UNIVERSITÀ DI PISA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Artificial Intelligence and Data Engineering

SmartParking

Internet of Things - Project report

Luana Bussu

Luca Caprioli

Academic Year: 2022/2023

Contents

1 Introduction

In the fast-paced urban landscapes of today, one of the most pervasive challenges faced by both city planners and individuals is the scarcity of parking spaces. As cities expand and the number of vehicles on the roads continues to rise, the traditional model of parking infrastructure struggles to keep up with the increasing demand. In response to these challenges, our project aim is to propose an innovative solution: the Smart Parking System harnessing Internet of Things (IoT) technologies.

1.1 Objectives of the Smart Parking System

Parking inefficiencies contribute significantly to traffic congestion, environmental pollution, and the overall stress levels of urban dwellers. The Smart Parking System aims to modernize the parking experience by integrating IoT devices and real-time data processing to create a more intelligent, efficient, and user-friendly parking ecosystem.
The primary objectives of our project are rooted in addressing the critical issues associated with urban parking:

- Optimizing Space Utilization: The system focuses on maximizing the utility of existing parking spaces by providing real-time information on their availability.
- Enhancing User Experience: By offering a seamless and convenient way for users to find parking spaces, the system aims to significantly improve the overall experience of urban mobility.
- Mitigating Traffic Congestion: The integration of IoT technologies provides a dynamic approach to managing traffic flow, thereby contributing to a decrease in congestion.
- Environmental Impact: The reduction of idle driving in search of parking not only improves traffic flow but also has a positive environmental impact, reducing carbon emissions and fostering a more sustainable urban environment.

This project report will delve into the architectural aspects, key features, and real-world benefits of our Smart Parking System, outlining how this innovative approach aligns with the evolving needs of urban landscapes.

The code is available in the following Github repository:
https://github.com/lubussu/Smart-Parking

2 SmartParking

In a typical environment, a smart parking ecosystem would utilize various sensors and actuators to monitor and control car traffic, lights, temperature and air quality levels, as well as check the occupancy of each parking spot. However, due to resource constraints and specific project requirements, the available hardware for our implementation is limited to six nRF52840 [1] dongles.

2.1 Architecture

Before going into details of the implementation, let's outline the selected sensor placement within a real scenario of a $48m \times 36m$ parking lot as in Figure 1. Considering the communication capabilities of the nRF52840 dongles, with a maximum outdoor range of $50m$ and an indoor range between $10m$ and $30m$, the decision was to locate the Border Router in the middle of the parking lot, two vibration sensor nodes placed respectively at the entrance and exit, both $24m$ from the BR and a final air-quality sensor at the opposite side of the entrance/exit, again at $24m$ from the BR to detect NO_2, in the spot with minimum air flow. Regarding the actuators, the entrance and exit bar actuators where located near their respective sensors, while the fan actuator was positioned near the BR to modulate the air flow directly in the center of the parking lot.
Once the total dimensions of $48m \times 36m$ were set, the dimensions and layout of the parking lots and travel lanes were chosen. Car slots, measuring $3m \times 6m$, were organized into two rows of 15 continuous slots at the top and bottom of the parking and two central rows of 10 contiguous parking slots, totaling in 50 parking spaces. The travel lanes, designed as one-way passages, were set at a width of $6m$ each to facilitate the smooth movement of vehicles, including larger cars, and to provide ample maneuvering space [2].

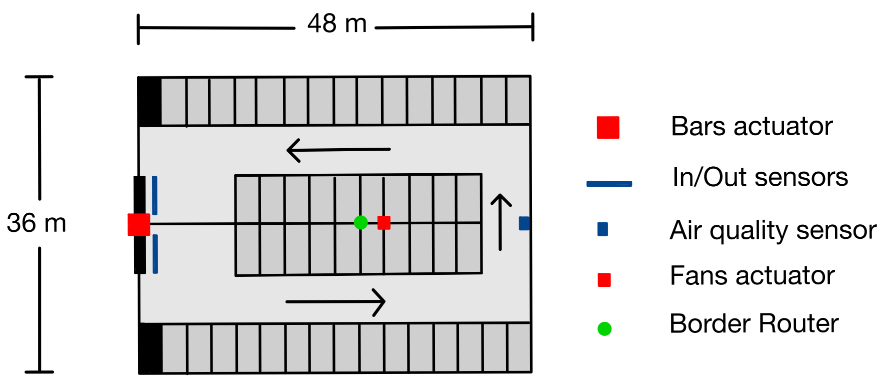

Figure 1: Parking layout

Figure 2 shows the overall architecture of the environment: the left side represent the sensor nodes and actuators that comunicate with the Application Side through a specific node that acts as Border Router. The Application Side collects and store data received from sensors and, if needed, send back a response to the actuators according to the application logic.

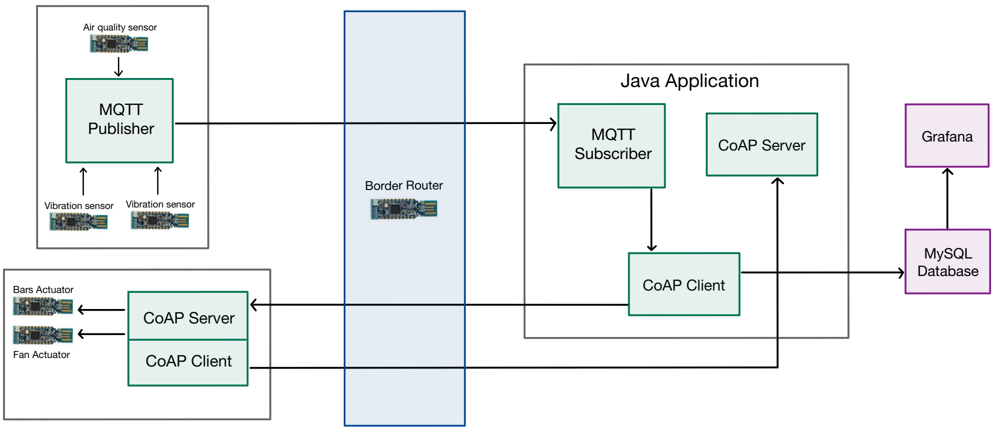

Figure 2: Project architecture

2.2 Implementation

In this section we're going into details of Sensor and Actuator nodes that compose the Low-Power and Lossy Network.

2.2.1 Message Format

The selected message format for both MQTT and CoAP nodes was JSON, chosen for its compatibility with multiple programming languages and platforms. JSON simplify integration with IoT devices and applications and helps in debugging due to its human-readable syntax. JSON was preferred over XML in this particular case for two main reasons:

- Given the focus on a smart parking lot application, a lighter message structure like JSON was chosen over XML due to its efficiency in communication, reducing the burden on the limited processing power and memory of the devices.
- XML's main advantage is its support for validation schema; however, for a smart parking lot, where data validation needs may be less stringent, a flexible and lightweight approach like JSON is preferred.

2.2.2 MQTT nodes

The MQTT nodes represent sensor nodes which main functionality is to sense the external environment and periodically publish the sensed values to a MQTT Broker. To emulate a realistic environment, vehicles traffic is generated using two of the nRF52840 dongles, one for entrance and one for exit. Since dongles have neither a real vibration sensor nor the floating point support, the decision was to let the dongles generate random vibrations (measured in $m/s^2 \times 10$) within ranges of $[0, 5]$ [3] and $[15, 25]$ [4] (to simulate both peoples and vehicles), both periodically or by pressing its button. This feature can be regulated thanks to the button and the RGB LED feedback:

- GREEN LED: The dongle will generate vibration measurements only when its button is pressed. Holding the button for more than two seconds will switch to automatic mode.
- BLUE LED: The dongle will generate vibration measurements periodically every five seconds. Holding the button for more than two seconds will switch to manual mode.

Additionally, a third dongle is used to simulate an air-quality sensor that generates nitrogen dioxide levels (NO_2, measured in ppb) in the range $[0, 200]$ [5], controlled thanks to the dongle button and with the dongle RGB LED as feedback:

- GREEN LED: The dongle will generate stable measurements in the set-up range or current range. Pressing the button for more than 2 seconds will prompt the dongle to generate increasing or decreasing measurements depending on its last state.
- BLUE LED: The dongle will generate increasing measurements. Pressing the button will prompt the dongle to generate decreasing measurements and holding it for more than 2 seconds will generate stable measurements.
- RED LED: The dongle will generate decreasing measurements. Pressing the button will prompt the dongle to generate increasing measurements and holding it for more than 2 seconds will generate stable measurements.

The structure of the JSON message is reported and described below:

```
{
    "app" : "smart_parking",
    "value": 22,
    "parking_id":1,
    "MAC" : "f4ce3605c3a3"
}
```

- "app": This field aim is to simplify the project deployment in an environment where multiple applications could coexist.

- "value": This field contains the measurement taken from the sensors. The same schema is utilized both for all the sensors.
- "parking_id": This field allows for the monitoring of multiple parking lot simultaneously.
- "MAC": This field serves to identify which sensor is the origin of the JSON message (entrance, exit or air-quality sensor) while providing debugging and fault tolerance needs.

2.2.3 CoAP nodes

The final two dongles (excluding the Border Router) act as two actuators designed to adjust the environment based on sensor measurements. The first dongle simulates a bar actuator that controls traffic flow in and out of the parking lot, providing feedback through LEDs and operating as follows:

- RGB LED: Emulates the entrance bar. While parking spots are available the LED is GREEN, emulating the opened bar that grants free entrance to the parking. Once the parking lot reaches full capacity, the LED switches to RED, simulating a closed bar."
- GREEN LED: Emulates the exit bar. The LED is turned off for most of the time, emulating the closed bar, but upon a vehicle is sensed by the exit sensor, the exit sensor senses a vehicles, it turns GREEN for 3 seconds, emulating the temporary opening of the exit bar.

The other dongle simulates a fan actuator used for air-quality control, whose speed is regulated by the levels of NO_2 sensed. The different speeds are represented by the RGB LED:

- LED OFF: The fan is turned off. This occurs when the NO_2 level is in the range $[0, 50)$.
- RED LED: The fan is turned on and its speed is set to "1". This occurs when the NO_2 level is in the range $[50, 100)$.
- BLUE LED: The fan is turned on and its speed is set to "2". This occurs when the NO_2 level is in the range $[100, 150)$.
- RED LED: The fan is turned on and its speed is set to "3". This occurs when the NO_2 level is in the range $[150, 200]$.

By employing this approach, the sensor network can emulate traffic flow in the parking lot with the correlated levels of pollution that traffic generates.

The CoAP nodes act both as clients and servers. In the initial step, a node registers itself with the Application Side by sending a JSON message containing its IP address and role. This registration process enables the Application Side to contact the nodes when necessary. The WHITE LED on the node is ON during the registration phase. The "role" field in the JSON message is crucial for distinguishing between the two types of actuators, allowing the Application Side to retrieve the IP addresses of the involved nodes. Once registered, the nodes function as servers, awaiting requests from the Application Side. Their responses to these requests trigger the specified actions described above.

2.2.4 Application Side

The Application Side, developed in Java, plays an important role in handling MQTT messages, storing information, implementing application logic (decision-making based on received data), and communicating with actuator nodes by sending CoAP messages. It is made up of several components:

- MQTT Subscriber: this component connects to the broker at 127.0.0.1 and subscribes to the topics "vibration" and "nox." Upon a new message is received, it stores all relevant information in the SensorData table of the MySQL Database and triggers the Actuator Handler.

- CoAP Server: it is responsible for receiving registration messages from CoAP Nodes containing IP addresses of Actuator Nodes, their roles, and the corresponding parking lot information. The CoAP Server, via the Database Handler, stores this information in the Actuators table of the MySQL Database.

- MySQL Database Handler: it handles the storage of sensor data and actuator IPs in the Actuators and SensorData tables, respectively.

- Actuator Handler: it makes decisions based on the received data from the MQTT Subscriber. When an action needs to be executed, it retrieves the IP addresses of the involved actuators (via the DB Handler) and sends a corresponding message to the CoAP Node on the server side.

3 Data Storage and Visualization

3.1 MySQL Database

The MySQL database is simply composed by two tables: Actuators and SensorData. The "Actuators" table in Figure 3 is designed to keep information about the actuator nodes within the parking environment, storing details such as role, IP address, and the associated parking lot.

Field	Type	Null	Key	Default	Extra
Actuator_IP	varchar(45)	NO	PRI	NULL	
Parking_ID	int(11)	YES		NULL	
Role	varchar(15)	YES		NULL	

Figure 3: Actuators table description

The "SensorData" table in Figure 4 functions as a repository for all data transmitted by MQTT nodes. It keeps information about the data topic (related to either a car passage or an air quality measurement), the MAC address of the node, the sensed value, and the timestamp of the measurement (for simplicity, the moment of reception is been used as timestamp).

Field	Type	Null	Key	Default	Extra
Sensor_MAC	varchar(45)	NO	PRI	NULL	
Topic	varchar(25)	YES		NULL	
Value	double	YES		NULL	
Parking_ID	int(11)	YES		NULL	
Timestamp	timestamp	NO	PRI	CURRENT_TIMESTAMP	

Figure 4: SensorData table description

3.2 Grafana

This section focuses on using Grafana to display essential statistics concerning the status of the parking area. aims to visualize data statistics of the current parking. Three main queries were performed to represent useful information.

Figure 5 shows the average number of cars present in the parking area for each hour between 9:00 to 20:00. This query is performed daily to monitor fluctuations in parking occupancy throughout the day.

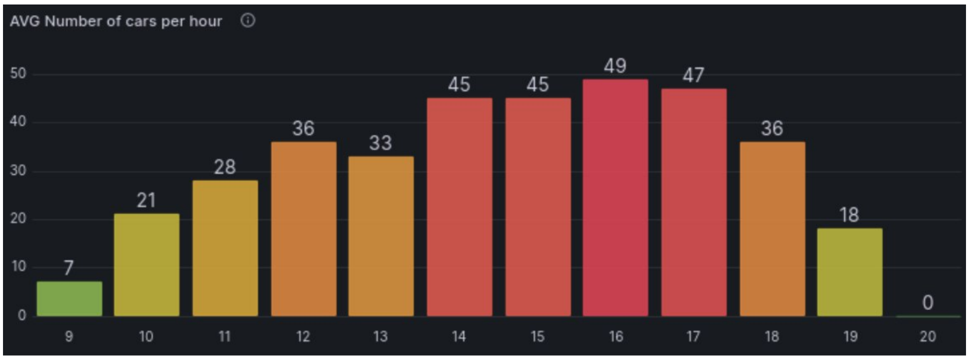

Figure 5: Average Number of Cars per Hour

Figure 6 displays the average number of cars entering and exiting the parking facility for each hour from 8:00 to 20:00. It considers all vehicles that enter and exit within the respective hour, providing a comprehensive view of traffic flow. This query is executed daily to analyze ingress and egress patterns.

Figure 6: Average Number of Entrance/Exit Cars per Hour

Figure 7 shows air quality measurements over the course of a day, in relation to the number of cars in the parking area. Observations indicate a correlation between increased parking occupancy and and high pollution levels. Notably, this experiment was conducted on a day without the utilization of fan actuators to underscore their important in improving air quality in the parking. The specific date chosen for analysis is '2024-01-30', allowing for a detailed examination of air quality over a day.

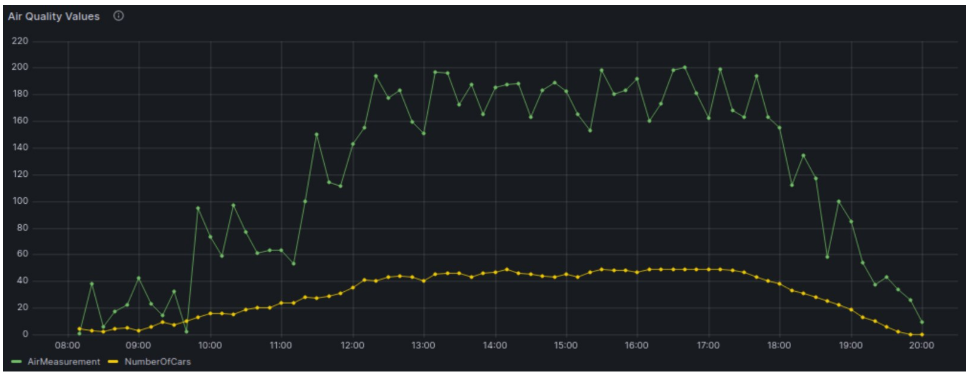

Figure 7: Air Quality Measurements in Relation to Parking Occupancy

4 Solutions and Improvements

The aim of this final chapter is to provide an overview of the existing market offerings regarding vehicle detection and air-quality monitoring sensors that can be connected to an nRF52840 device, as well as explore real-world actuators capable of controlling bar movements and air flow.

4.1 Vehicle detection sensor

Regarding vehicle detection, the choice of the sensor was mainly driven by the ease of integration with already existing parking lots and the method of detection. The EHIoT Self-powered vehicle detection system (Figure 8) [6] has some vital key points that were deemed on-point for the scope of this project:

- Ease of installation: The sensor can be installed on the entrance (or exit) of an already existing non-smart parking lot only by adapting its size and anchoring its base to the ground.
- Energy harvesting: The sensor powers-up and start functioning by absorbing the vibrations produced by passing vehicles [7], thanks to the piezoelectric effect, removing the need for a continuous power source.

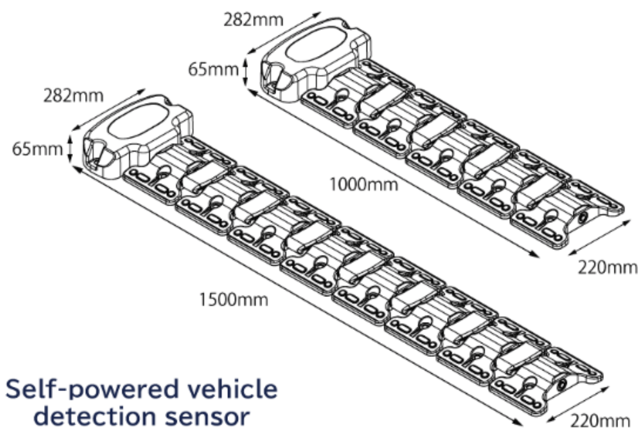

Figure 8: Self-powered vehicle detection system, Orbray Ltd.

4.2 Air-quality monitoring

For the air-quality monitoring the sensor of choice selected was the Alphasense NO2-B43F (Figure 9). The driven factors for this particular sensor were its lower cost comparatively to its competition and the fact that it was already tested in a parking lot scenario, obtaining good results [8]. Given the not so restrictive requirements, this precise sensor could be seamlessly linked to the nRF52840 to communicate its measurements to the BR.

Figure 9: Alphasense NO2-B43F

4.3 Possible Improvements

In considering improvements to our current parking solution, particularly in scenarios with greater capabilities, we can explore two possible solutions.

- Increase Entrance/Exit Points: To accommodate larger parking slots, one solution involves incorporating additional entrance and exit points as in Figure 10. This expansion would necessitate the utilization of extra sensors to monitor vehicle ingress and exit, along with supplementary sensors to simulate barrier functionality.

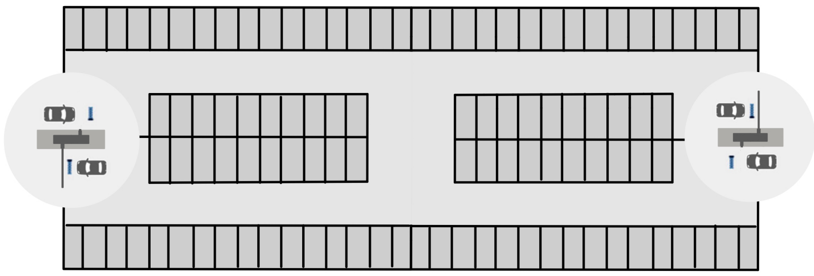

Figure 10: Example of Increase Entrance/Exit Points

- Comprehensive Sensor Coverage: Thinking of a scenario with a greater number of resources, one solution involves the implementation of a dedicated sensor for each parking space as in Figure 11. This approach not only facilitates the indication of available slots but also enables precise positioning information to be communicated effectively.

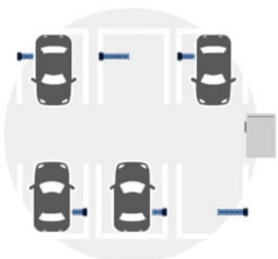

Figure 11: Example of Comprehensive Sensor Coverage

References

[1] Nordic Semiconductor. nrf52840.

[2] Dipartimento di Culture del Progetto Università IUAV di Venezia. Nozioni generali di progettazione dei parcheggi.

[3] Luca Martinelli, Vitomir Racic, Bruno Alberto Dal Lago, and Francesco Foti. Testing walking-induced vibration of floors using smartphones recordings. *Robotics*, 9(2):37, 2020.

[4] Daniel Garcia-Pozuelo, Antonio Gauchia, Ester Olmeda, and Vicente Diaz. Bump modeling and vehicle vertical dynamics prediction. *Advances in Mechanical Engineering*, 6:736576, 2014.

[5] EPA Unites States Environmental Protection Agency. Air quality guide for nitrogen dioxide.

[6] Orbray Ltd. Self-powered vehicle detection systems.

[7] Orbray Ltd. Generating electricity from mechanical vibrations - how it works and what it is used for.

[8] Andres Gonzalez, Adam Boies, Jacob Swanson, and David Kittelson. Measuring the air quality using low-cost air sensors in a parking garage at university of minnesota, usa. *International Journal of Environmental Research and Public Health*, 19(22), 2022.