

Protocol Comparison Report: MQTT, CoAP, and OPC UA in IIoT Sensor Networks

Lab 04 – IIAI 3377

Team: Andrew Kuruvilla, Marques Roverson, Ruben Valenzuela Alvarado, Ezra Bakatubia

Date: February 2026

Introduction

In this lab, we simulated temperature and humidity sensor data transmission using three prominent Industrial Internet of Things (IIoT) protocols: MQTT, CoAP, and OPC UA. Each protocol was implemented in Python to generate data every second and (for MQTT) visualize it in real time.

This report compares the protocols based on our hands-on experience and key technical characteristics, highlighting their architectures, strengths, limitations, and ideal applications in industrial settings.

Key Differences

Feature	MQTT	CoAP	OPC UA
Transport Layer	TCP (reliable, connection-oriented)	UDP (lightweight, connectionless)	TCP (reliable, with optional Pub/Sub)
Communication Model	Publish/Subscribe (broker-based)	Request/Response + Observe (RESTful)	Client/Server + Pub/Sub (information model)
Header/Message Size	Very small (~2 bytes minimum)	Small (~4 bytes minimum)	Larger (complex headers, semantics)
Reliability	High (QoS 0–2 levels, acknowledgments)	Medium (confirmable messages optional)	Very high (built-in acknowledgments, sessions)
Security	TLS + username/password	DTLS (TLS over UDP)	Built-in (certificates, encryption, authentication)
Resource Overhead	Low to medium	Very low	Higher (rich data modeling)

Scalability	Excellent (one broker handles many clients)	Good for constrained networks	Good for structured industrial networks
Complexity	Simple to implement	Simple for REST-like use	More complex (address space, namespaces)

Pros and Cons from Lab Experience

MQTT

- Pros: Extremely easy to set up (Mosquitto broker worked well after port fixes); lightweight and scalable; pub/sub model allowed seamless real-time visualization with Matplotlib. Performed reliably for continuous sensor data streaming.
- Cons: Requires a central broker; no built-in complex data modeling (we used simple JSON payloads).
- Best for: Cloud-connected monitoring, high-volume telemetry, event-driven systems (e.g., factory dashboards).

CoAP

- Pros: Very low overhead; observable resource pattern (push notifications) was efficient for simulated constrained sensors; no broker needed, direct client-server interaction. Ideal for low-power scenarios.
- Cons: UDP-based reliability is lower than TCP; harder to debug asyncio issues in our improved server implementation.
- Best for: Battery-powered edge devices, wireless sensor networks in remote or constrained industrial environments (e.g., field sensors).

OPC UA

- Pros: Rich information modeling (namespaces, variables, writable nodes); strong built-in security and reliability; felt "industrial-grade" when updating values on the server.
- Cons: More setup complexity (endpoint configuration, port 4840); higher resource use compared to the others; no simple visualization in our lab.
- Best for: Machine-to-machine communication in manufacturing, SCADA integration, secure and interoperable factory-floor systems.

Use Cases in Industrial IoT (IIoT)

- MQTT: Remote monitoring (e.g., oil/gas pipelines, smart grids), predictive maintenance telemetry to the cloud, or integrating thousands of sensors with big data/AI platforms.

- CoAP: Low-power wireless sensors in harsh environments (e.g., mining, agriculture, or asset tracking where bandwidth/power is limited).
- OPC UA: On-premise industrial automation (e.g., connecting PLCs, robots, and control systems in factories for real-time deterministic data exchange and semantic interoperability).

In hybrid setups (common in modern Industry 4.0), MQTT often handles cloud telemetry while OPC UA manages shop-floor M2M communication. Emerging trends include OPC UA over MQTT (Pub/Sub) for combining strengths.

Conclusion

Our simulations showed no single "best" protocol—choice depends on constraints:

- Use MQTT for simplicity, scalability, and cloud integration (easiest in our lab).
- Use CoAP for resource-constrained, low-power edge devices.
- Use OPC UA for secure, structured industrial environments requiring rich data semantics.

This hands-on comparison reinforced that thoughtful protocol selection is critical for efficient, reliable IIoT systems. Future work could include security additions (TLS/DTLS/certificates) and quantitative metrics (latency, bandwidth) to further validate these insights.