GATEWAY DEVELOPMENT USING BLE & MQTT

Andrew Brady

C20769825

COMP4000

TU821/4

26/11/2023

# Introduction

The following is a design document that outlines the comprehensive architecture and development plan for an IoT gateway development system that models a pet feeder. The system is designed around a gateway that employs Bluetooth Low Energy (BLE) technology to communicate with an end device, focusing on gathering and transmitting environmental data.

# Scope

The project consists of the creating of an individualised BLE central device application, operational on a memory stick, dedicated to leveraging BLE services and characteristics. This application interfaces with a BBC MicroBit BLE peripheral device, responsible for measuring and relaying environmental variables.

The peripheral device is engineered with two distinct BLE services. The primary service facilitates the transmission of sensor data from the MicroBit to the central device gateway. Concurrently, a secondary BLE service allows for interactive control, enabling actions like activating an LED, orchestrated by commands originating from the gateway.

The gateway application, scripted in NodeJS, undertakes pivotal roles within the system architecture. Its responsibilities encompass the retrieval of sensor data from the peripheral device, transmission of this data to an MQTT broker, and subsequently subscribing to the broker for incoming messages. These incoming messages from the broker empower the gateway to execute actions on the peripheral device, such as toggling the LED functionality.

To ensure efficient data transmission to and from the cloud, MQTT has been selected as the primary transport protocol. MQTT facilitates seamless communication between the gateway and the MQTT broker, serving as the conduit for sensor data transmission and enabling responsive command execution.

# Architecture Diagram



# Flowchart



# Application

The IoT pet feeder is designed to provide automated feeding for pets using an IoT-enabled mechanism. The system integrates a MicroBit with a button-triggered feeding mechanism and utilises MQTT for communication between the pet feeder and an MQTT broker. The Node.JS code implements the interaction and communication flow with the feeder device, MQTT broker and associated services.

## Functionality

**Button Interaction :** The system is configured to detect button presses on the MicroBit. Upon detecting a press, the button press count is incremented and monitored for specific conditions

**MQTT Communications :** Utilises MQTT to publish and subscribe to specific topics (‘feederSub’ and ‘feederPub’) on the configured MQTT broker**.** Publishes pertinent information about the feeder’s status, such as when it’s dispensing food or when it needs a refill.

**LED Indication :** Integrated an LED as a visual indicator, on the condition that a message is published from the broker to the device.Used as an indicator for when the feeder is full and the counter needs to be reset.

## Workflow

The workflow of the application can be described in a series of events as below :

* The code establishes a connection with the MQTT broker using specified credentials (username, password) and connection settings ( host, port, protocol).
* Initiates BLE device discovery to find the designated MicroBit device using its mac address
* Connects to the discovered device and explores its available services.
* Monitors button presses on the pet feeder device, tracking the count and triggering actions accordingly.
* When the button is pressed 10 times, a message is published to the MQTT broker and the terminal, indicating that the feeder needs a refill
* During a button hold, food dispensing is indicated and published to the MQTT topic.
* Publishes usage statistics when the button is released after feeding.
* Configures an LED service on the device
* Responds to MQTT messages by turning on the LED to signal warnings, providing visual feedback to users.

# Components

## BBC MicroBit

The BBC MicroBit (Named “Twitch” in this project) is what is technically known as a microcontroller development board. That is, it’s a printed circuit board (PCB) which contains a microcontroller on which you can run your own programs and connect your own hardware. [1] It has a variety of uses in IoT:

**Sensing and Data Collection :** The MicroBit has built-in sensors like an accelerometer, magnetometer, and temperature sensor. It can collect data from its surroundings, such as movement, orientation or temperature.

**Communication :** It can communicate with other devices using Bluetooth or radio signals. This capability allows it to interact with and control other IoT devices or systems.

**Actuation :** The MicroBit can also control external devices. For instance, it can turn on/off LEDs, activate motors, or trigger other actuators based on the data it collects or receives from other IoT devices.

In an IoT setup, the MicroBit might function as a sensor node, collecting data from its environment, transmitting this data to a central hub or server, and possibly receiving instructions or updates to control other connected devices based on the data it collects.

## BLE Gateway

A BLE Gateway is a communication portal between Bluetooth devices (e.g. beacons and sensors) and the Cloud/client server, comparable in principle to an internet router. [2] The BLE Gateway using Debian and NodeJS serves as a bridge between Bluetooth Low Energy devices and other systems in an IoT setting by :

**BLE Communication** **:** Bluetooth Low Energy devices use a low-power protocol to transmit data. A BLE gateway equipped with Debian 11 (a Linux distribution) and NodeJS can scan for nearby BLE devices and establish connections with them.

**Data Collection :** Once connected, the gateway collects data from these BLE devices. This data could include sensor readings, device information, or any other relevant data being broadcasted by the BLE devices.

**Processing and Aggregation :** NodeJS, being a server-side JavaScript runtime, allows for easy processing and manipulation of the data collected from the BLE devices. It can perform computations, filtering, or formatting of the data as required by the IoT application.

**Integration with IoT Systems :** The BLE gateway can then transmit this processed data to other IoT systems or cloud platforms. It uses MQTT protocols to send the collected data to a central IoT server.

**Control and Management :** The gateway may also receive instructions or commands from the central IoT system, allowing it to control BLE devices, modify settings, or trigger actions based on the received commands.

The BLE gateway using Debian and NodeJS acts as an intermediary that facilitates communication between Bluetooth Low Energy devices and larger IoT infrastructures. It plays a crucial role in collecting, processing, and transmitting data from BLE devices, enabling seamless integration into broader IoT applications or systems.

## MQTT Broker

The MQTT protocol is a Machine-to-Machine (M2M) and IoT connectivity protocol. MQTT is a lightweight messaging protocol that works with a broker-based publish-subscribe mechanism and runs on top of Transmission Control Protocol/ Internet Protocol (TCP/IP). [3] The MQTT (Message Queueing Telemetry Transport) broker serves as a pivotal component for facilitating communication between IoT systems by :

**Message Broker:** The MQTT broker acts as a central hub or intermediary server that receives messages from multiple IoT devices and clients.

**Publish/Subscribe Model:** MQTT uses a publish/subscribe messaging pattern. Devices can publish messages to specific topics, and other devices or systems interested in those topics can subscribe to receive those messages.

**Message Routing:** When a device publishes a message to a particular topic, the MQTT broker receives it and routes the message to all the clients that have subscribed to that specific topic. This allows for one-to-many communication.

**Efficient Communication:** MQTT is designed to be lightweight and efficient, making it ideal for IoT environments where devices might have limited processing power, memory, or battery life. It uses minimal bandwidth, making it suitable for remote or low-bandwidth scenarios.

**Reliability and Quality of Service (QoS):** MQTT supports different levels of Quality of Service, allowing devices to choose the level of reliability they need when sending or receiving messages. This ensures that messages are delivered reliably even in unreliable network conditions.

**Scalability:** MQTT brokers are highly scalable, capable of handling a large number of connected devices and managing message traffic efficiently.

**Integration:** MQTT brokers can integrate with various IoT platforms, cloud services, databases, or applications, enabling seamless data transfer and interaction between IoT devices and these systems.

The MQTT broker acts as a messaging middleman, enabling efficient, reliable, and scalable communication between IoT devices and the broader ecosystem.

# Sustainability

There are a few ways to implement sustainability when building an IoT system.

## Virtual Gateway Implementation

Instead of relying on physical gateway devices, which involve production costs and material resources, the implementation of a virtual machine as a gateway offers several sustainable advantages such as;

* Cost reduction : Eliminates the need for physical hardware, reducing both initial and maintenance costs.
* Resource Efficiency : Virtual machines consume fewer physical resources compared to dedicated hardware devices, thereby reducing the environmental impact associated with production.
* Scalability and Flexibility : Virtual machines offer scalability without requiring additional physical hardware, promoting adaptability and reducing waste.

## Adaptive Power Consumption

Implementing intelligent power management strategies can mitigate excessive energy consumption.

* Selective Power Usage : Devices can be programmed to power off during non-critical periods, conserving energy and extending battery life.
* Automated Shutdown and Reboot : Utilising sensors or timers, the system can power down during inactive or low-usage hours and reboot during operational times, optimising energy consumption.
* Enhanced Battery Life : By controlling the device’s operational hours based on specific criteria, the overall lifespan of batteries can be extended, reducing the frequency of replacements.

## Firmware and Software Optimisation

Efficient firmware and software development practices can lead to sustainable outcomes :

* Code Recyclability : Designing modular and well-documented code structures enables the reuse of code segments or modules in future iterations or different projects. This approach reduces redundant coding efforts, promotes efficiency in development, and minimises electronic waste generated from obsolete software.
* Version Control and Repository Management: Utilising version control systems and centralised repositories allows for better code management, facilitating code reuse and collaboration among developers. It streamlines the development process and encourages the reuse of optimised, sustainable code segments.

# Conclusion

In conclusion, this project to create a model of an automated pet feeding system by amalgamating BLE connectivity, MQTT communication and hardware integration was a success.

This project showcases the potential of IoT technology in remote device applications. It demonstrates how interconnected systems can simplify routine tasks and offer greater control over out-of-reach devices. The integration of hardware, communication protocols and user interaction makes systems like this IoT pet feeder an example of innovative solutions using IoT.

Future iterations of the pet feeder could focus on expanding the functionalities, such as incorporating sensors for portion control, scheduling feeding times remotely through a dedicated application, or even integrating AI to personalise feeding patterns based on pet behaviour.

# References

|  |  |
| --- | --- |
| [1] | G. Halfacree, The Official BBC micro:bit User Guide, John Wiley & Sons, 2017. |
| [2] | ELA Innovation, "What is a BLE Gateway? : ELA Innovation," 16 December 2022. [Online]. Available: https://elainnovation.com/en/iot-ble-gateway/. [Accessed 26 November 2023]. |
| [3] | G. C. Hillar, MQTT Essentials - A Lightweight IoT Protocol, Packt Publishing, 2017. |