Design Document: Edge Compute EoN (Edge of Network) Architecture for Drone Operations

# 1. Overview

This system is designed to manage, analyze, and control agricultural drones at the edge. It enables real-time communication, OTA (Over-the-Air) updates, telemetry processing via MAVLink, edge-based analytics, and interaction with a Precision Agricultural Control System (PACS). The architecture leverages MQTT for lightweight messaging and SQLite for local persistent storage.

# 2. Architecture Components

A screenshot of a computer

AI-generated content may be incorrect.

## 2.1. MQTT Gateway (Port 5001)

Acts as the central communication hub. Publishes and subscribes to telemetry, command, and system messages from/to the drone via MQTT. Persists essential metadata and state changes in SQLite. Bridges edge services and external MQTT Broker.

## 2.2. OTA Device Management (Port 5000)

Manages lifecycle operations of drone containers/services. Interfaces with MQTT Gateway to send updates or receive events.  
Endpoints: /health, /status, /start, /stop, /restart

## 2.3. MAVLink Interface Module (Port 5002)

Converts MQTT messages to MAVLink commands and vice versa. Handles telemetry parsing and mission command execution. Maintains session with the flight controller (ArduPilot, PX4).

## 2.4. Edge Analytics (Port 5003)

Runs real-time or batch inference for drone-collected data (e.g., crop health, obstacle detection). Consumes telemetry and image data. Publishes analytics insights back to MQTT Gateway.

## 2.5. Precision Agricultural Control System (PACS) (Port 5004)

Receives analyzed data and drone status to generate action plans. Sends high-level commands to MAVLink or MQTT Gateway. Provides farmer or backend dashboard integration.

# 3. Data Flow Description

1. Telemetry Ingestion: Drone → MAVLink → MAVLink Module → MQTT Gateway → MQTT Broker / SQLite  
2. Command Execution: PACS or OTA Module → MQTT Gateway → MAVLink Module → Drone  
3. Image/Video Analysis: Drone → Edge Analytics → MQTT Gateway → PACS  
4. OTA Operations: Remote REST Client → OTA Device Manager → MQTT Gateway → Affected Module/Service

# 4. Deployment Topology

- Edge Device: Raspberry Pi 4, Jetson Nano, or x86 Industrial PC  
- All components run as Docker containers  
- MQTT Broker is on cloud-hosted  
- Persistent Storage: Local SQLite database

# 5. Communication Protocols

|  |  |  |  |
| --- | --- | --- | --- |
| Service | Protocol | Port | Purpose |
| OTA Manager | HTTP REST | 5000 | Device & container lifecycle control |
| MQTT Gateway | MQTT | 5001 | Central data exchange hub |
| MAVLink Service | MAVLink/Serial | 5002 | Drone telemetry & control |
| Edge Analytics | HTTP/MQTT | 5003 | Image/data analysis |
| PACS | HTTP/MQTT | 5004 | Farm-level decision system |
| Grafana | HTTP | 9000 | Dashboard monitor |

# 6. Security Considerations

- JWT-based API authentication for all REST services  
- TLS for MQTT   
- WiFi or 4G fallback with store-and-forward logic  
- SQLite tamper-checks or migration to SQLCipher

# 7. Logging & Monitoring

- Local logging via logrotate and SQLite  
- Optional log push to cloud  
- Health check endpoints for all services

# 8. Failure Handling & Resilience

- MQTT reconnect logic with QoS  
- SQLite-based local buffering  
- Systemd watchdog or OTA Manager service restarts  
- OTA-controlled container/module management

# 9. Future Enhancements

- Web UI Dashboard for OTA and telemetry monitoring  
- Lora/5G integration  
- RTSP-based video pipeline with frame sampling  
- Federated learning model updates from cloud to edge