IoT Homework 2024/2025
PART 1 – Exercise 1

Leader: Giovanni Ni 10831328

Xinyue Gu 10840236

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.

First of all, considering that the warehouse is very large, we believe that using short-range technologies is inconvenient, as this would require installing many gateways throughout the facility.
We initially considered using two different technologies for the indoor and outdoor areas—ZigBee for the indoor area and LoRaWAN for the outdoor yard—but we dismissed this option for two main reasons:

1. ZigBee is not well-suited for underground environments.
2. All electric forklifts operate across both zones, and using two different technologies could lead to interoperability issues.

We have ultimately decided to build a unified system based on LoRaWAN for both the indoor and outdoor areas. The decision is supported by the following reasons:

1. Excellent Coverage
   a) Outdoor: Easily covers 1 km^2 with a single(or a few) gateway.
   b) Indoor (underground): LoRa signals penetrate concrete and metal much better than 2.4 GHz ZigBee or BLE, ensuring reliable communication even in challenging environments.

2. Low Infrastructure Cost
   a) A small number of gateways may be sufficient for complete coverage (potentially even just one).
   b) No need to build a complex mesh or install extra routers.
   c) Forklifts can connect directly to the gateway from anywhere.
3. Low Power + Lightweight Communication
   a) Asynchronous uplink (ALOHA protocol): No time synchronization required; devices can simply transmit and return to sleep.
   b) This makes LoRaWAN ideal for forklifts that may not have a constant power supply, such as during shutdown or charging periods.
4. LoRaWAN is particularly well-suited for mobile elements like forklifts due to its star topology, which allows each device to transmit directly to any gateway within range. There is no need to maintain links or reconnect when moving. In contrast, technologies like ZigBee, BLE, and 6LoWPAN rely on mesh topologies, which degrade rapidly in mobile or dynamic scenarios, making them less suitable for real-time forklift tracking.

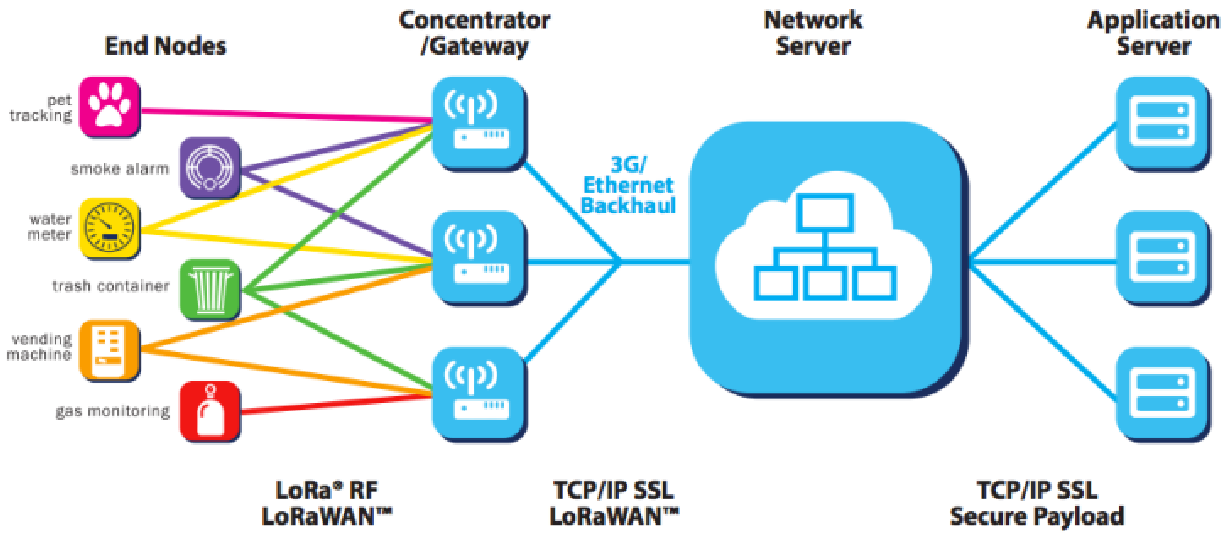
Now passing on the hardware part, we should install on each forklift:

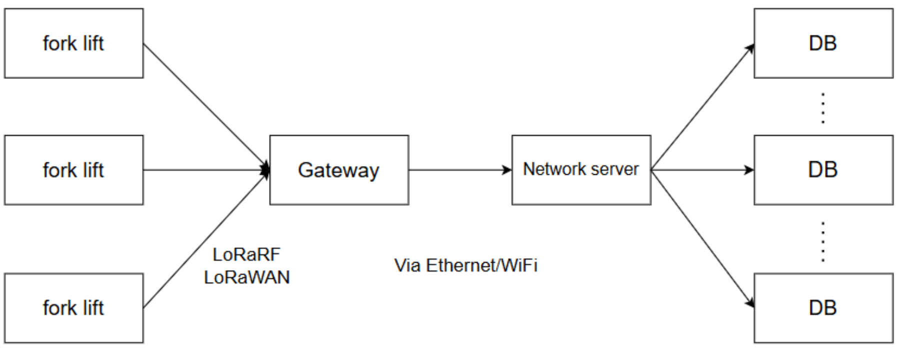
1. A GPS sensor since we should localize forklifts in real time.
2. Accelerometer/speedmeter since we want the maximum and average speed
3. Odometer since we want to monitor forklift's daily distance traveled.
4. impact sensor for impact detection
5. ESP32 acts as processing unit, collecting all data and inoltrate them to the gateway.
6. ESP32 ADC pin read constantly the battery level to monitor the forlift status and send "forklift should be recharged message" or directly control it to move to the recharge station.

Gateways should be installed inside the warehouse, but the exact number required depends on the layout of the areas and real-world factors such as wall materials, structural obstacles, and interference sources. A detailed site survey is necessary to determine the optimal number and placement of gateways to ensure reliable coverage across both indoor and outdoor zones.

To enable LoRaWAN communication, each forklift must be equipped with a LoRa transceiver module, such as the RAK4200. Additionally, external antennas are required for these modules. For optimal signal range and reliability, the antennas should be mounted externally on the forklifts.

The connectivity strategy is similar to that studied during the lecture:





In our case:

data transmission frequency:

- GPS position: every 10 seconds
- Speed data: every 10 seconds
- Distance traveled: every 15 minutes
- Max speed: every 15 minutes
- Average speed: every 15 minutes.
- Battery status: every 15 minutes.
- Impact detection: transmitted IMMEDIATELY upon detection
- Daily summary(at 23.59): total distance traveled, maximum and average speed, number of impacts detected during the day

For example the backend path could be:
LoRa Gateway → Ethernet/Wi-Fi → MQTT/HTTP → Network Server → Cloud Backend

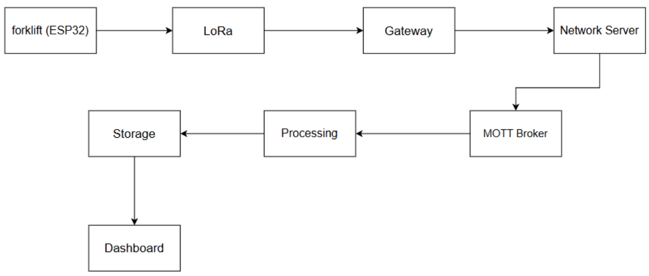
Backend architecture:

1. Data Ingestion:
   ESP32 + Lora: Collects sensor data (GPS,speed, voltage..) and sends via LoRa.
   LoRaWAN Gateway: Receives LoRa packets and forwards them to the backend
   LoRa Network Server: Manages device sessions, manage packets from multiple gateways, and queues messages for processing.
2. Data Processing Layer:
   MQTT Broker: Receives decoded uplink data from LoRa Network Server and publishes them to relevant topics.
   Node-RED: Enriches data: for example: battery %, distance, alert detection.

3. Data Storage:
   Databases:
   Time-Series DB which stores GPS, voltage, speed, and other sensor readings over time.
   Relational DB which stores info like forklift IDs and configuration settings.

4. Visualization
   Accessible through a web or mobile application, providing real-time dashboards, historical data trends and alerts.

Summarizing:



Pseudo code of ESP32:

WOKWI SAVE SHARE

sketch.ino • diagram.json Library Manager

```
Initialize sensors: GPS, accelerometer, odometer, battery monitor
                    impact sensor
Initialize LoRaWAN module

Loop forever:
    Read GPS position (latitude, longitude)
    Read speed
    Read accelerometer data
    Read battery voltage via ADC

    Compute:
        - ImpactDetected
        - AverageSpeed and TotalDistance
        - BatteryStatus

    If ImpactDetected:
        Prepare impact alert payload
        Send via LoRa immediately

    Every 10 seconds:
        Prepare status payload:
            - Timestamp
            - GPS position
            - Speed
        Send via LoRa

    Every 15 minutes:
        Prepare summary payload:
            - Total distance traveled
            - Max speed
            - Average speed
            - Battery status
        Send via LoRa
```

```
    Every 10 seconds:
        Prepare status payload:
            - Timestamp
            - GPS position
            - Speed
        Send via LoRa

    Every 15 minutes:
        Prepare summary payload:
            - Total distance traveled
            - Max speed
            - Average speed
            - Battery status
        Send via LoRa

    Every 24 hours:
        Prepare daylysummary payload:
            - Date
            - Total distance traveled
            - Max speed
            - Average speed
            - Battery status
            - Number of impact detected
    Sleep for a short duration to save power
```