Exercise 1

Chiara Thien Thao Nguyen Ba 10727985

May 24, 2025

1 Low-Cost IoT System for Forklift Tracking and Monitoring

Context
A logistics company operates a warehouse composed of a 500 m^2 underground indoor area and a 1 km^2 outdoor yard. Electric forklifts are used across both zones and return to specific docking stations to recharge.
Objective
Design a low-cost IoT system to:

1. localize forklifts in real time
2. monitor their status, including daily distance traveled, maximum and average speed, and impact detection.

Requirements
Your design must include:

- A description of the hardware installed on each forklift, including sensors, processing unit, and communication module.
- The chosen connectivity strategy, communication protocol, and data transmission frequency.
- A backend architecture detailing how data is ingested, processed, stored, and visualized.
- Pseudocode outlining the core logic running on each forklift (data collection, impact detection, communication)
- A graphical system-wide building block diagram showing the architecture from edge devices to the backend platform, including key data flows and system components.

In the following we describe a possible IoT system that provides real-time localization and comprehensive monitoring for electric forklifts operating across a 500 m^2 underground warehouse and 1 km^2 outdoor yard, addressing the challenges of mixed outdoor and underground environments while considering low cost, scalability and simplicity.

1.1 Hardware

Each forklift is equipped with a set of sensors, a central processing unit, and communication modules designed to balance **scalability**, **low cost**, and **simplicity**. The hardware components are detailed as follows:

- **Sensors:**
  - **GPS:** Used for real-time outdoor localization. It is a fully decentralized and infrastructure-free solution, making it highly scalable and cost-effective. It performs reliably in the wide and open outdoor area (1 km^2) of the yard.
  - **IMU (Inertial Measurement Unit):** Combines accelerometer and gyroscope to enable impact detection, speed estimation, and motion analysis. It is low-cost, compact, and effective in both indoor and outdoor settings. It integrates easily with the ESP32 and benefits from robust open-source libraries.
  - **Battery Level Sensor:** Monitors the forklift's current battery status in real time. This enables preventive maintenance and triggers alerts when the battery drops below a predefined threshold.
- **Processing Unit:**
  - **ESP32 Microcontroller:** Serves as the central unit for data processing and communication. It features built-in **Wi-Fi** and **Bluetooth Low Energy (BLE)**, supports **MQTT/MQTT-SN**, and has **low power consumption** with deep sleep capabilities. Its low cost, versatility, and ease of integration make it ideal for scalable IoT deployments.
- **Communication Modules:**
  - **Wi-Fi:** Used primarily in the indoor 500 m^2 warehouse where connectivity is stable. It allows direct and high-throughput communication with backend systems, facilitating seamless MQTT integration.
  - **LoRa:** Employed in the outdoor area for long-range, low-power communication. It covers the entire 1 km^2 yard and eliminates the need for cellular subscriptions or infrastructure, making it a **cost-efficient and scalable** choice.
  - **BLE:** Utilized for indoor positioning via BLE beacon triangulation. This enables reliable localization in areas where GPS signals are unavailable.

1.2 Connectivity strategy, Communication protocol and Data transmission frequency

This system employs a dual connectivity strategy tailored to the environmental constraints of the indoor (underground) and outdoor (open-air) areas. The chosen protocols and technologies aim to maximize coverage, reliability, and energy efficiency while minimizing cost and infrastructure requirements.

Outdoor Connectivity For the large outdoor yard (1 km^2), **LoRaWAN** is employed as the optimal solution. A single LoRaWAN gateway is sufficient to cover the entire area, thanks to its 2–5 km range capability in open spaces.

- **Data Collection:** Each forklift, equipped with an ESP32, collects GPS location, speed (via IMU), impact status, and battery level.
- **Data Transmission:** This data is compacted and transmitted over LoRa to a central gateway.
- **Gateway Role:** The LoRa gateway acts as an **MQTT-SN gateway**, converting received LoRa packets into MQTT-SN messages, which are then forwarded to a central MQTT broker via Ethernet or Wi-Fi.

Advantages:

- **Scalability:** A single LoRa gateway can support hundreds to thousands of devices.
- **Simplicity:** No need for routing or mesh logic, each node directly communicates with the gateway.
- **Cost-efficiency:** Unlike NB-IoT, LoRa has no recurring SIM fees or operator dependency.
- **Robustness:** Node failures do not affect the system as each forklift operates independently.

Indoor Connectivity For the 500 m^2 underground indoor warehouse, **Wi-Fi combined with BLE-based localization** provides a reliable and efficient solution.

- **Localization:** Fixed BLE beacons are installed (approximately 8–12 units spaced every 8–10 meters), emitting signals that are scanned by the forklift's ESP32.
- **Data Processing:** The ESP32 estimates location via RSSI and combines it with speed, impact, and battery level data.
- **Data Transmission:** Data is transmitted over **Wi-Fi** to a local edge gateway, which forwards it to the cloud backend via MQTT.

Advantages:

- **High Accuracy:** BLE-based indoor localization is well-suited to GPS-denied environments.
- **Low Power:** BLE offers ultra-low energy consumption.
- **Simple Integration:** Using Wi-Fi for data transfer allows direct communication with the backend without specialized infrastructure.

Communication Protocol

- **Outdoor: MQTT-SN** is used over LoRa. It is designed for non-IP, constrained networks, enabling efficient, lightweight communication from forklifts to the LoRa gateway.
- **Indoor: MQTT** is used over IP-based networks (Wi-Fi). It supports reliable, asynchronous message delivery, ideal for periodic updates (e.g., speed, location) and event-driven alerts (e.g., impact).

Advantages:

- **Lightweight and Open:** Both MQTT and MQTT-SN are open protocols, license-free, and suitable for low-bandwidth and constrained environments.
- **Flexible and Scalable:** The broker-based architecture supports many devices and allows decoupled communication, enhancing modularity and maintainability.

Zone Transition Detection To determine transitions between indoor and outdoor zones, a simple GPS signal monitoring mechanism is used:

- If the GPS signal is lost or degraded below a threshold for more than 10 seconds, the system assumes the forklift is indoors.
- When a valid GPS fix is restored, the forklift is considered to have returned outdoors.

This method requires no extra hardware, making it a cost-effective and scalable solution for area detection.

Power Management To optimize energy efficiency, the ESP32 enters deep sleep mode during idle periods and when the forklift is docked for recharging. While in the docking station, the device can also switch to a low-power operational mode where it continues periodic updates with reduced frequency, leveraging continuous power availability for background diagnostics and system health monitoring. This approach minimizes unnecessary energy use while ensuring that critical data is still collected and transmitted.

Data transmission frequency The data transmission frequency can be scheduled to balance responsiveness with power efficiency, which is essential for scalable and low-cost IoT deployments. Given the ESP32's support for low-power modes and the event-driven nature of forklift data, the following strategy is adopted:

- **Location/Speed/Distance updates**: Sent every 15 seconds when the forklift is moving, and reduced to every 60 seconds when idle. This dynamic scheduling saves bandwidth and energy, especially during idle periods. This is based also on the average speed of the forklifts, which is usually low around 5-10 km/h, thus we can assume that the movements are not so fast.
- **Impact alerts**: Transmitted immediately upon detection of sudden acceleration via the IMU.
- **System health and battery status**: Sent every 20 minutes to report system diagnostics and energy levels. This is enough to ensure regular monitoring without wasting communication resources.

1.3 Backend architecture

The backend system is designed to receive, store and visualize the data coming from the forklifts, providing scalability, simplicity and low cost implementation.

- **Data Ingestion**: A MQTT broker (e.g., Mosquitto) is used to collect data from forklifts. Indoor forklifts communicate via Wi-Fi using MQTT, while outdoor forklifts use MQTT-SN through a LoRa gateway. Each forklift publishes data to structured topics such as: *forklift/FL01/location, forklift/FL01/impact, forklift/FL01/battery*. This architecture enables real-time, asynchronous communication and supports scalability.
- **Data Processing**: Lightweight microservices (e.g., using Node-RED or Python) subscribe to MQTT topics and perform on-the-fly data processing. Tasks include: computing daily distance traveled, calculating maximum and average speed, counting impact events, tagging data with timestamps and forklift IDs. These services can also apply logic for generating alerts or forwarding data to storage.
- **Data Storage**: Two types of databases are used:
  1. A historical database (e.g., InfluxDB or TimescaleDB) stores long-term telemetry such as location history, speed trends, and impact logs. This data supports analytics, reporting, and operational optimization.
  2. A real-time operational database (e.g., Redis or a time-series DB with a live view) stores the current status of each forklift, including live position, speed, battery level, and impact events. This enables

instant alerting (e.g., low battery, collision) and real-time system monitoring.

- **Data visualization**: A real-time dashboard interface (e.g. ThingsBoard, or a custom web app) provides visual access to: live forklift positions (indoor zones or GPS-based maps), current speed and battery levels and alerts for impacts or maintenance conditions. The system includes a notification layer for sending email/SMS alerts or triggering maintenance actions based on predefined rules.

1.4 Pseudocode

The following part illustrates the pseudocode of the core logic running on each forklift.

```
initialize_sensors() // GPS, IMU, Battery sensor
initialize_communication_modules() // WiFi, LoRa, BLE

function publish_data(topic, data, zone):
    if zone == 'indoor':
        transmit_via_WiFi_MQTT(topic, data)
    else if zone == 'outdoor':
        transmit_via_LoRa_MQTT_SN(topic, data)

function setup_indoor_communication():
    connect_WiFi()
    initialize_MQTT_client()
    start_BLE_beacon_scanning()

function setup_outdoor_communication():
    disconnect_WiFi() // save power
    initialize_LoRa_module()
    stop_BLE_scanning() // save power

loop:
    // Read sensors
    gps_data = read_GPS()
    imu_data = read_IMU()
    battery_level = read_battery_sensor()

    // Determine zone (indoor/outdoor)
    if gps_data.fix_lost_for > 10_seconds ans gps_data.
        signal_power < GPS_THRESHOLD:
        zone = 'indoor'
        setup_indoor_communication()
    else:
        zone = 'outdoor'
        setup_outdoor_communication()

    // Calculate metrics
```

```
    if zone == 'indoor':
        speed = calculate_speed_from_IMU(imu_data)
        location = get_BLE_triangulated_position()
        distance = get_distance()
    else:
        speed = calculate_speed_from_GPS(gps_data)
        location = gps_data.coordinates
        distance = get_distance()

    is_moving = (speed > SPEED_THRESHOLD)

    // Detect impact
    if detect_impact(imu_data):
        impact_payload = {
            "timestamp": current_time,
            "impact_magnitude": imu_data.
                acceleration_magnitude,
            "location": location,
            "speed": speed,
            "zone": zone
        }
        publish_data(topic="forklift/<id>/impact", data=
            impact_payload)

    // Schedule transmissions
    if should_transmit("location", current_time, is_moving):
        location_payload = {
            "timestamp": current_time,
            "location": location,
            "speed": speed,
            "zone": zone,
            "distance": distance
        }
        publish_data(topic="forklift/<id>/location", data=
            location_payload)

    if should_transmit("battery", current_time):
        battery_payload = {
            "timestamp": current_time,
            "battery_level": battery_level,
            "system_health": get_system_diagnostics()
        }
        publish_data(topic="forklift/<id>/battery", data=
            battery_payload)

    // Power Management
    if is_docked_for_charging():
        reduce_transmission_frequency()
        enter_low_power_mode_while_charging()
    else if not is_moving and battery_sufficient():
```

```
        enter_light_sleep_until_next_transmission()

end loop
```

1.5 Graphical system diagram

The figure illustrates the architecture of the propose system: the edge devices (the forklifts) that represent the sensing layer with their sensors, the different gateways that compose the network layer, the storage databases with the analytics engine and the MQTT Broker that represent the data processing layer and then the dashboard/application that composes the final application layer.

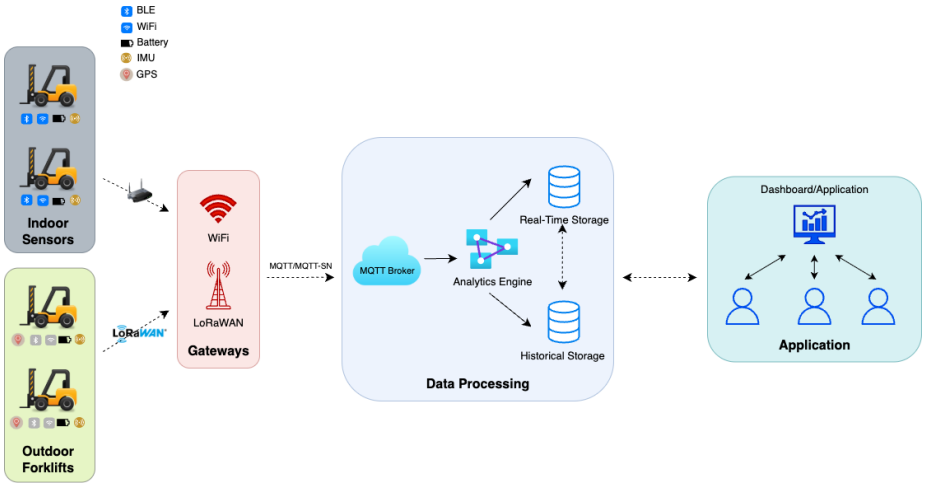

Figure 1: Diagram blocks of the proposed IoT system