be done in Python and C++; for this research, coding has been done in C++ because Python doesn't have LoRa modules. With Python, ns3 cannot be effective for LoRa networks. These are the reasons for using the ns3 simulator. Various researchers have used these simulators to research and validate their new theories. A comparative analysis is presented in Table 3.1.

Table 3.1 Comparative Analysis of Different Network Simulators

Simulator Name/year	Operating System	Mobility support	License	Language	GUI Support
OPNET (1986)	Windows 7 or above	Yes	Commercial	C and C++	Yes
TOSSIM (2000) [96]	Linux, MS Windows using (Cygwin)	Yes	Open Source	C++, nesC-TinyOS, Python	Yes
J-Sim (2001) [97]	Linux, Free BSD, macOS, MS Windows	Yes	Open Source	C++ and C	Yes
OMNET++ (2011)[98]	Linux, Free BSD, macOS, MS Windows	Yes	Open Source	C++, C#, JAVA	Yes
ns2 (2012) [99]	Linux, Free BSD, macOS, MS Windows	Yes	Open Source	C++	Yes
ns3 (2015) [100]	Linux, Free BSD, macOS, MS Windows using (Cygwin)	Yes	Open Source	C++, Python	Yes

3.2.1 Optimised Network Engineering Tool (OPNET)

OPNET is a discrete-event network simulator that supports C++ as the programming language. OPNET was proposed in 1986, is now well-established and available as a commercialsimulator [101], and is available online for universities and researchers. OPNET provides network hardware, such as antennas and transceivers, to monitor and execute several scenarios simultaneously. It allows users to develop models and graphs using a graphical user interface and provides numerous features, such as designing protocols, networks, devices, and applications [101].

3.2.2 TOSSIM

TOSSIM [102] is a discrete event simulator for TinyOS wireless sensor networks. UC Berkeley developed TOSSIM and TinyOS. TOSSIM and TinyOS were developed at UC Berkeley. In contrast to the network simulator, TOSSIM captures network behaviour and interactions at the network bit level rather than at the packet level. TinyOS is an operating system for sensor networks that uses the least energy and communication on so-called "motes," computing, and remote-sensing equipment. TinyOS sensor networks frequently consist of many hardware components and simulate motes through software modelling and hardware abstraction.

3.2.3 JavaSim (J-Sim)

Building quantitative mathematical models and analyzing them for experimental reference data are both possible using the Java-based simulation system known as J-SIM. An autonomous component programming model was constructed. Ordinary differential equations (ODEs), partial differential equations (PDEs), implicit equations, integrals, summations, discrete events, and procedural codes can be mixed [103] - Additionally, J-Sim imports and exports models in SBML and CellML archived formats. J-Sim is an open-source network simulator that allows models to be created using Java programming. It can be used with Linux, Mac OS X, Windows XP, or Vista.

It offers a dynamic thread execution framework for process-driven simulation that runs in real time. The J-SIM simulation engine extends the Worker Pool class and tracks every Worker Thread activity. It maintains a virtual system time monitored globally and is proportional to real-time. Use all the Internet integrated/differentiated/best-effort service protocols. This implementation has three implications: access to all Internet protocol classes, abstract classes, and component hierarchy.

3.2.4 Objective Modular Network Testbed in C++ (OMNET++)

OMNET++ was released to the public in 1997, and several researchers are still working on it [98] for wireless sensor networks. Compared to ns-2 and ns-3, OMNET++ is not considered a network simulator but is a discrete event-based network simulator mainly applied to the

network's domain. OMNET++ has an in-built INET package withnetwork protocols for simulation processes, such as the Mobility Framework and Castalia [104]. To understand the atomic behaviour of the network simulator, OMNET++ has simple modules for network models, and defining a network protocol is the easiest in OMNET++. However, they are compound modules if users want to create multiple link modules. The network simulation in OMNET++ can be set up in the NED (network description) language of OMNET++, as shown in Figure 3.1.

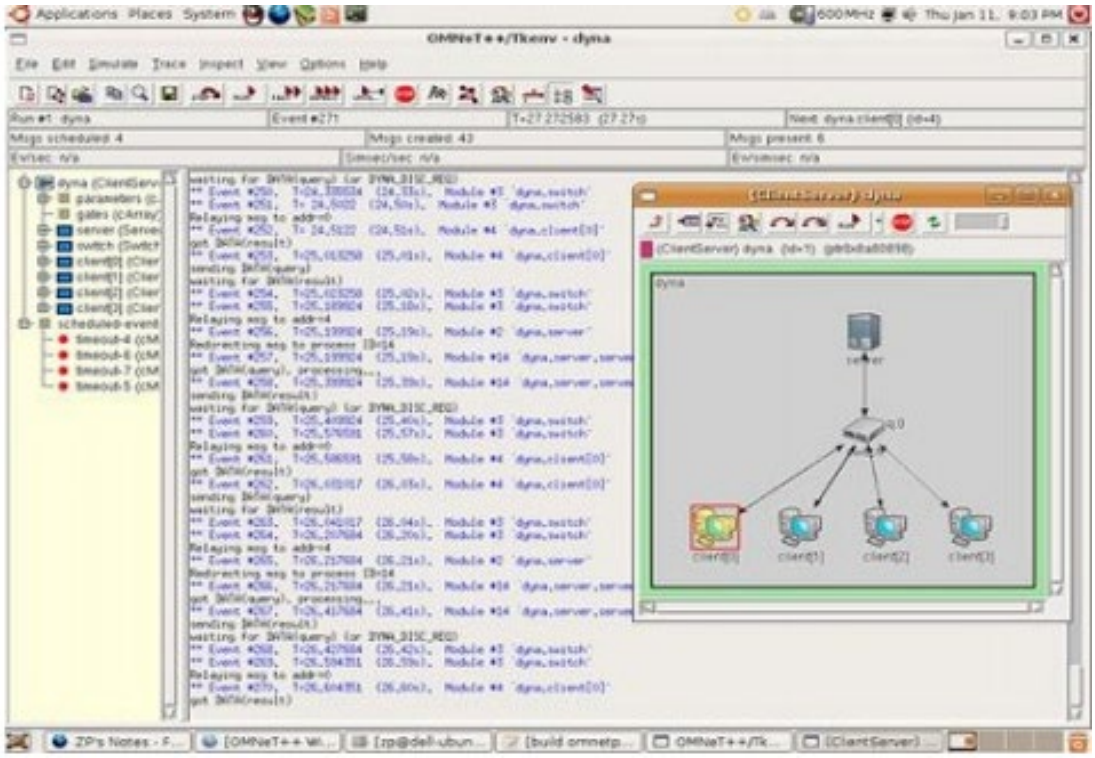


Figure 3.1 OMNET++ interface [104]

3.2.5 Network Simulator 2 (ns-2)

Network Simulator-2 (ns-2) is an open-source discrete-event network simulator where users can use different wired or wireless network topologies containing routers, links, and shared media. The scripting of ns-2 supports only the C++ language, provides a simulation interface using OTcL, and is required to write scripts in OTcL. The main program can be included along with the main parameters. ns-2 uses the network animator (NAM) and contains features that allow users to forward, pause, stop, and play the simulation for the graphical view shown in

Figure 3.2 [105]. The physical activity of the network can be determined using an event scheduler.

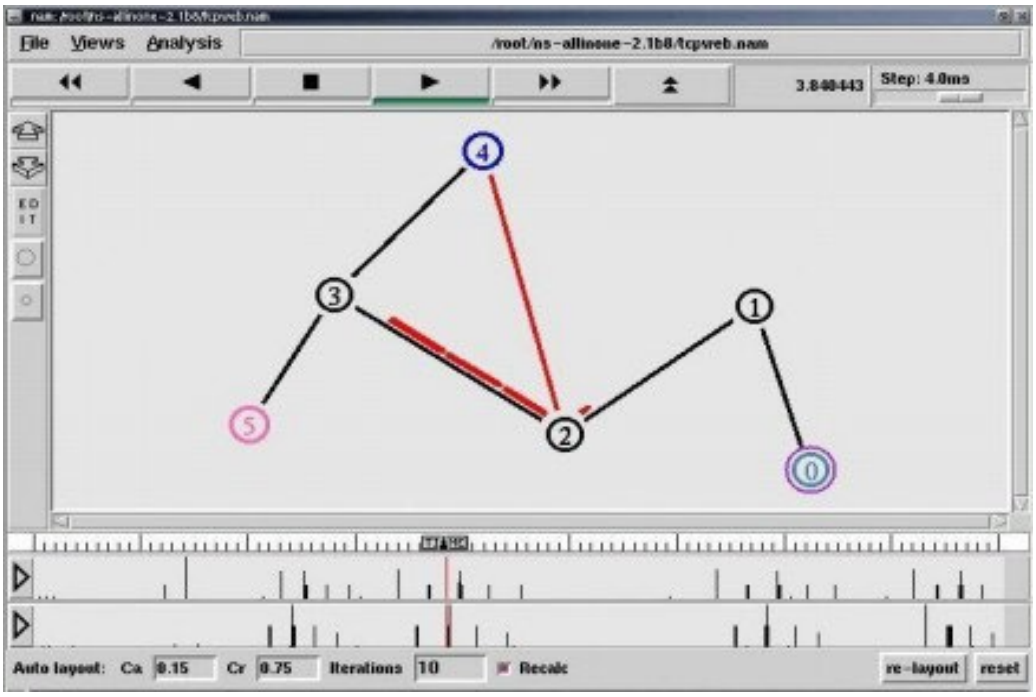


Figure 3.2 ns-2 interface [105].

3.2.6 Network Simulator 3 (ns-3)

The ns-3 simulator is an open-source discrete-event network simulator for research and education. The first ns-3 program was built in 2006 to monitor network performance and stability. ns-3 has an in-built model to explain packet flow during transmission and performs the simulation process for different models using several libraries linked with external libraries. The NS-3 simulator has a single graphical user interface (GUI) to perform various activities such as network animation, data analysis, and data visualization [106]. ns-3 users support C++ and Python as scripting languages on the command line. ns-3 is mainly used in Linux systems, although the support for FreeBSD, Cygwin (for Windows), and ns-3 simulator interfaces are shown in Figure 3.3.

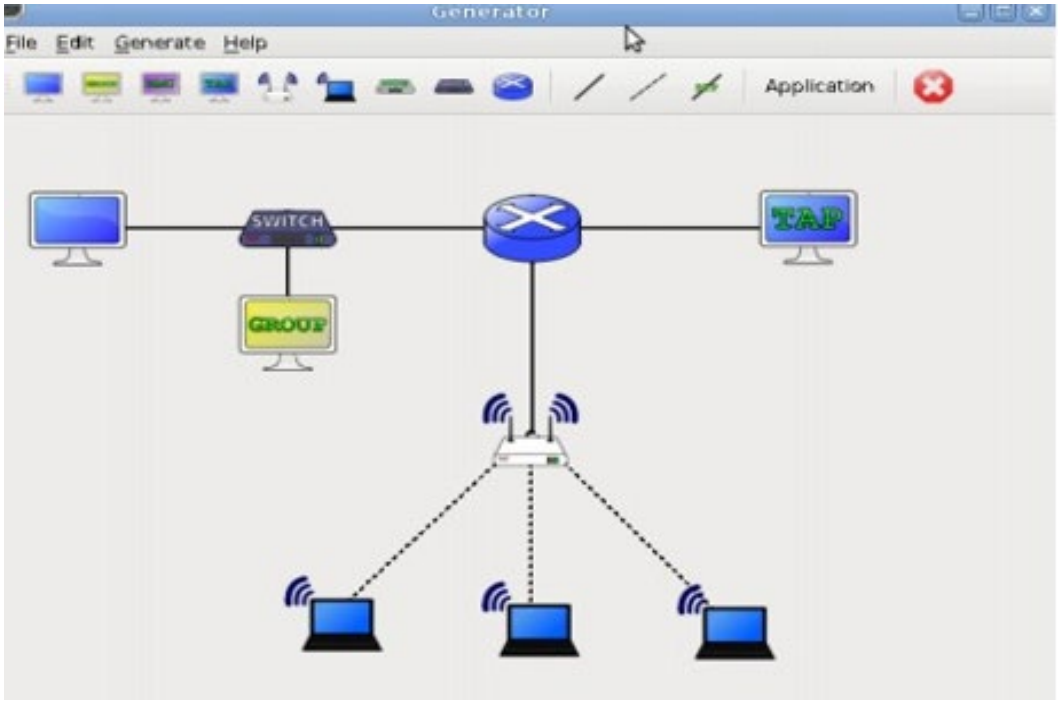


Figure 3.3 ns-3 interface [106].

3.3 Simulation Process

The simulation requires a diversity of steps to simulate the designed network, as explained below:

- **Topology Creation** includes the number of end devices used in the network and is instantiated as a collection of node objects. For each node, a mobilitymodel can be implemented to represent the node's position and how it changes over time.
- **Models**: A specific protocol stack is installed on the set of nodes. This is performed using the helpers and classes installed on the nodes using the various objects required in the OSI stack. This is a practical step for creating, sending, receiving, and interpreting packets belonging to the protocol.
- **Configuration**: The suggested models of the protocols can be configured using specific parameters and links between the nodes. This process is performed by subscribing multiple node PHY layers to the same channel objects.
- **Execution**: When the simulation starts, the simulator class processes the events and executes function calls. Trace source files and save the data into an appropriate data

structure or a specific file during the simulation process.

- **Performance Analysis**: After the completion of the simulation, the gathered data can be analyzed and visualized.

To perform the simulation process for the LoRa network, some parameters are shown in Table 3.2 to consider while setting up the network:

Table 3.2 Network Parameters

Parameter	**Description**
Simulator	ns-3.26
Topology	Fixed topology
Number of nodes	Topology1: One gateway, one server, 4 LoRa nodes and 16 dummy nodes Topology 2: One gateway, one server and 16 LoRa and 48 dummy nodes Topology3: One gateway, one server, 4 WSN nodes and 16 dummy nodes Topology 4: One gateway, one server and 16 WSN and 48 dummy nodes
LoRa Data Rate	1000 Mbps
Propagation Model	Range Propagation Loss Model
Channel type	Multi-Model Spectrum Channel
Packet size	1024 bytes
Packet rate	1 kb/sec
Traffic type	CBR[4], UDP[5]
Simulation Time	1800 sec

[4] Constant Bit Rate
[5] User datagram Protocol

To perform the simulation process for the WSN and LoRa network, the following parameters must be considered while setting up the network.

- The designed fixed area was 1000 m × 1200 m, where the WSN sensor nodes and LoRa nodes were placed according to the distance between them and the gateway.
- The network server validates the authenticity and integrity of end devices, deduplicates uplinks, selects gateways using EUI for downlinks, and sends ADR commands to optimize the data rate of the devices.
- The gateway manages secure gateway connections and configurations. The LoRa Gateway supports the legacy UDP forwarder and new LoRa Basics™ Station protocol, enabling remote updates and design, whereas the WSN gateway manages the CBR connections and configurations.

LoRa node data from the dummy nodes are sent to the gateway. Dummy nodes generated dummy pollutant values for particulate matter, temperature, CO_2, and humidity. The network server receives the pollutant values via the LoRa gateway from the LoRa sensor nodes. Dummy sensor nodes were assigned maximum and minimum values to generate randomly, as shown in Figure 3.4.

```
std::cout << "Dummy Nodes (Network id):  "  << m_node->GetObject<Node>
()->GetId () << " sending to Sensor nodes with Temprature :  " <<  m_random->GetInteger
(70, 150) << " in Celsius"  <<  " humidity : " <<  m_random->GetInteger(20,80)  << " g/kg"
 << " CO2: "  <<  m_random->GetInteger(1,15)  << " parts-per-million (ppm)"  <<  " DUST: "
 <<  m_random->GetInteger(20,45) << " µg/m3"  << std::endl;
```

Figure 3.4 Structure for Dummy Sensor Nodes

The network has two functions for sending dummy nodes to LoRa nodes after creating dummy values, as shown in Figure 3.5.

```
NS_LOG_FUNCTION (packet << dest << protocolNumber);

if ( m_txMachineState = GAP) {

//
// When we started transmitting the current packet, it was placed in
// m_currentPkt.  So we had better find one there.
//

m_channel->TransmitEnd (m_protocolNumber);

m_currentPkt = 0;

NS_LOG_LOGIC ("Schedule TransmitReadyEvent in " << mp_timer.GetSeconds () <<
"sec");

Simulator::Schedule (Seconds(mp_timer), &LoraNetDevice::TransmitReadyEvent, this);
}
```

Figure 3.5 Creating Functions for Dummy Sensor Nodes

The code shown in Figure 3.6 starts transmission from dummy nodes to LoRa nodes. When the simulation began, communication was initiated from dummy nodes. In the network design, the LoRa gateway and network server show a wired connection, and the LoRa gateway is set as the default. This cannot be changed; however, in.NetAnim file, the network does not show the wired links of LoRa and works with wireless connections.

```
if (m_txMachineState == READY)
 {
   {
    m_dest = dest;
    m_protocolNumber = protocolNumber;
    m_currentPkt = packet;
    clearchannelassignment (Ptr<Channel> c);
    TransmitStart ();
   }
 }
if (m_dummytrans)
SendFromDummynodes (packet, dest, protocolNumber);
Simulator::Schedule (Seconds(tEvent), &LoraNetDevice::NotifyTransmissionEnd, this);
NotifyReceptionEndOk (packet,protocolNumber);
else
SendFromDummynodes (packet, dest, protocolNumber);
```

Figure 3.6 Transmission Process of Dummy Nodes

3.4 LoRa Module

New LoRa modules were created for the simulation process using LoRaWAN networks [107]. The LoRa modules may explain how LoRa end devices (EDs) and gateways (GWs) behave in various physical and application layers. The collection of classes replicates the LoRaPhy and LoRaMac layer protocol stacks on the device [107]. The models for losses induced by structures, correlated shadowing, interference, and duty-cycle restrictions used other classes.

The work described in this chapter focuses on network implementations, and LoRa modulation was employed. In this model, the network server is considered a priority, and network modulation is the main part of the chapter.

3.4.1 PeriodicSender

A packet generator that generates zero-filled packets with various payload sizes is a part of the application layer class known as PeriodicSender [107]. It should be observed that link abstraction results in more realistic packet payload content. The transmission delay of random

packets can be estimated using the application transmission period m intervals [107]. An unexpected packet is scheduled for the event after transmission, which can be set as an attribute in the class. The packet transmission (LoRa technology) is forwarded to the LoRa MAC layer. The function then calls the packet transmission scheduled for the subsequent transmission an event [107]. The model or application will work and keep sending messages or packets until it is stopped using a function call.

3.4.2 LoRaMAC

The LogicalLoRaChannel-Helper object is used to track the available network channels on a LoRaWAN device, and the LoRaMac class simulates the MAC layer of the device to consider the duty cycle requirements [107]. In this class, sending messages from the upper layer is not the class's responsibility because it breaks down the duty cycle rule, puts them into the queue, and sends them at the appropriate time [107].

Record and categorize the duty cycles on the various sub-bands into the duty cycle logic from the class. For this purpose, the DutyCycleHelper class was developed, which has two subclasses, EndDevice LoRaMAC and GatewayLoRaMAC. EndDeviceLoRaMAC defines the class for the models created, affects the network's Eds, and manages the state of the PHY layer [107]. To handle this properly, the radio must be awakened from sleep, and a receiver window must be opened. The model will be simple if the current approach supports Class A devices and implements the behaviour in subsequent iterations. To implement specific behaviours in a subclass, inheritance is used [107]. The Gateway LoRaMAC class is different from EndDevice LoRaMAC and implements a simple, forward-only MAC layer, and these classes are responsible for interpreting MAC commands.

3.4.3 LoRaPhy

The physical layer of the Semtech LoRa device was modelled using LoRaPhy Class. The LoRa chips found in the EDs and GWs, SX1272 and SX1301, were accurately simulated by this class. This class delivers a message to the channel class from the MAC layer after the device sends a message [108]. It also determines whether a packet is appropriately retrieved from the appropriate channel. The device operates based on its power and interference levels. These

classes represent the chip that follows its state and generates the following values using an m-state attribute:

- Tx represents packet transmission.
- RC represents an incoming packet [108].
- IDLE when the channel listens to the incoming packets.

Each state of the channels can be linked to a different voltage and current consumption by the energy model that explains the network devices' energy expenditure and estimates the device's battery life [108]. The energy models are fully integrated with the network simulator but require specific care when designing the classes and combining LoRa and energy models. LoRaPhy classes have similar subclasses, such as LoRaMAC, EndDevice LoRaPhy, and Gateway LoRaPhy [108]. Both implementations compute interference, similarly, using the LoRaInterfereneceHelper class to record every signal that arrives at the device.

3.4.4 LoRaChannel

The LoRaChannel class models the wireless channels for all devices connected toa LoRaWAN network [108]. This class oversees packets from the PHY layer as a collection of LoRaPhy objects. LoRaPhy objects are added to the list of channel objects and registered during the configuration. The Send and Receive methods of the LoRaChannel class enable communication between registered PHYs.

A PHY uses the proposed parameters, such as the message's spreading factor, PHY-layer packet, duration, transmission power, and channel number, when it attempts to send a message in the channel by calling the send function [108]. When the channel calls this function, it goes through the list of PHYs and sets the schedule for receiving events for the registered nodes into the channel to listen to the communication model [108]. The channel uses PropagationDelayModel, a built-in ns-3 abstract class that offers the mobility model of the transmitter and receiver to calculate the time of the scheduled receive event. The user can calculate the delay using many models that have been used.

3.4.5 LoRaNetDevice

A LoRa Network Card was modelled by the LoRaNetDevice class. Numerous apps can communicate data to other LoRa devices via NetDevice, which can be connected to the node. A LoRaNetDevice is primarily used to house all LoRa objects, including LoRaMAC and LoRaPHY, which must be aggregated to a node [108].

The NetDevice class has only one consideration for IP communication. This is indicated in the documentation for the LrWpanNetDevice API, where the ns-3: GetMultiCast, Send, and SendTo functions of the IP-Specific API are available in NetDevice [108]. These methods are not effective for mapping the 802.15.4 MAC MCP Data requests. In ns-3,LoRa modules use NetDevice as an encapsulating class, and there is no support forconcepts such as multicast and IPv6 [108].

A LoRa Network was modelled by the LoRaNetDevice class. Numerous apps can communicate data to other LoRa devices via NetDevice, which can be connected to the node. A LoRaNetDevice is primarily used to house all LoRa objects, including LoRaMAC and LoRaPHY, which must be aggregated to a node [108].

3.4.6 Helpers and Tests

Apart from the different types of building blocks in ns-3, there are many helper classes for configuring the LoRa network to be more accessible [107]. Helpers in ns-3 are used to design elements intended to assist script creators in setting the topologies and network nodes so that users can configure them according to the desired module. When an instance of the periodic sender application is created, an event scheduler sets its transmission periods.

Network applications can deploy several nodes using the PeriodicSenderHelper class to simplify these class configurations [107]. The helper class determines the network node's reporting period according to the sensor node's specifications. ns-3 also has another set of helpers written according to the user’s appropriately installed and configured LoRa stack for each number of nodes [107]. LoRaNetDevice, LoRaMAC, and LoRaPHY objects can be created and deployed in a particular set of nodes using LoRaHelper, LoRaPhyHelper, and LoRaMacHelper. This ensures that the PHY layer is correctly connected to the LoRaChannel

and that the nodes are configured to communicate properly [107]. To verify that the program is accurate after each update and complies with ns-3 standards, ns-3 includes a set of tests and a module in place. Simple message delivery is tested to ensure that the PHY layer on the end device can receive messages within the range [107]. Channel separation tests are also conducted to ensure that the device listens to the channel on which the user cannot perform [107].

3.5 Performance Evaluation

For the LoRaWAN simulation, the ns-3 simulator requires proper software configuration. It is crucial to have a simulation script that manages the helper system to set up various devices [107]. Furthermore, a simulation script was used for data collection during the simulation process and to control the trace sourcesplaced in the significant class variables. When an event occurs during the simulation, the script-level function registers the event in the script data structures [107].

The end devices are formed and given a uniform random position in a circle of radius r using the default ns-3 class UniformDiscPositionAllocator [107]. After the LoRaChannel class, the propagation loss models have been established. The LoRaHelper class then connects the end devices to the channel instance after configuring the LoRaWAN stack on each device. End devices are preconfigured and can communicate in the network; therefore, a join procedure is unnecessary [107].

After the end devices, gateway nodes are created, allocated in a hex grid, configured to use the LoRaWAN stack and receive path allocation. The callback functions in the ns-3 modules are linked to trace sources in the code to gather data on changes to events in the simulation state. During the simulation, both the end devices and gateway devices must be fixed [107]. The last configuration step and PeriodicSender applications were installed on each end device and set up for a start and stop at designated times. Finally, the simulation begins, and when it ends, the scripts collect the results from their data structures and save them into the program file [107].

3.5.1 Variables and Metrics

"**Performance Metrics**: system or standard of measurementin a broad sense and quantities related to the performance and reliability of the network. Networks have many performance metrics, but throughput, packet delivery ratio, and end-to-end are important."

The simulation process explained above also requires the definition of variables that can be adjusted to determine their effects on the performance of the network system. The different variables were as follows:

- **Network Scale:** A high number of end devices in the network will also have a high probability of two-signal interference [108]. The gateway deployed in the fixed area affects the devices' coverage.
- **Model:** The model of the traffic generated by the application layer shows a constant device density; lower message inter-arrival times create more collisions among packets[108].
- **Components:** Path-loss model components can affect the gateway coverage range [108]. For example, external walls in buildings decrease the losscomputed by the channel.
- **Spreading factor distribution (SFD):** shows the performance of LoRa networks and explains how the spreading factors are distributed at different distances from the gateway [108].
- **Lost Messages or Packet drop:** messages were dropped at the application layer owing to duty cycle regulations, and the PHY layer messages were lost because of the power under sensitivity.
- **Network Throughput:** This focuses on the network efficiency and device probability by considering the raw data extraction rates.
- **Spreading Factor:** Before starting the simulation, each end device was assigned a spreading factor (SF). The simulation process measured the distance of each end device from the gateway and set the SF based on [108].
- **Packet delivery Ratio:** Packet delivery ratio (PDR) is a metric used in the field of networking and communication to measure the effectiveness of data transmission

within a network. It represents the ratio of successfully delivered packets to the total number of packets sent. In other words, PDR quantifies the percentage of data packets that reach their intended destination without being lost or dropped during transmission.

3.6 Results

This section analyses the metrics obtained to better understand the LoRa module in the ns-3. To analyze the performance of the LoRa network, it was designed for a specified area and with specific parameters. This section explains the developed system model, LoRa sensor network setup, and its implementation. The Last part of this section explains different performance metrics. The desired network was set up with a realistically modelled environment to evaluate packet loss in both implemented networks.

3.6.1 System Model

Multiple sensor nodes with comparable characteristics, such as processing power, low energy consumption, and wireless transmission range, were used in this study. These nodes have sensors and can send data directly to a single base station (BS) node. In contrast to sensor nodes, the BS functions as a sink node that accepts monitoring data from the sensing nodes. These data will be compiled by application type and stored locally for future users. Figure 3.7 shows that the simulation model implemented in ns-3 includes the network server, gateway, and nodes (LN1, LN2, LN3, LN4). The nodes are dummy sensor nodes S_1, S_2, and S_3. S_1 S_2 is assigned to temperature, PM or $PM_{2.5}$,and S_3 for CO_2. Dummy sensor nodes generate dummy values to show the pollutant level in the designated area of 1200m × 1000m and show the network topology with a connected network server, gateway, nodes, and dummy sensor nodes. Dummy sensor nodes transmit data to nodes where the nodes communicate with the gateway using channels.

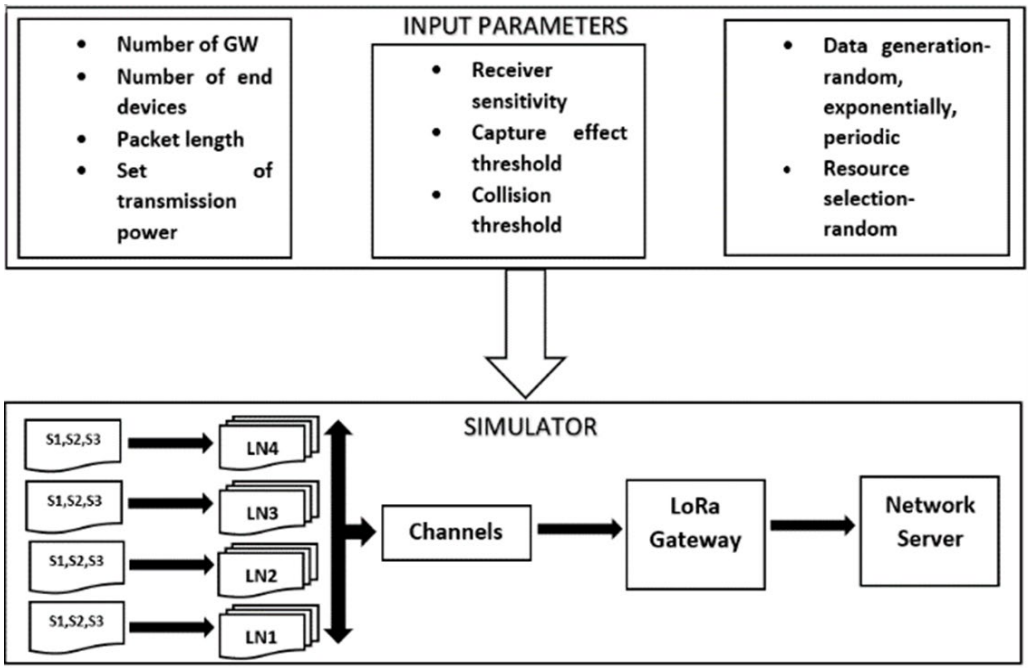


Figure 3.7 Simulation Model

ns-3 has some operations to follow to begin the development of new applications. The simulation environment used sensor nodes with comparable characteristics, such as computation capacity, energy limitations, and wireless transmission range. These sensor nodes have installed three separate environmental sensor systems, which transmit data to the gateway. Every sensor node in the network sends monitoring data to the gateway, which functions as a base station and receives it.

The sensor nodes (LN1-LN4) are distributed randomly throughout the designated area for data transmission over the network, which is based on a tree network topology. Sensor nodes are set up to provide monitoring data by the availability of the three dummy sensor nodes to which they are connected. Periodic data must be transmitted to each sensor node. The data are transmitted along with routine data whenever any event occurs.

3.6.2 LoRa Sensor Network Setup

An exciting technology for lightweight sensing on the Internet of Things is the long-range (LoRa) (IoT). It belongs to the brand-new network class known as the low-power wide area network (LPWAN). It establishes a radio layer based on Chirp Spread Spectrum (CSS)

modulation and the LoRaWAN straightforward channel access scheme [109]. LoRa can fill the gap between mobility and low power. Technological growth has increased research interest in exploiting the performanceand scalability of LoRa networks. Additionally, some researchers have attempted to increase the dependability of LoRa. LoRa is a physical layering method. To create a comprehensive communication system that functions well with LoRa, the MAC layer protocol, LoRaWAN, was proposed.

Nodes, gateways, and servers are the three key components of LoRaWAN, an open-source MAC protocol built around LoRa, as shown in Figure 3.8. LoRaWAN is primarily intended for low-power, long-distance communication and operates in unlicensed ISM bands. The nodes in the LoRaWAN can only send a few messages per day. LoRa employs a chirp spread spectrum with various spreading factors to trade off data for various ranges to facilitate long-distance communication. The SF can be chosen from SF7 to SF12 using the LoRa specifications. Lower SF chirps work faster over the bandwidth, leading to a higher transmission rate and shorter transmission time, thus requiring a higher SNR [109].

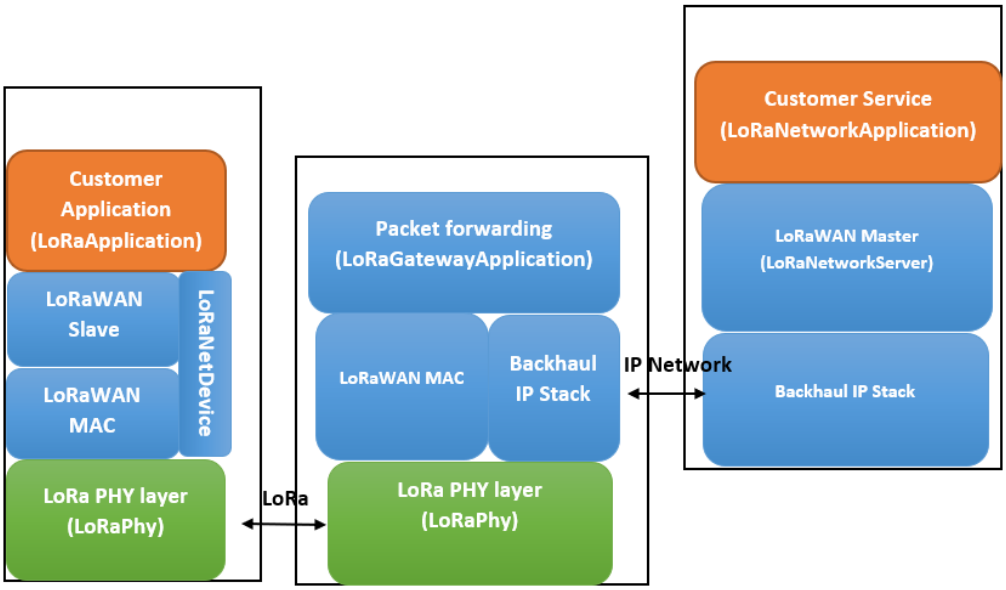


Figure 3.8 Structure of LoRaWAN network [109].

3.6.3 LoRa Sensor Network Implementation in ns-3

The LoRa sensor network setup starts from the physical MAC and application layer. The LoRa sensor network implemented PHY and MAC layers for the standard ns-3 callbacks for communication with the remainder of the network stack. This allows the network to reuse a significant amount of code within the ns-3 framework. The LoRaWAN NetDevice can easily be configured using default parameters but is altered according to the network requirements.

3.6.3.1 Physical Layer

The physical layer of the planned network is based on the SpectrumPhy code and is similar to that of the LR-WPAN. Users can build a spectrum channel that simulates the 915 MHz band for LoRa traffic using SpectrumPhy's standard interface [109]. To support quick simulations, the default LoRa channel given in LoRa is 915MHz. Packet channels are represented as a subclass of the spectrum signal parameter, in which the spreading factor and bandwidth of the signal are considered. Each node in the Phy layer monitors the noise across all channels and considers the applied spreading factors [109].

3.6.3.2 MAC Layer

There are two classes in the MAC layer implementation, owing to the distinction between uplink and downlink. The MAC layer of the sensor node was implemented in LoRaNetDevice, whereas the gateway MAC layer was implemented in LoRaGwNetDevice. The node's MAC layer is straightforward: when a message arrives, it looks for a free channel on the network that is open for data transmission. It must be thoroughly examined to ensure compliance with both LoRaWAN limitations and duty cycle requirements from a legal standpoint. The node waits for a random delay and sends a message if the network channel is open. If no channel is available, the delay continues, and another attempt is made. Two receiving slots for potential downlink messages were opened after the successful transmission. The second downlink stops operating if the slot is already in use.

3.6.4 Lora Sensor Network System Model

The LoRa sensor network operates based on ISM bands, where the network contains four LoRa modules connected to three dummy sensors named S_1, S_2, and S_3. The LoRa sensor modules transmit data periodically; however, events occur in the LoRa network. If events occur, the data are sent regularly to the LoRa gateway. Figure 3.9 shows the GUI of the ns-3 simulator. Both networks presented similarly in ns-3 when LoRa sensor nodes were successfully constructed. The next step is to move towards the main operating system of the designed network.

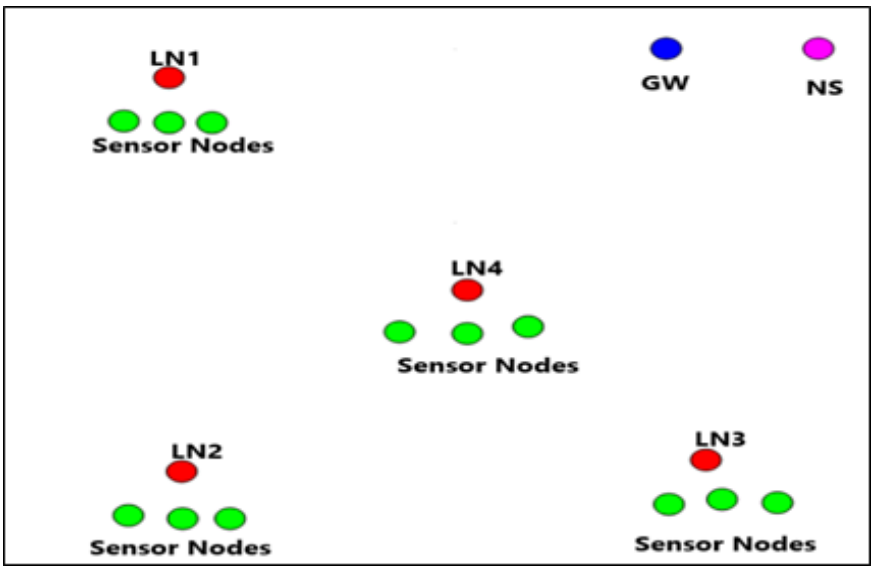


Figure 3.9 Network Topology GUI from ns-3 simulator

The designed network works based on two files: the main file includes nodes, gateway, and network server, and the second file (LoRaNetDevice.cc), which provides dummy sensor nodes, is interconnected with the main file. In the LoRaNetDevice file, sensor values are assigned limits to generate random values for the sensor, such as temperature, dust, humidity, particle matter, CO_2, and transmission over the network.

Figure 3.10 shows the LoRa sensor network connectivity in the ns-3 simulator. The LoRa sensor nodes connect with three sensors to monitor the environment's temperature, humidity, CO2, and PM and are labelled S_1, S_2, and S_3. These dummy sensor nodes transmit pollutant levels to LoRa sensor nodes using wired connectivity. The LoRa sensor nodes communicated

with the LoRa gateway using the ISM band at 915MHz. The gateway receives data from the LoRa sensor nodes with their MAC addresses and saves them according to the ID number of the nodes. The simulation process for the designed network was performed for 20 min in a specified area using different parameters. Network performance measurement is described in the next section.

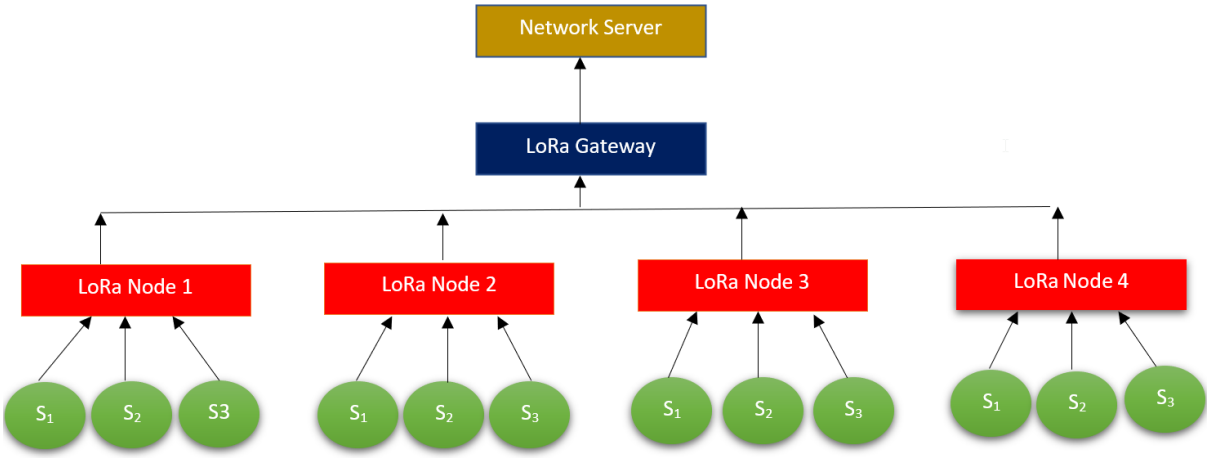


Figure 3.10 LoRa Sensor Network

3.6.4.1 Network Throughput

The network throughput in the designed network refers to the amount of data transmitted from one device to another in each time frame. The network throughput measures the number of packets arriving at the destination address. This is measured in bits per second, but this can also be measured in data per second, the network throughput, where packet arrival is considered a high-performance service within a network. Packet loss, latency, and jitter can all be addressed at a slow throughput speed.

In the designed network, the network simulation was performed for 20 min, during which the network system transmitted the desired number of packets from one device to another. Data transmission started from the dummy sensor nodes that transmitted the dummy sensor values to the LoRa modules connected and transmitted over the network. The network throughput of the designed network was measured in percentages, as shown in Figure 3.11. Data transmission through the LoRa sensor network was performed every 5-minute and packets were sent over

the network. The number of packets was transmitted by the dummy sensor nodes simultaneously, but the network throughput of the LoRa sensor network was the same as that of the others, with a slight change. The lowest network throughput of the LoRa sensor network was 98%, and the last batch was transmitted. The designed network does not show any packet loss or latency, which may lead to a slow performance of the network. The network throughput can be calculated using the formula below:

Equation 3.1

$$Throughput = (packet\ Delivered - Packet\ Drop\ \div Packet\ Delivered) \times 100$$

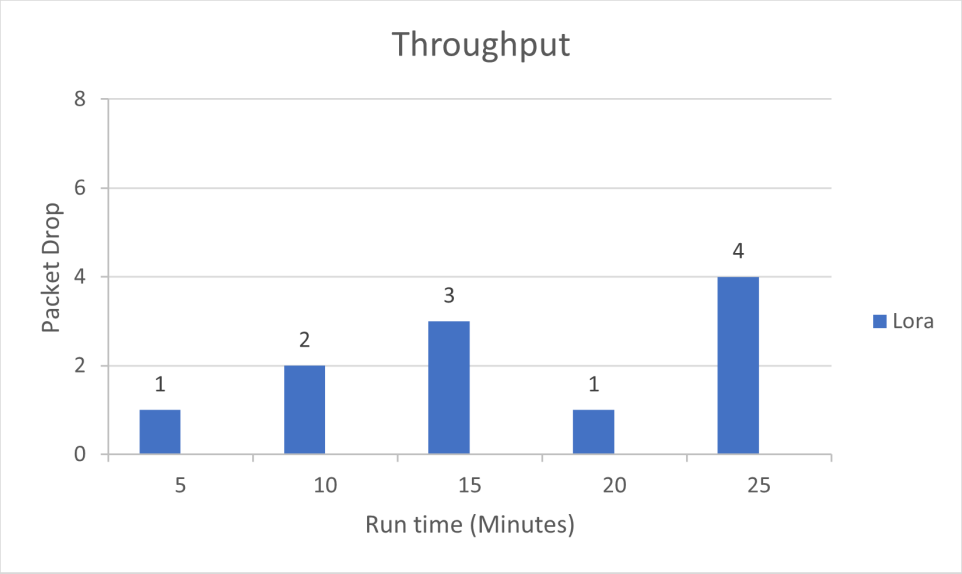


Figure 3.11 Chart showing Network Throughput results.

3.6.4.2 End-to-End Delay

The amount of time it takes a packet to travel through the network from source to destination is known as the end-to-end delay. It is distinct from round-trip time and is used frequently in IP network monitoring. It can come from a variety of locations, such as transmission, propagation, processing, and queue delays. The percentage of delivered packets partially determines end-to-end delay. The packet loss likelihood increased as the source and destination distance increased. Using the packets dropped in the network, determining the end-to-end delay in the specified network during network transmission was possible.

The transmission delay is 220.23 ms; after every 5 minutes, different packets are transmitted, and the end-to-end delay is increased to 908.25 ms when the last batch of packets is transmitted, as shown in Figure 3.12. A delay during the transmission occurred because of the traffic on the sensor network and failed to receive packets on time. The End-to-End delay of the network was calculated using the following formula:

Equation 3.2

$$d = N \times L \div R$$

where` d = delay (ms)

L= length (m)

N= number of links

R= transmission rate (ms)

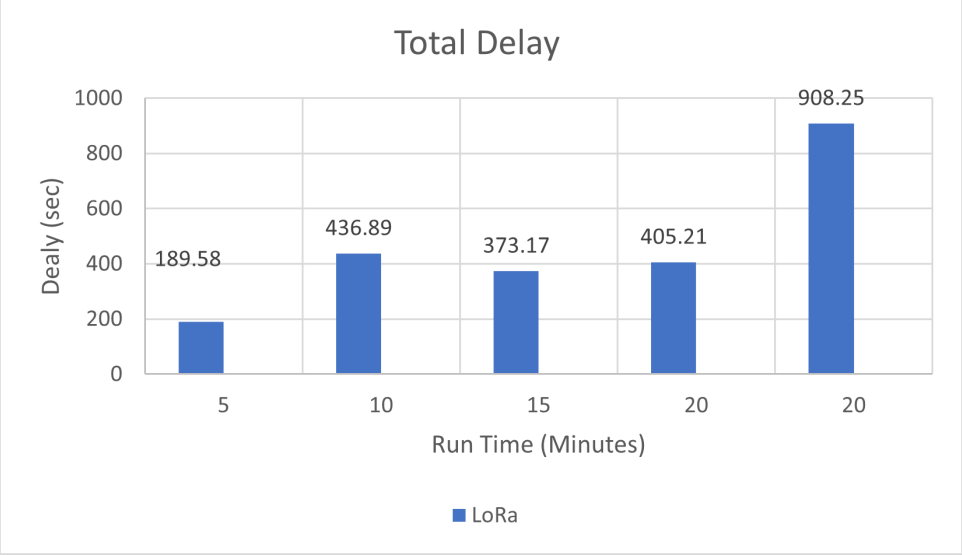


Figure 3.12 End-to-End Delay

3.6.4.3 Packet Delivery Ratio (PDR)

The packet delivery ratio is a crucial metric for assessing the effectiveness of a routing protocol in a network. Many parameters chosen during the network simulation determine the

performance of the protocol. The main factors are the packet size, node count, transmission range, and network topology. The network packet delivery ratio can be calculated by dividing the total number of data packets that have reached their destinations by the total number of packets transmitted from the sources. In other words, the packet delivery ratio is the proportion of packets transmitted from the source address to the number of packets received at the destination.

In the proposed network, 250 packets were transmitted to monitor the network performance with dummy sensor nodes that generated dummy values for sensors to monitor pollutant levels, as shown in
Figure 3.13. Network throughput was measured using different input setups. The simulation started with a simulation time of 5 minutes. Both networks transmit dummy values over the network as dummy sensor nodes. This indicates that thepacket delivery ratio of the LoRa network was the same. The lowest was 80 per cent of the packets delivered during the transmission. The PDR of the network was calculated using the following formula:

Equation 3.3

$$\text{PDR} = \text{Packet Delivered} - \text{Packet Drop} \div \text{Packet Delivered} \times 100$$

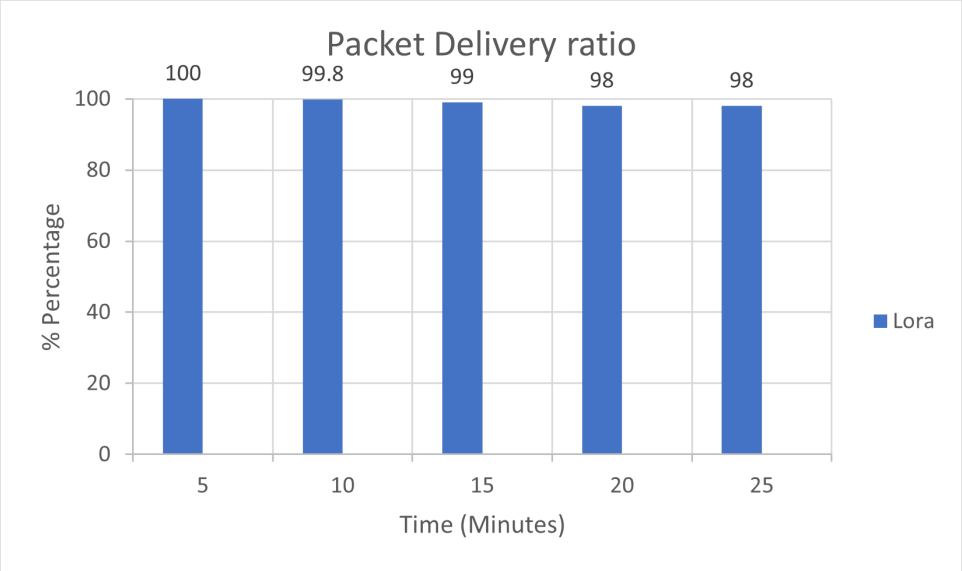


Figure 3.13 Packet Delivery Ratio

3.7 Compared with Wireless Sensor Networks

After the network simulation in the desired time, it was found that the network has better efficiency than other networks. It was compared with the WSNs using different performance metrics. Both networks are arranged with the same parameters and network area. The simulation was performed for 30 minutes for both networks for the first transmission; it took 5 minutes, the data was transferred to the network server using network gateways, and the final transmission took 25 minutes for the first transmission, it - has no packet drop, but after 10 minutes, the maximum numberof packets dropped because of network collision in WSN, whereas in the LoRa network, the packets dropped were less. Accordingto the findings, network throughput is better in the Lora network, where a smaller number of packets are dropped. The simulation process of dummy nodes and coding for both networks are given in Appendix A. Figure 3.14 shows that both networks contain four nodes connecting 12 dummy sensor nodes in an area of 3000 m^2. During the transmission, 250 packets with a size of 51 bytes were transmitted over the networks.

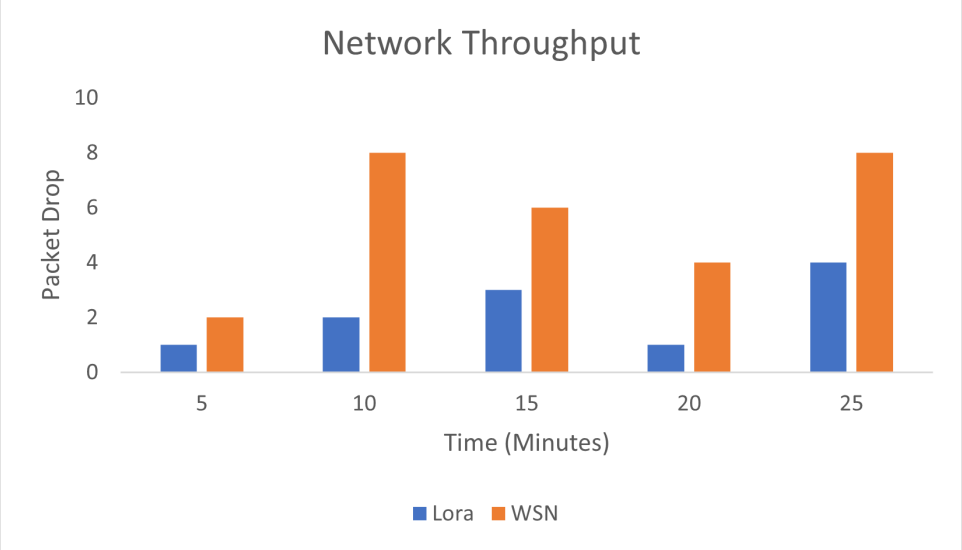


Figure 3.14 Network Throughput

The dummy sensor nodes were transmitting random temperature, humidity, dust, and CO2 values. The first transmission took 5 minutes, and the data was transferred to the network server using network gateways. The final transmission took 25 minutes. The first transmission has no packet drop, but after 10 minutes, the maximum number of packets dropped because of network collision in WSN,whereas in the LoRa network, the packets drop was less.

Accordingto the findings, network throughput is better in the Lora network, where a smaller number of packets are dropped.

Figure 3.15 shows the end-to-end delay on the designed networks to check the LoRa and WSN performance levels. The data transmission occurs through only two adjacent nodes but via a path which may include many intermediate nodes. End-to-end delay is the summation of delays experienced at each hop from the starting node to the last node. As per the simulation set-up, LoRa performs better than the WSN network when the dummy nodes generate dummy pollutant values and transmitting over the network after a given time period. Every simulation was performed using time intervals of 5 minutes to 25 minutes. For the first transmission, about the first 5 minutes, both networks have 0 delay in transmitting, and all the packetsare transmitted. Both networks transmit packets with some delay during time and packet size changes, but WSN shows the worst simulation results when performing for 30 minutes. The performance of the Lora network was better than that of WSN, with different packet sizes and transmission times.

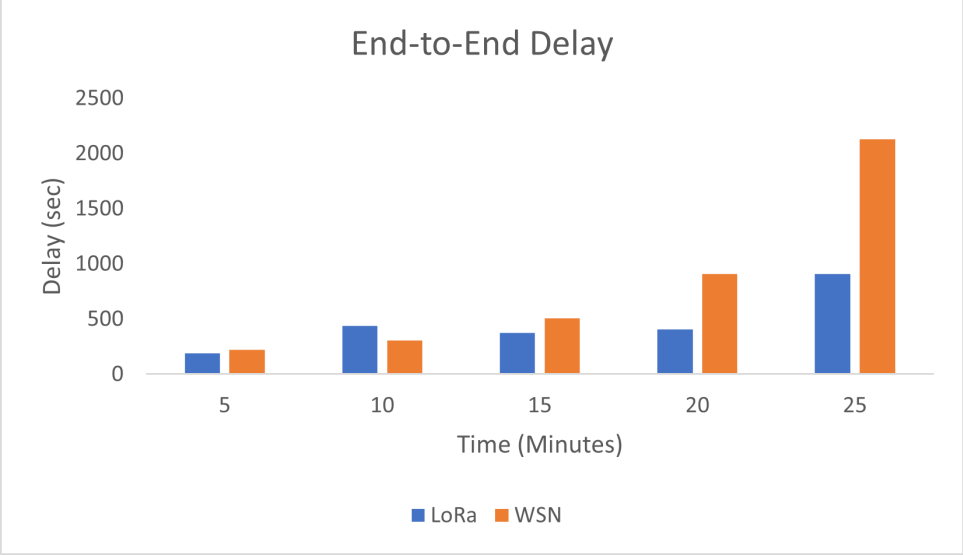


Figure 3.15 End-to-End Delay

Figure 3.16 shows the packet delivery ratio for LoRa andWSN networks. In the proposed networks, 250 packets were transmitted over the network to monitor the network's performance with the connection of dummy sensor nodes that generate dummy values for sensors to monitor pollutant levels. The results for throughput were measured with different input data

transmission setups; both networks slightly differ in packet delivery. The simulation was started with 5 minutes of simulation time in both networks as the dummy sensor nodes were transmitting dummy values over the web; it shows that the packet delivery ratio of the Lora network was the same, but in the WSN network, packet delivery ratio started decreased at few seconds and lowest was 80 per cent of packets delivered during the transmission time. WSN network has a small range of communication, which causes a lower packet delivery ratio over the network design.

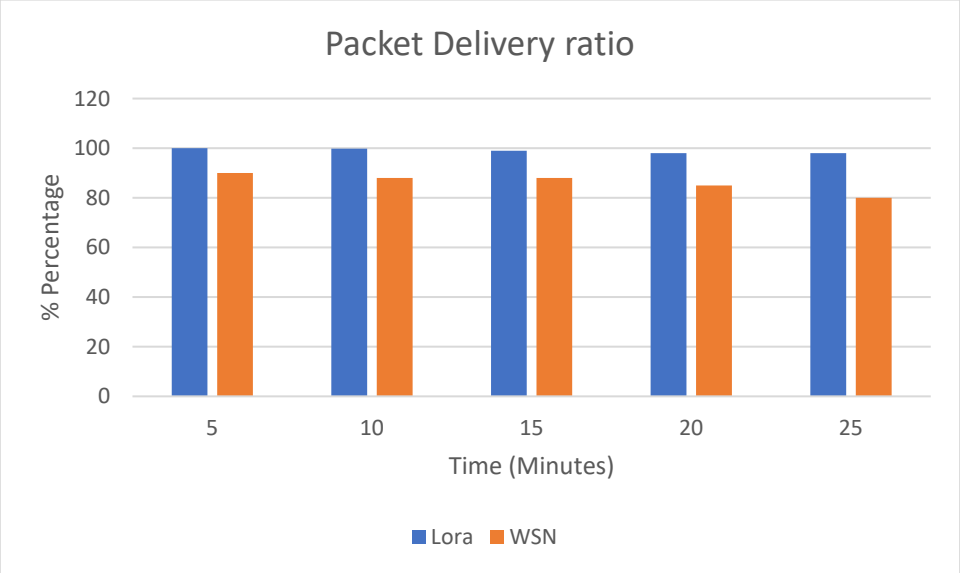


Figure 3.16 Packet Delivery Ratio

In the simulation, it was decided to randomly transmit 250 packets to deliver to the network server in the specified time period. Number of packets can be increased or decreased, as it can also affect the performance of both networks. For the larger simulation process, it can provide better results.

3.8 Estimated Pollution in the Specified Simulated Area

From the network simulation in the specified network area, they implement sensor nodes connected with three dummy nodes on each sensor node. Dummy sensor nodes were generating sensor pollution values at different heights and locations. This represents the results

from node 1, which shows the pollution values and changes in the pollutant concentration levels. Dummy sensor values were assigned in a specific range as below in Table 3.3.

Table 3.3 Dummy Sensor Range

Parameter	**Range**	**Unit**
Temperature	0-25	Celsius
Humidity	20-80	Percentage
CO_2	1-15	PPM
Dust	20-45	Ug/m3

The data collected for node 1 varies according to the server and gateway distance. It was placed with different dummy sensor nodes; results are shown in Figure 3.17 to Figure 3.23. The results show that the temperature levels were stable because the sensor node was at a fixed altitude. Humidity estimation values also depend on the temperature levels of the dummy sensor nodes. Regarding the CO_2, PM_1, $PM_{2.5}$, PM_{10}, PM_4 and dust levels were tested simultaneously. Furthermore, according to the simulation time of 25 minutes, pollution levels should not exceed the pre-set values, as the network generates dummy sensor values. From the different parameters set in the network methodology, particulate matter is the concerned parameter.

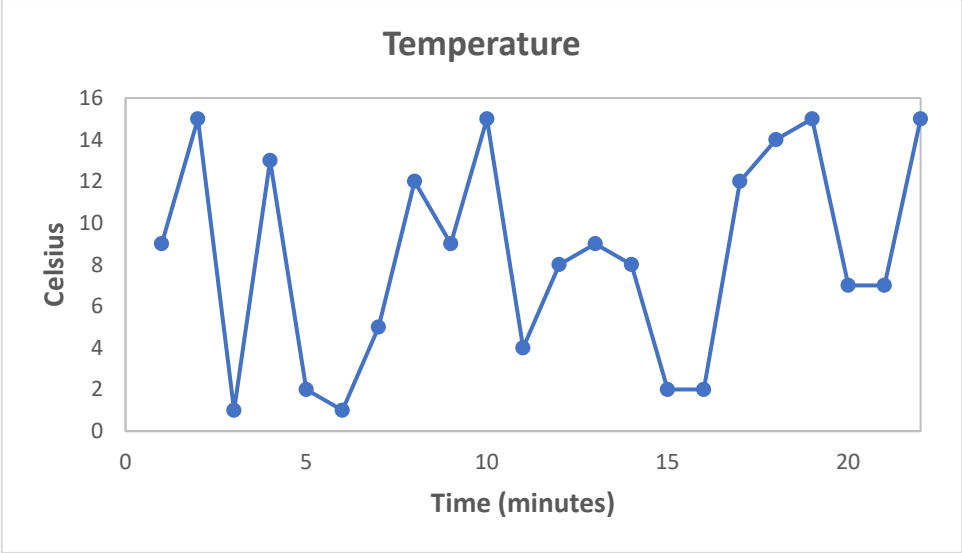


Figure 3.17 Temperature Readings

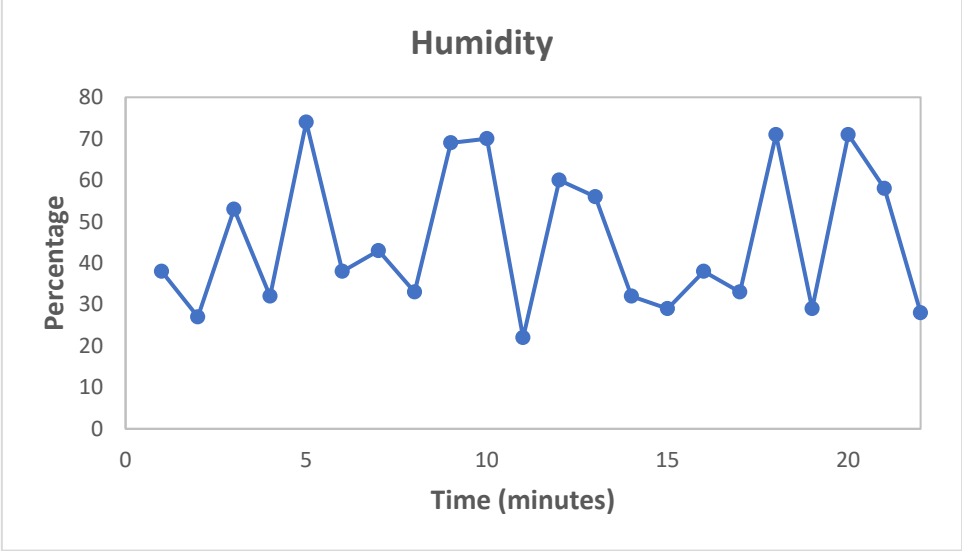


Figure 3.18 Humidity Readings

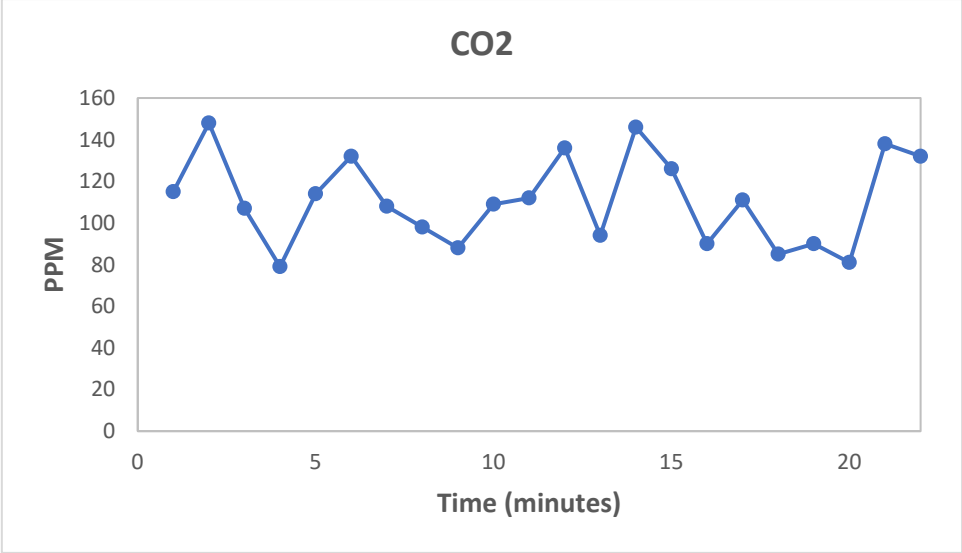


Figure 3.19 CO_2 Readings

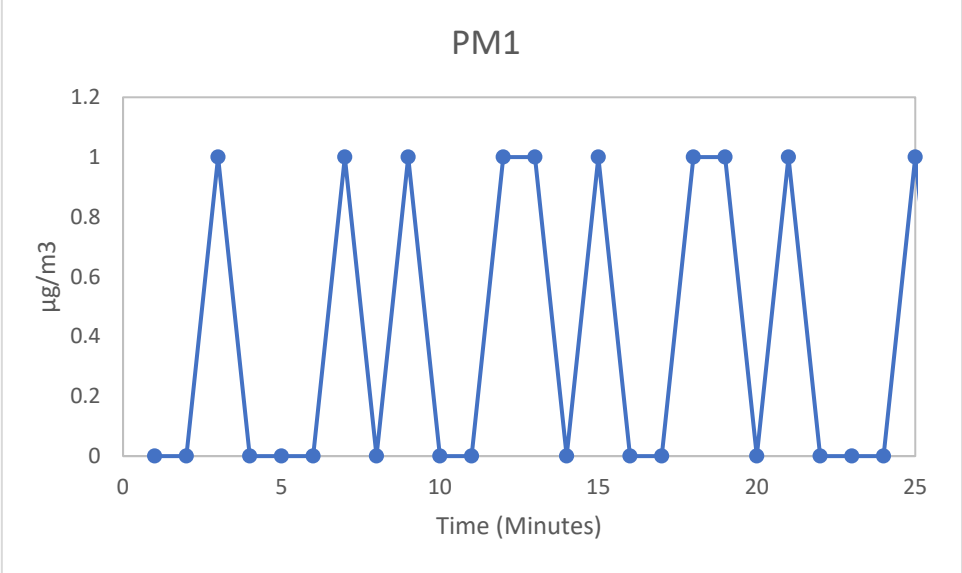


Figure 3.20 PM_1 Readings

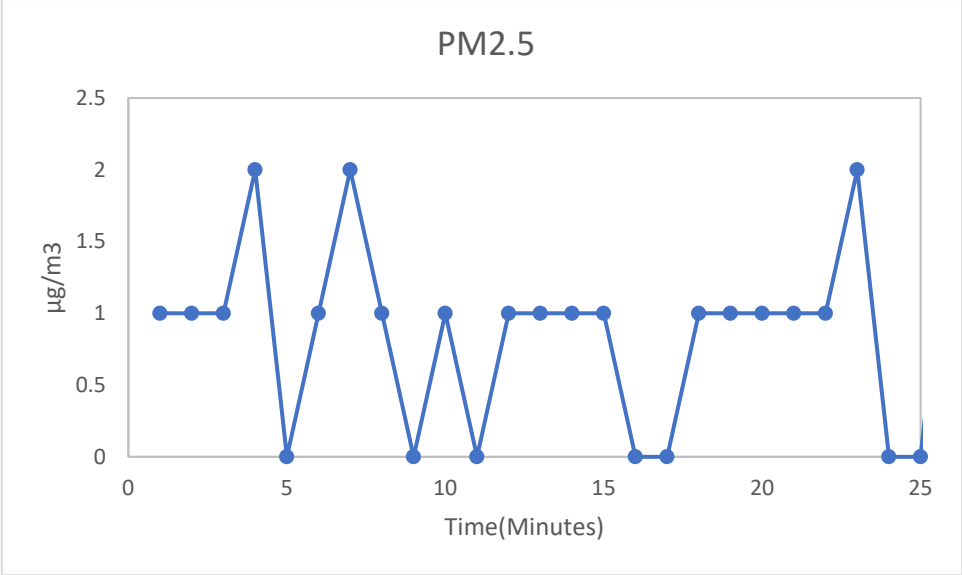


Figure 3.21 $PM_{2.5}$ Readings

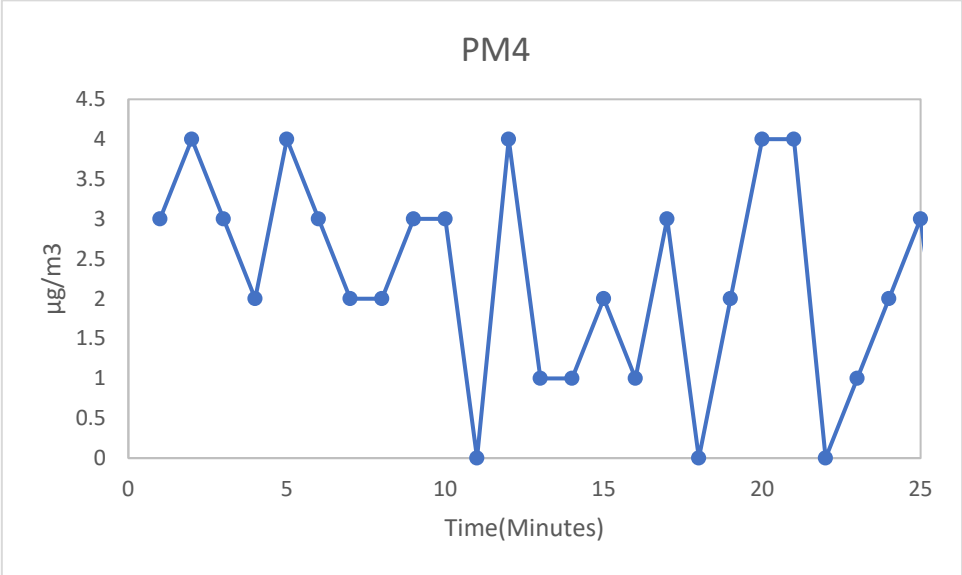


Figure 3.22 PM_4 Readings

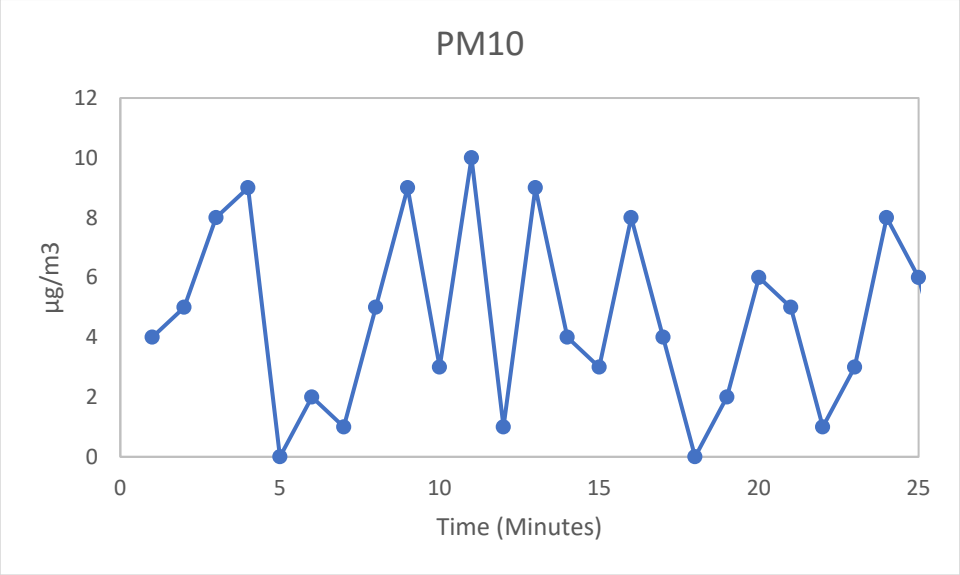


Figure 3.23 PM_{10} Readings

3.9 Summary and Contribution

This chapter covers the latest LPWAN technologies, such as the LoRa framework, and how it allows the network evaluation level performance of the LoRaWAN system. This chapter discusses the use of network simulation in evaluating the performance of LoRa sensor nodes. The ns-3 simulator is chosen for its ability to model a LoRa network with long-range communication and low power consumption. The chapter outlines the features, challenges, and key performance metrics used to assess network performance.

The three performance metrics analyzed are packet delivery ratio, end-to-end delay, and network throughput. The results show that the designed LoRa network outperforms other network devices like wireless sensor networks. The importance of selecting the right network topology for different applications and research purposes is also emphasized.

The network system was proposed to collect estimated values using LoRa NS3 simulation. LoRa network performed the simulations using end-to-end architecture, which will monitor air pollutants using dummy sensor nodes. Implementation of the proposed network of the external zone was ensured with ns-3. Each network was performed with different parameters during the simulation to monitor its performance. In this network simulation, the end-to-end network

architecture is introduced. The transmission starts from dummy sensor nodes, then LoRa nodes and collected data is transmitted to the network server using the LoRa gateway.

This comparison likely offers insights into how the LoRa sensor network monitors and addresses air pollution, demonstrating its potential effectiveness in this context.

In addition to the critical points outlined in the previous summary, the research is further explained in terms of the implications and significance of the chapter's content:

- **Simulation and Technology Selection:** The chapter underscores the importance of simulation tools in assessing network performance. By selecting the ns-3 simulator, which can accurately represent the unique characteristics of a LoRa network, researchers can gain insights into how this technology performs under various conditions. This simulator choice reflects the rigorous approach to evaluating LoRa's suitability for real-world applications.
- **Performance Metrics:** The chapter focuses on three critical performance metrics—packet delivery ratio, end-to-end delay, and network throughput. These metrics are vital for understanding how well the LoRa sensor network functions in practice. The packet delivery ratio reveals the network's ability to transmit data successfully; end-to-end delay indicates data transfer speed and network throughput provides insights into overall data handling capacity.
- **Comparative Analysis:** The chapter's findings, which highlight the superior performance of the LoRa network compared to other wireless sensor networks, have broad implications. This suggests that LoRa technology is not only a feasible but also a highly efficient choice for applications like air pollution monitoring. The ability to outperform existing solutions can lead to more accurate data collection and potentially more effective pollution management strategies.
- **Topology Selection:** The discussion on topology selection is crucial. It emphasizes that different network layouts may be required for various applications. Understanding how network topology impacts performance is essential for optimizing the deployment of LoRa networks in specific environments and use cases. This flexibility can make LoRa technology adaptable to diverse scenarios.

- **Air Pollution Monitoring:** The chapter goes beyond network performance and discusses the estimated pollution levels in a specific area. This detailed analysis is valuable as it links the technical aspects of the LoRa sensor network with its real-world application. The chapter likely explores how the network's data collection capabilities contribute to a more comprehensive understanding of air quality and pollution trends.

Chapter 4 .
Low-Cost Monitoring Wireless Sensor Network Design and Development

This chapter focuses on network deployment using long-range (LoRa) technology. It starts with classifying wireless communication protocols, such as Sigfox, Long-Range Wireless Area Network (LoRaWAN), and Narrow-Band Internet of Things (NB-IoT). The following section covers the LoRaWAN Specification and LoRa modulation, things network, radio communication theory, and air pollution scope. The following subsections describe the hardware and software used and issues during network deployment. The final section explains the experimental setup and gateway position.

4.1 Classification of Wireless Communication Protocols

Wireless Communication in networks is one of the most desirable modes of communication between two or more devices. This section discusses the modern technologies used in wireless communication.

4.1.1 SigFox

French entrepreneurs Ludovic le Mon and Christophe Fourtet established SigFox in 2010. They planned to link every piece of hardware, from the physical to the digital world. The SigFox system architecture is shown in Figure 4.1 [110]. Approximately 70 nations and regions are covered by Sigfox, which also links 15.4 million registered devices and sends about 4 million messages daily. SigFox offers an Internet of Things (IoT) network that connects approximately 1.1 billion individuals [14]. Recently, Sigfox developed a distinctive business model for network operators and employed a novel approach to the IoT concept [15]. SigFox employs differential Binary Phase-Shift Keying (BPSK) with Ultra-Narrow Band (UNB) modulation, operating at 100bps. This device enabled Sigfox to send three uplink packages using three random frequencies. The packages were successfully received from a base station. A new device begins to use the same frequency range for communication if any packets are

lost. Within an hour, Sigfox can only send data for 36s. As a result, only six messages with payloads of 4, 8, or 12 bytes were sent [16].

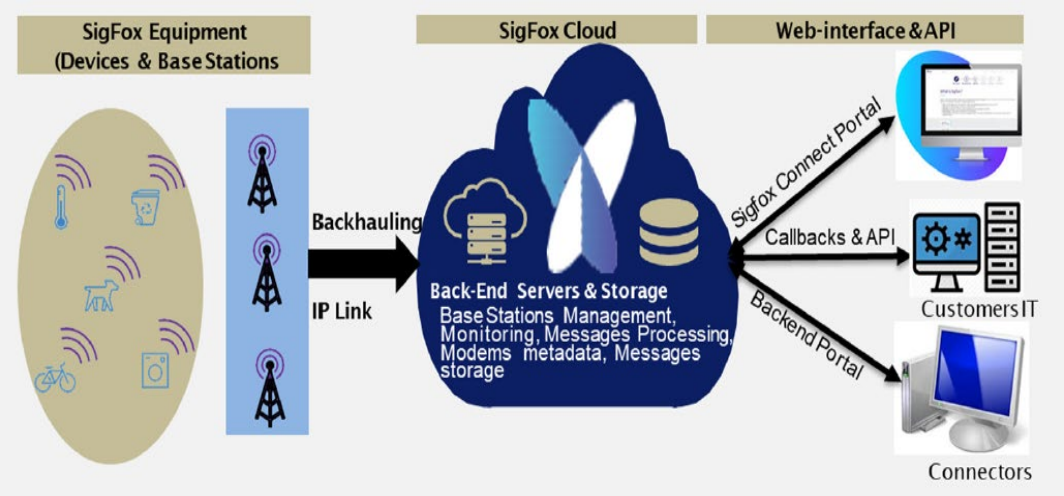


Figure 4.1 Sigfox Architecture [110].

Like LoRa, SigFox utilizes the unlicensed Industrial, Scientific, and Medical Radio (ISM) bands and efficiently uses the frequency bandwidth with a shallow noise level, low power consumption, high receiver sensitivity, and low-cost antenna designed by BPSK. SigFox services operate worldwide in the ISM and short-range devices, at 862–928 MHz, and other features are shown in Table 4.1.

Table 4.1 Sigfox features

Factors	**Values**
Number of messages over the uplink	140 messages/day
Maximum payload length for downlink messages.	12 bytes
Maximum payload length for uplink messages.	8 bytes
Maximum Throughput	100 bps

4.1.2 Long-Range Wide Area Network

The LoRa alliance created a long-range wide-area network (LoRaWAN) as a communication protocol for LoRa. LoRaWAN comprises three crucial parts: end devices (ED), gateways (GW), and network servers. EDs use gateways to gather data and connect to network servers (GWs), as shown in Figure 4.2 [111]. GW transports data between Eds and the network server (NS) [32]. LoRa modulation, which is based on the Chirp Spread Spectrum (CSS) with various spreading factors (SF7 to (SF12) to modify the data rate and range trade-off, is used by LoRaWAN to offer long-range communication. A lower SF makes a faster data rate possible at the expense of a shorter range and vice versa. Typically, a star or star topology is used to deploy a LoRaWAN network architecture [32].

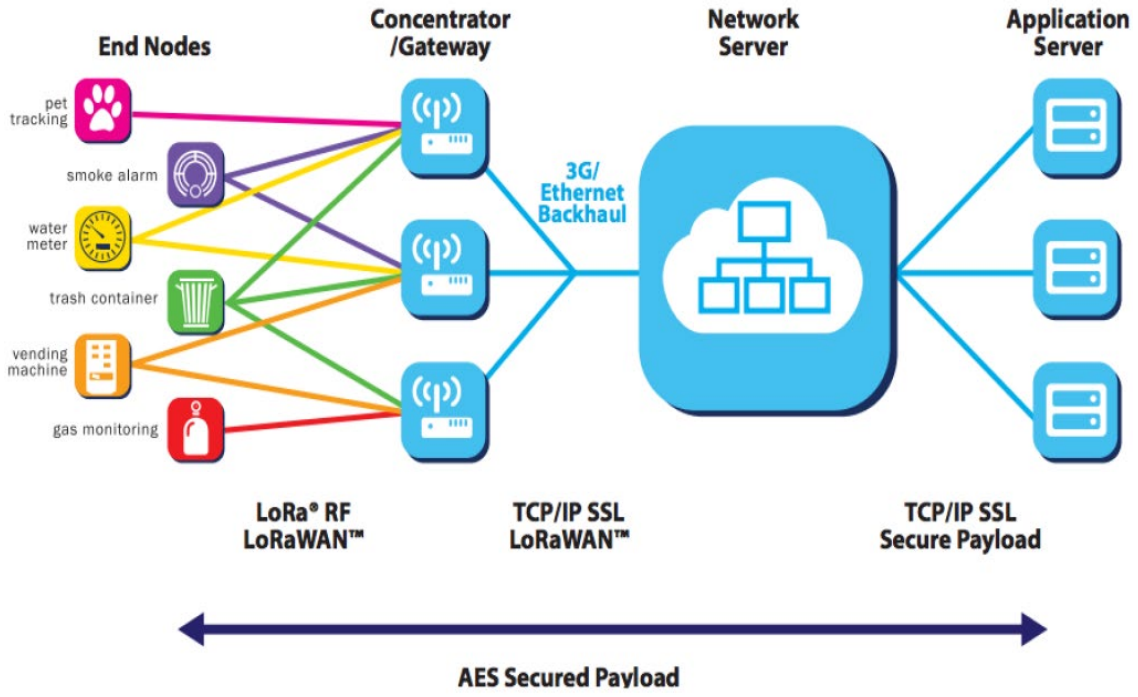


Figure 4.2 LoRaWAN Architecture [111]

After the EDs were activated, the LoRaWAN network began to communicate. Activation by Personalization (AP) or over-the-air activation (OTAA) are two ways to start the ED (ABP). Once an ED is running, it joins a LoRaWAN and offers UL and DL connectivity as two types of communication. Messages transmitted by EDs to the NS and relayed by one or more GWs are called UL communication. All gateways positioned within the transmission range received messages sent by the ED. The NS receives data from the GWs, verifies its accuracy, eliminates

redundant receptions, and selects the appropriate application server (AS) to transmit the data. The information delivered by the AS to the Eds matches the DL communication. This data can be the acknowledgement or a specific message the AS needs to send to an ED [111].

4.1.3 Narrow-Band Internet of Things

Using cellular network providers, narrow-band IoT (NB-IoT), also known as LTE Cat NB1, connects to 100,000 EDs per cell. It has a battery life of 8–10 years, is affordable, has a large coverage area, and has strong network security [112]. Numerous LTE capabilities can be recycled by NB-IoT and modified to meet IoT needs. The NB-IoT uses quadrature phase-shift keying (QPSK) modulation. It uses Orthogonal FDMA for downlink data transmission and Single Carrier Frequency Division Multiple Access (SC-FDMA) modulation for uplink data transfer [112]. 3GPP has suggested NB-IoT as an end-to-end data connection method for IoT. The NB-IoT protocol is the underlying protocol that concentrates on the communication protocol layers [113].

NB-IoT has been enhanced for cellular mobile communication systems, making it more appropriate for IoT applications. In general, the base station and IoT platform built by the operator are required for NB-IoT module communications [114]. The NB-IoT uses the COAP to transmit and receive messages. The device data goes through the IoT operator platform for message forwarding. The architecture of the NB-IoT system is shown in Figure 4.3. A comparison between the specifications of the different technologies is presented in Table 4.2 [114].

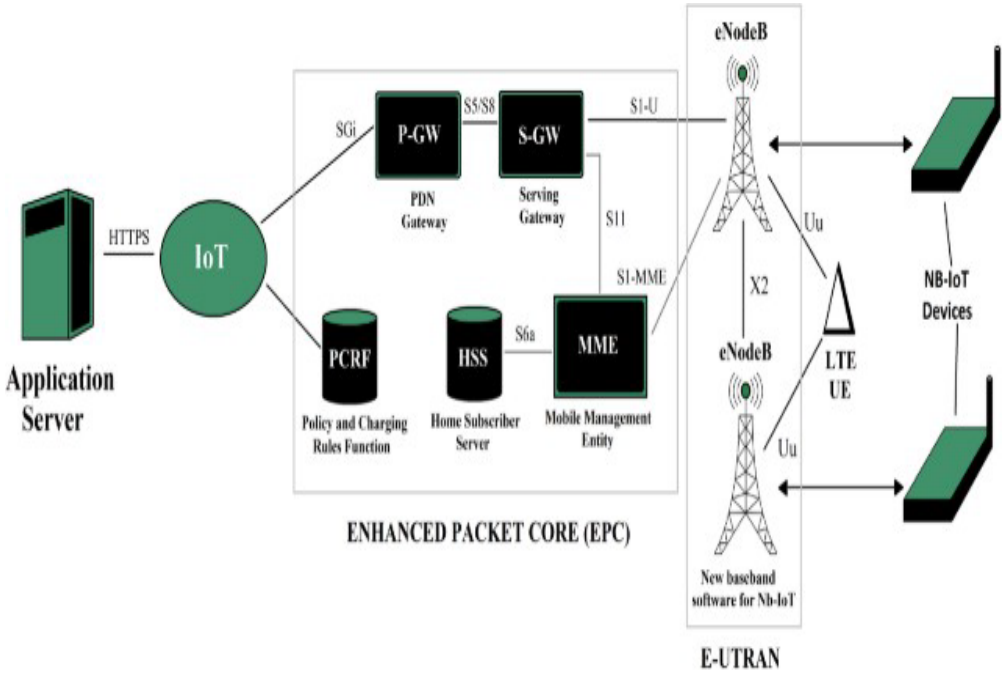


Figure 4.3 NB-IoT System Architecture [114]

Table 4.2 Comparison between different technologies

Features	Different Technologies		
	SigFox	**NB-IoT**	**LoRaWAN**
Battery Life	Ten years	Ten years	Ten years
Range	3-10 km urban, 20-40 km rural	1 km urban, 10 km rural	2-5 km urban, 10-20 km rural
Frequency	Sub GHz Ism bands	Sub GHz Ism bands	Sub GHz Ism bands
Network Topologies	Star	Star	Stars of stars
Maturity Level	Commercially usage	early stages	Some deployments
Modulation Techniques	Ultra-Narrowband	LTE -Based	Chirp Spread Spectrum
Number of devices	Only SigFox certified	100.000 per cell	No restrictions
Security	Not built-in	3GPP (128-256bit)	AES
MAC layer	ALOHA -Based	LTE-Based	ALOHA -Based
Data Rate	100bps	200bps	Between 300bps and 50bps
Maximum Payload	12 bytes for UL and 8 bytes for DL	1600 bytes	243 bytes
Adaptive Data Rate	No	No	Yes
Latency	High	High	Low
Deployment Model	Operator based	Operator based	Operator-based and Private

4.2 Long-Range and Long-Range Wide Area Network

Long-range (LoRa) refers to a spread spectrum modulation technique that was developed from chirp spread spectrum (CSS) technology and is supported by LoRa chips and gateways [115]. Semtech is famous for its physical layer. On the other hand, LoRaWAN is a MAC layer protocol created for massive public networks with a single operator. The LoRa alliance made it an open standard [115].

4.2.1 Long-Range

A patented low-power wide-area modulation called long-range (LoRa) is based on the spread spectrum of the chirp spread spectrum (CSS). It was created by Grenoble, France-based Cycleo, which Semtech, a founding member of the LoRa Alliance, later bought. Because LoRa technology is patented only, no helpful information is available. However, some technical materials from Semtech and the LoRa Alliance are accessible online, allowing users to learn about features such as time-of-air (ToA) [115].

4.2.2 LoRa- Chirp Spread Spectrum

The CSS represents the linear changes in the chirp frequency in the LoRa methods. The chirp is regarded as a linearly changing frequency signal. The symbol represents an instantaneous change in the frequency [116]. The signal is more robust to frequency-selective noise and the Doppler effect when a symbol is distributed across the spectrum bandwidth. This could result in a decrease in the spectral efficiency. Four parameters can influence the LoRa modulation [115].

- **Spreading Factor (SF)**: LoRa employs a diversity of spreading factors between 7 and 12. The SF in LoRa modulation provides a trade-off between the data rate andrange. A high spreading factor increases the range but decreases the data rateand thus increases the airtime for a message [115]. For example, a message with SF12 will take 25 s more to transmit than a message with SF7.

- **Bandwidth**: LoRa technology allows three different bandwidth ranges- 125kHz,250kHz, and 500kHz. If the bandwidth is more comprehensive, it increases the data rate but lowers receiver sensitivity. To double the data rate, we divided the receiver sensitivity by two. This makes the messages harder to decode [115].

- **Frequency or Channel**: Frequency at which the message is modulated. LoRaoperates at different frequencies, such as 433 MHz, 868 MHz, and 915 MHz. This allows receivers to receive multiple notifications with the same spreading factors simultaneously and provides them with different sub-bands [115].

- **Code Rate (CR)**: The code rate defines the amount of forwarding error correction included in the LoRa. it equals 4/(4+n), where n=1,2,3,4. A decrease in the code rate

effectively reduces the packet error rate [115].

4.2.3 Long-Range Wide Area Network

This section covers the technology stack; Figure 4.4 shows LoRa is the PHY layer. Wireless modulation was employed to establish communication links. LoRaWAN is an open networking specification the LoRa Alliance provides, offering localization, mobility, and secure bidirectional communication services [116].

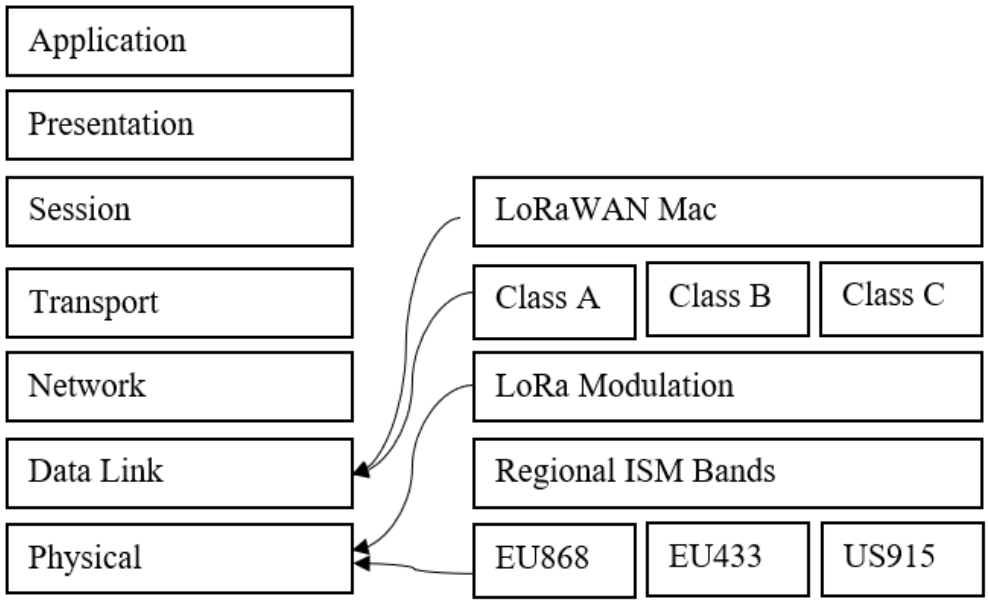


Figure 4.4 LoRaWAN technology stack [116].

4.2.4 Architecture

The LoRaWAN architecture is a MAC layer protocol for managing the link between the gateways and end nodes. The LoRa alliance maintained LoRaWAN. LoRa was first released in 2015: V1.0, later updated in 2016 and then in 2017, V1.1. LoRa technology is designed for the wireless connectivity of battery-based end nodes, either fixed or mobile. LoRa operates in the sub 1 GHz band and is deployed in stars with a star topology. A LoRaWAN network comprises end nodes, gateways, a networkserver, and an application server. Actual deployment is somewhat complex compared to different versions.

Figure 4.5 depicts the LoRaWAN network's top-level architecture. Depending on the implementation, the application, network, and join servers may be combined into a single server. The distribution of all communications (varying from 0.3 to 50 kb/s) between end nodes and gateways is based on frequency channels and data rates [117]. The trade-off between the communication range and message length informs the choice of the data rate. End nodes may use any channel at any data rate to transmit data; the channel choice depends on the region.

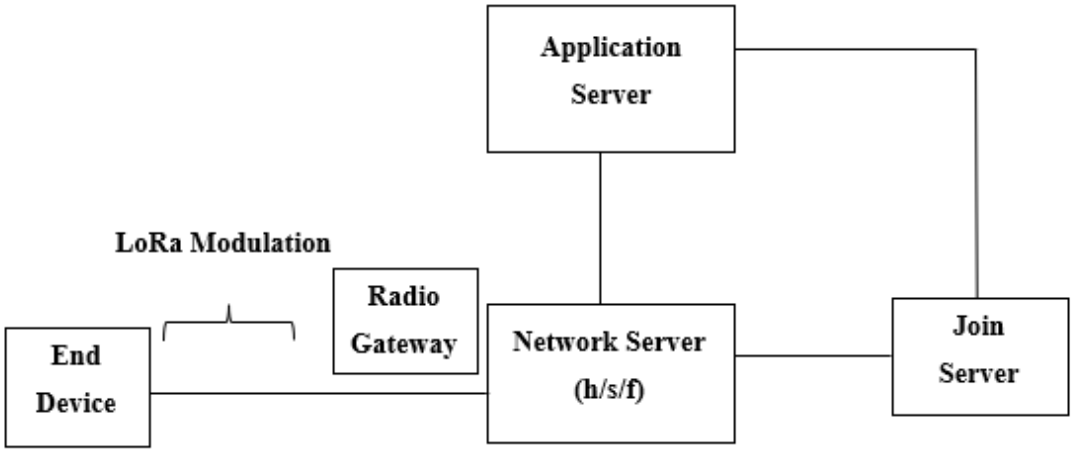


Figure 4.5 Top-level LoRaWAN Architecture [117].

4.2.5 Classes

In LoRaWAN networks, devices and applications can communicate using one of three classes of communication profiles (A, B, or C). The LoRa network classes each conduct various application functions and have optimized specifications for certain uses. The trade-off between the delay and power consumption is the primary distinction between profiles A, B, and C.

4.2.5.1 Class A operational mode

Class A is the default device class in LoRaWAN used for battery-powered sensors [117]. In Class A, each uplink message is followed by two receiving slots for the downlink messages, as shown in Figure 4.6 Class A (default) [117].. Downlink transmissions in the network are queued for the next available

receiving slot on the network [117]. There are RX1 and RX2 between the uplink and downlink transmissions, and the device is in sleep mode and consumes less power. The network will not send an uplink until the device wakes up [117].

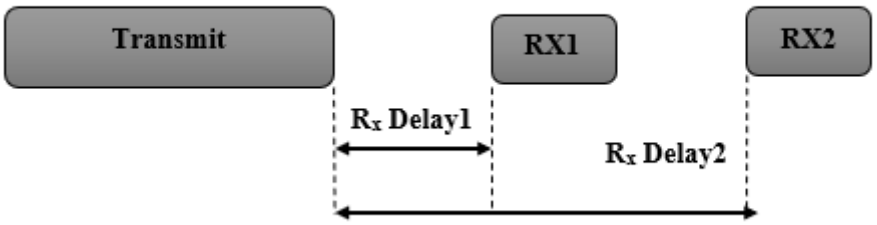


Figure 4.6 Class A (default) [117].

4.2.5.2 Class B operational mode

Class B is designed for use cases that require the opening of server-initiated receiver windows at fixed intervals [117]. It has two additional reception windows in Class B, as shown in Figure 4.7 Class B (beacon) [117]., and the gateway sends a beacon regularly to synchronize all end nodes in the network. It can open another reception window, called a ping slot [117]. These ping slots on the network server are used for downlink transmissions when required. The summary of this procedure is explained below:

- The application server queues the downlink message on the network server.
- The network server computes the next ping slot [117].
- The network server finds the best gateway based on the uplink of the device and schedules the current transmission.
- The network server queues downlink messages to the selected gateway [117].
- The gateway transmits the downlink message when the selected ping-slot time begins. Simultaneously, the device turns its receiver on and receives the downlink message.

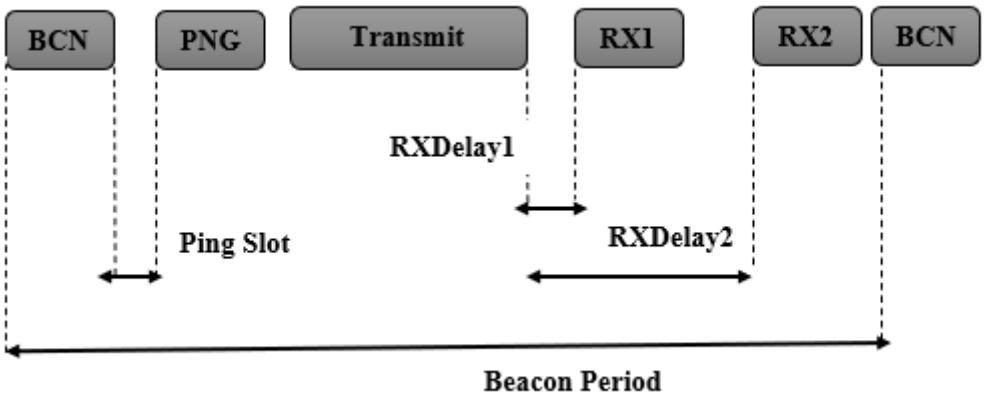


Figure 4.7 Class B (beacon) [117].

4.2.5.3 Class C operational mode

The Class C operational mode is used for the use cases to listen to uplink messages from the end devices [117]. In this mode, the end devices are not restricted by power consumption. After each uplink transmission, the end node must open the receiver window RX2 parameters, as shown in Figure 4.8 Class C (continuous) [117].. The reception window is opened using the same RX1 parameters based on RX2 [117]. Once RX1 is finished, it opens RX2 again until the subsequent uplink transmission.

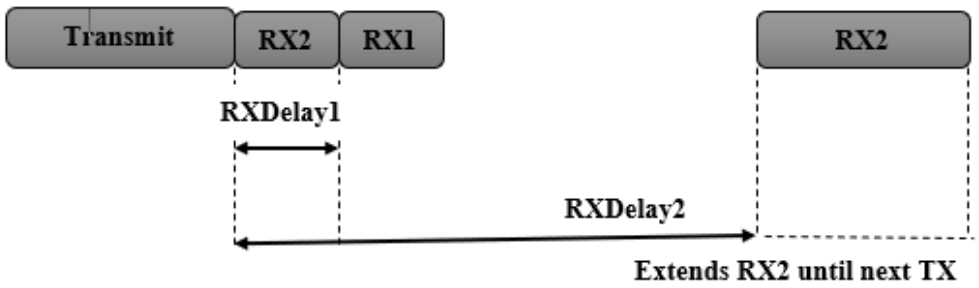


Figure 4.8 Class C (continuous) [117].

4.3 The Things Network

As the network server provider in this study, the IoT network acts as a conduit between the gateway and the application [111]. In addition to services, the IoT network offers various t are described in this section.

4.3.1 Network Server

The gateway begins receiving messages from the sensor nodes before sending data to the application [111]. Similarly, an application must choose a broadcast message for the gateway when it wants to send a message to the sensor nodes [111]. The network server enables communication between the sensor nodes and gateway, similar to LoRa and LoRaWAN. Data are routed between devices and applications by a network server [111]. The LoRa Alliance published several specifications for the endpoint authentication and MAC layer control backend interfaces of the network. The LoRaWAN network is yet to receive an official release [111].

There are a number of options available for the LoRaWAN network server, including ResIOT, LoRaServer, an open-source network server, and The Things Network, which was created by The Things Industries [111]. No new devices are accepted by The Things Network, which is the V2 console for application deployment. The V3 console is an improvement over the V2 console[111].

4.3.2 Features

IoT is a worldwide open-source community network for the Internet of Things using LoRa Technology [111]. Currently, the TTN server has more than 7600 gateways worldwide, as shown in Figure 4.9 , and gateways in Auckland, New Zealand, as shown in Figure 4.10 [111]. Any user can connect their gateways and devices to use the network freely. This is an excellent method for deploying extensive networks using different gateways.

Figure 4.9 Worldwide [111].

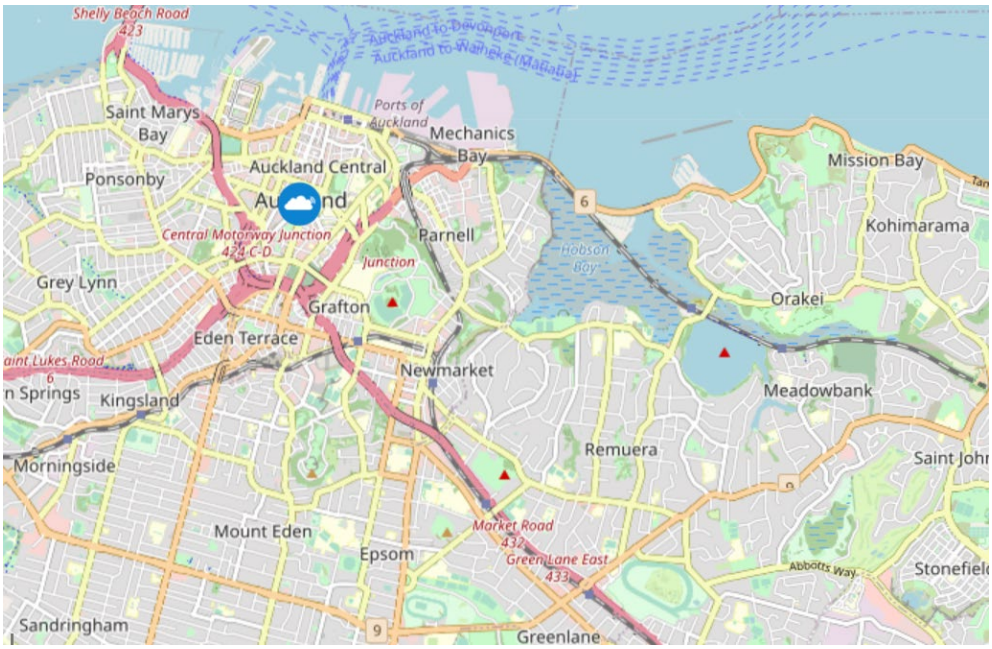


Figure 4.10 Auckland, New Zealand [111].

To use the network, users must create a TTN account on their website and register their end nodes. The TTN server provides the network key, session key, and device EUI for device registration [111]. The TTN then provides the keys to the node to encrypt and send data to the gateways. Connecting a gateway to the server is quite difficult, but it can be connected and start communication over the network. If the gateway is configured correctly, it will not create

any issues during the communication process. After the successful configuration, the gateway is online and sends a LoRa message to the TTN server, which takes care of forwards it to the correct owner. Only the message owner can access the data because they are encrypted [111]. The status of the LoRa gateways and LoRa end nodes can be monitored online on the IoT network console using the recent data received.

Any application can be created and used on the TTN server. TTN provides SDKs in Java, Python, Node.js, and many other languages that can easily communicate with servers and build applications [111]. There are many built-in applications on a TTN server that users can use. For example, "TTN Mapper" allows mapping of the coverage of Android phones.

4.3.3 Architecture

This section describes the architecture of an IoT network. It aims to perform all routing in a distributed and decentralized manner [111]. Therefore, it is split into the following responsibilities, as shown in Figure 4.11 .

- **Router**: A router is responsible for communicating with the gateways. When the gateway receives a message over LoRa, it forwards it to the router for translation into the TTN's internal protocol and sends it to the correct broker [111]. Every gateway is connected to one router alongside the GPS coordinates and gateway status to each uplink message.
- **Broker**: A broker is responsible for finding the correct handler for each uplink message [111]. When the network receives the same LoRa packet by multiple gateways,the broker duplicates the data by keeping only one redundant metadata, such as frequency and modulation, and records each gateway's signal strength, reception time, and GPs coordinates [111].
- **Network Server**: The network server monitors all the stages and states of all the devices. Before the broker forwards the uplink message to the handler, the network server receives an update regarding the state of the device [111].
- **Handler**: The handler is essential to take care of the decrypt uplink messages and encrypt downlink messages. It provides end-to-end encryption [111].
- **Discovery server**: The discovery server helps components determine the traffic route,

as TTN is a decentralized service [111]. At this point, the routers, brokers, and handlers announce themselves.

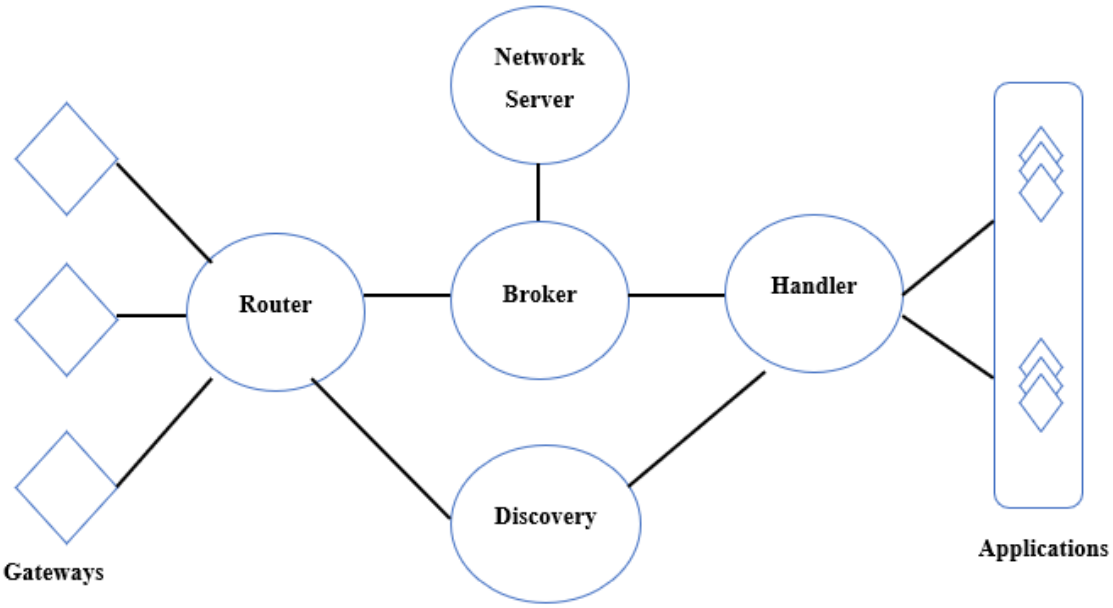


Figure 4.11 The Things Network Architecture [111].

4.4 Radio Communication Theory

Before placing the antennas, some theoretical aspects must be considered to understand the results. Some of these issues are discussed in this section.

4.4.1 Decibel and Decibel-milliwatt

The decibel and decibel-milliwatt are two different units, but they are important tounderstand and will be used in the calculations:

- Decibel (dB) was used to measure the ratio of electrical power on a logarithmic scale.
- Decibel milliwatts (dBm) is the unit of power. It can be defined as the gain relative to the electric power input of 1mW. The values expressed in dBm can thus always be translated into milliwatts.

4.4.2 Link budget and Link margin

Determine the link between two devices, known as link budget calculations. It accounts for all the gains and losses during transmission and allows one to understand which basic communication parameters can be modified and which cannot. A basic communication system can be composed of a transmitter and receiver with an antenna separated by a path. Figure 4.12 shows the components of a basic communication system used to calculate the link budget. It includes the following elements for link budget calculations:

- The transmission power is used to transmit a transmission and is expressed in dBm.
- The antenna gain describes how much power in the direction of peak radiation is transmitted; the higher the gain, the more directional the antenna.
- Cable loss occurs because of the loss of signal energy in cables and connectors. This depends on the length and type of the cable.
- The path loss includes all the losses between the two communicating antennas. This is mainly caused by free-space loss, attenuation, and scattering owing to some issuesin the Fresnel zones.
- The minimum receiver sensitivity describes the lowest signal level of the antennas and can be expressed in dBm.

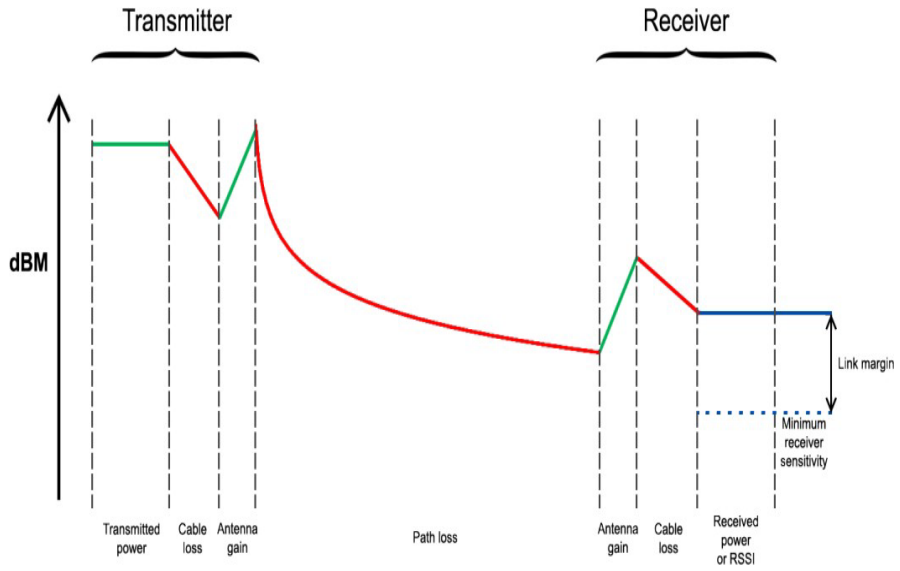


Figure 4.12 Components of a basic communication system to calculate the link budget.

4.4.3 RSSI and SNR

During the experiments, the RSSI and SNR were two basic metrics often used to measure the quality of the signal. This is important for a proper understanding. The received signal strength indication (RSSI) is the intensity of the signal received, expressed in dBm and shows the sum of all gains and losses, and corresponds to the blue line. The typical RSSI values of LoRa varied between -30 dBm and -120 dBm. This implies that it can receive a signal weaker than 10^{-12}mW. The signal-to-noise ratio (SNR) is a measurement that compares the RSSI to the background noise, which is expressed in dB.

4.4.3.1 Received Signal Strength Indicator

The received signal strength indicator (RRSI) is considered to be the measured signal power in milliwatts (dBm). This is performed to check how well the receiver can hear the signal given by the sender, as shown in Figure 4.13:

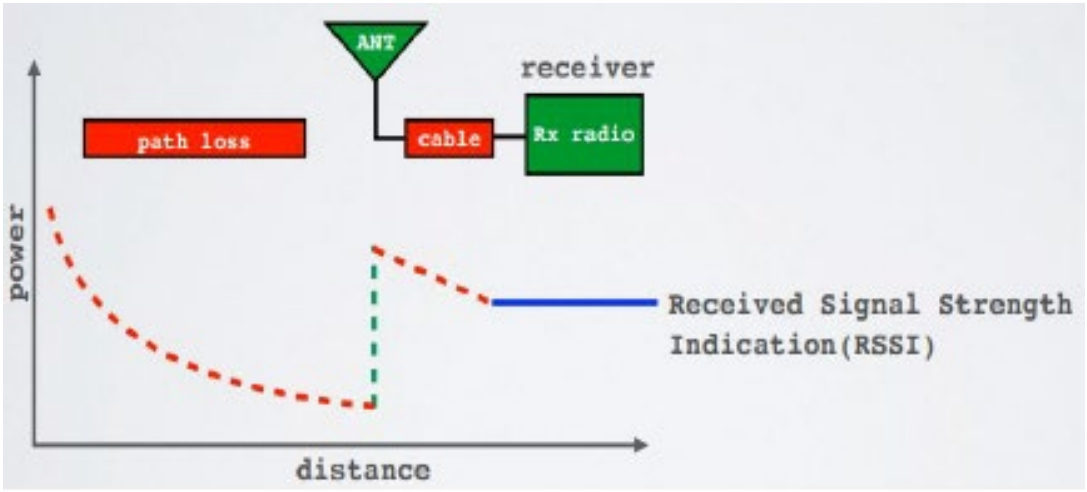


Figure 4.13 Received Signal Strength Indicator [111].

The RSSI is dBm and has a negative value. If it is close to zero, it indicates that it has a better signal strength. The typical LoRa RSSI values are:

- If the RSSI= -30 dBm, the signal is strong.
- If RSSI = -120 dBm, then the signal is weak.

4.4.3.2 Signal to Noise ratio

The signal-to-noise Ratio (SNR) is the ratio between the signal power and noise power level. This alludes to the presence of unwanted interfering signal sources that can skew the transmitted signal or prompt retransmissions. SNR is shown in Figure 4.14.

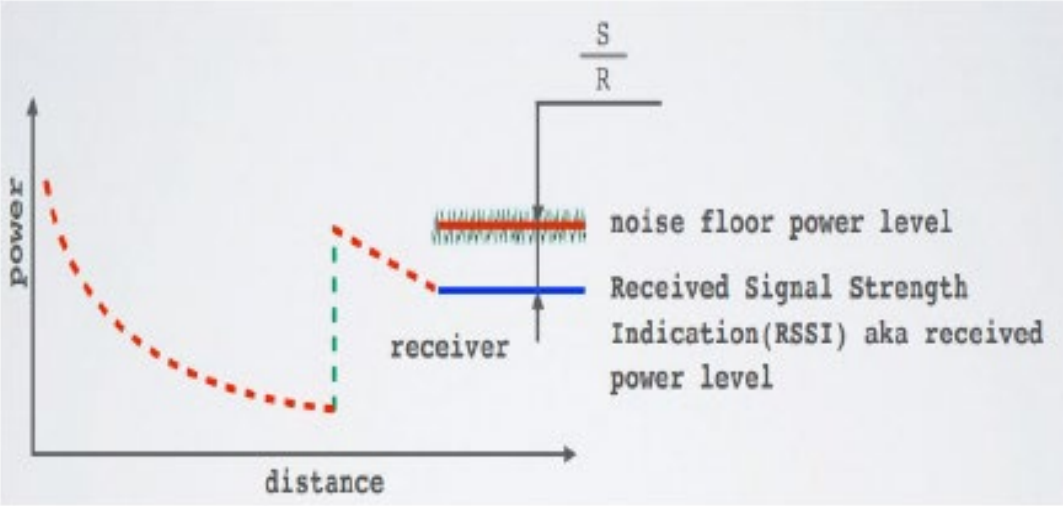


Figure 4.14 Signal to Noise Ratio [111].

Typical SNR values were between -20 dB and +10 dB. If the values are closer to +10 dB, then the received signal is less corrupted. However, LoRa transmission can demodulate signals which are -7.5 dB to -20 dB.

4.5 Air Pollution Monitoring Scope

The goal of air pollution monitoring is to gather data on the emissions, concentrations, makeup, and effects of air pollutants. It is currently commonly practiced with a focus on a specific set of issues caused by air pollution. The usual topics discussed include visibility, ecosystems, and human physical health, with an emphasis on this order. Monitoring ecological effects and various other non-health but still human-focused effects of air pollution is crucial to ensuring sustainable development, but these have received relatively little attention. The development of monitoring technologies provides opportunities to support sustainable development by concentrating monitoring efforts in these under-researched areas.

Although there is no universally accepted definition of sustainable development, the majority of technical analysts believe that steps must be taken toward it. Significantly, as long-term management or mitigation involves supporting metrics, measurement, and monitoring, the capacity to monitor has expanded alongside the types of problems that can be controlled. Sincere attempts to reduce the effects of pollutants to promote more sustainable development can often be successful, but these efforts rely on reliable monitoring data. Air pollution monitoring not only assists in informing policies and tactics but also draws attention to problems to aid in their mitigation and quantifies the effects of such efforts.

Significant gains in breadth and spatial resolution are possible owing to new technologies. Air pollution can be observed at a lower cost and with better spatial resolution using networks of microelectromechanical sensors than is feasible with traditional on-site monitoring. Global-scale air pollution observations now have better coverage and a more tolerable resolution owing to advancements in satellites.

The public is highly interested, and environmental issues are progressively being better understood. However, there is still room for improvement. Owing to the harmful effects of air pollution on the environment and human health, the need for clean air has gained international recognition. The major study in this research comprises results for nitrogen oxides, carbon monoxide, sulphur dioxide, and particulate matter based on air quality monitoring results in Auckland City. These contaminants are the most common and have the greatest effects on health. The report also mentioned $PM_{2.5}$, O_3, and metal monitoring.

Consider the fact that Such detectors are unable to detect gases in a variety of situations. As a result, several algorithms include additional fire gas properties for more precise identification. The model suggested in this study incorporates regular emissions and suggests an improved fire detection analysis to reduce the likelihood of air pollution and improve detection accuracy. With this method, the accuracy of the temperature and humidity measurements increased to 94.34% and 98.43%, respectively.

4.5.1 Star Topology

The most prevalent type of communication network is the star network, which arranges all sensor nodes around a central hub (sink), which is logically at the network's center. However,

if the central hub fails, the entire network fails. The star network extends a tree network or a clustered tree [10]. This network can be easily extended to different levels, as shown in Figure 4.15. The central node serves as a conduit for all the data traffic. Consequently, a smart central node is needed [11]. The best illustration of a start topology is a wireless personal area network (WPAN) with a smartphone connected to numerous sensors [11].

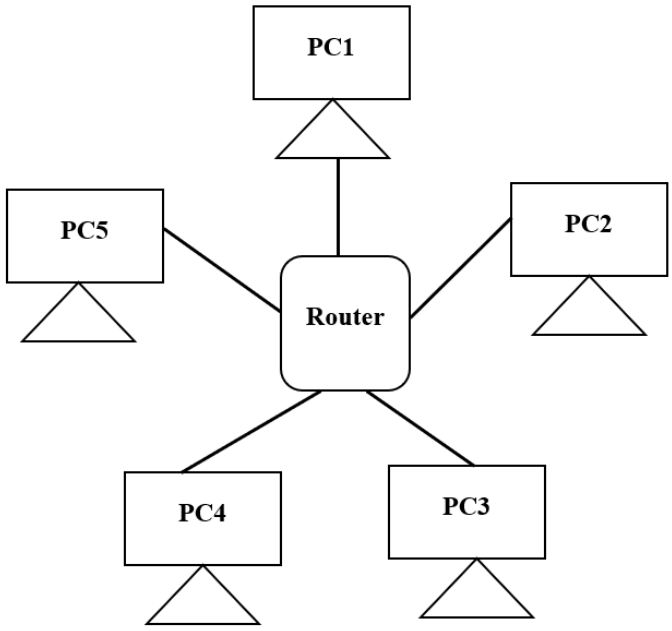


Figure 4.15 Star Topology [11]

4.5.2 Mesh Topology

A mesh network, also known as a mesh net, is a local network topology in which infrastructure nodes such as bridges, switches, and other infrastructure devices collaborate to route data from and to clients by connecting directly, dynamically, and non-hierarchically to as many other nodes as they can efficiently [11]. Mesh networks can handle heavy loads on traffic between two nodes. Nodes can simultaneously transmit data over a network. The main advantage of this network is that if a single node fails, it does not affect the entire network and can easily detect faults in the network nodes. However, mesh connections require a lot of cabling, and their maintenance and setup costs are high. There is a possibility of redundancy in the network. The wireless mesh network is illustrated in Figure 4.16.

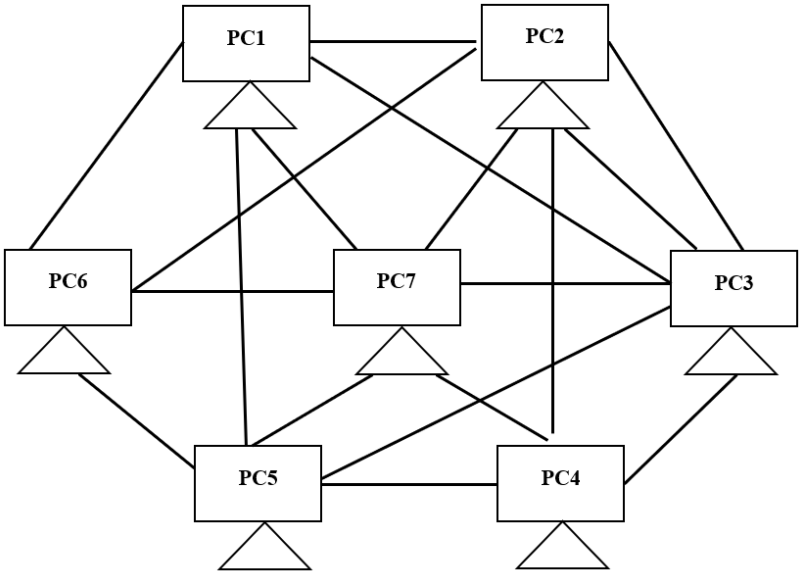


Figure 4.16 Mesh Topology [11]

4.5.3 Clustered Tree Topology

The sensor nodes in the CTT or clustered hierarchy build a set of clusters using their local data and elect a cluster head (CH) for each cluster. Owing to its low power consumption and quick responses to sensor nodes, CTT is more advantageous than star [118]. Clustered tree- the network model based on the tree is known as a cluster-tree network defined in the 802.15.4 standard. Cluster-tree networkis proven for energy efficiency. A cluster tree network exists in the form of a hierarchy [12]. During transmission in this topology, there is a diversity of clusters present, which can communicate using multi-hop routes. Each cluster in the network has a cluster head. The PAN coordinator handles the CTT as shown in Figure 4.17 [12]. The effectiveness of a network is based on network parameters such as scalability, data rate, and clusterdimensions [13].

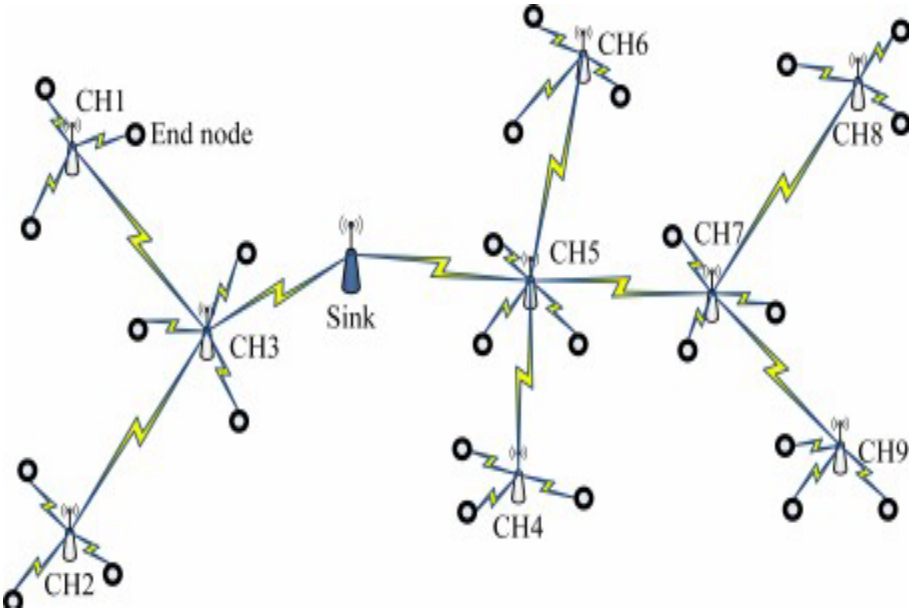


Figure 4.17 Tree Topology [12].

4.6 Hardware

The most difficult part is the selection of hardware devices for network deployment. During this process, researchers must consider different parameters such as cost, size, and compatibility with network systems. This section explains about hardware used in the network.

4.6.1 Gateway

A gateway is a network node used in the network for wired or wireless communication that has the ability to connect two networks using transmission protocols. The gateway used in the network functions as the sender and receiver. During this study, two types of gateways were used, as explained in the following section.

4.6.1.1 SparkFun LoRa Channel-1 Gateway

When the research began for air pollution monitoring using different sensors, the first gateway used was the Sparkfun Channel-1 Gateway, as shown in Figure 4.18. The LoRa Gateway Channel-1 can acts as a gateway or device, but not simultaneously. The gateway setup must

be ensured according to user requirements, where another LoRa device can listen and transmit packets to each other.

Figure 4.18 LoRa Sparkfun Channel-1 Gateway

Programming LoRa Gateway Channel-1, it requires a micro-B USB cable and a computer with an Arduino IDE installed. The use of a 915 MHz radio on the gateway requires a connection with the antenna. There are two choices for the antenna connections.

- The length of the solid wire was cut to approximately 3" and soldered to the gateway.
- Use a 915 MHz antenna with a UFL connector.

Features of LoRa Sparkfun Channel-1 gateway.

- Wi-Fi, BT+BLE microcontroller and integrated PCB antenna.
- Frequency Range: 868/915 MHz
- Spreading factor: 6-12
- SPI control interface.
- UFl antenna connector
- Reset and ESP32 pin0 buttons.

- 14 GPIO ESP32 pin-breakouts
- Power and user LEDs
- Qwiic connector
- CH340C USB-to-serial interface
- Micro-B USB connector for power and programming

4.6.1.2 LPS8 LoRa Gateway

The second issue to select the gateway for LoRa network deployment. There are multiple companies in The Things Industries that provide diverse gateways, such as LPS, Kerlink, and Lorix. The Things industry is the creator of The Things Network, which proposes commercial gateways and prices range from 500e to 1500e depending on their functionalities and weather resistance. The suggested gateways have the advantages of an easy setup and are ready for use in industrial applications.

It is also possible to create a gateway using different devices, assemble them, and configure the gateway. This is a slightly cheaper option, and the goal of this research is to evaluate the LoRa and LoRaWAN performances. The Things Industries have been released "The Things Indoor Gateway" and "The Things Outdoor Gateway."

For this study, the LPS8 gateway was used to monitor the LoRa Performance and Pollution levels in the air. The LPS8 is an open-source LoRaWAN gateway. It can build a bridge from a LoRa wireless network across Wi-Fi or Ethernet to an IP network. Users can communicate data across large distances using LoRa wireless technology at low data rates. The Semtech packet forwarder is used by LPS8, which is compatible with the LoRaWAN protocol. It comes with the SX1308 LoRa concentrator for LoRaWAN, which offers ten programmable parallel demodulation paths. Although LPS8 comes with preconfigured standard LoRaWAN frequency bands for use in various nations, users can also modify them according to their own. The Figure 4.19 shows the LPS8 architecture-

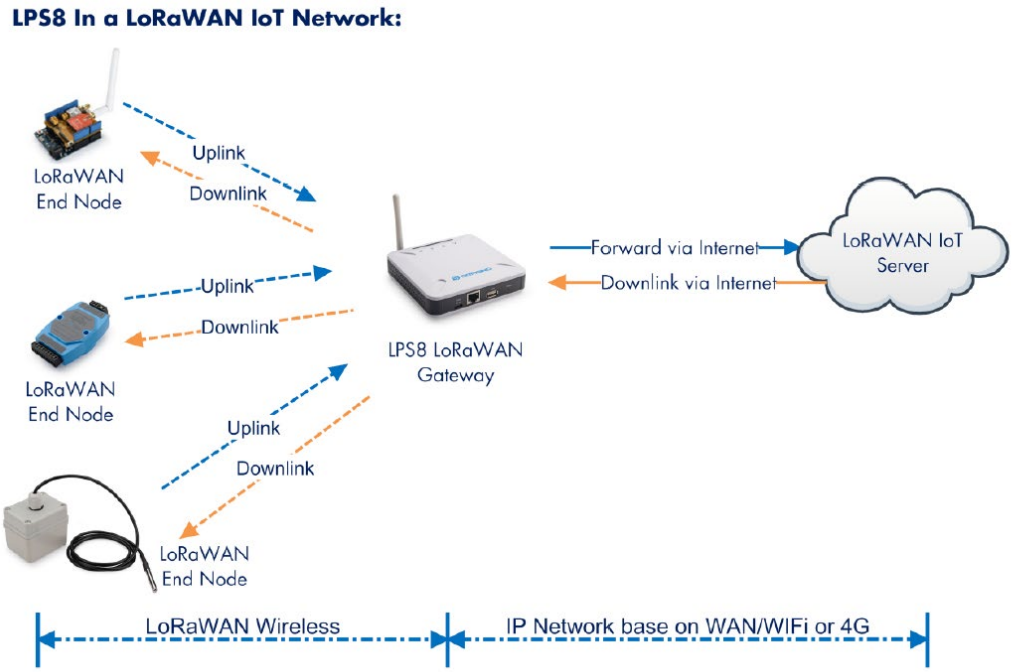


Figure 4.19 Architecture of LPS8 LoRa Gateway

4.6.1.3 Hardware system

When setting up the hardware system for an LPS8 Gateway, you'll need various components to ensure it operates efficiently. LPS8 LoRa Gateway has the below configurations for the hardware:

- 400Mhz ar9331 processor
- 64MB RAM
- 16MB Flash

4.6.1.4 Interface

This interface is used to manage various aspects of the gateway, including network settings, LoRaWAN parameters, and system information. Interface requirements for the LPS8 Gateway are as below:

- 10M/100M RJ45 ports x 1
- WiFi: 802.11 b/g/n
- LoRaWAN wireless
- Power Input: 5V DC, 2A, Type C
- USB 2.0 host connector x 1

4.6.1.5 WIFI Spec:

Wi-Fi, which stands for "Wireless Fidelity," refers to a set of wireless networking technologies that allow devices to connect to the internet and communicate with each other without the need for physical cables. Wi-Fi specifications are defined by the Institute of Electrical and Electronics Engineers (IEEE) and are categorized by various standards, each denoted by a letter or combination of letters (e.g., 802.11n, 802.11ac, 802.11ax). For the LPS8 Wi-Fi specs are:

- IEEE 802.11 b/g/n
- Frequency: 2.4-2.462GHz
- Tx Power: 11n TX power: mcs7/15:11db mcs0:17db
- Wi-Fi sensitivity: 11g 54M: -71dbm

4.6.1.6 LoRa Spec:

The LPS8 Gateway is a LoRaWAN gateway that is designed for long-range, low-power wireless communication for IoT (Internet of Things) devices. LoRaWAN is a protocol that operates in the sub-gigahertz unlicensed radio bands and is known for its long-range

capabilities and low power consumption. When it comes to LoRaWAN specifications in the context of the LPS8 Gateway, several key features are given below:

- Up to -140 dBm sensitivity
- 70 dB CW interferer rejection at a 1 MHz offset.
- Able to operate with negative SNR.
- Emulate 49 x LoRa demodulators and 1 x (G)FSK demodulator.
- 10 programmable parallel demodulation paths
- Opensource openWrt system
- Managed by Web GUI, SSH via WAN, or WiFi.
- Remote access with reverse SSH
- Emulates 49 x LoRa demodulators.
- LoRaWAN gateway
- Pre-configuration to support regional LoRaWAN settings.
- Allow customization of LoRaWAN regional parameters.
- Support level log in.

Overview of the LoRa gateway is shown in Figure 4.20 and Figure 4.21 shows the LPS8 applications.

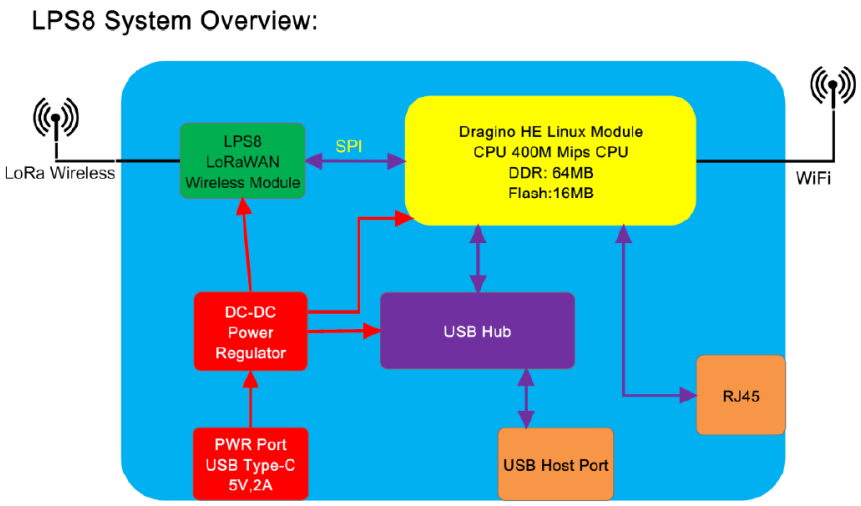


Figure 4.20 LPS8 System Overview

Figure 4.21 LPS8 Applications

4.6.1.7 LPS8 connectivity

To connect the LPS8 gateway, it is necessary to determine its IP address of the LPS8 gateway. Some steps must be followed. LPS8 gateway can connect with Wi-Fi as shown in Figure 4.22 or internet connection in different ways.

Figure 4.22 Connecting using Wi-Fi

When it is the first boot of LPS8, it generates a WiFi network called Dragino-xxxxxx, as shown in Figure 4.23, with the password dragino+dragino.

Figure 4.23 Showing Network after First Boot

The user can use a PC to connect to the Wi-Fi connection. The PC generated an ab IP address of 10.130.1. xxx and LPS8 has the default IP 10.130.1.1. This is an alternative to connect the

LPS8 Ethernet port to the router, and LPS8 obtains an IP address from the router, as shown in Figure 4.24 Then, the router's management portal is checked, where the users can find the IP address assigned to LPS8 by the router. Users can also connect to the IP.

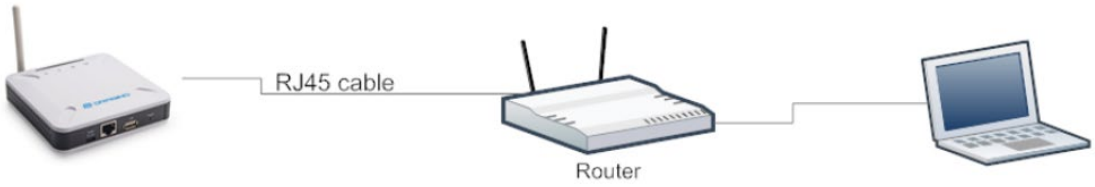


Figure 4.24 Connect using Ethernet with DHCP IP from the router.

If LPS8 is already connected to the router using the WiFi connection, the user can use the WIFI IP to connect LPS8, as shown in Figure 4.25.

Figure 4.25 Connect to WiFi with DHCP IP from the router.

After successful connection, it is necessary to configure the Web UI. To configure this, open browser, and type LPS8 IP address that is 10.130.1.1, then it will show the user with login interface of LPS8 as shown in Figure 4.26.

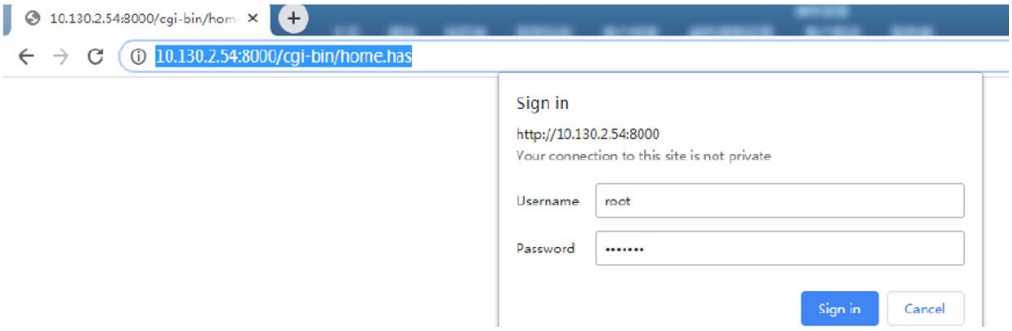


Figure 4.26 LPS8 Login Interface

Access to the web interface requires the usernames and passwords provided in the LPS8 user manual. LPS8 supports flexible network setup for different environments. The network setup included the following steps:

- WAN port internet mode
- WiFi client mode
- WiFi AP mode

Users can use the WAN port to access the internet. When LPS8's WAN port is connected to the upstream router, LPS8 receives an IP address from the router and accesses the Internet port through the upstream router. The network status can be checked as shown in Figure 4.27.

Figure 4.27 LPS8 LoRa Gateway System Overview

In WiFi client mode, LPS8 acts as a client and obtains DHCP from an upstream router using WiFi. Go to the settings System then WiFi, and then click on WiFi WAN Client settings, as shown in Figure 4.28.

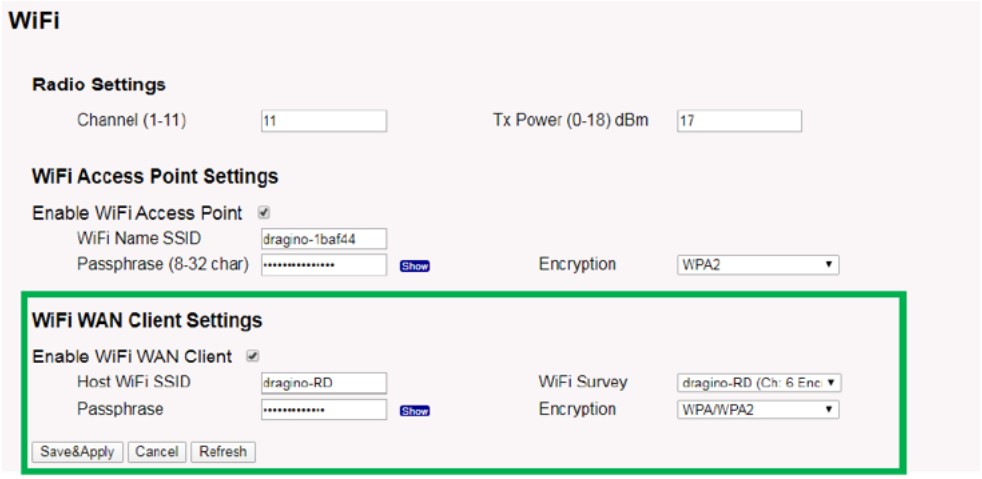


Figure 4.28 WAN Client Setting in LPS8

From the WiFi survey, select the WiFi AP and add the host SSID and pass phrase, then click on save and apply. To check successful Internet connections, go to the homepage and show a green tick if the network has an Internet connection. If not, go and check the settings again. Successful network connections are shown in Figure 4.29.

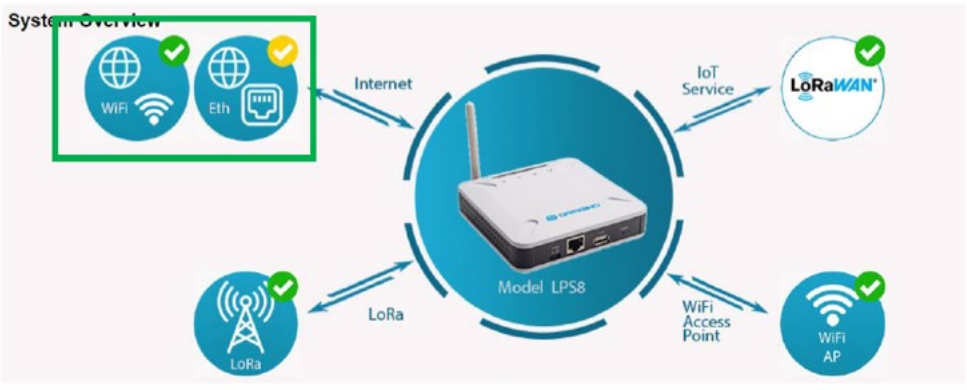


Figure 4.29 Successful Network Connection

The next step was to create a gateway on the TTN V3 server. Every gateway has their unique id and that can be found on LoRaWAN configuration page as shown in Figure 4.30.

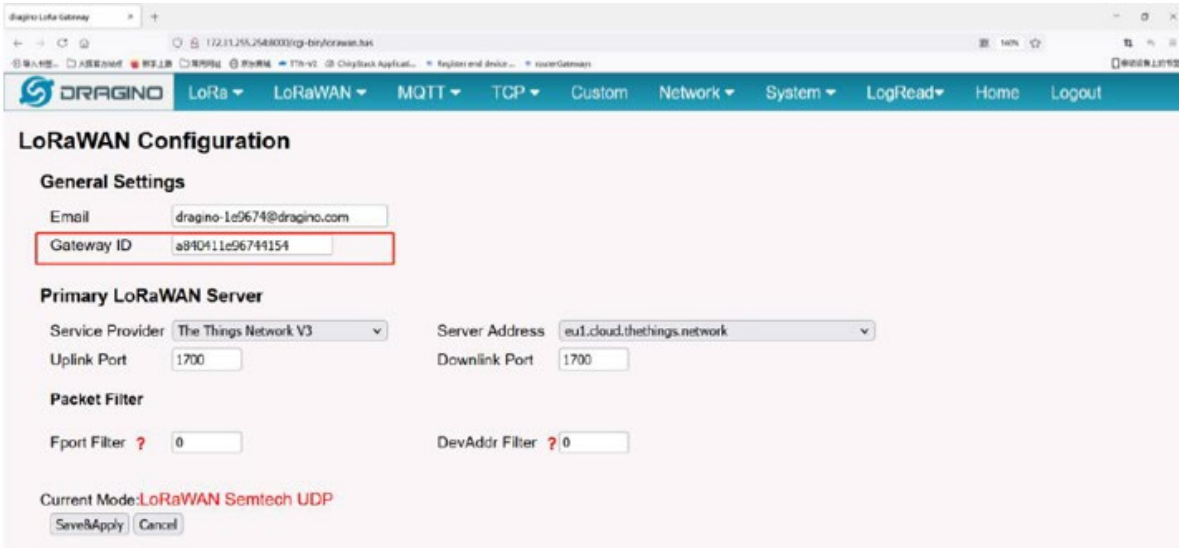


Figure 4.30 LoRaWAN Configuration

Sign-up or sign-in to the TTN server using a web browser as shown in Figure 4.31.

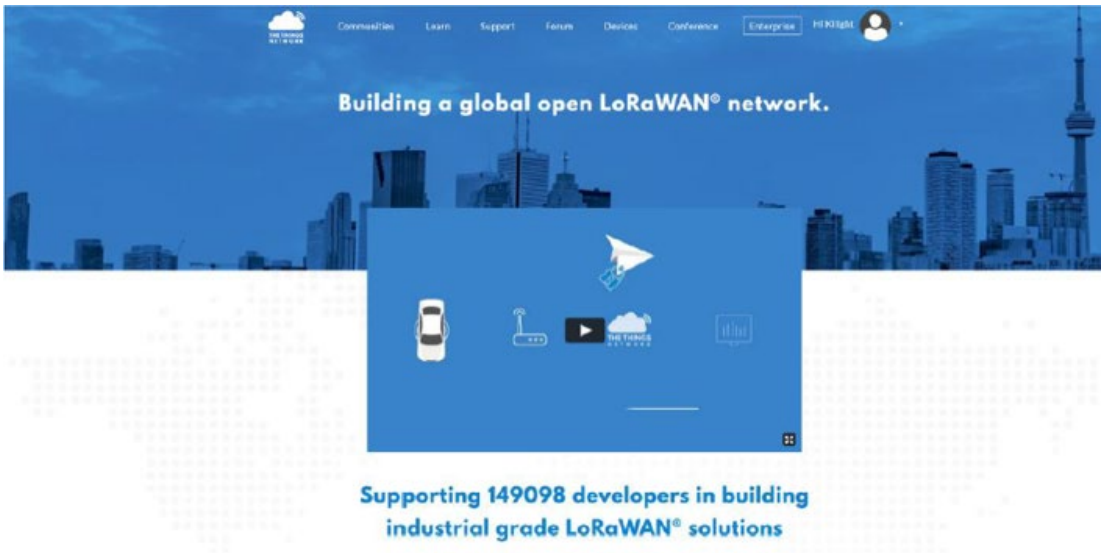


Figure 4.31 TTN Server Web Browser

A cluster on which the user wants to work is selected according to the location, as shown in Figure 4.32.

Figure 4.32 Cluster Selection

Subsequently, a gateway was created on the TTN server. Click on the gateway icon and then add the gateway as shown in

Figure 4.33.

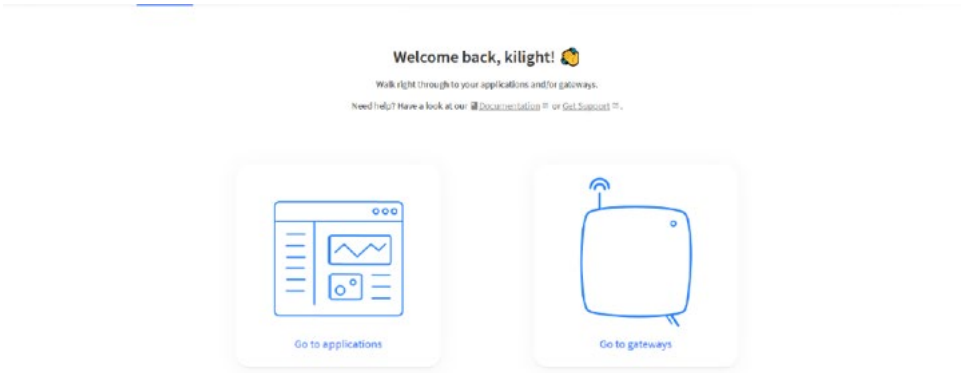


Figure 4.33 Creating Gateway

After clicking on the gateway, the following page was shown to add information. Figure 4.34 shows the gateway ID and server address required according to the location.

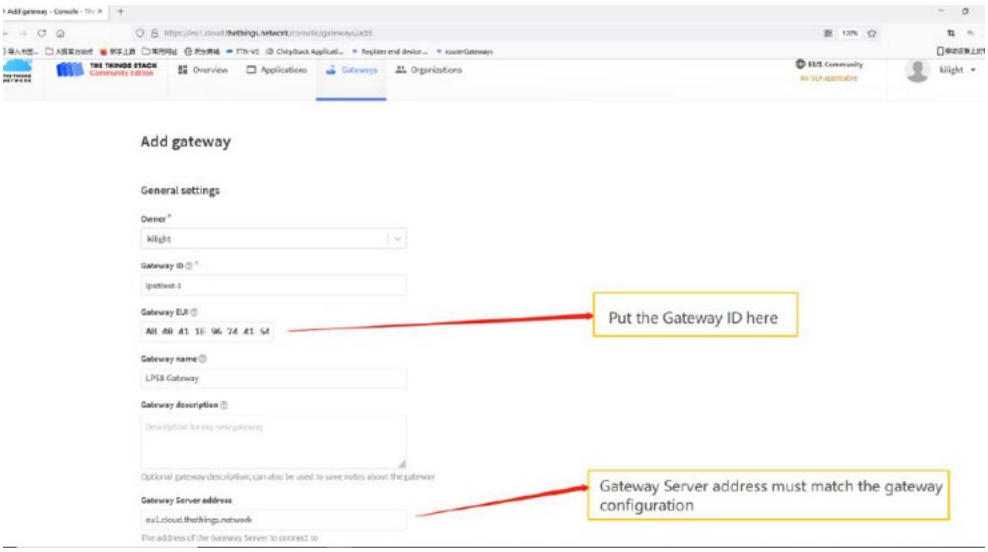


Figure 4.34 Gateway ID and Server address

Select the right frequency for the gateway when configuring the TTN server, as shown in Figure 4.35.

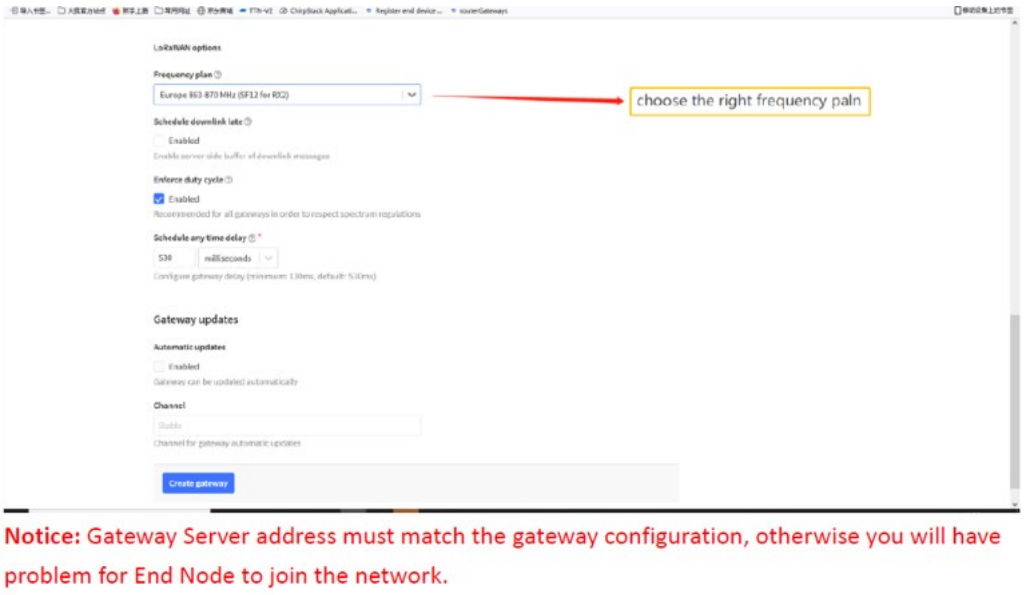


Figure 4.35 Frequency Plan Selection

After creating a gateway on the TTN server, it needs to configure on the TTN V3, but the user must make sure about the Internet connection. Select the right service provider and click on save and apply as shown in Figure 4.36.

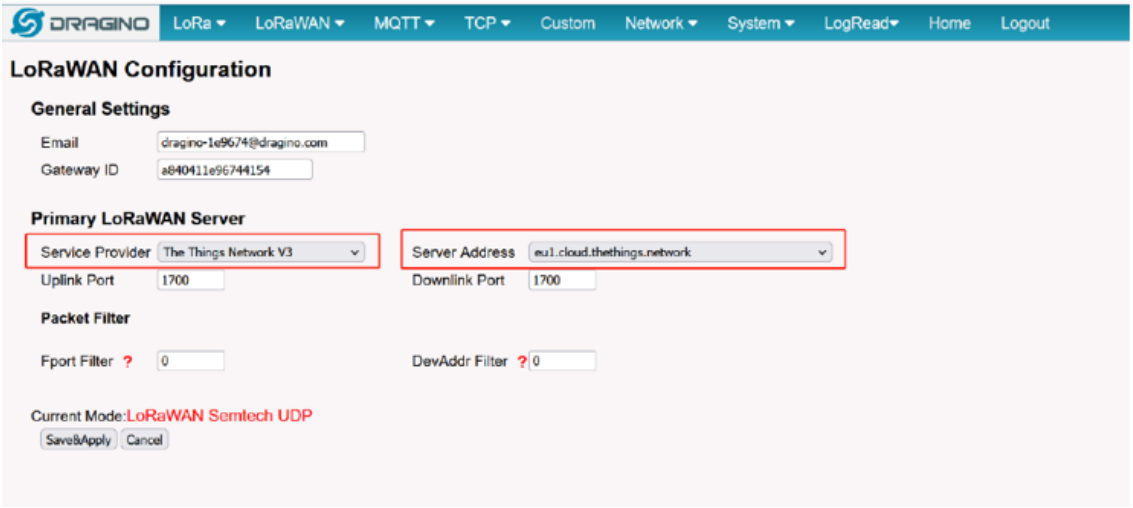


Figure 4.36 Service Provider Selection

Next, the frequency plan is configured as shown in Figure 4.37, in LPS8 that should be matched to the node the user is using for network deployment, so that it can receive packets from the LoRaWAN sensor.

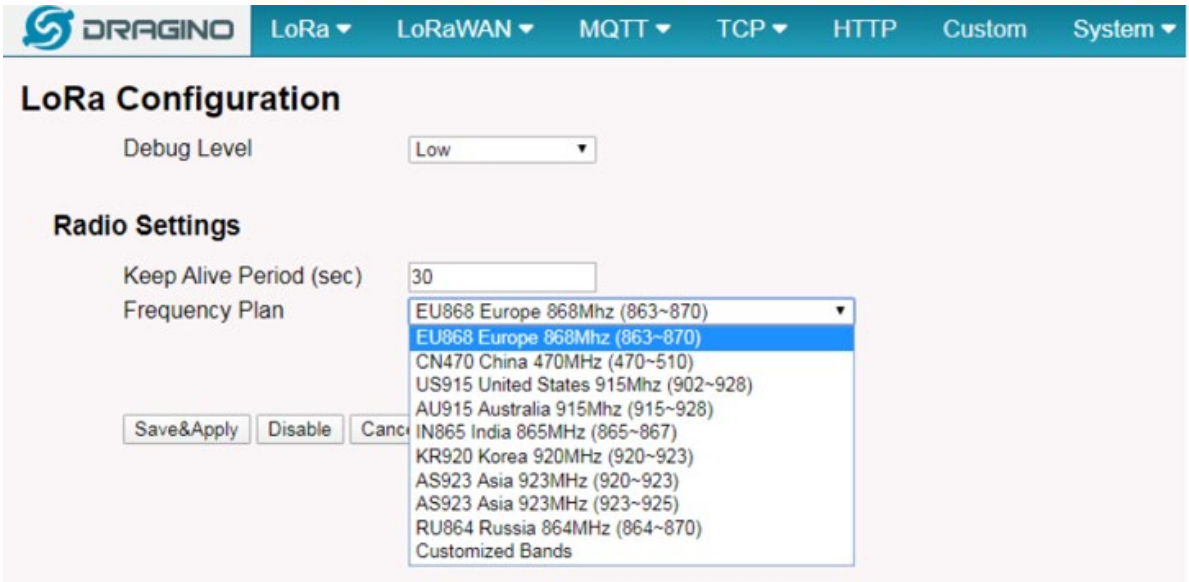


Figure 4.37 LoRa Configuration

4.6.2 End Nodes

In the network deployment, all LoRa end devices have a microcontroller, a LoRa radio module, and an antenna, as shown in Figure 4.38.

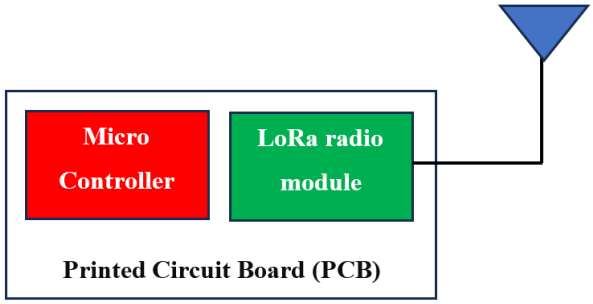


Figure 4.38 LoRa Node

The SX1276/SX1278 based LoRa node was designed for use in wireless sensor network applications, including building automation and monitoring in smart cities. When using a low-cost crystal and bill of materials, the LoRa ESP32 can achieve a sensitivity greater than -148 dBm because of Semtech's proprietary LoRa modulation technique.

The LiLyGo LoRa Module with chip SX1278 works as a sender and receiver at a signal of -9 dB. This was because the antenna placement was very close to it. However, if the antenna is moved away from it, it will start showing a weak signal strength of -124 dB and it will SOP receive information. It is important to know about ESP32, which has three chips: flash, memory, and Espressif, which is the Tensilica processor. It also has a USB to serial Ttl of silicon labs and the fourth chip of SX1278.

To understand network usage, LoRa chips communicate with each other to measure different parameters, such as pressure, position, temperature, and humidity, using the IoT in the LoRa network. For better communication, the LoRa network has a hub, which is the gateway that receives all signals, as shown in Figure 4.39. It then enters the TCP/IP network LoRaWAN SSL and follows the server side.

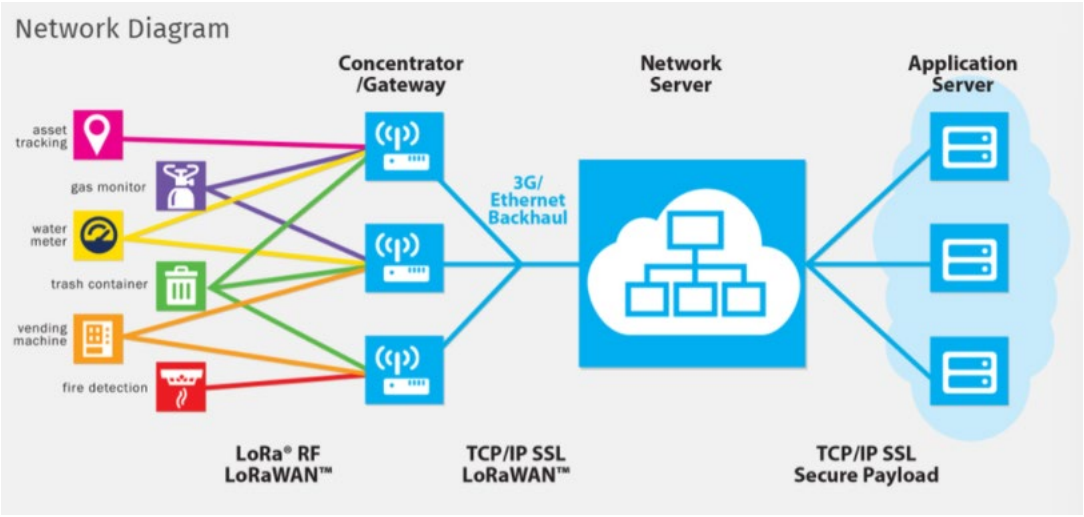


Figure 4.39 LoRaWAN Network Diagram

A protocol standard called LoRaWAN is included in the LoRa technology created by the LoRa alliance. It enables low-power and wide-area communication between LoRa sensor nodes and gateways by utilizing the unlicensed radio spectrum in Industrial, Scientific, and Medical (ISM) channels. It follows the accepted methodology for developing an LPWAN that enables the development of both public and private IoT networks and can be utilized everywhere utilizing various hardware and software. The LoRa network can provide the exact position and is bidirectional, secure, and interoperable.

Sensors in the LoRa sensor node ESP32 LoRa can also be easily designed for safety purposes, such as gas, temperature, and humidity. The microcontroller is used as the sender and receiver and communicates over the network. ESP32 reads the analog values and indicates the pollutant levels at the analyzed location. Then, the pollutant levels are shown on the server, and graphical representation is shown by the ubidots. The pin configuration of the LoRa sensor node is shown in Figure 4.40.

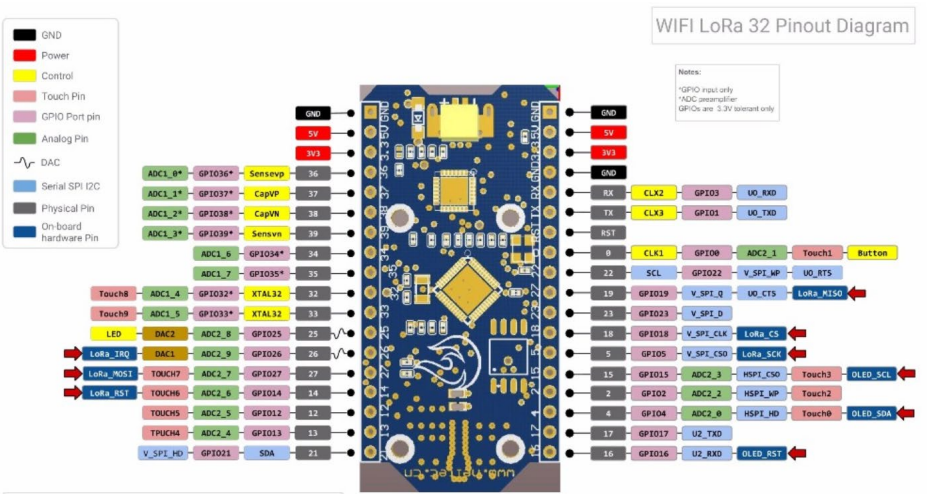


Figure 4.40 Pin Configuration of LoRa ESP32 TTGO Board [119]

Figure 4.41 is for the LoRa Module connections with different sensors (DHT22, CO_2, PM sensor, and dust sensor). For data transmission, the network used GPIO pins 36, 37, 38, and 4. Explanation of the working principle and how much time they consume to prepare the data and send them to the gateway. The schematic layout of the sensor controller unit is given in Appendix B.

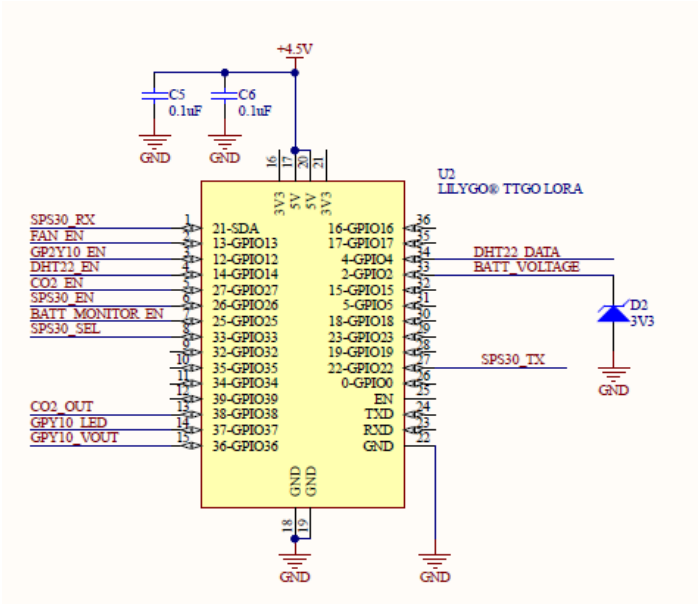


Figure 4.41 Pin Connectivity of sensors.

4.6.3 Sensors

A sensor (or transducer) converts physical parameters such as temperature, humidity, and speed into a signal that can be monitored electronically. The key drivers of the rise in low-cost sensors are aggressive marketing efforts and technological improvement. The key drivers of the rise in low-cost sensors are aggressive marketing efforts and technological improvement [82]. Low-cost sensors are advantageous for large-scale pollution monitoring because they use less power and have smaller packaging. According to research or project requirements, diverse sensors are available in the market. This section covers only the primary and the most commonly used sensors.

4.6.3.1 CO_2 sensors

CO_2 in air is considered the main pollutant that affects human health and the environment. This can be monitored with the help of MG811, MQ2, and NDIR sensors implemented on a sensor board with basic network requirements. The MG811 sensor is a board sensor component that amplifies the output signal and on-board heating circuit. MG811 is highly sensitive to CO_2 and less sensitive to alcohol and CO. It has four interlock connectors and interlock cables in a compact package. The specification of the sensor is given in Appendix C. The MQ2 sensor is compatible with monitoring all gases such as LPG, methane, carbon dioxide, alcohol, hydrogen, and smoke. The NDIR system (non-dispersive infrared gas detector) shown in Figure 4.42 work on a broadband IR source, and this target the gas will absorb more IR source as the concentration of gas increases. NDIR sensors have some features such as increasing gas tube length, optimizing the optical path, reflectively, and more sensitive and improved IR source performance.

Figure 4.42 CO_2 sensors

4.6.3.1.1 Working of CO2 sensor

The flowchart shown in Figure 4.43, describes the operation of the CO_2 sensor with the network server and how it transmits and communicates with the network gateway and server. Sensor communication with the network requires a pin setup that requires an analogue pin as the sensor reads analogue values. Once the pin setup was complete, the loop was started for data transmission. The CO_2 sensor took 3 min to warm up and then started reading the analogue values. If the data are not ready, it takes time to read; otherwise, collect the data and insert it into packets for transmission. When the LoRa packet is ready to be transmitted, it is transmitted to the LoRa gateway. It can also be displayed on an OLED if enabled. After data transmission, it is in sleep mode for the assigned time. Every transmission follows the same loop and is transmitted over the network.

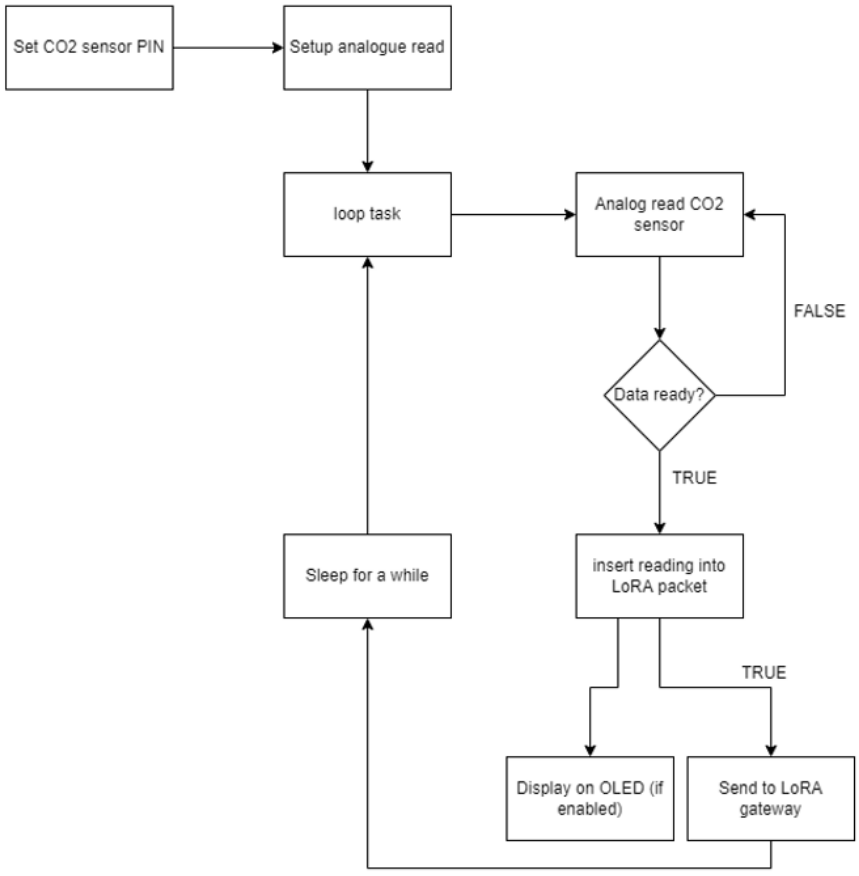


Figure 4.43 CO_2 sensor working.

4.6.3.2 DHT22

Temperature and humidity readings were collected using a digital-output capacitive-type relative humidity and temperature sensor/module (DHT22), as shown in Figure 4.44. The DHT22 sensor output a calibrated 8-bit digital signal. This sensor was used for the digital signal-collecting technique and humidity sensing technology, ensuring its reliability and stability. The specification of the sensor is given in Appendix C. The sensing elements are related to an 8-bit single-chip computer. Every sensor of this model was temperature-compensated and calibrated in an accurate calibration chamber.

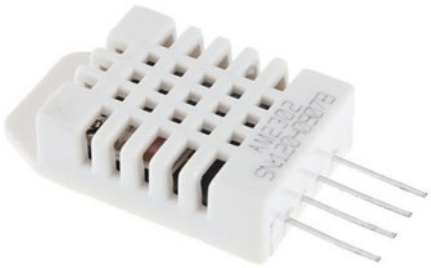


Figure 4.44 DHT22 sensors

4.6.3.2.1 Working of DHT22

The flowchart shown in Figure 4.45, first it shows the sensor pin and DHT selection (DHT11 or DHT22). Subsequently, the sensor initializes the data collection process. If the sensor connections are correct, it will show success and begin a loop task for data collection. Once it starts the loop task, it starts reading sensor values for temperature and humidity and creates LoRa packets to transmit over the network. If the OLED is enabled for the ESP32 board, it displays the data on the OLED and sends the data to the LoRa gateway. After sending data to the gateway, it will be in sleep mode, as assigned. If the initialization of the DHT22 sensor fails, then it will report an error or it will start showing 99.95 values for temperature and humidity.

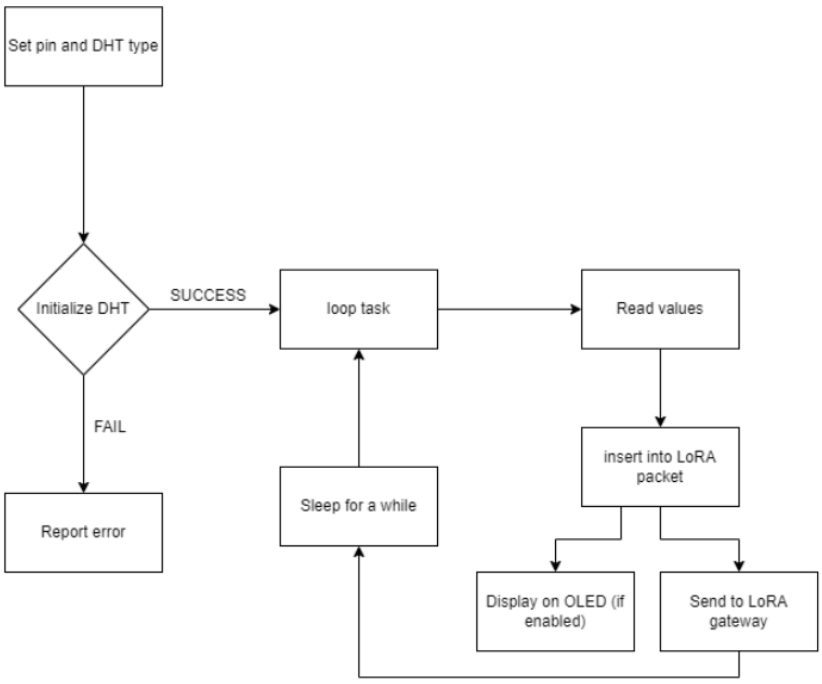


Figure 4.45 Temperature and Humidity sensor working.

4.6.3.3 Dust Sensor

The GP2Y1010AU0F is a dust sensor that uses an optical sensing system. In this device, an infrared emitting diode (IRED) and phototransistor were placed diagonally. Extremely small particles, including cigarette smoke, were successfully detected. The operation of the dust sensor is straightforward. When the sensor is powered, the LED begins to emit light that falls

on the lens, and the detector, which is positioned diagonally opposite the LED, detects the light. The Analog pin of the sensor measured the output voltage when dust was present. The specification of the sensor is given in Appendix C. The dust sensor is a low-power gadget that runs from 3.3V to 5V shown in Figure 4.46.

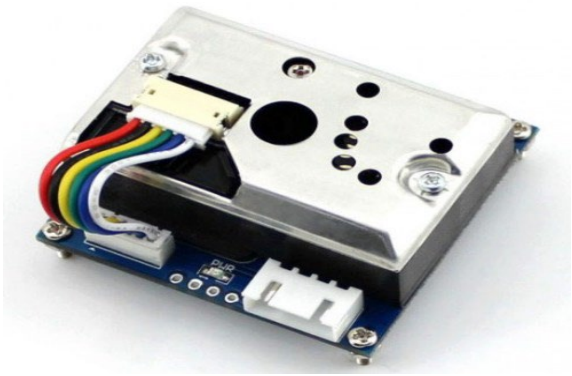


Figure 4.46 Dust sensors

4.6.3.3.1 Dust sensor working

The flowchart shown in Figure 4.47, describes the working of the CO_2 sensor with the network server and how it transmits and communicates with the network gateway and server. Sensor communication with the network requires a pin setup that requires an analogue pin as the sensor reads analogue values. Once the pin setup was complete, the loop was started for data transmission. The dust sensor starts sending the sensor values after a few seconds. If the data are not ready, it takes time to read; otherwise, collect the data and insert it into packets for transmission. When the LoRa packet is ready to be transmitted, it is transmitted to the LoRa gateway. It can also be displayed on an OLED if enabled. After data transmission, it is in sleep mode for the assigned time. Every transmission follows the same loop and is transmitted over the network.

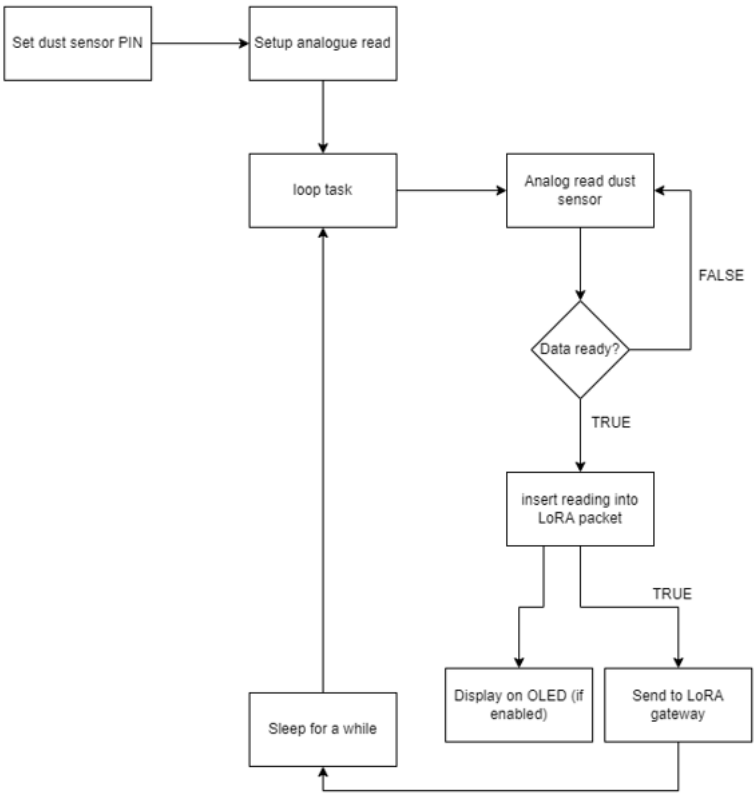


Figure 4.47 Dust Sensor Working

4.6.3.4 PM Sensor

One technological advance in optical particulate matter (PM) sensors is the SPS30 sensor for particulate matter (PM). The unique contamination resistance technology of Sensirion is made possible by the measuring concept of PM sensors, which is based on laser scattering. Precision measurements are made possible by the high-quality long-lasting components of this technology. The specification of the sensor is given in Appendix C. The SPS30 particulate matter (PM) sensor is a small, high-quality optical particle sensor that measures particulate matter using laser scattering and cutting-edge contamination resistance technology.

A dust sensor that had previously been utilized instead of SPS30 was used for pollutant level research and monitoring [117]. Owing to the inaccuracies and dust level monitoring qualities, the sensor was replaced with SPS30, as shown in Figure 4.48. includes high-quality, long-lasting components that allow for precise measurements and ten years of throughput. The five-pin interface of the SPS30 sensor can connect with either the UART or the I2C bus. to track

particulate matter using a network that was created. The sensor operates using UART according to the requirements of the TTGO LoRa32 SX1276 OLED Board. Working of particulate matter sensor is explained in Figure 4.49.

Figure 4.48 SPS30 Sensor for measuring Particulate Matter

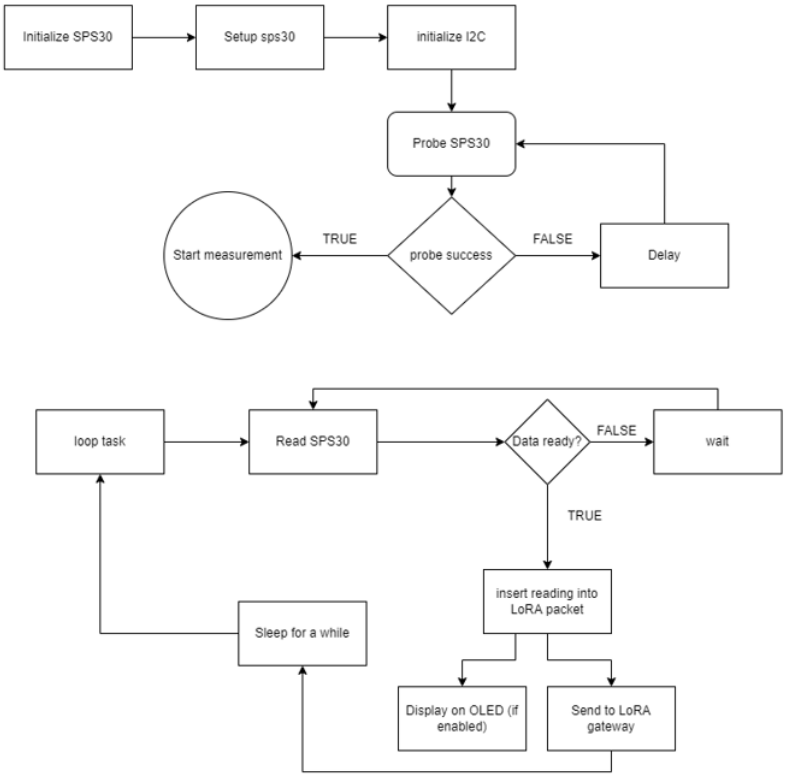


Figure 4.49 SPS30 sensor working.

4.6.4 Impact of Hardware

The hardware used in this research has some impact on the performance, efficiency, reliability, and features of technology systems. Impact of hardware is explained below in terms of:

- **Sensor Accuracy:** The quality and accuracy of the sensors or measurement devices used in data collection directly impacts the reliability of the obtained data. High-quality sensors with precise measurements are essential for accurate data recording.
- **Connectivity:** The hardware's connectivity options, such as LoRa, Sigfox, NB-IoT, Wi-Fi, etc., determine how well the data can be transmitted from the sensors to the data collection system. The range and stability of the connectivity influence data collection in different environments.
- **Power Consumption:** Hardware with low power consumption is crucial, especially for remote or battery-powered sensors. Efficient power management ensures that sensors can function for extended periods without frequent maintenance.
- **Data Storage Capacity:** The hardware's data storage capacity affects how much data can be stored locally before transmitting it to the central server. Sufficient storage capacity is essential, especially if there are temporary communication disruptions.

4.7 Software

Software in network deployment works as a data storage medium and shows a graphical user interface (GUI) representation of the collected data. The software or web applications used in this research were The Things Network, The Things Stack, Ubidots.

4.7.1 The Things Stack

An open-source, enterprise-grade LoRaWAN network server is called Things Stack. With the help of the things stack, users can set up and maintain LoRaWAN networks either on their own

hardware or in the cloud. The Things stack is available in many versions to accommodate the needs of various applications. This can be used for the following deployments:

- **Cloud:** SLA backed network server hosted by Things Industries (TTI).
- **Dedicated cloud:** all the benefits of the cloud of a cluster cloud.
- **AWS launcher:** The AWS marketplace makes deployment easier using a single click on new or existing clusters. This is integrated with the AWS IoT.
- **Enterprise:** installing the network server on the hardware using professional support from The Things Industries.
- **Community Edition:** This is run by The Things Network, and it is free to use the network server that provides the world's largest community based on the LoRaWAN network.

Things Industries are responsible for creating and maintaining The Things Stack. Things Industries offer a variety of things stack implementations, such as managed cloud services. For version 3 of a network server, Things Stack is a LoRaWAN Network Server stack. To start the Things Stack, users just require creating an account on The Things Network, so they can use the free version of The Things Stack Community Edition. It is free to use up to 25 devices for deployment, but for commercial use, it has some cost for businesses as a professional plan. Once users have access to The Things Stack, they can start adding their devices. If users are creating devices in urban areas, then there will be gateways available as public The Things Community.

Things Stack has a web interface that supports all the basic features of the Things Stack. However, it can be enhanced using a Command-Line Interface (CLI), where users can install, configure, and login with CLI to make changes to the web interface. The Things Stack has a console as a management application for LoRaWANs. It is a web application that can be used for registering devices, end devices, or gateways, monitoring traffic, or configuring networks. Console of The Things Stack as shown in Figure 4.50.

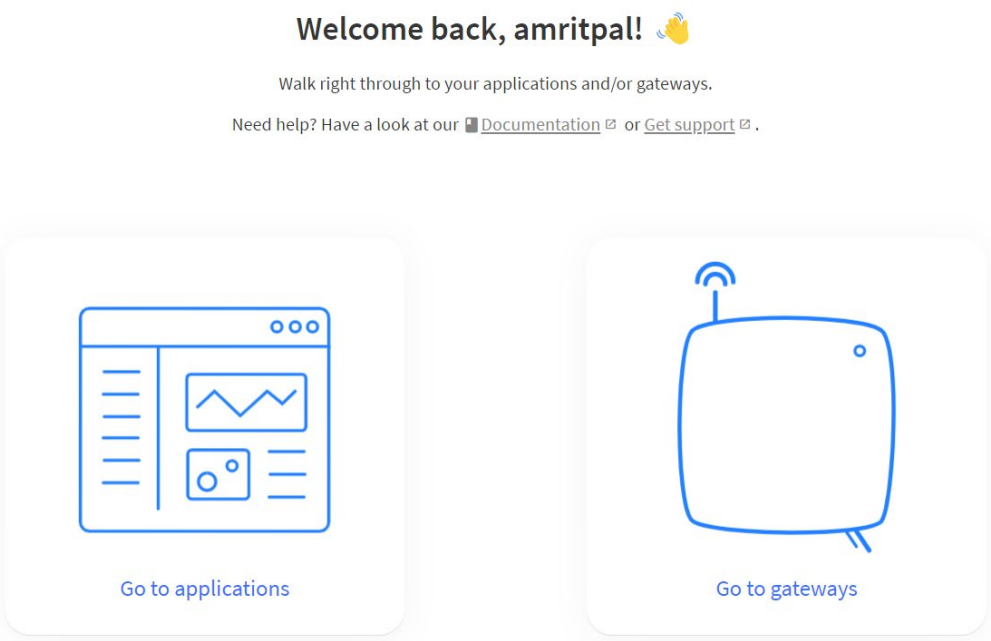


Figure 4.50 Things Stack Console.

If users are using The Things Stack Cloud, then the console is available using the following URL using the client id and region.

<tenant_id>. <eu1/eu2/au1/nam1/>. cloud.thethings.industries

If users are using The Things Stack Community Edition, then the console is available using URL as below:

Cloud.thethings.Network.

4.7.1.1 Integration Workflow

Connecting data from The Things Stack to the webhook is known as HTTP cloud-to-cloud integration. These integrations require intermediate steps to ensure the capabilities between a sender and receiving data cloud formats. Things Stack integration with ubidots requires a parsing step for the conversion of the native JSON format into an ubidots-compatible format.

4.7.1.2 Payload Formatter

The payload formatter in The Things Stack is known as a decoder. It is an in-built feature in The Things Industries V3 Stack, allowing the hexadecimal encoded data coming from devices to be parsed and converted to readable numerical values or any other kind of data conversion on uplinks and downlinks. Payload formatters allow users to process data going to and from the end devices. The IoT industry has three types of payload formatters: JavaScript, CayenneLPP, and Repository. Payload Formatter in V3 are like The Things Network community edition as shown in Figure 4.51 Payload matter.-

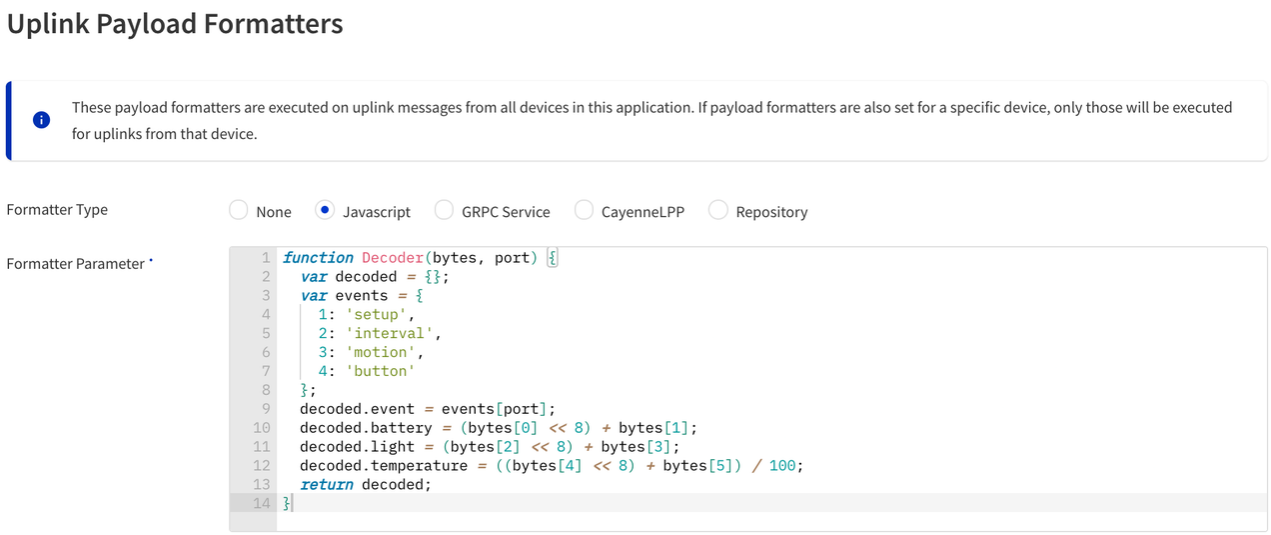


Figure 4.51 Payload matter.

4.7.2 ubidots

Ubidots provide a secure and easy method for building IoT for students, education providers, and researchers. It is used for sending data from any internet-based device to the cloud, triggering actions and alerts based on that data, and visualizing it.

4.7.2.1 Setup Ubidots

Users must account for ubidots, login into account, and find devices on the upper part of the ubidots dashboard. It is in the drop-down list and select plugins as Figure 4.52.

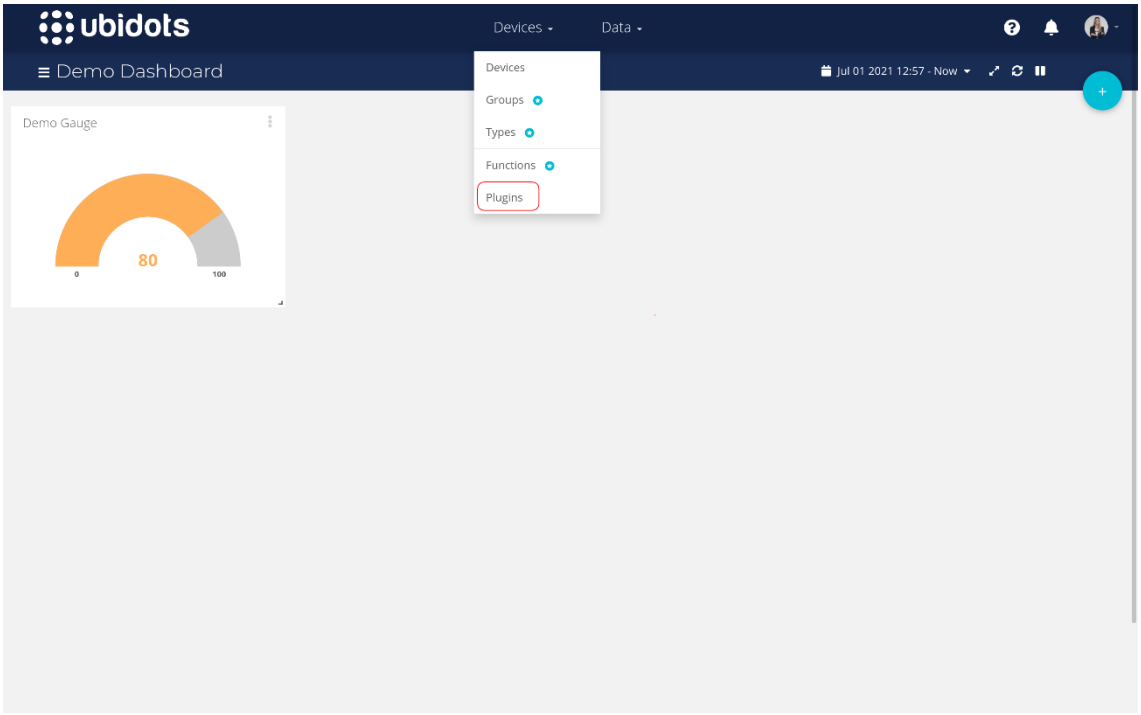


Figure 4.52 Demo Dahboard to select plugins.

Click on the plus button or create a data plugin to create a new plugin and select The Things Stack plugin shown in Figure 4.53.

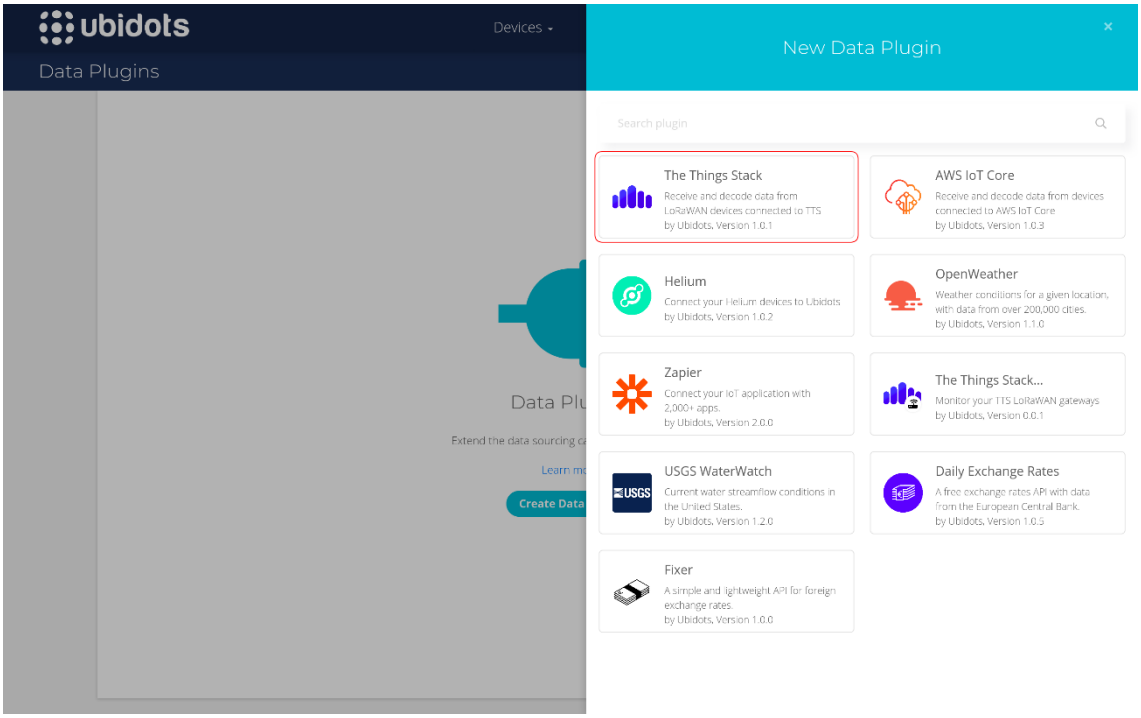


Figure 4.53 Things Stack Plugin

It will show users about inputs, usage, etc., and users will need to provide names for a ubidots type, after which a new device type will be created. For better performance users has to select ubidots token and always used default token as shown in Figure 4.54.

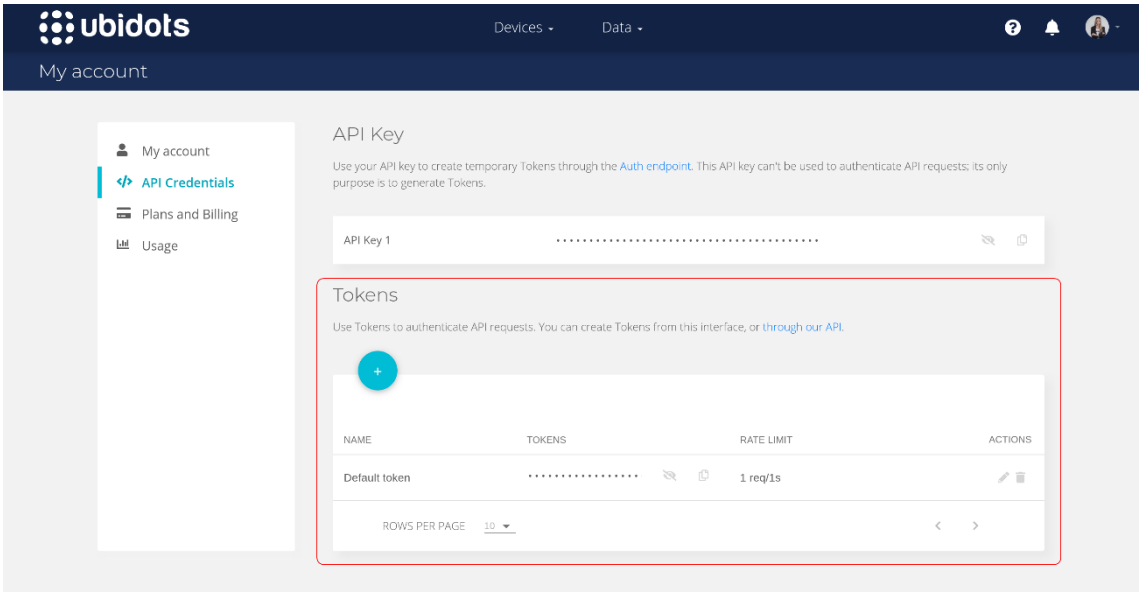


Figure 4.54 Token Selection

After finishing, it shows the running status. To complete the integration with The Thias shown in Figure 4.55.

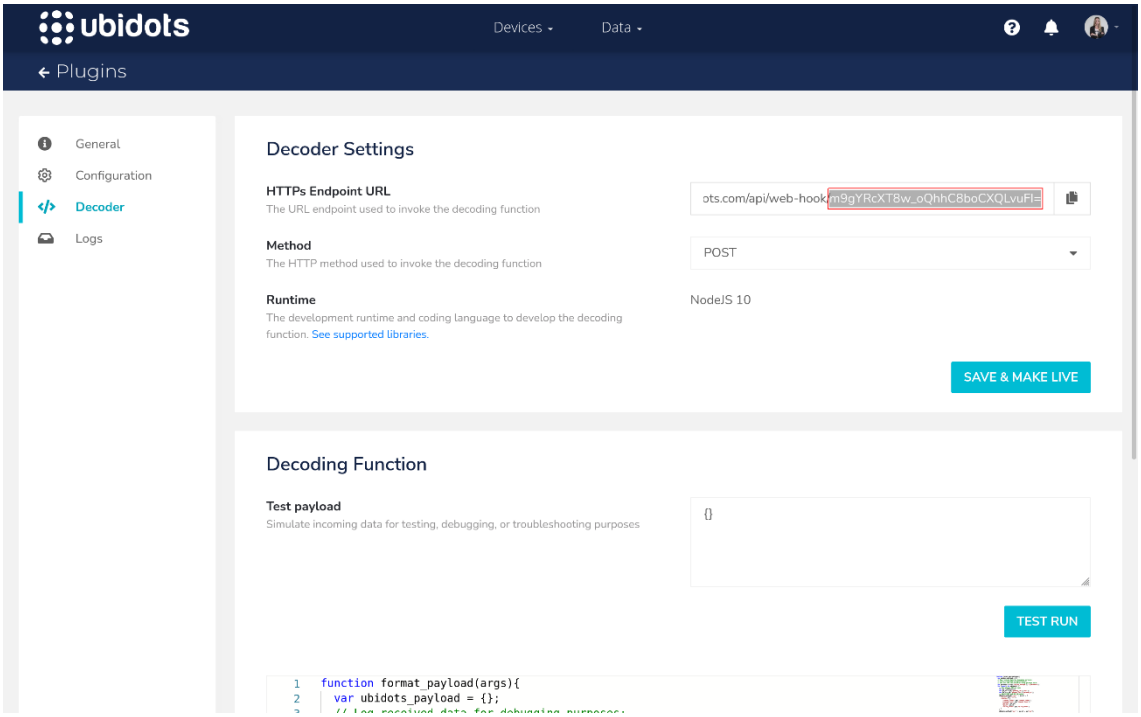


Figure 4.55 Endpoint URL

4.7.2.2 Payload Formatter

There are two ways to decode the payload that is sent from the end device in uplink messages and display findings on the ubidots dashboard, shown in Figure 4.56, it shows two options to build a Things Stack uplink payload formatter-

- To alter the example decoder that is preloaded on Ubidots, which decodes the raw payload that was received from The Things Stack and encoded in Base64 in the uplink message from the payload field.
- The decoded values are found in the uplink message decoded payload field if users use an uplink payload formatter on The Things Stack to decode the payload.

This is a portion of the sample payload decoder in Ubidots. Ubidots verify the decoded values by checking the decoded payload field, and they require uncommenting on the line in the format payload function.

```
var decoded_payload = args['uplink_message']['decoded_payload'];
let bytes =  Buffer.from(args['uplink_message']['frm_payload'], 'base64');
var decoded_payload = decodeUplink(bytes)['data'].
```

Figure 4.56 Payload Decoder

Users can also find the sample preloaded payload decoder in the new plugin menu on the left and scroll down to the decoding function window. Then click on save and make live as shown in
Figure 4.57.

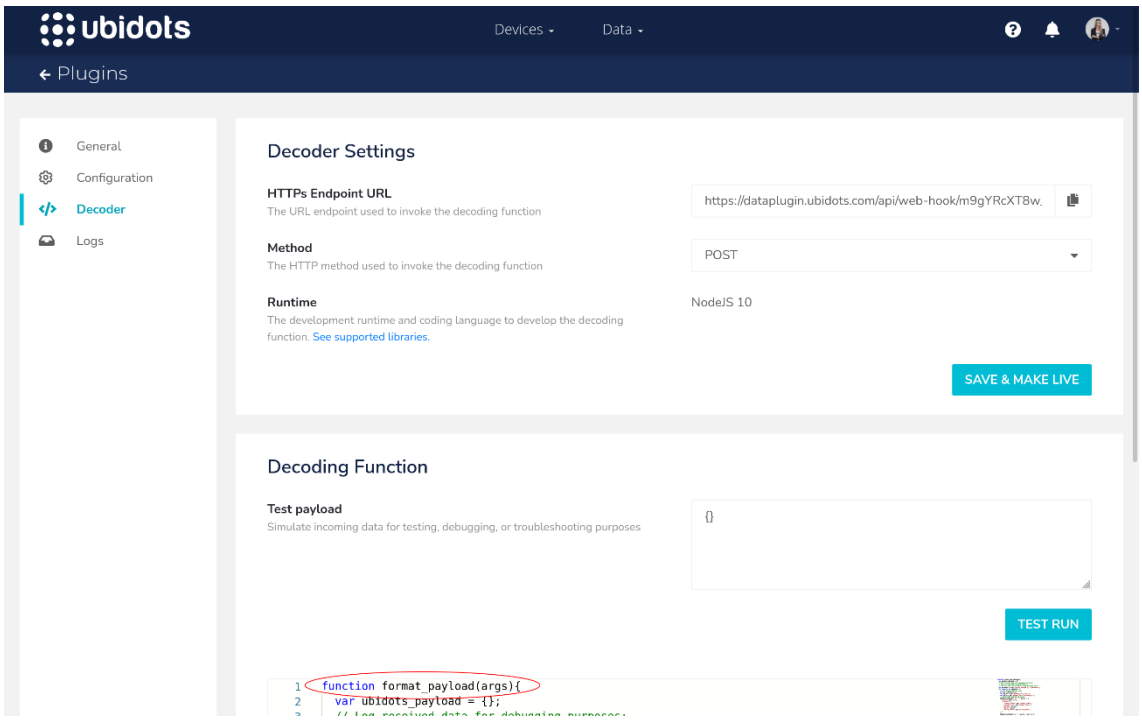


Figure 4.57 Decoding Function

4.7.2.3 Webhook Integration

When ubidots have been set up for The Things Stack, the next step is to create a webhook integration on The Things Stack. On the Things Stack, go to the integration and click on the Webhooks. Webhooks select the Ubidots webhook template, which requires a Webhook ID. Then paste Plugin ID and Ubidots token values from the ubidots and add those to the Webhook integration page on The Things Stack as shown in Figure 4.58.

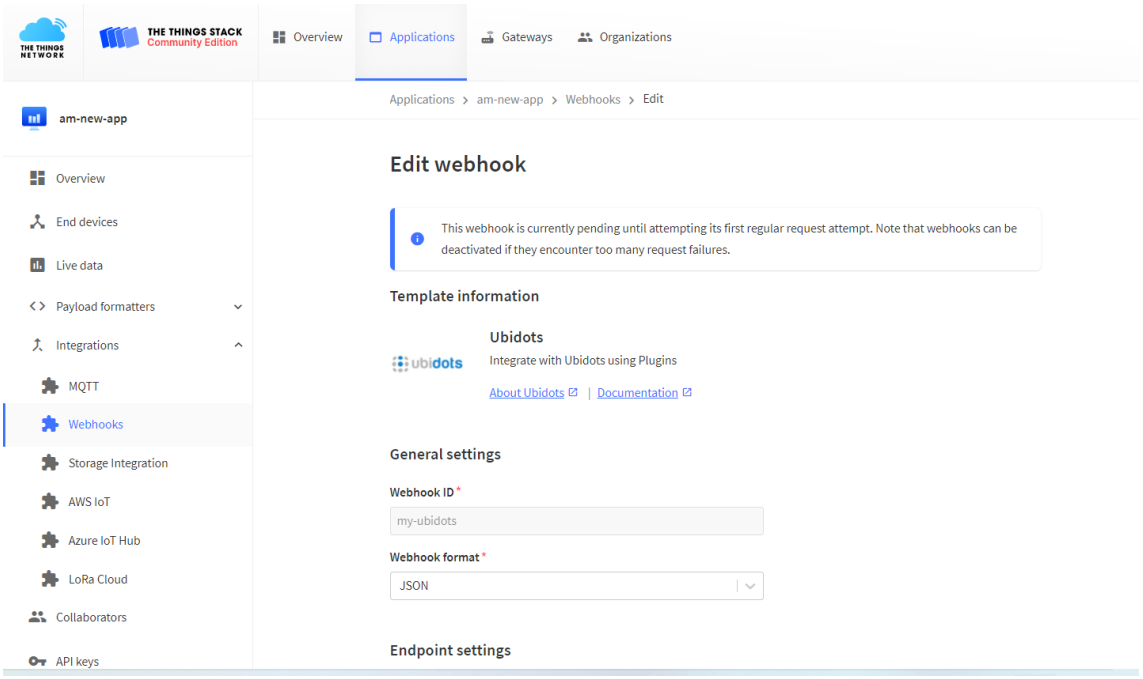


Figure 4.58 Webhook Integration

4.7.2.4 Monitoring Data

Once the integration is completed, navigate to the Devices menu shown in Figure 4.59; it will show the devices connected and send an uplink message.

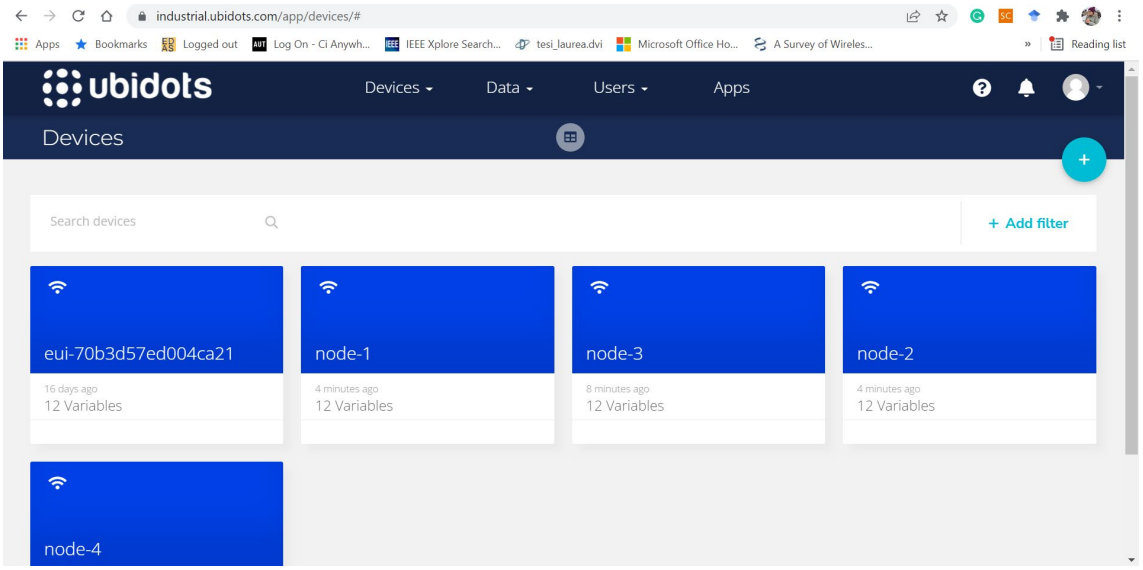


Figure 4.59 Devices on Ubidots

Select the device to look at the dashboard, it has some variables added automatically loaded from uplink metadata as shown in Figure 4.60.

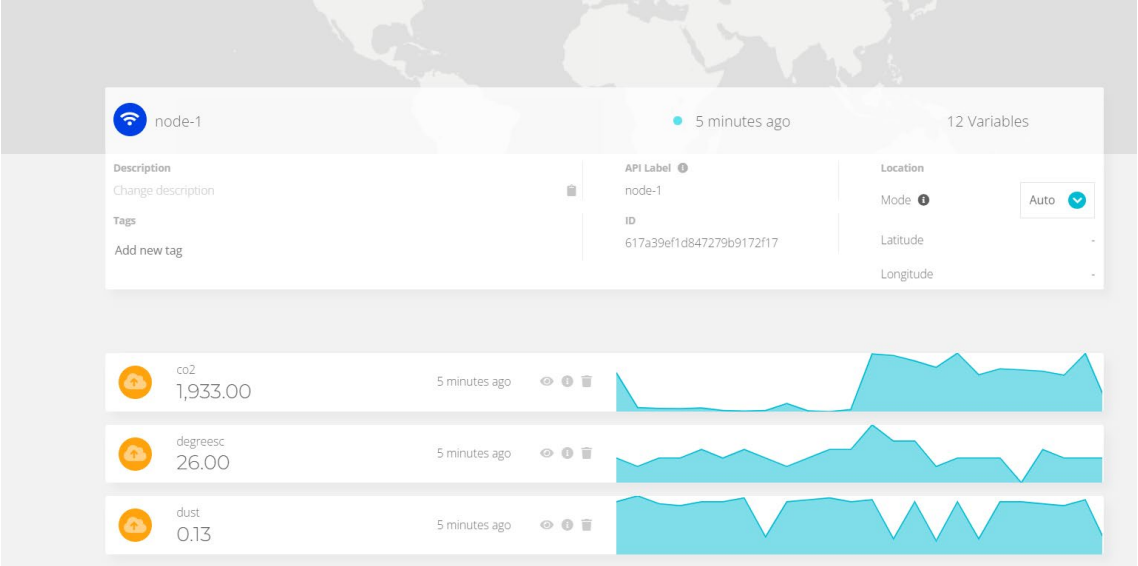


Figure 4.60 Different Variables on Device.

4.7.3 Impact of Software

Write something here

- **Data Processing and Analysis:** The software used for data processing and analysis can greatly affect the accuracy and insights gained from the collected data. Robust and well-optimized data processing algorithms can help in identifying patterns, trends, and anomalies effectively.
- **Data Visualization:** Visualization tools and software can enhance the understanding of the data by presenting it in a clear and intuitive manner. Proper visualization can aid researchers in identifying patterns and making informed decisions.
- **Data Management:** Efficient data management software ensures data integrity, storage, and retrieval. Proper data management is essential for maintaining data quality and preventing data loss or corruption.
- **Communication Protocols:** The software used for communication between the sensors or devices and the data collection server can impact data transmission

efficiency. Reliable communication protocols help ensure that data is transmitted accurately and without loss.

4.8 Limitations

Monitoring air pollution using LoRa (Long Range) networks at the street level can be a valuable approach to gather real-time data and understand local air quality conditions. However, like any technology, there are some limitations and challenges associated with using LoRa networks for street-level air pollution monitoring:

1. **Limited coverage area:** LoRa networks have a limited range, typically a few kilometers in urban environments. This means that to achieve comprehensive street-level monitoring, a significant number of LoRa gateways and nodes may be required, leading to increased infrastructure costs.

2. **Data accuracy:** The accuracy of air pollution measurements can be affected by various factors, such as sensor quality, calibration, and environmental conditions. Low-cost sensors used in LoRa-based devices may not provide the same level of accuracy as reference-grade instruments used in official air quality monitoring stations.

3. **Spatial resolution:** LoRa networks may not offer the same spatial resolution as a dense network of stationary air quality monitoring stations. This limitation can result in uneven coverage and potential gaps in data, especially in areas with limited LoRa network infrastructure.

4. **Interference and signal attenuation:** In urban environments with high buildings and structures, LoRa signals may face interference and attenuation, leading to reduced data transmission reliability.

5. **Power constraints:** LoRa devices are often designed with low power consumption in mind to extend battery life. This limitation may impact the frequency of data transmissions and the real-time nature of the monitoring system.

6. **Sensor calibration and maintenance:** Proper calibration and maintenance of air quality sensors are critical to obtaining accurate and reliable data. However, sensor calibration can be challenging, and periodic maintenance is required to ensure data

quality.

7. **Sensor drift and sensitivity:** Some air quality sensors used in LoRa devices may be sensitive to environmental changes and suffer from sensor drift over time, leading to potential inaccuracies in measurements.
8. **Data validation and quality control:** With many distributed sensors, data validation and quality control become more complex. Ensuring the accuracy and reliability of the collected data is essential for meaningful interpretation.
9. **Data integration and analysis:** Managing and integrating data from multiple LoRa nodes spread across a city can be challenging. Proper data analysis and interpretation require sophisticated algorithms and computing resources.
10. **Regulatory compliance:** For air pollution data to be used for regulatory purposes, it must meet certain standards and be comparable to data from official monitoring stations. Ensuring the compliance of LoRa-based air quality monitoring systems with regulatory requirements can be a significant challenge.

Despite these limitations, LoRa networks can still provide valuable insights into local air quality conditions, especially in areas with limited access to official air quality monitoring stations. By acknowledging these limitations and using complementary monitoring approaches, such as combining LoRa networks with traditional monitoring stations, it becomes possible to create a more comprehensive and accurate air quality monitoring system.

4.8.1 Impact of limitations

- **Range Limitations:** The range of communication technology (e.g., LoRa, Sigfox) can limit data collection in large or geographically dispersed areas. Inadequate coverage may result in data gaps or require additional infrastructure deployment.
- **Data Loss and Interference:** Environmental factors or technical limitations may lead to data loss or interference during transmission, affecting data completeness and accuracy.
- **Sampling Frequency:** Some hardware or communication protocols might limit the frequency at which data can be collected. Lower sampling frequency can impact

the temporal resolution of the data, potentially missing important events.

- **Environmental Conditions:** Environmental conditions, such as extreme temperatures, humidity, or electromagnetic interference, can affect hardware performance and, in turn, impact data quality.
- **Cost Constraints:** Budget limitations may affect the choice of hardware and software, leading to trade-offs between data accuracy, hardware features, and project cost.

4.9 Issues

Network deployment causes issues at different milestones. Owing to network complexities, there are diverse challenges faced by network operations that are not associated with the understanding of technology but its maintenance towards communication access.

4.9.1 Gateway

During network deployment, the first initial testing started with the Sparkfun LoRa channel 1 gateway. The Sparkfun gateway was supported by The Things Network. With an ESP32 module and an RFM95W LoRa Modem, it is a massive 3-network capable device. The 915 MHz frequency band was handled by RFM95W, and Bluetooth and Wi-Fi were handled by ESP32. The goal was to use it to turn LoRa (long-range) radio communications into data packets that could be accessed online.

The issues with gateway were found during the research as below-

- The Things Stack does not support the Sparkfun Gateway, as it is not listed on the gateway list in The Things Industries. This causes issues when the second testing experiment is performed, and packets are not forwarded to the web browser. When this gateway is created on Things Stack, it does not show any visibility on the web. Later, it was found that this is not actually the LoRa Gateway, but it acts as a LoRa gateway for communication.
- Sparkfun acts as a long-range communication device; however, it does not. During the

research, it was tested in different experiments; its antenna range was not vast; it was just less than 50m. When the gateway is placed more than 50m away, it does not communicate with the sensor nodes and does not transmit anything to the web cloud.

4.9.2 Upgraded with The Things Stack

When this research experiment started, it was working with The Things Network (v2 Console) by The Things Industries. The conference by The Things Industries in September 2021 announced that The Things Network V2 would be permanently shut down in December, and this will be extended using V3 The Things Stack (V3). The Things Network will not be able to access its devices after December 1, 2021, but users can migrate their devices to The Things Stack (V3).

It is again a time-consuming process to setup the migration of devices from the V2 to V3 console. Things Industries created three new clusters, and users can only use the gateway for further communication processes that are available on the Things Industries webpage. For the migration of the devices, it requires to follow the steps below:

- Users can use the same The Things ID to access things network.org and V2 console.
- Users need to select clusters according to their location and region, as created by Things Industries. Users can use the private network as packet brokers for LoRaWAN peering. Thus, users can contribute to an IoT network.
- The packet broker in the network routes the traffic from gateways connected to The Things Network V2 to V3, where the coverage of V2 is available in V3.
- Gateway migration does not occur automatically. Users must create a gateway on their own.

All of these processes take time and slow down research performance. The Things Stack is completely different from The Things Network. Therefore, it takes more time to understand.

4.10 Experimental Setup

To achieve the low-cost and long-range communication goals of a network system, there is a need to develop new equipment using IoT with a combination of a microcontroller and sensor modules. The microcontroller required a program for data collection. It is also important to build a server that receives data at the back end. A simulation network experiment will be tested first, and then more placement of hardware prototypes can be set at the street level to build an air pollution monitoring network to collect data and prove the efficiency of the network design.

The network design is separated into four layers for data collection, in which the sensing layer has the main task of collecting data, and the network layer is responsible for data transmission to the server. In the network, these two layers communicate directly with the physical and hardware components of the project. The development of the network focuses on the two layers and integrates them with the IoT development board, as shown in Figure 4.61.

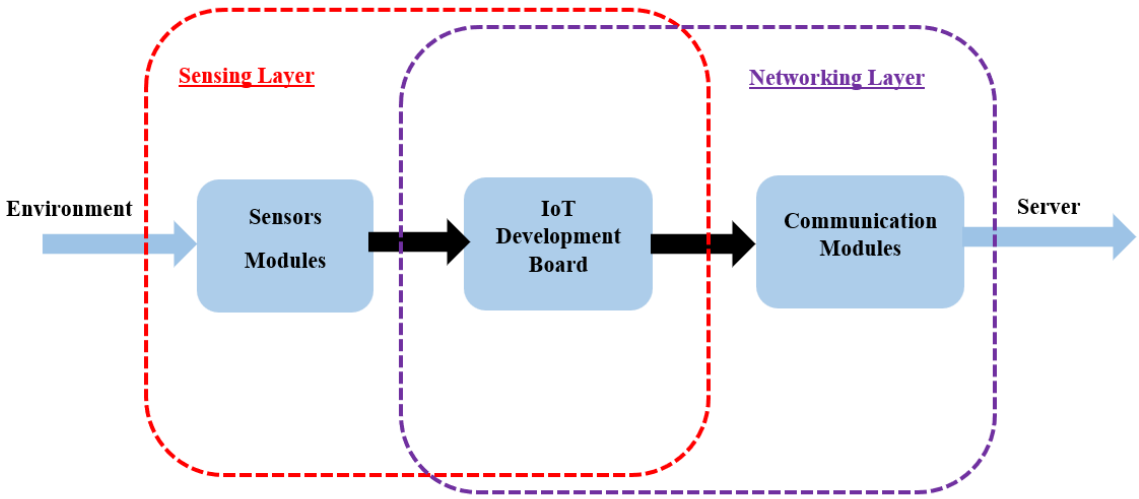


Figure 4.61 Network Block Diagram

The software part of the network design focuses on the following three main aspects, as shown in Figure 4.62.

- Hardware and software tools operate on the sensing and networking layers and oversee the gathering and transmitting of data.

- Server software, which is a part of the middleware layer, oversees the receiving and processing data on the server.
- The web server, which is a part of the application layer, is responsible for showing user data.

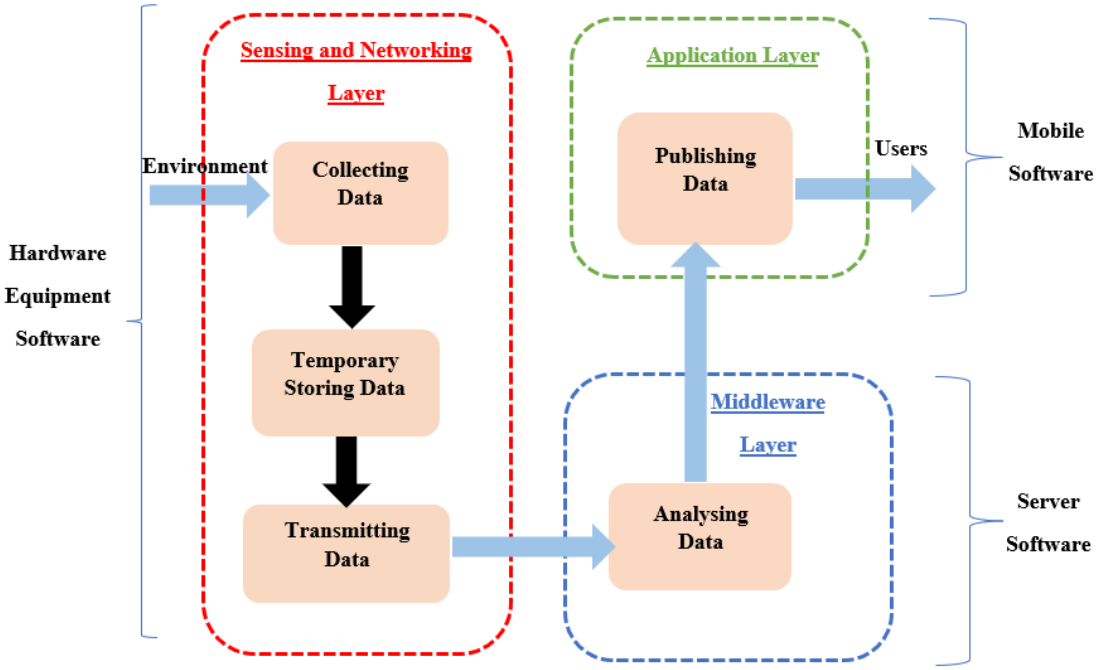


Figure 4.62 Network design and Working

4.10.1 Front-end Layer

The front-end layer in the network covers all the deployment of sensors to fulfil the requirements of air pollution monitoring. The network design supports a specific type of sensor to monitor pollution, which transmits data over the LoRa radio module. However, it uses the LoRaWAN protocol in the network to forward the data.

The IoT Dev Kit board's built-in temperature, humidity, dust, CO_2, and PM sensors were used to test the network's design. Sensors, a power supply unit (USB connection), an I2C communication bus, and a LoRa wireless access medium with two interfaces (868 MHz and 915 MHz) were all included on the board. The network design collects and combines data from temperature, humidity, dust, CO_2, and PM sensors to create a pollution monitoring system at the street level.

4.10.2 Middleware Layer

The deployment of gateways to meet the needs of a given application is performed by the middleware layer of the network. In this study, the LoRa network server was managed by a wireless communication network platform called a gateway. All devices linked to an LoRa end node were managed by the network server. The distribution of all channels, assignment of device addresses, and adaptability of device data rates are handled by the network server. MIC verification verifies the uplink frame integrity before starting the downlink transmission.

The application server controls hardware and data related to the program. This enables the integration of the API for creating applications centred around data recovery from sensors linked to LoRa end nodes. The sensors in the application direction oversee the reading data from the radio frames and store them. Additionally, it oversees downlink conversion to radio frames. The network data server integrates and stores sensor-generated data that can be used for future applications. The gateway may include the desired apps incorporated into it.

4.10.3 Back-end layer

The back-end layer in the network design covers the application of a client that connects to the application server. It is used for data consumption from the connected sensors and for declaring new devices and other possibilities. Once the device is declared and activated on the server, a new application is ready to be developed to receive sensor data.

The middleware layer contains servers for the infrastructure. To provide security in the network design, each IoT device has its own AppSKey, NwkSKey, and Device EUI, which are shared with the server during each session. The LoRa gateway sends the packet to the network server for uplink messages, where if the signature is allowed, it is verified using NwkSKey. The packet is then allowed, but the encrypted payload remains; therefore, the network server sends a notification to the application server so that AppSkey can be used to decrypt the packet.

The network server that has NwkSKey, which is utilized for encryption and signing, sends downlink messages to the IoT sensor nodes. Using NwkSKey, an IoT sensor can decrypt and verify the signature of the received downlink message.

4.10.4 Gateway Position

For each experiment, the gateway was placed at fixed location as below:

Latitude: -36.854197395105665,
Longitude: 174.76727041820428

4.10.4.1 Gateway Location at University

The location of the gateway is shown in Figure 4.63 and Figure 4.64, where the gateway is placed in the university's building.

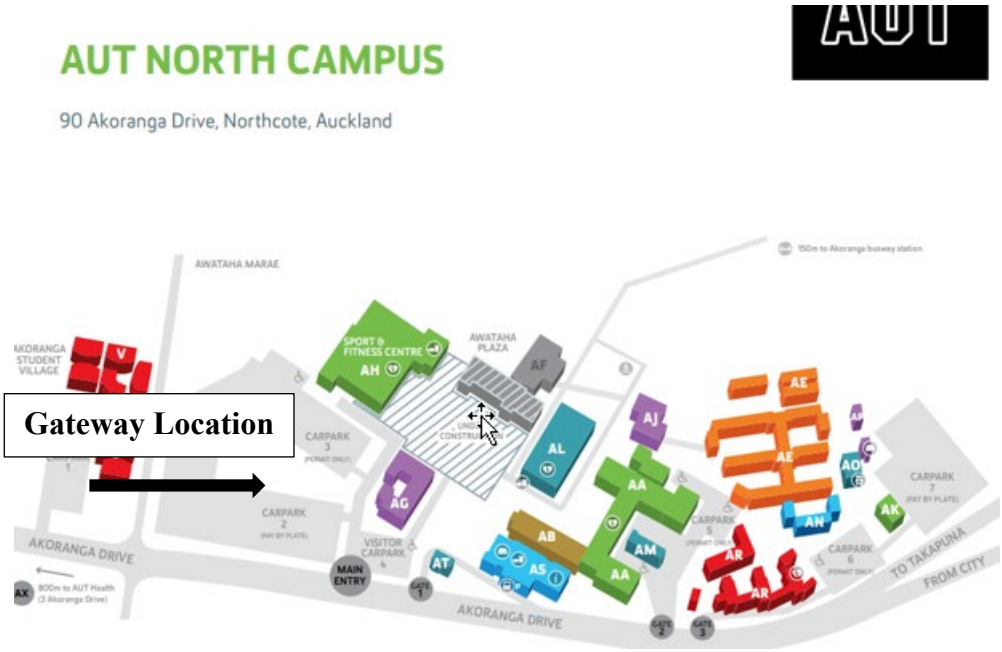


Figure 4.63 Gateway Position

Figure 4.64 Gateway Position Map

4.11 Summary and Contribution

This Chapter focuses on the different communication technologies used in wireless communication. This Chapter covers three main technologies, LoRaWAN, Sigfox, and NB-IoT, and presents a comparative analysis. This research is based on LoRa technology, as it explains the features better than other communication technologies in terms of long range and low power consumption. Another section in this chapter covers The Things Network, its network server, and the features for use in different applications. It also explains the architecture of the IoT network and how it works with different technologies and applications. This chapter provides information about the network topologies and the topology used in network deployment. How effective is this for improving network performance? Different hardware and software were used in the designed network to monitor the pollution levels.

Flowcharts for the operation of sensor units are also explained in a separate section, which explains how the sensors transmit and communicate with the things stack. The dust and CO_2 sensors read analogue values, but the CO_2 sensor took 3 min to warm up and read the sensor values. The DHT22 and SPS30 sensors sent immediate values once they were powered on.

Both sensors take a few seconds to transmit data. During sensor deployment, it was noticed that the dust sensor consumed more power and did not provide accurate values for dust levels. It was replaced with an SPS30 sensor after pilot testing.

There are some of the reasons why I prefer LoRa technology as explained below:

- **Long Range:** LoRa's long-range communication capabilities make it ideal for research projects that require data transmission over extended distances. It enables researchers to collect data from remote or inaccessible areas, such as rural regions, wildlife habitats, and environmental monitoring sites.
- **Low Power Consumption:** LoRa's low-power operation allows researchers to deploy battery-powered sensors and devices for long-term monitoring. This feature is especially valuable in projects where continuous data collection over extended periods is necessary, such as environmental monitoring, wildlife tracking, or agricultural research.
- **Flexibility and Customization**: LoRa is based on an open-source LoRaWAN protocol, offering researchers the flexibility to customize their network infrastructure and protocols. This adaptability allows them to tailor the network setup to their specific research needs and experiment with different configurations.
- **Scalability:** LoRa networks can scale to accommodate a large number of end-devices, making it suitable for research projects that involve deploying numerous sensors or devices over a wide area. Researchers can easily expand their LoRa network as the project requirements evolve.
- **Cost-Effectiveness:** LoRa technology operates in unlicensed frequency bands, eliminating the need for expensive spectrum licenses. This reduces the overall cost of setting up and maintaining LoRa-based research networks compared to other proprietary or licensed technologies.
- **Low Data Rates:** Some research applications may only require infrequent data transmission or small data payloads. LoRa's ability to support low data rates and intermittent communication aligns well with such use cases, where energy-efficient data exchange is crucial.

- **Community Support:** LoRa has a thriving community of developers, researchers, and enthusiasts, sharing knowledge and collaborating on projects. The availability of open-source tools and resources can significantly aid researchers in their work.
- **Environmental Monitoring:** LoRa is well-suited for environmental monitoring projects, such as air quality monitoring, water quality monitoring, and weather data collection. Its long range and low power consumption allow researchers to deploy sensors in diverse environmental conditions.
- **Research in Smart Cities:** LoRa technology plays a vital role in smart city research, enabling applications like smart parking, waste management, and public infrastructure monitoring. Its ability to cover extensive urban areas efficiently makes it a preferred choice for smart city projects.

In conclusion, this research topic used the LoRa technology due to its long-range capabilities, low power consumption, customization options, scalability, and cost-effectiveness. Its features make it a valuable tool for data collection and research in various domains, including environmental science, wildlife tracking, agriculture, smart cities, and more.

This research focused on monitoring PM and CO_2 levels in the air. After replacing the dust sensor with the SPS30 sensor, it monitored the PM_1, $PM_{2.5}$, PM_{10}, and PM_4 levels. PM levels were validated using a flowmeter,. This was a good decision to replace the dust sensor with an SPS30 sensor.

Chapter 5
Results and Discussion

The system components and the methodology used to evaluate the network performance and pollution monitoring system are then described. This section covers the results and discussion of the different experiments conducted during the research. Test experiments were performed to monitor the pollution levels in the air and monitor the performance of the LoRa network.

5.1 Measurement Scenario

In an outdoor pollution monitoring setup, the four end nodes were positioned in the balcony area of the university at a 20m distance and the gateway at 50m space. The balcony area was an open area where the sensors could directly contact the air and record better concentration levels. During all measurements at a specified time, the location of the base station was fixed in a room. The location for monitoring pollutant levels because radio waves must propagate through the core of the building to reach the receiver. Sensor nodes are implemented, as shown in Figure 5.1.

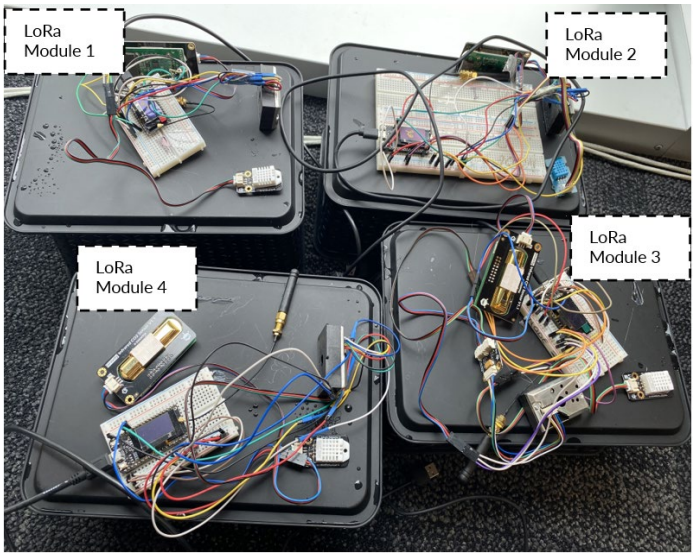


Figure 5.1 Sensing Modules

The microcontroller in the network is considered to be the main sensing module for converting analogue to digital values. The collected data were sent to the network server using the LoRa gateway (LPS8). The data processing and display software presented a graphical representation of the data. The sensor connectivity of the sensing module is shown in Figure 5.2. This is a difficult process for the sensor placement option during network deployment.

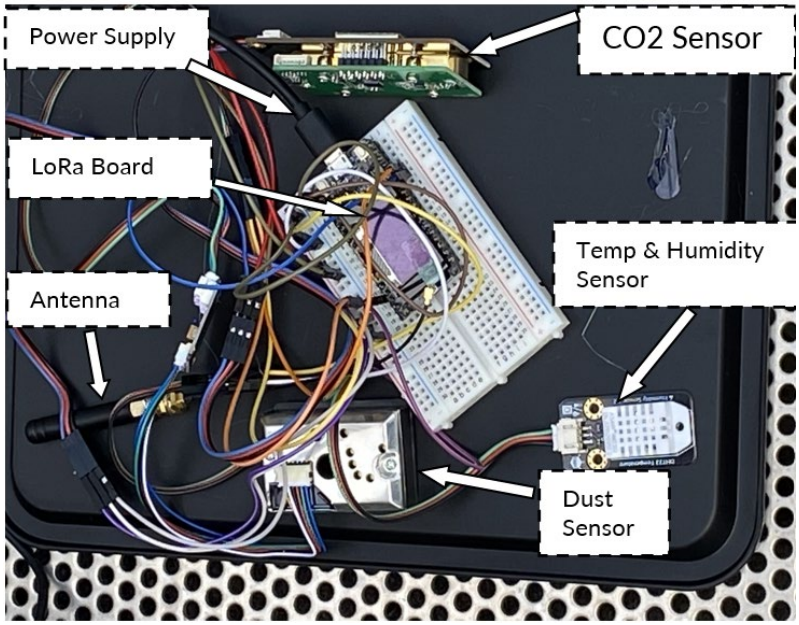


Figure 5.2 LoRa Modules with Sensors

Before placing sensors in the desired location, work is required on factors such as weather conditions, indoor or outdoor location, and gateway position. After going through the different selections, the university's balcony area was selected for testing.

Figure 5.3 and Figure 5.4 show sensor placement in the balcony area. To place it overnight, it needs to be heavy to secure it from damaging the sensor modules.

Figure 5.3 Sensor Placement

11

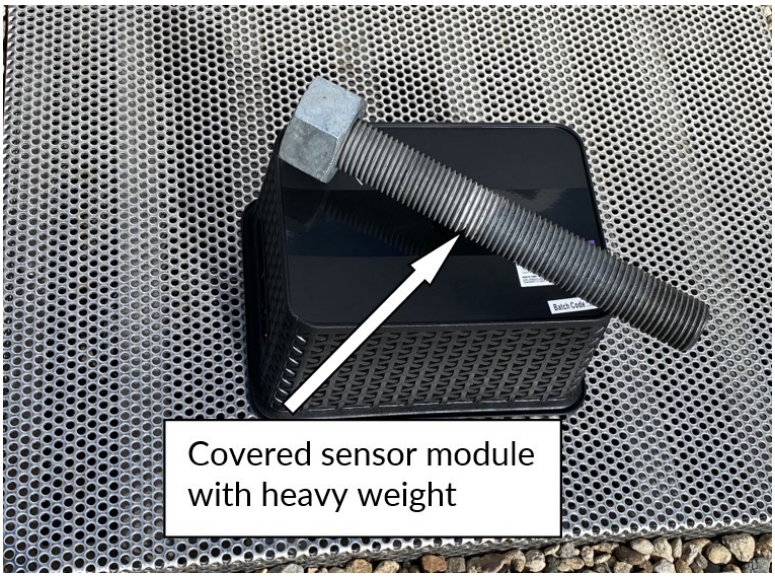


Figure 5.4 Sensor Placement

5.2 Pilot testing stage 1 with Dust sensor

During pollution measurements, each sensor node sent a packet that included temperature, humidity, CO_2, and dust levels in the air to the base station every 10 min. During the experiment, the node sent packets to 915 MHz with a bandwidth of 125 kHz. In New Zealand, the Things Network was implemented at 915-928MHz. The sensor node automatically counts the on-air time for each radio channel and follows duty-cycle restrictions. The transmission power for the end node was set as 14 dBm (25 mW). The purpose of this section is to monitor the pollution levels in the balcony area of the university using temperature, humidity, dust, and CO2 sensors.

5.2.1 Results and Discussion

To test this system, four sensing modules were created, as explained in Section 4, and were placed in the balcony area of the university. Four sensing modules were placed at different corners and in the middle of the balcony to measure the concentration of CO_2, temperature, dust, and humidity in the air. For the results, the LoRa TTGO board, which has built-in Wi-Fi features, was used as the base. The sensor data are transmitted from different sensors connected to the LoRa modules in packets. A payload is added to the network server to identify the address of the sensor module. All sensing modules work as senders and receivers during the data transmission process. There is a distance between the gateway and sensing modules. When the sensor module transmits sensor values, they are transmitted to the nearest gateway id. For testing, the sensing modules were implemented in a university balcony area. However, the discussion here only results from one sensing module. During the experiment, the sensing modules were named Node1 (N1), Node2 (N2), Node3 (N3), and Node4 (N4).

Figure 5.5, to Figure 5.8 shows the results obtained from the sensing modules from Node1 (N1) for 1200 min.

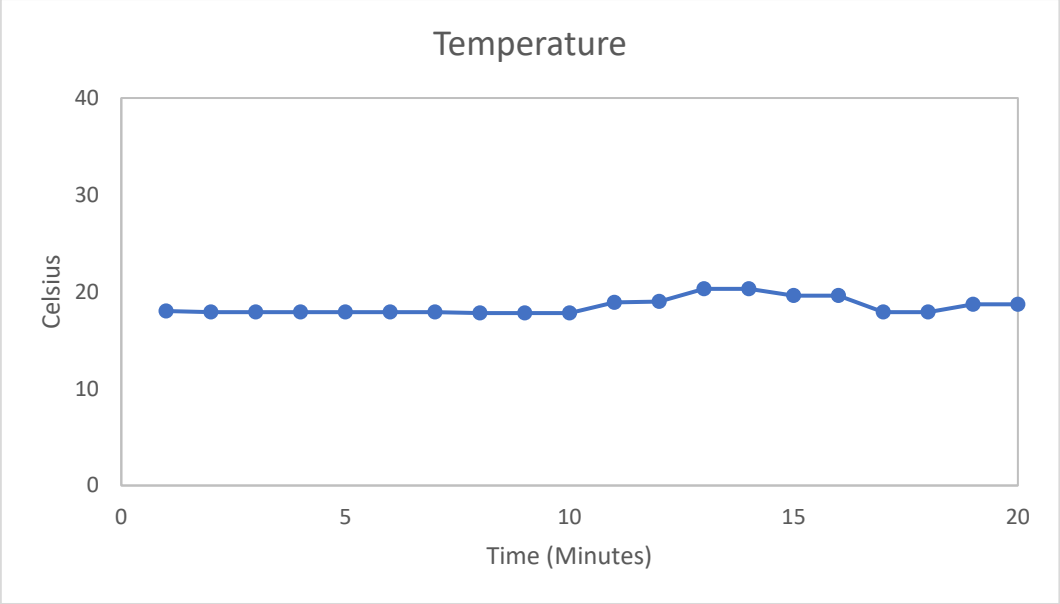


Figure 5.5 Temperature Readings

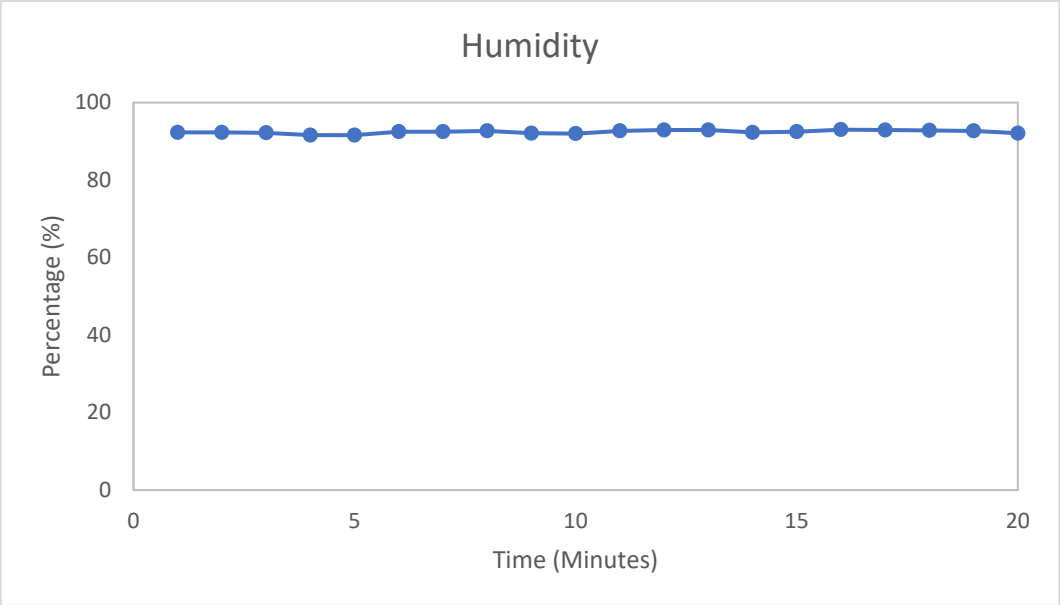


Figure 5.6 Humidity Readings

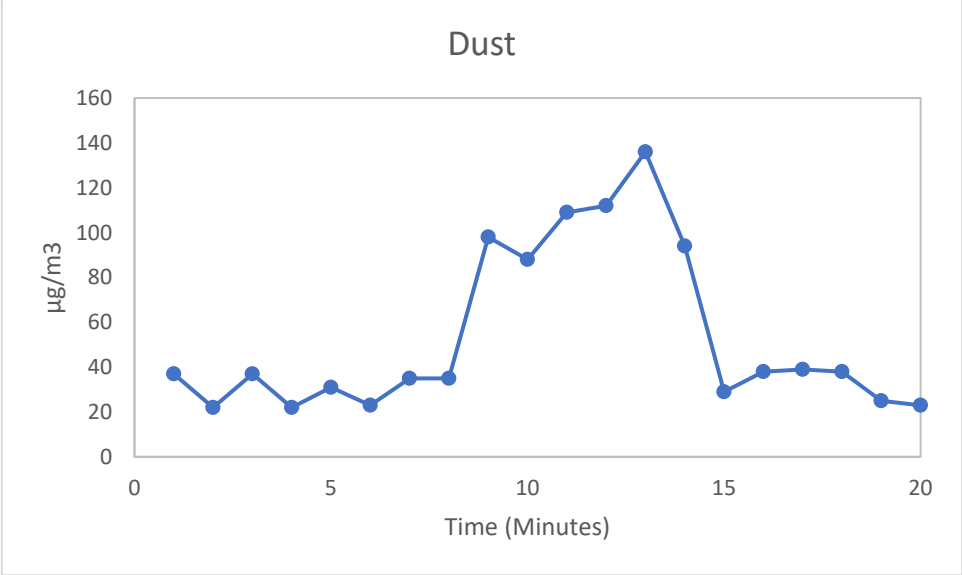


Figure 5.7 Dust Readings

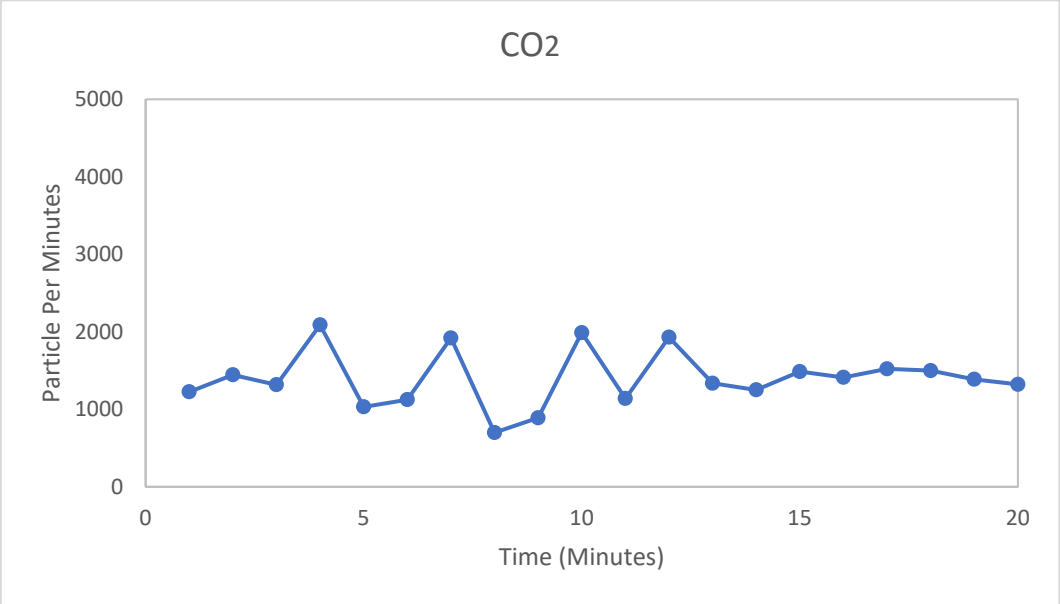


Figure 5.8 CO_2 Readings

5.2.1.1 Received Signal Strength Indicator (RSSI) of Node 1

The designed network works based on wireless communication, where the receiver needs a good signal strength from the modulated carrier. The RSSI is a relative measurement to determine whether the network signal is strong enough for communication with the transmitter. LoRaWAN supports bidirectional communication. The RSSI is considered a good measurement for both gateways and end devices. Figure 5.9 shows the RSSI of node 1, where it measures in dBm, and its value is negative. The RSSI shows good signal strength as the signal is between -30dBm and -50dBm. It is only affected by path loss, antenna gain, or cable/connector loss.

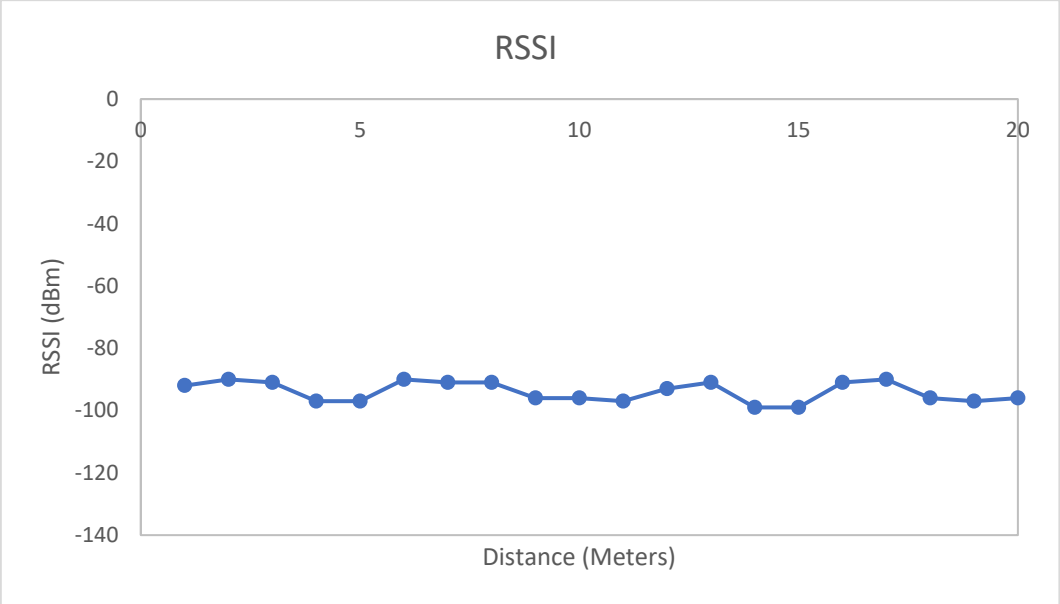


Figure 5.9 RSSI of Node 1

5.2.1.2 Signal-to-Noise Ratio (SNR) of Node 1

In the designed network, the SNR compares the level of signal power to the level of noise power while transmitting data over the network. This is expressed in decibels (dB). The SNR for node 1 is shown in Figure 5.10.

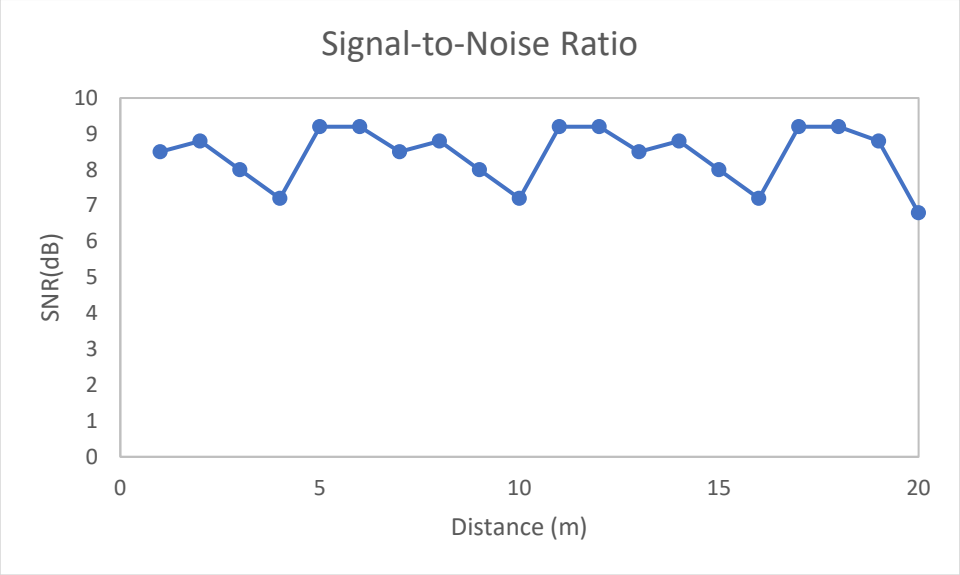


Figure 5.10 SNR of Node 1

5.3 Pilot testing stage 2 with four sensors

In the second test scenario, an outdoor air pollution monitoring system using four LoRa modules was used. These modules were placed outside the same balcony area of the university at 20m distance, and the gateway was placed at a distance of nearly 100m. These LoRa modules have four connected sensors: temperature and humidity, CO_2, dust, and PM sensors. The PM sensor in this test monitors the pollutant levels of PM_1, $PM_{2.5}$, PM_4, and PM_{10} in the air. The LoRa sensor module with four sensors is shown in Figure 5.11, and Figure 5.12 shows the different sensors connected to the LoRa sensor modules.

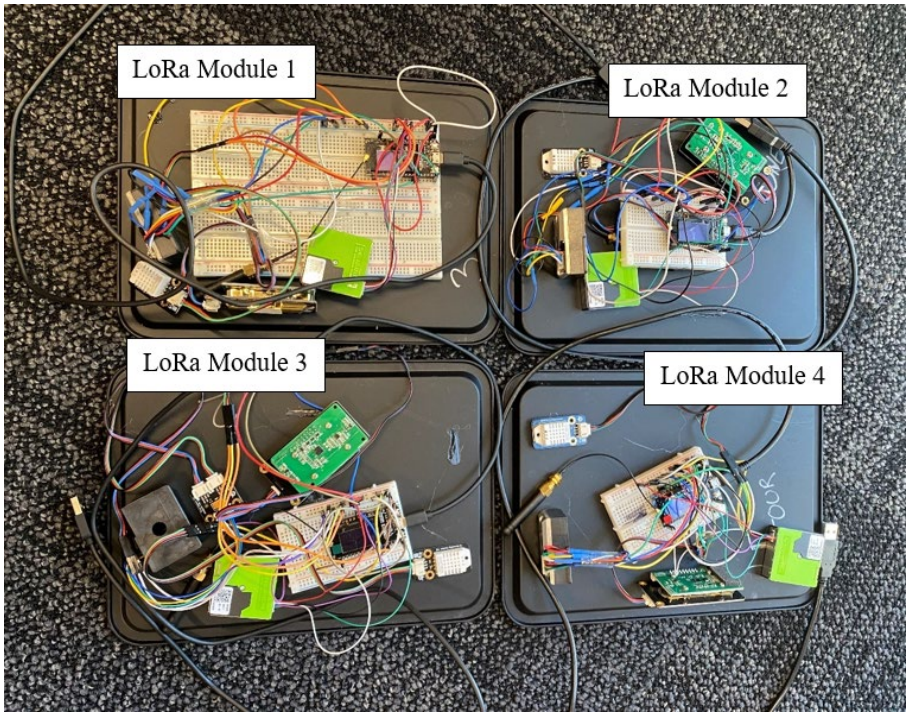


Figure 5.11 LoRa Sensing Module

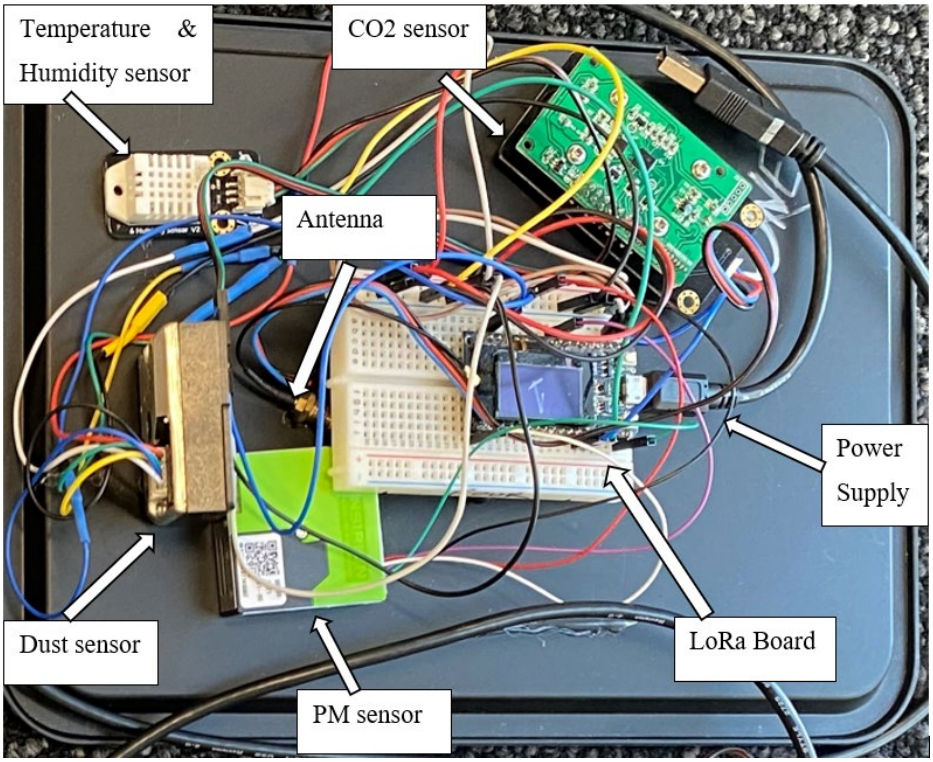


Figure 5.12 LoRa Sensing Modules with Sensors

5.3.1 Results and Discussion

The initial testing for pollution monitoring is explained above using the three sensors. This section describes the concentration levels of temperature, humidity, dust, particulate matter (PM), and CO_2. The purpose of this testing is to check the network performance, how it goes with four sensors, and how much power will be consumed. This test monitored communication range and battery life. Communication between LoRa sensor modules performed well, but it showed a lack of battery power during the transmission of the four pollutant levels to LPS8 and storage of values on The Things Stack. The reason for the higher power consumption is the dust sensor. It consumes more power than other sensors because it has an LED connector, and a flashing LED connector consumes more power. For the results and discussion of the pollutant levels, one LoRa module was considered among the four LoRa modules deployed in the network. Figure 5.13 to
Figure 5.19 shows the pollution levels in air.

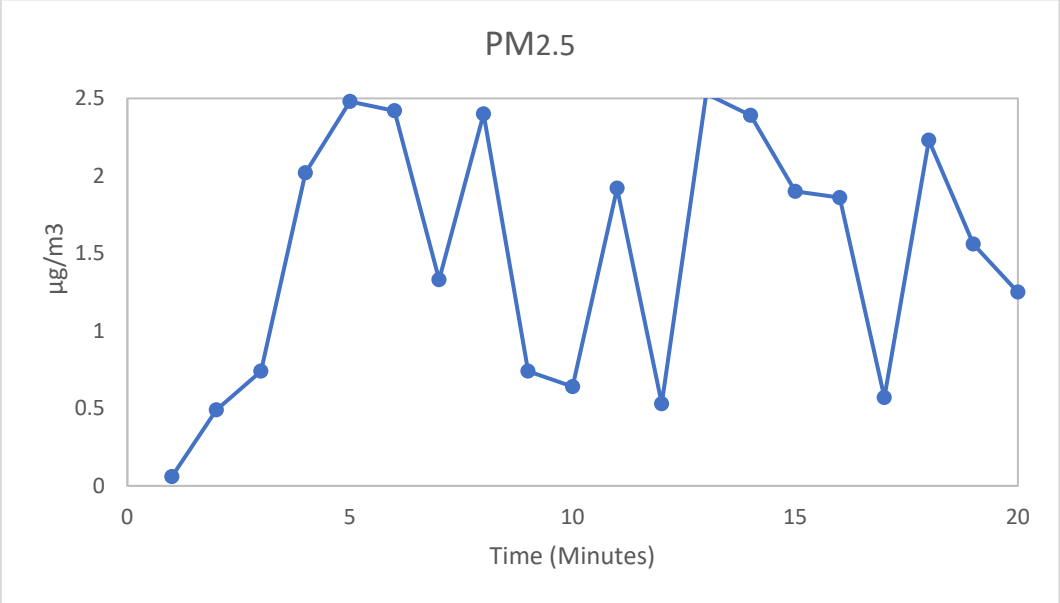


Figure 5.13 $PM_{2.5}$ Concentration Levels

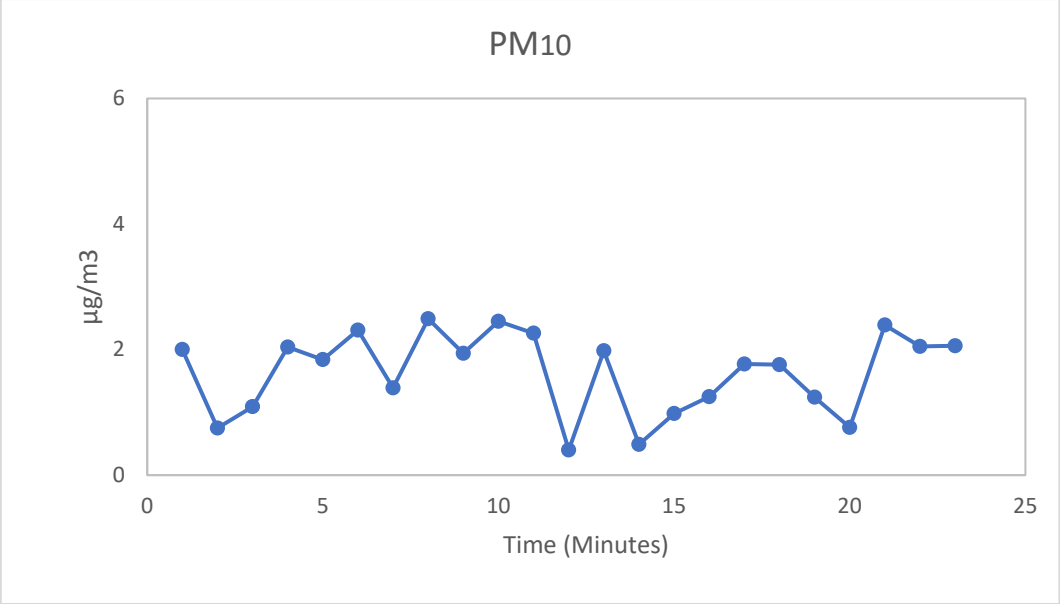


Figure 5.14 PM_{10} Concentration Levels

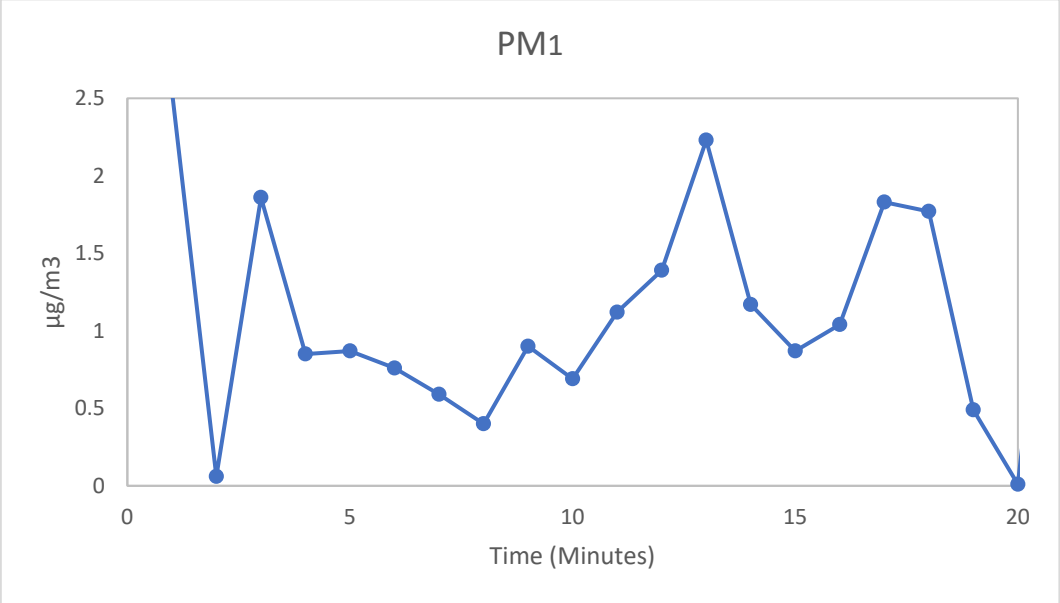


Figure 5.15 PM_1 Concentration Levels

Figure 5.16, PM measurements showed some changes, but the temperature and humidity were stable because the sensor units were placed at a fixed altitude. Regarding the PM_1, $PM_{2.5}$, and PM_{10} measurements, it was observed that the $PM_{2.5}$ value is always greater than other PM levels simultaneously.

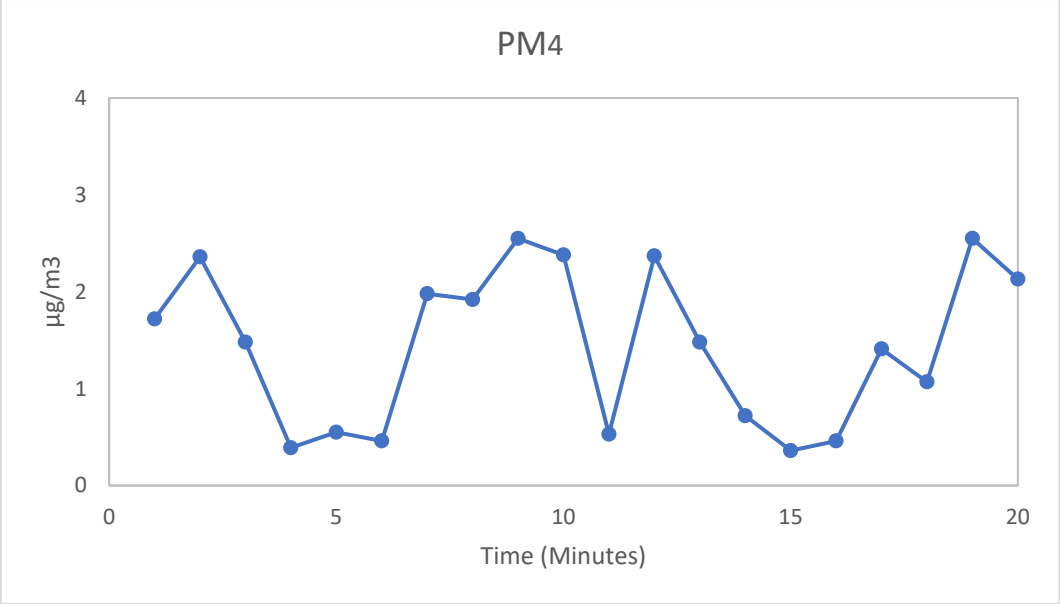


Figure 5.16 PM_4 Concentration Levels

As shown in Figure 5.17, the CO_2 level of the balcony at the Auckland University of Technology, level 6 of the WZ building. The implemented sensor units showed some fluctuations in PM levels, but the CO_2 levels had minor changes. Air flows into the enclosed boxes, and heat and dust are trapped inside the box and show fluctuations. When the data were collected, as per the weather station, it looked sunny, and the temperature was between 12 and 22°C in Auckland, New Zealand shown in Figure 5.18 and Figure 5.19.

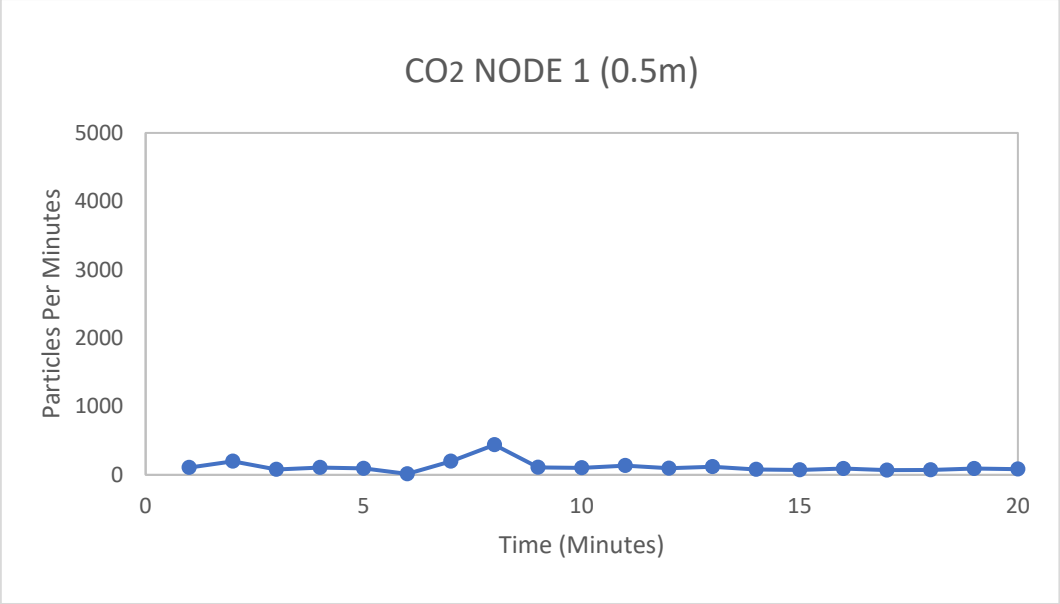


Figure 5.17 CO_2 Concentration Levels

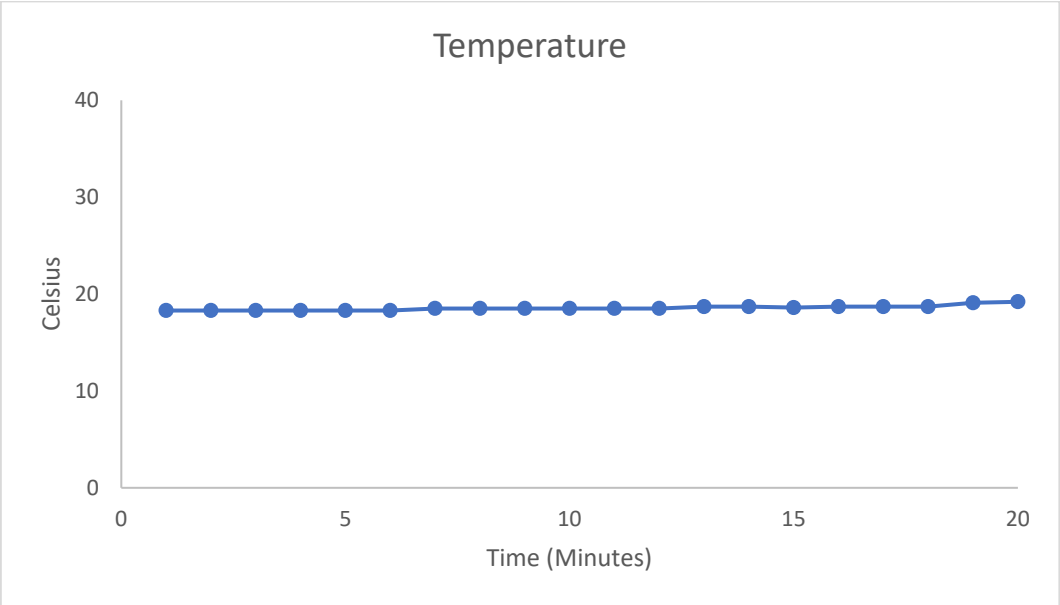


Figure 5.18 Temperature Levels

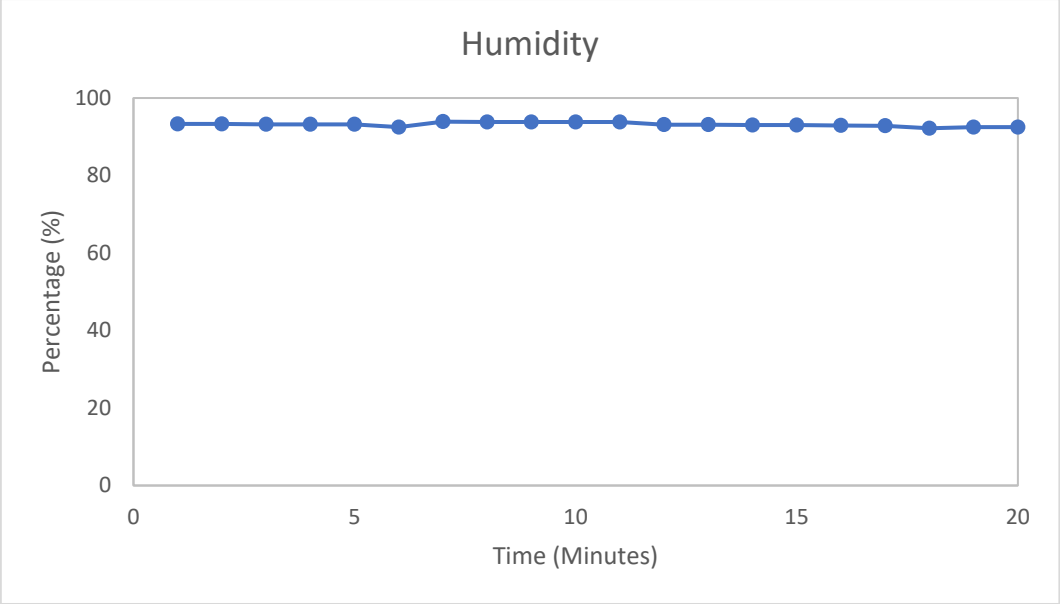


Figure 5.19 Humidity Levels

Figure 5.20 and Figure 5.201 show the RSSI and SNR of sensor module one. The RSSI and SNR were measured to characterize the link between LoRa communication. The received Signal Strength Indication (RSSI) and Signal-to-noise Ratio (SNR) are the two physical level indicators for wireless radio chirps. As shown the RSSI signal was negative dBm. The RSSI in the network is used to measure the signal strength and assess how well a receiver can hear the signal from the sender.

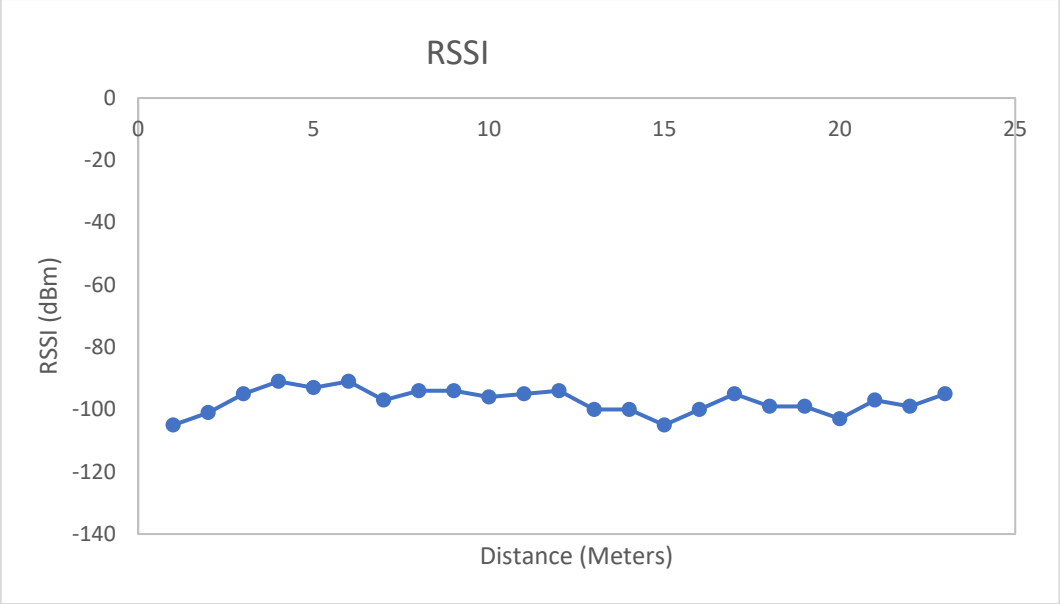


Figure 5.20 Received Signal Strength Indicator

Figure 5.20 represents the ratio of the signal power to the noise power in each transmission. A higher SNR value indicates a stronger and more reliable signal, whereas a lower SNR value suggests a weaker signal susceptible to interference and noise. Overall, the SNR values for "snr-gw2-dragino" varied from very low (e.g., 4.2) to relatively moderate (e.g., 10.0). Values in the higher range, such as 9.0, generally indicate better signal quality with less interference and noise. Conversely, values in the lower range, such as 7.0 and below, suggest a weaker signal that may result in communication issues and reduced data reliability. If connectivity problems or performance issues are encountered with this device, the SNR values may provide insights into the signal quality of the communication system.

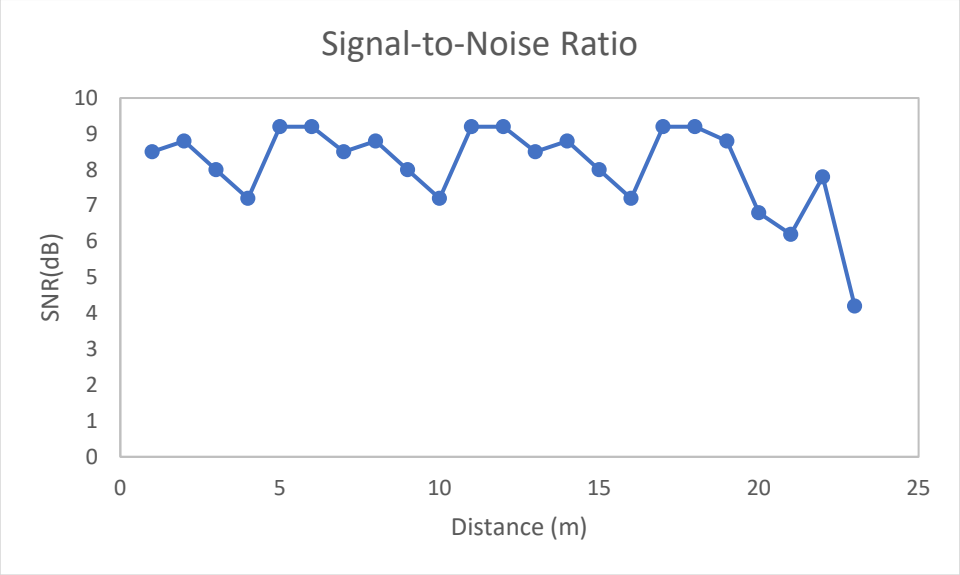


Figure 5.21 Signal to Noise Ratio

5.4 Final testing with Street Elevations

Street-level elevation air pollution monitoring refers to measuring and analyzing air pollution levels at the ground level, specifically on streets and roads in urban environments. Traditional air quality monitoring stations are usually located at fixed points, often on rooftops or open spaces, to monitor the regional air quality. However, street-level monitoring provides more localized and granular data, focusing on the pollution levels where people live, work, and commute.

Street-level elevation air pollution monitoring typically involves the following components.

- **Mobile Sensors:** Instead of fixed stations, mobile air quality monitoring devices are deployed in vehicles, bicycles, or even carried by pedestrians. These devices are equipped with various sensors to measure different air pollutants, such as particulate matter (PM), nitrogen dioxide (NO_2), ozone (O_3), carbon monoxide (CO), sulfur dioxide (SO_2), and volatile organic compounds (VOCs), and more.
- **Real-time Data:** Street-level monitoring aims to provide real-time data, enabling researchers, policymakers, and the public to access information about air quality

conditions in specific areas immediately. This is particularly useful for identifying pollution hotspots and understanding how pollution levels vary throughout the day.

- **Data Integration:** The data collected from these mobile sensors are typically integrated with other air quality data sources, including data from fixed monitoring stations and satellite-based measurements. This integration helps to create a comprehensive picture of air pollution at different spatial scales.
- **Geospatial Mapping:** Street-level air quality data are often visualized on maps to show pollution concentrations across different streets and neighborhoods. Geospatial mapping allows for the easy interpretation of pollution patterns and helps identify areas with higher exposure risks.
- **Health and Environmental Impact Assessment:** Street-level monitoring data can be used to assess the impact of air pollution on public health and the environment. By understanding the pollution levels in specific areas, policymakers can implement targeted measures to mitigate pollution and protect vulnerable populations.
- **Urban Planning and Policy Development:** Data from street-level monitoring can inform urban planning and transportation policies to reduce pollution exposure. For example, identifying areas with high pollution levels can lead to changes in traffic management or positioning of green spaces and vegetation.
- **Public Awareness:** By making real-time air quality data available to the public, street-level monitoring initiatives raise awareness of air pollution issues. Increased awareness can empower individuals and communities to take action to reduce their exposure to air pollutants.

Street-level elevation air pollution monitoring plays a crucial role in understanding air pollution patterns in urban environments and informs targeted strategies for improving air quality and public health. This complements the data collected by traditional air quality monitoring stations and helps create a more comprehensive and localized air quality assessment.

The objective of this research is to monitor street-level pollution at different elevations. To perform the final testing, pole 1 and pole 2 were placed with sensor units in the parking area at

the Auckland University of Technology, North Shore Campus, as shown in Figure 5.22 and Figure 5.23.

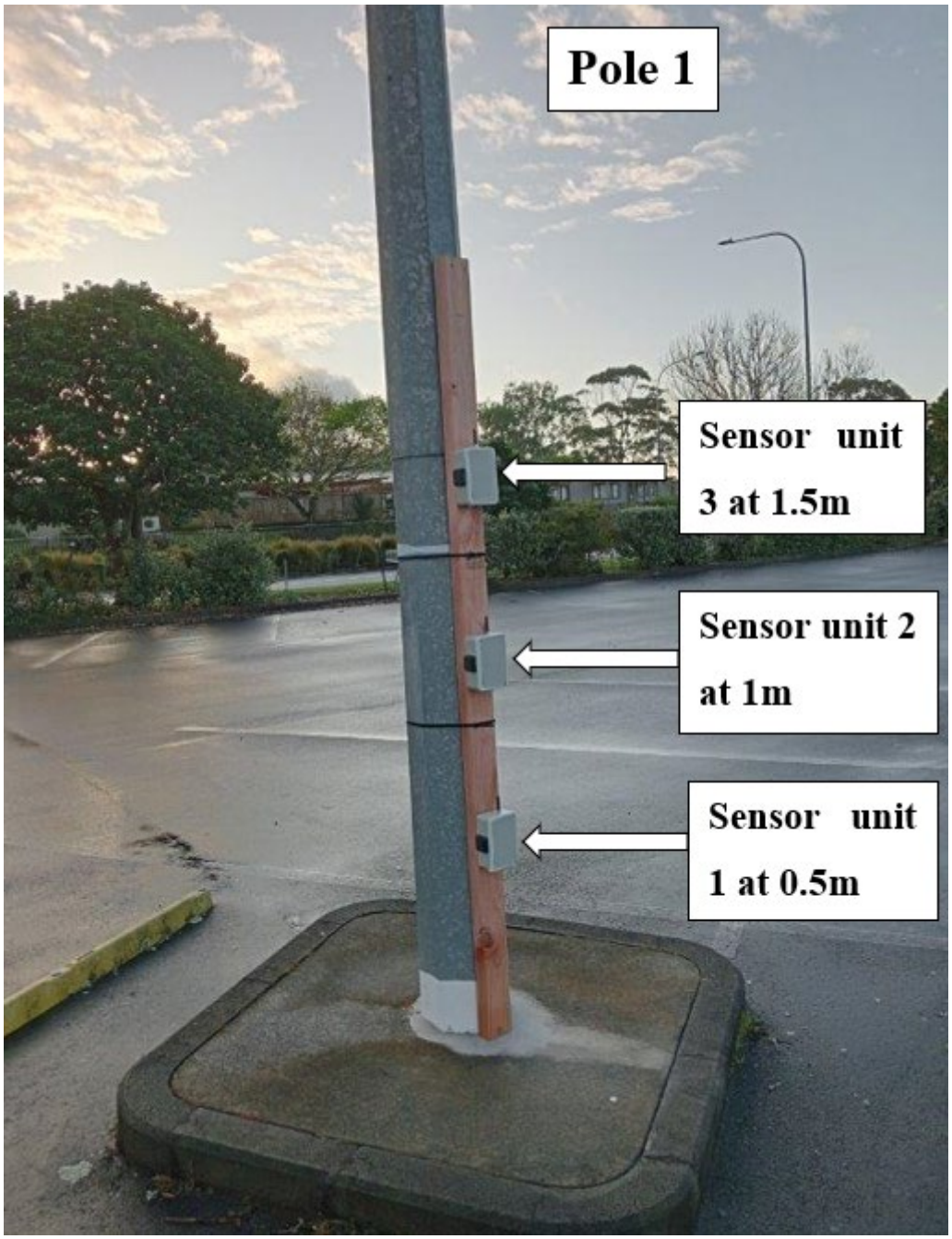


Figure 5.22 Pole 1 with sensor units at different distances

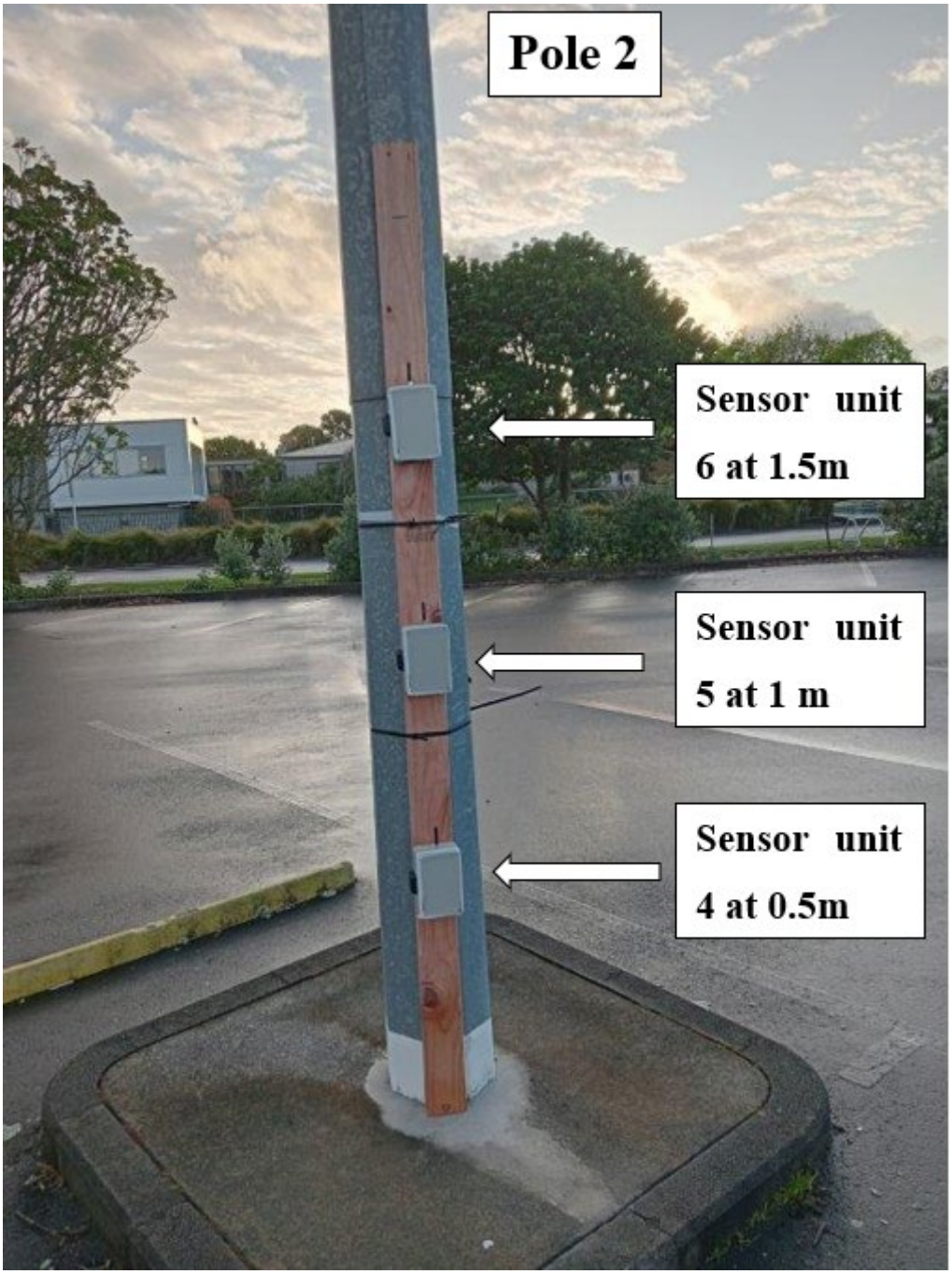


Figure 5.23 Pole 2 with sensor units at different distances

Six sensor units were implemented on the poles to monitor pollution at different levels at distances of 0.5m, 1m, and 1.5m. Sensor deployment was performed using a 2×3 matrix. The gateway was placed inside the reception building (AG106) at 100m from the sensor units. Thus, the gateway and the sensor nodes can communicate with each other.

5.4.1 Results

To evaluate the results and findings, the sensor units were placed on poles with a difference of 0.5m, 1m, and 1.5m. Six sensor nodes communicated with the gateway and network server to show the concentration levels at the university's parking area. In this section, it describes the pollutant levels of CO_2, $PM_{2.5}$, PM_1, PM_{10}, PM_4, temperature, and humidity at different levels. Results explained in Comparison of six nodes to varying heights for 1800 s (10h). After 1800 s, the sensor units were in sleep mode to save energy. The following section explains the example for testing node 1, and the rest of the results are provided in Appendix D.

5.4.1.1 CO_2 concentration levels

The results below are shown for nodes 1–6. Node 1, node 2, and Node 3 were at pole 1 with distances of 0.5m, 1m, 1.5m respectively. The CO_2 level at node 1 (the lowest distance from the ground) indicates a less polluted area. Pole 1 was placed in the middle of the parking area, where the movement of cars was less than that of the other poles. However, the sensor units connected to this pole still had different levels. As shown in Figure 5.24, the difference between the CO_2 levels at different distances can be seen in Appendix D, where the comparison results for all nodes are provided. At a height of 1.5m where node 3 was connected, CO_2 levels started to change with higher values of approximately 2000 ppm in the air, whereas node 1 at 0.5 a distance shows a less polluted area at the exact location.

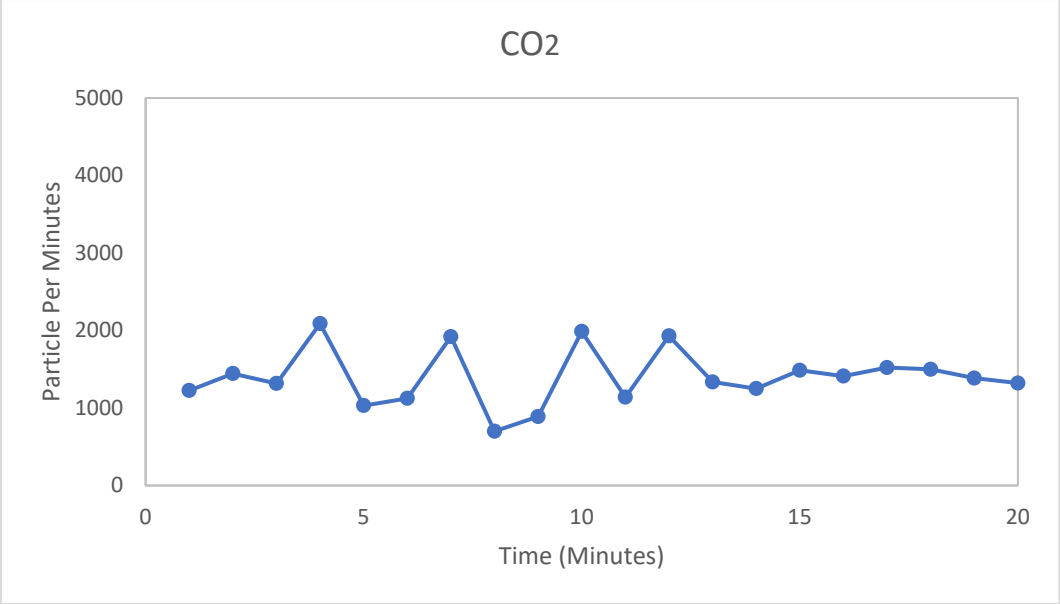


Figure 5.24 CO_2 concentration level at 0.5m (node 1)

Similarly, on Pole 2, three sensor units were connected as nodes 4, 5, and 6 at different distances of 0.5m, 1m, and 1.5m, respectively. Pole 2 is connected at the corner near the highway, where vehicles are moving more, and it can have a more polluted area. Node 4 was connected at the lowest distance from the ground (0.5 m). The CO_2 levels of node 4 were higher than those of node 1 connected at the lowest distance on Pole 1, as shown in Appendix D. Whereas node 5 was placed at 1m from the ground level, it had a higher concentration than node 4 with approx. 2200-2500ppm.

For the temperature measurements, poles 1 and 2 were placed in the university's parking area. Because of the sunny day, there was no significant difference in the temperature levels for the sensor nodes, as shown in Figure 5.25.

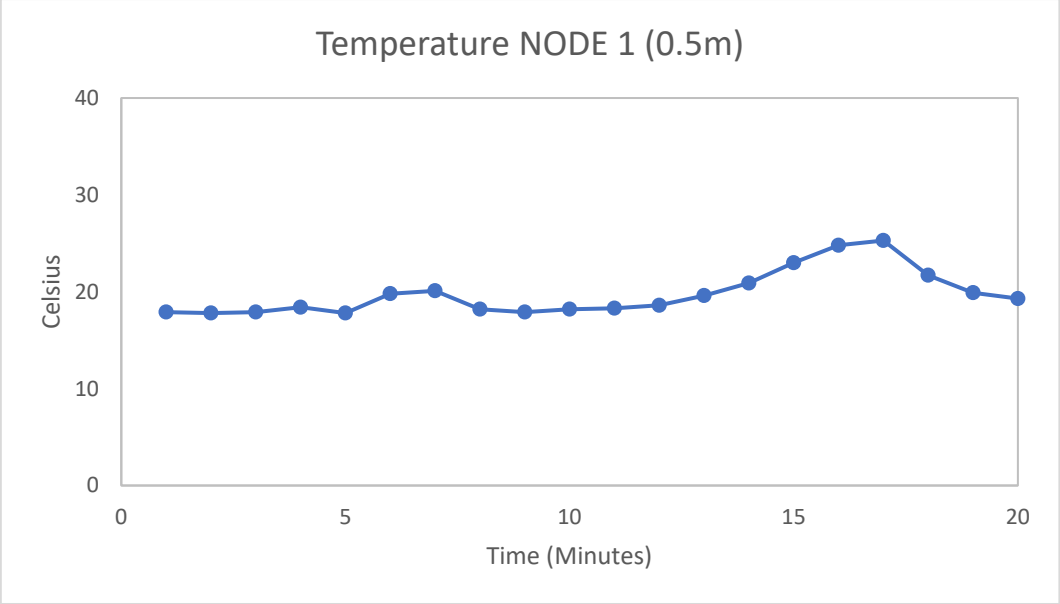


Figure 5.25 Temperature at 0.5m (node 1)

The testing results show fluctuations in the humidity levels of all nodes at different distances at pole 1 and pole 2. Figure 5.26 shows the result for node 1, where the humidity level decreased from approximately 98% to 78% when placed at a distance of 0.5m from the ground. Other sensor nodes at different levels on Pole 1 and Pole 2 show differences in fluctuations, as shown in Appendix D.

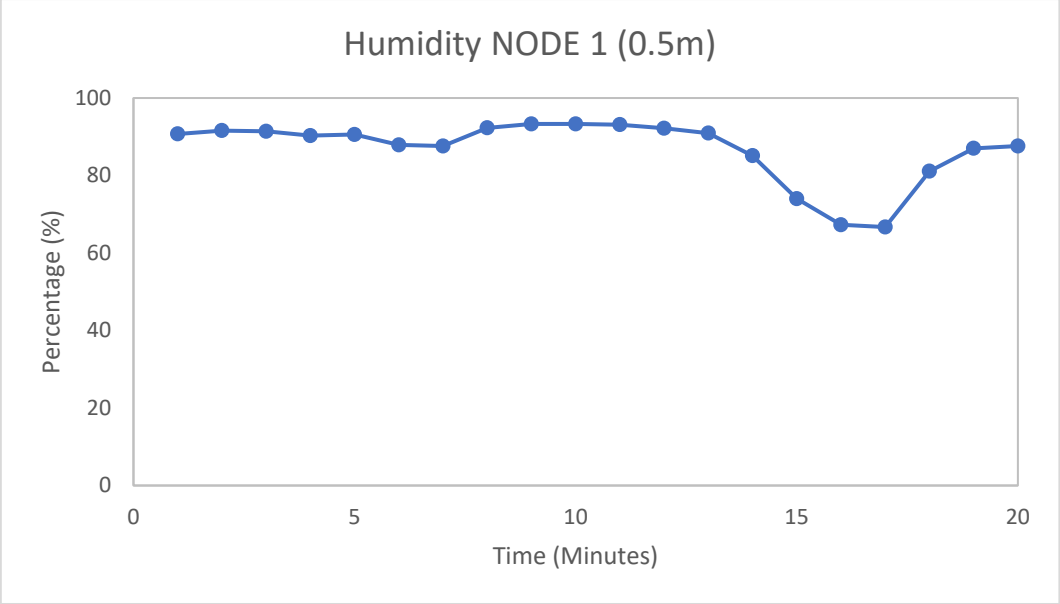


Figure 5.26 Humidity at 0.5m (node 1)

For particulate matter, readings were monitored in the form of PM_1, PM_{10}, $PM_{2.5}$, and PM_4. The concentrations are shown in Figure 5.27 to Figure 5.29 as node 1 is placed at the lowest distance with 0.5 m. It shows some variations at different distances in Pole 1 and Pole 2, see Appendix D.

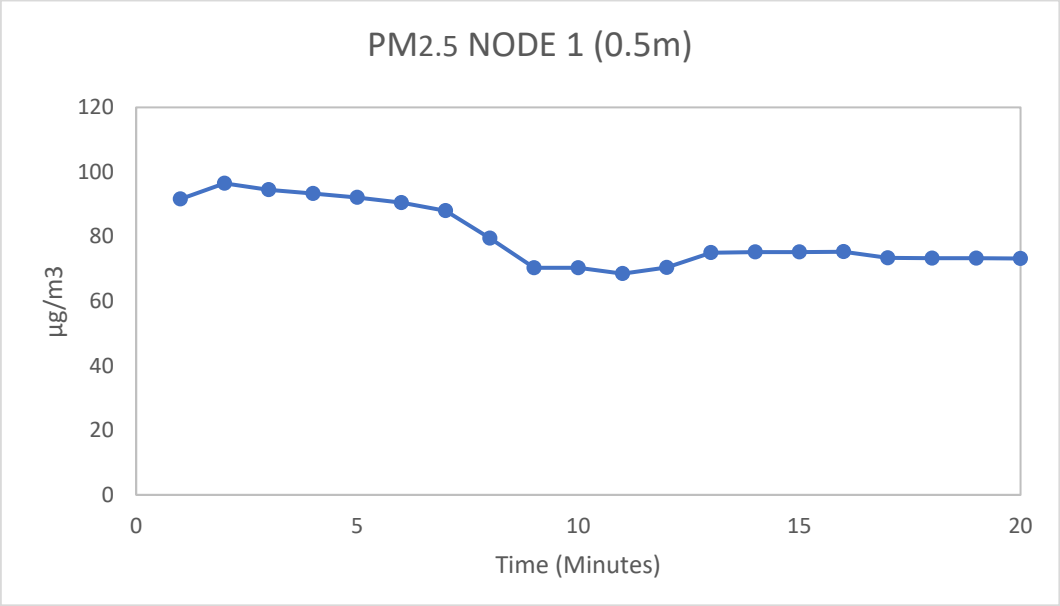


Figure 5.27 $PM_{2.5}$ at 0.5m (node 1)

Particulate Matter (PM) refers to tiny particles or droplets in air that can be inhaled into the lungs. These particles can vary in size and are classified based on their diameter. One standard classification is $PM_{2.5}$, which refers to particles with a diameter of 2.5 micrometres or smaller. $PM_{2.5}$ is particularly concerning because it can penetrate the respiratory system and have adverse health effects. As per the AQI standards, the $PM_{2.5}$ values obtained from testing were relatively low, indicating good air quality. Low concentrations of $PM_{2.5}$ are generally preferred, as high levels can have adverse health effects, particularly for sensitive groups such as children, the elderly, and individuals with respiratory conditions. Regular monitoring of $PM_{2.5}$ is crucial for understanding air quality and implementing appropriate measures to maintain a healthy environment. The $PM_{2.5}$ values range from 0.01 µg/m³ to 2.53 µg/m³. Most readings were relatively low, indicating good to excellent air quality. The highest $PM_{2.5}$ concentration recorded is 2.53 µg/m³, which is still considered to be at a low level. The lowest $PM_{2.5}$ concentration is 0.01 µg/m³, indicating an exceptional air quality.

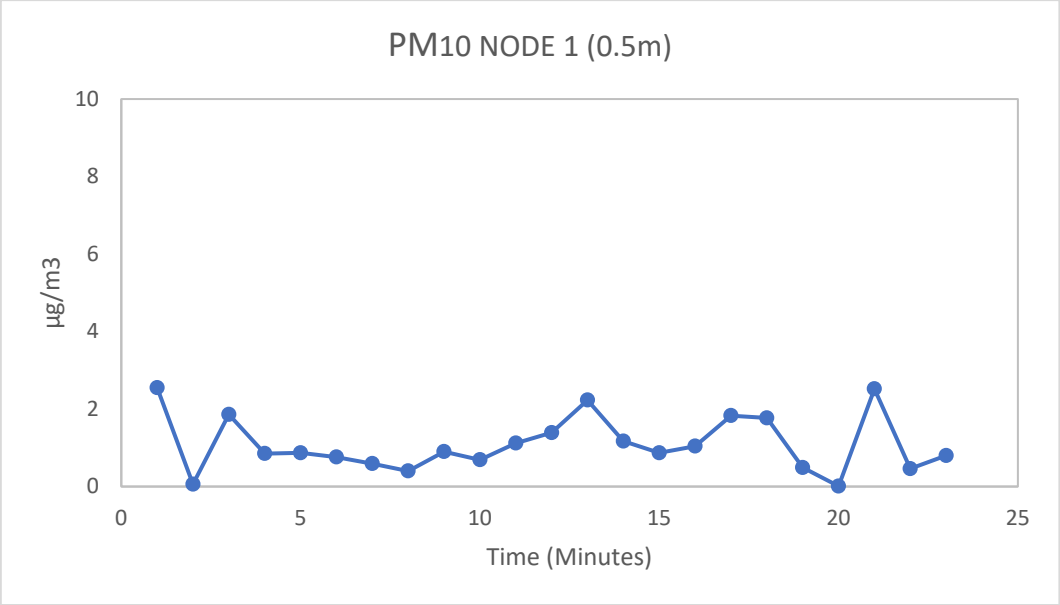


Figure 5.28 PM_{10} at 0.5m (node 1)

The PM_{10} values range from 0.06 µg/m³ to 2.55 µg/m³. Like $PM_{2.5}$, most readings were relatively low, suggesting good to excellent air quality. The highest PM_{10} concentration recorded was 2.55 µg/m³, which is still considered to be at a moderate level. The lowest PM_{10} concentration is 0.06 µg/m³, indicating excellent air quality.

$PM_{2.5}$ and PM_{10} values generally show relatively low concentrations, a positive sign of air quality. The $PM_{2.5}$ values are slightly lower than the PM_{10} values, indicating that finer particulate matter is more dominant in the air. In both cases, the lowest concentrations (0.01 µg/m³ for $PM_{2.5}$ and 0.06 µg/m³ for PM_{10}) indicate exceptional air quality. The highest $PM_{2.5}$ concentration (2.53 µg/m³) was slightly higher than the highest PM_{10} concentration (2.55 µg/m³), but both were still considered low to moderate levels.

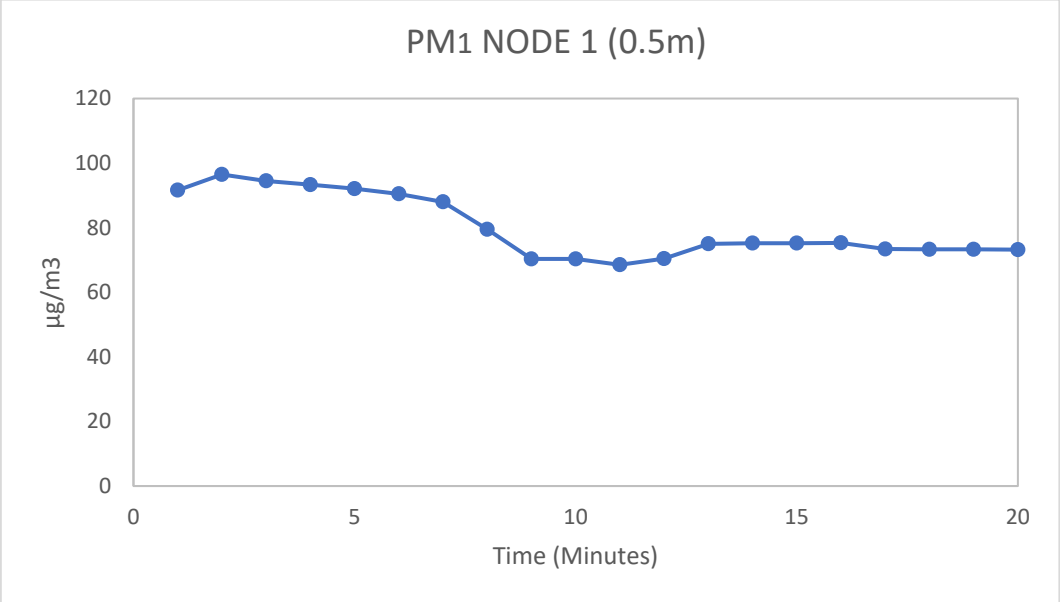


Figure 5.29 PM_1 at 0.5m (node 1)

To assess PM_1 levels and their potential effects on health, continuous monitoring and analysis of air quality data are crucial. Various factors, including industrial activities, vehicular emissions, construction, weather conditions, and geography, influence PM1 concentrations. Understanding PM_1 levels helps implement appropriate measures to mitigate air pollution and protect public health. The PM_1 values range from 0.00 µg/m³ to 2.55 µg/m³. The majority of the readings were relatively low, indicating good-to-excellent air quality. The lowest PM_1 concentration recorded was 0.00 µg/m³, indicating exceptional air quality with no detectable PM_1 particles. The highest PM_1 concentration recorded is 2.55 µg/m³, which is still considered low.

As compared to $PM_{2.5}$ and PM_{10} concentrations, as expected, PM_1 represents the smallest and finest particles. A lower PM_1 value indicates that fine particles (PM1) may not be as prevalent

in the air as larger particles ($PM_{2.5}$ and PM_{10}) in the measured environment. Monitoring PM_1 concentration is crucial because these fine particles can penetrate deeply into the respiratory system and potentially cause adverse health effects. Although PM_1 particles are less abundant in the data provided, they can still pose health risks, especially for sensitive individuals such as those with respiratory conditions or cardiovascular problems.

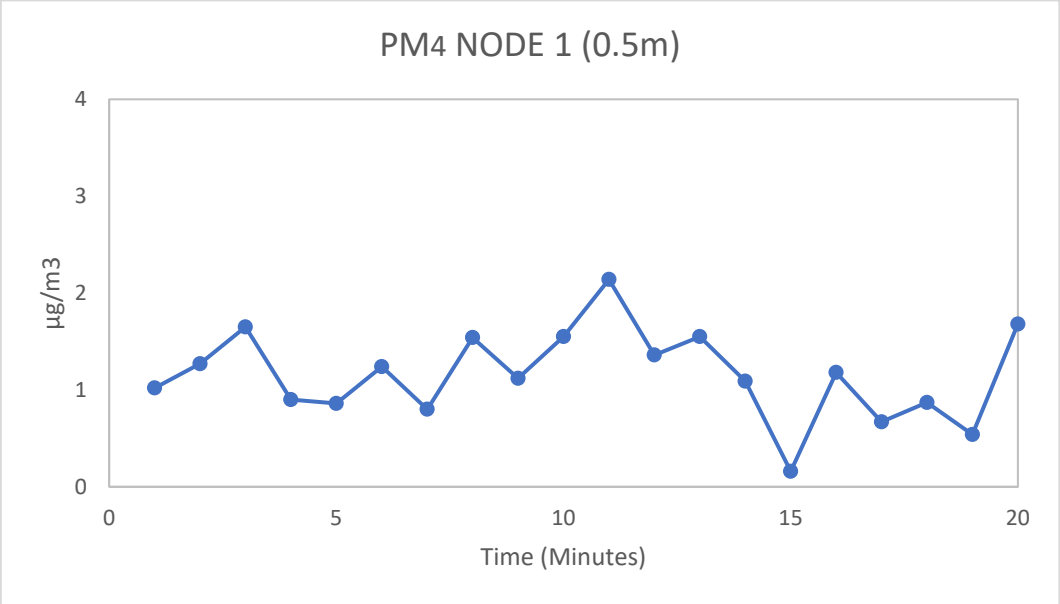


Figure 5.30 PM_4 at 0.5m (node 1)

The PM_4 values range from 0.36 µg/m³ to 2.55 µg/m³. Most readings appeared relatively low to moderate, indicating generally acceptable air quality. The lowest PM_4 concentration recorded was 0.36 µg/m³, which is low. The highest PM4 concentration recorded was 2.55 µg/m³, which is still within a moderate range. As mentioned, PM_4 is not a standard particulate matter category for air quality monitoring. However, it falls between $PM_{2.5}$ and PM_{10}. Given that PM_4 contains particles between 2.5 µm and 4 µm in diameter. The PM_4 values are generally lower than those of PM_{10} but may be higher than $PM_{2.5}$. PM_4 particles have the potential to penetrate deeper into the respiratory system than $PM_{2.5}$, but not as deep as PM_{10}.

5.4.1.2 Received Signal Strength Indicator

Each RSSI value is usually expressed in decibels (dBm), representing the strength of the received signal. Generally, higher RSSI values indicate a stronger signal, whereas lower values

indicate a weaker signal. Overall, it seems that the RSSI values for "rssi-gw2-dragino" are mainly in the range of -93 dBm to -113 dBm as shown in Figure 5.31, which indicates a relatively weak to extremely weak signal. Such weak signal strengths may lead to connectivity issues and signal drops in wireless communication scenarios.

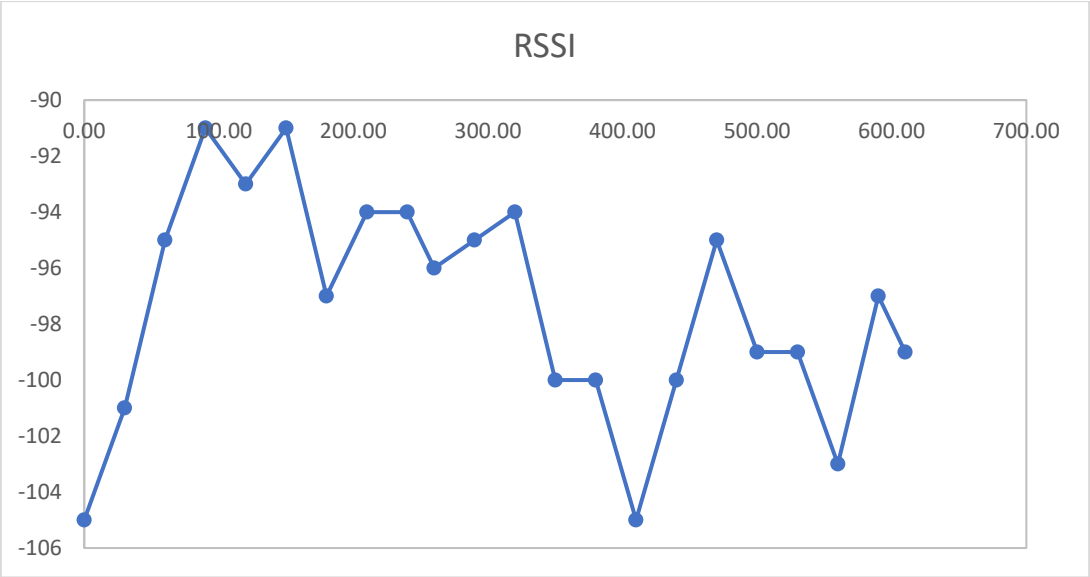


Figure 5.31 RSSI Node 1

5.5 Summary of Results

In the first test, three pollution sensors were used to monitor temperature, humidity, dust, and CO_2 levels in the air. The results were taken for a specific time to understand better the weather conditions and concentration levels of pollutants in the air. From the research and findings during the first test results, the pollution levels were the same for the temperature and humidity, but there were some fluctuations in the air for dust and CO_2. The testing day was sunny in the university balcony area. The LPS8 LoRa gateway was placed in a room 100 m apart. The RSSI is also explained in the graphs, along with other pollution levels. With the help of RSSI, network connectivity and communication between the user and the network server can be monitored using the LoRa gateway.

The second pilot test involved four sensors (PM, CO_2, Temperature, and Humidity) and was conducted over one day to assess their performance and identify any discrepancies in readings

and power consumption. While the sensor units, powered by batteries, generally performed well, the dust sensor consumed excessive power, excluding it from the final testing phase.

The final testing was conducted in a car park at the Auckland University of Technology, North Shore Campus, due to the challenges associated with obtaining permission from the Auckland Council for street deployment. In the car park, a 3x2 matrix configuration was implemented, with Pole 1 housing three sensor units (node 1, node 2, and node 3) and Pole 2 containing the other three units (node 4, node 5, and node 6) at varying heights above ground level (0.5m, 1m, and 1.5m). This testing primarily focused on pollution levels at different elevations and successfully demonstrated variations in pollution levels, providing valuable insights for further research and applications.

Although PM_4 is not as commonly measured as $PM_{2.5}$ or PM_{10}, it can still have health implications, especially for respiratory patients. Fine particles, including $PM_{2.5}$ and PM_4, are of particular concern because they can reach the alveoli in the lungs and enter the bloodstream. Therefore, continuous monitoring and analysis of particulate matter, including PM_4, can help understand air quality and its potential impact on public health. Overall, based on the provided data, both $PM_{2.5}$ and PM_{10} concentrations indicated relatively good air quality. However, continuous monitoring and data analysis are necessary to understand the air quality trends and potential health impacts over time. Air quality can vary depending on various factors, such as location, weather conditions, and human activities; therefore, it is essential to take proactive measures to maintain and improve air quality for public health and well-being.

PM_1 refers to particulate matter with a diameter of 1 μm or less. It is a subset of $PM_{2.5}$, which includes particles with a diameter of 2.5 micro-meters or less, and PM_{10}, which provides for particles with a diameter of 10 μm or less. PM1 particles are very small and can easily penetrate the respiratory system when inhaled, potentially reaching the alveoli of the lungs.

For comprehensive air quality monitoring, it is essential to consider data for PM_1, $PM_{2.5}$, and PM_{10} simultaneously because they provide valuable insights into different particle size ranges and their associated health risks. Monitoring and controlling particulate matter pollution is essential for maintenance. Overall, based on the provided PM_1 data, the air quality appeared relatively good, with most readings indicating low concentrations of fine particulate matter. It is essential to monitor air quality regularly to identify any potential changes and trends in particulate matter concentrations. Implications and significance of the second pilot test and the final testing phase:

- **Technical Refinement:** The second pilot test allowed for a critical evaluation of the technical aspects of the sensor network. It was a valuable opportunity to fine-tune the sensors' performance, ensuring they worked seamlessly together. This iterative process is essential for optimizing the network's reliability and accuracy in data collection.

- **Resource Efficiency:** The decision to exclude the power-hungry dust sensor from the final testing phase underscores the importance of resource efficiency in sensor network design. This choice reflects a pragmatic approach to balancing data granularity with practical limitations. It's a reminder that real-world deployments often require compromises to meet power constraints and budget considerations.

- **Urban Deployment Challenges:** The choice of the car park as a testing location sheds light on the complexities of deploying sensor networks in urban settings. The need to secure permissions from local authorities and navigate bureaucratic processes is a common challenge in urban IoT deployments. This experience highlights the importance of collaboration between research teams and municipal stakeholders.

- **Spatial Insights:** The matrix configuration of sensors at different heights within the car park offers spatial insights into pollution levels. This has wide-ranging implications for understanding how pollution varies across different city areas or even within multi-story buildings. Such granularity in data can inform targeted interventions to improve air quality.

- **Environmental Impact**: The successful demonstration of pollution level variations through the LoRa sensor network underscores its potential positive environmental impact. By providing real-time, localized data, it can support evidence-based policymaking and urban planning initiatives to reduce pollution and its adverse effects on the environment and public health.

- **Translational Research:** The findings from these tests bridge the gap between academic research and real-world applications. The LoRa sensor network has the potential to transition from research environments to practical urban solutions for pollution monitoring and control, showcasing the practicality and relevance of such technology.

In conclusion, the second pilot test and the final testing phase represent crucial stages in the development and validation of the LoRa sensor network. They highlight the need for technical refinement, resource-conscious design, and a deep understanding of the challenges associated with urban deployment. Ultimately, the network's ability to provide spatial insights and support evidence-based decision-making holds promise for addressing city environmental issues.

Chapter 6
Sensor Data Validation

This Chapter is based on a benchmarking tool used for pollution monitoring. Benchmarking is a process in which researchers can discover the gap between actual measurements. For this research, the flowmeter developed by Plume's lab was considered as the benchmarking tool. Data validation testing ensures that the data provided by the sensor units are correct and how this is used, imported, and processed. Data validation testing is required for the network to determine missing data, duplicate data, multiple formats, cluttered data, dependent values, and invalid data. For this research activity, several data validation tools, such as smart air quality monitor, Flowmeter 1, and Flowmeter 2, are in the next section.

6.1 Flow 1 and Flow 2

The flow is a portable device with interconnected sensors for monitoring air pollution. It monitors pollutants affecting human life and the environment, including $PM_{2.5}$, PM_{10}, NO_x, and volatile organic compounds (VOC). Flow was developed to help the community be aware of pollution levels and how these affect the environment and humans. There should be some equipped with it, as it is elegant, easy to use and carry, and accessible to non-scientific audiences. Currently, this is the most effective tool for sharing air pollution challenges and encouraging solutions for resolving pollution issues. Flow can provide instant insight into personal exposure. It was only required to press a single button for connectivity and animated LEDs, as shown in Figure 6.1 .

Figure 6.1 Flow 1

6.1.1 Working of Flow

A significant development in the history of air quality monitoring systems is the handheld revolution, known as Flow. Air quality monitoring usually involves many larger-than-flow instruments. This type of pollution monitoring equipment was never intended to evolve. Consumers can hold that in their hands by sufficiently condensing the vast array of scientific components inside the svelte polished steel sleeve. Three things combine to form a flow.

- **Hardware**: The device and components are tucked inside the chamber's grey casing.
- **Firmware**: programs uploaded onto the hardware that powers the sensors and churns out measures.
- **Software**: The application of the flow installed on the device (Android or iOS) and gateway to all data collected.

6.1.2 Principles

Step 1: Air enters flow: When air enters the flow, it uses the holes carved that go around, and then it is called “360 air intake” [120]. To ensure that the air enters the flow, it has a fan fit for the optimization of the air inside. It may be as small as 5 mm, but it can monitor 15,000 RPM, and the user can hear a gentle buzz using an eardrum [120].

Step 2: In this step, the PM sensor lights up and shoots the laser beam in the air brought into it using a fan connected inside [120]. When a particle hits, laser light is dispersed. The micro light shown can be detected by the photovoltaic cell inside it and translates the laser-deflected beam into an electrical current.

Step 3: The NO_2 and VOC sensors operate by violently collapsing any NO_2 and VOC molecules that pass across a small membrane heated to 350 °C. This gauges the energy fluctuations required to maintain the membrane temperature constant.Therefore, this type of difficult battery management necessitates that flow remains active all day. The warm-up took 15–30 min.

Step 4: All connected sensors are calibrated. Temperature and humidity vary as wood expands and contracts. This shows changes in string tension by a small amount [120]. The components of the flow can be changed, and the flow is unique every time. Calibration algorithms are used to ensure that all these differences are accounted for.

Step 5: Neural Networks in Flow are activated. These are trained programs for detecting patterns in the data. These effectively determine the PM, NO_2, and VOCs in the air pulled into the flow [120]. Neural networks have been designed to mimic the human brain.

Step 6: The created patterns were converted into measures - particles per billion (PPB) for VOCs and NO_2 and micrograms per cubic meter for PM. Flow connection analyses billions of particles every day [120].

Step 7: Flow is applied to filter good pollution monitoring. The flow is easy to carry wherever the user wants to go. Within a day or month, users can explore all types of places and environmental conditions [120]. This may show unexpected irregular spikes in the flow measurements. Flow has built virtual data scientists to review readings in real time and send misleading measures for homes faster than the "air quality index."

Step 8: Filtered messages from the flow are converted into AQI. It monitors particles and micrograms but, most importantly, determines their health impacts. Plumes have their own AQI to provide an immediate overview of the pollution levels [120].-

Step 9: The AQI receives visual treatment on the screen near the users. Open the Flow app, and Flow will show the AQI data stored in the smartphone as soon as it detects the Bluetooth connection [120]. Flow can also pair AQI data with locations into colourful maps.

Step 10: Air pollution data is improved when it is shared. When the smartphone is connected to Wi-Fi or 3G/4G, the data are stored on the servers safely. This is how Flow works and shows the pollution levels in the air [120].

6.1.3 Setting Connections

To start with Flow 1 and Flow 2, the app must first be downloaded from the app store on Android and iOS devices. It has two apps: Plume Labs and Flow [121]. Flow is an appropriate app for monitoring pollution levels. Once downloaded, it appears as in Figure 6.2 .

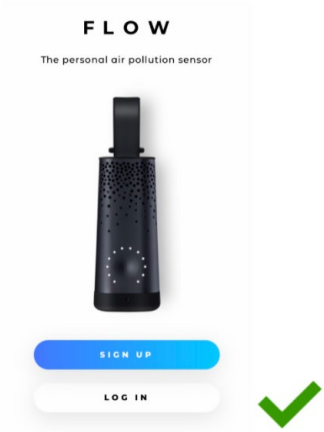


Figure 6.2 Signup with application

Returning users can sign up for the app directly. If users are new to the app, they must sign up and register the device with their mobile phones. The privacy policy for the Flow app is shown in Figure 6.3 [121].

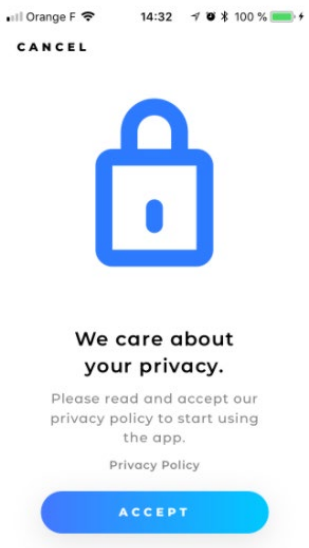


Figure 6.3 Privacy Policy

Once users accept the policies and privacy, they start the account setup using email and password, as shown in Figure 6.4 ; click next to add the profile's first name and last name, and click on create [121].

Figure 6.4 Creating Profie

To begin creating the profile and installing an application on the phone. The user needs to ensure that the flow is near the mobile device to avoid any issues with the Bluetooth connection. It will show you the below on the screen as Figure 6.5, and tap on the get help. It has some helpful articles for new users in the get-help section. Tap at the start of installation [121].

Figure 6.5 Step to Start Flow

Press the button to wake up the flow meter and click on next. Then, users will receive a notification on mobile with the detected flow, as shown Figure 6.6.

Figure 6.6 Flow Detected

If the Flow has been detected, it will alert users on the mobile phone, ask them to select the flow, and then tap on next. The detected flow near a user's mobile phone is shown in Figure 6.7. It offers all flows near the Bluetooth device. The user needs to match the first six digits of the flow and connect the correct flow with the mobile connection [121].

Figure 6.7 Selecting Flow

Once connected, it starts showing a flashing blue light around the button and clicking on the next for an update in Flow. The flow was updated, as shown in Figure 6.8. It takes only a few minutes to show the congrats message on the screen when updated.

Figure 6.8 Updating Flow.

6.2 Plume Lab AQI

AQI is known as Air Quality Indexing. According to its name, this is an indicator allowing users to access air quality with a whole or single value. The AQI shows the concentration levels expressed in µg / m3 for all different pollutants that can impact human health. If the AQI shows a higher value, it has a higher risk, whereas a lower AQI has a low health impact. AQI has different thresholds using different pollutants and calculation methods and uses several categories. Other countries have their own AQI, such as the Canadian AQI and Chinese AQI. Similarly, Plumes have their own AQI.

The Plumes Air Quality Index

At plume labs, they studied different AQIs. The plume lab observed two things [122]. First, none of the AQI is based on any concrete meaning for individuals and not on health impact thresholds. To address these issues, plume labs built their own Plume Air Quality Index [122]. It has seven pollution levels and thresholds. These are linked to the exposure limits outlined by the WHO, as shown in Figure 6.9 Plumes AQI [122]..

GAUGE	PAQI VALUE	MEANING
10 AQI	Low, 0-20	The air is clear—perfect for outdoor activities!
35 AQI	Moderate, 21-50	Air quality is considered acceptable. However, there may be certain health concerns for people with specific sensitivities.
75 AQI	High, 51-100	Above 50, pollution is high, everyone may start to experience more serious health effects. Long term exposure constitutes a real health risk.
125 AQI	Very High, 101-150	The air has reached a very high level of pollution. Effects can be immediatly felt by individuals at risk. Everybody feels the effects of prolonged exposure.
175 AQI	Excessive, 151-200	The pollution levels has reached a critical level. Individuals at risk feel immediate effects. Even healthy people may show symptoms for short exposures.
225 AQI	Extreme, 201-250	The pollution has reached extreme levels. Immediate effects on health.
275 AQI	Airpocalypse, 251+	Airpocalypse! Immediate and heavy effects on everybody.

Figure 6.9 Plumes AQI [122].

6.3 Results

According to the plume lab application flow, the PM (2.5, 10, 1), VOC, and NO_2 were monitored. It was placed indoors with a Bluetooth connection to communicate with the flowmeter. The flow collects the pollutant levels using the built-in sensor system and sends records to the application on a smartphone. After data collection, the user can export the data from the application in a CSV file. Users can create graphs to check the pollution levels. Pollution levels monitored from the flow are shown in Figure 6.10 - Figure 6.15.

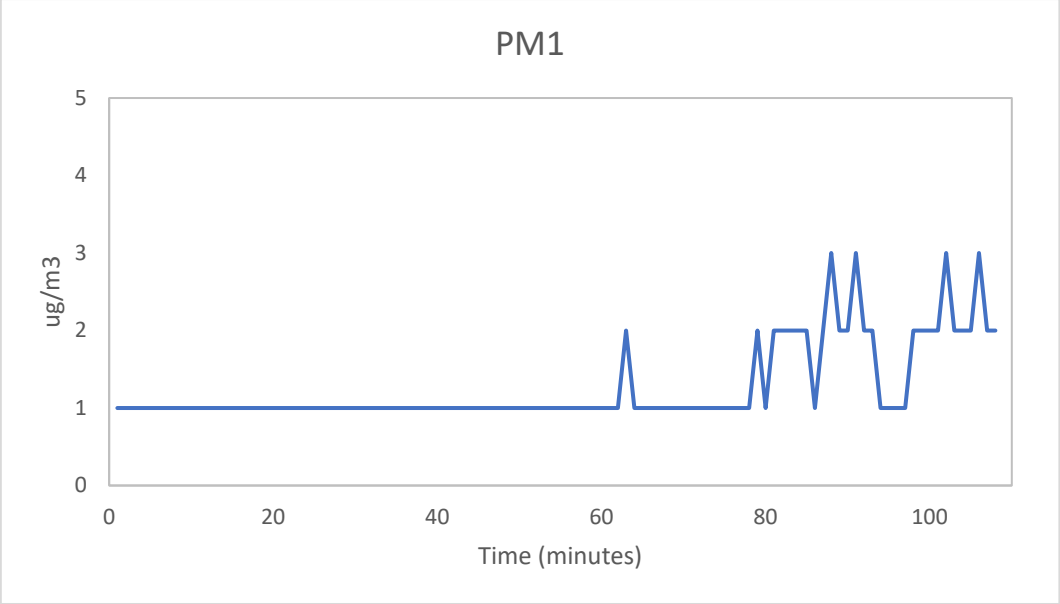


Figure 6.10 PM_1 Concentration Levels

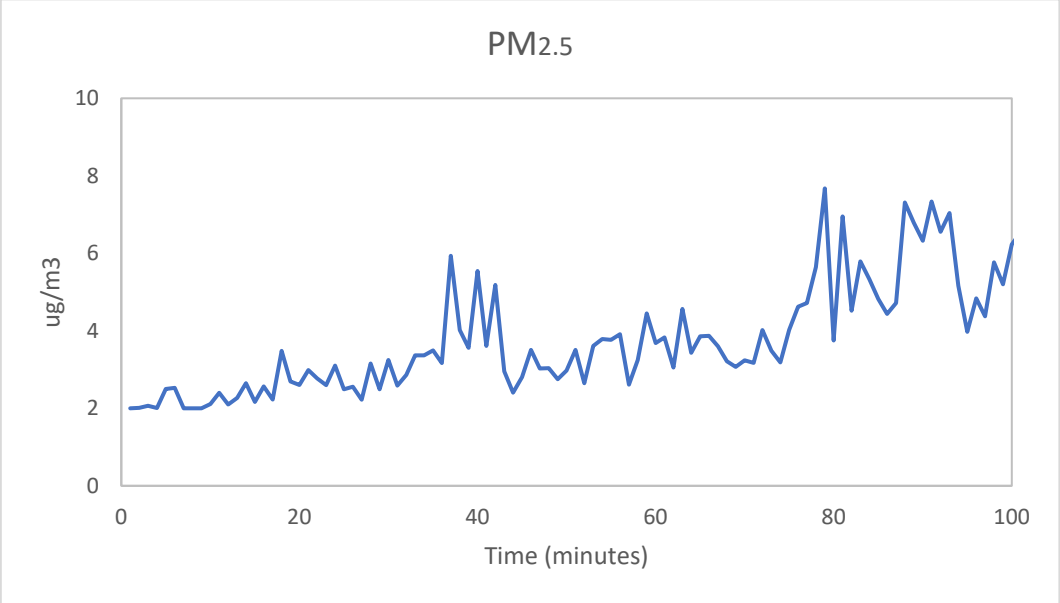


Figure 6.11 $PM_{2.5}$ Concentration Levels

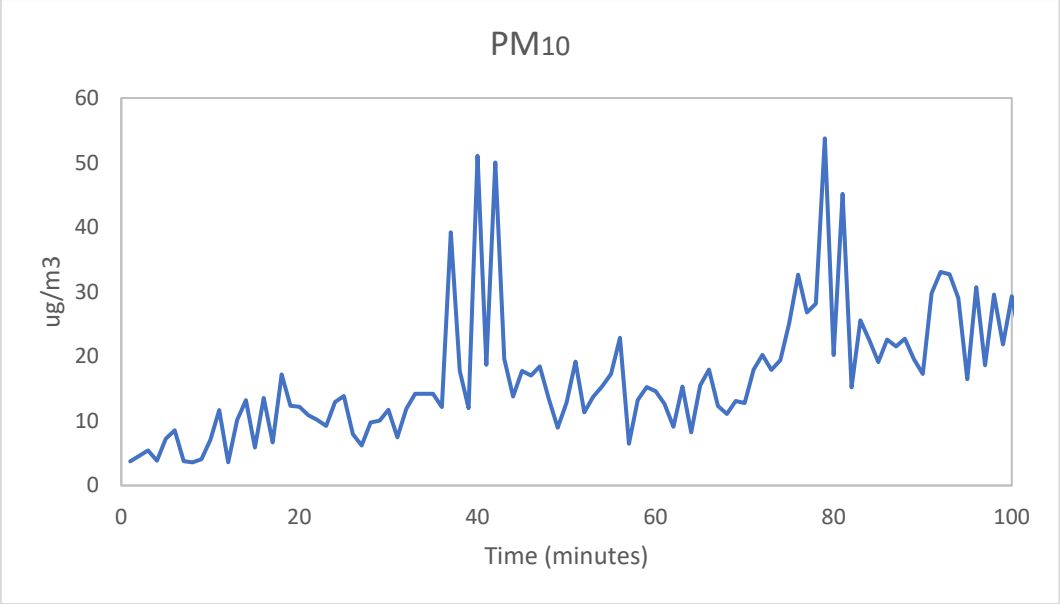


Figure 6.12 PM_{10} Concentration Levels

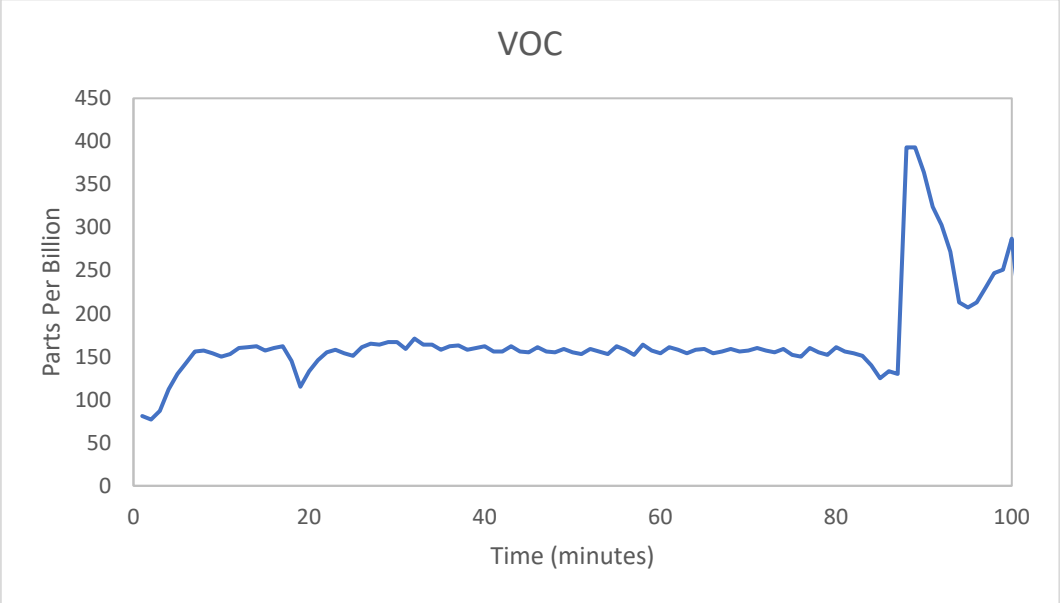


Figure 6.13 VOC Levels

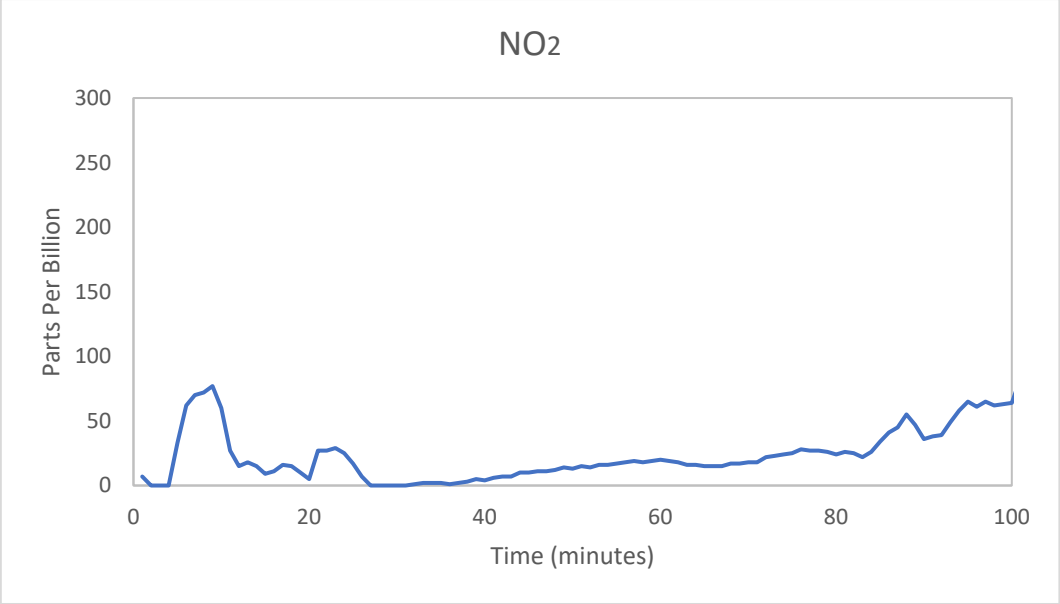


Figure 6.14 NO_2 Concentration Levels

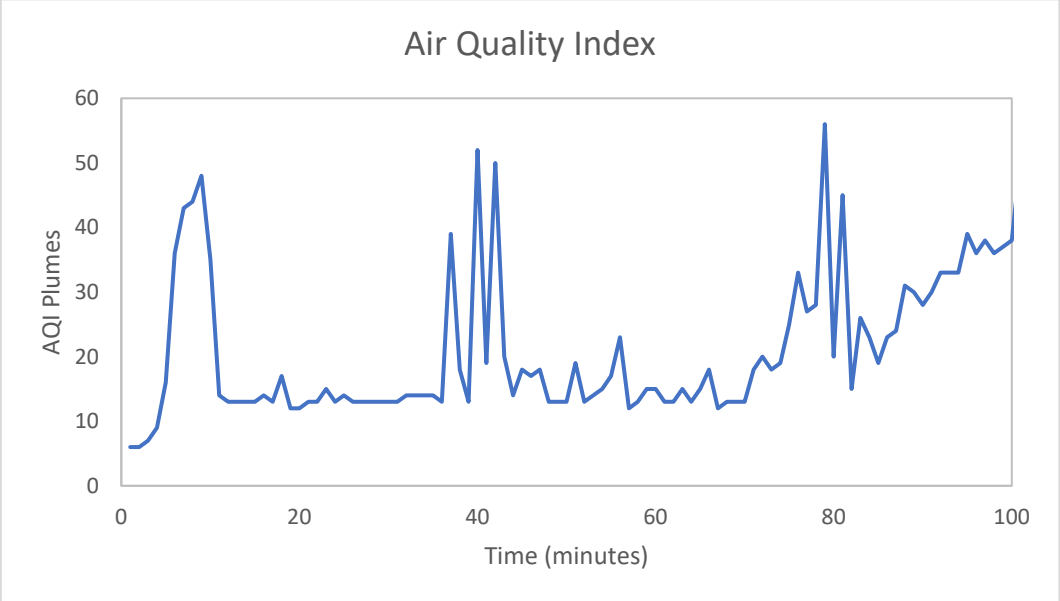


Figure 6.15 AQI by Plumes

6.4 Smart Air Quality Monitor

Smart air quality is a multifunctional air quality monitor that is used to monitor carbon dioxide (CO_2), formaldehyde (HCHO), Total Volatile Organic Compounds (TVOC), particulate

matter, temperature, and humidity with clock and recording functions, as shown in Figure 6.16 . This device combines multiple air sensors and a built-in fan to monitor concentration levels. It has some specific features for use indoors and outdoors or in cars. There was no need to buy separate sensors at all concentration levels.

Figure 6.16 Smart Air Quality Monitor

6.4.1 Features

Write something here

- It has 6 a 1 sensor system to monitor particle matter2, temperature, humidity, CO_2, and TVOC.
- It has two calibration modes: manual and automatic.
- It can easily connect to the Internet using the Tuya App, set alerts, and receive air quality measurements.
- It includes a wireless IoT controller that activates smart fans, air purifiers, and other appliances.
- This supports RS485, located on the rear of the device.
- It has a large LCD showing the alarm settings, colour changes, and pollution levels.

6.5 Summary and contribution

This chapter focuses on the data validation tools and methods used during the research activities. Before obtaining the results, it was necessary to check the accuracy and effectiveness of the data. Two different tools are used to verify the accuracy of the sensor data. As in the deployment, three sensors were used: CO_2, PM, temperature, and humidity. Flowmeters 1 and 2 are used to validate the PM values. First, they were tested inside the room and placed next to the sensor units. It shows the approx. There were similar values for particulate matter (PM_{10}, $PM_{2.5}$, and PM_{10}). Monitoring the CO2 concentration level takes time to warm up and shows the wrong values for the first few minutes, but the CO_2 data are validated with a new smart air quality monitor. These data validation tools are easily available at low cost, as they both have a self-calibration built-in system to show accurate sensor values.

Certainly, let's delve further into the significance of this data validation chapter and how it contributes to the overall research process:

- **Data Quality Assurance:** This chapter essentially serves as a data quality assurance checkpoint. Ensuring that the data collected accurately represents environmental conditions is crucial for drawing valid conclusions and making informed decisions based on the research findings. By meticulously validating sensor data, the research project maintains high data integrity.
- **Enhancing Research Reliability:** The validation of sensor data goes a long way in enhancing the reliability of the entire research endeavour. It minimizes the chances of erroneous or misleading data influencing research outcomes, which is especially critical in scientific studies where precision is paramount.
- **Scientific Rigor**: Rigorous data validation is a hallmark of scientific research. It demonstrates the commitment to sound research methodology and the scientific method's principles. The chapter exemplifies the dedication to adhere to best practices and the highest scientific rigour standards.
- **Transparency and Accountability:** In research, transparency and accountability are essential. The detailed description of the validation process, including the tools used and their calibration mechanisms, adds a layer of transparency to the research

methodology. This transparency allows other researchers to independently replicate the study and verify its results.

- **Generalizability:** Validated data holds greater potential for generalizability. When researchers confidently claim that the data collected accurately reflects the phenomena under investigation, the findings can be more confidently applied to broader contexts or populations.
- **Impact on Decision-Making:** Data accuracy directly impacts decision-making in fields like environmental monitoring and air quality management. Policy decisions, public health interventions, and environmental regulations rely on accurate data. The data validation ensures that the research can contribute meaningfully to such choices.
- **Continuous Improvement:** Finally, the chapter underscores a commitment to continuous improvement. If discrepancies or inaccuracies are identified during the validation process, they can be addressed and corrected. This iterative approach ensures that the research's data collection methods and instrumentation are refined for future studies.

This chapter serves as a basis for research quality, elevating the study's trustworthiness, credibility, and potential impact. It aligns with scientific inquiry's core principles and reinforces data accuracy's importance in advancing knowledge and addressing real-world challenges.

Chapter 7
Conclusion and Future Work

This chapter first reviews the motivations and objectives of this study. Then, the chapter will briefly outline the systems or scenarios used in this research. Ultimately, conclusions, recommendations, and possibilities for further study are required.

7.1 Reviews of Motivations and Objectives

Air pollution is a significant concern and a substantial air pollutant emitted from vehicles, construction, and many other sources. Monitoring is required to minimize the impact of pollution on human health and the environment. Different pollution monitoring methods have been implemented; however, street-level pollution is also the main consideration. Some of the implemented pollution monitoring systems are not cost-effective, efficient, or real-time. Different pollution-monitoring models have identified several limitations, including delays and changes to the network server and components. These monitoring results can be used to assist researchers in the development of real-world pollution monitoring systems. This research aims to estimate the LoRa network and implement it in a real-time system at the street level.

7.2 Reviews of Methods

For this research, three subsystems have been developed to support the main objective.

Sub-system 1- Simulation

The simulation of the desired network was performed in ns-3 simulator using different parameters, packet sizes and time frames. LoRa network was simulated with similar parameters to analyse their performance regarding packet delivery ratio, network throughput and total delay during transmission. The simulation was performed on the NS-3 simulator, and results were obtained in transmitting packets, receiving packets, packet size, packet drop, packet loss, total delay, delay per packet, and throughput.

Sub-system 2- Network Deployment

The network deployment is the second scenario of research, where the LoRa network was deployed using a 3 × 2 matrix with sensor modules in the final deployment. The first two tests were performed using four LoRa modules with CO2, temperature, humidity, dust, and PM sensors. These LoRa modules communicate with the LoRa LPS8 gateway using 915MHz, and the gateway communicates with The Things Stack using a wireless connection. The final proposed network for this research is based on a 2 × 3 matrix that will use a cluster tree network at Carpark North Shore Campus Auckland to monitor CO2, PM, temperature, and humidity concentration levels.

Sub-system 2- Data Validation

After data collection and assessment of the results formed from the real-time simulations done and compared with the Flow unit developed by Plume's labs, the Flow is an all-in-one team consisting of hardware, software and firmware, and the air being monitored enters the unit via '360 air-intakes'. Another tool used is the Smart Air Quality monitor to provide the concentration levels of carbon dioxide (CO_2), formaldehyde (HCHO), Total Volatile Organic Compounds (TVOC), particulate matter, temperature, and humidity.

7.3 Contributions

This study aims to develop a low-cost pollution monitoring system for the visualization and assessment of pollutants at the street level. This information can be used to evaluate LoRa network performance and pollution levels. Specifically, the proposed system can be expressed as:

- The network performance was estimated using the ns3 simulator for the LoRa network. How will the designed network monitor pollution levels in the real world? The ns3 simulator has the features of installing LoRa modules to implement the LoRa network and measuring the network's performance in terms of packet-to-delivery ratio, end-to-end delay, and network throughput in the designed network area.
- Monitoring real-world pollution levels at the street level using a low-cost air pollution

monitoring system. This provides additional possibilities for an existing air pollution monitoring system based on an expensive monitoring system.

- Air pollution was dynamically and continuously visualized to understand better the trends in particulate matter (PM_1, $PM_{2.5}$, PM_{10}, and PM_4).
- Enhance the monitoring system by providing a user-friendly interface to review pollution data that are ubidots, where pollution data can be visualized and broadcast live graphs.

7.4 Conclusion

The overall aim of the research reported in this thesis was to monitor the pollution levels at street elevations and to investigate and develop methods to check the variability in pollution levels. The analysis of these pollution monitoring systems will assist in the creation of more effective methodologies for pollution monitoring systems.

After thoroughly reviewing the existing literature, a new methodology was designed and implemented for this research. The literature review identified several techniques and methods for pollution monitoring systems, but none were performed based on street elevations. Hence, the new sensor modules were designed for street-level pollution monitoring and can be used for different applications. The designed sensor units include the LoRa ESP32 board's three sensors (CO_2, PM, and temperature and humidity sensors). LoRa ESP32 boards were programmed using an Arduino IDE using different in-built libraries.

In chapter 2, it underscores the critical importance of addressing air pollution as a global concern with severe health and environmental consequences. It highlights the evolution of air pollution monitoring systems using various network technologies and the need for more efficient and cost-effective solutions. In summary, addressing air pollution through innovative technologies like LoRa-based sensor networks is a crucial step towards mitigating the adverse effects of pollution on public health and the environment. These efforts enhance current monitoring capabilities and contribute to the long-term sustainability of urban environments and human well-being. The use of LoRa technology enables real-time data transmission over

extended distances. This capability allows for continuous air quality monitoring, providing timely information on pollution levels. It can also facilitate the development of early warning systems that alert authorities and the public to pollution spikes, helping individuals take necessary precautions. LoRa-based sensor networks can enhance data accessibility and transparency. These networks can promote greater awareness of air pollution issues by making air quality data readily available to the public, policymakers, researchers, and advocacy groups. Transparent data can foster community engagement and encourage collective efforts to address pollution. The deployment of advanced sensor networks bridges the gap between environmental science, technology, and policy. It encourages cross-disciplinary collaboration among scientists, engineers, urban planners, and policymakers. Such collaborations can lead to innovative solutions and data-driven policies to reduce pollution and improve overall urban quality of life.

Adopting low-power LoRa technology for air pollution monitoring is not merely a technical innovation but a comprehensive approach with wide-ranging implications. It can transform how we perceive, address, and mitigate air pollution, leading to healthier, more sustainable, and more equitable urban environments. It represents a step towards a future where technology, data, and collective action converge to combat one of our time's most pressing global challenges.

Chapter 3 highlights the significance of Low-Power Wide-Area Network (LPWAN) technologies, particularly the LoRa framework, in enabling advanced network evaluation and performance assessment within the LoRaWAN system. Network simulation, specifically employing the ns-3 simulator, is presented as a valuable tool for modelling LoRa networks with their characteristic long-range communication capabilities and low power consumption profiles. The chapter elaborates on the three crucial performance metrics used to evaluate LoRa networks, namely packet delivery ratio, end-to-end delay, and network throughput. These metrics provide a comprehensive understanding of how efficiently the LoRa network operates in various scenarios. The performance analysis results demonstrate that the designed LoRa network outperforms other network devices, such as conventional wireless sensor networks. This superiority underscores the potential of LoRa technology in providing reliable and efficient communication, particularly for applications like air pollution monitoring.

The chapter further integrates the insights gained from network performance evaluation with real-world applications. It provides a detailed explanation of estimated pollution levels in a specified area, comparing these findings with those presented in a previous chapter. This comparative analysis showcases how the LoRa sensor network excels in monitoring and addressing air pollution, underscoring its potential effectiveness in this critical context. Chapter 3 bridges the technical aspects of LPWAN technology and its practical application in addressing pressing environmental concerns, such as air pollution. It demonstrates how advanced network evaluation techniques can enhance the reliability and efficiency of LoRa-based sensor networks, offering a promising avenue for improving air quality monitoring and other related applications.

In the first scenario, a network simulation was performed to check the estimated performance of the network. Using different network parameters, a LoRa sensor network was designed and simulated. Variations in performance metrics can be used to determine the best combination of networks. Other parameters to compare the results for all network performance variations were the packet delivery ratio and throughput of the network.

The proposed network simulation was performed based on LoRa network simulation using the ns3 network simulator. The ns3 simulator is the only one with the features of installing LoRa modules and generating dummy sensor values. To create dummy sensor nodes, only ns3 has specific features where users can create dummy devices to communicate with other devices and provide the required results. This is quite difficult to install; however, after installation, it is good to go, and users can estimate the performance of the LoRa network. The proposed LoRa network was simulated in ns3.26, which is compatible with the GnuPlot and other important features. Two code files were used to perform the simulation process: the main code files and. NetDevice file. The NetDevice file contains elements that generate dummy sensors, such as PM, dust, CO_2, temperature, and humidity. These sensor nodes have a set of specific ranges to generate pollutant levels for the sensors and are deployed in a specific area of the network. The simulation showed excellent network performance in terms of packet delivery ratio, network throughput, and end-to-end delay.

Chapter 4 provides a comprehensive overview of various wireless communication technologies, focusing on LoRaWAN, Sigfox, and NB-IoT, and offers a comparative analysis. It establishes LoRa technology as the preferred choice due to its exceptional attributes,

particularly its long-range capabilities and low power consumption. Additionally, it introduces The Things Network and elucidates its network server's features and versatility for diverse applications. The chapter delves into IoT network architecture, encompassing different technologies and applications, elucidating network topologies, and highlighting their role in enhancing network performance. The addition of The Things Network underscores its significance as a versatile platform for deploying IoT solutions. The chapter outlines its architecture and how it interfaces with various technologies and applications, offering readers valuable insights into its adaptability. The discussion on network topologies helps readers understand the structural design of IoT networks. It highlights the importance of optimising /network performance and efficiency by selecting the correct topology for specific deployment scenarios. The chapter's exploration of the hardware and software used in the designed network offers a practical perspective on implementing pollution monitoring systems. This section showcases the technical elements involved in monitoring environmental data.

Explaining sensor unit operation via flowcharts elucidates the step-by-step data transmission and communication process with The Things Stack. This visual representation aids in comprehending how sensor data is collected, processed, and transmitted within the network. The chapter provides insights into the performance of various sensors, including dust and CO_2 sensors. It highlights issues related to power consumption and accuracy, leading to replacing the dust sensor with the more reliable SPS30 sensor. This chapter is a foundational guide to understanding wireless communication technologies in the context of IoT networks for pollution monitoring. It explains the technical aspects and addresses practical challenges and solutions encountered during network deployment, ultimately contributing to the development of efficient and reliable environmental monitoring systems.

In Chapter 5, the series of pilot tests conducted to monitor pollution levels using a network of sensors and LoRa technology have yielded valuable insights into the monitoring system's performance and provided essential data regarding air quality in specific environments. The initial test, which focused on monitoring temperature, humidity, dust, and CO_2 levels, revealed that temperature and humidity remained relatively consistent, while dust and CO_2 concentrations exhibited fluctuations. The sunny weather conditions on the testing day likely contributed to these variations. The LPS8 LoRa gateway enabled real-time monitoring and network connectivity assessment through RSSI data.

The second pilot test, involving four sensors (PM, CO_2, Temperature, and Humidity), assessed sensor performance and power consumption. It was determined that the dust sensor consumed excessive power, leading to its replacement. The final testing phase, conducted in a car park with a 3x2 matrix configuration, focused on pollution levels at varying heights above ground level. This configuration allowed for observing pollution variations at different elevations, providing valuable insights for future research and applications.

The pilot and main research studies will use a similar network design for network deployment. After determining the different performance parameters and results, they were compared with the network deployment results. This research will be extended according to network requirement data collection. The LoRa modules were placed in a 2 × 3 matrix to monitor pollution at different street elevations.

The Figure 7.1 taken from Google Maps, is the primary example of how the sensors were placed into the 2 ×3 matrix to obtain results based on the research objectives. Different LoRa modules will be placed at different elevations of the street, where the distance between the LoRa modules is 0.5m, 1m, and 1.5m.

Figure 7.1 Network Design

The final testing was successful as it was assumed to have different pollution levels to varying street heights or poles. Both Pole 1 and 2 with sensor units showed fluctuations between the results obtained, and it can be said that different elevations may have different pollutant levels.

Monitoring PM_4, in addition to $PM_{2.5}$ and PM_{10}, highlights the importance of assessing fine particulate matter, as it can have health implications, particularly for individuals with respiratory conditions. The continuous monitoring of these particulate sizes can contribute to a better understanding of air quality and its potential impact on public health. The data collected during these tests indicated relatively good air quality in the monitored areas, specifically for $PM_{2.5}$ and PM_{10} concentrations. However, it is essential to emphasize the need for continuous monitoring and analysis to detect air quality trends and potential health impacts over time. Air quality is influenced by various factors, including location, weather conditions, and human activities, underscoring the importance of proactive measures to maintain and improve air quality for public health and well-being.

In summary, these pilot tests have demonstrated the effectiveness of LoRa-based sensor networks in monitoring air pollution levels and have provided a foundation for further research and practical applications. Continuous monitoring, data analysis, and proactive measures are crucial for addressing air quality concerns and ensuring the well-being of communities in various environments.

7.5 Future Work

This research aims to develop a low-cost and efficient IoT-based monitoring system to facilitate air pollution monitoring, visualization, and assessment of the air pollution monitoring system for street-level pollution. It explores a low-cost and portable means of monitoring and assessing street-level pollutants based on microcontrollers, sensors, and IoT technology. The proposed network was designed as a 4 × 4 matrix for deployment. Because of network complexities and server issues, the plan was changed from 4 × 4 to 2 × 3.

7.5.1 Proposed Mathematical Modelling.

For the novelty of this research, mathematical modelling can be implemented in the proposed air pollution monitoring system. For the implementation of the mathematical modelling, it requires the following components-

- Sensor Nodes: These physical devices measure air quality parameters such as $PM_{2.5}$, PM_{10}, PM_4, PM_1, CO_2, temperature, and humidity.
- LoRa Network: Sensor nodes transmit data to LoRa gateways, which forward the data to a central server or cloud platform.
- Data Collection: The central server or cloud platform (The Things Network) collects data from multiple sensor nodes distributed across a region.
- Air Quality Index (AQI) Calculation: Calculate the AQI for each pollutant ($PM_{2.5}$, PM_{10}, PM_4, PM_1, CO_2, temperature, humidity using the relevant AQI formulae provided by environmental agencies.

Air Quality Health Index (AQHI) Calculation: AQHI is typically calculated based on AQI values for $PM_{2.5}$, PM_{10}, PM_4, PM_1, CO_2, temperature, and humidity. It needs to use region-specific AQHI calculation models, which may include weighting factors, health impact functions, and threshold values. In New Zealand, it doesn't have AQHI standards. This will contribute to providing the AQHI information by following the AQHI Canada. Steps in the modelling shown in Figure 7.2 .

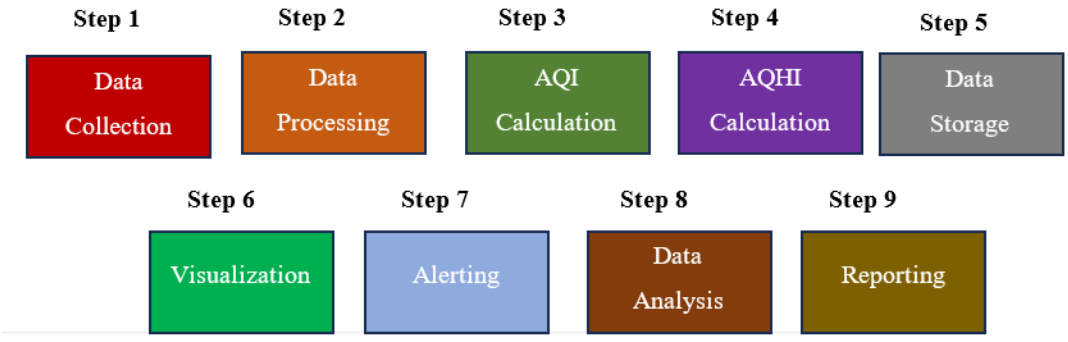


Figure 7.2 Proposed Mathematical Modelling

- **Data Collection:** Sensor nodes will be placed outside the car park to continuously collect air quality data and transmit data to the network server using the LoRa gateway.
- **Data Processing:** The network server preprocesses the data, including the pollutant levels for CO_2, PM levels, temperature and humidity, RSSI, and SNR for the proposed network.
- **AQI Calculation:** after the data is collected and processed over the network, it will be integrated with the Ubidots to use it as a graphical user interface. Data can be downloaded in .csv file. Then, calculate the AQI values for each pollutant using formulae provided by New Zealand authorities in section 1.6 in Chapter 1.
- **AQHI Calculation:** in New Zealand, the government is only calculating AQI, using those AQI values for pollutants, and using the calculated AQI values, along with meteorological and local data, to calculate the AQHI for each location. The AQHI model should consider the impact of multiple pollutants on health and apply weighting factors.
- **Data Storage:** Store the AQI and AQHI data and location and time information in a database.
- **Visualization:** Present the air quality data, AQI, and AQHI on a user-friendly dashboard or map for straightforward interpretation by stakeholders and the public.
- **Alerting:** Implement alerting mechanisms to notify relevant authorities or the public when AQI or AQHI values exceed predefined thresholds, indicating poor air quality and potential health risks.
- **Data Analysis:** Perform data analysis, trend analysis, and historical comparisons to identify patterns and trends in air quality over time.
- **Reporting:** Generate reports and summaries of air quality data, AQI, and AQHI for regulatory compliance and public awareness.

7.5.1.1 AQI for $PM_{2.5}$

The Average Air Quality Index (AQI) measures the overall air quality in a specific area over a certain period. It is typically calculated by taking measurements of various air pollutants, such as $PM_{2.5}$, PM_{10}, PM_4, PM_1, CO_2, temperature, and humidity, then converting these measurements into a single index value representing the overall air quality. For the calculation of the Air Quality Index (AQI) for $PM_{2.5}$ based on the collected data using the AQI equation for $PM_{2.5}$:

$$AQI = [(IHI - ILO) / (BPHI - BPLO)] \times (Cp - BPLO) + ILO$$

where:

- IHI = AQI value of the higher concentration category
- ILO = AQI value of the lower concentration category
- BPHI = Breakpoint concentration of the higher concentration category
- BPLO = Breakpoint concentration of the lower concentration category
- Cp = The measured concentration of $PM_{2.5}$

Table 7.1 Calculate the AQI for $PM_{2.5}$:

PM Values	AQI Calculation	AQI
0.06	[(50 - 0) / (12 - 0)] × (0.06 - 0) + 0	2.5
0.49	[(50 - 0) / (12 - 0)] × (0.49 - 0) + 0	20.42
0.74	[(50 - 0) / (12 - 0)] × (0.74 - 0) + 0	30.83
2.02	[(50 - 0) / (12 - 0)] × (2.02 - 0) + 0	84.33
2.48	[(50 - 0) / (12 - 0)] × (2.48 - 0) + 0	103.67
2.42	[(50 - 0) / (12 - 0)] × (2.42 - 0) + 0	100.83
1.33	[(50 - 0) / (12 - 0)] × (1.33 - 0) + 0	55.42
2.4	[(50 - 0) / (12 - 0)] × (2.4 - 0) + 0	100
0.74	[(50 - 0) / (12 - 0)] × (0.74 - 0) + 0	30.83
0.64	[(50 - 0) / (12 - 0)] × (0.64 - 0) + 0	26.67
1.92	[(50 - 0) / (12 - 0)] × (1.92 - 0) + 0	80
0.53	[(50 - 0) / (12 - 0)] × (0.53 - 0) + 0	22.08
2.53	[(50 - 0) / (12 - 0)] × (2.53 - 0) + 0	105.25
2.39	[(50 - 0) / (12 - 0)] × (2.39 - 0) + 0	99.58
1.9	[(50 - 0) / (12 - 0)] × (1.9 - 0) + 0	79.17
1.86	[(50 - 0) / (12 - 0)] × (1.86 - 0) + 0	77.5
0.57	[(50 - 0) / (12 - 0)] × (0.57 - 0) + 0	23.75
2.23	[(50 - 0) / (12 - 0)] × (2.23 - 0) + 0	92.92
1.56	[(50 - 0) / (12 - 0)] × (1.56 - 0) + 0	65
1.25	[(50 - 0) / (12 - 0)] × (1.25 - 0) + 0	52.08
1.21	[(50 - 0) / (12 - 0)] × (1.21 - 0) + 0	50.42
1.49	[(50 - 0) / (12 - 0)] × (1.49 - 0) + 0	62.08
1.67	[(50 - 0) / (12 - 0)] × (1.67 - 0) + 0	69.58
Total AQI for $PM_{2.5}$		**1203.61**
Average AQI	**1203.61/23**	**52.36 approx.**

The approximate average AQI for the one-day readings taken in parking at the University is 52.36. This value represents the overall air quality during that specific period, with lower values indicating better air quality and higher values indicating worse air quality. An AQI of 52.36 suggests a moderate level of air pollution on average for one day.

The average AQI value of 52.36 provides information about the air quality during the 10 hours. AQI values typically fall into different categories, such as:

Table 7.2 Different Categories of AQI

Range	Quality	Comments
0-50	Good	Air quality is satisfactory and poses little or no risk to health
51-100	Moderate	Air quality is acceptable; however, there may be a concern for some people who are unusually sensitive to air pollution.
101-150	Unhealthy for Sensitive Groups	Members of sensitive groups may experience health effects, but the public is unlikely to be affected.
151-200	Unhealthy	Everyone may begin to experience health effects; members of sensitive groups may experience more serious health effects.
201-300	Very Unhealthy	Health alert: everyone may experience more serious health effects
301-500	Hazardous	Health warning of emergency conditions; the entire population is likely to be affected

With an average AQI of 52.36, the air quality for the 24 hours falls into the "Moderate" category. AQI values can fluctuate daily based on weather conditions, pollutant sources, and other factors. It's essential to monitor air quality regularly, especially if someone has respiratory issues or is in a region prone to air pollution, to ensure you are aware of any potential health risks.

7.5.1.2 Air Quality Health Indexing (AQHI)

The Air Quality Health Index (AQHI) is a scale used to communicate the potential health risks associated with air pollution to the public [123]. It provides a simple and understandable way to convey the impact of air quality on health. AQHI values are typically reported on a scale from 1 to 10, with corresponding health risk categories.

The AQHI is typically calculated based on the concentrations of specific air pollutants, such as $PM_{2.5}$, PM_{10}, PM_4, PM_1, CO_2, temperature, and humidity. These pollutants are measured in real-time by air quality monitoring stations. The AQHI provides valuable information to the

public, helping individuals make informed decisions about outdoor activities, especially when air quality deteriorates due to industrial emissions, traffic pollution, wildfires, or other environmental events. It is an essential tool for public health authorities to issue advisories and warnings during episodes of poor air quality to reduce health risks associated with air pollution.

The Air Quality Health Index (AQHI) for $PM_{2.5}$ can be calculated by considering its concentration of PM2.5 and associated health risks. The AQHI formula for $PM_{2.5}$ is not standardized universally and can vary by region and organization. However, a general approach can be used to estimate the AQHI for $PM_{2.5}$. To calculate the AQHI for PM levels, it has the different ranges as given below:

- $PM_{2.5}$ ($\mu g/m^3$) $\leq$ 12.0: AQHI Category 1 (Low Risk)
- 12.1 $\leq$ $PM_{2.5}$ ($\mu g/m^3$) $\leq$ 24.0: AQHI Category 2 (Moderate Risk)
- 24.1 $\leq$ $PM_{2.5}$ ($\mu g/m^3$) $\leq$ 47.0: AQHI Category 3 (High Risk)
- $PM_{2.5}$ ($\mu g/m^3$) > 47.0: AQHI Category 4 (Very High Risk)

AQHI based on the collected data for $PM_{2.5}$ is given in Table 7.3–

Table 7.3 AQHI Calculation for $PM_{2.5}$

PM values (µg/m³)	**Category**	**Meaning**
0.06	1	Low Risk
0.49	1	Low Risk
0.74	1	Low Risk
2.02	2	Moderate Risk
2.48	2	Moderate Risk
2.42	2	Moderate Risk
1.33	1	Low Risk
2.4	2	Moderate Risk
0.74	1	Low Risk
0.64	1	Low Risk
1.92	2	Moderate Risk
0.53	1	Low Risk
2.53	2	Moderate Risk
2.39	2	Moderate Risk
1.9	2	Moderate Risk
1.86	2	Moderate Risk
0.57	1	Low Risk
2.23	2	Moderate Risk
1.56	2	Moderate Risk
1.25	2	Moderate Risk
1.21	2	Moderate Risk
1.49	2	Moderate Risk
1.67	2	Moderate Risk
Average AQHI	**45 / 23 ≈ 1.96**	

Based on the data calculated, the average AQHI for PM2.5 is approximately 1.96. This is a simplified estimate, and specific AQHI calculations may vary by region and organization. Some recommendations for future work are as follows:

- **Further Data Analysis:** Conduct further analysis of the pollution data. This could involve more in-depth statistical analysis, data modelling or identifying trends and patterns in the pollution data collected. This analysis can provide valuable insights into pollution sources and dynamics.
- **Sensor Longevity Testing:** Perform experiments to test the consistency and longevity of the sensors over an extended period. This is crucial for ensuring the reliability of the monitoring system over time.
- **Energy Consumption Optimization:** Research methods to optimize energy consumption in the monitoring system. This could involve using low-power sensors, sleep modes, or energy-efficient communication protocols to extend the sensor node's battery life.
- **Professional Subscription:** Consider obtaining a professional subscription from the Things Stack, which can provide more advanced features and support for your IoT network. This subscription can help with data collection, storage, and management and integrate with platforms like Ubidots for data visualization and analysis.

Overall, the research has the potential to contribute to addressing air pollution issues at the street level. Implementing these recommendations can enhance the reliability and effectiveness of the IoT-based air pollution monitoring system and contribute valuable insights to the field of environmental monitoring.

Appendix A ns-3 Coding Explanation

The code shown in Figure A.1 stores the data or elements using unordered_map. Unordered_map is an associated container that stores elements formed by combining key and mapped values. The key value uniquely identifies the element, and the mapped value is the content related to the key. Both key and value can be of any type, pre-defined or user defined. Internally, unordered_map is implemented using a Hash Table. The key provided to map is hashed into indices of the hash table, which is why the performance of the data structure depends on the hash function a lot, but on average, the cost of searching, inserting and deleting from the hash table is 0 (1).

```
class Hasher_brecht {
public:
    size_t operator() (const Address& key) const {     // the parameter type should be the same as the type of key of unordered_map
        uint64_t hash = 0;
                uint8_t buffer[8];
                memset(buffer,0,8);
                key.CopyAllTo(buffer,key.GetLength()+2);
        for(uint32_t i = 0; i < 8; i++) {
            hash += ((uint64_t) buffer[i]) << i;
        }
        return hash;
    }
};
```

Figure A.1 ns3 code for Data Storage.

The code shown in Figure A.2 is used to declare the input parameters, for example, the number of LoRa nodes, dummy nodes, gateways, servers, etc.

Here, it can be enabled true or false for some parameters or defined values in a given list below:

- Ack -> packets are transmitted reliably and receive an ACK request.
- RsLoRa -> to enable the gateway inside the network.
- Learning -> to install the network application to send and receive data.
- Optimized -> If optimized is true, then it will work based on and without power consumption.
- Interference -> to randomly create additional nodes inside the network to create interference inside the network.
- Length -> is input for interference to make a node for mobility direction.
- iterationCount -> to give how many times this simulation is to run.

- Pktsize -> packet size for data transmission.
- duration -> Duration of the simulation.
- measurementStart = 0 -> Start of the measured simulation.
- interval -> interval between packets, minutes.
- verbose -> enable logging (different from trace).printf statements.
- nakagami -> enable nakagami path loss.
- dynamic -> enable random mobility of the nodes (it is not needed for our work).
- nLoRanodes -> total LoRa nodes.
- ndummynodes -> total dummy nodes.
- nSensors -> total nodes inside the network.
- nGateways ->total gateways.
- csma -> to enable csma connection between gateway and server.
- spreading factor -> spreading factor input.
- Ptr<OutputStreamWrapper> m_stream → It is to create a txt file to store any outputs.

```
////////////////////////////////
// Configuration
////////////////////////////////
//std::vector <int, double> marked_list;
bool ack = true;
bool rslora = true;
bool learning = true;
bool optimized = true;
bool monitorEnergy = true;
bool interference = false;
bool randomSend = false;
double length = 1000;          //!< Square city with length as distance
double iterationCount = 30;        //!< Square city with length as distance
int pktsize = 1024;             //!< size of packets, in bytes
int duration = 1000;    //!< Duration of the simulation
int measurementStart = 0;   //!< Start of the measured simulation (gives time to settle)
double interval = 1;        // interval between packets, minutes

bool verbose = true;          // enable logging (different from trace)
bool nakagami = true;         // enable nakagami path loss
bool dynamic = false;          // enable random moving of pan node
uint32_t nloranodes = 4;
uint32_t ndummynodes = 12;
uint32_t nSensors = ndummynodes+nloranodes; // numbenir of sent packetss
uint32_t nGateways = 1; // numbenir of sent packets
bool csma = true;
double spreadingfactor = 1; //by varying spreading factor we plot graph for time in air ..with help of interval byvarying from 10,20 and 30
//uint32_t reportingInterval = 0; // numbenir of sent packets
Ptr<OutputStreamWrapper> m_stream = 0;
//std::stringstream filename;
```

Figure A.2 ns3 code for Declaring Input Parameters.

The code shown in Figure A.3 is used to create random variables as shown below:

- Randt ->creating a random variable. If needed.
- Unordered_map→to store the data as explained for class hasher_brecht

- Offset->for a gateway and LoRa device to set a beacon period. The timings of ping slots are then equally spaced in the broadcast period, starting from the offset.

```
Ptr<UniformRandomVariable> randT = CreateObject<UniformRandomVariable> ();
std::unordered_map<Address, Ptr<NetDevice>, Hasher_brecht> deviceMap;
//errormap: transmitted, received, received unique, received original, xlocation, ylocation
std::unordered_map<Address, std::tuple<uint32_t,uint32_t,uint32_t,uint32_t,uint32_t,uint32_t>, Hasher_brecht> errorMap;
Mac32Address server;
NetDeviceContainer gateways;
uint8_t offsets [7] = {2,3,3,1,2,2,3};


////////////////////////////////
// End configuration
////////////////////////////////
```

Figure A.3 ns3 code for Creating Random variables.

The code shown in Figure A.4 is used to transmit the packet to the nodes...when the current simulator time > 0, then it will start transmitting the packet.

```
/// Save that the message has been transmitted
    void
Transmitted (const Ptr<const Packet> packet)
{
    if (Simulator::Now().GetSeconds() > measurementStart)
    {
        Ptr<Packet> copy = packet->Copy();
        LoRaMacHeader header;
        copy->RemoveHeader(header);
        Address addr = (header.GetAddr());
        //std::tuple<uint32_t,uint32_t,uint32_t,uint32_t,uint32_t,uint32_t> tuple = errorMap[addr];
        std::get<0>(errorMap[addr])++;// = std::make_tuple(++std::get<0>(tuple),std::get<1>(tuple),std::get<2>(tuple),std::get<3>(tuple),std::get<4>(tuple));
    }
}
```

Figure A.4 ns3 code for Transmitting Packets to nodes.

The code shown in Figure A.5 is to receive the packet to the gateway, which is transmitted between dummy and LoRa nodes, and it will transmit to the gateway, which is closest to the sending LoRa nodes.

```
// save that a message has been received
    void
Received (const Ptr<const Packet> packet)
{
    if (Simulator::Now().GetSeconds() > measurementStart)
    {
    Ptr<Packet> copy = packet->Copy();
    LoRaMacHeader header;
    copy->RemoveHeader(header);
    GwTrailer trailer;
    copy->RemoveTrailer (trailer);
    Address addr = (header.GetAddr());
    std::get<1>(errorMap[addr])++;
    if ( trailer.GetGateway () == GetClosestGateway (deviceMap[addr]->GetNode ()->GetObject<MobilityModel>()))
        std::get<3>(errorMap[addr])++;
    }
}
```

Figure A.5 ns3 code for Receiving Packets.

The code shown in Figure A.6 is to get the time on air value based on the spreading factor and store it inside the results log file. Varying the spreading factor will take time on air values and will be plotted as a graph at the beginning.

```
void PrintStatistics ()
{
std::remove("results.log");
 double sf = 0;


        Ptr<Node> node = NodeList::GetNode (0);
        sf = node->GetObject <Node> ()->timeonair(spreadingfactor,interval);
        std::string LogfileName_RI = "results.log";
        std::ofstream outri (LogfileName_RI.c_str (),std::ios::app);
        outri<< "Time on Air =" << sf/100 << "\n";
        outri.close ();



}
```

Figure A.6 ns3 code for Spreading factor.

The code shown in Figure A.7 prints the statement as a received packet from sensors when a packet has been received.

```
void
ReceivePacket (Ptr<Socket> socket)
{
    //NS_LOG_UNCOND ("Sink Receiving packets from sensors!");

}
```

Figure A.7 ns3 code for Print Statement

The code shown in Figure A.8 shows the commented code that has not been used in the main code.

```
/*static void StopApplication (Ptr<Socket> socket)
    {
        socket->Close ();

    }*/

//////////////////////////////
// THIS IS THE ACTUAL CODE //
//////////////////////////////
```

Figure A.8 ns3 code for Commented code.

The code used for creating the main function is given in Figure A.9.
Randt -> is a random variable. Here max values have been set whenever needed.
The checksum is unnecessary for this LoRa Network or in the network coding.
Verbose is to print the printf statements given in LoRanetdevice.cc file and whatever files are required, which can be used. In this network code, it is not important to print all the statements, and it is disabled.

```
int
mainBody ()
{
    randT->SetAttribute("Max",DoubleValue(interval));
    //Enable checksum (FCS)
    GlobalValue::Bind ("ChecksumEnabled", BooleanValue (true));

    // Logging
    if (verbose)
    {
        //LogComponentEnable ("LoRaErrorModel", LOG_LEVEL_ALL);
// LogComponentEnable ("LoRaNetDevice", LOG_LEVEL_ALL);


    }
```

Figure A.9 ns3 code for Main Function Code

The code shown in Figure A.10 explains how to create a single LoRa server, LoRa gateways and total nodes as LoRa nodes, which contain dummy nodes and LoRa nodes - assigning this to all the nodes inside the main LoRa nodes.

```
//Create nodes
NodeContainer loraserver;
loraserver.Create (1);
NodeContainer loraGateway;
loraGateway.Create (nGateways);
NodeContainer loranodes;
loranodes.Create(ndummynodes+nloranodes);
NodeContainer loraBackendNodes(loraserver, loraGateway,loranodes);
```

Figure A.10 ns3 code for creating a single LoRa server.

The code shown in Figure A.11 is used to assign a position for the LoRa server. Figure A.12 and Figure A.13 assigning positions to nodes.

```
//Create mobility of basestations
MobilityHelper mobility;
Ptr<ListPositionAllocator> basePositionList = CreateObject<ListPositionAllocator> ();
basePositionList->Add (Vector (550.0,0.0,0.0));
mobility.SetPositionAllocator (basePositionList);
mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
mobility.Install(loraserver);
```

Figure A.11 ns3 code for Assigning Position to nodes.

```
MobilityHelper mobility2;
    Ptr<ListPositionAllocator> basePositionList2 = CreateObject<ListPositionAllocator> ();
        basePositionList2->Add (Vector (100,800,0));
        basePositionList2->Add (Vector (450,800,0));
        basePositionList2->Add (Vector (750,800,0));
        basePositionList2->Add (Vector (1050,800,0));


        basePositionList2->Add (Vector (50,1000,0));
        basePositionList2->Add (Vector (150,1000,0));
        basePositionList2->Add (Vector (250,1000,0));

        basePositionList2->Add (Vector (350,1000,0));
        basePositionList2->Add (Vector (450,1000,0));
        basePositionList2->Add (Vector (550,1000,0));

        basePositionList2->Add (Vector (650,1000,0));
        basePositionList2->Add (Vector (750,1000,0));
        basePositionList2->Add (Vector (850,1000,0));

        basePositionList2->Add (Vector (950,1000,0));
        basePositionList2->Add (Vector (1050,1000,0));
        basePositionList2->Add (Vector (1150,1000,0));
```

Figure A.12 ns3 code for the Base position for sensor nodes

```
mobility2.SetPositionAllocator (basePositionList2);
mobility2.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
mobility2.Install(loranodes);




//mobility.SetPositionAllocator (basePositionList);
//mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
//mobility.Install(loraBackendNodes);
```

Figure A.13 ns3 code for Assigning position for dummy sensor nodes

The propagation and delay models can be used inside LoRa communication to create a channel type, as shown in Figure A.14.

```
//Channel
std::cout << "Create channel" << std::endl;
SpectrumChannelHelper channelHelper;
channelHelper.SetChannel ("ns3::MultiModelSpectrumChannel");
    channelHelper.AddPropagationLoss ("ns3::RangePropagationLossModel", "MaxRange", DoubleValue(0));
channelHelper.SetPropagationDelay ("ns3::ConstantSpeedPropagationDelayModel");



Ptr<SpectrumChannel> channel = channelHelper.Create ();
LoRaHelper lorahelper;
lorahelper.SetChannel (channel);
```

Figure A.14 ns3 code for Creating Channel

The code shown in Figure A.15 is to create a gateway and assign the offset beacon period to that gateway. If rsLoRa is set as true, then it will work for the RS gateway (recommended Standard); otherwise, it will create a typical gateway.

```
// Configure gateways
std::cout << "Create gateways" << std::endl;
    if (rslora)
{
    gateways = lorahelper.InstallRsGateways (loraGateway);
    for (uint32_t i = 0; i< gateways.GetN(); i++)
        gateways.Get(i)->SetAttribute ("Offset",UintegerValue(offsets[i]-1));
}
else
    gateways = lorahelper.InstallGateways (loraGateway);
```

Figure A.15 ns3 code for Offset Beacon

The code shown in Figure A.16 assigns all nodes to the gateway nodes for communication.

```
// Create the nodes
std::cout << "Create lora nodes and dummy nodes" << std::endl;
NetDeviceContainer loraNetDevices;
if (rslora)
{
    loraNetDevices = lorahelper.InstallRs (loranodes);
    for (uint32_t i = 0; i< loraNetDevices.GetN (); i++)

        // this is a dirty workaround. Make sure to create first the network and then the gateways
        loraNetDevices.Get(i)->SetAttribute ("Offset2",UintegerValue(offsets[GetClosestGateway(loraNetDevices.Get(i)->GetNode()->GetObject<MobilityModel>())-1]-1));

}
else
    loraNetDevices = lorahelper.Install (loranodes);
```

Figure A.16 ns3 code for Assigning gateway to nodes.

If ack is set as true, it will enable reliable communication and start transmission based on the ack process shown in Figure A.17.

```
// Check if reliable
if (ack)
    for (uint32_t i = 0; i<loraNetDevices.GetN(); i++)
    {
        loraNetDevices.Get(i)->SetAttribute("Reliable",BooleanValue(true));


        Ptr<LoRaPhy> temp = StaticCast<LoRaPhy>(StaticCast<LoRaNetDevice>(loraNetDevices.Get(i))->GetPhy());
                  Simulator::ScheduleNow(&LoRaPhy::SetSpreadingFactor, temp, spreadingfactor);

    }
```

Figure A.17 ns3 code for Enable communication process.

To monitor the energy means, the energy model is assigned to all the nodes, and the initial energy of the nodes is 100 joules; this is given in Figure A.18.

```
// create energy source
if(monitorEnergy)
{
    std::cout << "Monitoring energy enabled" << std::endl;
    LoRaEnergySourceHelper sourceHelper;
    sourceHelper.Set("BasicEnergySourceInitialEnergyJ",DoubleValue(100));
    EnergySourceContainer energySources = sourceHelper.Install(loranodes);
    LoRaRadioEnergyModelHelper radioHelper;
    DeviceEnergyModelContainer deviceModels = radioHelper.Install (loraNetDevices, energySources);
}
```

Figure A.18 ns3 code for Monitoring energy.

To create a connection between gateway and server using CSMA with a data rate of 1000mbps and delay of 60sec, explain in Figure A.19 and attach this to LoRa back-end nodes, already assigned in the previous line of code. Creating an internet stack helper and assigning it to all LoRa back-end nodes containing the LoRa server, gateway, dummy and LoRa nodes.

```
// Connect gateways with network
std::cout << "Create wired network" << std::endl;
CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("1000Mbps"));
csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (60)));

std::cout << "Connect devices with wired network" << std::endl;
NetDeviceContainer csmaDevices;
csmaDevices = csma.Install (loraBackendNodes);


std::cout << "Install IP/TCP" << std::endl;
InternetStackHelper stack;
stack.Install (loraBackendNodes);
Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer interfaces = address.Assign (csmaDevices);
```

Figure A.19 ns3 code for Connection between gateway and server

The code shown in Figure A.20 sets the address for the gateway and server. If learning is optimized as true at the beginning of the code, it will enable and disable power applications and work as per that.

```
// set addresses
std::cout << "Set the addresses" << std::endl;
     lorahelper.FinishGateways (loraGateway, gateways, interfaces.GetAddress(0));
//lorahelper.FinishGateways (loraGateway, gateways, interfaces.GetAddress(5));
    //lorahelper.FinishGateways (loraGateway, gateways, interfaces.GetAddress(2));
    //lorahelper.FinishGateways (loraGateway, gateways, interfaces.GetAddress(3));
    //lorahelper.FinishGateways (loraGateway, gateways, interfaces.GetAddress(4));
// Reset the power after each succesfull message
if (learning)
{
    lorahelper.InstallNetworkApplication("ns3::LoRaSfControllerApplication");
    //Simulator::Schedule(Seconds(duration-1),&LoRaSfControllerApplication::PrintValues(),DynamicCast<LoRaSfControllerApplication>(apps.get(0)));
}
else
{
    if (optimized)
    {
        lorahelper.InstallNetworkApplication("ns3::LoRaPowerApplication");
    }
    else
    {
        lorahelper.InstallNetworkApplication("ns3::LoRaNoPowerApplication");
    }
}
Ptr<LoRaNetwork> loraNetwork = lorahelper.InstallBackend (loraserver.Get (0),loraNetDevices);
```

Figure A.20 ns3 code for Addresses for gateway and server

To get the position of all the nodes and trace the value from the transmitted function and received function coded before about the transmitted packets and received packets, if needed, is given in Figure A.21.

```
    // Let the nodes generate data
    std::cout << " Generate data from the nodes in the LoRa network" << std::endl;
    //ApplicationContainer apps = lorahelper.GenerateTraffic (randT, loranodes, pktsize, 0, duration, interval,randomSend);
//ApplicationContainer apps = lorahelper.GenerateTraffic (randT, loranodes.Get(0), pktsize, 0, 10, interval,randomSend);

    // hookup functions to the netdevices of each node to measure the performance
    for (uint32_t i = 0; i< loraNetDevices.GetN(); i++)
    {
        deviceMap[ (loraNetDevices.Get(i)->GetAddress())]=loraNetDevices.Get(i);
        uint32_t x  = loraNetDevices.Get(i)->GetNode()->GetObject<MobilityModel>()->GetPosition ().x;
        uint32_t y  = loraNetDevices.Get(i)->GetNode()->GetObject<MobilityModel>()->GetPosition ().y;
        errorMap[ (loraNetDevices.Get(i)->GetAddress())] = make_tuple (0,0,0,0,x,y);
        DynamicCast<LoRaNetDevice>(loraNetDevices.Get(i))->TraceConnectWithoutContext ("MacTx",MakeCallback(&Transmitted));
    }

    // hookup functions to the network for measuring performance
    loraNetwork->TraceConnectWithoutContext("NetRx",MakeCallback(&ReceivedUnique));
    loraNetwork->TraceConnectWithoutContext("NetPromiscRx",MakeCallback(&Received));
```

Figure A.21 ns3 code for Trace values transmitted.

If enable interference is set as true, then it will randomly create additional nodes inside the network and create interference. The code for interference is given in Figure A.22.

```
// Configure interference
if (interference)
{
    std::cout << "Interference Enabled" << std::endl;
    MobilityHelper mobilityInterference;
    Ptr<RandomDiscPositionAllocator> allocator = CreateObject<RandomDiscPositionAllocator>();
    Ptr<RandomVariableStream> radius = CreateObject<UniformRandomVariable>();
    radius->SetAttribute("Max",DoubleValue(2000));
    allocator->SetAttribute("Rho",PointerValue(radius));
    allocator->SetAttribute("X",DoubleValue(1500));
    allocator->SetAttribute("Y",DoubleValue(1500));
    Ptr<RandomVariableStream> speed = CreateObject<ConstantRandomVariable>();
    speed->SetAttribute("Constant",DoubleValue(3e9));
    Ptr<RandomVariableStream> pause = CreateObject<ConstantRandomVariable>();
    pause->SetAttribute("Constant",DoubleValue(0.5));

    mobilityInterference.SetMobilityModel ("ns3::RandomDirection2dMobilityModel","Bounds",RectangleValue(Rectangle(-length*3,length*3,-length*3,length*3)),"Speed",PointerValue(speed),"Pause",PointerValue(pause));
    mobilityInterference.SetPositionAllocator((allocator));
    lorahelper.AddInterference(mobilityInterference);
}
```

Figure A.22 ns3 code for Enable interference.

The code given in Figure A.23 is for data transmission between LoRa and dummy nodes.

```
UdpEchoServerHelper echoServer (9);
 uint32_t i = 0, j = 0;

  for (j = 0; j < loranodes.GetN(); j++)
  {
if(j== 0 || j== 1 || j== 2||j== 3 ) {
    ApplicationContainer serverApps = echoServer.Install (loranodes.Get(j));
    serverApps.Start (Seconds (0.0));
    serverApps.Stop (Seconds (5.0));
}
  }
    for (i = 0; i < loranodes.GetN(); i++)
    {
if((i== 4|| i== 5 || i== 6 || i== 7 ||i== 8 ||i== 9 ||i== 10 ||i== 11 ||i== 12 ||i== 13 ||i== 14 ||i== 15 )) {

       UdpEchoClientHelper echoClient (interfaces.GetAddress (i), 9);
       echoClient.SetAttribute ("MaxPackets", UintegerValue (100000000));
       echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
       echoClient.SetAttribute ("PacketSize", UintegerValue (pktsize));

       ApplicationContainer clientApps = echoClient.Install (loranodes.Get(i));
       clientApps.Start (Seconds (0.0));
       clientApps.Stop (Seconds (duration-1));
}
    }
```

Figure A.23 ns3 code for Data transmission between LoRa and dummy nodes

The code shown in Figure A.24 shows the communication between LoRa and gateway nodes.

```
UdpEchoServerHelper echoServer1 (9);

    ApplicationContainer serverApps1 = echoServer1.Install (loraGateway.Get (0));
    serverApps1.Start (Seconds (5.0));
    serverApps1.Stop (Seconds (10.0));
    for (i = 0; i < loranodes.GetN(); i++)
    {
 if(i== 0 || i== 1 || i== 2||i== 3 ) {

       UdpEchoClientHelper echoClient1 (interfaces.GetAddress (i), 9);
       echoClient1.SetAttribute ("MaxPackets", UintegerValue (100000000));
       echoClient1.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
       echoClient1.SetAttribute ("PacketSize", UintegerValue (pktsize));

       ApplicationContainer clientApps1 = echoClient1.Install (loranodes.Get(i));
       clientApps1.Start (Seconds (5.0));
       clientApps1.Stop (Seconds (duration-1));
}
    }
```

Figure A.24 ns3 code for Communication between LoRa and nodes

The code shown in Figure A.25 shows the communication between LoRa and server nodes, and Figure A.26 shows the communication between the server and LoRa nodes.

```
UdpEchoServerHelper echoServer2 (9);

    ApplicationContainer serverApps2 = echoServer2.Install (loraserver.Get (0));
    serverApps2.Start (Seconds (10.0));
    serverApps2.Stop (Seconds (duration-1));

for (i = 0; i < loranodes.GetN(); i++)
    {
    UdpEchoClientHelper echoClient2 (interfaces.GetAddress (i), 9);
        echoClient2.SetAttribute ("MaxPackets", UintegerValue (100000000));
        echoClient2.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
        echoClient2.SetAttribute ("PacketSize", UintegerValue (pktsize));

        ApplicationContainer clientApps2 = echoClient2.Install (loraGateway.Get (0));
        clientApps2.Start (Seconds (10.0));
        clientApps2.Stop (Seconds (duration-1));
}
```

Figure A.25 ns3 code for Gateway to server

```
UdpEchoServerHelper echoServer3 (9);

    ApplicationContainer serverApps3 = echoServer3.Install (loraGateway.Get (0));
    serverApps3.Start (Seconds (20.0));
    serverApps3.Stop (Seconds (duration-1));

for (i = 0; i < loranodes.GetN(); i++)
    {
    UdpEchoClientHelper echoClient3 (interfaces.GetAddress (i), 9);
        echoClient3.SetAttribute ("MaxPackets", UintegerValue (100000000));
        echoClient3.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
        echoClient3.SetAttribute ("PacketSize", UintegerValue (pktsize));

        ApplicationContainer clientApps3 = echoClient3.Install (loraserver.Get (0));
        clientApps3.Start (Seconds (20.0));
        clientApps3.Stop (Seconds (duration-1));
}
```

Figure A.26 ns3 code for Server to LoRa Nodes

The code in Figure A.27 contains the code for creating a netanim xml file, giving node size in netanim, and assigning node description for gateway and server, highlighting the node as server

and gateway in animation. In ns-3, a flow monitor agent is available to calculate parameters like PDR, delay and throughput. The network is using this agent to calculate our parameters.

```
    // Start the simulation
    std::cout << "start the fun" << std::endl;
AnimationInterface anim ("output.xml"); // Mandatory
for (uint32_t i = 0; i < loranodes.GetN()+2; i++)
  {
anim.UpdateNodeSize(i,50,50);


}


for (uint32_t i = 0; i < loraGateway.GetN(); i++)
  {

 anim.UpdateNodeDescription (loraGateway.Get(i), "GW");
  anim.UpdateNodeColor (loraGateway.Get(i), 0, 0, 255);

}

for (uint32_t i = 0; i < loraserver.GetN(); i++)
  {

  anim.UpdateNodeDescription (loraserver.Get(i), "SERVER");
  anim.UpdateNodeColor (loraserver.Get(i), 255, 0, 255);
}

  FlowMonitorHelper flowmon;
  Ptr<FlowMonitor> monitor = flowmon.InstallAll();

    Simulator::Stop (Seconds (duration));
         Simulator::Schedule (Seconds (duration-1) ,&PrintStatistics);

    Simulator::Run ();
```

Figure A.27 ns3 code for Generating NetAnim file.

The simulation will be stopped, and the simulation process will start again, as shown in Figure A.28.

```
monitor->CheckForLostPackets ();
Ptr<Ipv4FlowClassifier> classifier = DynamicCast<Ipv4FlowClassifier>
          (flowmon.GetClassifier ());
    std::map<FlowId, FlowMonitor::FlowStats> stats = monitor->GetFlowStats ();

    uint32_t txPacketsum = 0;
    uint32_t rxPacketsum = 0;
    uint32_t rxBytesum = 0;
    uint32_t DropPacketsum = 0;
    uint32_t LostPacketsum = 0;
    double Delaysum = 0;

for (std::map<FlowId, FlowMonitor::FlowStats>::const_iterator i =
        stats.begin (); i != stats.end (); ++i)
    {
        Ipv4FlowClassifier::FiveTuple t = classifier->FindFlow (i->first);
        std::cout << "Flow " << i->first << " (" << t.sourceAddress << " -> " << t.destinationAddress << ")\n";

        txPacketsum += i->second.txPackets;
        rxPacketsum += i->second.rxPackets;
        rxBytesum += i->second.rxBytes;
        LostPacketsum += i->second.lostPackets;
        DropPacketsum += i->second.packetsDropped.size();
        Delaysum += i->second.delaySum.GetSeconds();


    }
```

Figure A.28 ns3 code for Simulation Process

The code shown in Figure A.29 is to calculate PDR, delay, and throughput for the simulation process as below:

- PDR -> (number of packets received*100/number of packets sent), 100 is used here to convert the value into a percentage between 0 and 100.
- Delay -> (last packet transmitted time /number of packets received), Last packet transmitted time is stored in delaysum variable, by using this calculate average delay.
- Throughput -> (number of received bytes*8) / (data endtime-data starttime), divide by 1024 to convert bits per second to kbps.

```
    std::cout << "************************************************************************ ******" << "\n";
    std::cout << "  All Tx Packets: " << txPacketsum <<"                 " << "  All Rx Packets: " << rxPacketsum  << "\n";
    std::cout << "  All Lost Packets: " << LostPacketsum <<"               " << "  All Drop Packets: " << DropPacketsum << "\n";
    std::cout << "************************************************************************ ******" << "\n";
    std::cout << "  Packets Delivery Ratio: " << (((rxPacketsum) * 100) /txPacketsum) << "%"  << "\n";
    std::cout << "  All Delay: " << Delaysum / (rxPacketsum) << "\n";
    std::cout << "  Throughput: " << (rxBytesum * 8.0) / (duration-0)/1024 << " Kbps\n";



    return 0;
}
```

Figure A.29 ns3 code for Calculation of performance metrics

Appendix B Schematic Diagrams

(a) Main Control Unit

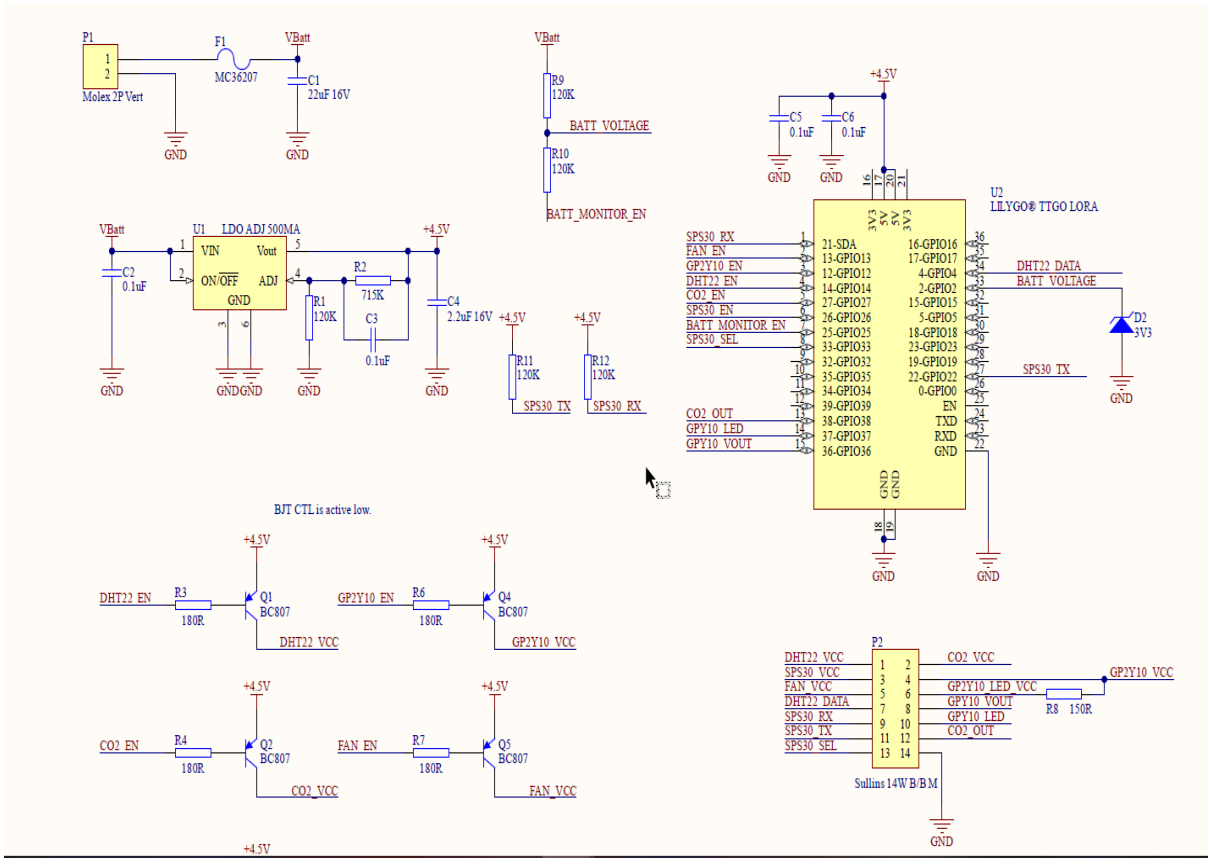


Figure B.1 Schematic Layout of ESP32 Board

(b) Connector pins

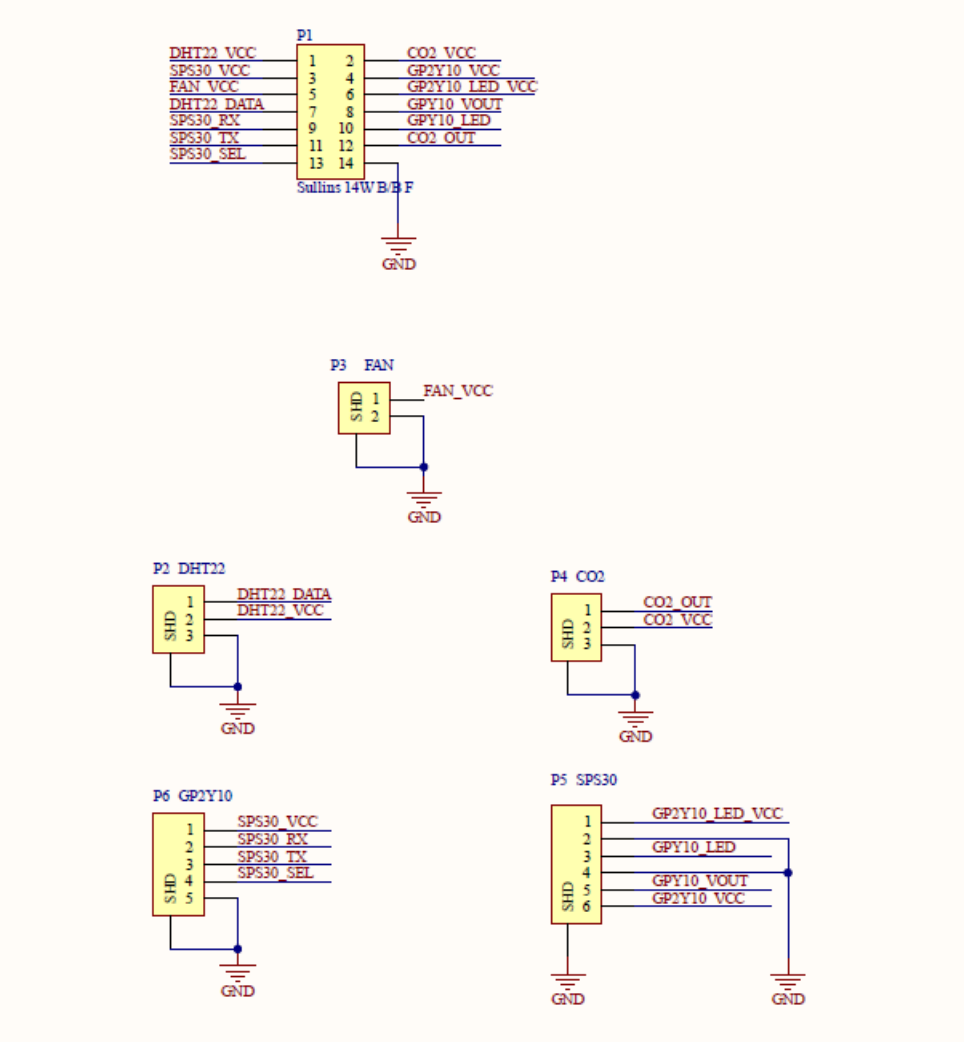


Figure B.2 Schematic for connector pins

Appendix C Sensors – Technical Information

This appendix covers the sensors used for the research activities. Different sensors and their specifications are given in below sections:

(a) Dust Sensor

GP2Y1010AU0F is a dust sensor by an optical sensing system. An infrared emitting diode (IRED) and a phototransistor are diagonally arranged into this device. It effectively detects very fine particles like cigarette smoke. The working of the dust sensor is simple. When power is provided to the given sensor, the LED emits the light, which will fall on the Lens, and the Detector will sense that light, which is arranged diagonally opposite the LED. When dustis present, the output voltage is measured by the analogue pin of the sensor. The dust sensor is a low-power device operated in 3.3V-5V, shown in Figure C.1.

Figure C.1 Dust sensors

Features:

- Compact, thin package
- Consumes less current.
- The photometry of only one pulse can detect the presence of dust.
- Enable to distinguish smoke from house dust.
- Lead-free.

This dust sensor, "GP2Y1010AU0F", is a device that detects house dust, cigarette smoke, etc. and is designed as a sensor for the automatic running of applications like air purifiers and air conditioners with air purifier functions. The circuit diagram of the sensor is shown in Figure C.2, different parameters are shown in Figure C.3, and the sampling time pulse is shown in Figure C.4.

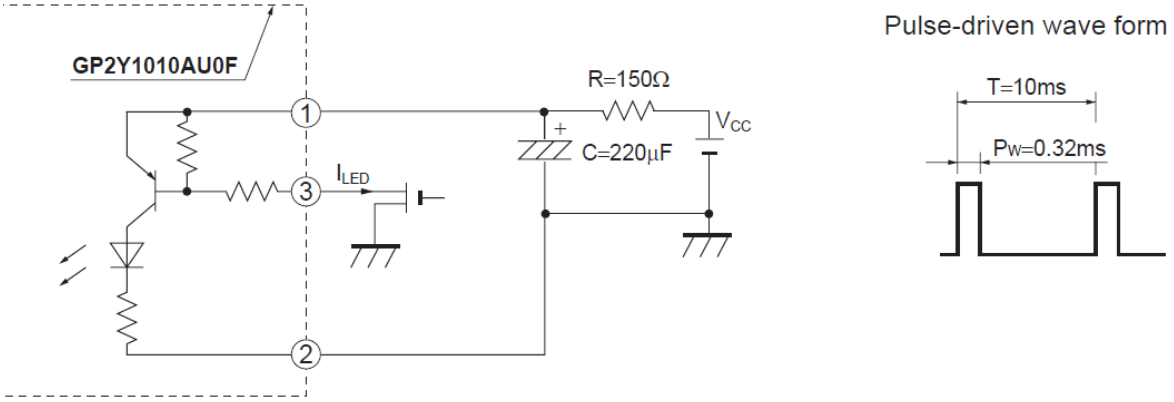


Figure C.2 Circuit Diagram

Parameter	Symbol	Specified condition	Recommended condition	Unit
Pulse cycle	T	10	10±1	ms
Pulse width	Pw	0.32	0.32±0.02	ms

Figure C.3 Different Parameters

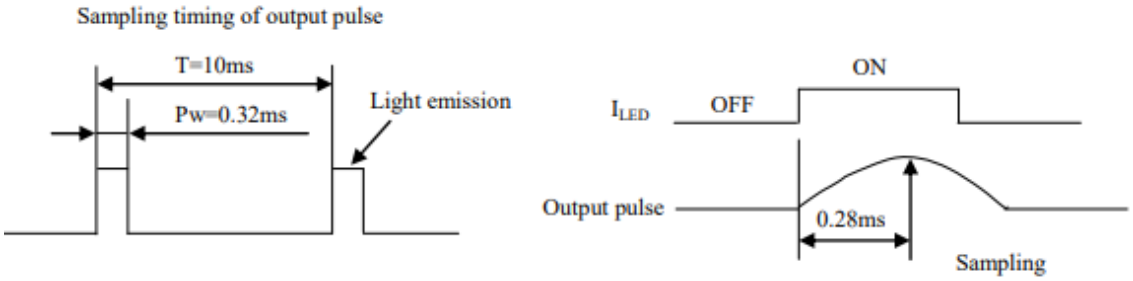


Figure C.4 Sampling Time Pulse

The resister, R1 = 150Ω and capacitor, C1 = 220uF, mentioned above, are required for the pulse drive of the LED of GP2Y1010AU0F. Without these components, the device does not work.

As input conditions of the LED terminal, please apply LED drive conditions mentioned in the Electro-optical characteristics chart of the specification. When it is impossible to apply those conditions, please make it within the recommended input conditions mentioned in the specification. The device's characteristics will be affected when the LED is driven under a condition beyond the specification.

- The LED emits pulse light. The amplifier circuit amplifies the detected signal and goes out as the output synchronized to the pulse emission of the LED.
- The specified output value is the one that is measured at 0.28 ms after the LED is turned on. Therefore, it is recommended that the microcomputer also read the output 0.28ms after the LED emission.
- The time required for the device to be ready to detect dust from when the system is turned on is less than 1 sec.

To develop the LoRa end device shown in Figure C.5, an interface must be built between the LoRa module and the Arduino board. It requires the particular library used for this, the LMIC library. The LMIC library has the total Class A and B implementation packages to support AU915 frequencies.

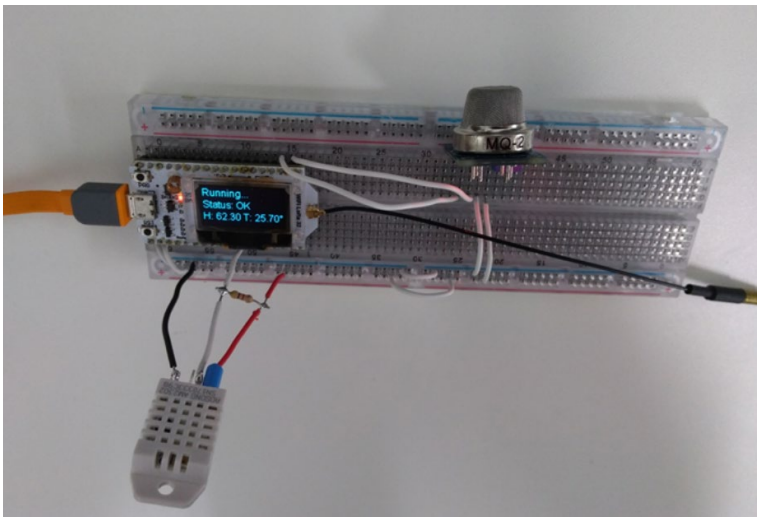


Figure C.5 LoRa Node

Code explanation

To define the dust sensor, use the line below in the coding.

#define USE_DUST 0

Define the enable pins for the dust sensor and LED of the dust sensor.

#define DUST_EN_PIN 0

#define DUST_LED_EN_PIN 15

The below code is used for the TTN configuration, with the help of the sensor, which will transmit data to the network server. The communication process requires network key, apps key and device address.

```
// TTN Configuration.
static const PROGMEM u1_t NWKSKEY [16] = {0xD2, 0x00, 0xC1, 0xBB, 0x4A, 0x1C, 0xCC, 0x48, 0xDC, 0x3A, 0xF9, 0xAC, 0x48, 0xD9, 0xE1, 0x93};
// LoRaWAN AppSKey, application session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const u1_t PROGMEM APPSKEY [16] = {0x65, 0xA6, 0x64, 0x3F, 0x16, 0x36, 0xC6, 0x3C, 0xAF, 0xFF, 0xE9, 0x97, 0x3B, 0x40, 0x31, 0x95};
// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR =0x260C51F8;
```

The pin numbers below are defined for measuring the sensor data.

```
#if USE_DUST
int measurePin = 36; //Connect dust sensor to Arduino A0 pin
int ledPower = 15;   //Connect 3 led driver pins of dust sensor to Arduino D2
int samplingTime = 280;
int deltaTime = 40;
int sleepTime = 9680;
float voMeasured = 0;
float calcVoltage = 0;
float dustDensity = 0;
```

The below code starts the sensor data calculation.

```
/* Dust sensor calculation */
void get_dust_value()
{
 pinMode(DUST_EN_PIN,OUTPUT); // CO2 EN
 pinMode(ledPower,OUTPUT);
 digitalWrite(DUST_EN_PIN,HIGH);
 while(1)
 {
 digitalWrite(ledPower,LOW); // power on the LED
  delayMicroseconds(samplingTime);

  voMeasured = analogRead(measurePin); // read the dust value
  delayMicroseconds(deltaTime);
  digitalWrite(ledPower,HIGH); // turn the LED off

  delayMicroseconds(sleepTime);

  // 0 - 5V mapped to 0 - 1023 integer values
  // recover voltage
  calcVoltage = voMeasured * (5.0 / 4095.0);
  if (calcVoltage > 0)
  {

  // linear eqaution taken from http://www.howmuchsnow.com/arduino/airquality/
  // Chris Nafis (c) 2012
  dustDensity = 0.17 * calcVoltage - 0.1;
  }
  else
  {
   dustDensity = 0;
  }
  Serial.print("DustDensity ");
  Serial.print(dustDensity);
```

```
    Serial.print("\n\r");
    dustDensity =  calcVoltage/100;
    delay(1000);
  }

}
#endif
```

To display the data on a serial monitor, the below code is used-

```
#if USE_DUST
  display.setCursor(50,0);
  display.print("dust: " + String(dustDensity) + String(" ug"));
#endif
```

The code below used the high and low payload for dust values to get the dust density value.

```
#if USE_DUST
      /* get dust density value */
      get_dust_value();
      uint16_t payloadDust = LMIC_f2sflt16(dustDensity);
      byte dustLow = lowByte(payloadDust);
      byte dustHigh = highByte(payloadDust);
      payload[4] = dustLow;
      payload[5] = dustHigh;
#endif

  pinMode(DUST_EN_PIN,OUTPUT);      // DUST EN
  pinMode(DUST_LED_EN_PIN,OUTPUT);  // DUST LED
```

(b) Particulate Matter Sensor

The SPS30 particulate matter (PM) sensor is a technological breakthrough in optical PM sensors. The measuring principle of the PM sensor is based on laser scattering and makes Sensirion's innovative contamination resistance technology. This technology is high quality, and long-lasting components enable precise measurements. The SPS30 particulate matter (PM) sensor is a compact, high-quality optical particle sensor that uses laser scattering and innovative contamination resistance technology to achieve particle matter measurement.

For the research and monitoring of the pollutant level, a previous dust sensor was used instead of SPS30. Due to the inaccuracies and only the dust level monitoring qualities, the sensor was replaced with SPS30, as shown in Figure C.6. The sensor has inbuilt high-quality and long-lasting components that enable the precision of measurements and provide the throughput for ten years and the specification and characteristics of the sensor is given in Table C.1. The SPS30 sensor has a five-pin interface that can communicate with two different interfaces: UART and I2C. For the designed network to monitor the particulate matter. The sensor is working based on UART as per the TTGO LoRa32 SX1276 OLED Board requirements.

Figure C.6 SPS30 Sensor for measuring Particulate Matter

Table C.1 Specification of the PM sensor

Parameter/technologies	LoRa	Sigfox	NB-IoT
Band	868/915 MHz	868/915 MHz	868/915 MHz
PHY	CSS	UNB	NB
Spreading Factor	2^7-2^{12}	-	-
Bandwidth	500-125 kHz affiliation information	1 kHz	180 kHz
Data Rate (kbps)	27-0.37	0.1	250-226.7
Range (km)	22	63	35

SPS30 sensor works based on the laser scattering principle. With the help of a fan, it creates controlled airflow, and the explanation is according to Figure C.7.

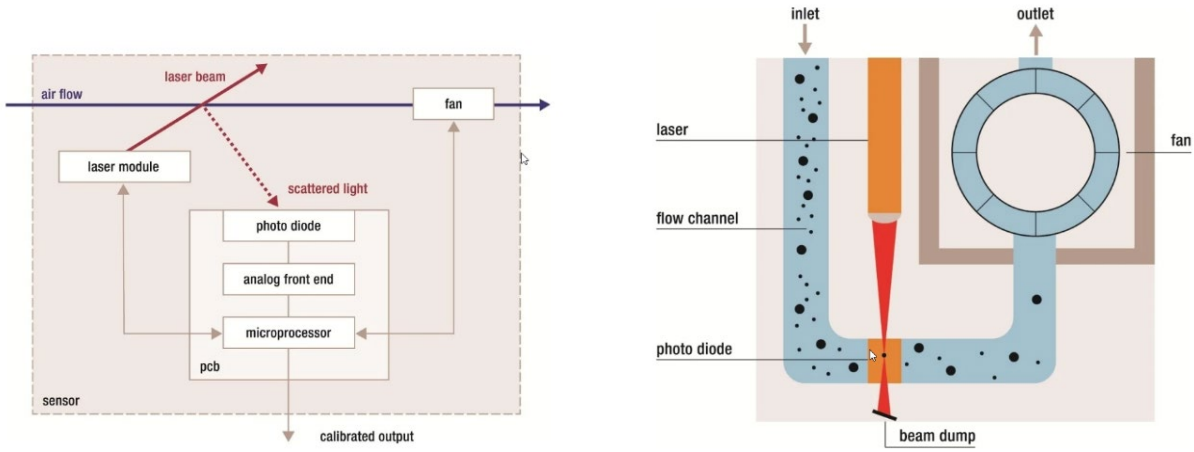


Figure C.7 SPS30 Working

The airflow loop between the fan and microprocessor stabilizes the fan speed and airflow through the sensor. PM levels travel in sensors from the inlet to the outlet carried by airflow. Particles in the airstream traverse through a focused laser beam in line with the photodiode, as marked in red, causing light scattering. The scattered light is then detected by the photodiode and converted to a mass/number concentration output via Sensirion's proprietary algorithms, which run on the SPS30 internal microcontroller.

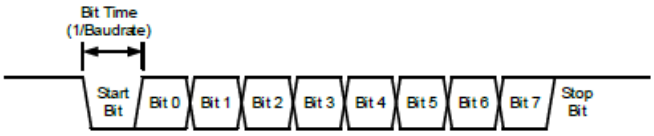
To learn about Automatic cleaning, then here is the explanation as follows:

The SPS30 sensor has an automatic fan cleaning function, which will accelerate the built-in fan to maximum speed for 10 seconds to blow out the dust accumulated inside the fan. The default automatic-cleaning interval is 168 hours (1 week) of uninterrupted use. Switching off the sensor resets this time counter. Disabling automatic cleaning or setting a manual interval is not supported now.

Data transmission

Using UART interface (we are not using this method)
SPS30 has two layers:

1. Physical layer that has separate Rx and Tx lines with unipolar logic levels and transmits 8-bit data.



2. The SHDLC layer (Sensirion's high-level data link control protocol) is easy to implement and is considered a serial communication protocol. Based on MOSI and MISO architecture (has one slave and one master architecture), the data is transferred in logical units as frames. MOSI starts sensing the frame, and the slave responds to that query.

Connection using I2C interface (using this in programming)

Max. speed for data transmission is 100 kbits/s. This connection required SCL and SDA lines for I/O pins on LoRa modules. 16-bit address pointer sets to read and write sensor data and collect measurement results. The sensor transmits data in 2-byte packets, and the checksum is

calculated. There are some I2C commands available in the datasheet that work automatically inside the sensor.

Electrical specification.

Voltage min 4.4v and Max 5.5v.
Current – Sleep mode (max. 50 µA), idle mode (min. 300 µA and max. 360 µA), measurement mode (min. 45mA and max. 65mA), fan start (max. 80mA).
The accuracy of the sensor is 10µg/m^3.
Transmission time is between 8-30 seconds.

Code Explanation

```
#include <SPI.h>
#include <sps30.h>
```

Sensor initializes after assigning pin numbers.

```
#define USE_SPS   1
#define SPS30_SEL_PIN 16
```

Time setting for the sensor unit to sleep. It can be modified according to the requirements of transmission.

```
#define TIME_TO_SLEEP (300) /* Time ESP32 will go to sleep (in seconds) */
```

To transmit data to the Things network or Things stack, it requires some settings to provide a network key and application key to identify the devices and applications.

```
// TTN Configuration.
static const PROGMEM u1_t NWKSKEY[16] = {0xD2, 0x00, 0xC1, 0xBB, 0x4A, 0x1C, 0xCC, 0x48, 0xDC, 0x3A, 0xF9, 0xAC, 0x48, 0xD9, 0xE1, 0x93};
// LoRaWAN AppSKey, application session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
```

```
static const u1_t PROGMEM APPSKEY[16] = {0x65, 0xA6, 0x64, 0x3F, 0x16, 0x36, 0xC6,
0x3C, 0xAF, 0xFF, 0xE9, 0x97, 0x3B, 0x40, 0x31, 0x95};
// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR =0x260C51F8;
```

The selection of SPS sensor pins is done in the code below, and the sps read function is created to read and print values for PM1, PM2.5, PM4, and PM_{10}.

```
#if USE_SPS
struct sps30_measurement m;
#define SPS30_SEL_PIN 16
void sps_read()
{

 char serial[SPS30_MAX_SERIAL_LEN];
 uint16_t data_ready;
 int16_t ret;
 sps32_setup();



 do {
  ret = sps30_read_data_ready(&data_ready);
  if (ret < 0) {
   Serial.print("error reading data-ready flag: ");
   Serial.println(ret);
  } else if (!data_ready)
   Serial.print("data not ready, no new measurement available\n");
  else
   break;
  delay(100); /* retry in 100ms */
 } while (1);

 ret = sps30_read_measurement(&m);
 if (ret < 0) {
```

```
    Serial.print("error reading measurement\n");
  } else {


    Serial.print("PM  1.0: ");
    Serial.println(m.mc_1p0);
    Serial.print("PM  2.5: ");
    Serial.println(m.mc_2p5);
    Serial.print("PM  4.0: ");
    Serial.println(m.mc_4p0);
    Serial.print("PM 10.0: ");
    Serial.println(m.mc_10p0);

    Serial.print("Typical partical size: ");
    Serial.println(m.typical_particle_size);
    Serial.println();

  }
}

void sps32_setup() {
  int16_t ret;
  uint8_t auto_clean_days = 4;
  uint32_t auto_clean;
pinMode(21,INPUT); // SPS30 sda
pinMode(22,INPUT); // SPS30 scl // low to ensure no leakage current into sps30 during startup.
pinMode(16,INPUT); // SPS30 SEL
pinMode(17,OUTPUT);
delay(500); // wait for sps to turn off
pinMode(16,OUTPUT); // SPS30 SEL low
delay(500); // wait for sps to turn off
pinMode(17,HIGH);
delay(1500); // wait for sps to turn off
sensirion_i2c_init();
```

```
  delay(2000);

  while (sps30_probe() != 0) {
    Serial.print("SPS sensor probing failed\n");
    delay(500);
  }

#ifndef PLOTTER_FORMAT
  Serial.print("SPS sensor probing successful\n");
#endif /* PLOTTER_FORMAT */

  ret = sps30_set_fan_auto_cleaning_interval_days(auto_clean_days);
  if (ret) {
    Serial.print("error setting the auto-clean interval: ");
    Serial.println(ret);
  }

  ret = sps30_start_measurement();
  if (ret < 0) {
    Serial.print("error starting measurement\n");
  }

  delay(1000);
}
#endif


#if USE_SPS
    sps32_setup();
```

Some test results taken during the sensors' testing are given in Figure C.8 to Figure C.11.

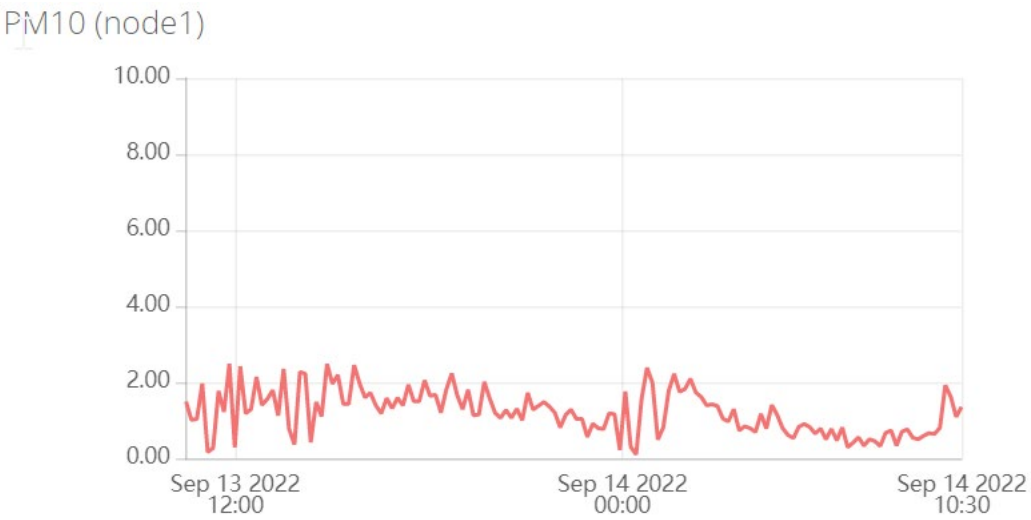


Figure C.8 PM_{10} readings

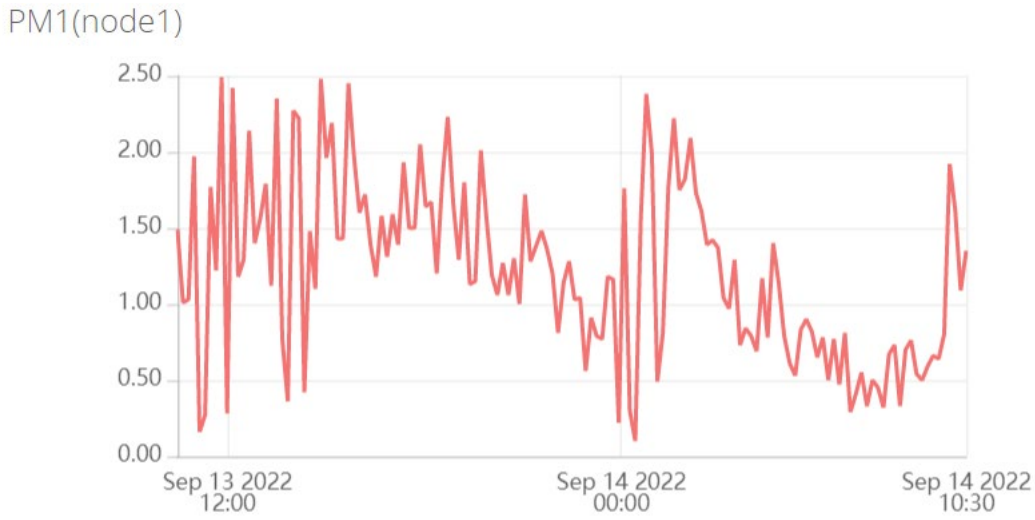


Figure C.9 PM_{1} readings

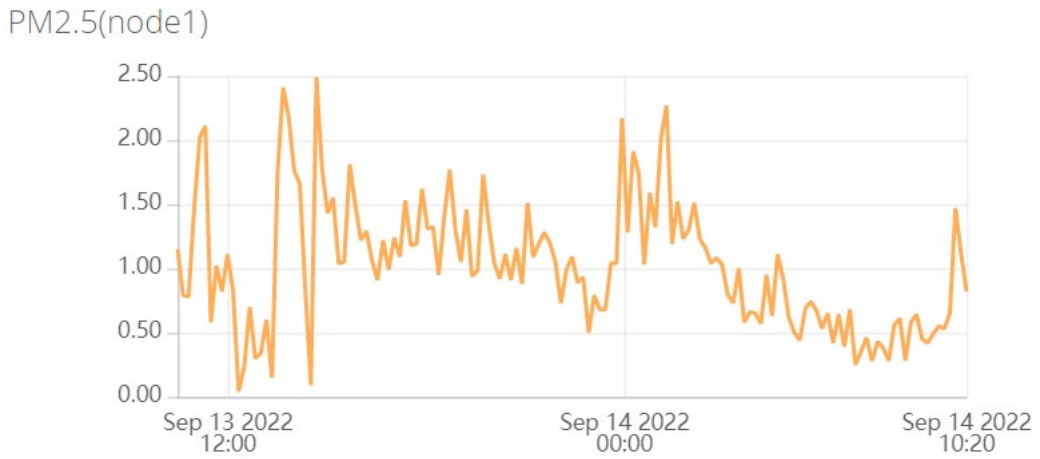


Figure C.10 $PM_{2.5}$ readings

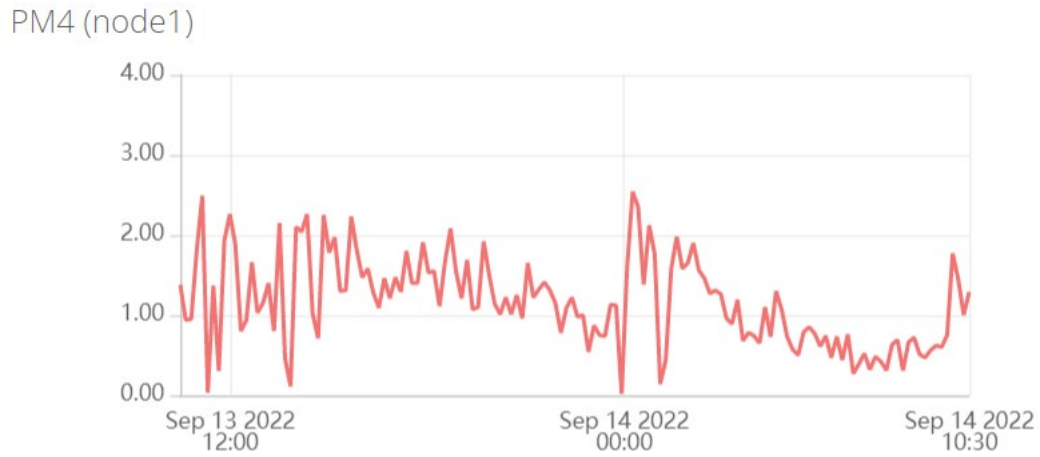


Figure C.11 PM_4 Readings

(c) Temperature and Humidity Sensor

The temperature and humidity readings werecollected using a digital-output capacitive-type relative humidity and temperature sensor/module (DHT22) shown in Figure C.12. The DHT22 sensor outputs a calibrated 8-bit digital signal. This sensor is used for the digital-signal-collecting technique and humidity sensing technology, assuring its reliability and stability. Its sensing elements relate to an 8-bit single-chip computer. The specification of the sensor is given in Table C.2. Every sensor of this model is temperature-compensated and calibrated in an accurate calibration chamber.

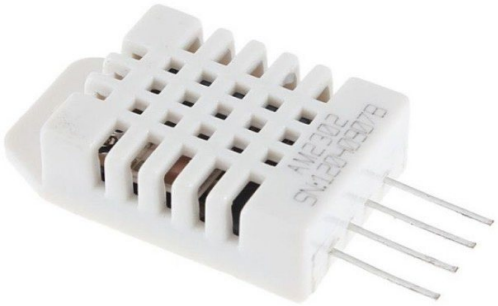
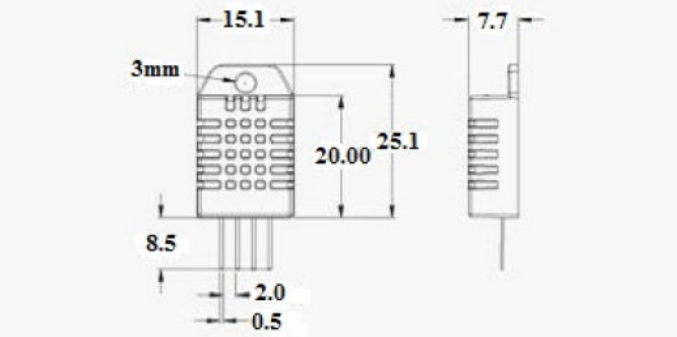


Figure C.12 DHT22 sensors

Table C.2 Technical specifications

Model	RHT03
Power Supply	3.3-6V DC
Output signal	Digital signal
Sensing element	Polymer humidity capacitor
Operating range	Humidity 0-100%RH Temperature -40-80Celsuis
Accuracy	Humidity +- 2%RH Temperature +-0.5Celsuis
Resolution	Humidity 0.1%RH Temperature 0.1Celsuis
Repeatability	Humidity +-0.1%RH Temperature +-0.2elsuis

Pins and dimensions are shown in Figure C.13, and the data and signal format of the DHT22 sensor are explained in Table C.3.



Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD—power supply
2	DATA–signal
3	NULL
4	GND

Figure C.13 Pinout and Dimensions of DHT22

Table C.3 Data and Signal format

Name	Single-bus data and signal format
Start signal	The microprocessor sets the SDA to LOW for a period of time (at least 800μs) to inform the sensor to prepare the data.
Response signal	The sensor sets the SDA to LOW for 80μs, then HIGH for 80μs, to respond to the start signal.
Data format	After receiving the start signal, the sensor reads a data string (40 bits) through SDA, with the high bit first out.
Humidity	The humidity resolution is 16 Bits, high bit first out; The value read out by the sensor is 10 times higher than the actual humidity.
Temp.	The temperature resolution is 16 Bits, high bit first out; The value read out by the sensor is 10 times higher than the actual temperature.

DHT22 is a digital sensor consisting of a thermistor and a capacitive sensor for determining the humidity. Figure C.14 shows the block diagram of DHT22.

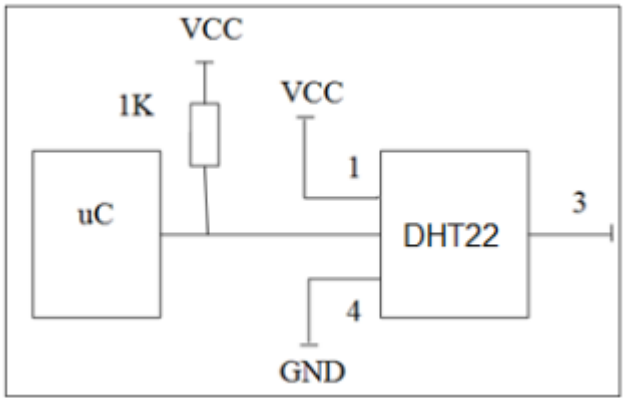


Figure C.14 Block Diagram of DHT22

The voltage supply must be between 3.3V and 6V (recommended 5V). The Arduino Uno board ATMEGA328 microcontroller and the DHT22 sensor are communicated through MaxDetect 1-wire.

Step 1

When the AM2302 has been powered up, it needs 2s to become stable, and the device cannot send any command read device. This sensor tests the temperature and humidity in the environment and collects relative data. When it records data and finishes, sensors go to sleep automatically, and then the SDA data line is pulled up and remains high as per the resistor effect. At this stage, SDA is in the input state and tries to detect the possible signals.

Step 2

The microprocessor chip of the sensor is set as the input/output where the output is set level low more than 800μs and its typical time for hold is 1ms. When the microprocessor chip is high, the bus will be released, and the SDA of the sensor will be increased because of the pull-up resistor. When the bus is released, AM2302 sends a low-level message of 80ms and shows a high output to inform the device of data reception. Signal transmission is shown in Figure C.15.

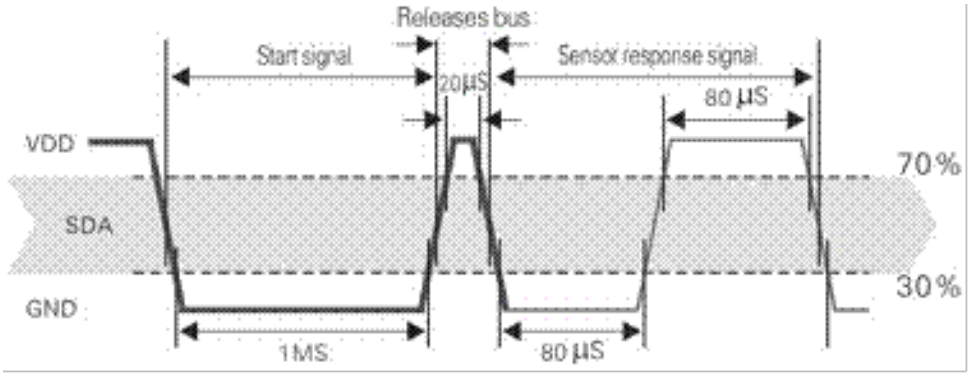


Figure C.15 Signal transmission

Step 3

After the AM2302 sends the response, the SDA continuously outputs a string of 40 bits of serial data, and the microprocessor receives the data according to the changes in the I/O level.

The bit data "0" signal is LOW for 50ms and HIGH for 26-28ms.

Bit data "1" signal: LOW for 50ms and HIGH for 70ms.

The relative signal diagram is shown in Figure C.16.

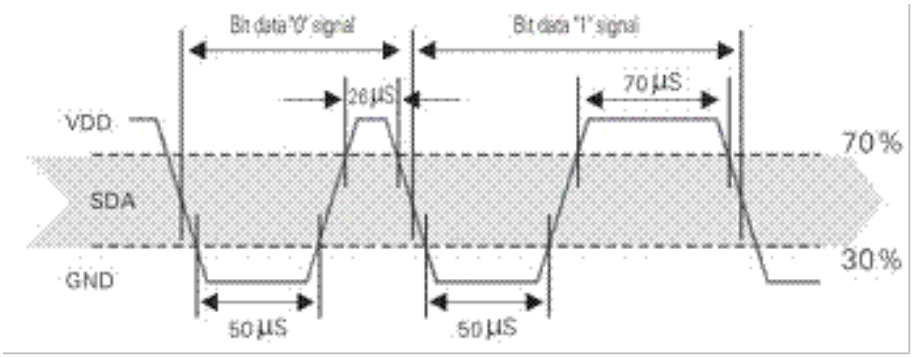


Figure C.16 Signal Diagram

Sensor connectivity is shown in Figure C.17, and some results taken during the testing are shown in Figure C.18 and Figure C.19.

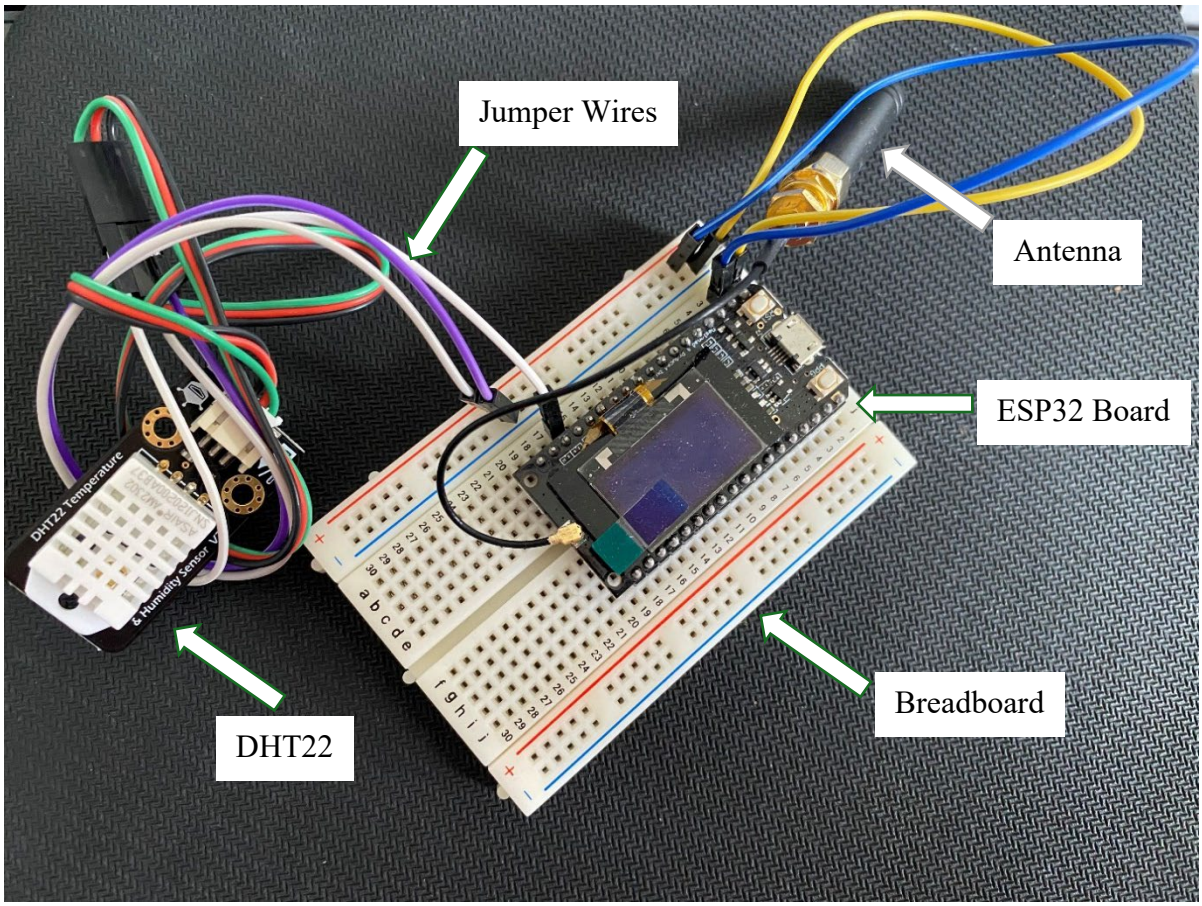


Figure C.17 Structure Diagram of DHT22 working with The Things Stack

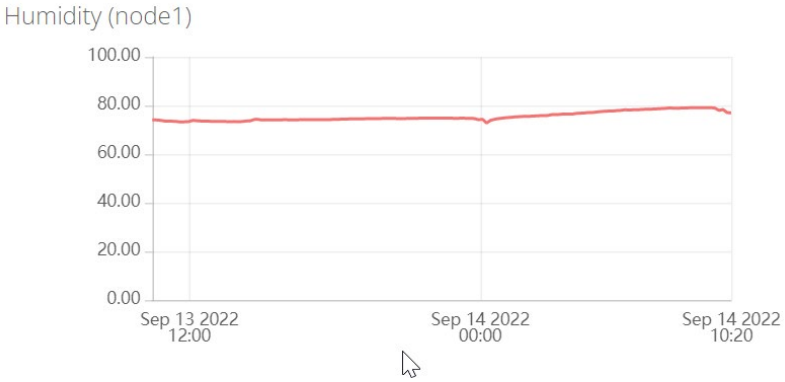


Figure C.18 Humidity Readings

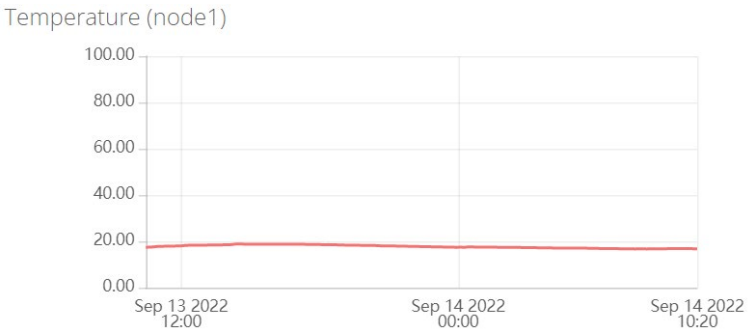


Figure C.19 Temperature Readings

Sensor connectivity code

#include "DHT.h" (DHT library for Arduino IDE)

#define USE_DHT 1 (to enable DHT22 sensor for data collection)

#define DHT22_EN_PIN 17 (DHT22 enable pin setup)

#define TIME_TO_SLEEP (300) (Sleep time for sensor unit to collect data, we can increase or decrease as per requirements)

The below code is for the TTN/The Things Stack configuration. After the data collection or reading the sensor data, the sensor data is transmitted to the things stack using the network key, App key and device address.

```
// TTN Configuration.
static const PROGMEM u1_t NWKSKEY[16] = {0xD2, 0x00, 0xC1, 0xBB, 0x4A, 0x1C, 0xCC, 0x48, 0xDC, 0x3A, 0xF9, 0xAC, 0x48, 0xD9, 0xE1, 0x93};
// LoRaWAN AppSKey, application session key
// This is the default Semtech key, which is used by the early prototype TTN
        // network.
```

```
static const u1_t PROGMEM APPSKEY[16] = { 0x65, 0xA6, 0x64, 0x3F, 0x16, 0x36, 0xC6,
0x3C, 0xAF, 0xFF, 0xE9, 0x97, 0x3B, 0x40, 0x31, 0x95 }
   ;
// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR =0x260C51F8;
```

the below code is used to printing data for temperature and humidity to the network server where the humidity is denoted by % and temperature is measured in Celsius.

```
#if USE_DHT
  display.print("H: " + String(dht.readHumidity()) + String("%"));
  display.setCursor(0,10);
  display.print("T: " + String(dht.readTemperature()) + String("C"));
#endif
```

The below code is used to create the DHT function

```
#if USE_DHT
#define SENSOR_PIN 4
float p_rHumidity,p_temperature, p_co2, p_dust;
DHT dht(SENSOR_PIN, DHT22);
```

After declaring the DHT function, it starts preparing data for temperature and humidity.

```
void prepare_dht_data()
{
   pinMode(DHT22_EN_PIN,OUTPUT); // DHT22 EN
   digitalWrite(DHT22_EN_PIN,HIGH); // DHT22 EN
   delay(1000);
   float temperature = dht.readTemperature();
   Serial.print("Temperature: "); Serial.print(temperature);
   Serial.println(" *C");
   /*
   Serial.print("tmp diff:  ");Serial.println(abs(temperature - p_temperature));
   if(abs(temperature - p_temperature) > 0.1 ) //threshold for temp
```

```
{
  Serial.println("tmp true");
  p_temperature = temperature;
}*/
// adjust for the f2sflt16 range (-1 to 1)
temperature = temperature / 100;

// read the humidity from the DHT22
float rHumidity = dht.readHumidity();
Serial.print("%RH ");
Serial.println(rHumidity);
/*
Serial.print("RH Diff:  "); Serial.println(abs(rHumidity - p_rHumidity));
if(abs(rHumidity - p_rHumidity) > 1) // threshold for humidity
{
Serial.println("rh true");
p_rHumidity = rHumidity;
} */

// adjust for the f2sflt16 range (-1 to 1)
rHumidity = rHumidity / 100;

// float -> int
// note: this uses the sflt16 datum (https://github.com/mcci-catena/arduino-lmic#sflt16)
uint16_t payloadTemp = LMIC_f2sflt16(temperature);
// int -> bytes
byte tempLow = lowByte(payloadTemp);
byte tempHigh = highByte(payloadTemp);
// place the bytes into the payload
payload[0] = tempLow;
payload[1] = tempHigh;

// float -> int
uint16_t payloadHumid = LMIC_f2sflt16(rHumidity);
```

```
    // int -> bytes
    byte humidLow = lowByte(payloadHumid);
    byte humidHigh = highByte(payloadHumid);
    payload[2] = humidLow;
    payload[3] = humidHigh;
}
#endif
#if USE_DHT
    prepare_dht_data();
```

When the process is completed, it started working in a loop if this is successful otherwise, it will report error.

```
#if USE_DHT
    dht.begin();
```

(d) Carbon Dioxide Sensor

CO_2 in the air is considered the main pollutant affecting human health and the environment. This can be monitored with the help of MG811, MQ2 and NDIR sensors implemented on the sensor board with basic network requirements. MG811 sensor is a board sensor component based onthe amplifying output signal and onboard heating circuit. MG811 is highly sensitive to CO_2 and less sensitive to alcohol and CO. It has four interlock connectors and interlock cables in a package of compact size. The MQ2 sensor is compatible with monitoring all gases such as LPG, methane, carbon dioxide, alcohol, hydrogen, smoke, and many more. NDIR system (non-dispersive infrared gas detector), shown in Figure C.20, works on the broadband IR source, and this target the gas will absorb more IR source as the concentration of gas increases. NDIR sensors have some features, such as increasing the gas tube length, optimizing the optical path, reflectingly, being more sensitive, and improving IR source performance.

Figure C.20 CO_2 sensors

The specification of the sensor and pinout configuration is given in Table C.4.

Table C.4 Pinout Configuration

Number	Label	Description
1	Signal	Analogue output (0.4 – 2v)
2	VCC	VCC(4.5-5.5V)
3	GND	GND

The concentration of carbon dioxide (usually 0.03%) is related to daily life. Recently, a study shows that the atmospheric CO2 content has reached 0.0385% (385 ppm), the highest value since 2.1 million years. DFRobot released its latest high-precision analogue infrared CO2 sensor. The effective measuring range is from 0 to 5000 ppm. Besides, its service life could be up to 5 years. It integrates temperature compensation and supports DAC output. Most importantly, the product is easy to use and compatible with all microcontrollers, such as Arduino with ADC function.

Features:

- Waterproof and anti-corrosion
- High sensitivity
- Low power consumption
- Excellent stability

- Temperature compensation
- Excellent linear output
- High cycle life
- Anti-water vapour interference
- No poisoning

Specification:

- Gas Detection: Carbon Dioxide
- Operating Voltage: 4.5 ~ 5.5V DC
- Average Current: &; lt;60mA @ 5V
- Peak Current: 150mA @ 5V
- Output Signal: Analog output (0.4 ~ 2V)
- Measuring Range: 0 ~ 5000ppm
- Accuracy: ± (50ppm + 3% reading)
- Preheating Time: 3min
- Response Time: 120s
- Operating Temperature: 0 ~ 50?
- Operating Humidity: 0 ~ 95% RH (no condensation)
- Service Life: &; gt;5 years
- Size: 37mm * 69mm
- weight: 34g

Code explanation

The below code is used to define the sensor in the Arduino code.

```
#define USE_CO2   1
```

The CO_2 sensor pin is defined as 23 in the code below.

```
#define CO2_EN_PIN 23
```

To transmit the data to the network server, it requires the app's key, network key and device address. This is defined as below from The Things Stack.

```
// TTN Configuration.
static const PROGMEM u1_t NWKSKEY[16] = {0xD2, 0x00, 0xC1, 0xBB, 0x4A, 0x1C, 0xCC, 0x48, 0xDC, 0x3A, 0xF9, 0xAC, 0x48, 0xD9, 0xE1, 0x93};
// LoRaWAN AppSKey, application session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const u1_t PROGMEM APPSKEY[16] = { 0x65, 0xA6, 0x64, 0x3F, 0x16, 0x36, 0xC6, 0x3C, 0xAF, 0xFF, 0xE9, 0x97, 0x3B, 0x40, 0x31, 0x95 };
// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR =0x260C51F8;
```

The CO_2 sensor pin has been defined as below:

```
#if USE_CO2
int sensor = 38;
int concentration = 0;
```

To know the CO_2 density, get a function called using the input and output pins defined.

```
/* get CO2 density */
void get_co2()
{
    pinMode(CO2_EN_PIN,OUTPUT); // CO2 EN
```

```
  digitalWrite(CO2_EN_PIN,HIGH);
  delay(2000);
  //Read voltage
   int sensor value = analogRead(sensor);
```

The CO_2 density values are converted from analogue to voltage using the below equations

```
 // The analog signal is converted to a voltage
 float voltage = sensor value*(3.3/4095.0);
 if(voltage == 0)
 {
  Serial.println("Fault in CO2 sensor");
  concentration = 0;
 }
 else if(voltage < 0.4)
 {
  Serial.println("preheating CO2 sensor");
  concentration = 0;
 }
 else
 {
  float voltage_diference=voltage-0.4;

  concentration=(int)((voltage_diference*5000.0)/1.6);

  Serial.print(" CO2 concentration: ");
  Serial.print(concentration);
  Serial.print(" ppm\t");
 }
}
#endif
```

To p-tint the CO_2 concentration levels, the statements below were used.

```
#if USE_CO2
 display.setCursor(50,10);
```

```
  display.print("CO2: " + String(concentration) + String("ppm"));
#endif
#if USE_CO2
      /* CO2 value */
      get_co2();
      uint16_t payloadCO2 = LMIC_f2sflt16(concentration);
      byte co2Low = lowByte(concentration);
      byte co2High = highByte(concentration);
      payload[6] = co2Low;
      payload[7] = co2High;
#endif
pinMode(CO2_EN_PIN,OUTPUT);       // CO2 EN
```

Some results taken during the testing of the sensors are given in Figure C.21.

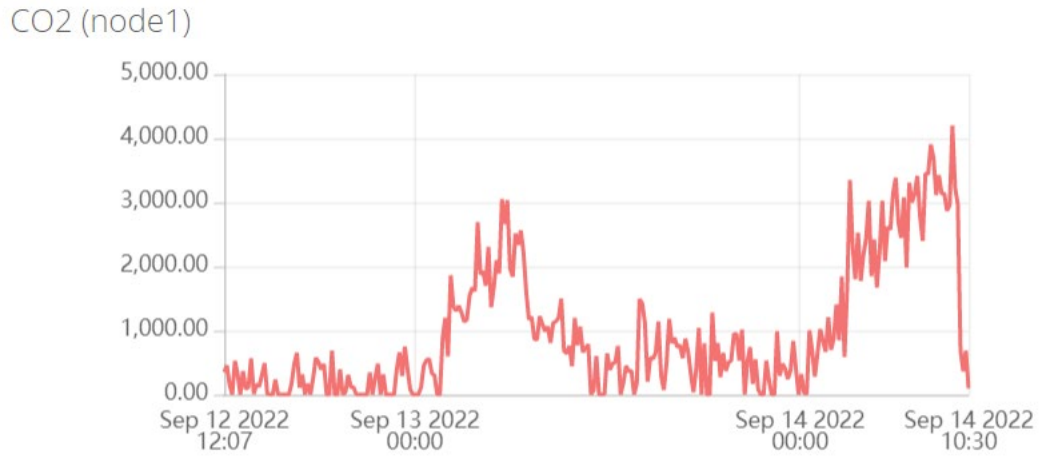


Figure C.21 CO_2 Reading

Appendix D Results

Figure D.1 and Figure D.2 shows the CO_2 concentration collected on Pole 1 and Pole 2 at the distances of 0.5m (CO2-1), 1m (CO2-2), and 1.5m (CO2-3).

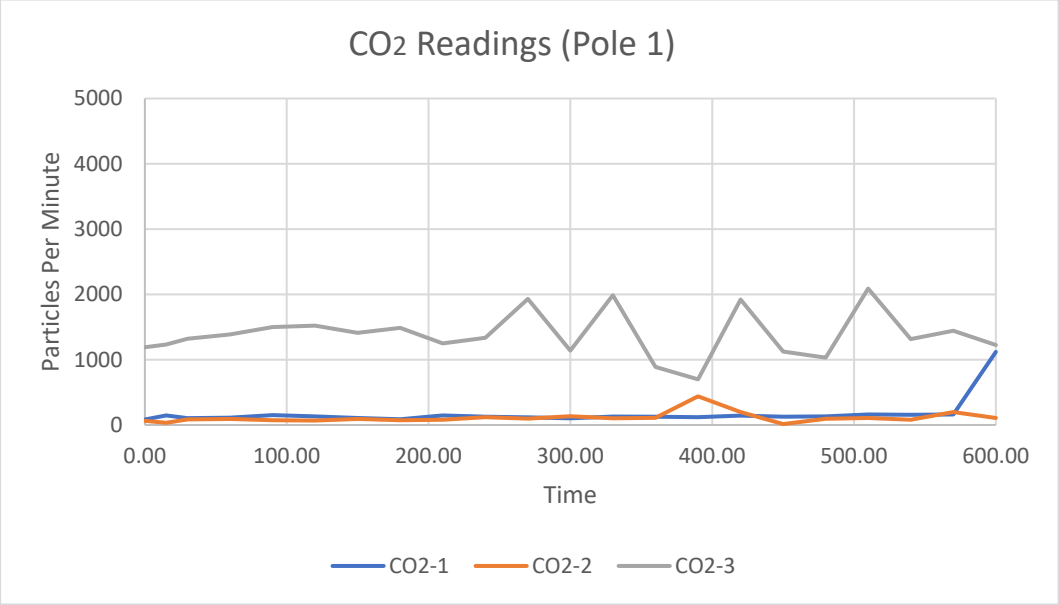


Figure D.1 Pole 1 CO_2 Readings

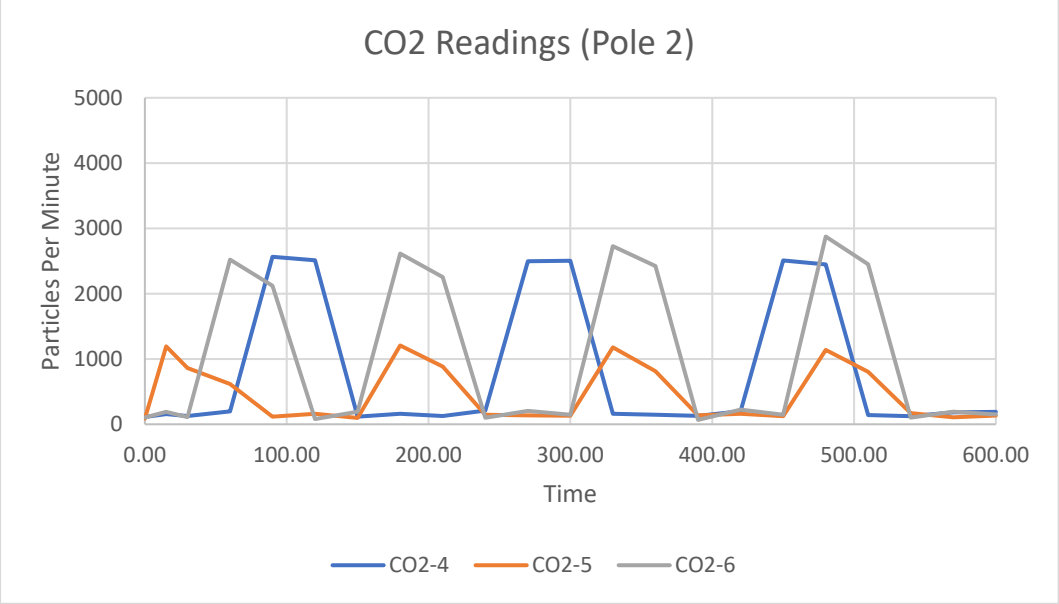


Figure D.2 Pole 2 CO_2 Readings

Figure D.3 to Figure D.4 shows the temperature and humidity levels collected on Pole 1 and Pole 2 at the distances of 0.5m (Temp-1), 1m (Temp-2), and 1.5m (Temp-3).

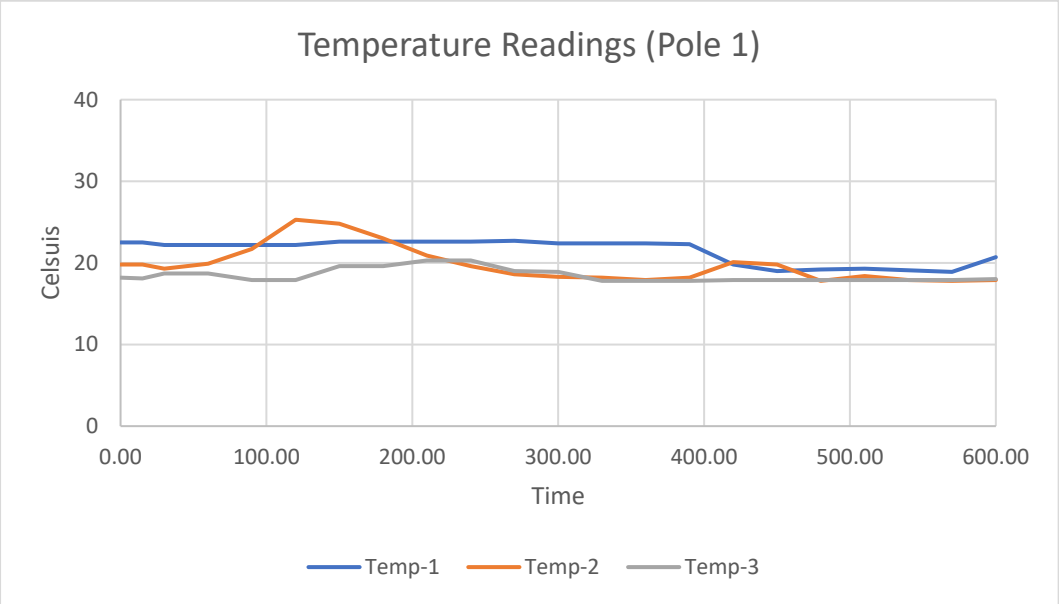


Figure D.3 Pole 1 Temperature Readings

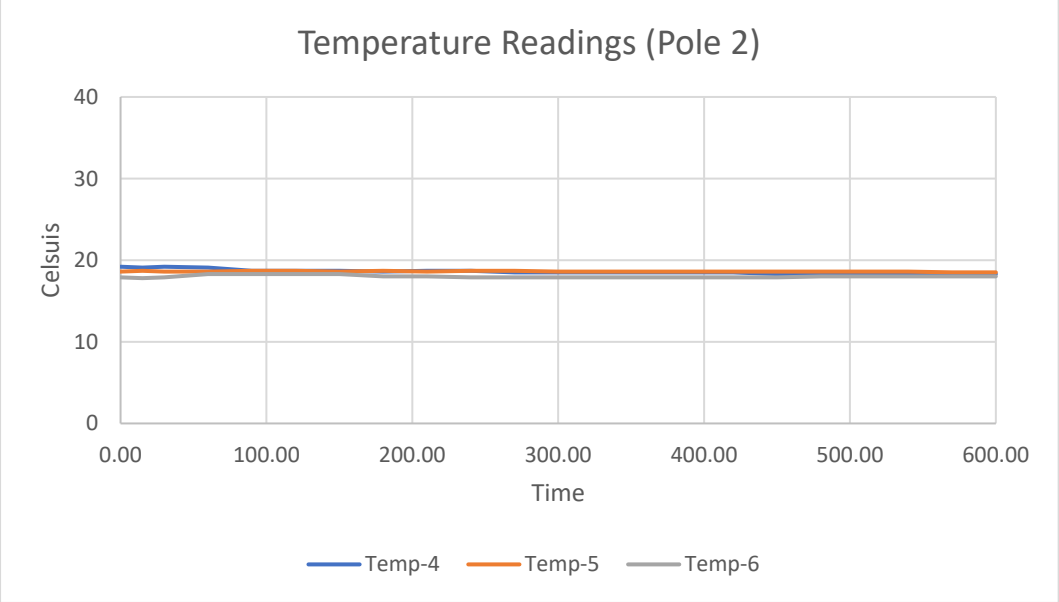


Figure D.4 Pole 2 Temperature Readings

Figure D.5 to Figure D.6 shows the temperature and humidity levels collected on Pole 1 and Pole 2 at the distances of 0.5m (Humidity-1), 1m (Humidity-2), and 1.5m (Humidity-3).

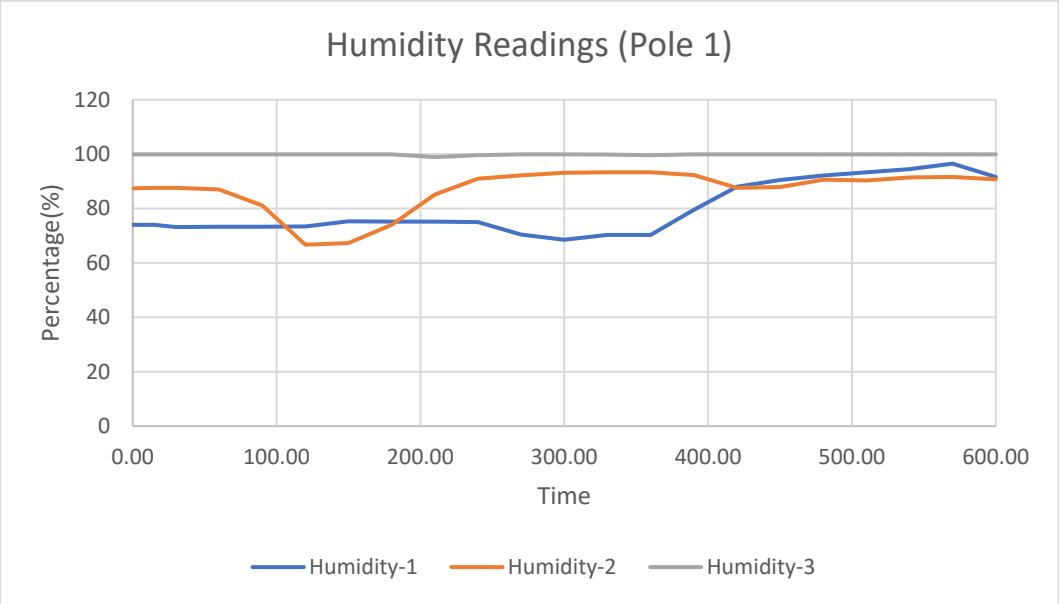


Figure D.5 Pole 1 Humidity Readings

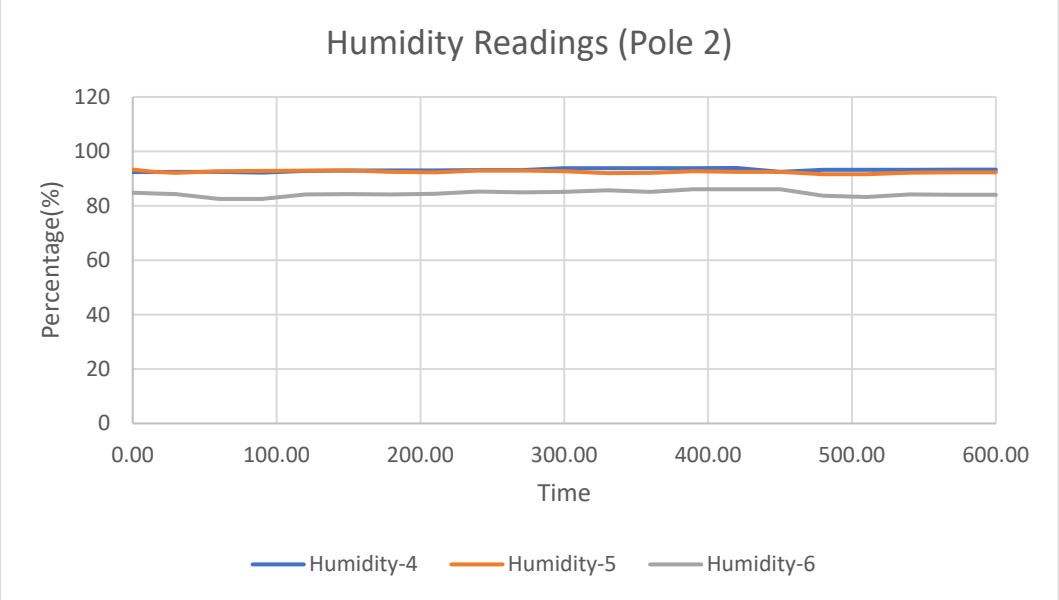


Figure D.6 Pole 2 Humidity Readings

Figure D.7 to Figure D.8 shows the PM_{10} levels collected on Pole 1 and Pole 2 at the distances of 0.5m (PM10-1), 1m (PM10-2), and 1.5m (PM10-3).

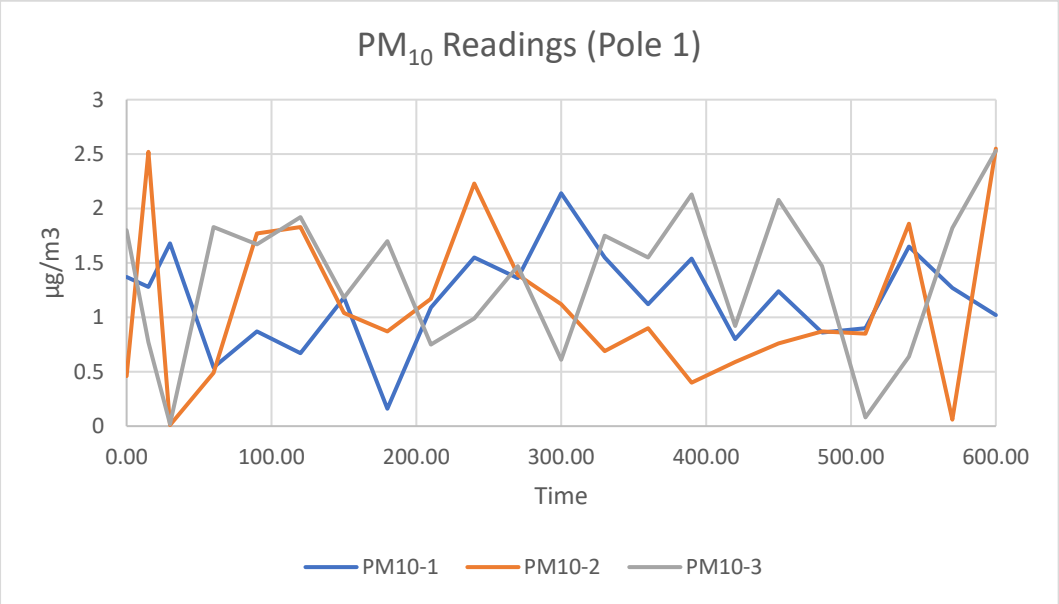


Figure D.7 Pole 1 PM_{10} Readings

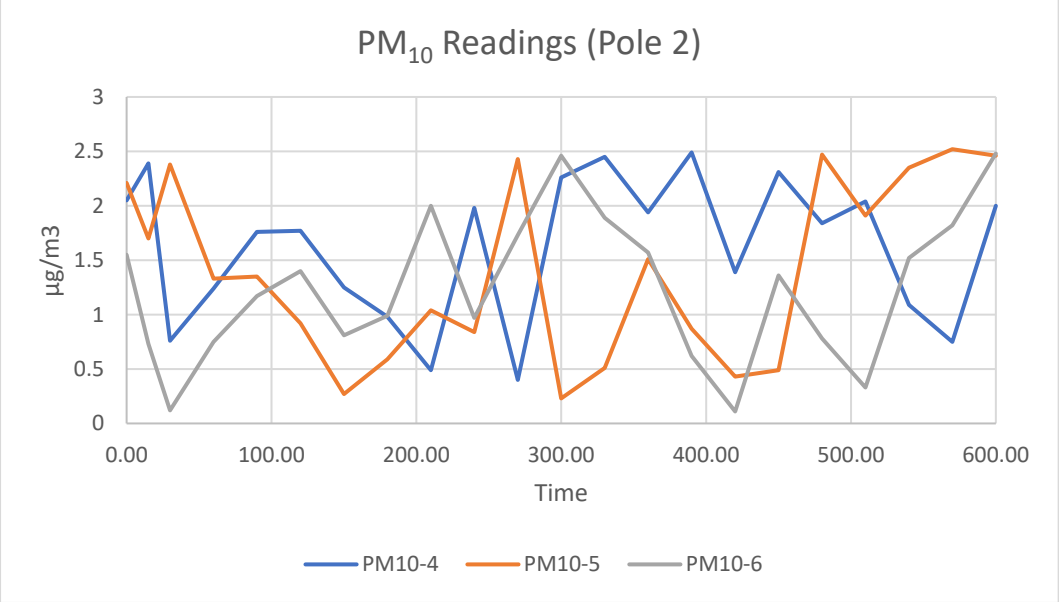


Figure D.8 Pole 2 PM_{10} Readings

Figure D.9 to Figure D.10 shows the PM_1 levels collected on Pole 1 and Pole 2 at the distances of 0.5m (PM_1-1), 1m (PM_1-2), and 1.5m (PM_1-3).

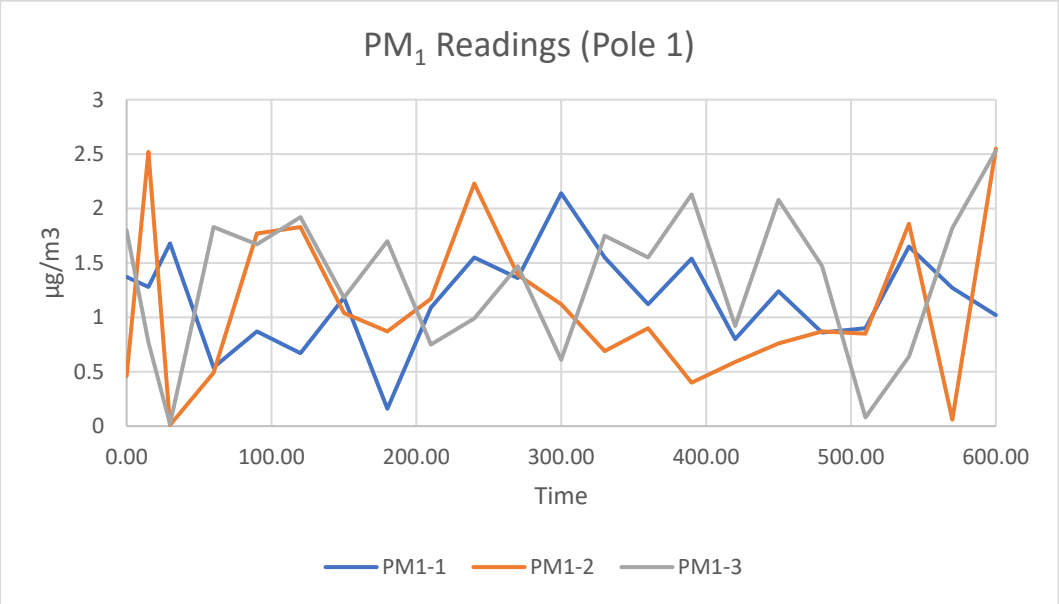


Figure D.9 Pole 1 PM_1 Readings

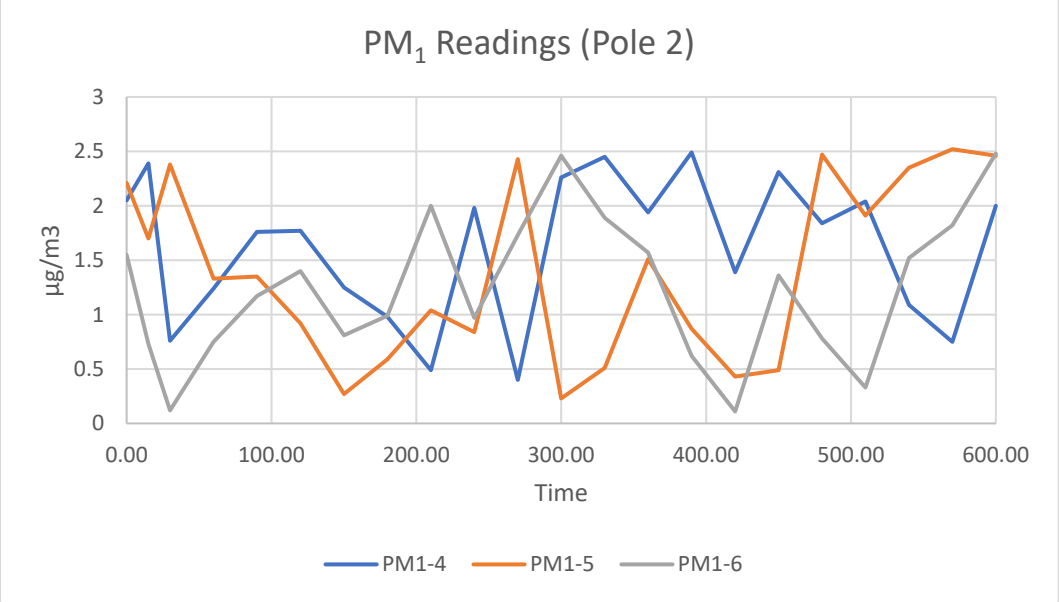


Figure D.10 Pole 2 PM_1 Readings

Figure D.11 to Figure D.12 shows the $PM_{2.5}$ levels collected on Pole 1 and Pole 2 at distances of 0.5m (PM2.5-1), 1m (PM2.5-2), and 1.5m (PM2.5-3).

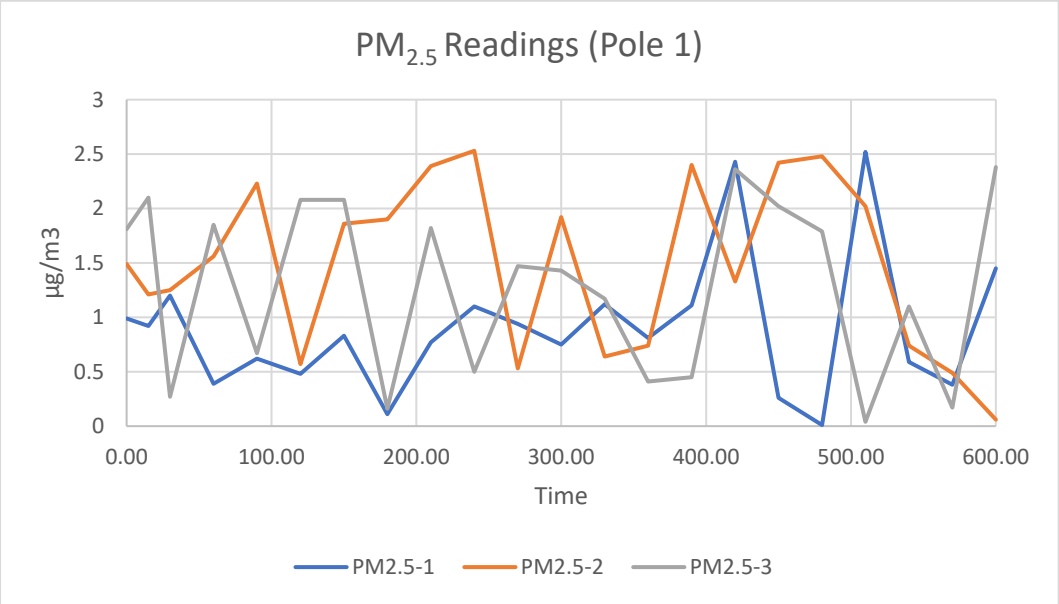


Figure D.11 Pole 1 $PM_{2.5}$ Readings

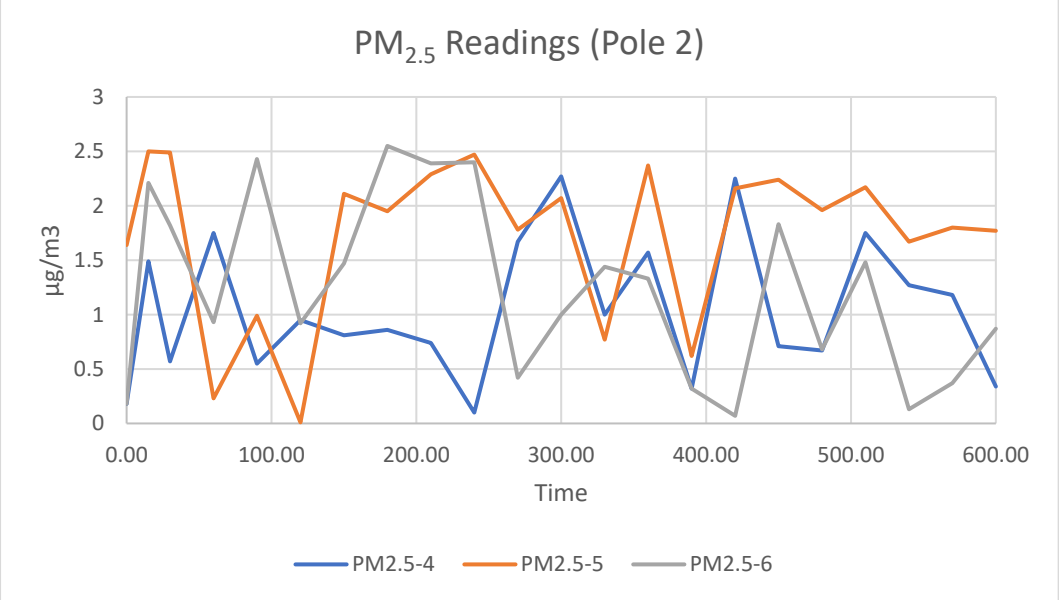


Figure D.12 Pole 2 $PM_{2.5}$ Readings

Figure D.13 to Figure D.14 shows the PM_4 levels collected on Pole 1 and Pole 2 at distances of 0.5m (PM4-1), 1m (PM4-2), and 1.5m (PM4-3).

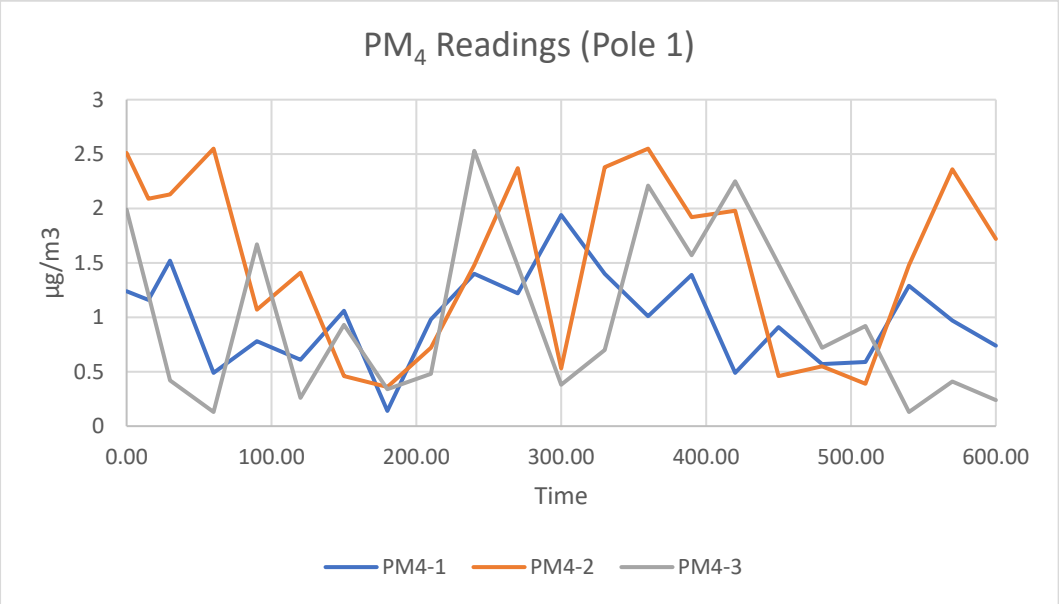


Figure D.13 Pole 1 PM_4 Readings

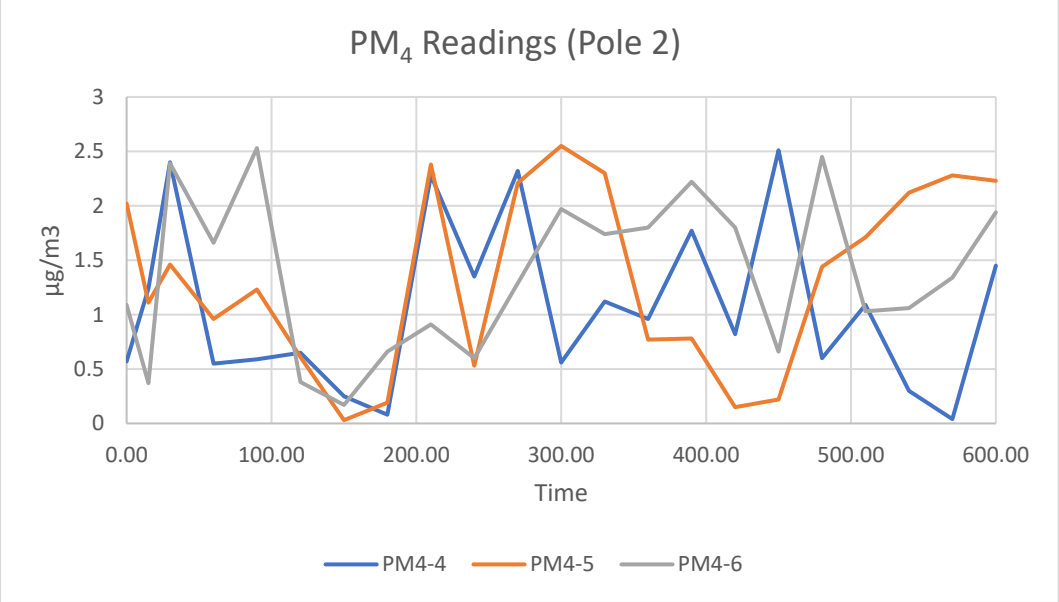


Figure D.14 Pole 2 PM_4 Readings

Figure D.15 to Figure D.16 shows the RSSI readings on Pole 1 and 2.

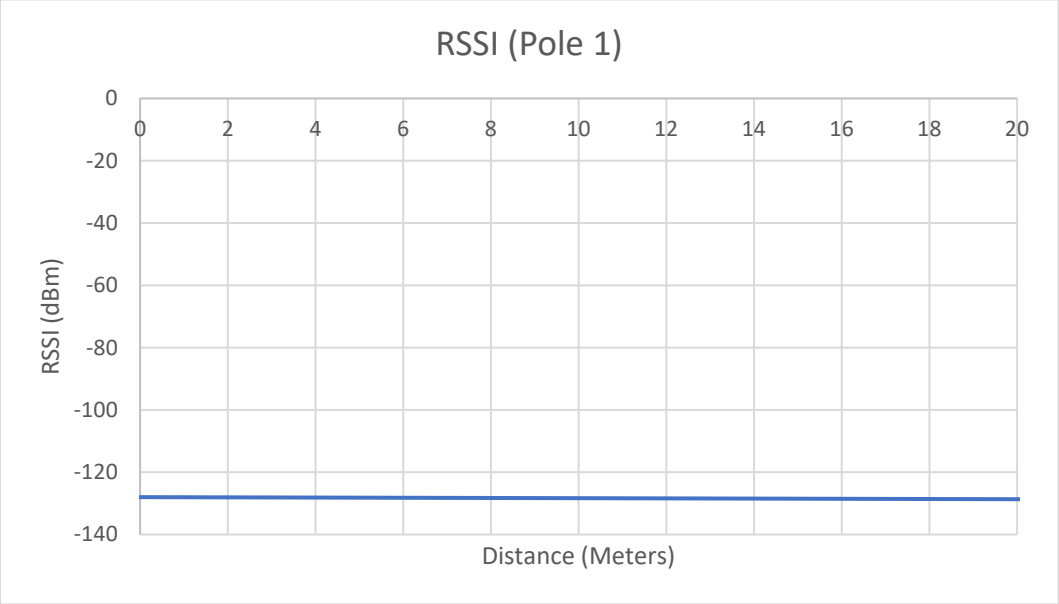


Figure D.15 Pole 1 RSSI Readings

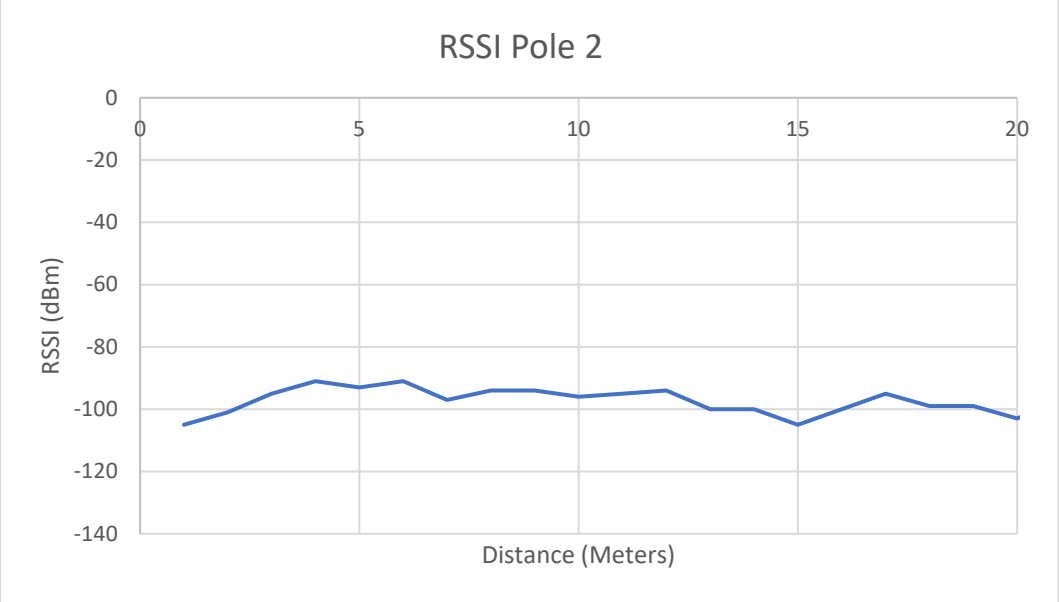


Figure D.16 Pole 2 RSSI Readings

Appendix E Conferences and Poster/Paper Publications

Conferences

1. A. Kaur and J. Kilby, *"Hyperlocal LoRa Air Pollution Monitoring Network Setup,"* 2021. 4th International Conference on Computer Networks, Big Data and IoT ICCBI 2021, 2021.
2. Amritpal Kaur, Jeff Kilby, *"LoRa Air Pollution Monitoring Network Using Data Fusion Algorithm,"* International Conference on Science, Technology, Engineering and Management (ICSTEM-2021) Winnipeg, Canada, 2021.
3. A. Kaur and J. Kilby, *"Wireless Sensor Networks (WSN's) for Air Pollution Monitoring: A Review,"* 4th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), 2022.
4. A. Kaur and J. Kilby, *"Performance Measurement of a Long Range (LoRa) Network using the ns-3 Simulator,"* Wireless Telecommunications Symposium (WTS 2022), 2022 Wireless Telecommunications Symposium, Virtual Conference, USA, 2022.

Publications

Paper 1. Kaur, A. (2018, August). *"Analysis of Low-Power Consumption Techniques in Wireless Sensor Networks"*, Poster session presented at the Auckland University of Technology Postgraduate Research Symposium, Auckland, New Zealand.

Paper 2. Kaur, A. (2019, August). *"Air Pollution Monitoring using Wireless Sensor Networks"*. Poster session presented at the Auckland University of Technology Postgraduate Research Symposium, Auckland, New Zealand.

Paper 3. Amritpal Kaur, Jeff Kilby, (2021) *"LoRa Air Pollution Monitoring Network Using Data Fusion Algorithm"*, International Journal of Advance Computational Engineering and Networking (IJACEN), pp. 4-9, Volume-9, Issue-11.

Paper 4. Kaur, A. (2021, November). *"LoRa Air Pollution Monitoring Network System"*. Poster session presented at the Auckland University of Technology Postgraduate Research Symposium, Auckland, New Zealand.

Paper 5. Kaur, A., Kilby, J. (2023). *"Development of a LoRa Network for Monitoring Particulate Matter"*. In: Smys, S., Lafata, P., Palanisamy, R., Kamel, K.A. (eds) Computer Networks and Inventive Communication Technologies. Lecture Notes on Data Engineering and Communications Technologies, vol 141. Springer, Singapore.

Paper 6. Kaur, A., Kilby, J. (2023). *"Wireless Sensor Networks (WSNs) in Air Pollution Monitoring: A Review"*. In: Rajakumar, G., Du, KL., Vuppalapati, C., Beligiannis, G.N. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. Lecture Notes on Data Engineering and Communications Technologies, vol 131. Springer, Singapore.

Paper 1 – Poster Presentation

AUT UNIVERSITY
TE WĀNANGA ARONUI O TAMAKI MAKAU RAU

Analysis of Low-Power Consumption Techniques in Wireless Sensor Networks

Amritpal Kaur

ABSTRACT

Wireless Sensor Networks (WSNs) has given rise to the growth in technology for sensing and performing tasks in different environments. This technology has many applications in everyday life, such as in farming, hospitals and emergency services, military and some offshore structures. WSNs is made up of a set of nodes that are capable of sensing, storing, processing, and communicating information. When developing a WSN applications there is always a limited amount of energy available in the senor nodes. Using this technology, the battery life of the sensor node can last years or be consume in a matter of weeks. So, researchers and developers always aim to use best practices in order to reduce the error estimation of the data and hence power consumption of the WSNs.

The main objective of my research is investigate possible ways to reduce the power consumption of sensor nodes when transferring or measuring the data over the network nodes for high performance and low-energy consumption using the existing protocols for energy saving such as the Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol.

A software design methodology will be used for my research, which will implement sensor nodes for transmitting data using simulation techniques with using existing low energy protocols. The simulation will use available WSNs software such as TinyOS and TOSSIM to measure the effective results of the low power consumption. This method will considered incremental improvements in battery efficiency. From my simulated findings I will introduce these the same setups and conditions using the actual low energy protocols investigated on WSNs with actual sensor nodes and measure the energy values during transmission for comparison.

CONTACT

INTRODUCTION

With the increase in the computational and sensing technology along with the battery life, the available wireless sensor nodes have the paradigm for horizontal ambient intelligence infrastructure feasible. WSNs are very popular these days for the different applications ranging from monitoring of environment, object tracking, and surveillance. WSNs owed the requirement for low device complexity together with low power consumption. These offers the benefits for different application as to reduce the threats like- eavesdropping and hardware tampering. There are several key management approaches that have been planned for establishing the exploiting different methods or techniques, such as distributing randomly different secret materials and transitory master key. Today a variety of techniques are available for WSN security protocols such as TDMA, Multi-hop Low-energy Adaptive Clustering Hierarchy (M-LEACH), Distributed Cluster Head Scheduling (DCHS) and the security requirements (Extensible Mark-up Language , Simple Object Access Protocol, Web Services Description Language) using the Threshold Policy, these analyse the levels of the energies in the sensor nodes and calculate the life period of each nodes. Sensor nodes consume more power when the networks are large, so a there will be a need to set up the activation and de-activation of the sensor nodes, this process can be done by a threshold policy. With the help of LEACH protocol which is a self-organised, adaptive to distribute energy among sensor nodes that uses homogeneous nodes and is cost effective and independent of environmental irregularity.

Experiment

For the experiment, implementation of LEACH will be done and starts measuring performance of the network as shown in figure 1. This is self-adaptive cluster organized and clusters consists of nodes. Each cluster has one cluster Head (CH). CH is working on the basis of two phase steady and set-up. Cluster formation is shown in figure 2. Selection of the clusters are based on the high power nodes. When the CH sends message to the node then it became CH. This can be measured with Threshold policy used in network. Number of nodes in the network is always less than the threshold. Set-up of the network is shown in figure 3 and parameters for the simulation are shown in Table 1.

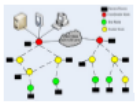


Figure 1. Wireless Network

Figure 2. Cluster Formation

Leach Protocol is effective for reducing traffic over the network, uses single hop routing, increases the lifetime of sensor nodes and no need of location while creating cluster.

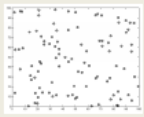


Figure 3. Simulated environment

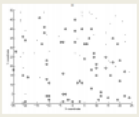


Figure 4. Live and dead nodes.

RESULTS

According to figure 4, simulation network is formed with 100 nodes and nodes will be tested randomly and measured in metres. LEACH is distributing the power among nodes equally and gives the double lifetime power for the simulated network.
When considering these parameters for the simulation process, then results of the assumed parameters are shown in figures 5 and 6.

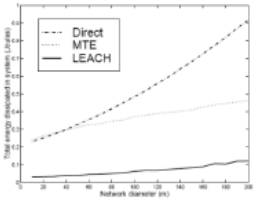


Figure 5. Power Dissipation in Network.

For 100 node's network assumption, the total energy is 200J. For LEACH, when the first dead node appears in the 400th round, the total consumption energy is 164.295587J and the energy is used up until the 530th round. However, for traditional Flooding protocol, the energy is used up before the 200th round. It proves that LEACH is a energy-effective protocol which could save energy consumption and improve the performance of network.

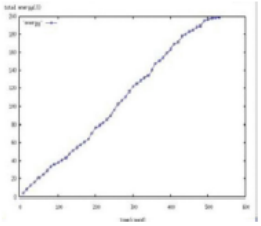


Figure 6. Simulation Results

Table 1. Parameters in Simulation.

Nodes	100
Network size	100 m X 100 m
Base station location	(50, 175)
Radio propagation speed	$3x10^8$ m/s
Processing delay	50 µs
Radio speed	1 Mbps
Data size	500 bytes

CONCLUSION

Figures 7 and 8 shows network live time only considering set-up phase on the condition of different clusters head proportion in each algorithm. Obviously, improved algorithm can obtain longer live time and the higher clusters heads proportion the longer live time in LEACH.

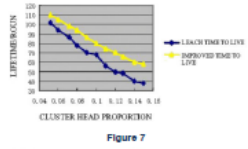


Figure 7

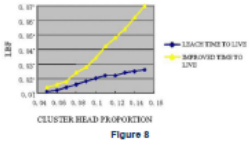


Figure 8

REFERENCES

Gangane, S., & Dhawan, G. (2013, 6 24). *Performance Analysis of LEACH Protocol.* Retrieved from International Journal of Science and Research (IJSR) : https://pdfs.semanticscholar.org/0b25/8e7e736f837ed0436ca4158f0477e8bc699b.pdf

Kaur Gill, R., Chawla, P., & Sachdeva, M. (2015, April 24). *Study of LEACH Routing Protocol.* Retrieved from sbsstc: http://www.sbsstc.ac.in/icccs2014/Papers/Paper42.pdf

Low Energy Adaptive Clustering Hierarchy. (2007, 1 24). Retrieved from) International Journal of Computer Science and Information Technologies,: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.589.470&rep=rep1&type=pdf

Paper 2 – Poster Presentation

AIR POLLUTION MONITORING USING WIRELESS SENSOR NETWORKS

Air Pollution Monitoring

- Air pollution is the introduction of biological materials, chemicals, particulates or toxic gases of into the atmosphere that can cause discomfort, disease or death to humans and damage other living things such as food crops or the natural environment.
- Common Air pollutants are NO_2, CO, SO_2, O_3, $PM_{2.5}$ and PM_{10}.
- For tracking the effect of air pollution on the environment and health, this is necessary to monitor the level of contamination in urban and suburban areas.

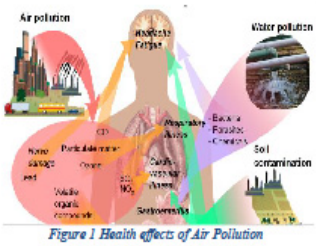


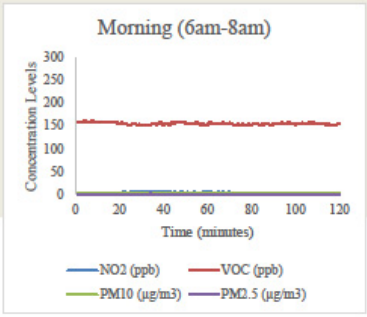
Figure 1 Health effects of Air Pollution

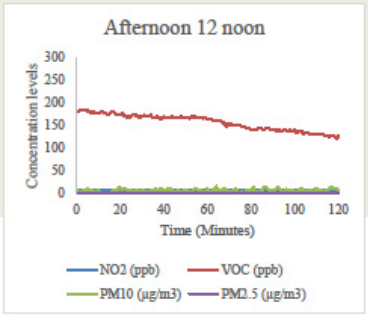
- Air pollution monitoring is essential for well-being for the humans, living animals and the environment.
- Figure 1 Health effects of Air Pollutionair pollution is causing health issues such as building syndrome, cancer, and adverse pregnancy outcomes, morbidity and mortality from cardiorespiratory and lung cancer.
- According to the World Health Organisation, outdoor air pollution and particulate matter from outdoor air pollution as carcinogenic to humans, based on enough evidence of carcinogenicity in humans and experimental animals and strong mechanistic evidence

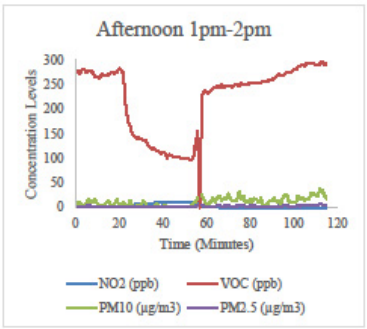
Air Quality Index - Ozone	
301–500	Hazardous
201–300	Very Unhealthy
151–200	Unhealthy
101–150	Unhealthy for Sensitive Groups
Spare The Air – reduce driving – when the AQI is forecast to meet or exceed 126.	
51–100	Moderate
0–50	Good

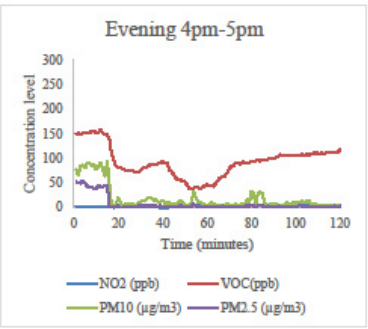
Figure 2 AQI New Zealand

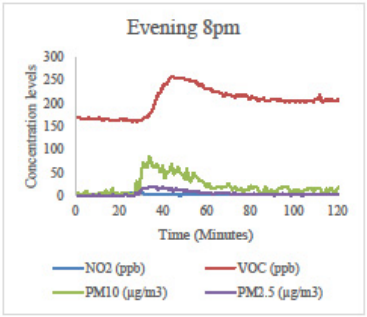
Flow-Meter Results











'Flow' developed by Plumes Lab

Discussion

From the above results obtained by a unit called *'Flow'* developed by Plumes Lab, shows the pollution level at various times of the day. The results presented above show that for early morning, the pollution level is low as compared to the evening time. Pollution level increases due to the rise in human activities and emission from transport. For the collection of data using the *Flow* which monitors the concentration of various gases and particles, a volunteer walked down Queen Street from Wellesley Street to the Britomart. The *Flow* unit has inbuilt sensor system that monitors the pollution levels and generates a map with GPS, which is connected to a mobile phone using Bluetooth and stores the data to the cloud for analysis for a later date.

The *Flow* is an all in one unit consisting of hardware, software and firmware, and the air being monitored enters the unit via *'360 air-intakes'*. For monitoring particulate matter (PM) a sensor lights up; photovoltaic cell translates the laser's light. Two tiny molecule-capturing membranes heated up for 250 degrees to measure VOC and NO_2. All the inbuilt sensors are calibrated together and generating pollution on the specified location after the activation of the neural networks. The measured values are stored as parts per billion (ppb) for both VOC and NO_2 and micrograms per cubic meters for PM, from which the *Flow* can calculate it as an Air Quality Index (AQI) value.

Amritpal Kaur
amritpal.kaur@aut.ac.nz
School of Engineering
Supervisors: Dr Jeff Kilby/Dr Hakilo Sabit

Paper 3 – Journal Paper

International Journal of Advanced Computational Engineering and Networking, ISSN(p): 2320-2106, ISSN(e): 2321-2063
Volume-9, Issue-11, Nov.-2021, http://iraj.in

LORA AIR POLLUTION MONITORING NETWORK USING DATA FUSION ALGORITHM

[1]AMRITPAL KAUR, [2]JEFF KILBY

[1,2] Department of Electrical and Electronic Engineering, Auckland University of Technology, New Zealand
E-mail: [1]amritpal.kaur@aut.ac.nz, [2]jeffrey.kilby@aut.ac.nz

Abstract - The success of the LoRa network deployment is entirely dependent on the quality of services (QoS), that considered the issues of data redundancy, delays, and network lifetime. The level of air pollution has increased over time due to several factors such as an increase in population, increased vehicle use, industrialization, and urbanization, which results in harmful effects on human well-being by directly affecting the people exposed to it. This paper presents a LoRa Air Pollution Monitoring system developed to monitor pollution using the LoRa networking components. The network system is based on the sensor connectivity to the LoRa modules. These are directly linked to The Things Network server to obtain results from the sensor nodes, and data is displayed/stored on the cloud with a graphical application using ubidots. The ubidots website generates a graphical user interface for online real-time monitoring of the air pollutants, carbon dioxide, particular matter and temperature/humidity values. The network generates continuous data, which can cause redundancy in the network and affect the network life cycle. To overcome the issue of the LoRa network to reduce data redundancy, it requires implementing a data fusion algorithm.

Keywords: LoRa, sensors, data fusion algorithm.

I. INTRODUCTION

Wireless Sensor Networks (WSNs) integrates nested computation and sensor techniques with distributed information techniques. WSNs are widely used in national defense, military affairs, environment monitoring, medical care, family, etc. Recent studies worldwide have revealed the relation between air pollution, fine particles and other toxic gases and human health[1-3]. The impact of air pollution is overwhelming these days[4]. One of the basic requirements of human health and well-being is clean air. However, the World Health Organization (WHO) estimates that around 1.4 billion urban residents worldwide live in areas with air pollution and reports that air pollution kills approximately 3 million people solely to outdoor pollution[5].

Creating complex monitoring systems or networks for air pollution monitoring usually has a high capital cost associated with an expensive pollution monitoring system. To overcome this, governments are installing low-cost sensor solutions. To date, several studies have been implemented to monitor environmental conditions. These include particle matters and toxicgases sensors, which monitor air pollution's effects on human health.

Hall et al. [6] describe the data fusion method that data fusion techniques can work after combining the data collected from sensors and achieving data redundancy and providing more accuracy. Data fusion techniques have been used to fuse and aggregate the data received from multiple sensors for multiple sensor deployment. Data fusion methods are a more reliable environment as compared to the baseline components. From the recent studies, data fusion algorithms have various types, such as data association, state estimation and decision fusion. This paper aims to provide the main steps involved in the implementation of a network design. A LoRa network is proposed to monitor pollution using three sensors:temperature/humidity, particulate matter (PM) and carbon dioxide (CO_2) sensors. Toreduce redundancy in sensor values, a sensor data fusion algorithm was used.

Thepaperisdividedinto several sections. Section 2 gives a brief literature review of different pollution monitoring techniques. Section 3 presents the network design for air pollution monitoring. Section 4 explained the data fusion algorithm and a brief explanationof different types and explained the process of data fusion algorithm in terms of the pseudocode. Section 4 and 5 covers both the hardware and software needed to set up the WSN. Section 6 describes the working of the algorithm implemented over the LoRa network. Section 7 presents the results from the research presented in the paper. The final section gives the conclusion and future work.

II. RELATED WORK

The literature review presented in this paper focused on different air pollution monitoring methods used with wireless sensor networks. After reviewing several papers found relevant air pollution monitoring, either simulation or deployment based.Other research methods were implemented by several researchers[7-10]. So, to monitor the effects of air pollution on human health. One literature survey was conducted to monitor air pollution, including the challenges in selecting biosensors[11]. The author proposed the ontology method for species

based on the integrative fuzzy knowledge-based system. This methodology can be utilized for long-term decision making and long-term policymaking for pollution monitoring. Researchers focused on CO_2 emission, and the proposed system implements the intelligent transport system to reduce fuel consumption and CO_2 emission; this is a vehicular-based sensor network[12]. Indoor pollution monitors the parameters such as temperature, humidity, air quality, etc.

Real-time simulation projects were implemented and tested in indoor and outdoor environments by researchers using wireless sensor networks. Researchers monitored indoor air pollution using a diversity of methods and techniques. Researchers proposed the network based on sensor level, node level and network level. So, to monitor toxic gases and particles in an indoor environment, various sensors were used, such as pyroelectric infrared sensors (PIR) and metal oxide sensors. Semiconductor gas sensors, commercial sensors,nano-technology sensors and low-cost sensors[13]. Simulation projects were implemented with wireless sensor networks to improve the sensor nodes' power while transmitting data over the network[14]. To ensure the accurate reading of CO_2 and SO_2, the P-Tree-based association rule mining algorithm was implemented and tested using the calibration method [15]. Alassi et al. proposed the network in Qatar city with three different locations for monitoring concentration levels of CO_2, NO_2, and methane (CH_4). This also monitors the temperature, humidity, solar radiance, pressure, wind speed, and direction [16]. The square estimation-based method was used to alter the behaviour and daily activities to improve air quality and make people aware that indoor air quality has higher health risks than outdoor pollution [17]. Machine learning methods were used with Low-Power Wide Area Network (LPWA) technology to address air quality issues [18]. It was implemented over the large coverage area using the Internet of Things (IoT) cloud [18]. Peng et al.[19]developed the methodology for power consumption of WSNs and improving the accuracy of gas sensors. The implementation processes using state-of-the-art technology and commercial off-the-shell photoionization detector gas sensors. Rohde et al. [20] developed the mapping approach for monitoring particles and toxic gases in the environment and derived the pollution maps for eastern China for four months.

III. NETWORK DESIGN

A block diagram of the network design shown in Figure. 1 shows three sensors connected to the LoRa module, and these works are based on a wired connection to form the sensor node. Next, the LoRa gateway and LoRa modules communicate based on the Industrial, Scientific, and Medical (ISM) radio frequency band of 915MHz. When the LoRa gateway receives the data, it is transmitted to the TTN server using its network ID. TTN server is a free cloud service to store pollutant data and is later displayed on a graphical user interface (GUI) usingubidots.

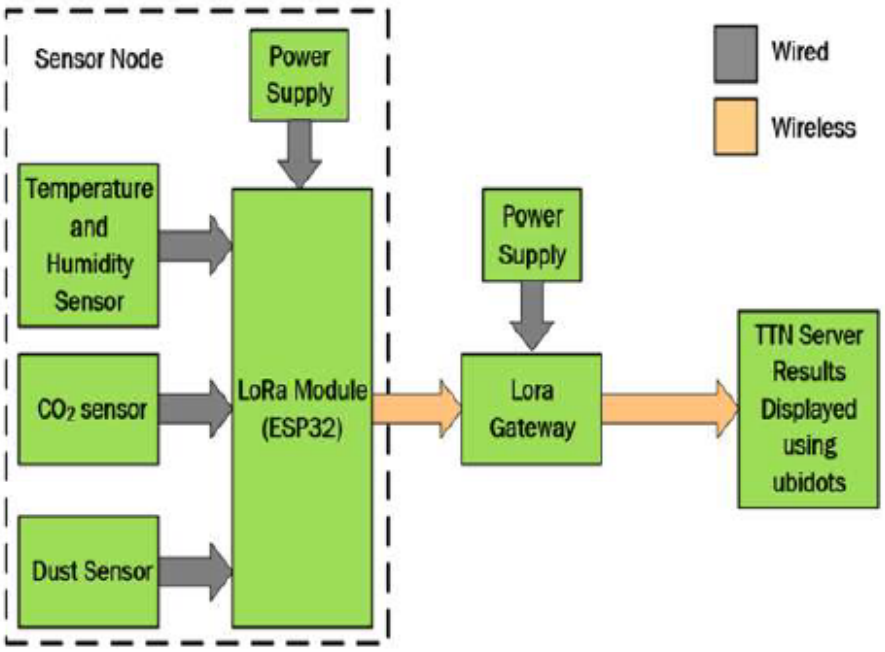


Figure. 1. Block Diagram of Proposed Network

Figure 1 shows the proposed network. Three different sensors such as temperature and humidity, CO_2, and dust are related to the LoRa module. LoRa module connects to the power supply so that it can transmit data over the network.

IV. HARDWARE

Thissectionexpandsonthehardwareusedforthe proposed network presented in the paper. Selecting the components for the network required the specifications and requirements of the network components selected based on range, type of response time, sensitivity, lifetime, cost, availability and dimensions of the sensor and other components. The hardware requirements for each sensor node are asfollow:

4.1.LoRaGateway

The LoRa Gateway is the modemused in the network. The modem works on the 915MHz frequency, while the ESP32 is responsible for the Wi-Fi capabilities. The gateway converts LoRa radio messages into data packets that can access the web, relaying their messages to thecloud [21].

4.2 .Temperature and HumiditySensor

The temperature and humidity readings were collectedusing a digital-output capacitive-type relative humidity and temperaturesensor/module.TheDHT22sensoroutputsa calibrated 8-bit digitalsignal [22].

4.3.Particular Matter

This is an optical air quality sensor designed to monitor and measure particle matters. It works based on an infrared emitting diode and a phototransistor to detect which measures the particles in the air that passes through it [23].

4.4.Carbon DioxideSensor

CO_2 was measured using a high-precision analogue infrared sensor ranging from 0 to 5,000 parts per million (ppm). This sensor is based on non-disperse infrared technology and has good selectivity and oxygen-free dependency. Most importantly, the product is easy to use; it is compatible with all microcontrollers with analogue to digital converters[24].

V. SOFTWARE

To configure the network devices presented in thispaper, code was required to write code to interface each sensor to the LoRa module and the LoRa gateway to transmit data to The Things Network (TTN). The TTN isan open-source, decentralized infrastructure for the IoT. This requires the server to be configured to communicate datawith the LoRa modules via the LoRa gateway. A dashboard via the internet needed to be developed to review the received data from the sensors stored on the TTN server. The following components were used to create the proposednetwork presented in this paper.

5.1. Integrated Development Environment

The ArduinoIntegrated Development Environment (IDE) was used to code all the network components, which uses the C++ language. This IDE is compatible with writing and uploading the code required to configure the sensors connected to the LoRa ESP32 boards. Also, code was needed to transmit the data via a wireless link to the LoRa gateway and the cloud server.

5.2. The Cloud Server

The data for the network was stored on a cloud server. The Things Network (TTN) The TTT server is a free cloud service to create innovative research applications. It provides a platform to store and transfer the data to a cloud server. Also, TTN provides a console that is the management application of The Things Stack for LoRaWAN. It is a web application that can be used to register applications such as the gateway.

5.3. Application Programming Interface

The developed Application Programming Interface (API) used ubidots, an IoT platform, to prototype and scale projects to production. The ubidots were configured using the Representational State Transfer (REST) API, which displays the data from the sensors stored on theTTN server on a newly designed dashboard, either in text or a graphicalform.

VI. IMPLEMENTATION OF FUSION ALGORITHM

Data fusion algorithms can be used in many tracking and surveillance systems and other applications where reliability is considered the primary concern. One solution for the designed systems is to deploy several sensor nodes with multiple connections and fuse the data received from sensor nodes [25]. The data to be fused can be obtained from a set of sources. Still, the main concentration should be on the data from sensors, such as radar, that can localize the different sources of energy being sensed [26]. Data fusion requires sensor networks from the increased numbers of sensor devices available to collect data, resulting in a high incoming data rate. Data fusion is working as effective management of data to maximize its usefulness.

The data fusion product is considered a picture, and it is carried on a tactical scale and can be provided as a tactical picture. The process of data fusion can occur at different levels of abstraction [6]. Data fusion is

divided into different levels- level one data fusion, sometimes called object assessment, is the tracking and classification of individual objects [27]. Level two data fusion also referred to as a situation assessment effectively finds out the understanding of an object's behaviour. Level three data fusion is known as impact assessment and predicts the effects of planned actions. Level four data fusion istherefinementof the datafusionprocessitself[27].

These doubts require the sensor data fusion algorithms to develop to combine information coherently and synergistically to generate robust, accurate data for the environment. Figure. 2 shows the general mechanism diagram for data fusion. The sensor nodes S_1, S_2, and S_3 collect data D_1, D_2, D_3 from the sensors connected with the LoRa module. Most of the situations depend on the condition of the LoRa sensor nodes; D_1, D_2, and D_3 might not be the required and actualvalues. The sensor will generate many useless data packets generated and transmitted to the LoRa sensor nodes. If the LoRa sensor node does not recognize that the data is fused, it does not further process the data. When the LoRa sensor node received the fused data, data will be transmitted to the LoRa gateway. The proposed algorithm is designed for the LoRa module witha wired connection with three different sensors shown in Figure. 3. LoRa module is collecting data from the temperature and humiditysensor,CO_2sensoranddustsensor.

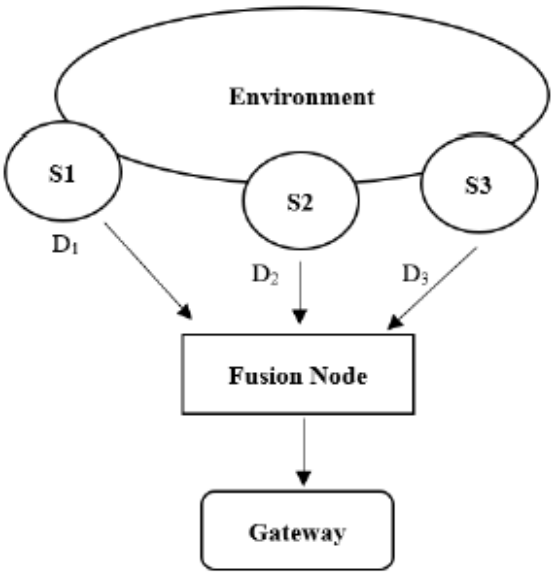


Figure. 2. Fusion Mechanism

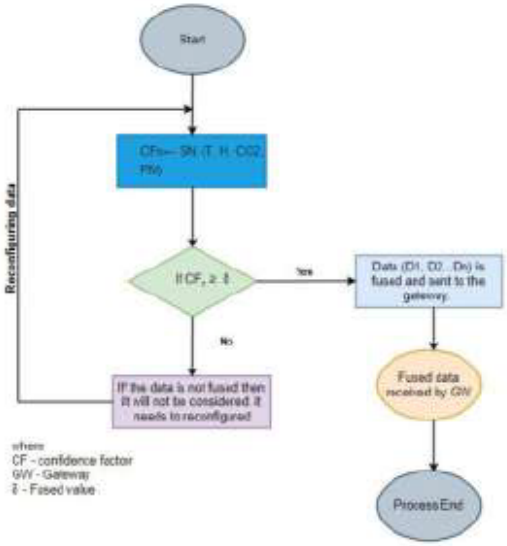
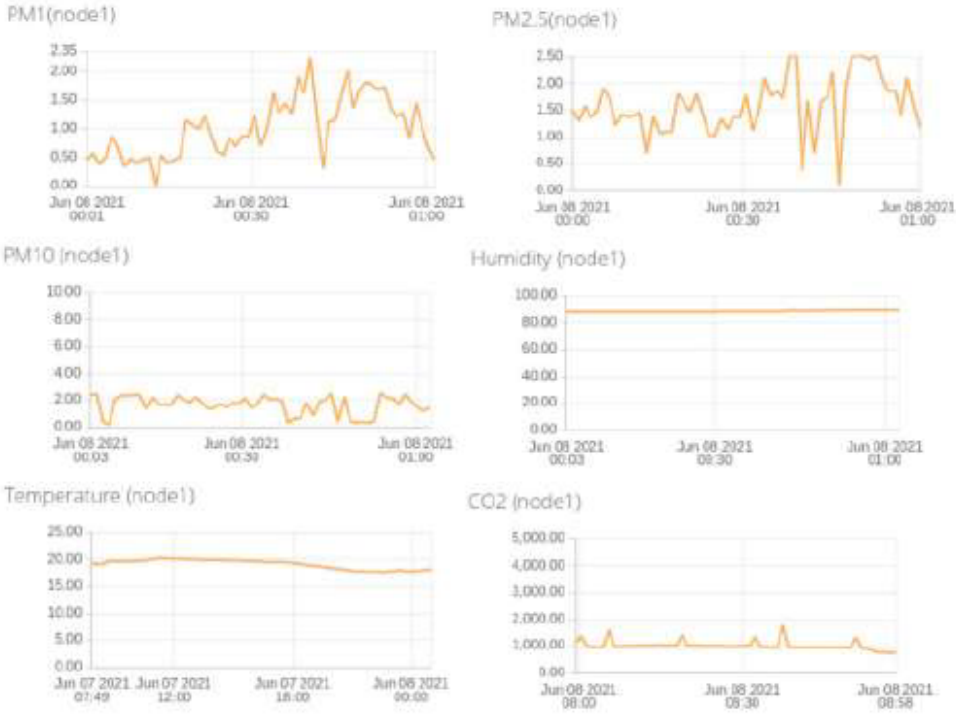


Figure. 3.Flowchart for Data Fusion Algorithm

In the algorithm, the algorithm's inputs are node temperature, humidity, carbon dioxide and particulate matter. The network's output is the fused data, known as accurate data,sent by sensors connected to the LoRa modules.

The purpose of the sensor nodes (SN) is to find the level of data collected according to the conditions of the sensor nodes. Each sensor node collects data and calculates the confidence factor (CF) using SN for every packet collected. SN considers the non-operating temperature and humidity, carbon dioxide (CO_2), particle matter (PM) ranges to create the membership functions. SN produces confidence factor (CF_1, CF_2,...,CF_n)foreachsensordata(D_1,D_2,D_n).Ifthe data is not fused, sensor nodes will not receive it, and data will be disregarded.



VII. RESULTS AND DISCUSSION

Preliminary testing and data collection are presented in this section for the LoRa network. The LoRa node with the sensors was placed atasetlocationatAucklandUniversityofTechnology.The results obtained from one node show the concentration levels of pollutant levels after every minute to check the pollutant level in the air: some of the results for the particulate matter (PM), carbon dioxide (CO_2), temperature and humidity are shown in Figure. 4. Results were taken every minute to check the pollutant levels. But in the figures, data is shown in terms of six-hour intervals.

VIII. CONCLUSIONS

The developed air pollution monitoring system to use Arduino IDE, IoT technology, is proposed to improve air quality. Furthermore, the TTGO LoRa module with pollutant sensors to allow low power consumption in the proposed network designed. the network system worked based on sleep function, which allows the devices for data collection over a long time. Different sensors were used to monitor the air's pollutant level is the main objective of this paper. Pollutant levels were monitored using different time intervals after implementing the real-time. TTN server receives the data from the sensor nodes implemented and sent to the ubidots considered online. From where it can be download and reuse data for further analysis and experiments. The research will continue to improve both hardware and software systems to determine the accurate values of pollutants by using calibration methods on every sensor.

REFERENCE

[1] Brienza, S., et al., A Low-Cost Sensing System for Cooperative Air Quality Monitoring in Urban Areas. Sensors, 2015. 15(6): p. 12242.

[2] Choi, S., et al., Micro sensor node for air pollutant monitoring: Hardware and software issues. Sensors, 2009. 9(10): p. 7970-7987.
[3] Karapistoli, E., I. Mampentzidou, and A.A. Economides, Environmental monitoring based on the wireless sensor networking technology: A survey of real-world applications. International Journal of Agricultural and Environmental Information Systems, 2014. 5(4): p. 1-39.
[4] Cohen, A.J., et al., Estimates and 25-year trends of the global burden of disease attributable to ambient air pollution: an analysis of data from the Global Burden of Diseases Study 2015. The Lancet, 2017. 389(10082): p. 1907-1918.
[5] Organization, W.H., Ambient air pollution: A global assessment of exposure and burden of disease. 2016.
[6] Hall, D.L. and J. Llinas, An introduction to multisensor data fusion. Proceedings of the IEEE, 1997. 85(1): p. 6-23.
[7] Yin, Z., et al. An ultra-short term load forecasting method based on improved human comfort index. in 4th International Conference on Electrical and Electronic Engineering (ICEEE). 2017.
[8] Tzortzakis, K., K. Papafotis, and P.P. Sotiriadis. Wireless-Self Powered Environmental Monitoring System for Smart Cities Based on LoRa. in Panhellenic Conference on Electronics and Telecommunications (PACET). 2017.
[9] Cordova-Lopez, L.E., et al., Urban emissions monitoring at the vehicle level, in Lecture Notes in Electrical Engineering. 2012. p. 249-268.
[10] Khedo, K., P. Rajiv, and M. Avinash, A Wireless Sensor Network Air Pollution Monitoring System. Vol. 2. 2010.
[11] Batzias, F.A. and C.G. Siontorou, Measuring Uncertainty in Lichen Biomonitoring of Atmospheric Pollution. IEEE Transactions on Instrumentation and Measurement, 2009. 58(9): p. 3207-3220.
[12] Suthaputchakun, C., Z. Sun, and M. Dianati, Applications of Vehicular Communications for Reducing Fuel Consumption and CO Emission: The State of the Art and Research Challenges. IEEE Communications Magazine, 2012. 50(12): p. 108-115.
[13] Girish, S.V., et al., A network model of GUI-based implementation of sensor node for indoor air quality monitoring, in Advances in Intelligent Systems and Computing. 2016. p. 209-217.
[14] Suriano, D., et al., A portable sensor system for air pollution monitoring and malodours olfactometric control. Lecture Notes in Electrical Engineering. Vol. 109 LNEE. 2012. 87-92.
[15] Yu, T.C., et al., Wireless Sensor Networks for Indoor Air Quality Monitoring. Medical Engineering and Physics, 2013. 35(2): p. 231-235.
[16] Alassi, A., et al., Development of innovative indoor/outdoor air quality monitoring for environmental impact assessment in the State of Qatar. WIT Transactions on Ecology and the Environment, 2014. 183: p. 103-115.
[17] Abraham, S. and X. Li, A Cost-effective Wireless Sensor Network System for Indoor Air Quality Monitoring Applications. Procedia Computer Science, 2014. 34: p. 165-171.
[18] Zheng, K., et al., Design and Implementation of LPWA-Based Air Quality Monitoring System. IEEE Access, 2016. 4: p. 3238-3245.
[19] Peng, C., K. Qian, and C. Wang, Design and Application of a VOC-Monitoring System Based on a ZigBee Wireless Sensor Network. IEEE Sensors Journal, 2015. 15(4): p. 2255-2268.
[20] Rohde, R.A. and R.A. Muller, Air Pollution in China: Mapping of Concentrations and Sources. PLOS ONE, 2015. 10(8): p. e0135749.
[21] LoRa Modem Datasheet. 2006.
[22] DHT22 Datasheet. 2017.
[23] Compact Optical Dust Sensor Datasheet. 2006.
[24] Infrared CO2 Sensor Datasheet. 2020.
[25] Vershinin, Y. A data fusion algorithm for multisensor systems. in Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002.(IEEE Cat. No. 02EX5997). 2002. IEEE.
[26] Peters, D.J., A practical guide to level one data fusion algorithms. 2001: National Defence, Defence R & D Canada, Defence Research Establishment Atlantic.
[27] Steinberg, A.N. and C.L. Bowman, Revisions to the JDL data fusion model. Handbook of multisensor data fusion. 2017: CRC press. 65-88.

★★★

LoRa Air Pollution Monitoring Network System

Amritpal Kaur, Jeff Kilby
amritpal.kaur@aut.ac.nz, jeffrey.kilby@aut.ac.nz

Problem

Due to the urbanization and rapid growth in transportation, Pollution levels goes up and causes some health issues such as respiratory disease. Human and environment both affecting with indoor and outdoor pollution. The common pollutant level is Particulate Matter (PM). Different sizes of PM are shown in Fig. 1. Most dangerous pollutant size is $PM_{2.5}$ [2].

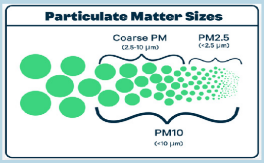


Fig. 1 Particulate Matter Distribution

Basic Concepts

Number of techniques are in trends for monitoring pollution, but the latest Technology is LoRa. LoRa stands for Long-Range. LoRa is a wireless modulation technique derived from Chirp Spread Spectrum (CSS) technology [1]. It encodes information on radio waves using chirp pulses similar to the way dolphins and bats communicate! LoRa modulated transmission is robust against disturbances and can be received across great distances. CSS in LoRa shown below in Fig. 2.

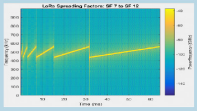


Fig. 2 Spreading Factor

Network Design

Network is designed to work on long distance communication of sensor nodes with gateway. in the network architecture, LoRa Dragino LPS gateway is used to communicate based on the frequency bands (915MHz). Three different sensors transmit the values over the network to monitor air pollutants in the air shown in Fig. 3. The most common and dangerous pollutant monitored is $PM_{2.5}$ that can affect respiratory system.

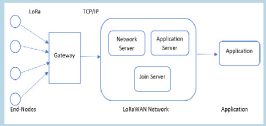


Fig. 3 Network Design

References

[1] Mroue, H., Nasser, A., Hamrioui, S., Parrein, B., Motta-Cruz, E. and Rouyer, G., 2018, April. MAC layer-based evaluation of IoT technologies: LoRa, SigFox and NB-IoT. In 2018 IEEE Middle East and North Africa Communications Conference (MENACOMM) (pp. 1-5). IEEE.

[2] Polichetti, G., Cocco, S., Spinali, A., Trimarco, V. and Nunziata, A., 2009. Effects of particulate matter (PM10, PM2. 5 and PM1) on the cardiovascular system. Toxicology, 261(1-2), pp.1-8.

Working Principle of LoRa

LoRa works by moving an RF tone around over time in a very linear way. This graph shows the chirps in a reverse waterfall—the newest data is at the top, which is called an "up chirp." You can see how this frequency of the tone is increasing over time [1] . LoRa transmissions work by chirping, breaking the chips in different places in terms of time and frequency in order to encode a symbol. The fact that LoRa transmissions jump from one place to another at a particular time might mean one bit string vs. another. It's not just simply binary—it has a lot of information you can convey (high symbol depth). Working of LoRa network shown in Fig. 4.

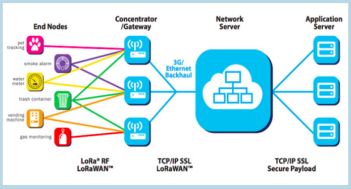


Fig. 4 LoRa Network

Findings and Discussion

For Finding out the results, two nodes were deployed at afternoon and evening time. The below images shows the of Node 1 and Node two. These sensor nodes were monitoring PM levels in the air at afternoon and evening times. It shows some fluctuations during the monitoring period as shown in Fig. 5-8. PM levels in afternoon and evening are changes and it depends on the air and environment.

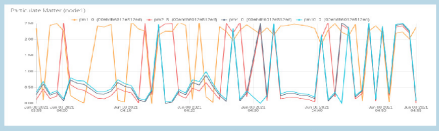


Fig. 5 Node 1 (Afternoon)

Fig. 6 Node 1 (Evening)

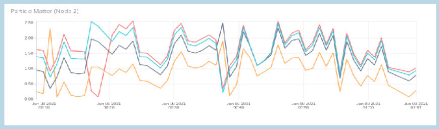


Fig. 7 Node 2 (Afternoon)

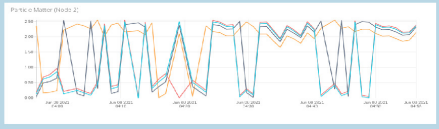


Fig. 8 Node 2 (Evening)

Conclusion and Future Work

It is concluded from the above discussion that Particulate matter have been affecting human life and environment. To know more about the pollutant levels in the air, it requires to monitor. Different technologies have been invented, but the LoRa networks are more efficient than others. LoRa Technology has prove the long distance communication in the network while transmitting the data. As the technology has CSS, that they selected automatically according to the distance and packet size. Lowest spreading factor shows the better communication if the LoRa technology is using highest spreading factor, it means poor communication. The test experiment was taken to know about pollutant levels in afternoon and evening times. Both sensor nodes were deployed in afternoon and evening times. Node 1 and Node 2 shows the fluctuations in the PM levels as per changes in the environment. This test is not finished yet because it requires to done more testing to find reason of sudden values down of in both nodes.

Development of a LoRa Network for Monitoring Particulate Matter

Amritpal Kaur and Jeff Kilby

Abstract Air pollution in cities has become an important topic due to its adverse effects on humans and air quality. The aim of this paper is to monitor particles less than 1 μm (PM_1), 2.5 μm ($PM_{2.5}$), 4 μm (PM_4), and 10 μm (PM_{10}). The concentration of particulate matter highly changed with location and time. Particulate matter in the air is considered as the primary pollutant, and it affects the environment band the risk of mortality and morbidity of respiratory disease. To address this issue, the research presents the design and development of the low-cost network for monitoring particulate matter using sensiron sensor (SPS30). These devices are equipped with LoRaWAN to test the low-power wide-area network coverage. The designed network contains the sensors connected to the ESP32 microcontroller towards the processing of LoRa modules (sensor nodes), which send data to the gateway using the frequency band, using the 'The Things Network (TTN)'. The sensor collects the different particle matter in the air. The proposed network design has been implemented at St. Paul Street Auckland. The designed network system allows the users to access the online dashboard to test and monitor the concentration levels of particulate matter in the air.

Keywords Air quality indexing · LoRa · Particulate matter

1 Introduction

With the increase in urbanization and industrialization, it decreases the air quality parameters. This is important to know about the air quality parameters using different methods and techniques. The government has developed some policies and procedures for monitoring air quality [1]. When the researchers are using the tra-

A. Kaur (✉) · J. Kilby
Auckland University of Technology, Auckland, New Zealand
e-mail: amritpal.kaur@aut.ac.nz
URL: https://www.aut.ac.nz

J. Kilby
e-mail: jeffrey.kilby@aut.ac.nz

S. Smys et al. (eds.), *Computer Networks and Inventive Communication Technologies*,
Lecture Notes on Data Engineering and Communications Technologies 141,
https://doi.org/10.1007/978-981-19-3035-5_24

Table 1 Major air pollutants

Parameter	Sources	Causes
CO and CO_2	Vehicle emissions	Respiratory problems
SO_2	3 industrial smoke, forest fires	Skin and eye irritation
NO_2	Fossil fuels vehicle emission	Asthma
Particle matter (PM)	Dust storms domestic coal burning and municipal waste	Premature deaths respiratory problems
Lead	Petrol and industries	Brain and organ development in children

ditional methods, these are not quite feasible if using the wireless sensor networks (WSNs) in large areas. There are some new techniques developed in WSNs to use low-cost sensors to collect and analyse the real-time air pollution monitoring. WSNs can be used for pollution monitoring with accuracy, calibration techniques, and fault tolerance features in topologies [1]. In the sensor network, it includes the number of sensor nodes that are interconnected with wireless network [2]. The rapid development in industrialization and population increase has led to the rise in pollution. In poor air quality, it includes the harmful gases and particles such as ozone, particle matter, nitrogen dioxide, sulphur dioxide, and carbon monoxide [3]. Table 1 shows the major air pollutants [4].

In the World Health Organization (WHO) [5], air pollution is considered as the bigger environmental issue that causes some health problems and responsible of 9 deaths every year. Industrial and scientific innovations have been developed to provide solution based on the conventional wireless sensor networks (WSNs) [2] for air pollution monitoring system. Air pollution occurs cause of toxic gases, dust, fuel or chemicals integrate with the environment. These toxic gases and particles are harmful to the environment and health. These substances that cause pollution are called pollutants [3]. Over the few decades, several types of research have been done to monitor air pollution. In air quality, it includes different portable particle matter and gas sensor available. The solution to these problems, numerous low-cost air pollution monitoring systems were developed [4].

Numerous studies have shown the pollutant's exposure to the air. But the particulate matter is considered the most dangerous pollutant that causes cardiovascular diseases and other health issues [6–9]. Particulate matters are made up of solid and liquid particles, which enter the atmosphere naturally or with any human activities [10]. The pollutant level of PM pollution can easily assess by continuous monitoring of the level PM_{10} (size less than 10 μm), $PM_{2.5}$ (size less than 2.5 μm), PM_4 (size less than 4 μm), and PM_1 (size less than 1 μm).

Table 2 Communication technologies

Parameter name	Technologies		
	LoRa	Sigfox	NB-IoT
Band (MHz)	868/915	868/915	868/915
PHY	CSS	UNB	NB
Spreading factor	2^7–2^{12}	–	–
Bandwidth (kHz)	500–125	1	180
Data rate (kbps)	27–0.37	0.1	250–226.7
Range (km)	22	63	35

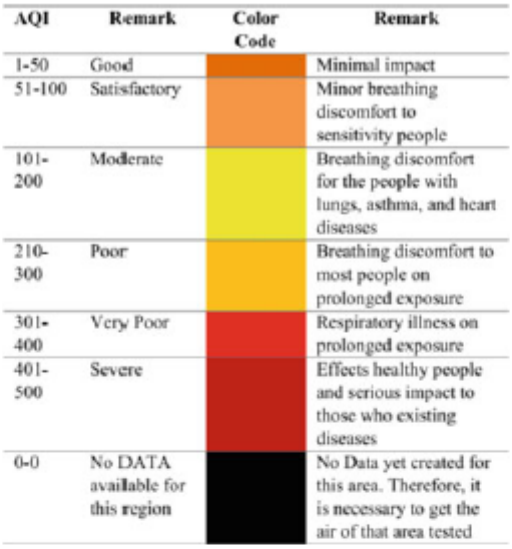
The network system was designed using different wireless communication technologies [4], such as LoRa, Sigfox and NB-IoT. In-depth analysis and understanding of pollution monitoring are an essential to develop the new methodology for pollution monitoring. The newly designed network can only be possible using the accurate monitoring system that helps the network study the pollutant levels in the air [4]. Different communication technologies are compared in Table 2 [5].

2 Air Quality Index (AQI)

AQI shows how the pollutant levels in the air in a particular area. AQI can be calculated using some consideration of pollutants such as carbon monoxide (CO), sulphur dioxide (SO_2), particulate matter (PM) [4]. This is an indicator of air quality, based on various air pollutants that affect human health and the environment. Many countries use air quality indexing, and different nations have their own policies to calculate pollutant levels [11] and as shown in Fig. 1 [12]. AQIs take the pollutant measurements from the air using the air pollution monitoring system and convert these values to a single figure. The nature and scale of the figure are flexible. The scale in AQI ranges from 'safe' to 'hazardous'. This classification can be described to guide health effects and awareness about environmental issues and the breakdown of the particulate matter in the air as given in Table 3.

3 Related Work

Zheng et al. [13] developed the dynamic filter system for outdoor pollution monitoring system for $PM_{2.5}$. The network design network is implemented indoor and outdoor for the testing $PM_{2.5}$ and explains the interrelationship between indoor and outdoor. First, the indoor $PM_{2.5}$ per hour studied and applied Gaussian distribution.



AQI	Remark	Color Code	Remark
1-50	Good		Minimal impact
51-100	Satisfactory		Minor breathing discomfort to sensitivity people
101-200	Moderate		Breathing discomfort for the people with lungs, asthma, and heart diseases
210-300	Poor		Breathing discomfort to most people on prolonged exposure
301-400	Very Poor		Respiratory illness on prolonged exposure
401-500	Severe		Effects healthy people and serious impact to those who existing diseases
0-0	No DATA available for this region		No Data yet created for this area. Therefore, it is necessary to get the air of that area tested

Fig. 1 Air quality indexing

Table 3 Air quality indexing

AQI	Remark	Cause	$PM_{2.5}$
1–50	Good	Minimal impact	0–30
51–100	Satisfactory	Minor breathing issues	31–60
101–200	Moderate	Asthma and heart disease	61–90
210–300	Poor	Prolonged exposure	91–120
301–400	Very poor	Respiratory illness	121–50
401–500	Severe	Serious impact	250+

It performs the statistical distribution of the indoor and outdoor $PM_{2.5}$ pollution level and applied some physical laws and collects data.

Tzai et al. [14] implemented the road network to monitor air pollution, and its affects on humans and environment. In this research, sensor nodes were deployed on crossroads. This paper explains the WSN framework to monitor pollutant levels for the street-level spatial–temporal changes of carbon monoxide. The authors proposed the urban air quality monitoring system based on WSNs that integrate with global system for mobile communication (GSM). The network system is deployed on roads for monitoring carbon monoxide (CO) concentration caused by vehicle emissions.

Chen et al. [15] proposed an open framework for $PM_{2.5}$ monitoring in smart cities. With an increase in population and human activities affects the pollution level in cities. The designed network deployed 2500 sensor nodes in Taiwan and 29 other countries. This paper works to resolve the issues of inaccuracies in the low-cost sensor

networks used for monitoring of particles. Facilitating the network deployment is to investigate the accuracy issue of low-cost particle sensor.

Badura et al. [16] evaluated the low-cost sensors for $PM_{2.5}$ monitoring. This paper presents the results of four models set using different sensors connected to different location under same temperature. The data collection of $PM_{2.5}$ was performed for almost half year. After the research and finding, results show the large dispersion of high relative errors in $PM_{2.5}$.

Guili et al. [17] physically based data analysis model was developed for estimation of concentration levels of $PM_{2.5}$. The research used the fine mode aerosol optical depth at 440, 550 and 675 mm, fine particle radius, humidity and boundary layer height data from 2015 to 2016. The data collected in 2015 was used for calculating the integrated extinction efficiencies based on the network model. The authors concluded the developed method has potential for retrieval of $PM_{2.5}$ using AERONET AOD in Beijing.

Jelicic et al. [18] designed the network system to perform reliable gas monitoring and operating on batteries for one year. Information about the pollutants extracted from the PIR sensors and gas sensors. The system should be able to predict the behaviour, adapt the duty cycle and send the alarm message if necessary. To monitor the reliability of the network, simulations were performed.

Agarwal et al. [19] proposed for the detection and monitoring of the pollution using wireless sensor networks, where the framework has five parts: deployment of sensor node, clustering of nodes, inter-cluster routing protocol and cluster-head selection algorithm. Sensor nodes were deployed in triangular fashion to avoid collision, minimum transmission, detecting pollution, etc. An efficient clustering algorithm was proposed to select the best path.

Knoll et al. [20] presented the denser sensor network to expand the air pollution measurement density, and data is fed into the dispersion models which only provides the approximate results. New network technologies were compared such as LoRa, Sigfox and NB-IoT. Denser sensor network is directly able to monitor air pollution values. The measurements were carried out in the city of Graz, for these measurements an STM32 Nucleo board with an expansion board, featuring LoRa SX1272 transceiver was utilized. Payload for each message contained 16 bytes including a message counter for verification. The sink node was positioned on the third floor outside the window at the university campus, and further nodes for the data collection were placed at the height of 1 m and measurements performed using different technologies. LoRa and Sigfox bring along the required attributes including long-range and small power consumption, where NB-IoT is the latest technology that is strongly driven by the mobile communication.

From the literature review, it shows some limitation of using the existing network systems. To overcome the limitation such as low-power consumption and long-range communication, new network design should be implemented.

4 Network Design and Setup

In this section, focus is on the network design and setup. The purpose of developing sensor network design is to monitor particulate matter and their ranges in the specified area. This network design is working based on software and hardware setups. Network design is shown in Fig. 2, where the LN_1, LN_2, LN_3 and LN_4 are considered as LoRa nodes on which three sensors S_1 (CO_2), S_2 (PM sensor) and S_3 (dust and temperature) placed outside. Before the network deployment, the network design was simulated to analyse the network performance.

4.1 Hardware Required

SPS30 The SPS30 particulate matter (PM) sensor is compact, high quality, optical particle sensor that uses the laser scattering and innovation contamination resistance technology to achieve particle matter measurement [21].

For the research and monitoring of the pollutant level, a previously dust sensor was used instead of SPS30. Due to the inaccuracies and only the dust level monitoring qualities, the sensor was replaced with SPS30. The sensor has inbuilt high-quality and long-lasting component that enables the precision of measurements and provides the throughput for ten years. The sensor package is shown in Fig. 3 [22]. The SPS30 sensor has a five-pin interface that can communicate with two different interfaces:

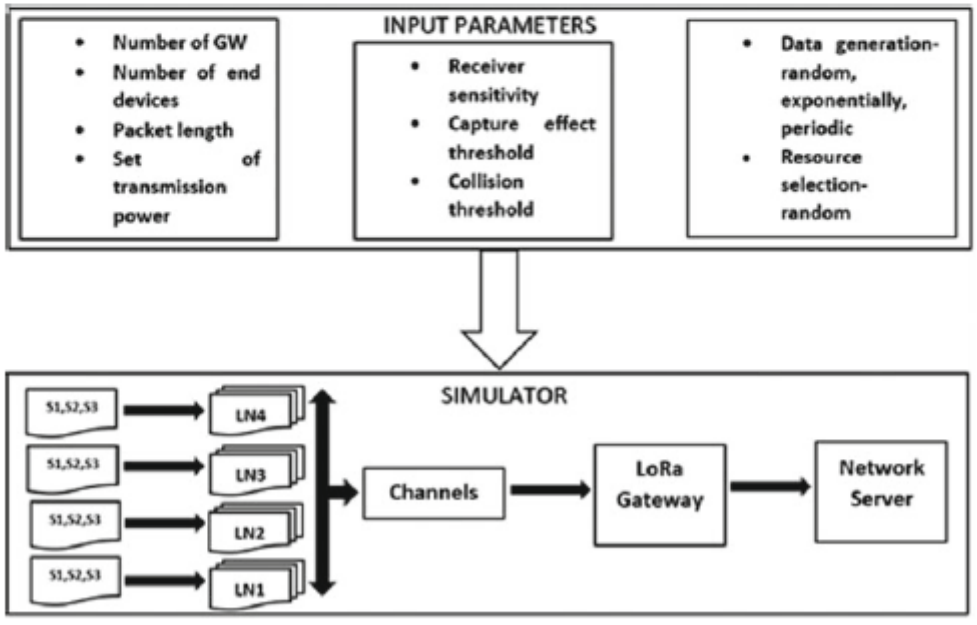


Fig. 2 Block diagram

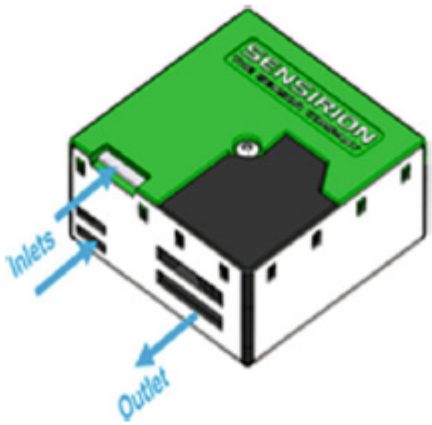


Fig. 3 SPS30 sensor for measuring particulate matter

UART and I2C. The designed network is to monitor the particulate matter. The sensor is working based on UART as per the requirements of the TTGO LoRa32 SX1276 OLED Board.

LoRa32 SX1276 OLED Board The TTGO LoRa32 SX1276 OLED board was found by Semtech, example of LoRa device. LoRa32 board is working for long-range radio devices for transmitting and receiving of data packets over the network. This device leads to less interference and signal loss [21]. Many countries are assigned with specific LoRa bands. LoRa boards have a high communication range as compared to the other networks such as cellular networks. LoRa devices can be used the public and private sectors. LoRa device is shown in Fig. 4.

Fig. 4 TTGO LoRa32 SX1276 OLED

4.2 Software Required

The software for the desired network outcomes is comprised of two parts: the software program coding for the microcontroller board and for the graphical output it requires.

TTN Server The Things Network is the provider for the network server in this research, that means it provides the interface between the gateway and the application. The things network is not providing only the service but also other tools, and these are explained in this section. Before the data transmission to the application, gateway starts receiving messages from the sensor nodes. Similarly, when an application wants to send message to the sensor nodes, it requires to select the broadcast message for gateway. This is the network server, similar to LoRa and LoraWAN allows the sensor nodes to communicate with the gateway. The network server is responsible for routing the data between devices and application. LoRa alliance provided some specifications for the back-end interfaces, for example, controlling the MAC layer, end-point authentication in the network. There is no official release for the LoraWAN network. There are number of solutions available for the LoRaWAN network server such as LoRaServer, open-source network server, ResIOT, and The Things Network developed by The Things Industries. The Things Network is the V2 console for the application deployment and is not accepting any new devices. This V2 console is upgraded with the V3 console.

Ubidots Ubidots is the cloud server that is considered as the webhook for storing and analysing data online. During the transmission, sensor nodes receive data from sensors and received data is shown on the ubidots's dashboard. The collected data can be used for further researches and analysis. The developed application programming interface (API) used ubidots, an IoT platform, and scale projects to production. The ubidots was configured using the representational state transfer (REST) API, which is used to display the data from the sensors stored on the TTN server on a newly designed dashboard, either in text or a graphical form.

5 Results and Discussion

Two LoRa sensor nodes were used to collect data from PM sensors. The pilot study was performed to monitor the PM factors in the air. Four LoRa nodes were deployed with the difference on 5 m apart from gateway. LoRa gateway has the ability to communicate with long-distance transmission. This test was just performed to test the communication range of LoRa and measuring the PM levels in the air. Gateway is registered with free TTN server cloud to which it transmits the collected data from the sensor nodes. The results from one LoRa node (LN_1).

The readings from all LoRa nodes were collected after every 10 min for continuous three days. The dataset collected is plotted for PM_1, $PM_{2.5}$, PM_4 and PM_{10}, shown in Figs. 5, 6, 7 and 8. The comparison among these sensor nodes showed the

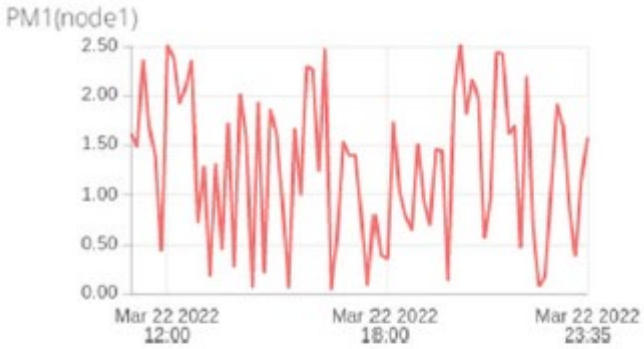


Fig. 5 PM_1 measurements

Fig. 6 $PM_{2.5}$ measurements

Fig. 7 PM_4 measurements

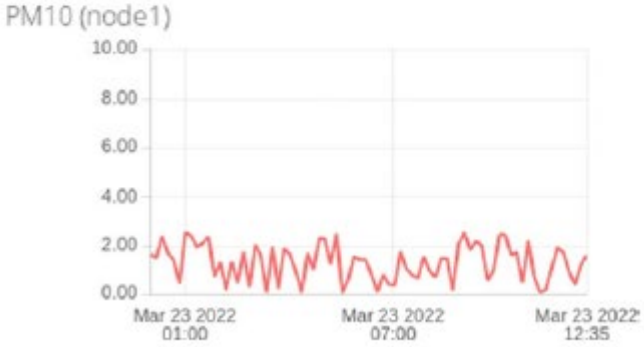


Fig. 8 PM_{10} measurements

excellent correlation between LoRa nodes for measuring the concentration levels. The differences in spread between PM_4, PM_{10} and $PM_{2.5}$ where the PM_1 shows the different concentration levels of both nodes. After few intervals of time, results were taken after 30 min and show the average concentration levels with accuracy of sensors. Concentration levels in the indoor air were not exceeded their limit suggested by the sensors connected.

Particulate measurements were obtained from the SPS30 (Sensiron Sensor) that shows the PM readings of the DustTrak at the sampling level of 1 min. The readings obtained from the SPS30 sensor show some fluctuations. Particulate matter readings are not accurate in this test experiment but these can be improved and shows the good relationship with actual concentration by using the diversity of calibration methods. High variability in PM concentration levels is shown in PM_1. During the monitoring period, measured concentration levels were changed.

6 Conclusion

In summary, number of sensors available in market for monitoring particulate matter and no sensor is ideal for all applications. There are numerous characteristics of SPS30 sensor performance, and choice of this sensor for the network can be based on the critical characteristics. For example, in the network design, the measurements of (PM_1), ($PM_{2.5}$), (PM_4) and (PM_{10}) were required. To monitor all these with low-power consumption, SPS30 sensor was selected that monitor all PM values with single unit. Software and hardware problems have been occurred such as communication range, node placement and time, power management. With the selected devices, 95% time of the transmission was saved using the LoRaWAN network. Different pollutant levels monitor using the time intervals of 1 min. The only down times were due to an outage of The Things Network. Since the sensor nodes were transmitting data every 46 s (the payload of each pollutant level is only 4 bytes). This designed network will be tested with multiple LoRa nodes to check the network performance and accuracy.

References

1. World Health Organization (2016) Ambient air pollution: a global assessment of exposure and burden of disease. Report 2(5)
2. Guanochanga B, Cachipuendo R, Fuertes W, Salvador S, Benítez D, Toulkeridis T, Meneses F (2018) Real-time air pollution monitoring systems using wireless sensor networks connected in a cloud-computing, wrapped up web services. In: Proceedings of the future technologies conference, Nov 2018. Springer, Cham, pp 171–184
3. Bhowmik T, Bhattacharya A, Banerjee I (2019) A low-cost air pollution monitoring system using ZigBee-based wireless sensor networks. In: Information and communication technology for intelligent systems. Springer, Singapore, pp 71–82

4. Rane MS, Naik AR, Vachhani K (2018) Real-time AQI monitoring system: an economical approach using wireless sensor network. In: 2018 9th international conference on computing, communication and networking technologies (ICCCNT), July 2018. IEEE, pp 1–6
5. Mroue H, Nasser A, Hamrioui S, Parrein B, Motta-Cruz E, Rouyer G (2018) MAC layer-based evaluation of IoT technologies: LoRa, SigFox and NB-IoT. In: 2018 IEEE Middle East and North Africa communications conference (MENACOMM), Apr 2018. IEEE, pp 1–5
6. Polichetti G, Cocco S, Spinali A, Trimarco V, Nunziata A (2009) Effects of particulate matter (PM10, PM2.5 and PM1) on the cardiovascular system. Toxicology 261(1–2):1–8
7. Ballester F, Rodriguez P, Iniguez C, Saez M, Daponte A, Galan I, Toro S (2006) Air pollution and cardiovascular admissions association in Spain: results within the EMECAS project. J Epidemiol Community Health 60(4):328–336
8. Zanobetti A, Schwartz, J (2005) The effect of particulate air pollution on emergency admissions for myocardial infarction: a multicity case-crossover analysis. Environ Health Perspect 113(8):978–982
9. Maheswaran R, Haining RP, Brindley P, Law J, Pearson T, Fryers PR, Campbell MJ (2005) Outdoor air pollution and stroke in Sheffield, United Kingdom: a small-area level geographical study. Stroke 36(2):239–243
10. Rodrıguez S, Querol X, Alastuey A, Viana MM, Alarcon M, Mantilla E, Ruiz CR (2004) Comparative PM10-PM2.5 source contribution study at rural, urban and industrial sites during PM episodes in Eastern Spain. Sci Total Environ 328(1–3):95–113
11. Auckland Council (2014) An air quality index for Auckland: review of some international techniques and example data
12. Dhingra S, Madda RB, Gandomi AH, Patan R, Daneshmand M (2019) Internet of Things mobile-air pollution monitoring system (IoT-Mobair). IEEE Internet of Things J 6(3):5577–5584
13. Zheng H, Xiong K, Fan P, Zhong Z (2019) Data analysis on outdoor-indoor air quality variation: buildings' producing dynamic filter effects. IEEE Syst J 13(4):4386–4397
14. Liu JH, Chen YF, Lin TS, Chen CP, Chen PT, Wen TH, Jiang JA (2012) An air quality monitoring system for urban areas based on the technology of wireless sensor networks. Int J Smart Sens Intell Syst 5(1)
15. Chen LJ, Ho YH, Lee HC, Wu HC, Liu HM, Hsieh HH, Lung SCC (2017) An open framework for participatory PM2.5 monitoring in smart cities. IEEE Access 5:14441–14454
16. Badura M, Batog P, Drzeniecka-Osiadacz A, Modzel P (2018) Evaluation of low-cost sensors for ambient PM2.5 monitoring. J Sens 2018
17. Chen G, Guang J, Xue Y, Li Y, Che Y, Gong S (2018) A physically based $PM_{2.5}$ estimation method using AERONET data in Beijing area. IEEE J Sel Top Appl Earth Obs Remote Sens 11(6):1957–1965
18. Jelicic V, Magno M, Brunelli D, Paci G, Benini L (2012) Context-adaptive multimodal wireless sensor network for energy-efficient gas monitoring. IEEE Sens J 13(1):328–338
19. Agarwal L, Dixit G, Jain AK, Pandey KK, Khare A (2016) Energy efficient pollution monitoring system using deterministic wireless sensor networks. In: Proceedings of the international congress on information and communication technology. Springer, Singapore, pp 301–309
20. Knoll M, Breitegger P, Bergmann A (2018) Low-power wide-area technologies as building block for smart sensors in air quality measurements. e i Elektrotech Inf 135(6):416–422
21. Sparkfun (2020) Particulate matter sensor
22. Sensirion (2020) Particulate matter sensor for air quality monitoring and control

Paper 6 – Springer Book

Wireless Sensor Networks (WSNs) in Air Pollution Monitoring: A Review

Amritpal Kaur and Jeff Kilby

Abstract Air pollution is the major concern in urban areas due to the impacts of air pollution on health and environment. A number of studies reveal the importance to be aware of air pollution and air pollution monitoring systems. This paper is a review article based on different methods implemented by the researchers to know about the concentration levels of particles and gases in air. The paper focuses on the different networks that are used for the pollution monitoring system and how they work effectively.

Keywords Wireless sensor networks · Fine particles · PM2.5 · sensors · Air pollution · Particulate matters

1 Introduction

Air pollution can be defined as the introduction of the chemical particles and toxic gases into the air that can affect the human health and environment [1]. Indoor air pollution has more risk than compared to the outdoor pollution, as mentioned in air quality report by World Health Organization (WHO) [2]. The fuel or cigarette smoke may lead to the health issues such as asthma, tuberculosis, and allergies. Air pollution monitoring is an important method to know the pollutant levels in the air as these can be dangerous to human beings, animals, and environment. Wireless sensor networks (WSNs) have been developed and improved for the air pollution monitoring systems [3].

For air pollution monitoring system, it requires the combination of two methods or systems: sensor network control system and air pollution monitoring system. The sensor networks contain a number of sensor units to be deployed to know more about

A. Kaur (✉) · J. Kilby
Auckland University of Technology, Auckland, New Zealand
e-mail: amritpal.kaur@aut.ac.nz
URL: https://www.aut.ac.nz/

J. Kilby
e-mail: jeffrey.kilby@aut.ac.nz

G. Rajakumar et al. (eds.), *Intelligent Communication Technologies and Virtual Mobile Networks*, Lecture Notes on Data Engineering and Communications Technologies 131,
https://doi.org/10.1007/978-981-19-1844-5_59

the pollutant levels. Pollutants in the air can be particles, toxic gases, and some other factors such as pressure, temperature, and humidity. [3].

This paper focuses on the detailed methods used for air pollution monitoring using wireless sensor networks. WSNs are vast these days, as these can be used in different applications to work on. The next section is based on the literature survey performed by different researchers from 2005–2019 to review the different pollution monitoring methods. This section has five subsections, as the literature review is conducted on different network systems used for pollution monitoring and prediction methods. The last section binds up with the conclusion based on the review and suggests the better technologies to monitor air pollution.

2 Literature Review

The literature review is performed to study the information relevant to air pollution monitoring using WSNs by locating the different databases such as Institute of Electrical and Electronics and Engineers (IEEE) and Scopus, using the following key terms namely, air pollution, smog, toxic gases, particulate matter, pollutants, air quality, and PM_{10}, $PM_{2.5}$.

After the screening of several papers, some of them have been used to make a review article about air pollution monitoring systems. The research is done based on the IEEE journals, conference papers, and articles as shown in Table 1. After

Table 1 Yearly distribution from 2008 to 2019 of the articles published relevant to air pollution monitoring using wireless sensor networks

Year	Journals books conferences transactions articles access IoT						
2005	2	–	–	4	–	2	–
2006	1	–	–	4	–	–	–
2007	1	–	–	2	–	–	–
2008	2	–	–	1	–	–	–
2009	1	–	–	2	–	–	–
2010	2	1	–	3	–	–	–
2011	–	–	–	–	–	–	–
2012	6	–	–	–	2	–	–
2013	3	–	2	1	3	–	–
2014	3	–	1	1	1	–	–
2015	4	–	–	1	1	–	–
2016	7	–	5	–	1	–	–
2017	2	–	3	4	1	3	2
2018	2	–	4	2	1	1	2
2019	–	–	3	–	–	–	–

reviewing the diversity of paper, few simulations done on the network system have been observed. The literature review includes different sections of the air pollution monitoring systems based on the following points, developed or implemented by researchers:

1. Simulation and network deployment
2. Network deployment
3. Cellular network deployment
4. Vehicular network deployment
5. Neural network deployment.

2.1 Simulation and Network Deployment

Karapistoli et al. [4] investigated the air pollution monitoring application using WSNs. The paper proved the real deployment and labs to be specific and important tool for researchers for exploring the environmental phenomena. The paper categorized the deployment areas such as agricultural, environmental, and pollution monitoring in air and water [4]. The paper covered different network topologies but main focused on mesh topology. Mesh topology was implemented for organizing the clusters to decrease the power consumption [4].

Kang et al. [5] developed the remote sensing system for monitoring the emission of on-road vehicles by implementing the sensors onto the vehicles. For implementing this system, effective location strategy had been designed. The paper formulated the issues related to the small number of subsets of roads and the traffic emission that can be monitored easily. Authors solved the issue by transforming it into a graph-theoretic problem and considered the different characteristics like traffic regulations and limits [5]. The first step was to design hyper graph-based set of circuits using the depth strategy. The approximation algorithm was implemented to find the greedy transversal to cover all the traffic circuits. These methodologies quantify the influencing factors such as geographical position and congestion situations of the traffic.

Yi et al. [6] conducted the survey for three different categories of networks such as static sensor networks (SSNs), community sensor networks (CSNs), and vehicle sensor networks (VSNs), to mitigate the impacts of air pollution on human health, environment, and economy, and to determine pollutant levels in the air using the conventional air pollution methods. Stationery monitors were deployed in Hong Kong. Conventional methods have some limitations such as large size, heavy weight, and extraordinary expenses. To resolve these issues in the network, the authors implemented 'The Next generation' Air Pollution Monitoring System (TNGAPMS) [6].

Khedo et al. [7] developed the air pollution monitoring network system named as wireless sensor network air pollution monitoring system (WAPMS). WAPMS network system was deployed in Mauritius with several air pollution monitoring

sensing units. The network was simulated in Java, in simulation time Jist and Scalable Wireless Ad hoc Network Simulator (SWANS). The method was used for the conversion of an existing virtual machine simulation platform embedded with code level. Deployment strategy of the desired network, shown in figure 1, shows where the network was simulated in small region as a prototype and extended to the whole island. The network performed simulation with single sink node and simplified the network using gateway. The system was based on air quality indexing (AQI) that is known as an indicator of air pollutants in the environment that can affect the human life and environment. Researcher's motivation was to develop the indexing system to categorize air pollution. Level of the AQI values showed the concerns in human health and environment.

Liu et al. [8] proposed the real-time air pollution monitoring system based on the LoRa technology. LoRa technology has low-power consumption feature as compared to other network technologies.

The network system included the microcontroller chip and pollution sensors for monitoring pollutants such as NO_2, SO_2, O_3, CO, PM_1, $PM_{2.5}$, and PM_{10}, as shown in Fig. 2. LoRa module transmitted the data or packets to the sensing unit, and then, the data was saved on the cloud server. LabVIEW was used for performing different functionalities as a GUI software. GUI was used to test the communication between the USB port and LoRa module by sending commands. Network system was successful and monitored the accurate values of gas concentration and PM concentrations.

Jelicic. V et al. [9] designed the network architecture shown in Fig. 3, for an indoor pollution monitoring working based on WSNs. The network system also worked on the issues such as power consumption. During the network deployment, 36 nodes were deployed on first floor of the building. Diversity of sensor nodes was used to monitor the temperature, humidity, CO, VOC, and particulate matters. This was also based on the static sensor networks.

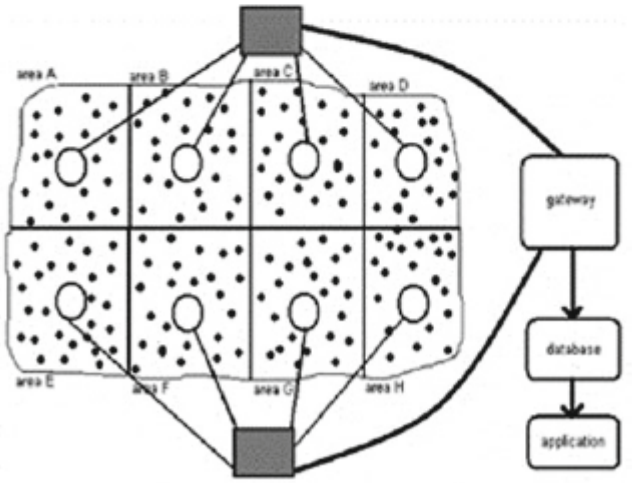


Fig. 1 Nodes deployment strategy [7]

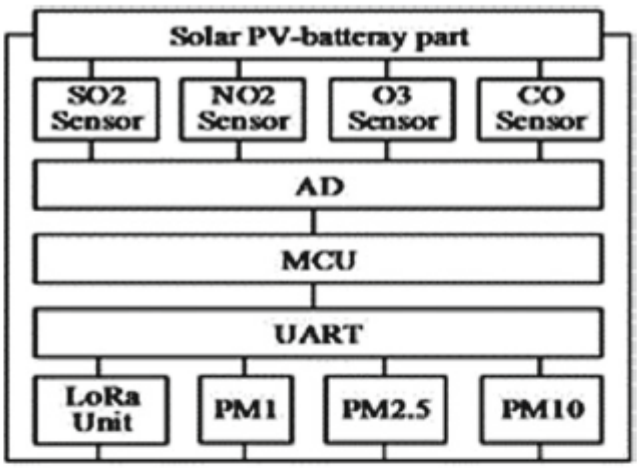


Fig. 2 Hardware architecture based on LoRa technology [8]

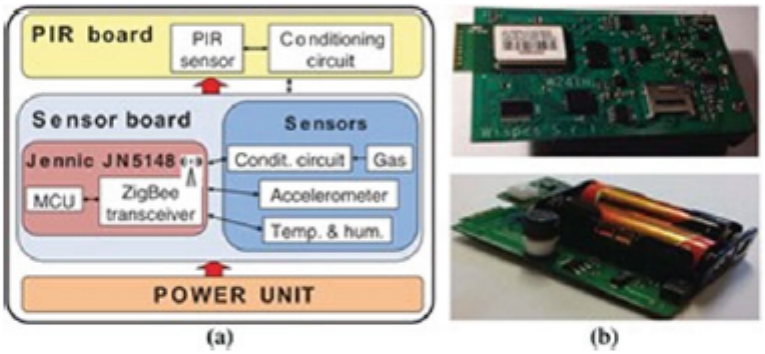


Fig. 3 Sensor node used for gas monitoring. **a** Block architecture (Left), **b** top side and bottom side (Right) [9]

Rahim et al. [10] proposed an alternative data gathering strategy for the collection of air pollutant data based on LoRa and LoRaWAN protocols and simulated the network in ns-3 simulator. System architecture is shown in Fig. 4. LoRa and LoRaWAN protocols allowed several devices in the network for the collection of pollutant levels and sent the collected data to the cloud server, and this did not affect the performance of the network. The network used the class A static battery to supply power to the devices, and the network was split into the monitoring period as time windows equal to subregions of the regions and gateway sleep duration. The network was simulated in ns-3 simulator with different radius such as 3000m and 6000m. Simulation time was fixed as 600 seconds in the region and 100 seconds in subregion. According to the results, the worst results were taken from the area within 3000m, and end devices used the SF7 that affected the performance of network. If the spreading factor was low, then its performance was high, and with high SF, the network showed low performance.

Folea et. al. [11] implemented the battery powered system to check the concentration level of CO_2, temperature, humidity, pressure, and intensity in an indoor

Fig. 4 System architecture based on LoRa and LoRaWAN protocols [10]

environment. All the sensor units were placed in the indoor environment and read the pollutant levels. For the transmission of data packets, the network used the UDP protocol, and ambient wireless sensor shown in figure 5 was used to monitor the pollutant levels [11].

Rajasegara et al. [12] proposed the system to improve the PM concentration using the existed high precision. The results revealed that accuracies in estimated particulate matter were better in the higher densities in low-precision sensor nodes. The sensor stations validate the model using the data collected by simulating the network in the nodes. It showed the usage of time series at environmental protection. The researchers Web site EPAV sites as a basis for interpolation and extrapolation over the spatial domain [12].

Fig. 5 Ambient wireless sensor [11]

2.2 Network Deployment

Arularasi et al. [13] implemented a methodology for outdoor air pollution monitoring with a task of transferring the embedded information without finding the gases emitted from automobile exhausts and controlled using the gas sensor array with connectivity of ZigBee wireless technology [13]. The paper proposed an image-based steganography that used least significant bits technique, pseudo-random coding technique, and partial optimization technique. The network set up was for monitoring the dust, temperature, humidity, and ZigBee module with Wi-Fi or Bluetooth technology [13].

Mansour et al. [14] presented the network architecture shown in Fig. 6, for the outdoor pollution monitoring in urban areas using WSNs. The system was effective to measure the amount of O_3, CO, and NO_3 in the environment. Libelium sensor nodes were used for the sensor network. For the communication process, ZigBee communication was used for data retrieval with different gas sensors used in the system generated. All this was possible with the mobile applications and emails for the retrieval or capturing of data. For the improvement of the efficiency of the network and topologies, clustering protocol of air sensor was used that was effective for the improvements in the communication rate and energy consumption of the network.

Mendez et al. [15] designed the new technology for the outdoor air pollution monitoring by using the community sensor networks known as participatory sensing approach. It collected the information about CO, CO_2, VOCs, H_2, temperature, and humidity in the environment. Pollution data collected was sent to the server with

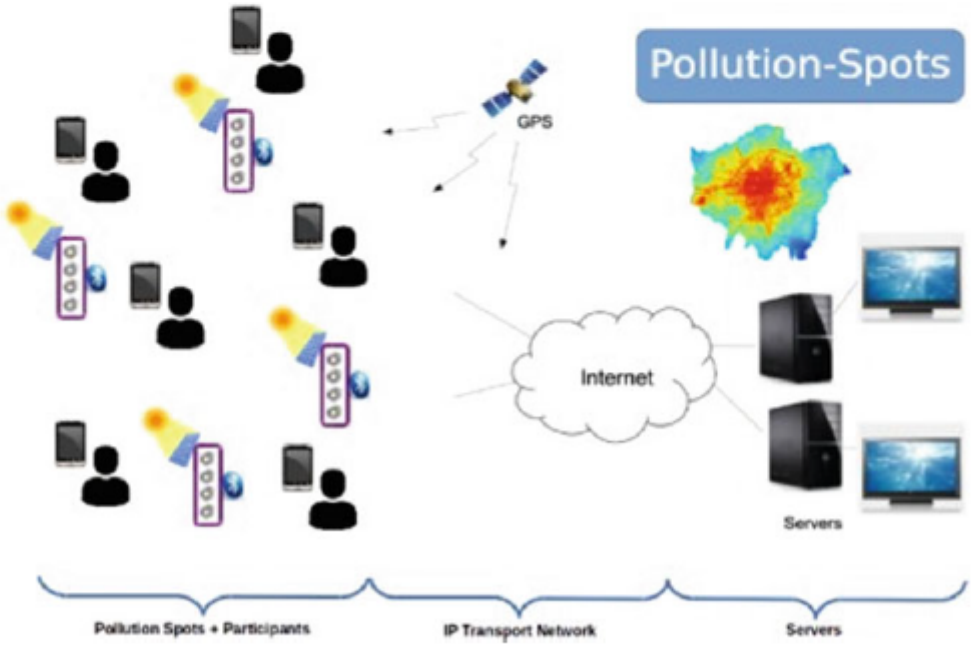


Fig. 6 General architecture of pollution-spot system [14]

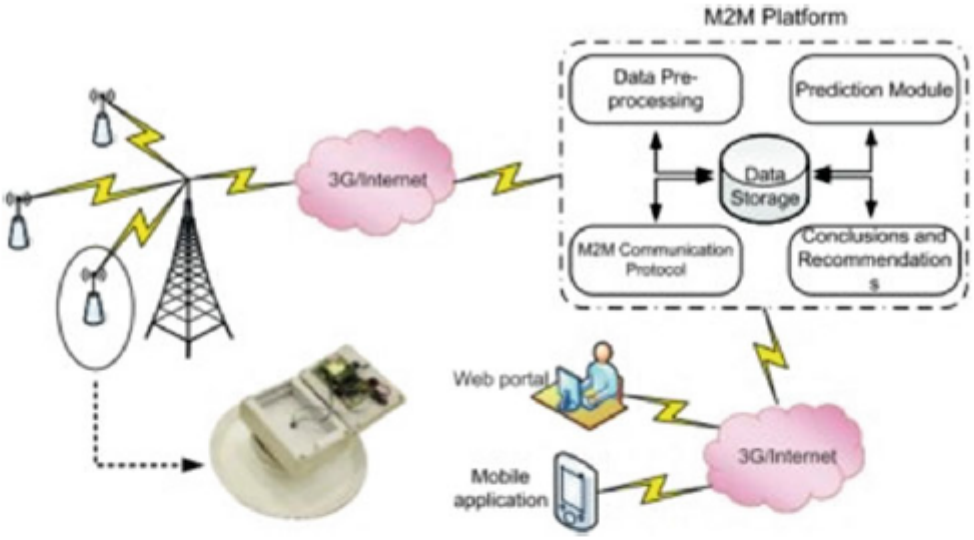


Fig. 7 Air quality monitoring system architecture [16]

smart phones that were connected through the Bluetooth. When the information was collected, it was sent to the server and faced some practical issues in the deployment of sensor nodes.

Elias Yaacoub et al. [16] showed the representation of WSNs system as shown in Fig. 7 for the monitoring of air pollution in Doha, Qatar. It included the multi-gas monitoring stations used with the M2M communication. For the deployment of the nodes, smart network used two methods such as the filtering and data processing tasks in the server. For the data collection, the network system used the software system and open-air package. During the project implementation, real measurement data was also measured.

Gyu-Sik Kim et al. [17] represented the network structure to measure the pollutant levels of the humidity, temperature, PM_{10}, and CO_2. This project was implemented in the Seoul Metro and Metropolitan Rapid Transit Corporation. PM measuring instrument measured the accuracy and precision of the PM_{10} concentrations in the air quality. For this measurement, author used the linear regression technique.

2.3 Cellular Networks

Liu et al. [18] proposed network for the wireless sensor networks GSM mobile services called micro-scaled air quality monitoring system for monitoring the air pollution in urban areas as shown in Fig. 8. The network system proposed was implemented in Taipei City to collect the pollutant level of CO with vehicle emissions and high-resolution meteorological data [18]. A real-time proposed system was implemented in the real environment. Mics-5525 sensor was used for monitoring the

Fig. 8 Micro-scaled air quality monitoring system architecture [18]

pollutant level of CO. Sensor nodes worked based on bridge module and wireless communication modules [18].

Hu et al. [19] designed and evaluated the network system using the low-cost sensing system that includes the sensor units, smart phones, cloud computing, and the mobile applications called as HazeWatch, and the system structure is shown in Fig. 9. For visualizing and estimating the air pollutant level by sensor units, the mobile application based on the mobility patterns was found useful.

Fig. 9 HazeWatch system architecture [19]

Chen et al. [20] developed the hybrid sensor network structure for the VOC monitoring. The aim of the network was to lower down the limitations of photo ionization detection device approach, real-time detection approach and portable gas chromatography approach [20]. This approach was effective than of other approaches to overcome the issues in the real-time detection devices. The detection devices consisted of sampling collection and pre-concentration unit and sensors for monitoring VOC [20]. The hybrid device sensor was combined with pre-concentration and tuningbased detection principles into single wireless device.

2.4 *Vehicular Networks*

Ngom et al. [21] developed the real-time WSNs system to monitor air pollution in Dakar. The network system shown in Fig. 10 included the CO sensors, CO_2 sensors, PM_{10}, $PM_{2.5}$, and PM_1 sensors. The acquisition architecture included the detection unit, power supply unit, location unit, processing unit, and transmission unit. The pollution measurement kit can be fixed as landmark or embedded in a car in motion. The dataset can be exploited by different applications.

Blaschke et al. [22] established the flap-control system to control the gases from vehicle emission with the help of micro-electrochemical oxide gas sensors that makes the mass market applications. Different events are included in the study such as

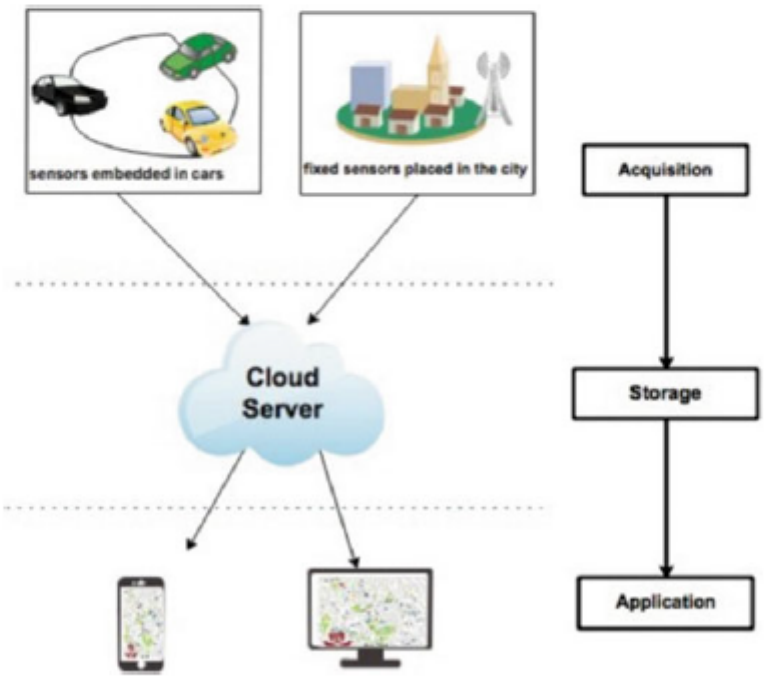


Fig. 10 Global system architecture [21]

cigarette smoke, food odour, and bioeffluents. The network system had metal-oxide sensor array connected with three different sensors. Authors used the data evaluation approach combined with human sensory data and MEMS sensor data. It developed two independent algorithms. The results are achieved with varying background changes and independent of flow changes [22].

Ahmed et al. [23] developed the complementary metal-oxide semiconductor (CMOS) architecture with micro-nano scale WSN for monitoring, protection, and controlling air pollution [23]. These can be available from the communication technologies and CMOS levels of architecture. If the nanotechnology is successful, then for environmental applications, it is evitable that communication in the WSNs can be better as compared to wired communication with high security and integrity [23].

2.5 Neural Networks

Octavin A Postolache et al. [25] presented the network structure for the indoor and outdoor air pollution monitoring. For the implementation of the network, the sensor nodes were implemented over the rooks, and the tixdioxide sensor arrays were used [25]. The experiment was performed with wireless network and with wired network. The network monitored the temperature and humidity with embedded Web server and neural network. The paper compared the classical polynomial modelling and neural network modelling for calculating polynomial coefficients [25]. Neural networks processing has the drawback of large number of multiplication and usage of nonlinear functions. During implementation of the network architecture, three different tasks such as sensing nodes data reading, air pollution event detection, and data logging and publishing were carried out [25]. The network was advantageous for providing extended capabilities for air pollution conditions, good accuracy on gas concentration, and read and write functions can be implemented in lab view [25].

Hobbs et al. [25] proposed the dis-aggregating emissions projections to a scale compatible with different air quality simulation models [25]. The system implemented three models that site new power plants consistent with historical patterns. The network dis-aggregated the NOx emissions [25]. Haiku methods were used for the calculations for the electricity demand, electricity prices, and electricity supply in 21 regions and monitored the emission of NOx, SO_2, mercury, and CO_2. The network system worked based on four time periods super-peak, peak, shoulder, and baseload hours [25]. The method requires to be improved because it was tough for predictions and forecasting the specific location and its environment, because of the future policy, technology, economic, and environmental conditions [25].

Bashir et al. [26] implemented low-cost air pollution monitoring method to monitor the air quality using motes connected with gas and meteorological sensors. Authors implemented the three machine learning algorithms (SVR, M5P model trees, and ANN) to develop the one step model for measuring pollutant levels of ozone, nitrogen dioxide, sulphur dioxide, and two types of modelling were used such as univariate and multivariate [26], and it can be effective for accurate forecasting. In

Fig. 11 AirBox device: **a** overview and **b** internal components [27]

the architecture, artificial neural networks got the worst results due to the poor generalization ability to work with the measurements collected. It can be improved for the future work by considering the data changes over time for real-time forecasting and can be included to increase data seasonality [26].

Ling-Jyh Chen et al. [27] developed the paper for concentration of $PM_{2.5}$ sensing system with the major consideration of air pollution in urban areas and degrading the air quality. Large-scale $PM_{2.5}$ sensing system was deployed in different overseas cities. There was a biggest challenge to ensure the data quality. The authors developed the anomaly detection framework (ADF). ADF can be used for the identification of outliers in the measurement of raw data and anomaly events [27]. ADF had four modules using the neural networks. Neural network was used for the large-scale air pollution monitoring in some cities. Figure 11 shows the AirBox used for the monitoring of $PM_{2.5}$, and the data for the identification of properties of datasets was collected by AirBox devices from 1 October 2016, to October 2016. The ADF module was highly extensible for supporting the large-scale environment sensing system [27].

3 Conclusion

This review paper is designed to give a detailed literature survey of air pollution monitoring techniques coupled with the evidence of the potential applications such as mobile sensing and vehicular sensing. Based on the assessment of air pollution using diversity of methods, it can be said that new technologies are better than traditional methods. For monitoring of air pollution, researchers have applied methods designed by low-cost sensors and algorithms. The content of this paper focuses on the air pollution monitoring due to urbanization and industrialization. Researchers

developed the real-time networks and simulated them for better and accurate monitoring of concentration levels of $PM_{2.5}$ and PM_{10}. According to researchers, $PM_{2.5}$ should be less than 2.5 μm and PM_{10} should be less than 10 μm. These particles affect human health and environment as well.

Data collection was done by using the air quality indexing, and the network was developed on roads to monitor the pollution by traffic and transportation since these are the biggest cause of the air pollution.

Low-cost sensors used for monitoring CO_2 emissions and particles ($PM_{2.5}$ and PM_{10}) are the information source on air quality and appropriate for air pollution monitoring.

The aim of this paper is to discuss the development of air pollution monitoring methods using WSNs and methods for getting accurate pollutants levels of gases and particles in air. Vehicular air pollution monitoring can be known as advanced networks for knowing more about the pollutants in the air. The performance of the vehicular networks can be monitored using the IoT technology, and data collection should be done using server for the future analysis.

References

1. Ghorani-Azam A, Riahi-Zanjani B, Balali-Mood M (2016) Effects of air pollution on human health and practical measures for prevention in Iran. J Res Med Sci: Official J Isfahan Univ Med Sci 21
2. Zhang J, Smith KR (2003) Indoor air pollution: a global health concern. Br Med Bull 68(1):209–225
3. Haider HT, See OH, Elmenreich W (2016) A review of residential demand response of smart grid. Renew Sustain Energy Rev 59:166–178
4. Karapistoli E, Mampentzidou I, Economides AA (2014) Environmental monitoring based on the wireless sensor networking technology: a survey of real-world applications. Int J Agricult Environ Inf Syst (IJAEIS) 5(4):1–39
5. Kang Y, Li Z, Zhao Y, Qin J, Song W (2017) A novel location strategy for minimizing monitors in vehicle emission remote sensing system. IEEE Trans Syst Man Cybernetics: Syst 48(4):500–510
6. Yi WY, Lo KM, Mak T, Leung KS, Leung Y, Meng ML (2015) A survey of wireless sensor network based air pollution monitoring systems. Sensors 15(12):31392–31427
7. Khedo KK, Perseedoss R, Mungur A (2010) A wireless sensor network air pollution monitoring system. arXiv preprint arXiv:1005.1737
8. Liu S, Xia C, Zhao Z (2016) A low-power real-time air quality moni- toring system using LPWAN based on LoRa. In 2016 13th IEEE international conference on solid-state and integrated circuit technology (ICSICT), IEEE, pp 379–381
9. Jelicic V, Magno M, Brunelli D, Paci G, Benini L (2012) Context-adaptive multimodal wireless sensor network for energy-efficient gas monitoring. IEEE Sensors J 13(1):328–338
10. Rahim H, Ghazel C, Saidane LA (2018) An alternative data gathering of the air pollutants in the urban environment using lora and lorawan. In 2018 14th international wireless communications mobile computing conference (IWCMC), IEEE, pp. 1237–1242
11. Folea SC, Mois G (2014) A low-power wireless sensor for online ambient monitoring. IEEE Sens J 15(2):742–749

12. Rajasegarar S, Havens TC, Karunasekera S, Leckie C, Bezdek JC, Jamriska M, Palaniswami M et al (2013) High-resolution monitoring of atmospheric pollutants using a system of low-cost sensors. IEEE Trans Geosci Remote Sensing 52(7):3823–3832
13. Arularasi A, Divya S, Meena K, Vasuki JT (2015) Pollution monitoring and controlling cause by automobile exhaust gases using zigbee technology. Int J Engg Res Sci Tech 240
14. Kianisadr M, Ghaderpoori M, Jafari A, Karami M (2018) Zoning of air quality index (PM10 and PM2. 5) by Arc-GIS for Khorramabad city, Iran. Data in brief, 1131–1141.
15. Mendez D, Diaz S, Kraemer R (2016) Wireless technologies for pollution monitoring in large cities and rural areas. In 2016 24th telecommunications forum (TELFOR), IEEE, pp 1–6
16. Kadri A, Yaacoub E, Mushtaha M, Abu-Dayya A (2013) Wireless sensor network for real-time air pollution monitoring. In 2013 1st international conference on communications, signal processing, and their applications (ICC- SPA), IEEE, pp. 1–5
17. Kim GS, Son YS, Lee JH, Kim IW, Kim JC, Oh JT, Kim H (2016) Air pollution monitoring and control system for subway stations using environmental sensors. J Sensors
18. Liu JH, Chen YF, Lin TS, Chen CP, Chen PT, Wen TH, Jiang JA (2012) An air quality monitoring system for urban areas based on the technology of wireless sensor networks. Int J Smart Sens Intell Syst 5(1)
19. Hu K, Sivaraman V, Luxan BG, Rahman A (2016) Design and evaluation of a metropolitan air pollution sensing system. IEEE Sens J 16(5):1448–1459
20. Chen C, Tsow F, Campbell KD, Iglesias R, Forzani E, Tao N (2013) A wireless hybrid chemical sensor for detection of environmental volatile organic compounds. IEEE Sens J 13(5):1748–1755
21. Ngom B, Seye MR, Diallo M, Gueye B, Drame MS (2018) A hybrid measurement kit for real-time air quality monitoring across senegal cities. In 2018 1st international conference on smart cities and communities (SCCIC), IEEE, pp. 1–6
22. Blaschke M, Tille T, Robertson P, Mair S, Weimar U, Ulmer H (2006) MEMS gas-sensor array for monitoring the perceived car-cabin air quality. IEEE Sens J 6(5):1298–1308
23. Mahfuz MU, Ahmed K (2005) A review of micro-nano-scale wireless sensor networks for environmental protection: prospects and challenges. Sci Technol Adv Mater 6(3–4):302–306
24. Postolache OA, Pereira JD, Girao PS (2009) Smart sensors network for air quality monitoring applications. IEEE Trans Instrument Meas 58(9):3253–3262
25. Hobbs BF, Hu MC, Chen Y, Ellis JH, Paul A, Burtraw D, Palmer KL (2010) From regions to stacks: spatial and temporal downscaling of power pollution scenarios. IEEE Trans Power Syst 25(2):1179–1189
26. Shaban KB, Kadri A, Rezk E (2016) Urban air pollution monitoring system with forecasting models. IEEE Sens J 16(8):2598–2606
27. Chen LJ, Ho YH, Hsieh HH, Huang ST, Lee HC, Mahajan S (2017) ADF: an anomaly detection framework for large-scale PM2. 5 sensing systems. IEEE Internet of Things J 5(2):559–570

References

[1] M. I. Abd-El-Barr, M. A. M. Youssef, and M. M. Al-Otaibi, "Wireless sensor networks - part I: topology and design issues," in *Canadian Conference on Electrical and Computer Engineering, 2005.*, 2005, pp. 1165-1168, doi: 10.1109/CCECE.2005.1557184.

[2] B. Bathiya, S. Srivastava, and B. Mishra, "Air pollution monitoring using wireless sensor network," presented at the 2016 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), 19-21 Dec. 2016, 2016.

[3] M. S. Manshahia, "A Firefly Based Energy Efficient Routing in Wireless Sensor Networks," *African Journal of Computing & ICT,* vol. 8, no. 4, 2015.

[4] T. Singla and M. Manshahia, "Wireless Sensor Networks for Pollution Monitoring and Control," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology,* vol. 2, pp. 663-670, 2017.

[5] C. Y. Chong and S. P. Kumar, ""“Sensor networks: evolution, opportunities, and challenges,”", 2003, vol. 91, pp. 1247–1256.

[6] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks,* vol. 38, no. 4, pp. 393-422, 2002.

[7] M. A. Berlin and S. Muthusundari, *Ad hoc and Sensor Networks*. 2016. Technical Publications.

[8] Y. J. Jung, Y. K. Lee, D. G. Lee, K. H. Ryu, and S. Nittel, "Air Pollution Monitoring System based on Geosensor Network," in *IGARSS 2008 - 2008 IEEE International Geoscience and Remote Sensing Symposium*, 7-11 July 2008 2008, vol. 3, pp. III - 1370-III - 1373, doi: 10.1109/IGARSS.2008.4779615. [Online]. Available: https://ieeexplore.ieee.org/ielx5/4757194/4779256/04779615.pdf?tp=&arnumber=4779615&isnumber=4779256

[9] J. Jo, B. Jo, J. Kim, S. Kim, and W. Han, "Development of an IoT-Based Indoor Air Quality Monitoring Platform," *Journal of Sensors,* vol. 2020, p. 8749764, 2020/01/14 2020, doi: 10.1155/2020/8749764.

[10] A. Lavric and V. Popa, "Internet of things and LoRa™ low-power wide-area networks: a survey," in *2017 International Symposium on Signals, Circuits and Systems (ISSCS)*, 2017: IEEE, pp. 1-5.

[11] M. Saari, A. M. bin Baharudin, P. Sillberg, S. Hyrynsalmi, and W. Yan, "LoRa—A survey of recent research trends," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2018: IEEE, pp. 0872-0877.

[12] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things journal,* vol. 1, no. 1, pp. 22-32, 2014.

[13] D. Croce, M. Gucciardo, S. Mangione, G. Santaromita, and I. Tinnirello, "Impact of LoRa imperfect orthogonality: Analysis of link-level performance," *IEEE Communications Letters,* vol. 22, no. 4, pp. 796-799, 2018.

[14] I. Daramouskas, V. Kapoulas, and T. Pegiazis, "A survey of methods for location estimation on Low Power Wide Area Networks," in *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*, 2019: IEEE, pp. 1-4.

[15] J. Finnegan and S. Brown, "A comparative survey of LPWA networking," *arXiv preprint arXiv:1802.04222,* 2018.

[16] A. Carlsson, I. Kuzminykh, R. Franksson, and A. Liljegren, "Measuring a LoRa network: Performance, possibilities and limitations," in *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*: Springer, 2018, pp. 116-128.

[17] O. Khutsoane, B. Isong, and A. M. Abu-Mahfouz, "IoT devices and applications based on LoRa/LoRaWAN," in *IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017: IEEE, pp. 6107-6112.
[18] A. J. Wixted, P. Kinnaird, H. Larijani, A. Tait, A. Ahmadinia, and N. Strachan, "Evaluation of LoRa and LoRaWAN for wireless sensor networks," in *2016 IEEE SENSORS*, 2016: IEEE, pp. 1-3.
[19] C. I. Davidson, R. Phalen, and P. Solomon, *Airborne Particulate Matter and Human Health: A Review*. 2005, pp. 737-749.
[20] C. Peng, K. Qian, and C. Wang, "Design and Application of a VOC-Monitoring System Based on a ZigBee Wireless Sensor Network," *IEEE Sensors Journal,* vol. 15, no. 4, pp. 2255-2268, 2015, doi: 10.1109/JSEN.2014.2374156.
[21] "Where Does Air Pollution Come From?" https://www.nps.gov/subjects/air/sources.htm (accessed.
[22] (2018). *New Zealand's environmental reporting series: Our air 2018*. [Online] Available: https://www.stats.govt.nz/information-releases/new-zealands-environmental-reporting-series-our-air-2018
[23] S. Fuzzi *et al.*, "Particulate matter, air quality and climate: lessons learned and future needs," *Atmospheric chemistry and physics,* vol. 15, no. 14, pp. 8217-8299, 2015.
[24] L. Mainetti, L. Patrono, and A. Vilei, "Evolution of wireless sensor networks towards the internet of things: A survey," in *SoftCOM 2011, 19th international conference on software, telecommunications and computer networks*, 2011: IEEE, pp. 1-6.
[25] Q. Wang and I. Balasingham, "Wireless Sensor Networks–An Introduction, Wireless Sensor Networks: Application-Centric Design, Yen Kheng Tan (Ed.), ISBN: 978-953-307-321-7, InTech," ed, 2010.
[26] S. K. Dash, S. Mohapatra, and P. K. Pattnaik, "A survey on applications of wireless sensor network using cloud computing," *International Journal of Computer Science & Emerging Technologies,* vol. 1, no. 4, pp. 50-55, 2010.
[27] D. Fowler *et al.*, "A chronology of global air quality," *Philosophical Transactions of the Royal Society A,* vol. 378, no. 2183, p. 20190314, 2020.
[28] B. Graham and H. Narsey, "Air pollution monitoring in New Zealand, 1960-1992," *Report no. MESC,* vol. 94, p. 27, 1994.
[29] A. Ghorani-Azam, B. Riahi-Zanjani, and M. Balali-Mood, "Effects of air pollution on human health and practical measures for prevention in Iran," (in eng), *Journal of research in medical sciences : the official journal of Isfahan University of Medical Sciences,* vol. 21, pp. 65-65, 2016, doi: 10.4103/1735-1995.189646.
[30] D. S. M. Dr. M. A. Berlin, *Ad hoc and Sensor Networks*. 2016. Technical Publications.
[31] M. Loxham and M. J. Nieuwenhuijsen, "Health effects of particulate matter air pollution in underground railway systems–a critical review of the evidence," *Particle and fibre toxicology,* vol. 16, no. 1, pp. 1-24, 2019.
[32] A. Shrestha and L. Xing, "A Performance Comparison of Different Topologies for Wireless Sensor Networks," in *2007 IEEE Conference on Technologies for Homeland Security*, 16-17 May 2007 2007, pp. 280-285, doi: 10.1109/THS.2007.370059.
[33] Q. Mamun, "A qualitative comparison of different logical topologies for Wireless Sensor Networks," (in eng), *Sensors (Basel),* vol. 12, no. 11, pp. 14887-14913, 2012, doi: 10.3390/s121114887.
[34] K. Khedo, P. Rajiv, and M. Avinash, *A Wireless Sensor Network Air Pollution Monitoring System*. 2010.
[35] Liu *et al.*, "An Air Quality Monitoring System for Urban Area Based on the Technology of Wireless Sensor Networks," *International Journal on Smart Sensing*

and Intelligent Systems, Article vol. 5, no. 1, pp. 191-214, 2012, doi: 10.21307/ijssis-2017-477.

[36] Y. Kang, Z. Li, Y. Zhao, J. Qin, and W. Song, "A Novel Location Strategy for Minimizing Monitors in Vehicle Emission Remote Sensing System," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 48, no. 4, pp. 500-510, 2018, doi: 10.1109/TSMC.2016.2607221.

[37] V. Jelicic, M. Magno, D. Brunelli, G. Paci, and L. Benini, "Context-adaptive multimodal wireless sensor network for energy-efficient gas monitoring," *IEEE Sensors journal,* vol. 13, no. 1, pp. 328-338, 2012.

[38] L. Agarwal, G. Dixit, A. K. Jain, K. K. Pandey, and A. Khare, "Energy efficient pollution monitoring system using deterministic wireless sensor networks," in *Advances in Intelligent Systems and Computing* vol. 438, ed, 2016, pp. 301-309.

[39] H. Rahim, C. Ghazel, and L. A. Saidane, "An Alternative Data Gathering of the Air Pollutants In the Urban Environment using LoRa and LoRaWAN," in *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 25-29 June 2018 2018, pp. 1237-1242. [Online]. Available: https://ieeexplore.ieee.org/ielx7/8410977/8450266/08450329.pdf?tp=&arnumber=8450329&isnumber=8450266&ref=. [Online]. Available: https://ieeexplore.ieee.org/ielx7/8410977/8450266/08450329.pdf?tp=&arnumber=8450329&isnumber=8450266&ref=

[40] V. Shakhov and O. Sokolova, "On Modeling Air Pollution Detection With Internet of Vehicles," in *2021 15th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, 4-6 Jan. 2021 2021, pp. 1-3, doi: 10.1109/IMCOM51814.2021.9377350. [Online]. Available: https://ieeexplore.ieee.org/stampPDF/getPDF.jsp?tp=&arnumber=9377350&ref=

[41] S. Winberg and S. Singh, "Real-Time Event-driven Air Quality Inspection Framework for City-wide Pollution Level Monitoring," in *2021 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, 2021: IEEE, pp. 1-6.

[42] E. Karapistoli, I. Mampentzidou, and A. A. Economides, "Environmental monitoring based on the wireless sensor networking technology: A survey of real-world applications," *International Journal of Agricultural and Environmental Information Systems,* Article vol. 5, no. 4, pp. 1-39, 2014, doi: 10.4018/ijaeis.2014100101.

[43] W. Y. Yi, K. M. Lo, T. Mak, K. S. Leung, Y. Leung, and M. L. Meng, "A Survey of Wireless Sensor Network Based Air Pollution Monitoring Systems," *Sensors,* vol. 15, no. 12, p. 29859, 2015. [Online]. Available: http://www.mdpi.com/1424-8220/15/12/29859.

[44] S. C. Folea and G. Mois, "A Low-Power Wireless Sensor for Online Ambient Monitoring," *IEEE Sensors Journal,* vol. 15, no. 2, pp. 742-749, 2015, doi: 10.1109/JSEN.2014.2351420.

[45] S. Rajasegarar *et al.*, "High-Resolution Monitoring of Atmospheric Pollutants Using a System of Low-Cost Sensors," *Geoscience and Remote Sensing, IEEE Transactions on,* vol. 52, pp. 3823-3832, 07/01 2014, doi: 10.1109/TGRS.2013.2276431.

[46] A. Alassi *et al.*, "Development of Innovative Indoor/Outdoor Air Quality Monitoring for Environmental Impact Aassessment in the State of Qatar," *WIT Transactions on Ecology and the Environment,* Article vol. 183, pp. 103-115, 2014, doi: 10.2495/AIR140091.

[47] S. Vishwas *et al.*, "Design and Development of Arduino Based Portable Air Quality Monitoring Systems," in *2021 1st Odisha International Conference on Electrical*

Power Engineering, Communication and Computing Technology (ODICON), 2021: IEEE, pp. 1-4.

[48] S. V. Girish, R. Prakash, S. N. H. Swetha, G. Pareek, T. S. Kumar, and A. B. Ganesh, "A network model of GUI-based implementation of sensor node for indoor air quality monitoring," in *Advances in Intelligent Systems and Computing* vol. 397, ed, 2016, pp. 209-217.

[49] X. Su *et al.*, "Intelligent and Scalable Air Quality Monitoring With 5G Edge," *IEEE Internet Computing,* vol. 25, no. 2, pp. 35-44, 2021, doi: 10.1109/MIC.2021.3059189.

[50] D. Mendez, S. Diaz, and R. Kraemer, "Wireless technologies for pollution monitoring in large cities and rural areas," in *2016 24th Telecommunications Forum (TELFOR)*, 22-23 Nov. 2016 2016, pp. 1-6, doi: 10.1109/TELFOR.2016.7818710. [Online]. Available: https://ieeexplore.ieee.org/ielx7/7803479/7818703/07818710.pdf?tp=&arnumber=7818710&isnumber=7818703

[51] L. Mocerino, F. Murena, F. Quaranta, and D. Toscano, "A methodology for the design of an effective air quality monitoring network in port areas," *Scientific Reports,* vol. 10, no. 1, p. 300, 2020/01/15 2020, doi: 10.1038/s41598-019-57244-7.

[52] P. Siagian and E. Fernando, "Smartphone Application As An Air Quality Monitor Using Raspberry Pi For Reducing Air Pollution," in *2021 2nd International Conference on Innovative and Creative Information Technology (ICITech)*, 23-25 Sept. 2021 2021, pp. 179-183, doi: 10.1109/ICITech50181.2021.9590187. [Online]. Available: https://ieeexplore.ieee.org/stampPDF/getPDF.jsp?tp=&arnumber=9590187&ref=

[53] S. Devarakonda, P. Sevusu, H. Liu, R. Liu, L. Iftode, and B. Nath, "Real-time air quality monitoring through mobile sensing in metropolitan areas," in *Proceedings of the 2nd ACM SIGKDD international workshop on urban computing*, 2013, pp. 1-8.

[54] S. Abraham and X. Li, "Design of A Low-Cost Wireless Indoor Air Quality Sensor Network System," *International Journal of Wireless Information Networks,* vol. 23, no. 1, pp. 57-65, 2016/03/01 2016, doi: 10.1007/s10776-016-0299-y.

[55] A. Kadri, E. Yaacoub, M. Mushtaha, and A. Abu-Dayya, "Wireless sensor network for real-time air pollution monitoring," in *2013 1st International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, 2013: IEEE, pp. 1-5.

[56] J. Kang and J. Y. Kim, "Portable RF-Sensor System for the Monitoring of Air Pollution and Water Contamination," *Journal of analytical methods in chemistry,* vol. 2012, 2012.

[57] J. Park, Y. Oh, H. Byun, and C. Kim, "Low Cost Fine-Grained Air Quality Monitoring System Using LoRa WAN," in *2019 International Conference on Information Networking (ICOIN)*, 9-11 Jan. 2019 2019, pp. 439-441.

[58] G. Dooly, C. Fitzpatrick, P. Chambers, and E. Lewis, *Development of a Fibre-Optic DOAS Sensor for the Detection of Exhaust Gases Using Ratiometric Separation Techniques*. 2007, pp. 1287-1290.

[59] V. Sajjan and P. Sharma, "Analysis Of Air Pollution By Using Raspberry Pi-IoT," in *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, 20-22 Jan. 2021 2021, pp. 178-183, doi: 10.1109/ICICT50816.2021.9358535. [Online]. Available: https://ieeexplore.ieee.org/document/9358535/

[60] E. Twahirwa, K. Mtonga, D. Ngabo, and S. Kumaran, "A LoRa enabled IoT-based Air Quality Monitoring System for Smart City," in *2021 IEEE World AI IoT Congress (AIIoT)*, 2021: IEEE, pp. 0379-0385.

[61] M. A. Reyna *et al.*, "A Low Cost Personal Environmental Monitor with Wireless Communication," in *2021 Global Medical Engineering Physics Exchanges/Pan American Health Care Exchanges (GMEPE/PAHCE)*, 15-20 March 2021 2021, pp. 1-1, doi: 10.1109/GMEPE/PAHCE50215.2021.9434849.

[62] B. Perumal, J. Deny, K. Alekhya, V. Maneesha, and M. Vaishnavi, "Air Pollution Monitoring System by using Arduino IDE," in *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 4-6 Aug. 2021 2021, pp. 797-802, doi: 10.1109/ICESC51422.2021.9533007. [Online]. Available: https://ieeexplore.ieee.org/document/9533007/

[63] S. Od, H.-H. Huang, and J.-B. Wei, "Apply LoRa Technology to Construct an Air Quality Monitoring IoT System," in *2021 IEEE 3rd Eurasia Conference on Biomedical Engineering, Healthcare and Sustainability (ECBIOS)*, 2021: IEEE, pp. 88-91.

[64] S. Tao, J. C. Fanguy, and T. Sarma, "A Fiber-Optic Sensor for Monitoring Trace Ammonia in High-Temperature Gas Samples With a ${\hbox {CuCl}} _ {2} $-Doped Porous Silica Optical Fiber as a Transducer," *IEEE Sensors Journal,* vol. 8, no. 12, pp. 2000-2007, 2008.

[65] H. Nandanwar and A. Chauhan, "IOT based Smart Environment Monitoring Systems: A Key To Smart and Clean Urban Living Spaces," in *2021 Asian Conference on Innovation in Technology (ASIANCON)*, 2021: IEEE, pp. 1-9.

[66] D. Wang, D. P. Agrawal, W. Toruksa, C. Chaiwatpongsakorn, M. Lu, and T. C. Keener, "Monitoring ambient air quality with carbon monoxide sensor-based wireless network," *Communications of the ACM,* Article vol. 53, no. 5, pp. 138-141, 2010, doi: 10.1145/1735223.1735257.

[67] Z. Liu, G. Wang, L. Zhao, and G. Yang, "Multi-points indoor air quality monitoring based on internet of things," *IEEE Access,* vol. 9, pp. 70479-70492, 2021.

[68] D. Zhang and S. S. Woo, "Real time localized air quality monitoring and prediction through mobile and fixed IoT sensing network," *IEEE Access,* vol. 8, pp. 89584-89594, 2020.

[69] R. W. Wies, R. A. Johnson, A. N. Agrawal, and T. J. Chubb, "Simulink model for economic analysis and environmental impacts of a PV with diesel-battery system for remote villages," *IEEE transactions on power systems,* vol. 20, no. 2, pp. 692-700, 2005.

[70] W. Zeng, F. Qin, X. Lu, and N. Bai, "Intelligent Monitoring Platform for Urban Pollution in Liaoning Province Based on Big Data," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*, 2020: IEEE, pp. 1985-1989.

[71] Y. Yong, "Continuous measurement of particulate emissions," *IEEE Instrumentation & Measurement Magazine,* vol. 8, no. 4, pp. 35-39, 2005, doi: 10.1109/MIM.2005.1518620.

[72] T. C. Yu *et al.*, "Wireless sensor networks for indoor air quality monitoring," *Medical Engineering and Physics,* Article vol. 35, no. 2, pp. 231-235, 2013, doi: 10.1016/j.medengphy.2011.10.011.

[73] G.Xiao, "Application and research of computer network technology in remotemonitoring,," presented at the IEEE International Conference on Artificial Intelligenceand Information Systems (ICAIIS), 2020.

[74] K. Zheng, S. Zhao, Z. Yang, X. Xiong, and W. Xiang, "Design and Implementation of LPWA-Based Air Quality Monitoring System," *IEEE Access,* vol. 4, pp. 3238-3245, 2016.

[75] I. Aicardi, N. Grasso, A. M. Lingua, F. Noardo, and F. Gandino, *A Low-Cost Solution for the Monitoring of Air Pollution Parameters through Bicycles* (Lecture Notes in Computer Science). Springer Verlag (in English), 2017, pp. 105-120.

[76] S. Brienza, A. Galli, G. Anastasi, and P. Bruschi, "A Low-Cost Sensing System for Cooperative Air Quality Monitoring in Urban Areas," *Sensors,* vol. 15, no. 6, p. 12242, 2015.

[77] E. Polychronidou, A. Lalas, D. Tzovaras, and K. Votis, "A systematic distributing sensor system prototype for respiratory diseases," in *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2019: IEEE, pp. 191-196.

[78] T.-H. Wen, J.-A. Jiang, C.-H. Sun, J.-Y. Juang, and T.-S. Lin, "Monitoring street-level spatial-temporal variations of carbon monoxide in urban settings using a wireless sensor network (WSN) framework," *International journal of environmental research and public health,* vol. 10, no. 12, pp. 6380-6396, 2013.

[79] M. Knoll, P. Breitegger, and A. Bergmann, "Low-Power Wide-Area technologies as building block for smart sensors in air quality measurements," *e & i Elektrotechnik und Informationstechnik,* vol. 135, no. 6, pp. 416-422, 2018, doi: 10.1007/s00502-018-0639-y.

[80] L. Chen *et al.*, "An Open Framework for Participatory PM2.5 Monitoring in Smart Cities," *IEEE Access,* vol. 5, pp. 14441-14454, 2017, doi: 10.1109/ACCESS.2017.2723919.

[81] M. S. Jamil, M. A. Jamil, A. Mazhar, A. Ikram, A. Ahmed, and U. Munawar, "Smart Environment Monitoring System by Employing Wireless Sensor Networks on Vehicles for Pollution Free Smart Cities," *Procedia Engineering,* vol. 107, pp. 480-484, 2015.

[82] M. R. Chowdhury, S. De, N. K. Shukla, and R. N. Biswas, "Energy-Efficient Air Pollution Monitoring with Optimum Duty-Cycling on a Sensor Hub," in *Twenty Fourth National Conference on Communications (NCC)*, 2018, pp. 1-6, doi: 10.1109/NCC.2018.8600133.

[83] J. Botero-valencia, L. Castano-Londono, D. Marquez-Viloria, and M. Rico-Garcia, "Data Reduction in a Low-Cost Environmental Monitoring System Based on LoRa for WSN," *IEEE Internet of Things Journal,* vol. 6, no. 2, pp. 3024-3030, 2019, doi: 10.1109/JIOT.2018.2878528.

[84] K. Tzortzakis, K. Papafotis, and P. P. Sotiriadis, "Wireless-Self Powered Environmental Monitoring System for Smart Cities Based on LoRa," in *Panhellenic Conference on Electronics and Telecommunications (PACET)*, 2017, pp. 1-4.

[85] K. Hu, V. Sivaraman, B. G. Luxan, and A. Rahman, "Design and Evaluation of a Metropolitan Air Pollution Sensing System," *IEEE Sensors Journal,* vol. 16, no. 5, pp. 1448-1459, 2016, doi: 10.1109/JSEN.2015.2499308.

[86] A. R. Al-Ali, I. Zualkernan, and F. Aloul, "A Mobile GPRS-Sensors Array for Air Pollution Monitoring," *IEEE Sensors Journal,* vol. 10, no. 10, pp. 1666-1671, 2010, doi: 10.1109/JSEN.2010.2045890.

[87] P. Sun, J. Weng, C. Yang, S. Chen, and J. Liu, "The Implementation of Air Pollution Monitoring Service Using Hybrid Database Converter," in *7th International Conference on Cloud Computing and Big Data (CCBD)*, 2016, pp. 269-274, doi: 10.1109/CCBD.2016.060.

[88] L. E. Cordova-Lopez, A. Mason, A. Shaw, and A. I. Al-Shamma'A, "Urban emissions monitoring at the vehicle level," in *Lecture Notes in Electrical Engineering* vol. 146 LNEE, ed, 2012, pp. 249-268.

[89] W.-Y. Yi, K.-S. Leung, and Y. Leung, "A Modular Plug-And-Play Sensor System for Urban Air Pollution Monitoring: Design, Implementation and Evaluation," *Sensors,* vol. 18, no. 1, p. 7, 2018.
[90] B. Ngom, M. R. Seye, M. Diallo, B. Gueye, and M. S. Drame, "A Hybrid Measurement Kit for Real-time Air Quality Monitoring Across Senegal Cities," in *2018 1st International Conference on Smart Cities and Communities (SCCIC)*, 24-26 July 2018 2018, pp. 1-6, doi: 10.1109/SCCIC.2018.8584551. [Online]. Available: https://ieeexplore.ieee.org/ielx7/8574341/8584545/08584551.pdf?tp=&arnumber=8584551&isnumber=8584545&ref=
[91] M. R. Pavani, P Trinatha., "Urban Air Pollution Monitoring Using Wireless Sensor Networks: A Comprehensive Review," *International Journal of Communication Networks and Information Security;,* vol. 9, no. 3, pp. 439-449., 2017.
[92] D. Suriano *et al.*, *A portable sensor system for air pollution monitoring and malodours olfactometric control* (Lecture Notes in Electrical Engineering). (in English), 2012, pp. 87-92.
[93] G. Carneiro, P. Fortuna, and M. Ricardo, "Flowmonitor: a network monitoring framework for the network simulator 3 (ns-3)," in *Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools*, 2009, pp. 1-10.
[94] ns-3. "latest release ns-3." snam.org. (accessed.
[95] M. Kabir, S. Islam, M. Hossain, and S. Hossain, "Detail Comparison of Network Simulators," 2014.
[96] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, 2003, pp. 126-137.
[97] A. Sobeih *et al.*, "J-sim: A simulation environment for wireless sensor networks," in *38th Annual Simulation Symposium*, 2005: IEEE, pp. 175-187.
[98] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, 2008, pp. 1-10.
[99] T. Issariyakul and E. Hossain, "Introduction to network simulator 2 (NS2)," in *Introduction to network simulator NS2*: Springer, 2009, pp. 1-18.
[100] G. F. Riley and T. R. Henderson, "The ns-3 Network Simulator," in *Modeling and Tools for Network Simulation*, K. Wehrle, M. Güneş, and J. Gross Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 15-34.
[101] A. S. Toor and A. Jain, "A survey on wireless network simulators," *Bulletin of Electrical Engineering and Informatics,* vol. 6, no. 1, pp. 62-69, 2017.
[102] M. Karl, "A Comparison of the architecture of network simulators NS-2 and TOSSIM," *National Institute of Technology Karnataka surathkal. wireless information network group,* 2005.
[103] S. Islam, "Detail Comparison of Network Simulators," *International journal of Scientific and Engineering Research,* vol. 5, no. 10, 2010.
[104] H. N. Pham, D. Pediaditakis, and A. Boulis, "From simulation to real deployments in WSN and back," in *2007 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, 2007: IEEE, pp. 1-6.
[105] C. Hanle and M. Hofmann, "Performance comparison of reliable multicast protocols using the network simulator ns-2," in *Proceedings 23rd Annual Conference on Local Computer Networks. LCN'98 (Cat. No. 98TB100260)*, 1998: IEEE, pp. 222-237.

[106] E. Weingartner, H. Vom Lehn, and K. Wehrle, "A performance comparison of recent network simulators," in *2009 IEEE International Conference on Communications*, 2009: IEEE, pp. 1-5.
[107] M. Davide, "Network level performances of a LoRa system," *Directores: Lorenzo Vangelista,* 2017.
[108] I. M. M. Bolivar, "Jamming on LoRaWAN Networks: from modelling to detection," Institut National des Sciences Appliquées de Rennes, 2021.
[109] B. Reynders, Q. Wang, and S. Pollin, "A LoRaWAN module for ns-3: Implementation and evaluation," in *Proceedings of the 10th Workshop on ns-3*, 2018, pp. 61-68.
[110] M. A. Almuhaya, W. A. Jabbar, N. Sulaiman, and S. Abdulmalek, "A Survey on LoRaWAN Technology: Recent Trends, Opportunities, Simulation Tools and Future Directions," *Electronics,* vol. 11, no. 1, p. 164, 2022.
[111] L. H. Haug, "An Indoor/Outdoor Air Quality Relationship Analysis Using Internet of Things," The University of Bergen, 2019.
[112] M. J. McGrath and C. N. Scanaill, "Sensor network topologies and design considerations," in *Sensor Technologies*: Springer, 2013, pp. 79-95.
[113] S. Saha, Kumar Acharjee, Uzzal, Tahzib-Ul-Islam, Md, *A Survey on Wireless Mesh Network and its Challenges at the Transport Layer*. 2014, pp. 169-177.
[114] M. Ouadou, O. Zytoune, D. Aboutajdine, Y. E. Hillali, and A. Menhaj-Rivenq, "Improved Cluster-tree Topology Adapted for Indoor Environement in Zigbee Sensor Network," *Procedia Computer Science,* vol. 94, pp. 272-279, 2016.
[115] R. Schoenmaeckers, Y. Deville, and R. Sadre, "Evaluating LoRa and LoRaWAN performance in Louvain-la-Neuve," Master's thesis, UCLouvain, 2019.
[116] M. Knight and B. Seeber, "Decoding LoRa: Realizing a modern LPWAN with SDR," in *Proceedings of the GNU Radio Conference*, 2016, vol. 1, no. 1.
[117] P. S. Cheong, J. Bergs, C. Hawinkel, and J. Famaey, "Comparison of LoRaWAN classes and their power consumption," in *2017 IEEE Symposium on Communications and Vehicular Technology (SCVT)*, 14-14 Nov. 2017 2017, pp. 1-6, doi: 10.1109/SCVT.2017.8240313.
[118] A. Lavric, A. I. Petrariu, and V. Popa, "SigFox Communication Protocol: The New Era of IoT?," presented at the International Conference on Sensing and Instrumentation in IoT Era (ISSI), 2019.
[119] E. Systems. "ESP32 Datasheet." https://cdn.sparkfun.com/datasheets/IoT/esp32/datasheet/.pdf (accessed.
[120] P. Lab. "How Does Flow Work." https://plumelabs.zendesk.com/hc/en-us/articles/360009014973-How-does-Flow-work- (accessed.
[121] P. Lab. "Flow-Getting-Started-Guide " https://plumelabs.zendesk.com/hc/en-us/articles/360006045613-Flow-Getting-Started-Guide (accessed.
[122] P. Lab. "what-is-an-aqi " https://air.plumelabs.com/learn/en/what-is-an-aqi (accessed.
[123] A. Council, "An Air Quality Index for Auckland: Review of Some International Techniques and Example Data," 1 July 2014. [Online]. Available: Reid, N and Rolfe, K (2014). An air quality index for Auckland: review of some international techniques and example data