NGHIÊN CỨU VÀ ỨNG DỤNG NỀN TẢNG ĐIỆN TOÁN BIÊN MÃ NGUỒN MỞ EDGEX FOUNDRY CHO HỆ THỐNG ĐIỀU KHIỂN - GIÁM SÁT IOT DỰA TRÊN LORA

Phan Văn Hải [1*]***, Trịnh Thị Hậu*** [1]
[*]***Tác giả liên hệ, Email: pvhai12@hou.edu.vn***

Ngày tòa soạn nhận được bài báo: 02/06/2025
Ngày phản biện đánh giá: 08/09/2025
Ngày bài báo được duyệt đăng: 15/10/2025

DOI: 10.59266/houjs.2025.734

Tóm tắt: *Bài báo này trình bày về một nghiên cứu và triển khai giải pháp hệ thống Internet vạn vật (IoT) điều khiển - giám sát, dựa trên công nghệ LoRa và nền tảng điện toán biên mã nguồn mở EdgeX Foundry. Mục tiêu cốt lõi của công trình là xây dựng một hệ thống có khả năng thu thập, lưu trữ, xử lý và tự động thực thi các quy tắc nghiệp vụ ngay tại biên, đồng thời dễ dàng tích hợp với các dịch vụ đám mây. Chúng tôi đã phát triển một dịch vụ thiết bị LoRa tùy chỉnh, được tích hợp vào EdgeX, để đảm bảo khả năng kết nối với các thiết bị LoRa chuyên biệt. Kết quả từ thực nghiệm cho thấy hệ thống hoạt động ổn định, với dữ liệu được thu thập và xử lý một cách hiệu quả; độ trễ trung bình cho các tác vụ tại biên được ghi nhận dưới 30ms. Hơn nữa, việc triển khai hệ thống trên Raspberry Pi đã minh chứng tính khả thi và hiệu suất cao của giải pháp điện toán biên trong các môi trường IoT có tài nguyên hạn chế, thể hiện qua mức tiêu thụ CPU và RAM rất thấp.*

Từ khóa: *EdgeX Foundry, LoRa, IoT, điện toán biên, điều khiển - giám sát, dịch vụ thiết bị*

I. Đặt vấn đề

Internet vạn vật (IoT) đang phát triển rất nhanh và tạo ra một lượng lớn dữ liệu rất lớn từ hàng tỷ thiết bị kết nối (Kong et al., 2022; "State of IoT 2024," 2024). Tuy nhiên, việc truyền tải toàn bộ dữ liệu này lên đám mây để xử lý sẽ dẫn đến các thách thức đáng kể về độ trễ, băng thông mạng và vấn đề bảo mật dữ liệu (Milani et al., 2023; Zhong & Nie, 2024). Trong khi đó, nhiều ứng dụng IoT như điều khiển thời gian thực, giám sát y tế hoặc công nghiệp, độ trễ thấp là một yếu tố then chốt (Andriulo et al., 2024).

Điện toán biên (Edge Computing) đã phát triển như một giải pháp hiệu quả để giải quyết những hạn chế này. Khi năng lực xử lý và ra quyết định được đưa gần hơn tới nguồn phát sinh dữ liệu, công nghệ điện toán biên góp phần đáng

[1] Trường Đại học Mở Hà Nội

kể vào việc giảm thiểu độ trễ, tối ưu hóa băng thông mạng và nâng cao tính an toàn thông tin (Andriulo et al., 2024; Milani et al., 2023; Zhong & Nie, 2024). EdgeX Foundry (Foundation, n.d.) là một nền tảng điện toán biên mã nguồn mở linh hoạt và dễ mở rộng cho việc xây dựng các ứng dụng biên, được phát triển bởi Linux Foundation. Tuy nhiên, mặc dù EdgeX Foundry cung cấp một hệ sinh thái phong phú các Device Service cho nhiều giao thức phổ biến như Modbus, BACnet, OPC-UA, MQTT, hay SNMP (EdgeX Foundry Project, n.d.) nhưng hiện tại chưa có một Device Service chính thức hoặc được hỗ trợ rộng rãi dành riêng cho công nghệ LoRa.

Nghiên cứu hiện tại tập trung vào việc tích hợp EdgeX Foundry với công nghệ truyền thông LoRa (Long Range) – một phương thức truyền dẫn không dây tiêu thụ ít năng lượng và có tầm phủ sóng rộng, được ứng dụng phổ biến trong các hệ thống IoT (Gaffurini et al., 2024; Milani et al., 2023; Zhong & Nie, 2024). Các mục tiêu trọng tâm của công trình này bao gồm: (1) xây dựng một LoRa Device Service tùy chỉnh nhằm giao tiếp với các thiết bị LoRa chuyên biệt; (2) thực hiện và đánh giá năng lực của hệ thống trong việc thu thập, lưu trữ, chuyển tiếp dữ liệu và xử lý các quy tắc nghiệp vụ trực tiếp tại biên thông qua EdgeX Foundry; (3) đo lường định lượng hiệu suất của hệ thống về độ trễ và mức tiêu thụ tài nguyên. Phạm vi của nghiên cứu này giới hạn ở các khía cạnh kết nối và xử lý dữ liệu, không đi sâu vào chi tiết các vấn đề bảo mật.

Bài báo sẽ tiếp tục trình bày chi tiết về kiến trúc tổng thể, quá trình triển khai hệ thống và phân tích các kết quả thực nghiệm nhằm làm rõ hiệu quả cũng như tiềm năng ứng dụng của giải pháp đề xuất.

II. Cơ sở lý thuyết và thiết kế hệ thống

2.1. Kiến trúc nền tảng điện toán biên EdgeX Foundry

EdgeX Foundry là một khung vi mô (framework microservice) mã nguồn mở được thiết kế để chuẩn hóa kết nối các thiết bị IoT tại biên (Foundation, n.d.). Kiến trúc của EdgeX được tổ chức thành bốn lớp chính:

- Lớp Device Services (Dịch vụ Thiết bị): Lớp dưới cùng, có chức năng giao tiếp với các thiết bị. Nó đóng vai trò là lớp trừu tượng hóa cho các loại giao thức truyền tin khác nhau.

- Lớp Core Services (Dịch vụ Lõi): Có các chức năng cốt lõi như lưu trữ dữ liệu (Core Data), quản lý thiết bị (Core Metadata), lưu trữ cấu hình (Registry & Config) và xử lý lệnh điều khiển (Core Command).

- Lớp Supporting Services (Dịch vụ Hỗ trợ): Có các dịch vụ như lập lịch (Scheduler), thông báo (Notifications) và xử lý luật nghiệp vụ (Rule Engine) để thực hiện các phân tích và tác vụ cục bộ.

- Lớp Application Services (Dịch vụ Ứng dụng): Lớp trên cùng, chịu trách nhiệm chuyển đổi và xuất dữ liệu biên sang các nền tảng đám mây, các ứng dụng doanh nghiệp hoặc các giao thức khác (ví dụ: MQTT, HTTP) (Foundation, n.d.).

2.2. Công nghệ truyền thông LoRa

LoRa là một công nghệ điều chế vô tuyến cung cấp khả năng truyền thông tầm xa với công suất thấp (*LoRa PHY | Semtech - What Is Lora | Semtech*, n.d.; "(PDF) A Study of LoRa," 2025). Các dải tần số mà nó hoạt động không cần cấp phép (ví dụ: 433 MHz, 868 MHz, 915 MHz). Do đó, chúng tôi đã chọn LoRa và triển khai một mô hình mạng hình sao cục bộ (local star topology). Theo đó, các thiết bị cuối LoRa (End-Devices) hay còn gọi là nút (Node) giao tiếp trực tiếp với một cổng LoRa (Gateway) tại biên, nơi sẽ sử dụng EdgeX Foundry để trao đổi và xử lý dữ liệu. Mô hình này giúp giảm thiểu độ phức tạp của hệ thống và tối ưu hóa độ trễ cho các tác vụ xử lý tại biên.

2.3. Định nghĩa cấu trúc gói tin truyền - nhận LoRa

Chúng tôi đã định nghĩa một cấu trúc gói tin cho việc truyền - nhận giữa các Node LoRa và LoRa Device Service. Cấu trúc gói tin gồm các trường sau:

- Header Byte: Kích thước 1 byte, là byte đầu tiên của gói tin với giá trị cố định là 0x55.

- Length: Kích thước 1 byte, chỉ tổng độ dài của phần dữ liệu nằm sau byte này.

- Device ID: Kích thước 6 byte, dùng để định danh thiết bị (Node).

- Danh sách các mã lệnh - dữ liệu (Command - Data):

- Command: Kích thước 1 byte, thể hiện mã lệnh hoặc loại thông tin.

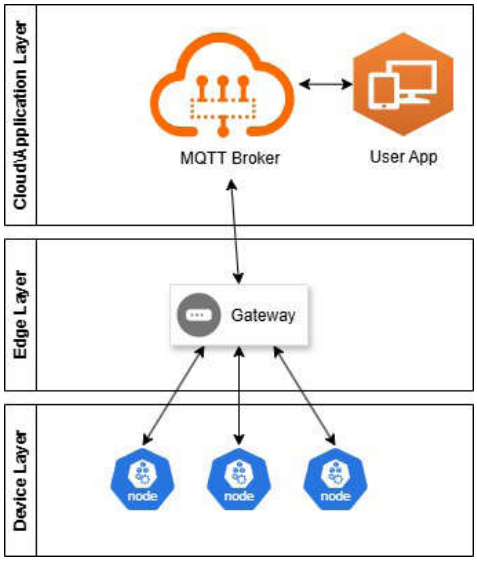
- Data: Kích thước tùy thuộc vào Command, chứa dữ liệu của lệnh.

Các mã lệnh (Command Codes) được xác định trước và dùng chung trong toàn hệ thống. Ví dụ:

- 0x11, 0x12, 0x13: Tương ứng là các lệnh báo cáo thông tin nhiệt độ, độ ẩm, ánh sáng.

- 0x32, 0x35: Tương ứng là các lệnh thay đổi trạng thái rơ-le 1, rơ-le 2

2.4. Kiến trúc hệ thống



Hình 1. Kiến trúc của hệ thống đề xuất

Kiến trúc của hệ thống đề xuất gồm ba lớp như ở Hình 1. Trong đó:

- Lớp Thiết bị (Device Layer): Gồm các thiết bị cảm biến, chấp hành tại hiện trường (cụ thể trong thí nghiệm là 2 thiết bị cảm biến và 1 thiết bị chấp hành). Ngoài ra, trong mạng LoRa, các thiết bị sẽ đóng vai trò là các Node LoRa.

- Lớp Biên (Edge Layer): Gồm một thiết bị Gateway biên, được triển khai trên phần cứng Raspberry Pi 4, và phần mềm sử dụng nền tảng EdgeX Foundry cùng với LoRa Device Service tùy chỉnh. Gateway cũng hỗ trợ mạng LoRa để có

thể giao tiếp với các Node ở lớp Thiết bị. Đây là trung tâm của hệ thống, chịu trách nhiệm thu thập, giải mã, lưu trữ dữ liệu cục bộ, thực thi các luật nghiệp vụ và điều khiển thiết bị tại chỗ.

- Lớp Đám mây/Ứng dụng (Cloud/ Application Layer): Bao gồm một MQTT Broker (EMQX) trên đám mây để nhận dữ liệu và thông báo từ Gateway thông qua giao thức MQTT. Lớp này có thể tích hợp với các ứng dụng lưu trữ, phân tích dữ liệu dài hạn hoặc giao diện người dùng.

Luồng dữ liệu trong hệ thống diễn ra như sau: Dữ liệu từ các thiết bị cảm biến LoRa được gửi không dây đến Gateway biên. LoRa Device Service trong EdgeX nhận và giải mã gói tin theo cấu trúc gói tin đã trình bày ở trên, sau đó chuyển tiếp dữ liệu đến Core Data để lưu trữ. Core Data có thể được truy vấn thông qua API để lấy dữ liệu lịch sử. Đồng thời, dữ liệu được chuyển tiếp lên đám mây thông qua Application Service, cũng như được chuyển đến Rule Engine để xử lý luật nghiệp vụ. Kết quả của việc xử lý luật hoặc các lệnh điều khiển từ xa được gửi lại thiết bị thông qua LoRa Device Service, hoặc được chuyển tiếp lên đám mây thông qua Application Service.

III. Phương pháp nghiên cứu và triển khai hệ thống

3.1. Thiết bị và phần mềm sử dụng

3.1.1. Phần cứng:

Gateway biên: Raspberry Pi 4 Model B (4GB RAM) kết nối với mô-đun LoRa SX1278 qua bộ chuyển đổi USB-to-TTL.

Node cảm biến 1: Vi điều khiển (dùng Arduino) kết nối với mô-đun LoRa SX1278 và cảm DHT21, đặt tên trong hệ thống là CB-NhietDo-DoAm.

Node cảm biến 2: Vi điều khiển (dùng Arduino) kết nối với mô-mô-đun LoRa SX1278 và cảm biến BH1750 , đặt tên trong hệ thống là CB-AnhSang.

Node chấp hành: Vi điều khiển (dùng Arduino) kết nối với mô-đun LoRa SX1278 và 2 bóng LED minh họa cho 2 rơ-le, đặt tên trong hệ thống là Relay2Kenh.

3.1.2. Phần mềm:

Hệ điều hành: Raspberry Pi OS (trên Raspberry Pi).

Nền tảng Container: Docker và Docker Compose được sử dụng để triển khai các thành phần vi dịch vụ của EdgeX Foundry một cách linh hoạt và độc lập. Docker là nền tảng container hóa phổ biến, cho phép đóng gói ứng dụng và môi trường của nó vào các container nhẹ, trong khi Docker Compose hỗ trợ định nghĩa và vận hành các nhóm container phối hợp theo cấu trúc microservices (*Docker Compose*, 100 C.E.; *What Is Docker?*, 200 C.E.). Trong nhiều nghiên cứu gần đây, container hóa bằng Docker đã được chứng minh là giải pháp hiệu quả cho triển khai hệ thống IoT và điện toán biên (Hitimana et al., 2022).

Nền tảng Điện toán biên: EdgeX Foundry (phiên bản 3.1.1).

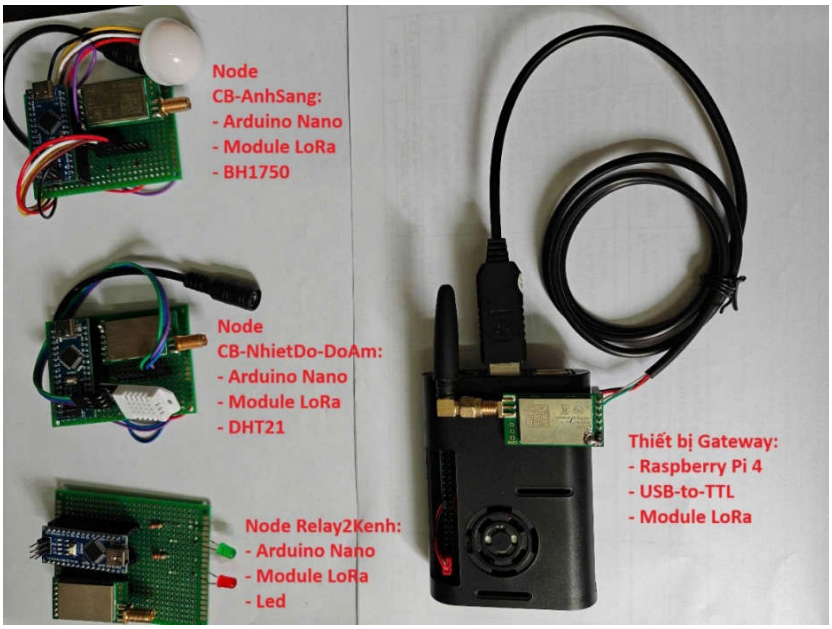
Ngôn ngữ phát triển Device Service: Go.

Công cụ kiểm thử API: Postman.

MQTT Broker: EMQX Broker (*EMQX Overview | EMQX Docs*, n.d.) (triển khai trên đám mây).

MQTT Client: MQTTX (để giám sát dữ liệu trên đám mây).

Tổng hợp các thiết bị có trong hệ thống được minh họa ở Hình 2.



Hình 2. Tổng hợp phần cứng của hệ thống

3.2. Triển khai LoRa Device Service

Để tích hợp các thiết bị LoRa vào EdgeX Foundry, chúng tôi triển khai một LoRa Device Service tùy chỉnh. Device Service này được viết dựa trên Device-SDK-Go (*Edgexfoundry/Device-Sdk-Go*, 2018/2025) của EdgeX Foundry và có các chức năng sau:

- Giao tiếp với mô-đun LoRa SX1278 thông qua cổng USB của Raspberry Pi.

- Giải mã các gói tin LoRa nhận được theo cấu trúc gói tin tùy chỉnh đã định nghĩa, chuyển đổi dữ liệu thô thành các Reading (là kiểu dữ liệu của EdgeX).

- Gửi dữ liệu đã giải mã đến Core Data Service.

- Tiếp nhận các lệnh điều khiển từ Core Command Service của EdgeX và mã hóa chúng thành gói tin LoRa để gửi đến các thiết bị chấp hành (ở đây là Relay2Kenh).

- Định nghĩa và tạo các thông tin về các loại dữ liệu, tên, thuộc tính tương ứng cho các Node LoRa (CB-NhietDo-DoAm, CB-AnhSang, Relay2Kenh) trong EdgeX Foundry để mô tả các dữ liệu và lệnh mà chúng hỗ trợ.

3.3. Triển khai các Node Lora

Các Node LoRa được triển khai sử dụng Arduino và mô-đun LoRa SX1278. Mỗi node được lập trình với firmware riêng biệt:

- Node CB-NhietDo-DoAm: Đọc thông tin nhiệt độ, độ ẩm từ DHT21, đóng gói dữ liệu theo cấu trúc và gửi định kỳ qua LoRa.

- Node CB-AnhSang: Đọc thông tin cường độ ánh sáng từ BH1750, đóng gói dữ liệu theo cấu trúc và gửi định kỳ qua LoRa.

- Node Relay2Kenh: Nhận các gói tin LoRa chứa lệnh điều khiển. Khi nhận được lệnh hợp lệ (ví dụ: bật rơ-le 2), nó sẽ điều khiển trạng thái của bóng LED tương ứng, sau đó gửi gói tin báo trạng thái để phản hồi.

IV. Kết quả và đánh giá

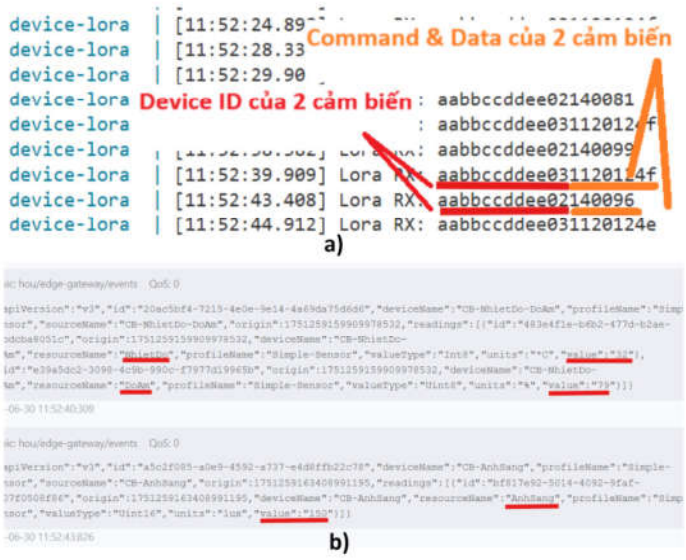
4.1. Thiết lập môi trường

Hệ thống được triển khai trên một Raspberry Pi 4 đóng vai trò là Gateway biên. Các Node LoRa được đặt trong môi trường thử nghiệm gần Gateway để đảm bảo kết nối ổn định. EdgeX Foundry và LoRa Device Service được cài đặt bằng Docker Compose. EMQX Broker được sử dụng làm MQTT Broker trên một dịch vụ đám mây công cộng.

4.2. Kết quả thử nghiệm và đánh giá

4.2.1. Kịch bản 1: Thu thập dữ liệu và chuyển tiếp lên đám mây

Kịch bản này nhằm đánh giá khả năng thu thập dữ liệu từ cảm biến DHT21 và BH1750, sau đó chuyển tiếp lên EMQX Broker. Kết quả cho thấy hệ thống thu thập dữ liệu thành công từ Node CB-NhietDo-DoAm và CB-AnhSang. LoRa Device Service đã giải mã chính xác các gói tin LoRa và đẩy dữ liệu chuẩn hóa vào Core Data. Đồng thời, Application Service của EdgeX đã chuyển tiếp dữ liệu này lên EMQX Broker trên đám mây.

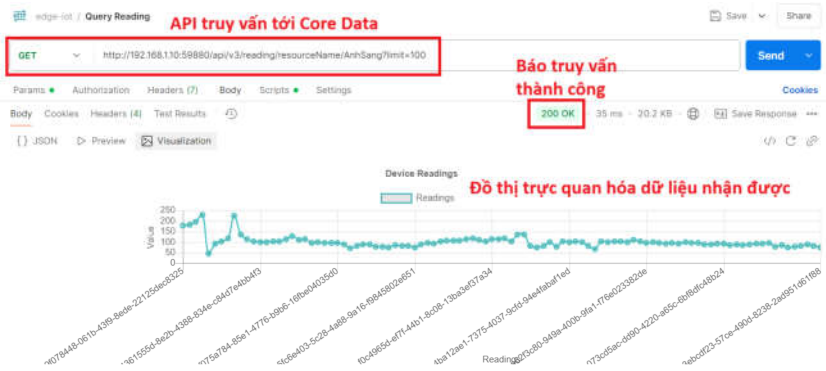


Hình 3. a) Nhật ký của Device Service khi nhận dữ liệu từ các cảm biến. b) Giao diện MQTTX hiển thị dữ liệu cảm biến từ EdgeX gửi tới.

Hình 3 minh họa rõ ràng luồng dữ liệu từ thiết bị biên lên đám mây. Hình 3 (a) hiển thị đoạn nhật ký từ LoRa Device Service, trong đó các gói tin nhận được từ các node cảm biến được giải mã thành công, loại bỏ phần Header Byte và Length, chỉ để lại phần Device ID, Command và Data. Điều này xác nhận khả năng của Device Service trong việc nhận và phân tích dữ liệu thô từ các node LoRa. Tiếp đó, Hình 3 (b) cho thấy giao diện của MQTTX đang lắng nghe các MQTT Topic liên quan, hiển thị các tin nhắn MQTT với nội dung chứa dữ liệu cảm biến (vừa nhận được như trong Hình 3 (a)) đã được đẩy lên đám mây. Luồng dữ liệu liền mạch này chứng minh khả năng thu thập và chuyển tiếp dữ liệu của hệ thống từ tầng thiết bị đến tầng đám mây.

4.2.2. Kịch bản 2: Lưu trữ và truy vấn dữ liệu

Kịch bản này nhằm đánh giá khả năng lưu trữ cục bộ và truy vấn thông qua API. Dữ liệu cảm biến được EdgeX Foundry lưu trữ liên tục trong Core Data. Chúng tôi đã sử dụng Postman để truy vấn 100 điểm dữ liệu cường độ ánh sáng gần nhất thông qua API của EdgeX Core Data như ở trong Hình 4.



Hình 4. Truy vấn và trực quan hóa lịch sử dữ liệu sử dụng công cụ Postman

Việc truy vấn thành công này xác nhận tính năng lưu trữ dữ liệu cục bộ của EdgeX Foundry. Phần dưới trong hình là đồ thị trực quan hóa của 100 điểm dữ liệu cường độ ánh sáng, cho thấy sự biến thiên rõ rệt theo thời gian. Đồ thị này được tạo ra từ dữ liệu thực tế thu thập được, chứng minh khả năng hiển thị và phân tích dữ liệu đã lưu trữ một cách hiệu quả.

4.2.3. Kịch bản 3: Điều khiển và xử lý luật nghiệp vụ tại biên

Kịch bản này nhằm đánh giá khả năng gửi lệnh điều khiển và tự động thực thi các tác vụ phức tạp (xử lý luật) ngay tại biên. Một luật được cấu hình trong Rule Engine của EdgeX: nếu giá trị cường độ ánh sáng từ Node CB-AnhSang thấp hơn 100 lux, hệ thống sẽ đồng thời thực hiện hai hành động: (1) gửi lệnh bật Rơ-le 2 trên Node Relay2Kenh và (2) gửi một thông báo "rule:bat_relay2_theo_anhsang da kich hoat" lên đám mây qua MQTT để báo luật đã được kích hoạt.

Hình 5. a) Nhật ký của Device Service khi nhận gói tin từ cảm biến và gói tin gửi tới thiết bị chấp hành. b) Giao diện MQTTX khi nhận được thông báo luật đã được kích hoạt.

Hình 5 cung cấp minh chứng trực quan về khả năng xử lý luật nghiệp vụ tại biên. Hình 5 (a) là đoạn nhật ký từ LoRa Device Service, ghi lại hai sự kiện với dấu thời gian theo định dạng HH:MM:SS.mmm. Vào lúc 12:35:26.483, Device Service nhận một gói tin từ CB-AnhSang báo cáo cường độ ánh sáng là 15 lux, và ngay sau đó, lúc 12:35:26.501

Device Serivce gửi gói tin bật Rơ-le 2 cho Relay2Kenh. Khoảng thời gian rất ngắn giữa hai dấu thời gian này (trong hình là khoảng 18ms) phản ánh độ trễ thấp trong quá trình xử lý. Hình 5 (b) cho thấy giao diện của MQTTX nhận được một tin nhắn MQTT thông báo rằng luật về ánh sáng đã được kích hoạt. Chúng tôi đã tiến hành thử nghiệm lặp lại 100 lần liên tiếp, kết quả cho thấy rằng thời gian xử lý từ lúc nhận dữ liệu đến khi gửi kết quả đi (bao gồm cả việc xử lý luật) luôn dưới dưới 30ms. Điều này chứng minh rằng xử lý tại biên có độ trễ rất thấp, rất hiệu quả.

4.2.4. Đánh giá việc tiêu thụ tài nguyên

Kết quả ở Bảng 1 cho thấy việc triển khai EdgeX Foundry cùng LoRa Device Service trên Raspberry Pi 4 đạt hiệu quả cao về tiêu thụ tài nguyên. Từ đó cho thấy có thể triển khai hệ thống một cách gọn nhẹ và hiệu quả trên các thiết bị biên có tài nguyên phần cứng hạn chế, vốn là một yêu cầu quan trọng trong nhiều kịch bản IoT thực tế.

Bảng 1. Tài nguyên tiêu thụ của hệ thống EdgeX-LoRa trên Raspberry Pi 4

Thông số	Giá trị
Kích thước chương trình (toàn bộ Image Docker)	~ 540 MB
Mức sử dụng CPU (trung bình)	< 5%
Mức sử dụng RAM (trung bình)	< 1%

IV. Kết luận

Nghiên cứu này đã thiết kế, phát triển và triển khai thành công một hệ thống điều khiển - giám sát IoT dựa trên nền tảng điện toán biên EdgeX Foundry và công nghệ LoRa. Chúng tôi đã xác nhận được khả năng thu thập dữ liệu từ các cảm biến thực tế (DHT21, BH1750), lưu trữ cục bộ, chuyển tiếp lên đám mây, và đặc biệt là xử lý các luật nghiệp vụ phức tạp trực tiếp tại biên với độ trễ thấp (dưới 30ms). Hiệu suất tiêu thụ tài nguyên thấp trên Raspberry Pi 4 cũng là một minh chứng quan trọng cho tính khả thi của giải pháp. Tuy nhiên, vẫn còn hạn chế như các khía cạnh về bảo mật chưa được nghiên cứu xét đến.

Hướng nghiên cứu tiếp theo có thể tập trung vào việc mở rộng hệ thống để hỗ trợ nhiều loại cảm biến và bộ chấp hành hơn, tích hợp các thuật toán học máy tại biên để xử lý nâng cao, và đánh giá hiệu suất khi triển khai quy mô lớn hơn với nhiều gateway và node LoRa.

Tài liệu tham khảo:

[1]. Andriulo, F. C., Fiore, M., Mongiello, M., Traversa, E., & Zizzo, V. (2024). Edge Computing and Cloud Computing for Internet of Things: A Review. *Informatics*, *11*(4), Article 4. https://doi.org/10.3390/informatics11040071

[2]. *Docker Compose*. (100 C.E., 58:30 + +0100). Docker Documentation. https://docs.docker.com/compose/

[3]. *EdgeX Foundry Project*. (n.d.). GitHub. Retrieved June 30, 2025, from https://github.com/edgexfoundry

[4]. *Edgexfoundry/device-sdk-go*. (2025). [Go]. EdgeX Foundry Project. https://github.com/edgexfoundry/device-sdk-go (Original work published 2018)

[5]. *EMQX Overview | EMQX Docs*. (n.d.). Retrieved June 30, 2025, from https://docs.emqx.com/en/emqx/latest/

[6]. Foundation, T. L. (n.d.). *EdgeX Foundry | #1 Open Source Edge Platform*. Https://Www.Edgexfoundry.

Org. Retrieved June 30, 2025, from https://www.edgexfoundry.org

[7]. Gaffurini, M., Flammini, A., Ferrari, P., Fernandes Carvalho, D., Godoy, E. P., & Sisinni, E. (2024). End-to-End Emulation of LoRaWAN Architecture and Infrastructure in Complex Smart City Scenarios Exploiting Containers. *Sensors*, *24*(7), Article 7. https://doi.org/10.3390/s24072024

[8]. Hitimana, E., Bajpai, G., Musabe, R., Sibomana, L., & Jayavel, K. (2022). Containerized Architecture Performance Analysis for IoT Framework Based on Enhanced Fire Prevention Case Study: Rwanda. *Sensors*, *22*(17), Article 17. https://doi.org/10.3390/s22176462

[9]. Kong, L., Tan, J., Huang, J., Chen, G., Wang, S., Jin, X., Zeng, P., Khan, M., & Das, S. K. (2022). Edge-computing-driven Internet of Things: A Survey. *ACM Comput. Surv.*, *55*(8), 174:1-174:41. https://doi.org/10.1145/3555308

[10]. *LoRa PHY | Semtech—What Is Lora | Semtech*. (n.d.). Retrieved June 30, 2025, from https://www.semtech.com/lora/what-is-lora

[11]. Milani, S., Chatzigiannakis, I., Garlisi, D., Fraia, M. D., & Pisani, P. (2023). Enabling Edge processing on LoRaWAN architecture. *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, 1–3. https://doi.org/10.1145/3570361.3614074

[12]. (PDF) A Study of LoRa: Long Range & Low Power Networks for the Internet of Things. (2025). *ResearchGate*. https://doi.org/10.3390/s16091466

[13]. State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally. (2024, September 3). *IoT Analytics*. https://iot-analytics.com/number-connected-iot-devices/

[14]. *What is Docker?* (200 C.E., 26:47 + +0200). Docker Documentation. https://docs.docker.com/get-started/docker-overview/

[15]. Zhong, C., & Nie, X. (2024). A novel single-channel edge computing LoRa gateway for real-time confirmed messaging. *Scientific Reports*, *14*(1), 8369. https://doi.org/10.1038/s41598-024-59058-8

RESEARCH AND APPLICATION OF AN OPEN-SOURCE EDGE COMPUTING PLATFORM EDGEX FOUNDRY FOR LORA-BASED IOT CONTROL AND MONITORING SYSTEMS

Phan Van Hai[2], Trinh Thi Hau[2]

Abstract: *This paper presents the research and application of an Internet of Things (IoT) control and monitoring system based on LoRa technology and the open-source edge computing platform EdgeX Foundry. The study aims to build a system capable of collecting, storing, processing, and automatically executing business rules directly at the edge, while offering flexible integration with cloud services. We developed a custom LoRa Device Service integrated into EdgeX to connect with specific LoRa devices. Experimental results demonstrate the system's stable operation and efficient data collection and processing, with an average latency of under 30ms for edge-side tasks. Furthermore, its deployment on a Raspberry Pi, exhibiting very low CPU and RAM consumption, confirms the feasibility and efficiency of this edge computing solution in resource-constrained IoT environments.*

Keywords: *EdgeX Foundry, LoRa, IoT, edge computing, control - monitoring, device Service*

[2] Hanoi Open University