

**Đại học Quốc gia Thành phố Hồ Chí Minh**  
**Trường Đại học Bách Khoa**  
**Khoa Khoa học và Kỹ thuật Máy tính**



## **Đề Cương Luận Văn**

**Đề tài:** Hiện thực mô hình điều khiển tự động quadcopter theo cơ chế bầy đàn

Hội đồng : Kỹ Thuật Máy Tính  
Giáo viên hướng dẫn : TS. Phạm Hoàng Anh  
Giáo viên phản biện : Tên GV Phản Biện

Nhóm sinh viên thực hiện : **Trần Lê Minh** - 1412319  
**Lê Huy Cường** - 1410432  
**Nguyễn Hoàng Thương** - 1513396

Ngày 30 tháng 10 năm 2018

I think, therefore I am;  
— René Descartes

# Lời cam đoan

Nhóm cam đoan mọi điều được ghi trong báo cáo, cũng như mã nguồn là do nhóm tự thực hiện - trừ các kiến thức tham khảo có trích dẫn cũng như mã nguồn mẫu do chính nhà sản xuất cung cấp, hoàn toàn không sao chép từ bất cứ nguồn nào khác. Nếu lời cam đoan trái với sự thật, nhóm xin chịu mọi trách nhiệm trước Khoa và Nhà trường.

Nhóm sinh viên thực hiện đề tài

# Lời cảm ơn

Sau bốn năm theo học tại khoa Khoa học Kỹ thuật Máy tính trường Đại học Bách khoa TP. Hồ Chí Minh, em đã được áp dụng tất cả những kiến thức đã học để thực hiện luận văn tốt nghiệp, đề tài cuối cùng mang tính quyết định kết quả học tập và quá trình tốt nghiệp.

Em xin chân thành cảm ơn quý thầy cô trong Khoa đã mang đến những kiến thức bổ ích trong suốt quãng đời sinh viên, đặc biệt là thầy Phạm Hoàng Anh đã giúp đỡ em trong quá trình thực hiện cũng như quá trình hoàn thiện luận văn. Em cũng gửi lời cảm ơn đến một số bạn bè trong Khoa đã hỗ trợ giúp em tiết kiệm được thời gian tìm hiểu đề tài.

Em đã cố gắng hết sức để tránh các sai sót, nhưng nếu có phát hiện thì mong quý thầy cô góp ý để em càng hoàn thiện đề tài hơn. Cuối cùng em xin gửi lời chúc sức khỏe và cảm ơn chân thành nhất.

Nhóm sinh viên thực hiện đề tài



# Tóm tắt

Work in progress..

# Mục lục

## Lời cam đoan

Lời cảm ơn	i
------------	---

Tóm tắt	ii
---------	----

Mục Lục Hình	vi
--------------	----

Thuật ngữ & từ viết tắt	viii
-------------------------	------

<b>1 Giới thiệu về tài</b>	<b>1</b>
1.1 Nhiệm vụ cần đạt . . . . .	3
1.2 Sơ đồ khôi . . . . .	4
1.3 Hoạt động . . . . .	4
<b>2 Cơ sở lý thuyết</b>	<b>5</b>
2.1 Bluetooth Mesh . . . . .	5
2.1.1 Bluetooth trước Bluetooth Mesh . . . . .	5
2.1.2 Bluetooth Mesh ra đời . . . . .	6
2.1.3 Sứ mạng của Bluetooth Mesh . . . . .	7
2.2 Ưu điểm của Bluetooth Mesh . . . . .	8
2.2.1 Tính bảo mật . . . . .	8
2.2.2 Tính liên kết . . . . .	10
2.2.3 Dễ mở rộng . . . . .	10
2.2.4 Tiết kiệm năng lượng . . . . .	10
2.2.5 Một số ưu điểm khác của Bluetooth Mesh . . . . .	11
2.3 Kiến trúc Bluetooth Mesh . . . . .	12
2.3.1 Lớp Model . . . . .	13
2.3.2 Lớp Foundation Model . . . . .	13

2.3.3	Lớp Access . . . . .	13
2.3.4	Lớp Upper Transport . . . . .	13
2.3.5	Lớp Lower Transport . . . . .	13
2.3.6	Lớp Network . . . . .	14
2.3.7	Lớp Bearer . . . . .	14
2.4	Nguyên lý hoạt động của Bluetooth Mesh . . . . .	15
2.4.1	Phần tử trong mạng . . . . .	15
2.4.2	Truyền tin trong mạng . . . . .	17
2.5	Quản lý các phần tử trong mạng . . . . .	18
2.5.1	Tham gia mạng - Provisioning . . . . .	18
2.5.2	Thiết bị mới - Provisionee . . . . .	21
2.5.3	Thiết bị Provisioner . . . . .	22
2.5.4	Rời khỏi mạng . . . . .	23
2.6	Khái niệm Friendship . . . . .	24
2.6.1	Các bước thiết lập Friendship . . . . .	25
2.6.2	Giao tiếp trong Friendship . . . . .	26
<b>3</b>	<b>Thiết kế</b>	<b>28</b>
3.1	Mô hình thiết kế . . . . .	28
3.2	Phần cứng . . . . .	28
3.3	Phần mềm . . . . .	31
3.3.1	S132 Softdevice . . . . .	31
3.3.2	nRFgo Studio . . . . .	31
3.3.3	Nordic Mesh SDK . . . . .	31
3.3.4	Nordic SDK . . . . .	32
3.3.5	Thư viện giao tiếp module SIM . . . . .	32
3.3.6	ThingSpeak . . . . .	33
3.3.7	Embedded Studio . . . . .	33
<b>4</b>	<b>Hiện thực ứng dụng</b>	<b>34</b>
4.1	Chỉnh sửa model Simple OnOff . . . . .	34
4.1.1	Model Simple OnOff . . . . .	34
4.1.2	Khó khăn . . . . .	34
4.1.3	Thực hiện . . . . .	35
4.2	Kết hợp thư viện SDK common . . . . .	35
4.2.1	Khó khăn . . . . .	35

4.2.2	Thực hiện . . . . .	35
4.3	Hiện thực node cảm biến . . . . .	36
4.3.1	Đọc giá trị cảm biến . . . . .	36
4.4	Hiện thực node gateway . . . . .	36
4.4.1	Thư viện giao tiếp với module SIM . . . . .	36
<b>5</b>	<b>Kịch bản thử nghiệm</b>	<b>37</b>
5.1	Mục tiêu của thử nghiệm . . . . .	37
5.2	Quá trình thử nghiệm . . . . .	37
5.2.1	Kịch bản I: Thử nghiệm giao tiếp . . . . .	38
5.2.2	Kịch bản II: Thử nghiệm khoảng cách . . . . .	39
5.2.3	Kịch bản III: Thử nghiệm truyền dữ liệu trong mạng mesh .	40
<b>6</b>	<b>Kết luận</b>	<b>42</b>
6.1	Đánh giá kết quả . . . . .	42
6.1.1	Thành quả đạt được . . . . .	42
6.1.2	Một số hạn chế của ứng dụng thử nghiệm . . . . .	42
6.2	Hướng phát triển . . . . .	43
6.2.1	Tiềm năng trong thực tế . . . . .	43
<b>A</b>	<b>Một số Model chuẩn</b>	<b>45</b>
A.1	Foundation models . . . . .	45
A.2	Generic models . . . . .	45
A.3	Sensors . . . . .	46
A.4	Time and scenes . . . . .	46
A.5	Lighting . . . . .	47
<b>B</b>	<b>Flowchart quá trình remote provisioning</b>	<b>48</b>
<b>C</b>	<b>Hướng dẫn thực thi ứng dụng</b>	<b>49</b>
	<b>Tài liệu tham khảo</b>	<b>52</b>

# Danh sách hình vẽ

1.1	Một hệ thống IoT tiêu chuẩn . . . . .	2
1.2	Sơ đồ khái của ứng dụng . . . . .	4
2.1	Các ứng dụng sử dụng Bluetooth beacons . . . . .	6
2.2	Mối quan hệ giữa Bluetooth mesh và BLE . . . . .	6
2.3	Hệ thống đèn của một tòa nhà ứng dụng Bluetooth Mesh . . . . .	7
2.4	Mạng mesh với node "bạn bè" và low power . . . . .	12
2.5	Kiến trúc Bluetooth mesh . . . . .	12
2.6	Cấu trúc của một node . . . . .	16
2.7	Cấu trúc của một node bóng đèn . . . . .	16
2.8	Sơ đồ của cơ chế publish-subscribe . . . . .	18
2.9	Quá trình provisioning gồm 5 bước . . . . .	19
2.10	Provisioning scenarios . . . . .	21
2.11	Provisionee flowchart . . . . .	22
2.12	Provisioner flowchart . . . . .	23
2.13	Hình mô tả ReceiveDelay và ReceiveWindow . . . . .	25
2.14	Hình mô tả PollTimeout . . . . .	25
2.15	Hình mô tả quá trình giao tiếp . . . . .	26
3.1	Mô hình thiết kế . . . . .	29
3.2	Hình ảnh KIT phát triển PCA10040 . . . . .	30
3.3	Hình ảnh module 3G SARA click . . . . .	30
3.4	Các lớp trong một chương trình . . . . .	31
5.1	Hình ảnh thử nghiệm I trong thực tế . . . . .	38
5.2	Hình ảnh note gateway nhận được dữ liệu từ node cảm biến . . . . .	38
5.3	Hình ảnh dữ liệu được cập nhật lên cloud . . . . .	39
5.4	Hình ảnh thử nghiệm II trong thực tế . . . . .	40

## Danh sách hình vẽ

---

5.5	Hình ảnh thử nghiệm III . . . . .	40
5.6	Hình ảnh thử nghiệm III trong thực tế . . . . .	41
5.7	Hình ảnh điều khiển LED trong thực tế . . . . .	41
B.1	Quá trình remote provisioning . . . . .	48
C.1	Giao diện phần mềm nRFgo Studio . . . . .	49
C.2	Giao diện phần mềm Embedded Studio Project . . . . .	50
C.3	Giao diện phần mềm J-link RTT Viewer . . . . .	50

# Thuật ngữ & từ viết tắt

- AES** ..... Advanced Encryption Standard
- BLE** ..... Bluetooth Light Energy
- BR/EDR** .... Basic Rate/Enhanced Data Rate
- CBC-MAC** ... Cipher Block Chaining Message Authentication Code
- CCM** ..... Counter with CBC-MAC
- Cloud** ..... Trong phạm vi đề tài, dùng để chỉ nơi lưu trữ dữ liệu trên internet
- CMAC** ..... Cipher-based Message Authentication Code
- ECDH** ..... Elliptic-curve Diffie–Hellman
- GCS** ..... Ground Control Station
- GSM** ..... Global System for Mobile Communications
- GUI** ..... Graphical User Interface
- IPv4** ..... Internet Protocol version 4
- IPv6** ..... Internet Protocol version 6
- LPN** ..... Low power node
- Mesh** ..... Mạng mesh là mạng mà các phần tử đều có kết nối với nhau
- MITM** ..... Man-in-the-middle: Chen giữa một kết nối nhằm mục đích xấu
- Node** ..... Node là một phần tử trong mạng mesh

**OOB** ..... Out-of-band

**P2P** ..... Peer-to-peer

**TTL** ..... Time To Live

**Sleep** ..... Trạng thái ngủ giúp tiết kiệm năng lượng

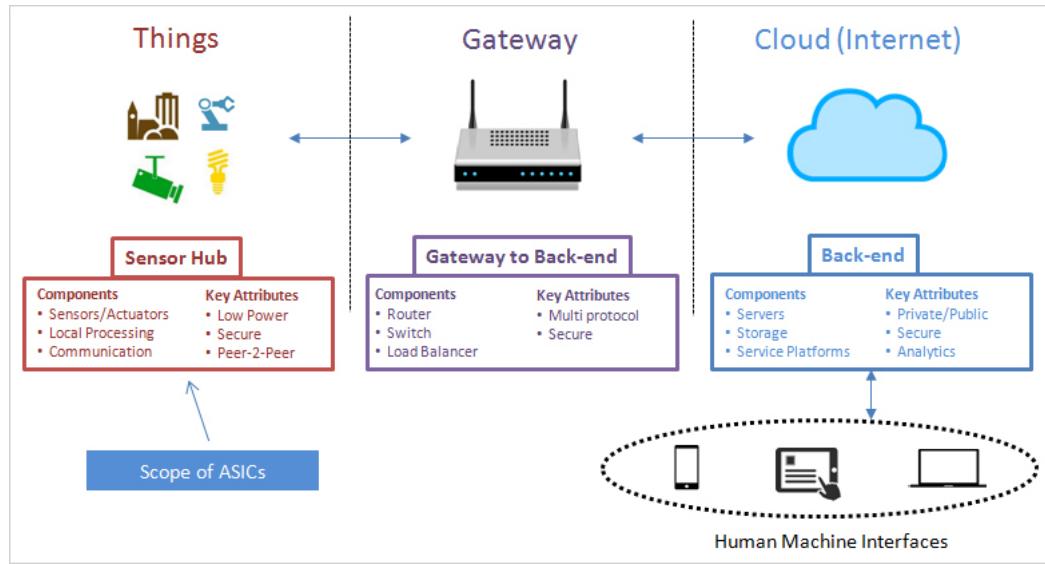
# 1 Giới thiệu đề tài

Từ lúc bắt đầu có khái niệm IoT cho đến nay, đã có rất nhiều ứng dụng trong những lĩnh vực khác nhau, mang lại hiệu quả và độ chính xác cao hơn hẳn so với sức người, giảm chi phí cho sản xuất, bảo vệ môi trường,... Do đó, IoT được dự báo là công nghệ của tương lai, đóng vai trò quan trọng trong “Cách mạng công nghiệp 4.0”.

Hình dưới đây mô tả một hệ thống IoT tiêu chuẩn: các thiết bị và cảm biến trong nhóm Things sẽ kết nối, giao tiếp với nhau thông qua một mạng nội bộ, một số ứng dụng chỉ cần nhóm này là đủ nếu không có nhu cầu trao đổi dữ liệu với cloud. Gateway sẽ đảm nhiệm vai trò kết nối giữa hai loại mạng khác nhau, chẳng hạn như mạng nội bộ với cloud, mạng WiFi với ZigBee hay như trong đề tài này là mạng Bluetooth Mesh với GPRS. Sau cùng là cloud với vai trò lưu trữ dữ liệu cũng như phân tích nó, cloud còn đảm nhiệm giao tiếp với một số thiết bị có giao diện người dùng như máy tính, điện thoại,...

Nhóm Things có thể bao gồm rất nhiều thiết bị, chẳng hạn như một hội trường cỡ vừa đã có hàng trăm bóng đèn, hàng trăm cái ghế. Do đó việc sử dụng kiểu kết nối hình sao, vòng, P2P hay cây đều có những bất lợi cũng như khó mở rộng, kết nối mesh dễ dàng đánh bại những kiểu kết nối khác trong cuộc đua này.

Cho đến nay đã có rất nhiều giải pháp mạng mesh không dây (wireless) sử dụng nhiều loại công nghệ khác nhau. Mục tiêu ban đầu của những giải pháp này là ứng



Hình 1.1: Một hệ thống IoT tiêu chuẩn

dụng trong quân sự, sau đó là công nghiệp và cuối cùng là nhà thông minh. Lý do chính dẫn đến thất bại của WiFi và ZigBee trong việc hiện thực mạng mesh chính là thiếu khả năng liên kết giữa các nhà sản xuất, Bluetooth Mesh sẽ thay đổi điều đó[2].

Nhằm tìm hiểu tiềm năng trong tương lai của Bluetooth Mesh, nhóm đã thực hiện đề tài này và tổng hợp những điểm quan trọng của giao thức này trong báo cáo, làm cơ sở tham khảo cho những ai quan tâm và có ý định phát triển ứng dụng trên nền Bluetooth Mesh.

Báo cáo sẽ được trình bày theo thứ tự như sau:

- Chương đầu tiên sẽ giới thiệu sơ lược về đề tài và giao thức Bluetooth Mesh, liệt kê những nhiệm vụ cần đạt cũng như sơ đồ khái và cách thức hoạt động của ứng dụng thử nghiệm.
- Chương tiếp theo sẽ tập trung vào những gì nhóm nghiên cứu được về giao thức Bluetooth Mesh, điểm qua lịch sử, kiến trúc, nguyên lý hoạt động và cách thức quản lý các phần tử trong mạng.
- Chương 3 là danh sách những công cụ cả về phần cứng và phần mềm mà

## **1.1. Nhiệm vụ cần đạt**

---

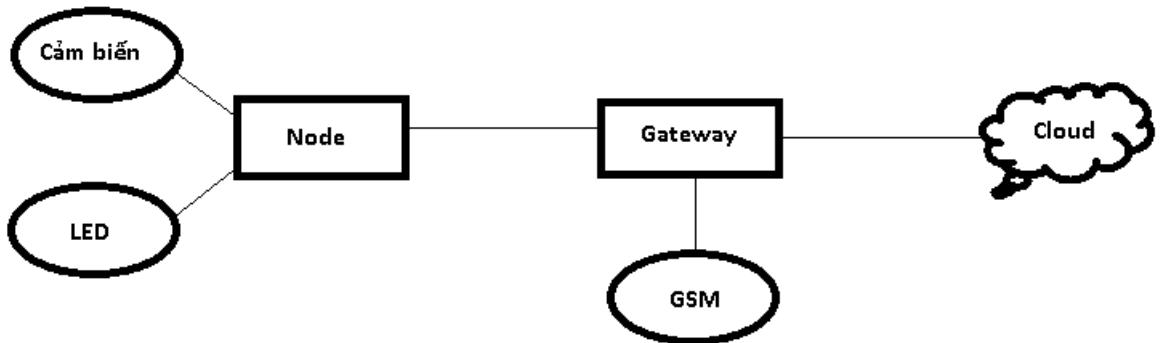
nhóm dùng để hiện thực ứng dụng thử nghiệm, có bao gồm mô hình thiết kế của ứng dụng.

- Chương 4 ghi lại quá trình hiện thực ứng dụng dựa trên công cụ của nhà sản xuất và tùy chỉnh một số mã nguồn mẫu, hiện thực thư viện giao tiếp với module GSM.
- Chương 5 là các kịch bản nhóm dựng lên để thử nghiệm ứng dụng, bao gồm thử nghiệm khoảng cách tối đa giữa 2 thiết bị trong mạng, giao tiếp trong mạng và hoạt động của node gateway.
- Chương 6 là chương kết luận, bao gồm việc đánh giá và đưa ra gợi ý cho những ứng dụng trong thực tế.
- Cuối cùng là tài liệu tham khảo và một số phụ lục: các model chuẩn, flowchart quá trình remote provisioning, hướng dẫn sơ lược cách sử dụng các chương trình nhóm dùng để hiện thực ứng dụng.

### **1.1 Nhiệm vụ cần đạt**

- Tìm hiểu lý thuyết về Bluetooth Mesh, nguyên lý hoạt động, ưu nhược điểm,... Mục tiêu này đã hoàn thành bước đầu trong giai đoạn Đề lương luận văn. Giai đoạn luận văn sẽ nghiên cứu sâu hơn và cụ thể hơn các lý thuyết đã được trình bày ở giai đoạn Đề cương.
- Sử dụng các công cụ hiện có, hiện thực ứng dụng có khả năng giao tiếp giữa các thiết bị cuối - đảm nhiệm nhiệm vụ thu nhập dữ liệu và điều khiển thiết bị - và thiết bị gateway - đảm nhiệm việc giao tiếp với cloud.
- Đề ra các phương án áp dụng Bluetooth Mesh vào các ứng dụng thực tế, cũng như các hướng phát triển trong tương lai.
- Xây dựng kịch bản để thử nghiệm, chứng minh ưu điểm của Bluetooth Mesh so với các chuẩn giao tiếp khác.

## 1.2 Sơ đồ khái



Hình 1.2: Sơ đồ khái của ứng dụng

## 1.3 Hoạt động

- Trước tiên cần tiến hành bước khởi tạo: cho phép các node cảm biến tham gia vào mạng.
- Án nút trên node gateway để kích hoạt/tắt chức năng giám sát của hệ thống. Khi trong chế độ giám sát, sau mỗi chu kỳ nhất định, node gateway sẽ gửi yêu cầu lần lượt tới toàn bộ các node cảm biến có trong mạng - quá trình này cũng cho phép kiểm tra sự tồn tại của các node trong mạng - sau đó gửi toàn bộ dữ liệu lên server. Chức năng này thể hiện hướng dữ liệu từ node cảm biến đến node gateway.
- Án một nút khác cũng trên node gateway để điều khiển LED trên tất cả các node cảm biến, chức năng này giúp cho việc kiểm thử dễ dàng hơn. Chức năng này thể hiện hướng dữ liệu từ node gateway đến node cảm biến.

## 2 Cơ sở lý thuyết

### 2.1 Bluetooth Mesh

#### 2.1.1 Bluetooth trước Bluetooth Mesh

Trước khi Bluetooth Mesh ra đời, Bluetooth đã phát triển đến phiên bản 5.0, trong đó có thể chia làm hai giai đoạn (hay hai sứ mạng) bao gồm Bluetooth Basic Rate/Enhanced Data Rate (BR/EDR) và Bluetooth Low Energy (LE). BR/EDR mang lại thế hệ thiết bị không dây mới như tai nghe, chuột, bàn phím không dây,... trong khi đó BLE mang lại các thiết bị chạy pin như đồng hồ thông minh, các thiết bị thể thao, định vị,... Cả hai có thể tồn tại trong cùng một thiết bị nhưng không giao tiếp với nhau.

Trong khi BR/EDR chỉ hỗ trợ kết nối 1:1, dễ dàng nhận thấy điều này qua việc một bàn phím không dây không thể dùng cho hai máy tính cùng lúc, hoặc một điện thoại không thể "bắn bluetooth" cho nhiều điện thoại cùng lúc; thì BLE hỗ trợ thêm kết nối 1:nhiều bằng cách thức broadcast, chúng ta có thể tìm thấy kết nối này trong các ứng dụng Bluetooth beacons.

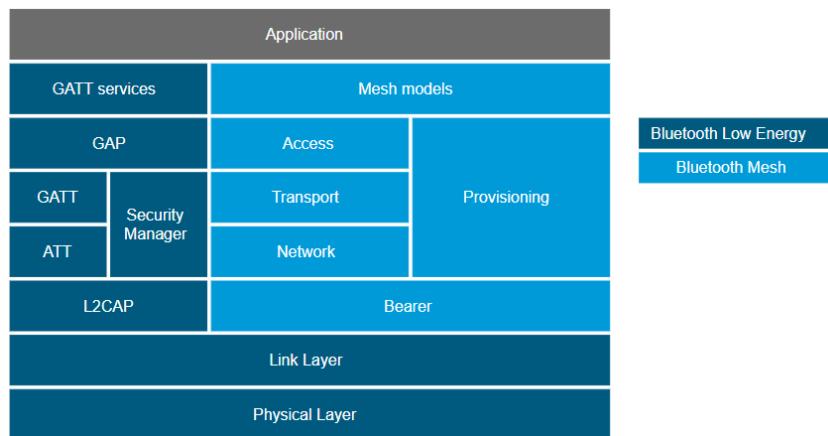


Hình 2.1: Các ứng dụng sử dụng Bluetooth beacons

*Vậy Bluetooth Mesh có thể mang lại những tiềm năng gì?*

### 2.1.2 Bluetooth Mesh ra đời

Bluetooth Mesh được tổ chức SIG đưa ra các giao thức (protocol) chung vào tháng 7 năm 2017[3] và chia sẻ nhiều đặc điểm chung với BLE. Về cơ bản, Bluetooth Mesh sử dụng cấu trúc mạng của BLE nên các thiết bị sử dụng mạng BLE có thể tham gia mạng mesh, cấu hình của các gói tin (package) và phương thức quảng bá (advertisement) của cả 2 giống nhau. Tuy nhiên vẫn có những điểm khác nhau giữa lớp Host của cả 2 mạng. Do đó, Bluetooth là một giao thức kết nối, còn Bluetooth Mesh là một giao thức mạng.

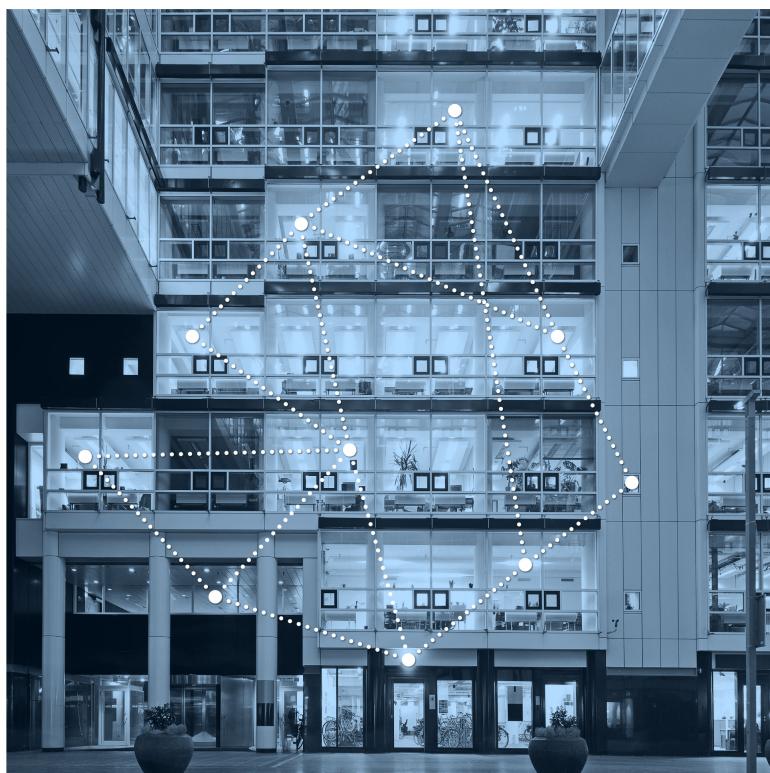


Hình 2.2: Mối quan hệ giữa Bluetooth mesh và BLE

### 2.1.3 Sứ mạng của Bluetooth Mesh

Bluetooth Mesh hướng đến những ứng dụng điều khiển hoặc giám sát cần các kết nối nhiều:nhiều (một mạng mesh có khả năng phân tán tốt), số thiết bị có thể lên đến hàng ngàn, theo Nordic thì Bluetooth Mesh hỗ trợ tối đa 32767 thiết bị trong một mạng (con số này dựa trên trường địa chỉ mà giao thức BLE Mesh hỗ trợ) với đường kính mạng tối đa là 127 hops[4]. Chính Nordic đã thử nghiệm trên hệ thống hơn 1000 thiết bị và đạt kết quả khả quan.

Định dạng gói tin được tối ưu hóa cho các gói tin nhỏ (kiểu short-burst) chẳng hạn như tắt mở thiết bị, giá trị của số vài cảm biến, giá trị độ sáng bóng đèn,... không dành cho việc truyền dữ liệu hoặc các ứng dụng băng thông cao khác như streaming nhạc và video.



Hình 2.3: Hệ thống đèn của một tòa nhà ứng dụng Bluetooth Mesh

### 2.2 Ưu điểm của Bluetooth Mesh

Để tiến hành triển khai các ứng dụng IoT trong thực tế, vấn đề bảo mật là cực kỳ quan trọng, đặc biệt trong các ngành y tế, giao thông và an ninh, hiện nay các hệ thống IoT sử dụng các giao thức thông thường như WiFi, Ethernet, GSM bị xem là bảo mật kém. Kể cả trong các ứng dụng điều khiển trong nhà thì việc một người lạ có quyền tắt mở đèn, quạt trong nhà của chúng ta thì cũng đủ mang lại nhiều phiền phức rồi.

Vấn đề liên kết giữa các nhà sản xuất cũng quan trọng không kém, chẳng hạn chúng ta mua một bộ gồm đèn và remote của hãng A, sau một thời gian thì thấy đèn của hãng ko bền nên muốn dùng đèn của hãng khác, nhưng remote của hãng A lại không điều khiển được đèn của hãng khác mà mua một remote khác thì tốn kém hơn mua đèn nhiều.

Ngoài ra còn hai vấn đề không kém phần quan trọng là tính mở rộng và khả năng tiết kiệm năng lượng, tuy nhiên hai vấn đề này lại phụ thuộc vào các ứng dụng thực tế, một số ứng dụng thì cần nhưng một số lại không. Chẳng hạn như những hệ thống nhỏ, có số lượng thiết bị không đáng kể thì không cần thiết có khả năng mở rộng; những ứng dụng như hệ thống đèn có khả năng kết nối trực tiếp với mạng điện lưới thì tiết kiệm năng lượng trở nên không mấy quan trọng.

**Bluetooth Mesh là giải pháp hội đủ cả bốn tiêu chí trên, rất thích hợp để làm nền tảng triển khai các ứng dụng từ công nghiệp cho đến dân dụng.**

#### 2.2.1 Tính bảo mật

- Bluetooth Mesh dùng 3 key để mã hóa: Devive Key, Application Key và Network Key dùng để mã hóa gói tin ở lớp network và application. NetKey giúp chia một mạng mesh thành nhiều subnet, chẳng hạn như subnet trong nhà và subnet ngoài vườn, do tính năng relay gói tin được hiện thức ở lớp Network nên nếu không cùng NetKey thì không thể relay, nghĩa là khác

## 2.2. Ưu điểm của Bluetooth Mesh

---

subnet sẽ không truyền tin qua lại được. Như vậy một mạng mesh có thể dùng nhiều NetKey trong đó có một NetKey cho toàn mạng và nhiều NetKey cho các subnet. AppKey mã hóa ở 2 lớp trên cùng là Model và Foundation Model, do đó chỉ có người nhận mới đọc được gói tin ở lớp này.

- NetKey còn được dùng để tạo ra Privacy Key, key này dùng để xáo trộn header của gói tin khiến cho kẻ tấn công không thể xác định địa chỉ nguồn, địa chỉ nhận.
- Mỗi gói tin đều có trường authentication (một cách chứng minh gói tin này xuất phát từ một node trong mạng) dài 64-bits, có thể tăng thêm.
- Chống tấn công kiểu relay bằng cách bắt buộc chỉ số sequence phải thay đổi mỗi lần gửi tin, ngoài ra còn hỗ trợ thêm một trường nonce như một cách redundant, trường này sẽ được tạo mới mỗi khi gửi nhận tin dựa trên 2 thông số là Initialisation Vector (IV) index và chỉ số sequence. Mỗi khi bộ đếm sequence gần cận kiệt (tràn số hoặc không thể tạo ra nonce mới, nonce mới ở đây yêu cầu phải khác tất cả nonce cũ), lúc này sẽ tiến hành quy trình cập nhật IV cũng tương tự quy trình tạo mới key chống trashcan dưới đây.
- Chống tấn công kiểu brute-force bằng độ dài key vào mạng 128-bits, trường authentication 64-bits.
- Chống tấn công kiểu man-in-the-middle (MITM) bằng cách áp dụng quy trình trao đổi key ECDH[5] cùng với out-of-band authentication (xác thực thông qua một bên thứ ba, có thể là một kênh giao tiếp khác).
- Chống tấn công kiểu trashcan: kẻ tấn công lấy thông tin như key vào mạng từ một node bị hỏng hoặc lỗi nên bị loại khỏi mạng. Cách thức chống kiểu tấn công này sẽ được thảo luận trong phần 2.5.4.
- Chống tấn công kiểu vật lý: vì Bluetooth Mesh hỗ trợ mạng lưới thiết bị rộng lớn nên không tránh khỏi việc một số node không an toàn về vật lý: đèn ở ngoài vườn có độ an toàn vật lý thấp hơn hẳn đèn trong phòng ngủ, do đó kẻ tấn công có thể khai thác lỗ hỏng này bằng cách tấn công vào node kém an toàn và từ đó điều khiển toàn bộ mạng. Bluetooth Mesh có khả năng phân loại key: key dành cho nhóm kém an toàn khác với key của nhóm an toàn nên các node kém an toàn không điều khiển được các node an toàn.

- Chống tấn công kiểu visitor bằng cách cấp cho các node tạm (trong một số ứng dụng sẽ phát sinh các node cần truy cập tạm thời vào mạng để gửi nhận thông tin) key có thời gian sống giới hạn, sau một thời gian node tạm không còn khả năng truy cập mạng nữa.
- Ngoài ra các gói tin còn được xác thực bằng các thuật toán như AES-CMAC, AES-CCM,...

### 2.2.2 Tính liên kết

Bluetooth Mesh có khả năng liên kết các nhà cung cấp với nhau. WiFi và Zigbee đang được sử dụng phổ biến trong các giải pháp mà Bluetooth Mesh dự định thay thế: smart building, smart home,... nhưng lại thiếu sự tương tác giữa các nhà cung cấp. Các thiết bị của hai hãng khác nhau không thể giao tiếp với nhau, do cấu trúc lệnh của mỗi hãng là do họ tự định nghĩa. Đối với Bluetooth Mesh thì khác, SIG đã định nghĩa sẵn các model như cách BLE đã làm với các profile chuẩn, các nhà sản xuất chỉ việc làm theo đúng chuẩn đó thì điều khiển hãng này dùng cho bóng đèn hãng khác không còn là vấn đề.

### 2.2.3 Dễ mở rộng

Do đặc thù là mạng mesh nên ưu điểm của Blueetooth Mesh sẽ là tính phân tán, các node không cần phải đặt tập trung vào một gateway hay access point mà có thể đặt bất cứ nơi đâu tùy theo nhu cầu ứng dụng. Khi áp dụng giao thức BLE Mesh vào trong IoT, phạm vi hoạt động của mạng lưới cảm biến sẽ được mở rộng rất nhiều, chẳng hạn như node A nằm ngoài phạm vi kết nối của node B, tuy nhiên chỉ cần đặt thêm node C vào giữa hai node này, thế là A có thể giao tiếp với B thông qua C, thử tưởng tượng với 127 hops mà khoảng cách giữa 2 node khoảng 8 - 10m (kết quả thử nghiệm thực tế), phạm vi tối đa sẽ đạt tới mức nào?

### 2.2.4 Tiết kiệm năng lượng

Bluetooth Mesh là giao thức mạng, kết thừa các lớp của BLE nên cũng kế thừa khả năng tiết kiệm năng lượng. Ngoài ra, Bluetooth Mesh còn đưa ra khái niệm Friendship, giúp cho các node cần tiết kiệm năng lượng có thể ở trạng thái sleep

## 2.2. Ưu điểm của Bluetooth Mesh

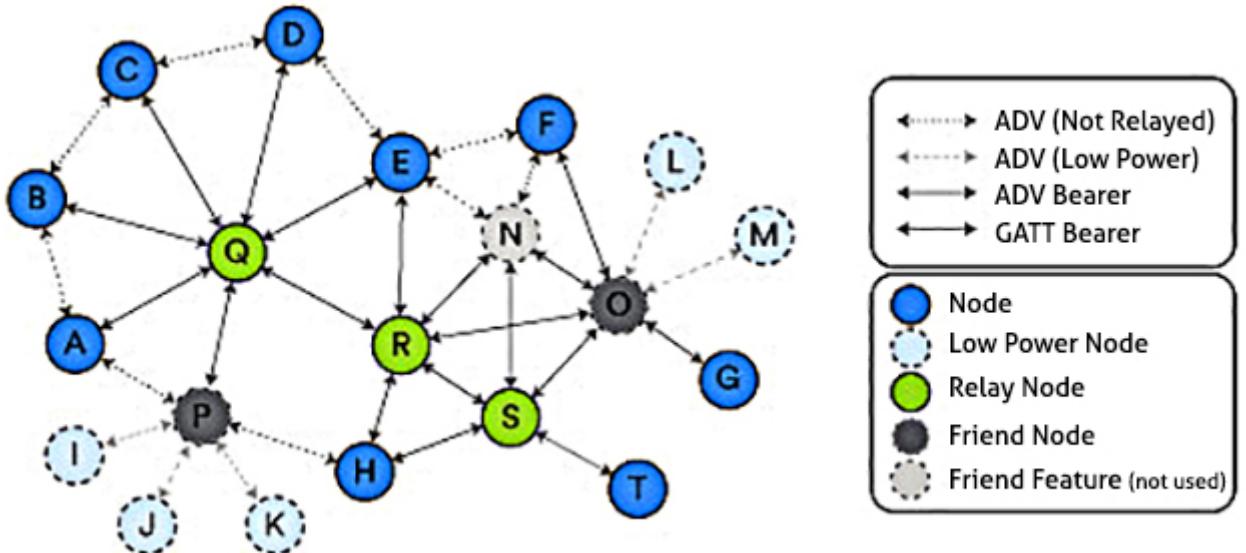
---

lâu hơn mà không bị loại ra khỏi mạng hoặc bỏ lỡ những gói tin được gửi tới nó.

### 2.2.5 Một số ưu điểm khác của Bluetooth Mesh

- Bluetooth Mesh đạt các yêu cầu cho một ứng dụng công nghiệp với đầy đủ các yêu cầu: đáng tin cậy (reliability), dễ mở rộng (scalability) và tính bảo mật cao.
- Sản phẩm dễ được thị trường đón nhận hơn vì Bluetooth đã trở thành thường thức với phần đông dân số thế giới, do đó khi nhà sản xuất miêu tả sản phẩm này áp dụng giao tiếp Bluetooth thì khách hàng dễ hình dung hơn là Zigbee hay Lora nhiều.
- Chức năng Bluetooth vốn tích hợp sẵn trên mọi chiếc điện thoại thông minh, thậm chí điện thoại không thông minh, cũng như rất nhiều thiết bị công nghệ như máy tính bảng, laptop,... Mức độ phổ biến của nó tương ứng với tiềm năng phát triển trong tương lai, khi mà Bluetooth Mesh chấp nhận giao tiếp với thiết bị non-mesh (các thiết bị dùng Bluetooth hiện nay đều là non-mesh), mà theo như thông tin nhóm tìm hiểu thì các thiết bị BLE từ 4.0 trở lên và có đủ điều kiện (chẳng hạn như dung lượng bộ nhớ còn trống trong chip) thì có thể giao tiếp với một mạng mesh thông qua ứng dụng.
- Khi mạng lưới càng lớn thì khả năng tự hồi phục của mạng càng mạnh, việc một vài node bị hỏng sẽ không ảnh hưởng tới mạng bất kể node đó có phải relay hay không, bởi vì còn rất nhiều con đường khác - những node relay khác - để gói tin có thể đi qua.
- Khả năng thiết lập kết nối nhanh, khi có phần tử mới muốn tham gia vào mạng, chỉ cần được thiết lập với key hợp lệ.

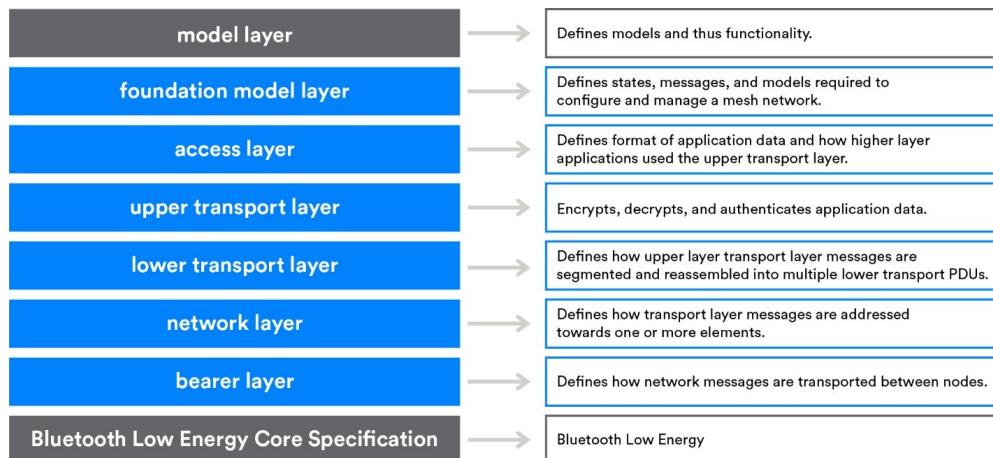
### 2.3. Kiến trúc Bluetooth Mesh



Hình 2.4: Mạng mesh với node "bạn bè" và low power

## 2.3 Kiến trúc Bluetooth Mesh

Trong phần này, nhóm sẽ trình bày những đặc điểm chính của từng thành phần trong kiến trúc Bluetooth Mesh. Mô hình kiến trúc được thể hiện trong hình bên dưới:



Hình 2.5: Kiến trúc Bluetooth mesh

### 2.3.1 Lớp Model

Lớp model chịu trách nhiệm hiện thực các model, nghĩa là hiện thực các hành vi, trạng thái, gói tin điều khiển,... Tham khảo thêm về các model chuẩn trong phần phụ lục A.

### 2.3.2 Lớp Foundation Model

Lớp Foundation Model như một lớp đệm cho lớp Model, thiết lập và quản lý lớp Model cho thích hợp với mạng mesh.

### 2.3.3 Lớp Access

Lớp Access kết nối và giúp 2 lớp trên (lớp Model và Foundation Model, nếu gộp 2 lớp này lại thì giống với lớp Application trong mô hình OSI) sử dụng tài nguyên là các lớp bên dưới.

- Định nghĩa cấu trúc của 2 lớp trên
- Định nghĩa và điều khiển quá trình mã hóa, giải mã trong lớp Upper transport
- Kiểm tra và xác nhận gói tin được gửi từ lớp Upper transport là đúng mạng và đúng ứng dụng trước khi chuyển lên lớp cao hơn

### 2.3.4 Lớp Upper Transport

Lớp Upper transport chịu trách nhiệm mã hóa, giải mã và xác thực gói tin chuyển xuống (lúc gửi) hoặc trước khi chuyển lên (lúc nhận) lớp Access. Lớp này cũng chịu trách nhiệm tạo ra và vận chuyển các gói tin đặc biệt qua lại giữa các node: bao gồm các gói tin liên quan đến friendship và heartbeats.

### 2.3.5 Lớp Lower Transport

Lớp Lower transport nhận gói tin từ lớp Upper transport layer và chia nhỏ nếu gói tin quá dài, vượt quá giới hạn của lớp này. Trong trường hợp nhận, lớp Lower

transport sẽ chịu trách nhiệm ghép gói tin lại nếu nó đã bị chia nhỏ, sau đó chuyển lên lớp trên.

#### 2.3.6 Lớp Network

Lớp Network định nghĩa các kiểu địa chỉ (unicast, vitural, group) cũng như cấu trúc của gói tin ở lớp này, cho phép gói tin từ lớp Transport đi xuống lớp Bearer. Lớp này hỗ trợ nhiều bearer (một dạng kênh truyền), trong đó mỗi bearer lại mang nhiều network interface, network interface là cách để các element của các node giao tiếp với nhau, bao gồm cả các element trong cùng 1 node (local network interface).

Lúc gửi dữ liệu lớp Network layer sẽ quyết định gửi gói tin tới network interface nào, sau đó lớp này sẽ thông qua bộ lọc để lọc các dữ liệu được đưa xuống lớp Bearer. Lúc nhận gói tin từ lớp Bearer, một bộ lọc sẽ lọc xem gói tin nào sẽ được đưa tiếp lên lớp trên (khi đúng network interface), gói tin nào sẽ bỏ qua. Tính năng Relay và Proxy được hiện thực ở lớp Network.

#### 2.3.7 Lớp Bearer

Các gói tin mesh cần một nền tảng để gửi nhận, nền tảng đó chính là BLE, lớp Bearer sẽ định nghĩa cách các gói tin được xử lý và vận chuyển. Hiện nay có hai Bearer là Advertising Bearer và GATT Bearer.

Advertising Bearer kế thừa tính năng quảng bá (advertising) và dò tìm (scanning) của BLE GAP để gửi nhận các gói tin mesh. Còn GATT Bearer cho phép các thiết bị không hỗ trợ Advertising Bearer giao tiếp gián tiếp với các node trong mạng mesh thông qua giao thức Proxy. Node sử dụng giao thức này gọi là node Proxy, node sử dụng Proxy và node sử dụng Advertising Bearer có thể giao tiếp với nhau một cách bình thường.

Giao thức Proxy sử dụng profile GATT vốn là profile cơ bản nhất của BLE, trong đó giao thức này sẽ sử dụng một số characteristics được định nghĩa sẵn để giao tiếp (Mesh Proxy Data). Gói tin sẽ được write vào characteristics Mesh Proxy

Data In của node proxy, gói tin sẽ được lấy ra từ characteristics Mesh Proxy Out.

## 2.4 Nguyên lý hoạt động của Bluetooth Mesh

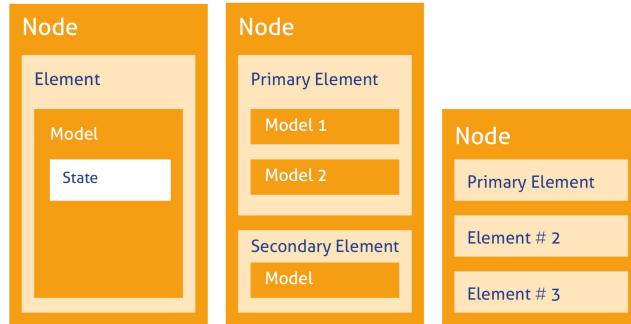
### 2.4.1 Phản tử trong mạng

Mỗi thiết bị tham gia trong mạng được gọi là một node, trong một ứng dụng thực tế thì mỗi một node đều có thể có nhiệm gióng hoặc khác nhau, chẳng hạn như node đọc giá trị cảm biến do tính chất đọc theo chu kỳ nên có thể dùng nguồn pin và chạy low power, node gateway lại cần cấp nguồn liên tục để cập nhật tình hình với server,... Trong mạng Bluetooth Mesh có 4 loại node đặc biệt như sau:

- Low-power: tiêu tốn năng lượng cực thấp, có thể hoạt động trong vòng vài năm với nguồn pin coin (coin battery).
- Friend: bắt buộc phải có nếu trong mạng có node low-power, bởi vì mỗi khi node low-power muốn giao tiếp, nó sẽ chỉ giao tiếp với node friend của nó, còn node friend có nghĩa vụ lưu lại những dữ liệu mà các node khác gửi cho node low-power. Do đó node friend cần cấp nguồn liên tục.
- Relay: node relay sẽ broadcast bất cứ dữ liệu nào được gửi đến nó mà nó không phải người nhận duy nhất, giúp mở rộng mạng mesh. Node relay cũng cần cấp nguồn liên tục.
- Proxy: node này liên kết các node trong mạng mesh với các thiết bị sử dụng BLE và không nằm trong mạng mesh, node proxy cũng cần năng lượng và thêm khả năng tính toán của vi xử lý.

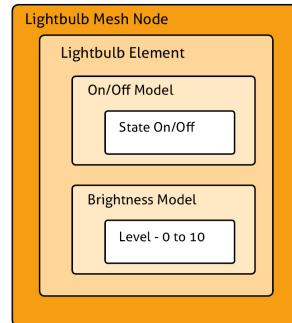
## 2.4. Nguyên lý hoạt động của Bluetooth Mesh

Dù các node có hay không 4 chức năng trên, cấu trúc một node cũng gồm nhiều thành phần nhỏ hơn được thể hiện trong các hình sau:



Hình 2.6: Cấu trúc của một node

Mỗi node phải gồm ít nhất một element - element sẽ định nghĩa các state (trạng thái) hoặc chức năng (functionality) của node. Ví dụ một node điều khiển nhiều thiết bị, các thiết bị là các element và mỗi element lại có nhiều trạng thái khác nhau, mỗi trạng thái là một model (model server). Hình bên dưới minh họa element đèn bao gồm 2 model trạng thái đèn và model độ sáng đèn.



Hình 2.7: Cấu trúc của một node bóng đèn

Các model trong giao thức Bluetooth Mesh được phân làm 3 loại:

- Server: thể hiện trạng thái, như trong ví dụ trên. Sau đó nhận tin từ các node khác để thay đổi trạng thái và xuất tín hiệu ra các chân I/O, PWM tương ứng.

## 2.4. Nguyên lý hoạt động của Bluetooth Mesh

---

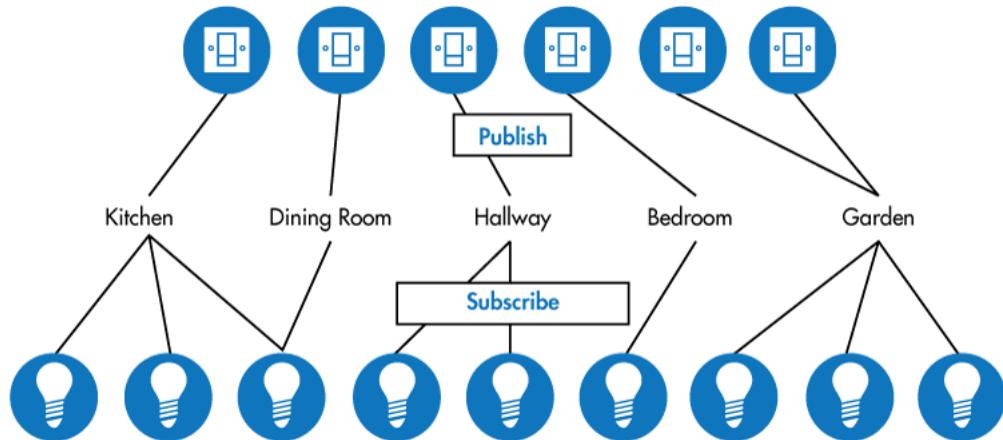
- Client: thể hiện chức năng, bao gồm các hàm có chức năng thay đổi trạng thái của các model server.
- Control: thể hiện cả trạng thái và chức năng, cộng thêm liên hệ logic giữa trạng thái và chức năng. Ví dụ một model control quạt dựa vào nhiệt độ, model này có chức năng lấy dữ liệu nhiệt độ từ cảm biến (của một node khác), sau đó kiểm tra ngưỡng rồi quyết định bật hay tắt quạt.

Trong đề tài này node gateway sẽ dùng model Simple OnOff client, node cảm biến sẽ dùng model Simple OnOff server. Node cảm biến lưu giữ trạng thái đèn (nhóm đã cài tiến thành lưu giữ giá trị cảm biến, xem phần 4.1), node gateway sẽ đọc dữ liệu này để gửi lên server, nhóm có hiện thực thêm chức năng client gửi tín hiệu tắt/mở LED để tiện demo. Do đó có hiện tượng cùng một biến 8-bits nhưng có lúc lưu giữ giá trị cảm biến, có lúc lưu trạng thái đèn.

### 2.4.2 Truyền tin trong mạng

Bluetooth Mesh dùng cơ chế flooding để lan truyền gói tin, nghĩa là gói tin cần gửi sẽ được broadcast ra các node xung quanh, các node được broadcast sẽ kiểm tra mình có phải người nhận hay không, nếu không thì tiếp tục broadcast (còn gọi là relay, các node relay cần cung cấp nguồn đầy đủ và liên tục) cho đến khi tới được người nhận nếu là địa chỉ là unicast (hoặc một nhóm nhận nếu là địa chỉ multicast). Quá trình flooding này dừng lại khi đạt giới hạn 127 hops hoặc vượt quá thông số Time To Live (TTL) - thông số này tương đương số lần gói tin được relay[8].

Sau đó, Bluetooth Mesh dùng cơ chế publish-subscribe để các gói tin tới được nơi cần tới. Các địa chỉ unicast được đánh cho các element trong node và được gọi là publish address, còn có multicast address và virtual address bao gồm nhiều element (địa chỉ multicast thường đại diện cho một tầng lầu, một phòng,... trong thực tế). Khi có tin mới được gửi đến một địa chỉ publish address, bất kỳ những model nào (element bao gồm nhiều model) trong element và có đăng ký nhận tin (subscribe) ở địa chỉ đó sẽ nhận được tin.



Hình 2.8: Sơ đồ của cơ chế publish-subscribe

## 2.5 Quản lý các phần tử trong mạng

### 2.5.1 Tham gia mạng - Provisioning

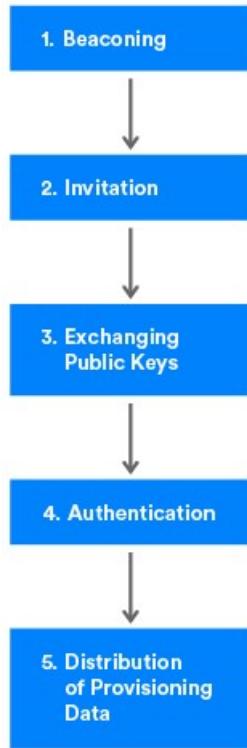
Để một thiết bị mới tham gia vào mạng mesh, cần thực hiện quá trình provisioning. Đây cũng được xem như quá trình bắt tay giữa các thiết bị khi tham gia vào mạng mesh, sau khi hoàn thành quá trình provisioning gồm 5 bước, thiết bị chính thức trở thành node - một thành phần trong mạng. Bluetooth Mesh hỗ trợ hai cách để provision một thiết bị:

- PB-ADV: Sử dụng gói advertising của BLE để tiến hành provisioning bằng cách thay đổi một số trường.
- PB-GATT: Sử dụng các characteristics của GATT để tiến hành provisioning, chỉ dùng khi thiết bị mới không hỗ trợ advertising hoặc không cho tùy chỉnh gói advertising.

Ngoài ra, khi thiết bị mới nằm ngoài tầm phủ sóng của provisioner, Bluetooth cũng hỗ trợ provisioning thông qua các node trung gian hoạt động như các relay để gửi các gói tin liên quan provisioning, quá trình này gọi là remote provisioning. Quá trình này có thể không cần thiết trong đa số ứng dụng trong thực tế vì người dùng hoàn toàn có thể mang theo thiết bị provisioner như smartphone, sau đó đặt

## 2.5. Quản lý các phần tử trong mạng

thiết bị mới vừa tầm và tiến hành provisioning. Có thể tham khảo flowchart của quá trình này trong phần B.



Hình 2.9: Quá trình provisioning gồm 5 bước

Tiếp theo, nhóm sẽ trình bày chi tiết hơn về quá trình provisioning thông qua PB-ADV:

- Bước 1 - Beaconing: Khi một thiết bị mới (unprovisioned) muốn gia nhập mạng mesh, thiết bị này sẽ cần Network Key của mạng đó, sau đó tiến hành phát beacon để thông báo sự hiện diện của mình, tùy nhà sản xuất mà quá trình có thể kích hoạt bằng cách nhấn hoặc giữ nút,... Lúc này thiết bị đóng vai trò provisioner - thường sẽ là điện thoại, máy tính bảng có cài ứng dụng đặc biệt - sẽ tiến hành scan thiết bị mới, khi provisioner phát hiện được thiết bị mới thì tiến hành bước tiếp theo. Trong đề tài này, nhóm sử dụng node gateway để thực hiện chức năng provisioning vì chưa phát triển được ứng dụng trên điện thoại.
- Bước 2 - Invitation: Provisioner sẽ gửi một lời mời, một gói tin Provisioning

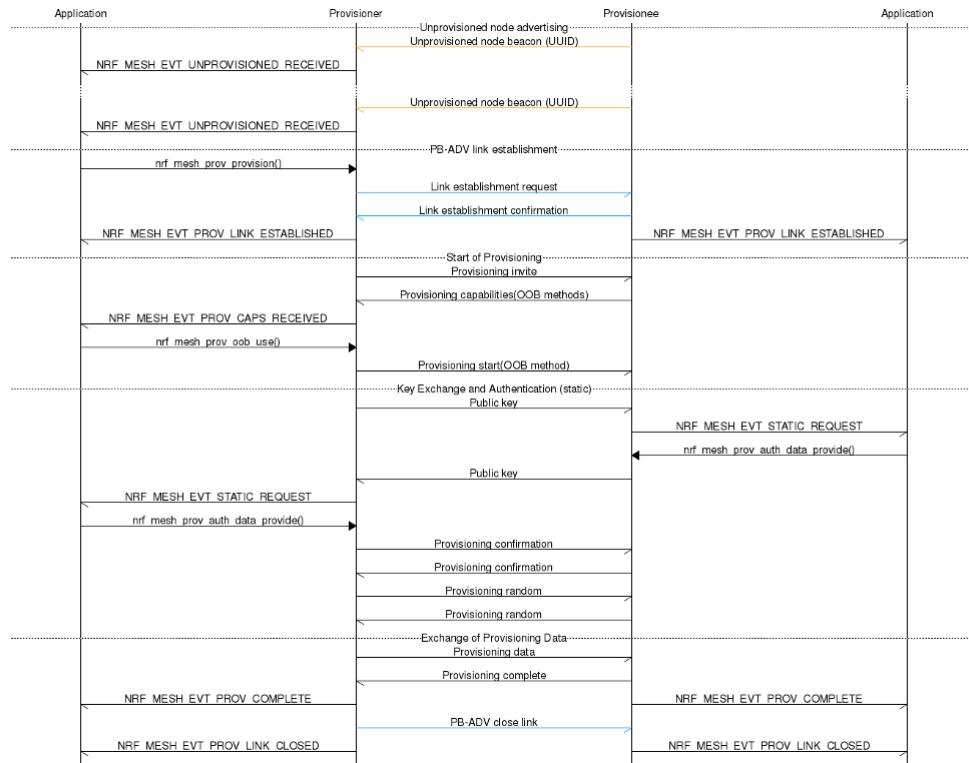
## 2.5. Quản lý các phần tử trong mạng

Invite PDU, đến thiết bị cần provisioning, thiết bị này sẽ đáp lại bằng một gói Provisioning Capabilities PDU, trong đó chứa một số thông tin của nó như số element,... đặc biệt là các input output mà nó hỗ trợ để tiến hành authentication sau này.

- Bước 3 - Exchanging Public Keys: Thiết bị mới và provisioner sẽ tiến hành trao đổi public key để dùng cho 2 bước cuối cùng của quá trình provisioning áp dụng giải thuật mã hóa bất đối xứng FIPS P-256 Elliptic Curve Algorithm. Quá trình trao đổi này có thể dùng phương pháp out-of-band để tăng tính bảo mật, out-of-band bao gồm các cách như quét QR code,...
- Bước 4 - Authentication: Sau khi biết được các input output mà thiết bị mới hỗ trợ nhờ bước 2, provisioner sẽ gửi yêu cầu sử dụng một trong các phương thức output, chẳng hạn như màn hình LCD hoặc LED, để thể hiện một hoặc nhiều con số (giống với mã PIN khi kết nối Bluetooth giữa 2 điện thoại). Người dùng sẽ quan sát và nhập con số này vào ứng dụng, provisioner sẽ gửi và thiết bị mới sẽ kiểm tra để hoàn thành bước này (input/output authentication), hoặc đơn giản hơn là định nghĩa sẵn một dạng mật khẩu nào đó và để 2 bên xác thực bằng mật khẩu này (static authentication). Quá trình kiểm tra này có áp dụng hash để bảo mật.
- Bước 5 - Distribution of the Provisioning Data: Provisioner sẽ gửi Network Key và cấp Unicast Address cho các element (quá trình này sẽ áp dụng giải thuật mã hóa FIPS P-256 Elliptic Curve), lúc này thiết bị mới đã chính thức trở thành node - một thành viên trong mạng mesh.
- Nếu có lỗi xảy ra trong quá trình provisioning, kết nối giữa hai thiết bị sẽ bị hủy.

## 2.5. Quản lý các phần tử trong mạng

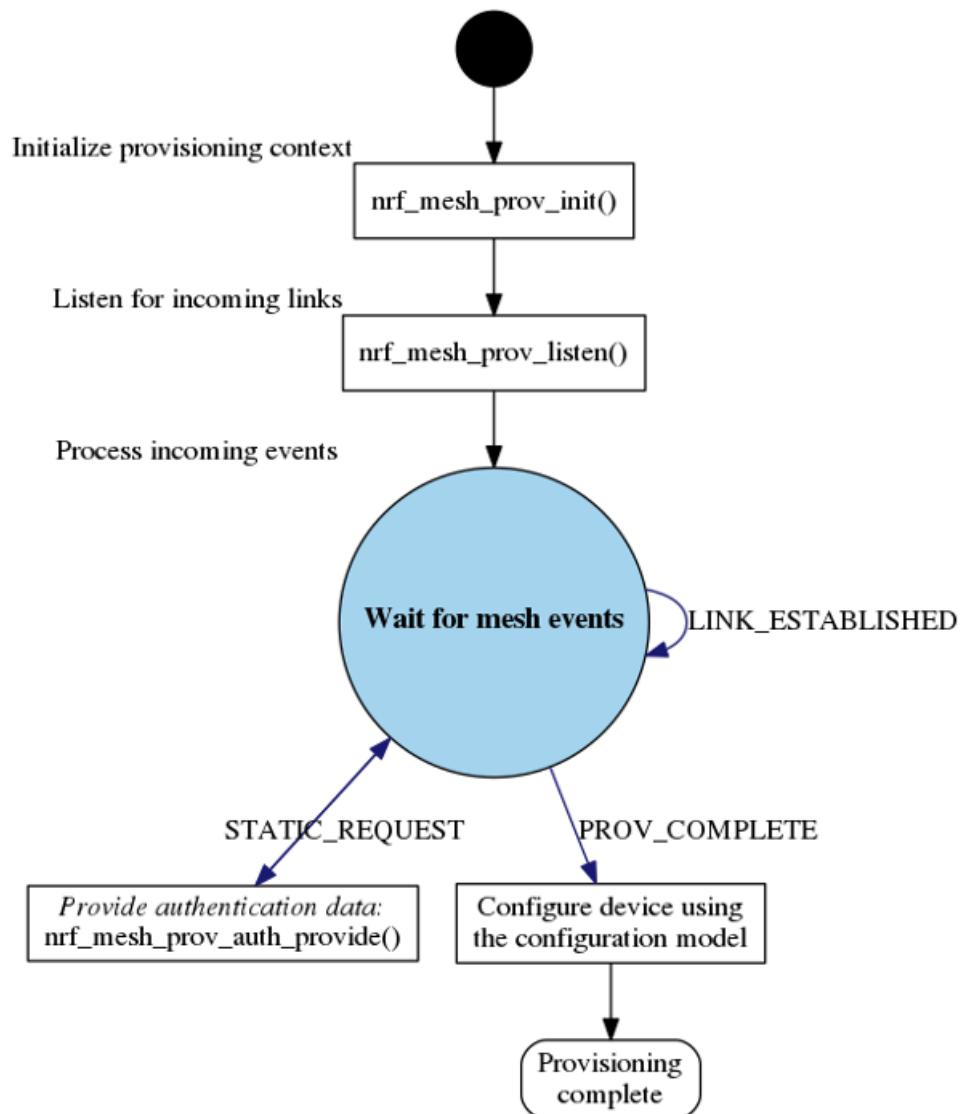
Sơ đồ dưới đây minh họa chi tiết hơn các bước của quá trình provisioning, dùng để tham khảo khi lập trình với Nordic Mesh SDK:



Hình 2.10: Provisioning scenarios

### 2.5.2 Thiết bị mới - Provisionee

Cách thức hoạt động của một thiết bị mới - unprovisioned hay provisionee: Tiến hành khởi tạo - bao gồm phát beacon để báo rằng mình cần tham gia provisioning, sau đó ở trạng thái chờ sự kiện. Khi bắt được sự kiện yêu cầu kết nối - lời mời provisioning từ provisioner thì đáp lại, tiếp đến chờ đáp lại các sự kiện xác thực, cuối cùng là kết thúc quá trình.



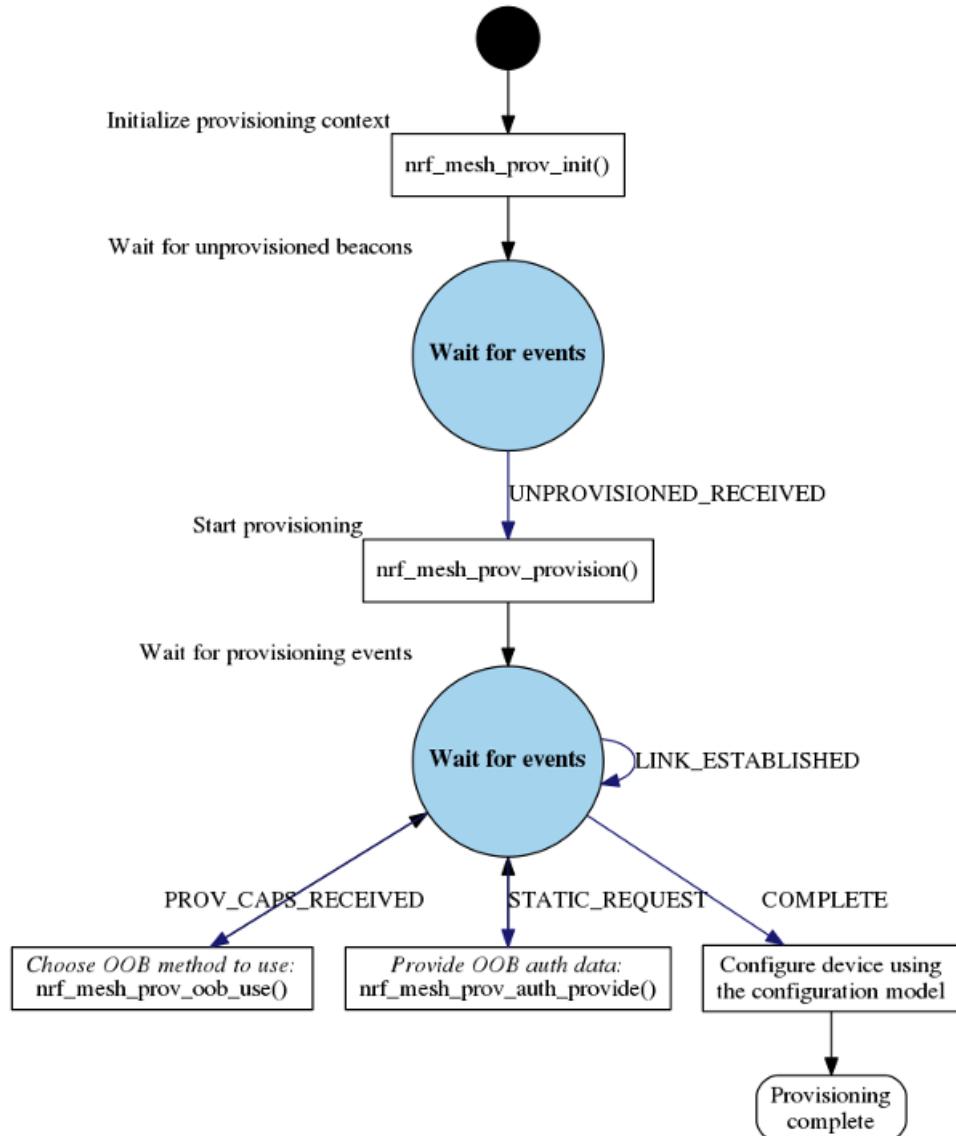
Hình 2.11: Provisionee flowchart

### 2.5.3 Thiết bị Provisioner

Là thiết bị hoặc node chịu trách nhiệm thực hiện quá trình provisioning trong mạng mesh. Chúng thường là một phần của các thiết bị gateway - các thiết bị cung cấp một cầu nối giữa mạng mesh và các giao thức kết nối khác. Cách thức hoạt động của một provisioner: sau các bước khởi tạo thì chờ beacon từ thiết bị mới (unprovisioned), sau đó mời tham gia quá trình provisioning. Trong bước xác thực, tùy theo đối tượng cần provisioning có hay không có hỗ trợ xác thực out-of-band

## 2.5. Quản lý các phần tử trong mạng

(OOB) mà tiến hành xác thực theo cách tương ứng.



Hình 2.12: Provisioner flowchart

### 2.5.4 Rời khỏi mạng

Sau một thời gian hoạt động, việc phải loại bỏ một số node ra khỏi mạng là điều không thể tránh khỏi, có thể vì nhiều lý do như: node bị hỏng, chuyển node sang một khu vực khác với một mạng mesh khác, thay node mới,... Trong trường hợp nếu không có quy trình loại bỏ node, mạng sẽ đổi mất với nguy cơ bị tấn công kiểu

Trashcan, kẻ tấn công có thể khai thác những node bị bỏ đi, lúc này chắc chắn lớp bảo vệ sẽ kém, để lấy thông tin và xâm nhập mạng. Bluetooth Mesh áp dụng 2 bước xử lý sau:

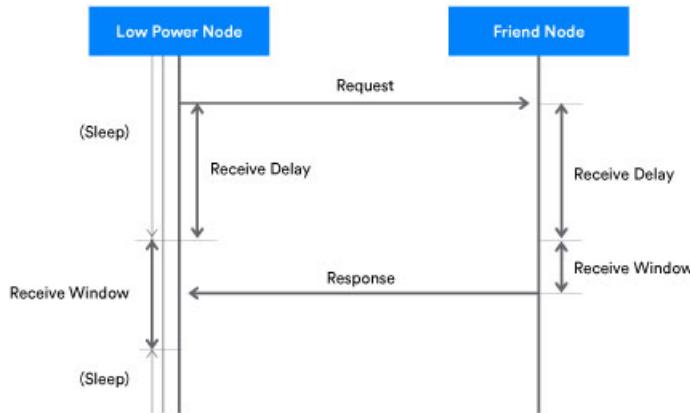
- Cho node bị loại bỏ vào danh sách đen.
- Tiến hành cấp lại NetKey cho các node trong mạng trừ node trong danh sách đen.

Quá trình này cần có một thời gian chuyển tiếp, trong thời gian này cả 2 key cũ và mới đều hợp lệ. Sở dĩ cần có thời gian chuyển tiếp này là vì các node low-power thỉnh thoảng mới hoạt động, do đó cần có thời gian để chúng nhận được chìa mới từ node friend của mình.

## 2.6 Khái niệm Friendship

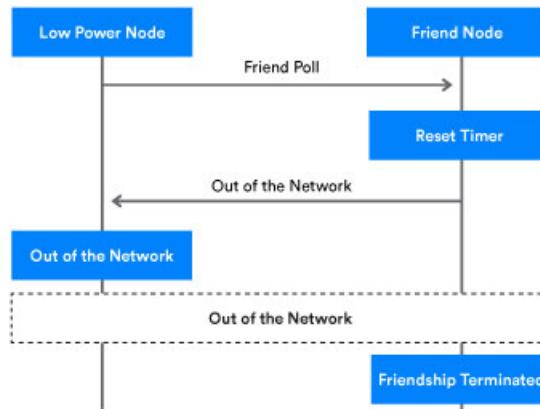
Để có thể ở trong trạng thái sleep lâu dài, các node low power (LPNs) cần tiến hành thiết lập mối quan hệ Friendship với một node “hàng xóm” của mình, sau khi thiết lập xong thì node đó trở thành “friend” của LPN. Trước khi đi sâu hơn về khái niệm này, chúng ta cần làm quen với một vài tham số:

- ReceiveDelay: Khoảng thời gian từ lúc LPN gửi yêu cầu đến node friend cho đến lúc nó thức dậy lần nữa để bắt đầu nhận dữ liệu. Trong thời gian này LPN vẫn tiếp tục sleep, node friend thì tiến hành chuẩn bị dữ liệu để đáp lại yêu cầu đó.
- ReceiveWindow: Khoảng thời gian LPN thức dậy và lắng nghe, lúc này node friend mới có thể gửi dữ liệu cho LPN.



Hình 2.13: Hình mô tả ReceiveDelay và ReceiveWindow

- PollTimeout: Khoảng thời gian tối đa giữa 2 lần LPN gửi yêu cầu cho node friend, nếu quá thời gian này mà vẫn không nhận được yêu cầu từ LPN, node friend sẽ kết thúc “tình bạn” với nó.



Hình 2.14: Hình mô tả PollTimeout

### 2.6.1 Các bước thiết lập Friendship

1. Node LPN gửi “yêu cầu kết bạn”, yêu cầu này sẽ không được relay vì nó chỉ có thể kết bạn với những node trong phạm vi kết nối của mình, ngoài ra các node trong phạm vi nhưng không hỗ trợ chức năng Friendship cũng sẽ bỏ

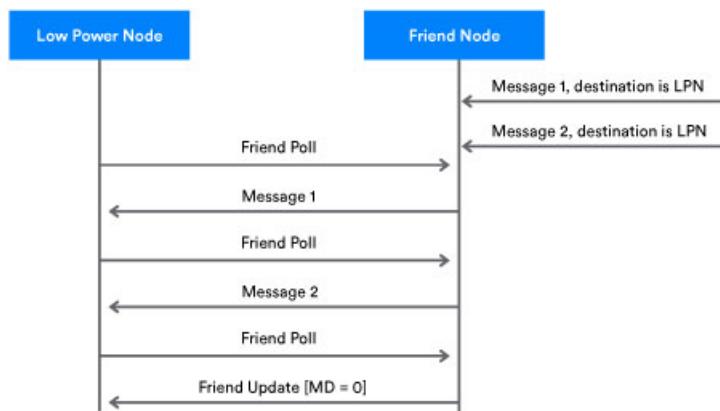
## 2.6. Khái niệm Friendship

qua yêu cầu này. Yêu cầu kết bạn này bao gồm cả 3 thông số ReceiveDelay, ReceiveWindow và PollTimeout.

2. Node thỏa mãn các yêu cầu để trở thành Friend sẽ gửi lại Friend Offer cho LPN. Gói này bao gồm nhiều tham số như ReceiveWindow có thể hỗ trợ, sức chứa của buffer - buffer này để chứa “hộ” các dữ liệu được gửi tới LPN trong lúc nó ngủ và được gọi là Friend Queue,...
3. Khi LPN nhận được Friend Offer, nó sẽ dùng một thuật toán để chọn ra 1 Friend (nếu có nhiều Friend Offer), thuật toán này do người phát triển ứng dụng định nghĩa: có thể là ưu tiên node gần hơn hoặc node có khả năng hỗ trợ Friend Queue lớn hơn.
4. Sau khi đã chọn được Friend, LPN gửi Friend Poll cho node đó.
5. Sau khi nhận được Friend Poll, node Friend trả lời bằng Friend Update. Mỗi quan hệ Friendship chính thức được thiết lập.

### 2.6.2 Giao tiếp trong Friendship

Sau khi đã “kết bạn”, LPN và node Friend của nó sẽ tiến hành giao tiếp theo các bước sau:



Hình 2.15: Hình mô tả quá trình giao tiếp

## **2.6. Khái niệm Friendship**

---

1. Khi node Friend nhận được dữ liệu được gửi đến node LPN là “bạn” của nó, nó sẽ lưu vào Friend Queue. Trong hình trên, có thể thấy node Friend đã lưu lại Message 1 và Message 2 thay cho node LPN.
2. Sau một chu kỳ ngủ, node LPN sẽ gửi yêu cầu Friend Poll đến cho node Friend của mình để kiểm tra có dữ liệu mới hay không.
3. Nếu có dữ liệu mới, node Friend sẽ gửi ngược lại cho LPN, cứ mỗi lần LPN nhận được dữ liệu nó lại gửi thêm Friend Poll.
4. Đến khi nào nhận được Friend Update (với tham số More Data = 0) nghĩa là không còn dữ liệu nữa, quá trình giao tiếp này sẽ kết thúc, LPN không gửi Friend Poll nữa.

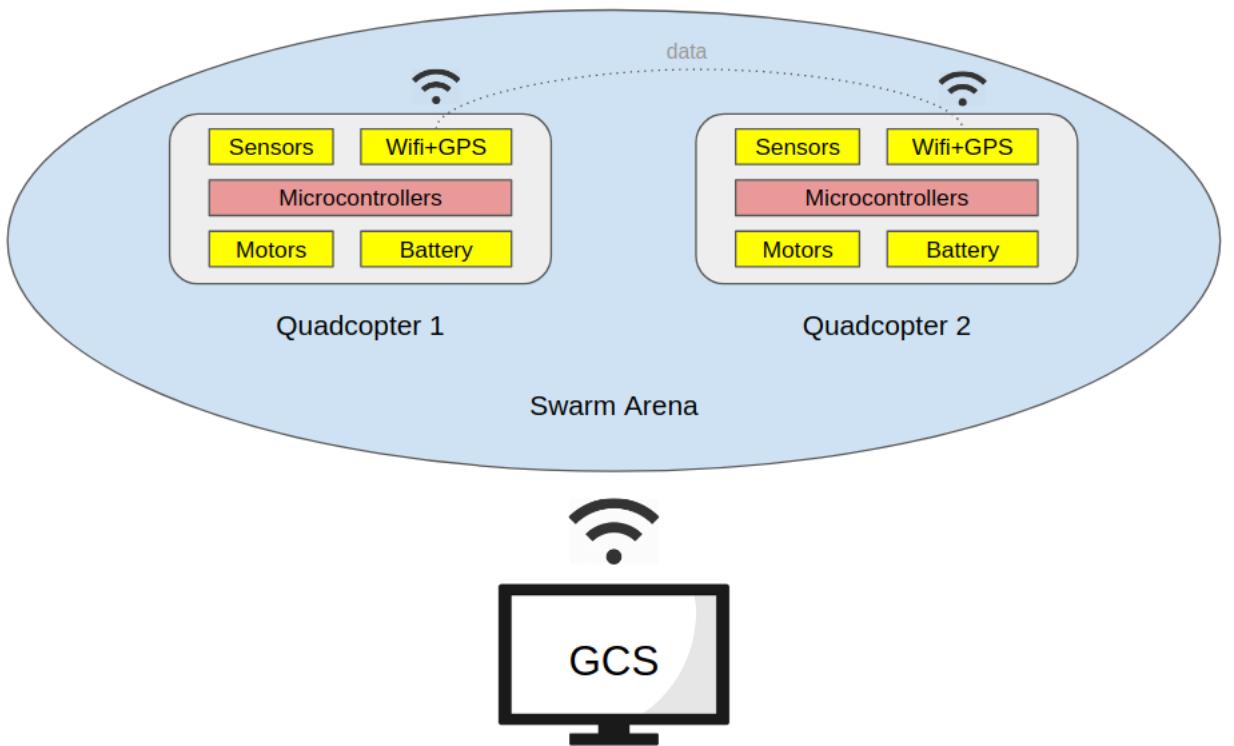
Các gói tin giữa node Friend và LPN đều được mã hóa bằng một loại key Friend và chỉ có 2 node này mới có khả năng đọc được.

## **3** Thiết kế

### **3.1 Mô hình thiết kế**

Với mục tiêu đề tài giúp các quadcopter bay được ổn định đồng thời vẫn giữ được trạng thái bầy đàn nhóm để xuất ra mô hình bay như sau:

- Mô hình sẽ có ít nhất hai chiếc quadcopter có thể hoạt động độc lập ổn định. Như vậy cần có các cảm biến cần thiết để triển khai phương pháp điều khiển PID cho quadcopter. Ngoài ra còn cần mạch thu phát sóng Wifi và một vi xử lý phụ nhằm xử lý dữ liệu bay.
- Trạm điều khiển mặt đất chạy phần mềm để thiết lập thông số các quadcopter đồng thời hiển thị dữ liệu và điều khiển các quadcopter bay theo các chế độ mong muốn.
- Mỗi chiếc quadcopter kết nối với trạm điều khiển đồng thời trao đổi dữ liệu cảm biến qua lại với nhau nhằm tính toán để giữ đội hình bầy đàn.



Hình 3.1: Mô hình thiết kế

## 3.2 Phần cứng

### 3.2.1 IMU - Đơn vị đo lường quán tính

### 3.2.2 MCU - Vi điều khiển

### 3.2.3 ESC - Bộ điều tốc

### 3.2.4 ESP8266

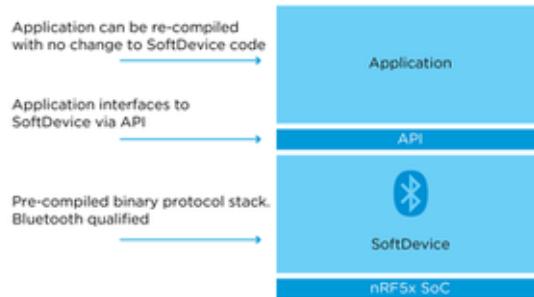
### 3.2.5 Ultrasonic Sensor - Cảm biến siêu âm

## 3.3 Phần mềm

### 3.3.1 S132 Softdevice

S132 Softdevice hiện thực các lớp trong mô hình BLE được viết cho chip nRF52832<sup>29</sup>, hỗ trợ 4 vai trò trong giao thức BLE: Central, Peripheral, Broadcaster, Observer.

Softdevice đã được build thành file nạp (.bin), người dùng nạp file vào và tương tác với lớp Softdevice này thông qua API được cung cấp. Nhóm sử dụng phiên bản Softdevice s132 v3.1.0[15].



Hình 3.2: Các lớp trong một chương trình

#### 3.3.2 nRFgo Studio

nRFgo Studio là chương trình dùng để nạp Softdevice cho các KIT phát triển, hỗ trợ nhiều chức năng liên quan đến Bluetooth, hiển thị trạng thái bộ nhớ hiện tại của chip,...

#### 3.3.3 Nordic Mesh SDK

Trong đề tài này, nhóm sử dụng nRF5 SDK for Mesh phiên bản v1.0.1 vì đây là phiên bản mới nhất vào thời điểm bắt đầu thực hiện luận văn. Ở thời điểm báo cáo luận văn thì Nordic đã cho ra mắt phiên bản v2.0.1 với một số thay đổi như: thêm chức năng provisioning thông qua GATT (PB-GATT), hỗ trợ chức năng Proxy cho các node và đặc biệt nhất: Mesh stack chính thức hoạt động ổn định.

Ngoài ra còn có một số tính năng mà Nordic đang và sẽ cung cấp trong SDK này. Lưu ý: Một số tính năng chưa được hiện thực trong phiên bản SDK hiện tại.

- Device State Manager (DSM): DSM lưu trữ vào bộ nhớ flash những thông tin như Network key của mạng hiện tại, địa chỉ của các element,... Sau khi quá trình provisioning cho node mới hoàn tất, DSM sẽ lưu trữ thêm các thông

tin của model, bao gồm địa chỉ và key để các model khác có thể giao tiếp với model này.

- Device Firmware Update (DFU): DFU giúp lập trình viên cập nhật firmware cho tất cả node trong mạng mà không cần tiếp xúc trực tiếp với từng node bằng cách áp dụng khả năng lan truyền của mesh.
- Hỗ trợ kết nối GATT/GAP, liên kết mạng mesh với các thiết bị như máy tính, điện thoại thông minh,...
- Serial, RTT: Cung cấp kênh giao tiếp giữa chip và máy tính, có thể truyền lệnh hay dữ liệu qua lại.

#### 3.3.4 Nordic SDK

SDK này cung cấp các API để người lập trình dễ dàng lập trình trên các dòng chip của Nordic, cụ thể là chip nRF52832 trong đề tài này. Tuy nhiên do phát triển không đồng thời với Mesh SDK nên khi sử dụng chung có thể gây ra xung đột, đặc biệt là các ứng dụng có dùng Softdevice vì đa phần các API của Nordic SDK không tương thích với Softdevice. Nhóm sử dụng Nordic nRF5 SDK v14.2.0[7].

#### 3.3.5 Thư viện giao tiếp module SIM

Thư viện do nhóm tự phát triển và sử dụng trong một dự án trước đây, nhằm mục đích giao tiếp với module 3G SARA click. Thư viện có sẵn một số API cung cấp chức năng cơ bản như: khởi tạo, nghe gọi, xử lý tin nhắn, GPRS, HTTP request,... Trong đề tài này nhóm hiện thực thêm phần giao tiếp TCP để kết nối với cloud.

#### 3.3.6 ThingSpeak

Nhóm sử dụng một cloud thông dụng dành cho các ứng dụng IoT giám sát đơn giản là ThingSpeak. Dữ liệu sau khi upload lên sẽ được hiển thị dạng biểu đồ và bất kỳ ai có liên kết đều có thể quan sát.

#### **3.3.7 Embedded Studio**

Chương trình hoàn toàn miễn phí, đóng vai trò IDE được Nordic khuyên dùng, các ví dụ được cung cấp trong các SDK đều được viết dưới dạng 1 project Segger Embedded Studio. Đặc biệt các project này đã cấu hình sẵn địa chỉ nạp để tránh đụng độ với Softdevice.

# 4 Hiện thực ứng dụng

## 4.1 Chính sửa model Simple OnOff

### 4.1.1 Model Simple OnOff

Trong Mesh SDK v1.0.1 của Nordic có cung cấp sẵn một model là Simple OnOff, cho phép gửi/nhận một 1 bit. Node gateway có thể gửi tín hiệu để tắt/mở LED trên node cảm biến bằng hàm set, cũng như dùng hàm get để yêu cầu node cảm biến gửi trạng thái hiện tại của LED cho mình.

### 4.1.2 Khó khăn

Tuy nhiên vì mục đích chỉ là một ví dụ cho người dùng làm quen với cách lập trình sử dụng SDK nên cách hoạt động của model Simple OnOff cũng rất đơn giản, nếu muốn sử dụng model này vào trong đề tài, buộc nhóm phải chỉnh sửa lại model này hoặc tạo một model mới. Nhóm đã tham khảo trong tài liệu của Bluetooth Mesh, cụ thể là Mesh Model Specification 1.0 [6], phát hiện SIG có định nghĩa một model Sensor với đầy đủ các thuộc tính ID, chu kỳ, hàm đọc giá trị,... nghĩa là trong tương lai Mesh SDK sẽ hỗ trợ model này. Do đó việc tạo một model mới để đọc giá trị cảm biến là không cần thiết, nhóm sẽ chỉnh sửa model Simple OnOff để phục vụ đề tài.

### 4.1.3 Thực hiện

Việc can thiệp vào model Simple OnOff đòi hỏi nhóm phải đọc mã nguồn và tìm ra những điểm liên quan đến gói tin giao tiếp của model này. Kiểu dữ liệu của gói tin được định nghĩa trong file simple\_on\_off\_common.h, còn các hàm liên quan đến việc get/set có trong file simple\_on\_off\_client.h và simple\_on\_off\_server.h, nhóm chỉnh sửa các kiểu dữ liệu để đồng bộ với nhau và model này đã có thể gửi những dữ liệu với kích thước lớn hơn, giới hạn là 10 bytes cho gói đơn lẻ, 384 bytes cho gói fragmented (chia nhỏ gói tin để gửi nhiều lần, sau đó ghép lại)[14]. Sở dĩ có giới hạn này là Bluetooth Mesh sử dụng sóng radio, mà gói tin nhỏ sẽ giúp giảm bớt dung độ, giới hạn 10 bytes đã được tính toán đủ cho một số ứng dụng thông thường.

## 4.2 Kết hợp thư viện SDK common

### 4.2.1 Khó khăn

Việc hiện thực ứng dụng đòi hỏi phải có timer để xử lý các chu kỳ, UART để giao tiếp với module SIM (khác với UART dùng để giao tiếp với máy tính), nhưng Mesh SDK lại không hỗ trợ hai thư viện này mà chỉ có một thư viện hal (Hardware abstract layer) rất khó sử dụng. Do đó nhóm cần phải kết hợp Mesh SDK với SDK common của Nordic cung cấp để lập trình trên kit phát triển của hãng, phiên bản dùng trong đề tài là nRF5\_SDK\_14.2.0[7].

### 4.2.2 Thực hiện

Nhóm thử tiến hành kết hợp bằng 2 hướng tiếp cận:

- Chép các file từ common SDK sang mesh SDK: không thành công vì xuất hiện rất nhiều lỗi liên quan đến việc liên kết giữa các header và file mã nguồn nhưng vì số lượng file quá nhiều nên không sửa hết các liên kết này.
- Chép các file từ mesh SDK sang common SDK, hướng này cũng là hướng dẫn của chính Nordic: không thành công vì khi liên quan đến softdevice, một số drivers của common SDK bị xung đột.

### **4.3. Hiện thực node cảm biến**

---

Sau đó, nhóm sử dụng giải pháp chép các file từ common SDK sang mesh SDK nhưng không phải toàn bộ mà chép từng phần. Nếu cần dùng UART thì chép những file liên quan đến UART, cần dùng Timer thì chép file liên quan đến Timer và cách này đã hoạt động tốt.

## **4.3 Hiện thực node cảm biến**

### **4.3.1 Đọc giá trị cảm biến**

Node cảm biến không thực hiện thêm chức năng gì so với mã nguồn mẫu ban đầu của SDK, chỉ chỉnh sửa hàm get\_cb (được kích hoạt khi node gateway gửi yêu cầu đọc dữ liệu cảm biến) để hàm trả về giá trị cảm biến hiện tại. Việc đọc giá trị cảm biến được softdevice hỗ trợ sẵn với hàm sd\_temp\_get().

## **4.4 Hiện thực node gateway**

### **4.4.1 Thư viện giao tiếp với module SIM**

Thư viện này nhóm viết bằng ngôn ngữ C++ mà cả common SDK và mesh SDK đều dùng C, nên khi sử dụng trong đề tài này đã gặp phải khó khăn: không compile được. Nhóm phát triển dựa trên mã nguồn và file Segger Embedded Studio nên mọi thông số mặc định đều dành cho ngôn ngữ C, cách giải quyết lại chỉnh sửa thư viện của nhóm để phù hợp với compiler.

# 5 Kịch bản thử nghiệm

## 5.1 Mục tiêu của thử nghiệm

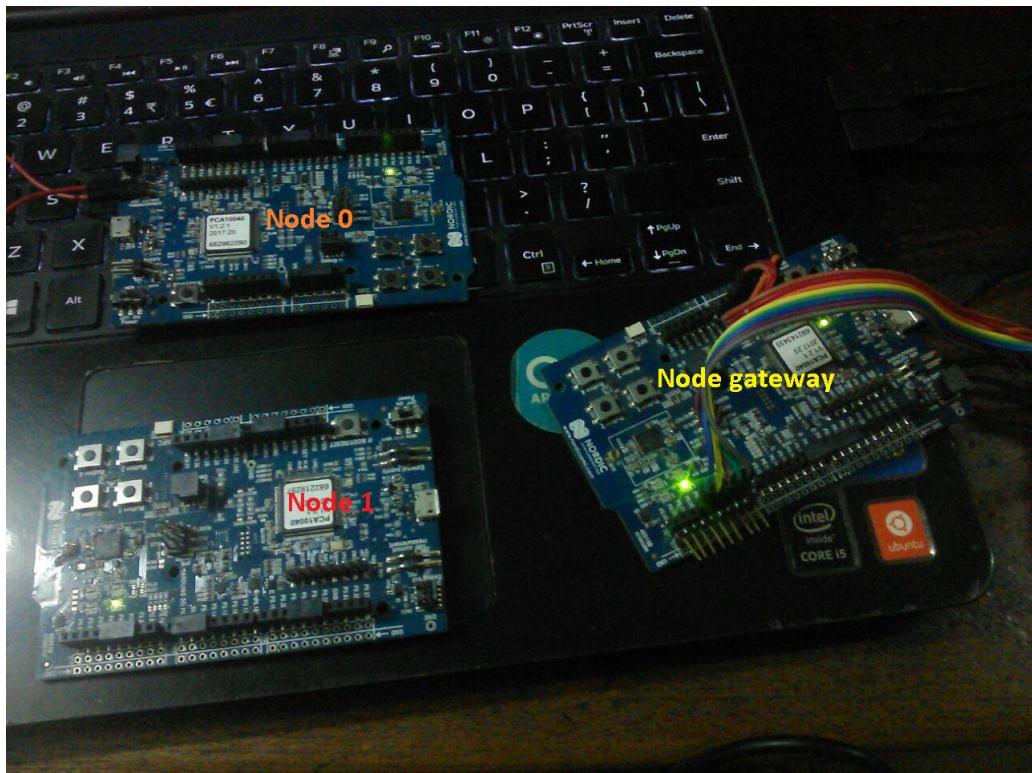
- Kiểm tra hoạt động của ứng dụng: node gateway thu thập dữ liệu từ các node cảm biến sau đó gửi lên cloud, ấn nút trên node gateway để điều khiển LED trên các node cảm biến.
- Chứng minh ứng dụng được hiện thực bằng giao thức mạng Mesh, cụ thể là vai trò của node relay.
- Chứng minh các node trong mạng có thể giao tiếp với nhau bằng cả 2 chiều gửi và nhận: node cảm biến gửi dữ liệu cho node gateway và nhận tín hiệu bật/tắt LED từ node gateway.
- Kiểm tra khoảng cách hoạt động tối đa giữa các node.

## 5.2 Quá trình thử nghiệm

Nhóm thử nghiệm với 3 board phát triển, một board là node gateway còn lại là node cảm biến lần lượt gọi là node 0 và node 1. Đầu tiên cần tiến hành provision cho 2 node cảm biến: chỉ cần bật nguồn node gateway trước, rồi lần lượt bật nguồn node 0 và node 1. Sau đó tiến hành thử nghiệm bằng các kịch bản sau:

### 5.2.1 Kịch bản I: Thử nghiệm giao tiếp

- Đặt 2 node 0 và node 1 ở gần node gateway, sau đó thử ấn nút gửi tín hiệu điều khiển LED thì thấy LED trên cả 2 node cảm biến chớp tắt.



Hình 5.1: Hình ảnh thử nghiệm I trong thực tế

- Sau đó thử kích hoạt chế độ đọc và gửi dữ liệu cảm biến thì thấy node gateway nhận được dữ liệu và gửi lên server thành công. Có thể thử thay đổi dữ liệu bằng cách tăng nhiệt độ ở khu vực gần SoC nRF52832 trên Kit phát triển.

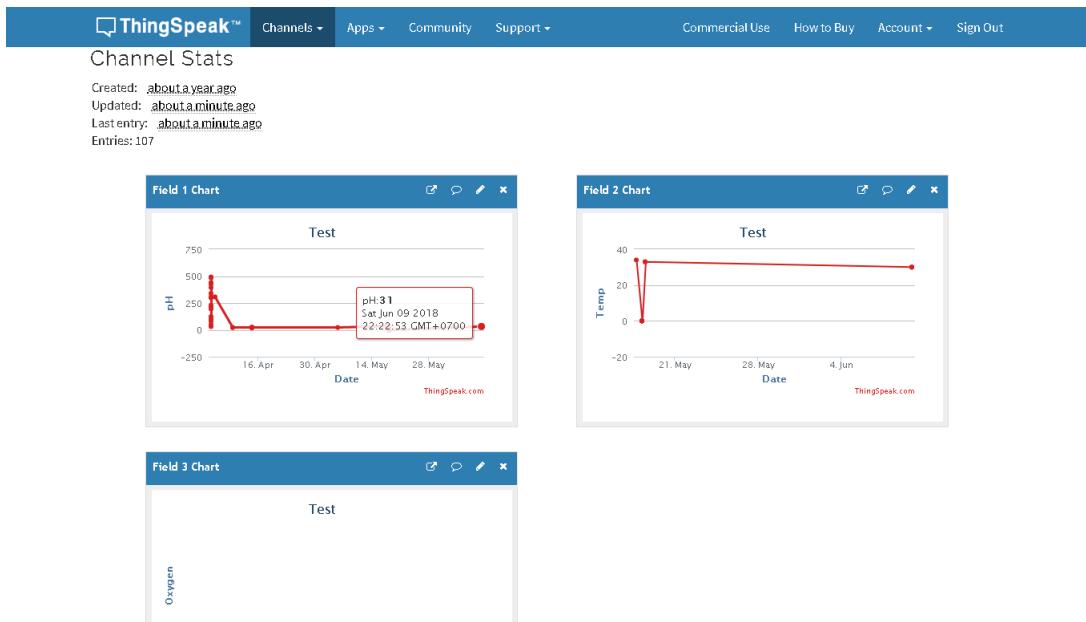
```

0> <: 7034218>, at_mobile.c.c, 402, Data is: /update?api_key=YRSGAFOJYKQXR63&field1=31&field2=30
0> <: 7108431>, main.c, 558, Turn off GPRS to save data charge
0> <: 7196967>, main.c, 602, Upload data success
0> <: 7222885>, main.c, 460, App timer event
0> <: 7222885>, main.c, 290, Update temperature 31 from node 0
0> <: 7527040>, main.c, 292, Update temperature 30 from node 1
0> <: 7688838>, at_mobile.c.c, 402, Data is: /update?api_key=YRSGAFOJYKQXR63&field1=31&field2=30
0> <: 7754574>, main.c, 558, Turn off GPRS to save data charge
0> <: 7840518>, main.c, 602, Upload data success

```

Hình 5.2: Hình ảnh note gateway nhận được dữ liệu từ node cảm biến

## 5.2. Quá trình thử nghiệm



Hình 5.3: Hình ảnh dữ liệu được cập nhật lên cloud

- Thử tắt nguồn node 0 thì thấy node gateway báo lỗi không đọc được dữ liệu từ node 0, khi bật nguồn lại thì hoạt động bình thường trở lại.

**Kết quả thử nghiệm của kịch bản I chứng minh khả năng hoạt động của ứng dụng khi không có lỗi phát sinh.**

### 5.2.2 Kịch bản II: Thử nghiệm khoảng cách

- Tiến hành với một node gateway và một node cảm biến là node 0, node gateway sẽ gửi lệnh điều khiển LED liên tục bằng cách nhấn nút.
- Sau đó dời node cảm biến dần dần ra xa cho đến khi LED trên node cảm biến không còn chớp tắt nữa thì dừng lại.
- Khoảng cách giữa node gateway và node 0 lúc này là khoảng cách tối đa giữa 2 node của Bluetooth Mesh. Sau đó dời node 0 ra xa thêm khoảng 2m để tiến hành thử kịch bản III.

**Kết quả thử nghiệm của kịch bản II chứng tỏ khoảng cách giao tiếp tối đa giữa 2 node trong thực tế (không có vật cản) là khoảng từ 8-10m.**

## 5.2. Quá trình thử nghiệm



Hình 5.4: Hình ảnh thử nghiệm II trong thực tế

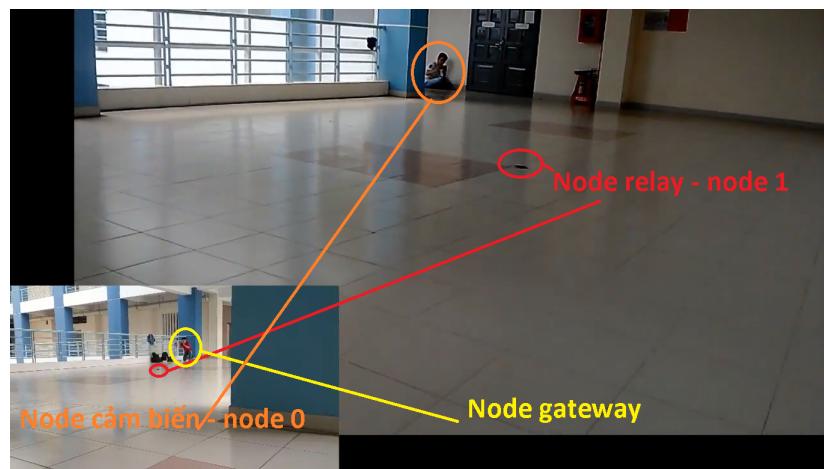
### 5.2.3 Kịch bản III: Thử nghiệm truyền dữ liệu trong mạng mesh



Hình 5.5: Hình ảnh thử nghiệm III

- Sau khi node 0 đã nằm ngoài phạm vi giao tiếp với node gateway, đặt node 1 vào khoảng giữa 2 node này.

## 5.2. Quá trình thử nghiệm



Hình 5.6: Hình ảnh thử nghiệm III trong thực tế

- Sau đó tiến hành gửi tín hiệu điều khiển LED bằng node gateway, lúc này LED trên cả node 0 lẫn node 1 đều chớp tắt, chứng tỏ tín hiệu đã được node 1 relay lại và node 0 đã nhận được tín hiệu điều khiển (tín hiệu này gửi multicast nên cả 2 node 0 và 1 đều chớp LED).



Hình 5.7: Hình ảnh điều khiển LED trong thực tế

Kết quả thử nghiệm của kịch bản III chứng tỏ rằng node 1 đã đóng vai trò relay, giúp chuyển tiếp dữ liệu từ node gateway đến cho node 0.

# 6 Kết luận

## 6.1 Đánh giá kết quả

### 6.1.1 Thành quả đạt được

- Tổng kết thành quả của nhóm trong quá trình nghiên cứu lý thuyết về Bluetooth Mesh: nguyên nhân ra đời, tiềm năng cạnh tranh, sơ lược về các lớp trong kiến trúc, cách truyền nhận dữ liệu giữa các node trong mạng, cách quản lý việc kết nối/ngắt kết nối.
- Kết quả nghiên cứu có thể được dùng làm tài liệu tham khảo cho những ai muốn làm quen với Bluetooth Mesh, giúp tiết kiệm thời gian tiếp cận.
- Hiện thực thành công ứng dụng sử dụng giao thức Bluetooth Mesh để giao tiếp. Mặc dù ứng dụng chưa hoàn thiện để có thể đưa vào thực tế, tuy nhiên cũng chứng minh được dữ liệu có thể đi theo 2 chiều gửi và nhận.

### 6.1.2 Một số hạn chế của ứng dụng thử nghiệm

- Chưa hỗ trợ thiết lập các thông số một cách linh hoạt: chu kỳ đọc, chu kỳ gửi, thông số cloud,...
- Chưa hỗ trợ lấy thông tin thiết bị
- Chưa hỗ trợ xóa node khỏi mạng
- Chưa hỗ trợ cơ chế xác thực mạnh khi provisioning

## 6.2 Hướng phát triển

Do không đủ nhân lực nên nhóm chưa hiện thực tốt ứng dụng thử nghiệm, tuy nhiên nhóm có đề xuất một số cải tiến để ứng dụng thử nghiệm có thể trở thành ứng dụng trong thực tế:

- Tăng khả năng bảo mật bằng cách tăng giảm thông số node cảm biến tối đa trong mạng thông qua node gateway. Ví dụ hiện đang có 2 node cảm biến trong mạng, thông số node tối đa là 2, node cảm biến tiếp theo muốn tham gia vào mạng phải chờ node gateway tăng thông số node tối đa lên mới tham gia được. Bước này giúp tránh việc có node cảm biến tình cờ có được key vào mạng liền vào mạng với ý định xấu.
- Hiện thực giao diện cho node gateway để dễ dàng tiến hành các thao tác cấu hình như: thiết lập chu kỳ đọc dữ liệu, chu kỳ gửi dữ liệu, địa chỉ remote server, số node cảm biến tối đa, xóa node cảm biến ra khỏi mạng,...
- Hiện thực giao diện người dùng lấy dữ liệu từ remote server.
- Hiện thức một giao thức giao tiếp giữa các node để tránh tình trạng nghẽn mạng, quy trình xử lý khi node gateway mất kết nối hoặc node cảm biến mất kết nối.

### 6.2.1 Tiềm năng trong thực tế

- Hệ thống đèn lớn: Yêu cầu chính của hệ thống này là một bộ điều khiển có thể điều khiển một nhóm (có thể là cùng một tầng hoặc cùng một phòng) hay toàn bộ đèn, Bluetooth Mesh cực kỳ thích hợp trong ứng dụng này. Chỉ với một node client điều khiển và hàng ngàn node server với tốc độ đáp ứng nhanh, chức năng điều khiển đơn giản chỉ là bật tắt - thậm chí trong tương lai model này còn hỗ trợ độ mạnh yếu của đèn - là hệ thống có thể hoạt động tốt.
- Hệ thống cảm biến lớn: Những hệ thống như dây chuyền sản xuất tự động trong công nghiệp đòi hỏi có rất nhiều cảm biến đặt rải rác ở nhiều khâu trên dây chuyền. Tất cả các cảm biến đó sẽ gửi toàn bộ thông tin về một node

## **6.2. Hướng phát triển**

---

trung tâm, sau đó node trung tâm này sẽ gửi 1 loạt dữ liệu đó lên server hoặc lưu vào một loại database nào đó, vì số lượng cảm biến trong hệ thống rất nhiều nên nếu mỗi node mỗi gửi dữ liệu sẽ dẫn đến một cuộc "DoS" nhẹ!

- Hệ thống theo dõi trong nhà: Rất nhiều cảm biến GPS hiện nay tín hiệu không đủ mạnh để định vị trong nhà, đặc biệt là ở tầng thấp. Bluetooth Mesh có thể khắc phục điểm yếu đó và kết hợp tốt với GPS, chúng ta có thể dựa vào mức độ mạnh yếu của tín hiệu và áp dụng thêm các thuật toán để xác định vị trí cũng như độ cao cụ thể của người hay vật trong nhà với độ chính xác cao.

# A Một số Model chuẩn

## A.1 Foundation models

Foundation models have been defined in the core specification. Two of them are mandatory for all mesh nodes.

- Configuration Server (mandatory)
- Configuration Client
- Health Server (mandatory)
- Health Client

## A.2 Generic models

- Generic OnOff Server, used to represent devices that do not fit any of the model descriptions defined but support the generic properties of On/Off
- Generic Level Server, keeping the state of an element in a 16-bit signed integer
- Generic Default Transition Time Server, used to represent a default transition time for a variety of devices
- Generic Power OnOff Server & Generic Power OnOff Setup Server, used to represent devices that do not fit any of the model descriptions but support the generic properties of On/Off

- Generic Power Level Server & Generic Power Level Setup Server, including a Generic Power Actual state, a Generic Power Last state, a Generic Power Default state and a Generic Power Range state
- Generic Battery Server, representing a set of four values representing the state of a battery
- Generic Location Server & Generic Location Setup Server, representing location information of an element, either global (Lat/Lon) or local
- Generic User/Admin/Manufacturer/Client Property Server, representing any value to be stored by an element
- Generic OnOff Client & Generic Level Client
- Generic Default Transition Time Client
- Generic Power OnOff Client & Generic Power Level Client
- Generic Battery Client
- Generic Location Client
- Generic Property Client

### **A.3 Sensors**

- Sensor Server & Sensor Setup Server, representing a sensor device. Sensor device may be configured to return a measured value periodically or on request; measurement period (cadence) may be configured to be fixed or to change, so that more important value range is being reported faster.
- Sensor Client

### **A.4 Time and scenes**

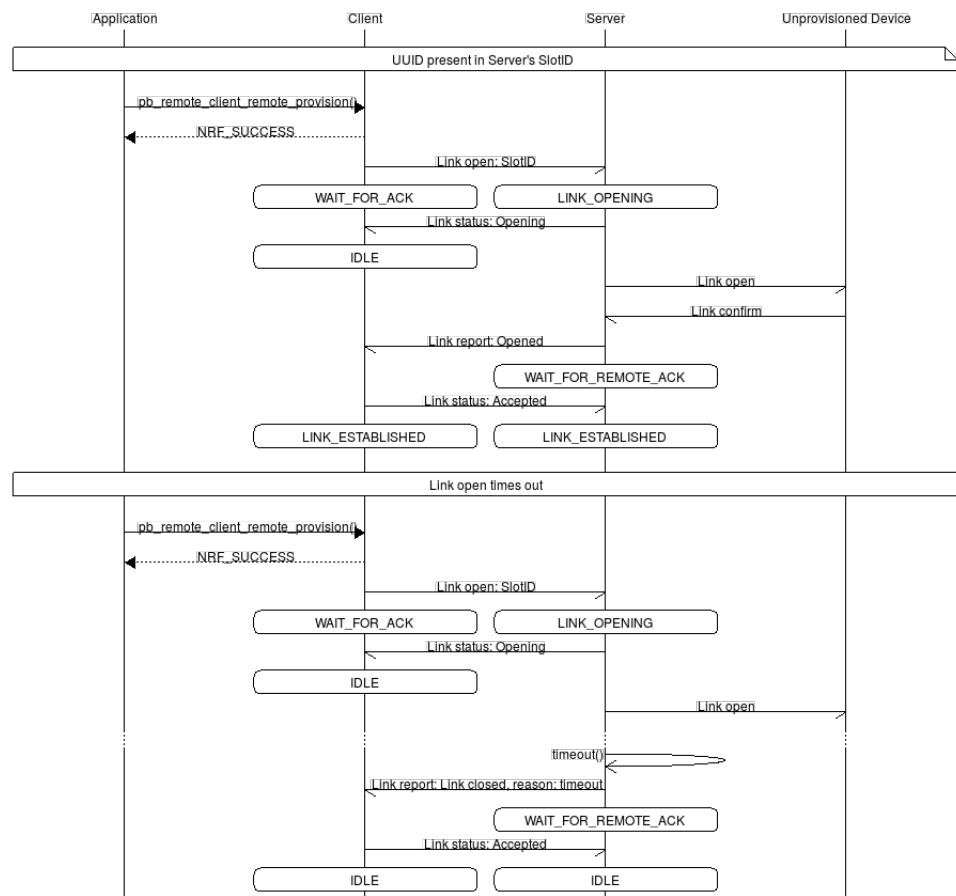
- Time Server & Time Setup Server, allowing for time synchronization in mesh network

- Scene Server & Scene Setup Server, allowing for up to 65535 scenes to be configured and recalled when needed.
- Scheduler Server & Scheduler Setup Server
- Time Client, Scene Client & Scheduler Client

## A.5 Lighting

- Light Lightness Server & Light Lightness Setup Server, representing a dimmable light source
- Light CTL Server, Light CTL Temperature Server & Light CTL Setup Server, representing a CCT or "tunable white" light source
- Light HSL Server, Light HSL Hue Server, Light HSL Saturation Server & Light HSL Setup Server, representing a light source based on Hue, Saturation, Lightness color representation
- Light xyL Server & Light xyL Setup Server, representing a light source based on modified CIE xyY color space.
- Light LC (Lightness Control) Server & Light LC Setup Server, representing a light control device, able to control Light Lightness model using an occupancy sensor and ambient light sensor. It may be used for light control scenarios like Auto-On, Auto-Off and/or Daylight Harvesting.
- Light Lightness Client, Light CTL Client, Light HSL Client, Light xyL Client & Light LC Client

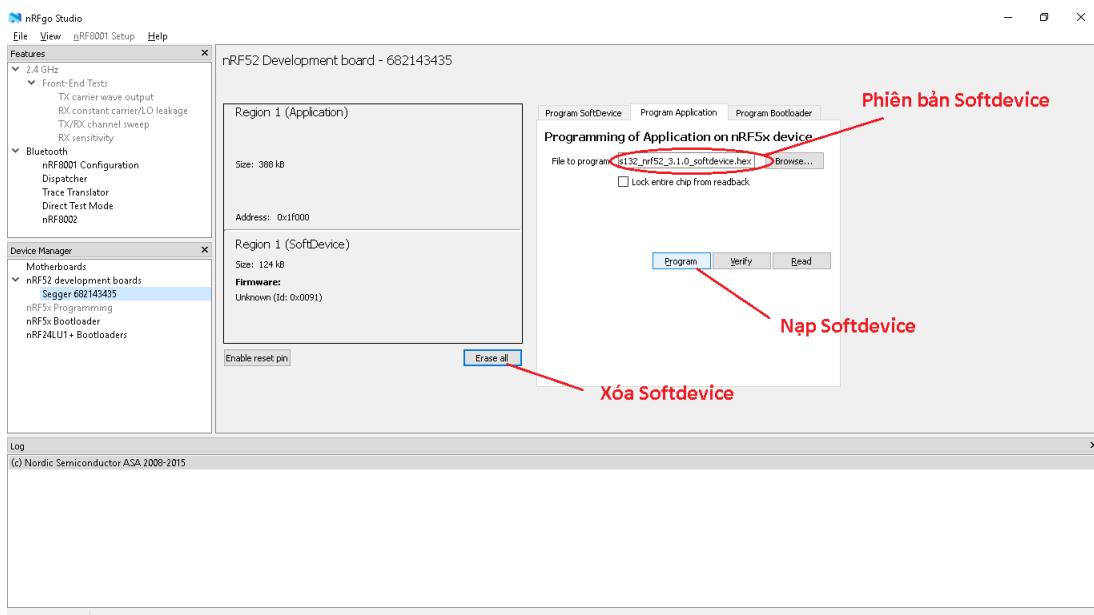
## B Flowchart quá trình remote provisioning



Hình B.1: Quá trình remote provisioning

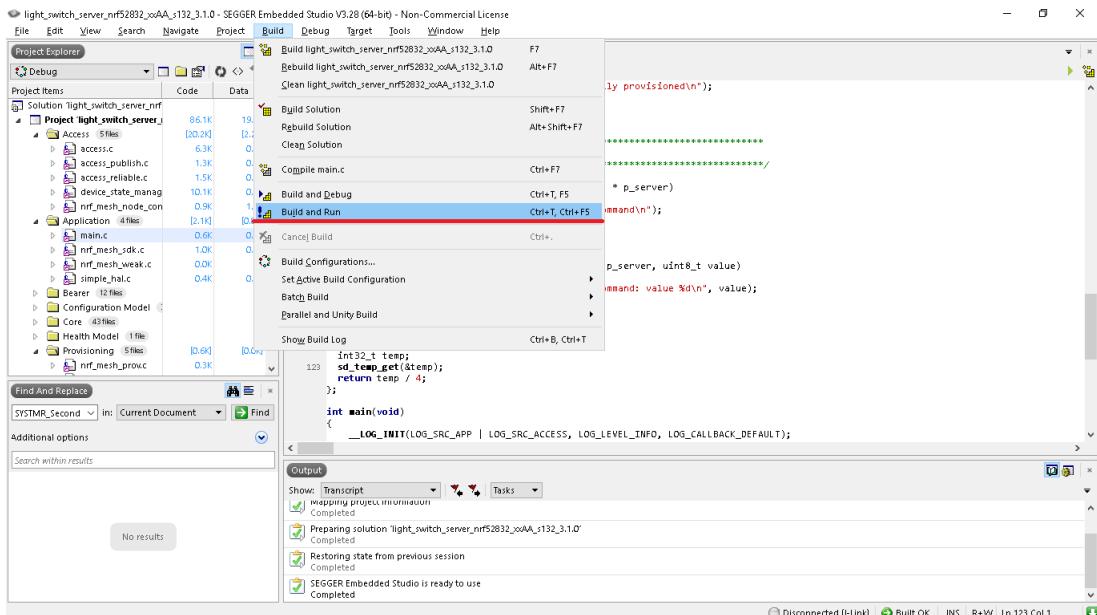
# C Hướng dẫn thực thi ứng dụng

- Sau khi kết nối Kit PCA10040 với máy tính, sử dụng phần mềm nRFgo Studio để nạp Softdevice cho Kit.



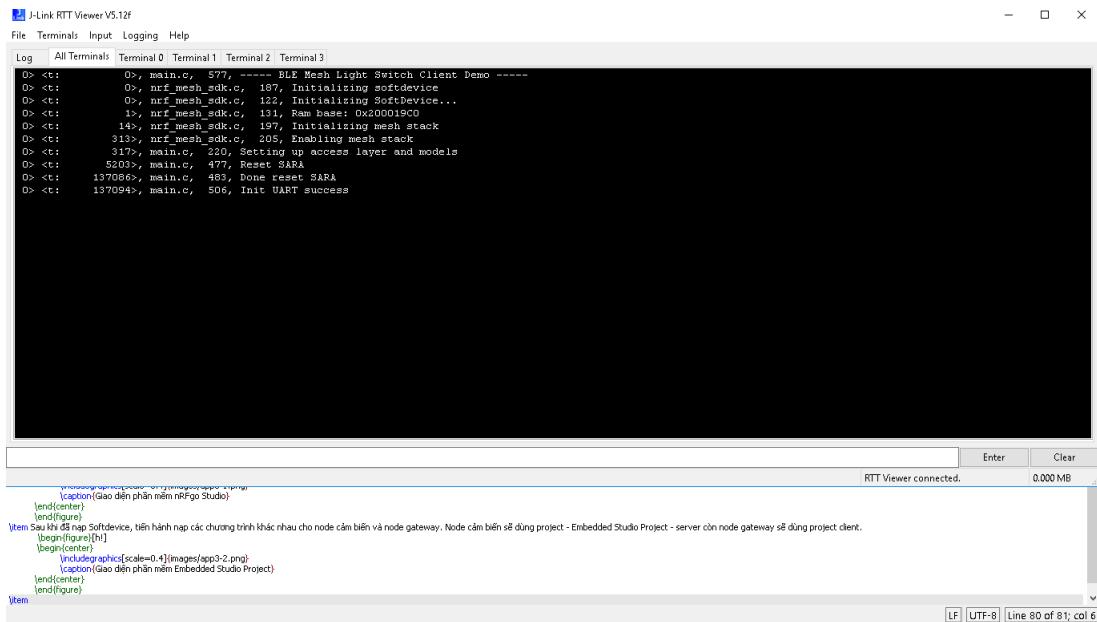
Hình C.1: Giao diện phần mềm nRFgo Studio

- Sau khi đã nạp Softdevice, tiến hành nạp các chương trình khác nhau cho node cảm biến và node gateway. Node cảm biến sẽ dùng project - Embedded Studio Project - server còn node gateway sẽ dùng project client.



Hình C.2: Giao diện phần mềm Embedded Studio Project

3. Để theo dõi quá trình thực thi của chương trình, sử dụng J-link RTT Viewer để xem quan sát quá trình.



Hình C.3: Giao diện phần mềm J-link RTT Viewer

# Tài liệu tham khảo

- [1] Cách mạng Công nghiệp 4.0 là gì?

<https://news.zing.vn/cach-mang-cong-nghiep-40-la-gi-post750267.html>

- [2] Implementing a mesh network? Here's why you should use Bluetooth

<http://www.embedded-computing.com/iot/implementing-a-mesh-network-here-s-why>

- [3] Bluetooth SIG Announces Mesh Networking Capability

<https://www.bluetooth.com/news/pressreleases/2017/07/bluetooth-sig-announces-mesh-networking-capability>

- [4] Basic Bluetooth Mesh concepts

[http://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.meshsdk.v1.0.1%2Fmd\\_doc\\_introduction\\_basic\\_concepts.html&cp=4\\_1\\_0\\_0\\_0](http://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.meshsdk.v1.0.1%2Fmd_doc_introduction_basic_concepts.html&cp=4_1_0_0_0)

- [5] Elliptic-curve Diffie–Hellman

[https://en.wikipedia.org/wiki/Elliptic-curve\\_Diffie%E2%80%93Hellman](https://en.wikipedia.org/wiki/Elliptic-curve_Diffie%E2%80%93Hellman)

- [6] Mesh Networking Specifications

<https://www.bluetooth.com/specifications/mesh-specifications>

- [7] nRF5 SDK

<https://www.nordicsemi.com/eng/Products/Bluetooth-low-energy/nRF5-SDK>

- [8] Bluetooth Mesh Glossary of Terms

<https://www.bluetooth.com/bluetooth-technology/topology-options/le-mesh/mesh-glossary>

- [9] Bluetooth Mesh Networking: Friendship  
<http://blog.bluetooth.com/bluetooth-mesh-networking-series-friendship>
- [10] A Closer Look at the New Bluetooth Mesh  
<https://www.betasolutions.co.nz/Blog/12/A-Closer-Look-at-the-New-Bluetooth-Mesh>
- [11] Bluetooth Mesh FAQ  
<https://www.bluetooth.com/bluetooth-technology/topology-options/le-mesh/mesh-faq>
- [12] The Fundamental Concepts of Bluetooth Mesh Networking Part 1  
<https://blog.bluetooth.com/the-fundamental-concepts-of-bluetooth-mesh-networking-part-1>
- [13] Bluetooth Mesh Security Overview  
<https://blog.bluetooth.com/bluetooth-mesh-security-overview>
- [14] How to add uart function to light control client demo at nrf5\_SDK\_for\_Mesh\_v0.9.2-Alpha  
[https://devzone.nordicsemi.com/f/nordic-q-a/25877/how-to-add-uart-function-to-light-control-client-demo-at-nrf5\\_sdk\\_for\\_mesh\\_v0-9-2-alpha](https://devzone.nordicsemi.com/f/nordic-q-a/25877/how-to-add-uart-function-to-light-control-client-demo-at-nrf5_sdk_for_mesh_v0-9-2-alpha)
- [15] S132 SoftDevice  
<https://www.nordicsemi.com/eng/Products/S132-SoftDevice>