IOT Homework exercise #1

Francesco Spangaro - Luca Tosetti

25 May 2025

POLITECNICO
MILANO 1863

Academic Year 2024 - 2025

Contents

1 Hardware

1.1 Forklift's sensors

- ESP32-H2-DevKitM-1: the board that will be installed on each forklift, connected to the other sensors. We chose this board because it already contains a 6LoWPAN module, allowing us to use the protocol without the need to purchase a separate module. Alternatively, a normal ESP32 with a 6LoWPAN module can be used.
- Accelerometer: the sensor will be used to detect any crashes that may occur during the day. Its measurements will be used by the board to compute whether the forklifts have accidentally collided with anything, and if that is the case, it will send the information to the back-end server.
- GPS module: used to perform outdoor localization.
- BLE receiver: used to perform indoor localization. It will communicate with the BLE beacons placed indoors to triangulate the forklift's position.
- SIM module: we use the NB-IoT protocol for communication while the forklift is outside. The protocol is based on LTE connectivity; therefore, we need a SIM card.
- Wheel RPM sensor: used to compute the RPM of the forklift's wheels. With it, the server will be able to compute the average speed, maximum speed, and total kilometers traveled.
- Battery (optional): if we cannot attach the board and sensors to each forklift's internal battery, we will need a battery module. A lithium battery will be necessary for maximum lifespan.

1.2 Architectural components

- BLE beacons: needed to communicate with the BLE receiver on each forklift. Each forklift BLE receiver will scan the surrounding area to search for BLE beacons, gather their RSSIs and IDs, and send them to the back-end for the position computation.
- 6LoWPAN routers: router nodes deployed indoors. These nodes must be strategically placed, in order to ensure a good coverage and connectivity. They need to be connected to the power grid.
- 6LoWPAN border router: needed to let IoT devices interact with the back-end server. It acts as a bridge between the 6LoWPAN network and the external network.

2 Design decisions

2.1 Outdoor positioning

In the outdoor storage we can use the GPS module connected to the main ESP board. The module will be configured accordingly to use as little battery power as possible while still producing precise measurements. We decided to adopt this solution for the outside position tracking, as it is the most simple and convenient solution.

2.2 Indoor positioning

Because the indoor storage unit is underground, using the GPS module will not provide reliable measurements. Thus, we decided to use triangulation for indoor positioning. By placing a BLE receiver on the forklift and multiple BLE beacons with unique IDs in the compound, we can measure the RSSI received by the IoT device by each reachable beacon. The data will be sent to the back-end server, which will precisely triangulate the forklift's position.

2.3 Communication protocol

In order to send sensors measurements to the back-end, we decided to use the MQTT-SN protocol. This protocol is particularly adequate for our use case, being designed for constrained devices and networks. It makes efficient use of the bandwidth using smaller packets compared to MQTT and the UDP protocol. It is optimized for intermittent connectivity, allowing clients to go into a "sleep mode" or be temporarily disconnected.

2.4 Position switching

For understanding the forklift's relative position, if it is in the internal or external warehouse, we decided to check the BLE connectivity. If the device is in communication range with at least three BLE beacons, then we assume that the forklift is indoors. The number three was chosen since it is the minimum number of RSSI data needed in order to correctly triangulate the forklift's position. The switch needs to be precise because forklifts need to change their connectivity too, from NB-IoT to 6LoWPAN going from the outside to the inside.

2.5 Connectivity protocol indoors

The total inside area is $500m^2$. We can achieve full area coverage with a network of routers using the 6LoWPAN protocol. Each forklift will connect to a router and share sensor measurements with the back-end server. The routers will be strategically placed to achieve complete indoor coverage. Each router will be implemented with a microcontroller connected to the power grid. We decided to use this particular protocol for several reasons:

- The 6LoWPAN multi-hop communication and ability to form a mesh network, allowing us to cover a large area, bypassing obstacles, with the deployment of a small amount of routers in the underground area.
- It is a low power consumption protocol.
- It is inexpensive, allowing IPv6 packets to be transmitted with low-power wireless microcontrollers.
- It allows us to create a scalable network. By adding more 6LoWPAN devices, we can strengthen the mesh network by providing more routing options.

2.6 Connectivity protocol outdoors

The total outside area is $1km^2$, due to this, we decided to use the Narrowband IoT protocol for communication between forklifts and the MQTT broker. Each IoT device is equipped with a SIM module, needed because the NB-IoT protocol communicates using LTE networks. We chose this protocol because of its long-range connectivity and its ability to work on existing mobile networks. This protocol is easily scalable, since it is integrated in each IoT device, and cheap, because we do not need to create a new network infrastructure, but we can leverage the already existing one. The only costs come from the purchase of the SIM module and the carrier subscription fee for each device. This protocol has a very low power consumption.

2.7 Back-end

Assuming that the company already owns an on-premise server, this will act as an MQTT broker for both indoor and outdoor forklifts. The server will also act as an MQTT-SN gateway, to allow forklifts using the cellular network to send MQTT-SN packets to the gateway. Each packet will contain the following:

- Collision flag: flag computed on each forklift that shows whether the forklift has suffered an impact.
- RPM value: needed to evaluate maximum speed, average speed, and total distance traveled.
- RSSI values of the BLE beacons: used to triangulate the indoor positioning of the forklift. Will be set to null if the forklift is outside.
- GPS measured position: will be set to null if the forklift is inside.

The back-end server processes the RPM values received from the forklifts and computes the velocity. Then, on request, it evaluates the average speed and the total distance traveled. In the server is instantiated an MQTT client subscribed to all the different topics needed, so that it can receive all measurements from the forklifts.

2.8 Data storage and visualization

All data are stored on the server and, upon request, can be displayed on an appropriate dashboard on the machine, or online by connecting to the server remotely. In the dashboard it is set up an alarm system. Users can insert their email addresses, and, when a crash occurs, the server will receive a precise packet. It will parse it and notify all email addresses of the crash.

2.9 Pseudocode

```
FUNCTION loop():
    call setup()
    loop forever:
        delay(100 ms)
        duty_cycle()

FUNCTION setup():
    6LoWPAN.init()
    6LoWPAN.connect_to_network() // Attempt to join the 6LoWPAN mesh
    NB_IoT.init()
    NB_IoT.connect() // Connect to NB-IoT network
    MQTT_SN.init(MQTT_SN_GATEWAY_IP, MQTT_SN_GATEWAY_PORT)
    MQTT_SN.connect(FORKLIFT_ID) // Connect to MQTT-SN gateway
    Accelerometer.init()
    Accelerometer.set_impact_threshold(IMPACT_ACCEL_THRESHOLD)
    GPS.init()
    GPS.set_power_saving_mode(TRUE) // Start in power-saving mode
    BLE_Receiver.init()
    BLE_Receiver.start_scan() // Start scanning for beacons
    ProximitySensor.init()

FUNCTION setup_timers():
    declare timer_config as esp_timer_create_args_t
    set timer_config.callback to forklift_status_read
```

```
    set timer_config.name to "forklift_status_read"
    call esp_timer_create(&timer_config,&timer_handle)
    call esp_timer_start_periodic(timer_handle,5_000_000) // 5s

FUNCTION forklift_status_read():
    IF GPS.is_data_available() THEN
        current_position = GPS.read_data()
    ELSE
        ble_beacon_readings = BLE_Receiver.get_scanned_beacons() // Returns list of (ID, RSSI)
        IF ble_beacon_readings.size() < 3 THEN current_position = prev_position
        ELSE
            BLE_Receiver.clear_scanned_beacons() // Clear for next scan cycle
            queue_mqtt_messages.push(ble_msg(ble_beacon_readings))
    queue_mqtt_messages.push(status_msg(current_position))

    VAR current_rpm_count = ProximitySensor.get_rpm_count()
    queue_mqtt_messages.push(rpm_msg(current_rpm_count))

FUNCTION impact_detection():
    VAR accel_x, accel_y, accel_z = Accelerometer.read_data()
    impact_detection = check_for_impact(accel_x,accel_y,accel_z)
    IF impact_detection THEN queue_mqtt_messages.push(impact_msg(accel_x,accel_y,accel_z))

FUNCTION duty_cycle():
    location = determine_location_context() // indoor or outdoor
    manage_gps_power(location)
    switch_connectivity()
    impact_detection()
    IF queue_mqtt_messages.size()>0 THEN
        FOR each message IN queue_mqtt_messages.popAll()
            send_telemetry_data(message)

FUNCTION determine_location_context(ble_beacon_readings):
    // Decide if indoor or outdoor based on BLE beacon presence
    VAR num_visible_beacons = ble_beacon_readings.size()
    VAR strongest_rssi = -1000 // very low initial value

    FOR each beacon_reading IN ble_beacon_readings:
        IF beacon_reading.RSSI > strongest_rssi THEN strongest_rssi = beacon_reading.RSSI

    IF  num_visible_beacons > 2 AND
        strongest_rssi > INDOOR_AREA_THRESHOLD_RSSI THEN is_indoor = TRUE
    ELSE is_indoor = FALSE

FUNCTION manage_gps_power():
    IF is_indoor THEN GPS.set_power_saving_mode(TRUE) // Turn off GPS when indoors
    ELSE IF is_forklift_stationary THEN GPS.set_power_saving_mode(TRUE) // Turn off GPS when stationary outdoors
    ELSE GPS.set_power_saving_mode(FALSE) // Keep GPS active when moving outdoors

FUNCTION send_telemetry_data(message):
    VAR payload_data = CREATE_COMPACT_BINARY_PAYLOAD(message.data) // Function to pack data into binary
    VAR topic = "forklift/" + FORKLIFT_ID + "/" + message.type

    IF is_indoor THEN
        IF MQTT_SN.is_connected() THEN MQTT_SN.publish(topic, payload_data, QOS=1)
        ELSE MQTT_SN.connect(FORKLIFT_ID)
```

2.10 Block-Diagram

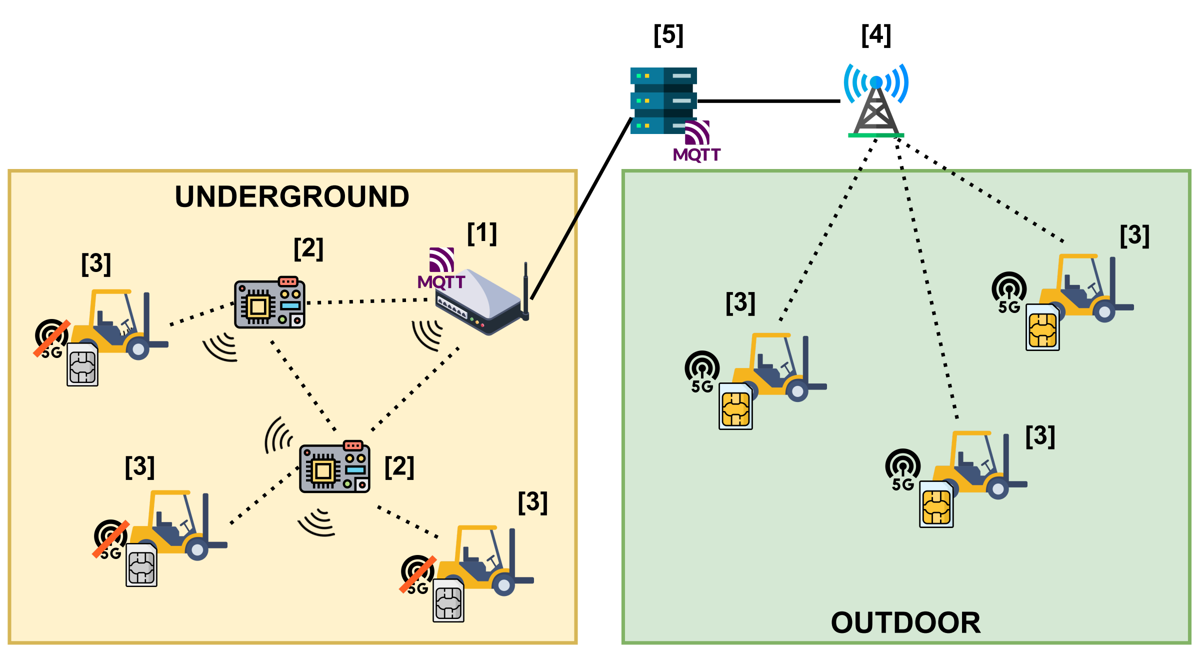

Figure 1: System architecture

2.10.1 Legend

1. 6LoWPAN Edge Router. Used as a bridge between the 6LoWPAN network and the external one. It transforms the 6LoWPAN IPv6 packets into the standard IP format. In the same node, an MQTT-SN gateway is also installed that receives the MQTT-SN packets sent by the forklifts while connected to the 6LoWPAN network.

2. 6LoWPAN network router. It is a microcontroller connected to the power grid that has the task of permanently acting as a 6LoWPAN router. They are deployed to ensure that the underground area is completely covered by the 6LoWPAN network.

3. Forklifts that are used in the two areas (indoor and outdoor). They are equipped with a 6LoWPAN compatible ESP board, several sensors, and a SIM module to enable use of the NB-IoT protocol.

4. Cellular tower to which the different forklifts connect, in case they are used outdoors. NB-IoT connectivity is used to send the different MQTT-SN packets.

5. Back-end server. It ingests, processes, and visualizes data and measurements collected by forklifts during their use. In it, the MQTT broker and the MQTT-SN gateway are instantiated, in order to ingest data from the 6LoWPAN edge router and the NB-IoT cellular tower, respectively.