Avaliando o Protocolo Matrix para Ecossistemas IoT com Recursos Restritos

Rodrigo da Silva Carvalho Maia[1] **e Paulo Antonio Leal Rego**[1]

[1]Departamento de Computação
Universidade Federal do Ceará (UFC)
Fortaleza – CE – Brasil

rodrigomaia@alu.ufc.br, paulo@dc.ufc.br

Abstract. *This study investigates adapting the federated Matrix protocol via the CoMatrix gateway for resource-constrained IoT devices to overcome data sovereignty issues and single points of failure inherent in centralized architectures. Comparative tests against MQTT showed that the CoAP-Matrix model reduced network traffic by 50.5% and stabilized latencies around 200 ms, effectively mitigating Head-of-Line Blocking in high-frequency scenarios. The architecture proves technically viable for resilient applications, though it requires further adaptations for real-world deployments.*

Resumo. *Este estudo investiga a adaptação do protocolo federado Matrix, via gateway CoMatrix, para dispositivos IoT com recursos restritos, visando superar problemas de soberania de dados e pontos únicos de falha comuns em arquiteturas centralizadas. Testes comparativos com o protocolo MQTT demonstraram que o modelo híbrido CoAP-Matrix reduziu o tráfego de rede em 50.5% e estabilizou as latências na faixa de 200 ms, mitigando bloqueios em cenários de alta frequência. A arquitetura comprovou sua viabilidade técnica para aplicações resilientes, embora ainda exija adaptações para implantações em cenários reais.*

1. Introdução

O crescimento acelerado de implantações IoT intensifica um problema estrutural: a maioria dos ecossistemas concentra todo o fluxo de dados em plataformas de nuvem operadas por poucos provedores [Hazra et al. 2021, Lee et al. 2021]. Até 2035, a densidade de dispositivos conectados pode superar 100 por pessoa [Shahrokhi and Ahmadi 2023], o que torna esse ponto de concentração cada vez mais crítico. O protocolo MQTT, padrão de fato para mensageria IoT [Banks and Gupta 2014], agrava o problema ao exigir um *broker* central: qualquer interrupção nesse componente paralisa toda a rede [Hazra et al. 2021]. Plataformas de *middleware* com APIs proprietárias completam o cenário, fragmentando o ecossistema em *data silos* e limitando a autonomia dos operadores [Chataut et al. 2023].

Protocolos federados atacam essa dependência ao eliminar a autoridade central única [Linagora 2025]. O Matrix é um padrão aberto para comunicação descentralizada: diferentes organizações operam seus próprios *homeservers* e sincronizam históricos sem depender de um único intermediário [Matrix.org Foundation 2025]. O problema é que o Matrix foi projetado para a web, com suas APIs usando HTTP/HTTPS com serialização JSON, sendo esses requisitos incompatíveis com dispositivos de Classe 2 [Bormann et al. 2014], como o ESP32, que possuem dezenas de kilobytes de RAM e processamento limitado.

O projeto CoMatrix resolve essa incompatibilidade com um *gateway* intermediário [Buchberger and Kramer 2021]: os dispositivos restritos comunicam-se via CoAP com o *gateway*, que traduz as mensagens para eventos Matrix e as persiste no *homeserver* local. Este trabalho avalia a viabilidade técnica dessa abordagem em um *testbed* controlado e compara seu desempenho com a arquitetura MQTT convencional. As contribuições principais são:

- Caracterização quantitativa da latência, do consumo de recursos e da eficiência de banda do *gateway* CoMatrix sob 12 cenários de estresse (3 tamanhos de *payload* $\times$ 2 frequências $\times$ 2 repetições);
- Demonstração experimental da supressão do *Head-of-Line Blocking* (HOLB) na abordagem CoAP/Matrix em comparação com MQTT;
- Verificação de que a sobrecarga do *gateway* permanece compatível com a operação em *soft real-time* em redes LLN típicas.

O restante do artigo está organizado da seguinte forma: a Seção 2 apresenta a fundamentação teórica, a Seção 3 apresenta alguns trabalhos relacionados; a Seção 4 descreve a metodologia experimental; a Seção 5 analisa os resultados; e a Seção 6 conclui o trabalho com as limitações e direções futuras.

2. Fundamentação Teórica

Esta seção apresenta os principais conceitos teóricos necessários para a compreensão do estudo. Inicialmente, apresenta-se a taxonomia de dispositivos restritos, seguida por uma visão geral do protocolo Matrix e uma análise comparativa dos protocolos de transporte MQTT e CoAP no contexto de IoT.

2.1. Taxonomia de Dispositivos e Redes Restritas

A RFC 7228 [Bormann et al. 2014] classifica dispositivos IoT em três classes seguindo limitações de RAM e memória flash: Classe 0 ($<$10 KB RAM, $<$100 KB flash), Classe 1 ($\approx$10 KB RAM, $\approx$100 KB flash) e Classe 2 ($\approx$50 KB RAM, $\approx$250 KB flash). Dispositivos de Classe 2, como o ESP32 utilizado neste trabalho, suportam pilhas IP completas, mas impõem restrições severas ao *overhead* dos protocolos de comunicação. As redes que interconectam esses dispositivos são denominadas LLNs (*Low-power and Lossy Networks*) e caracterizam-se por baixa largura de banda, alta taxa de perda de pacotes e topologia instável [Bormann et al. 2014]. Essas propriedades tornam inviável o uso direto de protocolos projetados para redes cabeadas de alto desempenho.

2.2. O Protocolo Matrix e a Arquitetura Federada

O Matrix é um padrão aberto para comunicação descentralizada baseado em Grafos Acíclicos Dirigidos (DAGs) [Matrix.org Foundation 2025]. Cada organização opera um *homeserver* independente que sincroniza o histórico de eventos de salas virtuais com outros servidores sem depender de uma autoridade central [Matrix.org Foundation 2025]. A consistência eventual dos dados é garantida pelas DAGs, que permitem a convergência dos históricos mesmo após particionamentos de rede.

A API nativa do Matrix utiliza chamadas *RESTful* sobre HTTP e serialização JSON [Martins et al. 2026]. Esses requisitos são incompatíveis com dispositivos de Classe 2,

pois o *overhead* de estabelecimento de conexão TCP e a verbosidade do JSON superam as capacidades de processamento e memória desses dispositivos, tornando inviável a implementação nativa na borda.

2.3. Protocolos de Transporte para IoT: MQTT e CoAP

O MQTT consolidou-se como padrão de mercado para IoT por combinar um modelo publicador/assinante com cabeçalho fixo de apenas 2 bytes. Por operar sobre TCP, o protocolo herda o fenômeno de *Head-of-Line Blocking* (HOLB): a perda de um único segmento bloqueia a entrega de todos os pacotes subsequentes até que a retransmissão seja concluída [Tanenbaum and Wetherall 2011]. Em LLNs com perda de pacotes elevada, esse comportamento resulta em picos de latência de múltiplos segundos.

O CoAP foi projetado especificamente para dispositivos restritos e redes LLN [Shelby et al. 2014]. Operando sobre UDP, o protocolo elimina o HOLB e reduz o *overhead* de gerenciamento de conexão. Seu modelo REST facilita a integração com serviços web por meio de *proxies*, e sua codificação binária compacta é compatível com as restrições de Classe 2. Datta e Bonnet [Datta and Bonnet 2021] avaliaram o *overhead* de segurança de CoAP e MQTT em dispositivos restritos e confirmaram que o CoAP mantém latência inferior ao MQTT em cenários de alta perda, consolidando-o como protocolo preferido para a borda em LLNs.

2.4. A Arquitetura CoMatrix

O projeto CoMatrix introduz um *gateway* intermediário para desacoplar os dispositivos de borda da complexidade do protocolo Matrix [Buchberger and Kramer 2021]. Na borda da rede, os dispositivos (identificados como *Cliente CoMatrix* na Figura 1) coletam dados de sensores físicos e enviam mensagens via CoAP/CBOR ao *gateway*. O *gateway* realiza a tradução semântica dessas mensagens para eventos Matrix em formato HTTP/JSON e os persiste no *homeserver* local. Com esse desacoplamento, a borda opera com protocolo leve enquanto o *gateway* absorve o gerenciamento de *tokens*, sessões e criptografia exigidos pelo Matrix. A Figura 1 ilustra como esses componentes se conectam na prática.

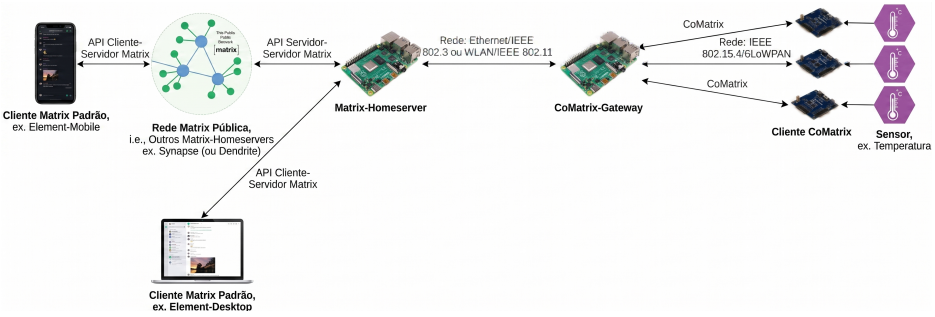

Figura 1. Arquitetura híbrida do CoMatrix.

3. Trabalhos Relacionados

A seleção de protocolo de mensageria em arquiteturas IoT híbridas é extensamente estudada na literatura. [Naik 2017] compara MQTT, CoAP, AMQP e HTTP em termos de *overhead*, confiabilidade e adequação a diferentes cenários. Os autores mostram que o AMQP oferece semântica de fila robusta e suporte à federação entre *brokers*, mas introduz *overhead* de

banda e latência superiores ao MQTT em redes restritas. Esse resultado justifica a exclusão do AMQP como protocolo de borda no presente trabalho e direciona a escolha para o CoAP.

No que diz respeito ao *overhead* de *gateway*, um *framework* adaptativo recente [Żatuchin and Azarskov 2025] avaliou MQTT, CoAP e HTTP sob condições variáveis de perda de pacotes (0–20%), latência e largura de banda. Os resultados mostram que o CoAP supera o MQTT em eficiência energética sob alta perda, mas o MQTT é preferível em transmissões de alta frequência com rede estável. O CoMatrix emprega uma tradução estática entre CoAP e Matrix, sem mecanismos de ajuste automático às variações de rede. Embora essa rigidez limite a adaptabilidade do sistema, ela é necessária para garantir a persistência e a sincronização do histórico na arquitetura federada.

Para protocolos *broker-less*, Profanter et al. [Profanter et al. 2019] comparam DDS, OPC UA, ROS e MQTT em aplicações industriais, medindo RTT e *overhead* de rede sob diferentes cargas de CPU. Embora o DDS elimine o *broker* central, sua arquitetura não prevê persistência distribuída de histórico de eventos nem federação interorganizacional, capacidades nativas do Matrix via DAGs e *homeservers* independentes. Essa limitação estrutural justifica a escolha do Matrix como protocolo federado neste trabalho.

O presente trabalho se diferencia das abordagens anteriores ao introduzir o *gateway* como camada de tradução semântica entre CoAP e um protocolo de federação com histórico persistente. A avaliação quantitativa do *overhead* dessa tradução, em termos de latência, consumo de recursos e eficiência de banda, sob diferentes níveis de *payload* e frequência preenche uma lacuna não endereçada pelos trabalhos citados.

4. Metodologia

O procedimento experimental foi conduzido para comparar o desempenho da arquitetura híbrida CoAP-Matrix com um modelo MQTT convencional. A pesquisa seguiu quatro etapas principais: adaptação da arquitetura CoMatrix; modelagem do cenário experimental; montagem do *testbed*; execução e análise dos resultados.

4.1. Cenário de Experimentação (*Testbed*)

O ponto de partida para a elaboração dos experimentos foi o estudo do código-fonte do Matrix, com o objetivo de compreender o funcionamento e realizar adaptações para o ambiente controlado. Para garantir a reprodutibilidade dos resultados, os testes foram realizados em um cenário controlado. A arquitetura do *testbed* consistiu em dois caminhos de comunicação paralelos que convergem em um sistema de monitoramento unificado. A Figura 2 apresenta o mapeamento de ambos.

De maneira simplificada, o caminho de cada cenário é descrito da seguinte forma:

1. MQTT: O dispositivo envia dados diretamente para um *broker* Mosquitto via TCP.
2. CoAP-Matrix: O dispositivo comunica-se via CoAP (UDP) com um *script gateway* Python, que traduz as requisições para a API do *homeserver* Matrix Synapse via HTTP.

A topologia de rede estrela foi adotada e implementada sobre uma infraestrutura Wi-Fi local, apresentando latência de comunicação entre 2 e 5 ms, o que implica um impacto desprezível no desempenho dos protocolos em nível de aplicação.

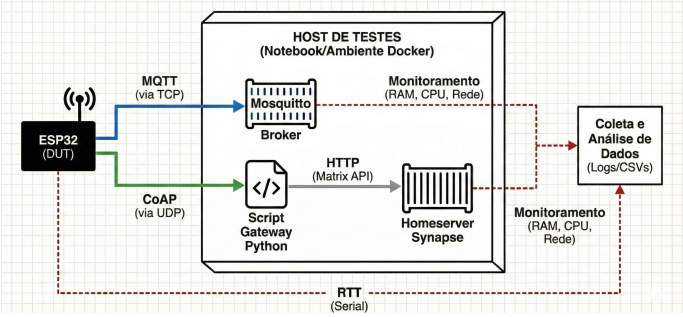

Figura 2. Arquitetura do *testbed* de protocolos.

4.2. Implementação de Hardware e Software

O *Device Under Test* (DUT) utilizado foi o microcontrolador ESP32-WROOM-32 (Classe 2), com processador dual-core de 240 MHz e 520 KB de SRAM. O Servidor *gateway* foi um notebook Dell Inspiron (Intel Core i7, 16 GB RAM) executando os serviços em contêineres via Docker. Para a persistência de dados, o *homeserver* Matrix Synapse foi configurado utilizando o banco de dados padrão SQLite. Utilizamos um dispositivo com recursos computacionais avantajados para evitar gargalos de processamento no lado do servidor. Os firmwares do ESP32 foram implementados em C++, sendo esses clientes os que geram *payloads* fixos, visando garantir isonomia na comparação.

4.3. Design da Experimentação e Métricas

A variável independente é a arquitetura de comunicação. Os fatores manipulados incluíram três níveis de *payload* (16, 64 e 256 bytes) e dois níveis de frequência de transmissão (1 Hz (nominal) e 5 Hz (estresse)), totalizando 12 cenários experimentais. As métricas avaliadas foram:

1. Latência (RTT): Tempo entre o envio da mensagem e o recebimento da confirmação (PUBACK no MQTT ou resposta CoAP), medido diretamente no firmware do ESP32.
2. Consumo de Recursos (RAM e CPU): Medido nos processos do Mosquitto e Synapse via comando de aferição de estatísticas.
3. Eficiência de Rede: Calculada pelo volume total de bytes trafegados e pelo *overhead* percentual em relação à carga útil confirmada.

Cada cenário foi executado por 600 segundos, precedido por uma fase de aquecimento (*warm-up*) de 120 segundos para garantir a estabilização das conexões e a sincronização inicial do *homeserver* no cenário do CoAP-Matrix. Foram realizadas duas execuções para cada experimento, coletando-se aproximadamente 3.000 medições por execução, o que resultou em cerca de 6.000 amostras por cenário para análise estatística.

Para a análise de significância estatística das médias de latência, aplicamos o teste t de Welch com nível de significância $\alpha = 0,05$. A escolha deste teste justifica-se pela presença de de variâncias desiguais nos dados, uma vez que o protocolo MQTT apresentou desvios-padrão superiores aos do CoAP-Matrix em cenários de estresse. Complementarmente, calculamos o d de Cohen para quantificar o tamanho do efeito das diferenças observadas.

5. Resultados

Nesta seção, apresentamos os dados obtidos no *testbed* sob as óticas de consumo de hardware, latência e eficiência de rede.

5.1. Análise de Consumo de Recursos

O uso de recursos no *gateway* CoMatrix é o principal fator de viabilidade para a arquitetura. Com base nos resultados apresentados na tabela 1, observamos que o *homeserver* Matrix Synapse consome, em média, 130 MB de RAM em operação, valor aproximadamente 100 vezes superior ao exigido pelo *broker* Mosquitto, de cerca de 1,5 MB. Esse *overhead* do Synapse deve-se à complexidade do modelo de dados (DAG de eventos) e ao uso da linguagem Python no servidor.

Quanto à CPU, ambos os serviços apresentaram baixo impacto, com o Synapse variando entre 0,42% e 1,84% nos cenários de estresse, enquanto o MQTT permaneceu abaixo de 0,2%. Esses valores indicam que, embora o custo de entrada na federação seja mais elevado em termos de memória, ainda assim é compatível com hardwares de borda modernos, como o Raspberry Pi 4.

Tabela 1. Consumo de recursos entre Mosquitto (MQTT) e Synapse (Matrix).

Sistema/Protocolo	Memória RAM (MB)			Uso de CPU (%)		
	Idle	1 Hz	5 Hz	Idle	1 Hz	5 Hz
Mosquitto (MQTT)	1,25	1,44	1,48	0,03%	0,06%	0,15%
Synapse (Matrix)	124,41	130,92	129,31	0,42%	1,51%	1,84%

5.2. Análise de Latência (*Round-Trip Time*)

A análise de latência revelou um comportamento de inversão do desempenho em função da frequência, conforme detalhado na Tabela 2. Em baixa frequência (1 Hz), o MQTT apresentou latências médias inferiores (de 115,2 a 163,9 ms), refletindo a eficiência da conexão TCP persistente e o baixo *overhead* de processamento do *broker*. O CoMatrix registrou médias superiores ($\approx$ 500 ms) devido ao custo fixo de tradução semântica e ao mapeamento de métodos REST para o sistema de eventos do Matrix realizado no *gateway*.

Contudo, a alta frequência (5 Hz) indica uma degradação crítica no MQTT. Mesmo com perda de rede mínima, o fenômeno de HOLB gerou picos de latência superiores a 9 segundos. De maneira oposta, o CoAP-Matrix manteve-se estável com latências máximas de 200 ms, evidenciando que a abordagem baseada em UDP é mais resiliente para redes IoT densas.

Tabela 2. Resultados Consolidados de Latência (1 Hz e 5 Hz).

Payload	Prot.	Média	IC 95%	Máx	Desvio	p-val	Efeito
Frequência de 1 Hz							
16B	MQTT	115,2	[100,5; 129,9]	831,2	159,6	$< .001$	2,14
16B	CoMatrix	489,1	[471,8; 506,4]	932,4	188,5		
256B	MQTT	118,9	[104,1; 133,8]	973,3	165,6	$< .001$	2,24
256B	CoMatrix	510,5	[494,0; 527,0]	845,6	183,8		
Frequência de 5 Hz							
16B	MQTT	87,5	[75,9; 99,1]	7.574,3	258,0	0,139	0,04
16B	CoMatrix	96,8	[92,8; 100,7]	200,3	63,9		
256B	MQTT	76,2	[55,9; 96,4]	9.379,1	409,2	0,036	0,07
256B	CoMatrix	98,2	[94,1; 102,3]	200,4	65,6		

5.3. Eficiência e Utilização de Banda

A arquitetura híbrida CoAP-Matrix demonstrou uma vantagem expressiva na preservação da banda de rede. No cenário mais exigente (256B a 5 Hz), a abordagem reduziu o volume total de tráfego em 50,5% em relação ao MQTT puro (0,81 MB vs 1,63 MB). Essa economia deve-se à natureza otimizada do transporte UDP e ao cabeçalho compacto do CoAP, projetado especificamente para LLNs.

Os resultados da Tabela 3 indicam um baixo índice de confirmações recebidas pelo CoAP no cenário de estresse (982 de 2.400). Ressalta-se que o protocolo foi configurado com mensagens confirmáveis (CON). Dado que a perda de pacotes na rede foi inferior a 0,5% em todos os testes, o baixo recebimento de confirmações evidencia que o gargalo não reside no canal de comunicação, mas na capacidade de processamento do *homeserver* Matrix Synapse, que falha em processar a persistência de eventos nas DAGs a um ritmo acelerado de 5 Hz, resultando em timeouts na camada de aplicação.

Ademais, observa-se que o CoAP apresenta um *overhead* percentual superior ao do MQTT (+3,8 pp). Vale ressaltar que essa métrica pode induzir a uma interpretação equivocada, pois penaliza o CoAP devido à incapacidade do servidor Synapse em confirmar 59% das mensagens enviadas sob estresse. Como o cálculo de eficiência relativa contabiliza apenas a carga útil de mensagens confirmadas, o volume trafegado pelas mensagens que sofreram *timeout* no servidor é computado integralmente como *overhead*. Entretanto, ao analisar o volume absoluto, a abordagem CoAP reduziu o tráfego total em 50,5%, provando sua economia para a infraestrutura de rede, independentemente do gargalo de processamento no *homeserver*.

Tabela 3. Análise Volumétrica e de *Overhead*.

Métrica	MQTT (TCP)	CoAP (UDP)	Variação
Mensagens Planejadas	2.400	2.400	-
Dados Enviados pelo ESP32	2.226	2.375	+6,7%
Confirmações Recebidas (ACK/CON)	2.226	982	-55,9%
Volume Total Trafegado	1,63 MB	0,81 MB	-50,5%
Overhead Percentual	64,9%	68,7%	+3,8 pp
Volume Médio / Msg Confirmada	**767,8 B**	**864,9 B**	**+12,6%**

* O *overhead* percentual do CoAP é penalizado por mensagens enviadas mas não confirmadas pelo servidor.

6. Conclusão e Trabalhos Futuros

Os resultados evidenciam que o CoMatrix é tecnicamente viável para aplicações de tempo *real soft*, como o monitoramento residencial, em que a estabilidade do tempo de resposta é prioritária em relação à latência mínima. O custo fixo de tradução gerado no *gateway* é justificado pelos ganhos em soberania de dados, interoperabilidade federada e resiliência sob estresse. Os resultados também indicam que a abordagem federada é viável, destacando-se pela estabilidade temporal em redes sob estresse, embora implique custos operacionais superiores na infraestrutura de borda. Trabalhos futuros devem explorar o impacto da implementação de segurança fim a fim em dispositivos restritos, além de analisar a escalabilidade do modelo em topologias *multi-hop* e o consumo energético direto no hardware. Além disso, futuras pesquisas podem analisar o custo imposto pela federação, realizando experimentos em cenários controlados em que dois ou mais *homeservers* possuem usuários que participam da mesma sala. Tal trabalho deve identificar o impacto do processo de persistência de eventos das DAGs em um cenário de federação efetiva.

Referências

Banks, A. and Gupta, R. (2014). MQTT version 3.1.1: OASIS standard. Technical report, OASIS. Disponível em: `http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html`. Acesso em: 2026-03-13.

Bormann, C., Ersue, M., and Keranen, A. (2014). Terminology for constrained-node networks. Technical Report RFC 7228, IETF.

Buchberger, T. and Kramer, I. (2021). CoMatrix: How to overcome kernel panic, deal with buffer limitations and LOL at random. In *RIOT Summit 2021*. Disponível em: `https://summit.riot-os.org/2021/wp-content/uploads/sites/16/2021/09/s03-02.pdf`.

Chataut, R., Phoummalayvane, A., and Akl, R. (2023). Unleashing the power of IoT: A comprehensive review of iot applications and future prospects in healthcare, agriculture, smart homes, smart cities, and industry 4.0. *Sensors*, 23(16):7194.

Datta, S. K. and Bonnet, C. (2021). Performance evaluation of CoAP and MQTT with security support for IoT environments. *Computer Networks*, 196:108201.

Hazra, A. et al. (2021). A comprehensive survey on interoperability for IIoT: Taxonomy, standards, and future directions. *ACM Computing Surveys*, 55(1):1–35.

Lee, E. et al. (2021). A survey on standards for interoperability and security in the internet of things. *IEEE Communications Surveys & Tutorials*, 23(2):1020–1047.

Linagora (2025). Matrix protocol: Secure decentralised communication. Disponível em: `https://linagora.com/en/topics/matrix-protocol`. Acesso em: 2026-03-12.

Martins, J. A., Rego, P. A., de Macêdo, J. A., Silva, F. A., and Lagrota, V. (2026). Matrix protocol: a comprehensive systematic mapping study. *Journal of Cloud Computing*, 15(1):20.

Matrix.org Foundation (2025). Matrix specification. Disponível em: `https://spec.matrix.org/`. Acesso em: 2026-03-11.

Naik, N. (2017). Choice of effective messaging protocols for iot systems: Mqtt, coap, amqp and http. In *2017 IEEE international systems engineering symposium (ISSE)*, pages 1–7. IEEE.

Profanter, S., Tekat, A., Dorofeev, K., Rickert, M., and Knoll, A. (2019). OPC UA versus ROS, DDS, and MQTT: Performance evaluation of Industry 4.0 protocols. In *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*.

Shahrokhi, A. and Ahmadi, M. (2023). Power evaluation of iot application layer protocols. In *2023 7th International Conference on Internet of Things and Applications (IoT)*, pages 1–7. IEEE.

Shelby, Z., Hartke, K., and Bormann, C. (2014). The constrained application protocol (CoAP). Technical Report RFC 7252, IETF.

Tanenbaum, A. S. and Wetherall, D. J. (2011). *Computer Networks*. Prentice Hall, Upper Saddle River, 5th edition.

Żatuchin, D. and Azarskov, M. (2025). An adaptive protocol selection framework for energy-efficient iot communication: Dynamic optimization through context-aware decision making. In *Informatics*, volume 12, page 125. MDPI.