

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**LUẬN VĂN TỐT NGHIỆP
TRUYỀN THÔNG VÀ MẠNG MÁY TÍNH**

Đề tài

**HỆ THỐNG QUẢNG CÁO SẢN PHẨM DỰA TRÊN
CÔNG NGHỆ BLUETOOTH – MOBILE APP**

**Sinh viên thực hiện :
Nguyễn Thanh Huy
Khóa : K43**

Cần Thơ, 12/2021

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**LUẬN VĂN TỐT NGHIỆP
NGÀNH TRUYỀN THÔNG VÀ MẠNG MÁY TÍNH**

Đề tài

**HỆ THỐNG QUẢNG CÁO SẢN PHẨM DỰA TRÊN
CÔNG NGHỆ BLUETOOTH – MOBILE APP**

**Giáo viên hướng dẫn:
TS. Ngô Bá Hùng**

**Sinh viên thực hiện:
Nguyễn Thanh Huy
Khóa : K43**

Cần Thơ, 12/2021

**XÁC NHẬN CHỈNH SỬA LUẬN VĂN
THEO YÊU CẦU CỦA HỘI ĐỒNG**

Tên luận văn: Hệ thống quảng cáo sản phẩm dựa trên công nghệ Bluetooth – Mobile App.

Họ tên sinh viên: Nguyễn Thanh Huy

MASV: B1709280

Mã lớp: DI17Y9A1

Đã báo cáo tại hội đồng ngành: Truyền thông & Mạng máy tính

Ngày báo cáo: 27/12/2021

Luận văn đã được chỉnh sửa theo góp ý của Hội đồng.

Cần Thơ, ngày tháng năm 20...

Giáo viên hướng dẫn

(Ký và ghi họ tên)

NHẬN XÉT CỦA GIẢNG VIÊN

[illegible]

LỜI CẢM ƠN

Em xin chân thành cảm ơn thầy Ngô Bá Hùng – Giảng viên hướng dẫn bộ môn luận văn tốt nghiệp đã giành thời gian quý báu để hướng dẫn cho em trong môn học này. Cảm ơn các thầy cô Khoa Công Nghệ Thông Tin & Truyền Thông đã tạo điều kiện và cung cấp những kiến thức quan trọng để em tích lũy và thực hiện đề tài một cách tốt nhất.

Thông qua quá trình thực hiện luận văn tốt nghiệp, em phần nào củng cố và tích lũy được những kiến thức về công nghệ truyền thông không dây tầm gần – Bluetooth cũng như từng bước triển khai hạ tầng kết nối Server-Client. Mặc dù, đã cố gắng hết sức trong tất cả quá trình từ học tập cho đến thực hiện đề tài này. Nhưng em đã không thể tránh khỏi những sai sót nhất định. Em rất mong được sự thông cảm, bỏ qua và góp ý tận tình từ thầy cô và các bạn.

Em xin chân thành cảm ơn !

Cần Thơ, ngày 24 tháng 12 năm 2021

Người viết

Nguyễn Thanh Huy

MỤC LỤC

PHẦN I. GIỚI THIỆU	1
1. Đặt vấn đề	1
2. Tóm tắt lịch sử giải quyết vấn đề	2
3. Mục tiêu đề tài	2
4. Đối tượng và phạm vi nghiên cứu	3
5. Nội dung nghiên cứu.....	4
6. Những đóng góp chính của đề tài	4
PHẦN II. NỘI DUNG	5
Chương 1. Mô tả bài toán	5
1.1. Đặc tả yêu cầu	5
1.1.1. Tổng quan mô hình	6
1.1.1.1. User story người bán hàng.....	6
1.1.1.2. User story người khách hàng (Consumer).....	6
1.1.1.3. User story hệ thống trung gian (BT Advertiser).....	6
1.2. Công nghệ Bluetooth	7
1.2.1. Tổng quan.....	7
1.2.2. Đặc điểm.....	7
1.3. Bluetooth Low Energy(BLE).....	8
1.3.1. Tổng quan.....	8
1.3.2. Vai trò.....	8
1.3.3. BLE Protocol Stack.....	9
1.3.3.1. Tổng quan	9
1.3.3.2. Các tầng giao thức chính	9
1.3.3.3. Physical Layer(PHY).....	10
a. Tổng quan	10
b. Dải tần số	10
1.3.3.4. Link Layer(LL).....	10
a. Tổng quan	10
b. Kênh truyền	10
1.3.3.5. Host Controller Interface(HCI)	12
1.3.3.6. Logical Link Control & Adaptation Protocol(L2CAP).....	12
1.3.3.7. Security Manager(SMP)	13
1.3.3.8. Attribute Protocol(ATT).....	13

1.3.3.9. Generic Attribute Protocol(GATT)	13
a. Tổng quan	13
b. Universally Unique Identifier(UUID)	13
1.3.3.10. Generic Access Profile(GAP).....	14
a. Tổng quan	14
b. Vai trò	14
1.3.4. BLE Advertisements	16
1.3.4.1. Tổng quan	16
1.3.4.2. Kiến thức cơ bản về quảng cáo BLE	16
1.3.4.3. Physical BLE Advertisements	16
1.3.4.4. BLE Advertisements Interval	17
1.3.4.5. Các loại quảng cáo khác nhau	18
1.3.4.6. Advertisements Packet Structure	19
a. Advertising Packet – Preamble.....	19
b. Advertising Packet – Access Address	19
c. Advertising Packet – Protocol Data Unit	19
d. Advertising Packet – CRC	20
e. Advertising Packet – Link Layer	20
f. Protocol Data Unit(PDU).....	20
1.4. Raspberry Pi 4B	23
1.4.1. Tổng quan.....	23
1.4.3. Hệ điều hành tương thích	24
1.5. Lập trình ứng dụng trên Android Studio.....	24
1.5.1. Thư viện hỗ trợ Bluetooth trên Android Studio	24
1.5.1.1. Tổng quan	24
1.5.1.2. Class Interface	24
1.5.2. Tìm hiểu về cơ sở dữ liệu SQLite trong Android Studio.	25
1.5.2.1. Tổng quan	25
1.5.2.2. Class Interface	26
1.6. Thư viện BlueZ	26
1.6.1. Tổng quan.....	26
1.6.2. Tính năng, đặc điểm	26
1.6.3. API BlueZ Advertisement	26
1.7. RabbitMQ.....	27

1.7.1. Tổng quan.....	27
1.7.2. Thư viện Pika	28
1.7.2.1 IO and Event Looping	28
1.7.2.2. Continuation-Passing Style	28
1.7.2.3. TCP Backpressure	28
1.7.2.4. Class và Module của Pika.....	28
Chương 2. Thiết kế và cài đặt giải pháp	29
2.1. Kiến trúc hệ thống:.....	29
2.1.1. Mô hình tổng thể các thành phần trong hệ thống.....	29
2.1.2. Mô hình Use case Consumer.....	29
2.1.3. Mô hình Use case Advertiser	30
2.1.4. Cơ sở dữ liệu quan hệ SQLite	30
2.1.5. Mô hình Sequence nhận catalogue từ RabbitMQ	31
2.1.6. Mô hình Sequence phân tách dữ liệu quảng cáo và truyền vào ngăn xếp BLE	32
2.1.7. Mô hình Sequence phát gói tin quảng cáo cho Consumer	33
2.1.8. Mô hình Sequence lọc gói tin quảng cáo và ghi dữ liệu vào SQLite.....	34
2.1.9. Mô hình Sequence xem danh sách catalogue quảng cáo	35
2.1.10. Mô hình Sequence xóa catalogue quảng cáo	35
2.1.11. Data Flow Diagram nhận message từ RabbitMQ	36
2.1.12. Data Flow Diagram Advertiser	36
2.1.13. Data Flow Diagram Consumer.....	37
2.1.14. Activity Diagram Advertiser	38
2.1.15. Activity Diagram Consumer	39
2.2. Thiết kế giải thuật xử lý	40
2.2.1. Phương pháp sử dụng các byte làm cờ và các byte đệm (Flag byte with byte stuffing)	40
2.2.2. Trên Raspberry Pi 4B (Advertiser)	40
2.2.3. Giải thuật Maximum Transmission Unit(MTU)	41
2.2.4. Cấu trúc gói tin phát quảng cáo.....	43
2.2.5. Trên Android Studio lập trình giao diện mức thấp	48
2.2.5.1. Quét và lọc gói tin quảng cáo Bluetooth	48
2.3. Cài đặt giải pháp	54
2.3.1. Cài đặt và quản trị hệ điều hành Raspbian trên Raspberry Pi	54
2.3.2. Thiết lập môi trường phát triển ứng dụng và cài đặt thư viện BlueZ.....	55

2.3.3. Cấp quyền cho ứng dụng Android	55
2.3.4. Khởi chạy ứng dụng trên Smartphone	56
2.3.5. Kiểm tra kết quả	58
2.3.5.1. Dữ liệu Catalogue Manager truyền qua RabbitMQ	58
2.3.5.2. Chuyển dữ liệu phát gói quảng cáo Bluetooth	59
2.3.5.2. Database trên SQLite	60
Chương 3. Kiểm thử và đánh giá	60
3.1. Kiểm thử	60
3.2. Đánh giá	63
PHẦN III. KẾT LUẬN	63
1. Kết quả đạt được	63
2. Hạn chế	63
3. Hướng phát triển	63
TÀI LIỆU THAM KHẢO	64

DANH MỤC HÌNH

Hình 1. Ngăn xếp giao thức năng lượng thấp Bluetooth.....	9
Hình 2. Dải băng tần Physical layer	10
Hình 3. Nhảy tần thích ứng của Link Layer(LL)	11
Hình 4. Nhảy tần thích ứng trách đưng độ với Wifi.....	12
Hình 6. Mô hình không kết nối Generic Access Profile(GAP).....	15
Hình 7. Mô hình định hướng kết nối Generic Access Profile(GAP)	15
Hình 8. Dải tần số các kênh Bluetooth và Wifi.....	17
Hình 9. BLE Advertisements Interval	18
Hình 10. Raspberry Pi 4B.....	23
Hình 11. Hệ điều hành Raspbian.....	24
Hình 12. Cơ sở dữ liệu SQLite.....	25
Hình 13. Mô hình RabbitMQ	27
Hình 14. Mô hình hệ thống quảng cáo sản phẩm dựa trên công nghệ Bluetooth	29
Hình 15. Mô hình Use case Consumer.....	29
Hình 16. Mô hình Use case Advertiser	30
Hình 17. Mô hình Sequence nhận catalogue từ RabbitMQ	31
Hình 18. Mô hình Sequence phân tách dữ liệu quảng cáo	32
Hình 19. Mô hình Sequence phát gói tin quảng cáo cho Consumer	33
Hình 20. Mô hình Sequence lọc packet quảng cáo	34
Hình 21. Mô hình Sequence xem danh sách catalogue quảng cáo	35
Hình 22. Mô hình Sequence xóa catalogue quảng cáo	35
Hình 23. Data Flow Diagram nhận message từ RabbitMQ	36
Hình 24. Data Flow Diagram Advertiser	36
Hình 25. Data Flow Diagram Consumer.....	37
Hình 26. Activity Diagram Advertiser.....	38
Hình 27. Diagram Activity Consomer	39
Hình 28. Cấu trúc quảng cáo được phát với packet Header đầu tiên	44
Hình 29. Cấu trúc quảng cáo được phát với packet Data đầu tiên	45
Hình 30. Cấu trúc quảng cáo được phát với packet Header cuối cùng	46
Hình 31. Cấu trúc quảng cáo được phát với packet Data cuối cùng	47
Hình 32. Yêu cầu cho phép ứng dụng bật Bluetooth	56
Hình 33. Xem danh sách các sản phẩm quảng cáo	57
Hình 34. Smartphone nhận đầy đủ các gói quảng cáo	60

Hình 35. Smartphone không nhận đầy đủ các gói quảng cáo	60
---	----

DANH MỤC BẢNG

Bảng 1. Vai trò Generic Access Profile(GAP).....	14
Bảng 2. So sánh tầng vật lý giữa BLE & Bluetooth Classic	16
Bảng 3. Những loại quảng cáo PDU	19
Bảng 4. PDU Type quảng bá dữ liệu truyền phát.	21
Bảng 5. Cấu hình cơ sở dữ liệu quan hệ SQLite	30
Bảng 6. Phân đoạn gói đầu tiên – gói cuối cùng	43
Bảng 7. Quảng cáo PDU payload data Header gói đầu tiên.	45
Bảng 8. Quảng cáo PDU payload data Data gói đầu tiên.	46
Bảng 9. Quảng cáo PDU payload data Header gói cuối cùng.....	47
Bảng 10. Quảng cáo PDU payload data Data gói cuối cùng.....	48
Bảng 11. So sánh thời gian chu kỳ tổng hợp gói tin catalogue	62

Tóm lược: Trong bài báo cáo này, em đề xuất xây dựng *Hệ thống quảng cáo sản phẩm dựa trên công nghệ Bluetooth – Mobile App* được triển khai trên Raspberry Pi(Advertiser) và mobile app quét các gói tin quảng cáo từ phía Advertiser truyền. Giúp thông tin đến khách hàng những sản phẩm, dịch vụ mới hoặc là các chương trình khuyến mãi từ các nhãn hàng và các voucher giảm giá cũng như có thể lưu lại thông tin sản phẩm mà khách hàng quang tâm. Khách hàng chỉ mở app Bluetooth và chờ quảng cáo xuất hiện lên màn hình, khi khách hàng đi quang khu vực có vùng phủ sóng Bluetooth của Advertiser thì, Smartphone tự động quét và nhận gói tin quảng cáo về và tổng hợp hiển thị lên màn hình mà không cần thao tác nhấn vào chấp nhận kết nối. Và điều đặc biệt là khách hàng không cần điện thoại kết nối internet mà vẫn nhận được sản phẩm quảng cáo.

Abstract: In this report, I propose to build an Adver system based on Bluetooth technology- Mobile App deployed on Raspberry Pi (Advertiser) and mobile app that scans advertising packets from Advertiser transmitting. Help inform customers of new products, services or promotions from brands and discount vouchers as well as save product information that customers are interested in. Customers only open the Bluetooth app and wait for the ad to appear on the screen, when the customer walks around the area with Bluetooth coverage of the Ad-vertiser, the Smartphone automatically scans and receives the advertisement packet and displays it. to the screen without having to click accept the connection. And the special thing is that customers do not need a phone to connect to the internet and still receive advertising products.

PHẦN I. GIỚI THIỆU

1. Đặt vấn đề

Với sự bùng nổ của cuộc cách mạng công nghiệp 4.0, đi cùng với sự phát triển nhanh chóng tốc độ của Internet hiện nay, chúng ta đã thực hiện được nhiều công việc với tốc độ nhanh hơn và chi phí thấp hơn nhiều so với cách thức truyền thống. Chính vì điều này, đã thúc đẩy sự khai sinh ra mạng lưới thiết bị kết nối không dây ra đời làm thay đổi đáng kể bộ mặt thế giới. Kỹ thuật không dây phục vụ rất nhiều nhu cầu khác nhau của con người, từ làm việc, học tập đến giải trí như chơi game, xem phim, nghe nhạc... Kỹ thuật không dây đã đưa ra nhiều chuẩn với đặc điểm kỹ thuật khác nhau như WLAN với chuẩn 802.11, Lora, Zigbee, Bluetooth... Mỗi chuẩn kỹ thuật đều có những ưu, nhược điểm khác nhau, và Bluetooth đang dần nổi lên là kỹ thuật không dây tầm ngắn và hầu hết các chiếc Smartphone và các máy tính nhúng hiện nay đều trang bị module Bluetooth. Cùng với sự phát triển của thời đại thì, xu hướng tiêu dùng của con người cũng hiện đại theo, kéo theo ngành Marketing hằng ngày, hằng giờ phải đổi mới sáng tạo để bắt kịp với tốc độ phát triển của xã hội. Để có thể giới thiệu các sản phẩm, dịch vụ tới người dùng cuối thì các công ty, doanh nghiệp, hộ kinh doanh, các cửa hàng áp dụng là Marketing sản phẩm, dịch vụ đến người tiêu dùng. Một số phương pháp Marketing hiện nay được áp dụng là quảng cáo truyền thông trên tivi hoặc các nền tảng xã hội, hoặc dùng phương pháp thủ công là đi phát tờ rơi và dán poster trên tường, cột điện việc đó gây ảnh hưởng rất lớn đến mỹ quan đô thị, cùng với sự gia tăng lây nhiễm đại dịch toàn cầu(Covid-19) có thể lây bệnh khi tiếp xúc trực tiếp hoặc gián tiếp với người nhiễm bệnh, nên những người tiêu dùng ngại tiếp xúc và giao tiếp để tìm hiểu những sản phẩm dịch vụ mà họ được tiếp cận. Cho nên, cần phải tạo ra hệ thống quảng cáo sản phẩm, dịch vụ cho các công ty, cửa hàng... dùng để tiếp cận nhanh nhất những sản phẩm, dịch vụ mới nhất, cùng với đó là các chương trình khuyến mãi hoặc mã voucher đến tay khách hàng tiềm năng, đồng thời tạo một môi trường quảng cáo đồng bộ từ người sản xuất đến người tiêu dùng đầu cuối, tạo ra một tâm lý thoải mái cho khách hàng tiếp cận quảng cáo online không cần kết nối đến bất kỳ thiết bị nào dựa trên công nghệ Bluetooth.

Xuất phát từ lý do trên em thực hiện đề tài “Hệ thống quảng cáo sản phẩm dựa trên công nghệ Bluetooth – Mobile App”.

2. Tóm tắt lịch sử giải quyết vấn đề

Trong nước, các đề tài hoặc ứng dụng làm về Bluetooth Marketing rất ít. Một trong những công ty đi đầu về Bluetooth Marketing là Centech và FPT Promo với chiến dịch “Pepsi Bluetooth Zone” năm 2008. Hiện nay các công ty này đã ngừng cung cấp dịch vụ Bluetooth Marketing. Còn về việc quảng bá dữ liệu phục vụ cho cá nhân thì các module nhiệt độ, độ ẩm truyền giá trị qua Bluetooth thì thuộc về sản phẩm của nước ngoài.

Ngoài nước, các đề tài và ứng dụng làm về Bluetooth Marketing rất nhiều. Tiêu biểu là các sản phẩm iBeacon của Apple, những thiết bị IoT trong nhà của Beacon...

Những vấn đề còn chưa được giải quyết hiện nay là xây dựng nền tảng phục vụ cho công ty, cửa hàng... cần Marketing sản phẩm, dịch vụ nhanh nhất tới khách hàng mà không bị giới hạn dữ liệu nhập vào, có thể thiết đặt thời gian địa điểm quảng cáo, và khách hàng đầu cuối làm sao họ nhận quảng cáo nhanh nhất và không cần thao tác nhấn hoặc chấp nhận kết nối từ thiết bị quảng cáo Bluetooth. Những vấn đề đó hầu như các đề tài hoặc mô hình quảng cáo chưa đề cập tới vì do 1 phần các gói quảng cáo Bluetooth bị giới hạn (0 – 31 bytes) hoặc nếu muốn truyền dữ liệu nhiều hơn như hình ảnh, video ngắn thì cần khách hàng phải thao tác tay nhấn chấp nhận kết nối Bluetooth với thiết bị phát quảng cáo Bluetooth.

3. Mục tiêu đề tài

Chúng ta xây dựng hệ thống Advertiser lấy catalogue từ website truyền về, và phát các gói tin quảng cáo catalogue đó cho những thiết bị Smartphone có trang bị công nghệ Bluetooth. Khi người dùng bước vào vùng phủ sóng của Advertiser thì người dùng sẽ nhận các gói tin quảng cáo truyền liên tục, về phía ứng dụng Consumer trong Smartphone có trách nhiệm thu các gói tin quảng cáo catalogue lưu lại trong cơ sở dữ liệu và xuất hiện thông tin quảng cáo ra màn hình khi tổng hợp các gói tin catalogue đầy đủ.

Cách vận hành và quản lý đơn giản từ phía người quản lý sản phẩm quảng cáo và người khách hàng. Giúp Marketing sản phẩm, dịch vụ nhanh nhất tới khách hàng mà không bị giới hạn dữ liệu nhập vào, có thể thiết đặt thời gian địa điểm quảng cáo, và khách hàng đầu cuối làm sao họ nhận quảng cáo nhanh nhất và không cần thao tác nhấn hoặc chấp nhận kết nối từ thiết bị quảng cáo Bluetooth. Cung cấp khả năng lưu trữ cho khách hàng để có thể xem lại những thông tin sản phẩm cũng như có thể lưu lại những mã voucher từ cửa hàng.

4. Đối tượng và phạm vi nghiên cứu

Với nhu cầu của các công ty, doanh nghiệp, hộ kinh doanh cá thể và cửa hàng về việc quảng cáo giới thiệu các sản phẩm, dịch vụ các chương trình khuyến mãi, tặng voucher đến tay khách hàng. Đảm bảo việc giãn cách, hạn chế tiếp xúc để giảm lây nhiễm do Covid-19 khi khách hàng được tiếp cận đến sản phẩm quảng cáo.

Tìm hiểu công nghệ Bluetooth bao gồm:

Khái niệm, đặc điểm về Bluetooth

Tìm hiểu về Bluetooth Low Energy(BLE):

Khái niệm, vai trò.

Cấu hình truy cập chung(GAP).

Các tầng giao thức.

Cấu trúc gói tin.

Tìm hiểu về Raspberry Pi 4B(Advertiser):

Cấu hình phần cứng, phần mềm.

Cài đặt thư viện BlueZ hỗ trợ module Bluetooth.

Cài đặt thư viện Pika hỗ trợ RabbitMQ.

Tìm hiểu lập trình ứng dụng trên Android Studio:

Thư viện hỗ trợ Bluetooth trên Android Studio.

Tìm hiểu về cơ sở dữ liệu SQLite trong Android Studio.

Tìm hiểu và lập trình Bluetooth bằng thư viện BlueZ:

Thư viện hỗ trợ trên BlueZ.

Lập trình sử dụng API BlueZ phát quảng cáo Bluetooth.

Tìm hiểu và lập trình RabbitMQ bằng thư viện Pika:

Thư viện hỗ trợ trên RabbitMQ.

Lập trình nhận message trên hàng đợi RabbitMQ về Advertiser.

Lập trình gửi message từ Advertiser lên hàng đợi RabbitMQ.

5. Nội dung nghiên cứu

Nội dung nghiên cứu của nhóm	
Huy	Tín
<ul style="list-style-type: none"> + Tìm hiểu và cài đặt biến môi trường trên Raspberry Pi. + Tìm hiểu về các tầng giao thức và cấu trúc gói tin quảng bá Bluetooth(GAP). + Lập trình trên thư viện BlueZ. + Tìm hiểu và lập trình Bluetooth trên Android Studio. + Tìm hiểu và truy vấn CSDL SQLite trên Android Studio. + Tìm hiểu và lập trình giao thức truyền và nhận message trên hàng chờ RabbitMQ. 	<ul style="list-style-type: none"> + Công nghệ truyền dữ liệu từ Server(RabbitMQ) đến cục phát Bluetooth. + Tìm hiểu và phân tích các chức năng, phân tích yêu cầu hệ thống đối với người dùng là cửa hàng cũng như là bên người dùng là khách hàng của cửa hàng. + Tìm hiểu thiết kế giao diện đẹp và tiện lợi cho khách hàng, tối ưu các bước không cần thiết.

6. Những đóng góp chính của đề tài

- + Giúp tạo thêm một xu hướng mới trong lĩnh vực Bluetooth Marketing.
- + Góp phần giải quyết vấn đề giới hạn dữ liệu phát quảng cáo Bluetooth.
- + Giúp Marketing sản phẩm, dịch vụ nhanh nhất tới khách hàng mà không cần thao tác nhấn hoặc chấp nhận kết nối từ thiết bị quảng cáo Bluetooth.
- + Cung cấp khả năng lưu trữ cho khách hàng để có thể xem lại những thông tin sản phẩm cũng như có thể lưu lại những mã voucher từ cửa hàng.
- + Giúp đồng bộ hóa thiết đặt thời gian, địa điểm quảng cáo cùng lúc hoặc đặt lịch quảng cáo chương trình khuyến mãi hoặc phát mã voucher.

PHẦN II. NỘI DUNG

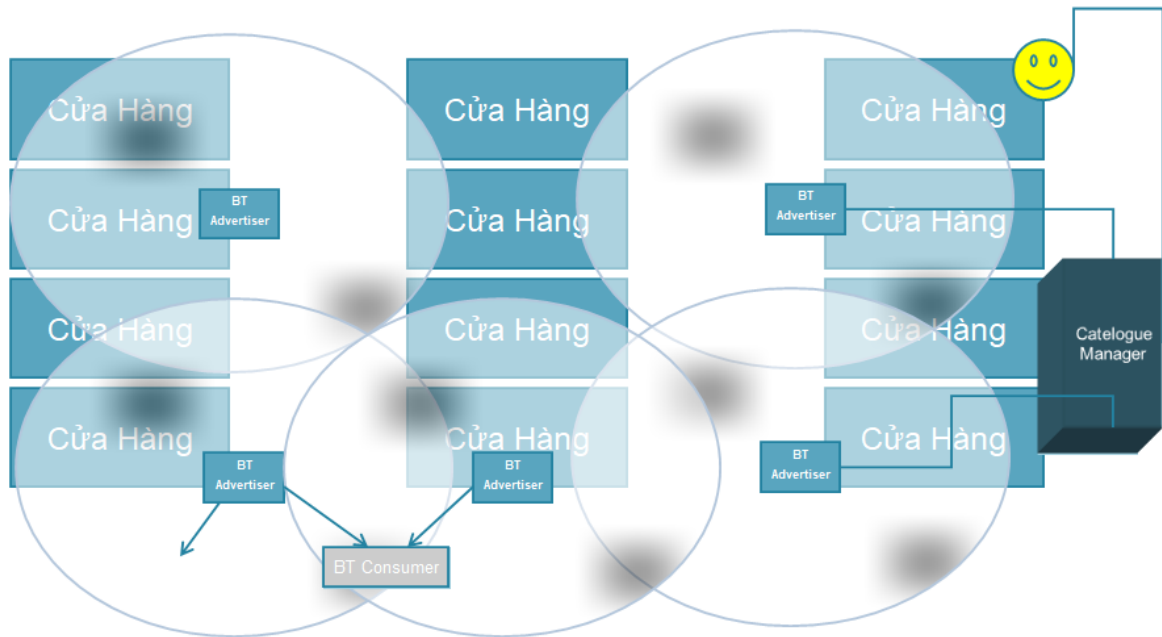
Chương 1. Mô tả bài toán

1.1. Đặc tả yêu cầu

Câu hỏi được đặt ra, chúng ta làm thế nào để phát quảng cáo dựa trên công nghệ Bluetooth. Đương nhiên là chúng ta có thể làm được, một số giải pháp phần mềm hoặc các các thiết bị IoT đo nhiệt độ, độ ẩm Bluetooth wireless đều có thể truyền dữ liệu bằng cách phát quảng bá các gói tin nhưng bị giới hạn kích thước dữ liệu hoặc muốn truyền dữ liệu lớn hơn(image, video, music...) thì cần phải bắt buộc người dùng phải chất nhận kết nối Bluetooth đến thiết bị phát, điều đó rất bất tiện cũng như về mặt hạn chế của giải pháp. Nhưng khi chúng ta áp dụng Bluetooth Low Energy(BLE) quét và quảng cáo(GAP) thì có sự nâng cấp hơn, với *Hệ thống quảng cáo sản phẩm dựa trên công nghệ Bluetooth* thì có thể phát dữ liệu quảng cáo không giới hạn, bằng cách chia nhỏ dữ liệu quảng cáo thành nhiều gói tin và sau đó Smartphone của người dùng quét Bluetooth nhận các gói quảng cáo này về, và tổng hợp lại hiển thị lên màn hình điện thoại các sản phẩm, dịch vụ quảng cáo.

- Phần mềm nhận và truyền message catalogue trên hàng chờ RabbitMQ:
 - + Thiết lập kênh truyền giữa Catalogue Manager(Web Server) và Advertiser(Raspberry Pi) để nhận catalogue về và lưu lại.
 - + Trả tính hiệu về cho Catalogue Manager xem coi Advertiser còn hoạt động không, và kênh truyền được thiết lập dựa trên ID catalogue, nếu không có kết quả trả về thì Advertiser đã ngừng hoạt động hoặc mất tính hiệu Internet.
- Phần mềm phát gói tin quảng cáo:
 - + Đọc dữ liệu lên sau khi file catalogue được lưu về, sau đó phân tích cú pháp chia nhỏ dữ liệu để phù hợp với kích thước gói đi phát quảng cáo.
 - Ứng dụng trên điện thoại thì có bốn chức năng chính:
 - + Lưu catalogue sản phẩm quảng cáo sau khi tổng hợp các gói tin quảng cáo.
 - + Xem các catalogue sản phẩm mới, sản phẩm khuyến mãi các mã voucher được tổng hợp.
 - + Trong trường hợp catalogue có ba sản phẩm nhưng chỉ tổng hợp được hai sản phẩm, thì hiển thị ra ngay không cần chờ thêm thời gian để tổng hợp thêm sản phẩm thứ ba.
 - + Chức năng xóa các catalogue sản phẩm quảng cáo đi khi không cần thiết.

1.1.1. Tổng quan mô hình



1.1.1.1. User story người bán hàng

Là người bán hàng, tôi muốn có account trên website để đăng sản phẩm cần được quảng cáo hoặc các chiến dịch khuyến mãi. Trên website có các chức năng như thêm, sửa, xóa các sản phẩm, ngày giờ phát quảng cáo, có thể tùy chọn lựa form có sẵn hoặc tự thiết kế form catalogue để tôi có thể trang trí và thêm những thông tin cần thiết của catalogue.

1.1.1.2. User story người khách hàng (Consumer)

Là người khách hàng, khi tôi muốn nhận thông báo quảng cáo từ các cửa hàng mà tôi đi qua (đi ngoài đường, trung tâm thương mại,...) thì tôi mở app kèm bật Bluetooth. Khi đi ngang qua một điểm thiết bị quảng cáo (BT Advertiser) gần với cửa hàng của người bán hàng quảng cáo thì điện thoại tôi rung lên tự động nhận catalogue về. Sau đó khách hàng(Consumer) vào app xem danh sách catalogue nhận được mà không cần mở kết nối Internet.

1.1.1.3. User story hệ thống trung gian (BT Advertiser)

Là thiết bị trung gian, Khi định kỳ có 1 danh sách catalogue mới cần được cập nhật thì sẽ lên hàng chờ RabbitMQ để nhận message về và lưu lại. Sau đó sẽ phát ra từng gói tin quảng cáo của catalogue để cho điện thoại của khách hàng (Consumer) nhận và tổng hợp thành 1 catalogue hoàn chỉnh, sau đó sẽ hiển thị lên cho khách hàng xem.

1.2. Công nghệ Bluetooth

1.2.1. Tổng quan

Bluetooth là công nghệ không dây cho phép các thiết bị điện tử giao tiếp với nhau trong khoảng cách ngắn bằng sóng vô tuyến qua băng tần chung ISM (Industrial, Scientific, Medical) trong dãy tần 2.40 – 2.48 GHz. Đây là dãy băng tần không cần đăng ký, được dành riêng để dùng cho các thiết bị không dây trong công nghiệp, khoa học, y tế... Bluetooth được thiết kế nhằm mục đích thay thế dây cáp, kết nối vô tuyến giữa các thiết bị điện tử lại với nhau một cách thuận lợi với giá thành rẻ. Khi kích hoạt, Bluetooth có thể tự động định vị những thiết bị khác có chung công nghệ trong vùng phủ sóng xung quanh và bắt đầu kết nối với nhau. Nó được định hướng sử dụng cho việc truyền dữ liệu, hình ảnh lẫn tiếng nói.

1.2.2. Đặc điểm

- Tiêu thụ năng lượng thấp, cho phép ứng dụng được cho nhiều loại thiết bị, bao gồm cả thiết bị cầm tay.
- Giá thành thấp.
- Khoảng cách giao tiếp cho phép: Giữa hai thiết bị đầu cuối có thể lên đến 10m ngoài trời và ở môi trường có vật cản, khoảng cách các thiết bị đầu cuối và Access point có thể lên tới 100m và môi trường có vật cản là 30m.
- Bluetooth sử dụng dải băng tần không cần đăng ký 2.4 GHz trên băng tần ISM. Tốc độ truyền tải dữ liệu có thể đạt tối đa 1 Mbps (do sử dụng tần số cao) mà các thiết bị không cần phải thấy trực tiếp nhau (light-of-sight requirements).
- Dễ dàng trong việc phát triển ứng dụng: Bluetooth kết nối một ứng dụng này với một ứng dụng hoặc phần mềm khác thông qua các chuẩn “Bluetooth profiles”, do đó có thể độc lập về phần cứng cũng như hệ điều hành sử dụng.
- Bluetooth Low Energy sử dụng 40 kênh truyền khác nhau, 3 kênh phát quảng cáo chính và 37 còn lại dùng để truyền dữ liệu.
- Bluetooth được dùng trong giao tiếp dữ liệu và âm thanh: Có 3 kênh để truyền âm thanh và 7 kênh để truyền dữ liệu trong một mạng cá nhân.
- An toàn và bảo mật: Được tích hợp với sự xác nhận và mã hóa (build in authentication encryption).

1.3. Bluetooth Low Energy(BLE)

1.3.1. Tổng quan

Bluetooth Low Energy (BLE) là công nghệ giao tiếp không dây công suất thấp có thể được sử dụng trong một khoảng cách ngắn để cho phép các thiết bị thông minh giao tiếp. Một số thiết bị bạn tương tác hàng ngày như điện thoại thông minh, đồng hồ thông minh, thiết bị theo dõi thể dục, tai nghe không dây và máy tính đang sử dụng BLE để tạo trải nghiệm liền mạch giữa các thiết bị.

BLE là một tiêu chuẩn Bluetooth tương đối mới được xác định bởi nhóm lợi ích đặc biệt Bluetooth (SIG) với trọng tâm là năng lượng thấp. Nó cho các nhà sản xuất thiết bị thêm giao diện truyền thông năng lượng thấp trên các giải pháp hiện có. Nó cũng được sử dụng để tạo ra các thiết bị tiêu thụ điện năng thấp như đèn hiệu có thể được cung cấp năng lượng bằng pin đồng xu nhỏ trong nhiều tháng hoặc thậm chí nhiều năm.

BLE có nhiều khả năng và được triển khai trong một loạt các lĩnh vực như sức khỏe, thể dục, bảo mật, tự động hóa gia đình, giải trí gia đình, công nghiệp thông minh và IoT (Internet of Things). Nó cũng được ứng dụng trong điện thoại thông minh và máy tính xách tay mà chúng ta sử dụng hàng ngày.

1.3.2. Vai trò

Để hiểu cách hoạt động của BLE, chúng ta cần xác định một số thuật ngữ và khái niệm. Thiết bị BLE hoạt động ở vai trò trung tâm hoặc ngoại vi và đôi khi còn được gọi là client hoặc máy server.

Trung tâm(Client): Thiết bị khởi tạo các lệnh và yêu cầu cũng như chấp nhận các phản hồi.

Ngoại vi(Server): Thiết bị nhận lệnh và yêu cầu và trả về phản hồi.

Chúng ta thường nghĩ về thiết bị BLE với vai trò thiết bị ngoại vi qua những thứ như tai nghe, máy theo dõi thể dục, máy đo nhịp tim,... Nếu thiết bị báo tính khả dụng của nó để kết nối và cung cấp giao diện để giao tiếp thì nó chính là thiết bị ngoại vi.

Thay vào đó nếu thiết bị phát hiện, kết nối và tương tác với các thiết bị ngoại vi thì nó chính là thiết bị trung tâm. Chẳng hạn như máy tính hoặc điện thoại thông minh. Trung tâm thường quét các thiết bị ngoại vi gần đó và hiển thị danh sách các thiết bị được tìm thấy. Nếu thiết bị ngoại vi có thể kết nối, trung tâm có thể kết nối với thiết bị ngoại vi được chỉ định và bắt đầu trao đổi thông tin.

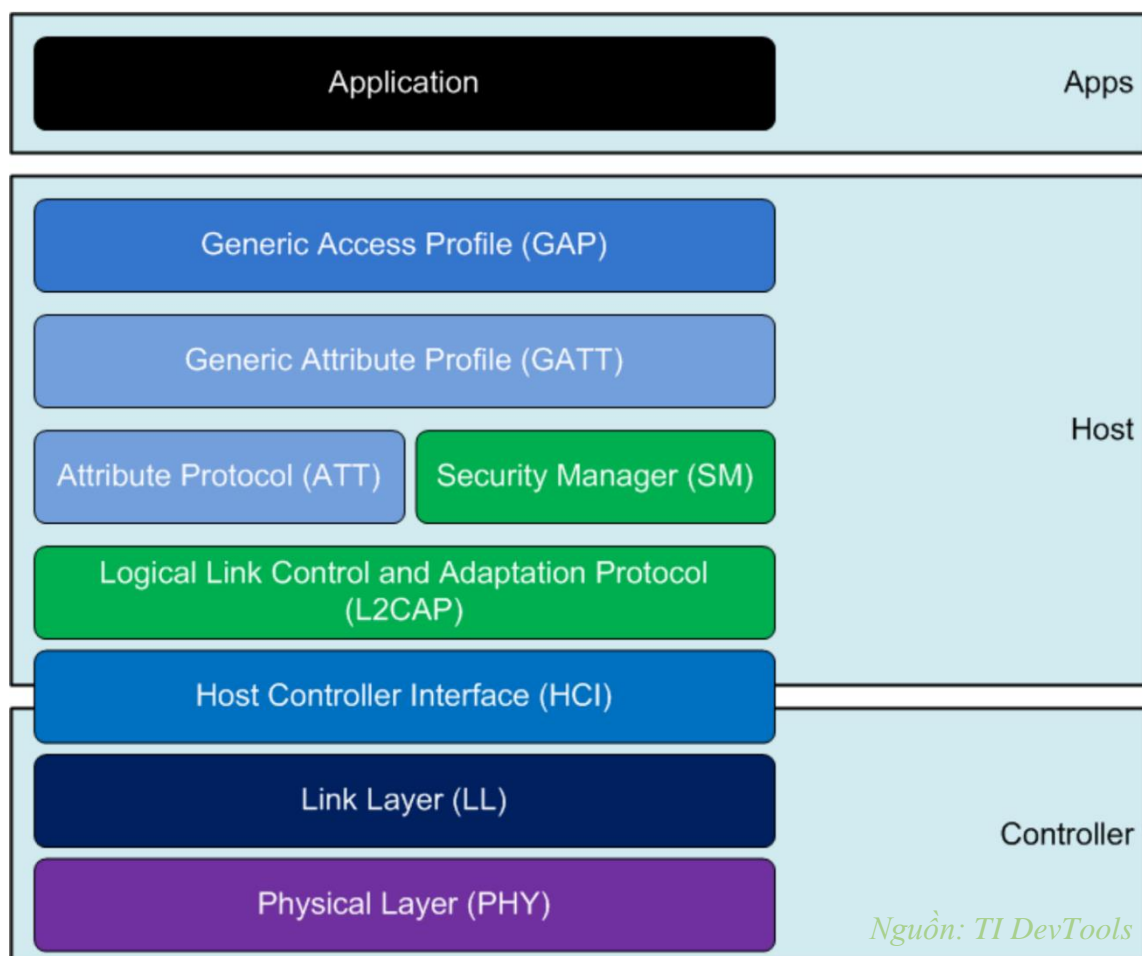
1.3.3. BLE Protocol Stack

1.3.3.1. Tổng quan

BLE Protocol Stack mô tả chức năng của ngăn xếp giao thức năng lượng thấp Bluetooth(BLE) và cung cấp danh sách các API để giao tiếp với ngăn xếp giao thức. Dự án ngăn xếp và các tệp liên quan của nó phục vụ để triển khai tác vụ ngăn xếp giao thức năng lượng thấp Bluetooth. Đây là tác vụ ưu tiên cao nhất trong hệ thống và nó thực hiện ngăn xếp giao thức năng lượng thấp Bluetooth.

Hầu hết ngăn xếp giao thức năng lượng thấp của Bluetooth là mã đối tượng trong một tệp thư viện duy nhất. Một nhà phát triển phải hiểu chức năng của các lớp ngăn xếp giao thức khác nhau và cách chúng tương tác với ứng dụng và cấu hình.

1.3.3.2. Các tầng giao thức chính



Hình 1. Ngăn xếp giao thức năng lượng thấp Bluetooth

Trên hình 1 là sơ đồ mô tả kiến trúc các lớp chính của ngăn xếp giao thức BLE bao gồm:

- + Controller.
- + Host.
- + Application.

1.3.3.3. Physical Layer(PHY)

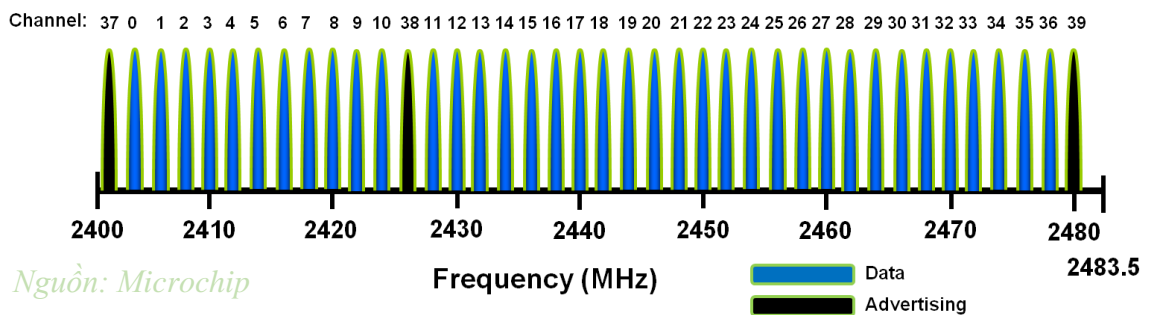
a. Tổng quan

Physical layer Bluetooth Low Energy(PHY) chứa mạch phát và nhận tín hiệu, chịu trách nhiệm dịch tín hiệu số và truyền qua sóng vô tuyến. Physical layer là lớp thấp nhất của ngăn xếp giao thức BLE, chịu trách nhiệm cung cấp dịch vụ cho Link Layer(LL).

b. Dải tần số

Physical layer giao tiếp với nhau trong khoảng cách ngắn bằng sóng vô tuyến qua băng tần chung ISM(Industrial, Scientific, Medical) trong dãy tần 2.40 – 2.48 GHz, dải băng tần này được chia làm 40 kênh, mỗi kênh cách nhau khoảng 2 MHz, khoảng cách từ 2400 MHz đến 2483.5 MHz và được bắt đầu từ 2402 MHz.

Dải băng tần 40 kênh được chia thành 3 kênh quảng cáo(37, 38, 39), và 37 kênh còn lại dùng để truyền dữ liệu(0 - 36).



Hình 2. Dải băng tần Physical layer

1.3.3.4. Link Layer(LL)

a. Tổng quan

Link Layer Bluetooth Low Energy(LL) là lớp giao tiếp trực tiếp với Physical Layer(PHY), Link Layer chịu trách nhiệm quảng cáo, quét và khởi tạo cũng như duy trì kết nối.

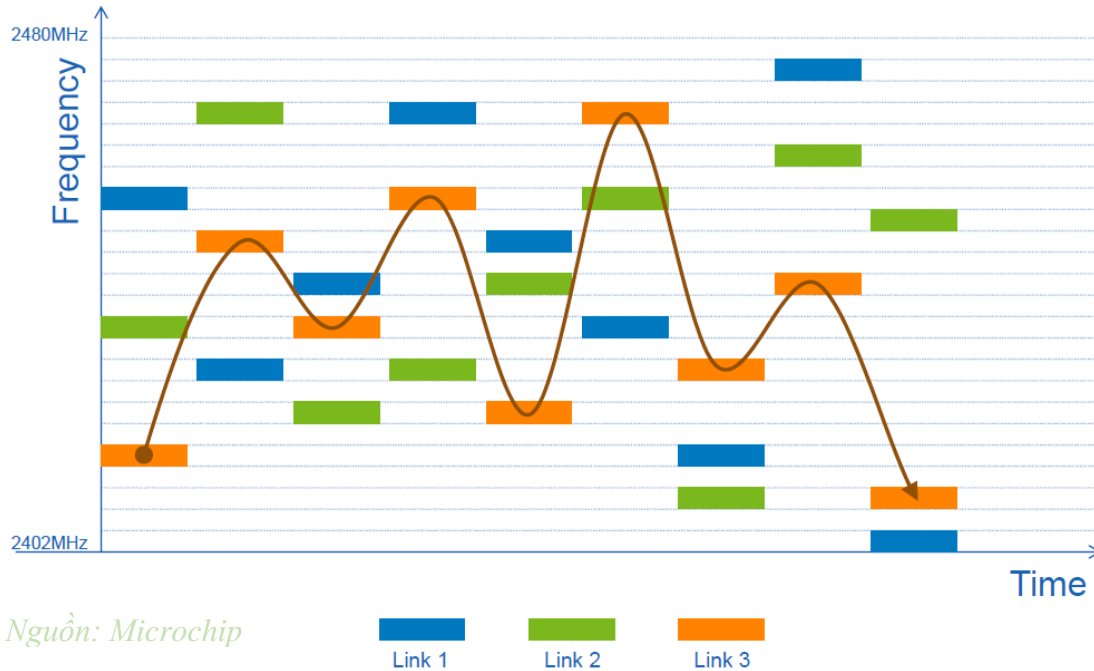
b. Kênh truyền

- Các kênh quảng cáo có chức năng:
 - + Khám phá thiết bị.
 - + Thiết lập kết nối.
 - + Truyền phát sóng quảng bá.
- Các kênh dữ liệu có chức năng:
 - + Giao tiếp 2 chiều giữa các thiết bị kết nối.
 - + Nhảy tần thích ứng được sử dụng cho các sự kiện tiếp theo.
- Nhảy băng tần thích ứng:
 - + Khi ở trong một kết nối dữ liệu, một thực toán nhảy tần được sử dụng để chuyển qua 37 kênh truyền dữ liệu:

$$f_n + 1 = (f_n + hop) \bmod 37$$

Trong đó $f_n + 1$ là tần số (kênh) sẽ sử dụng cho sự kiện kết nối tiếp theo và hop là giá trị có thể nằm trong khoảng từ 5-16 và được đặt khi kết nối được tạo. Nó được thêm vào mô-đun tần số cuối cùng 37.

Sơ đồ sau đây mô tả ba kết nối BLE đang hoạt động, hiển thị trình tự nhảy tần (nhảy tần trên Liên kết 3):

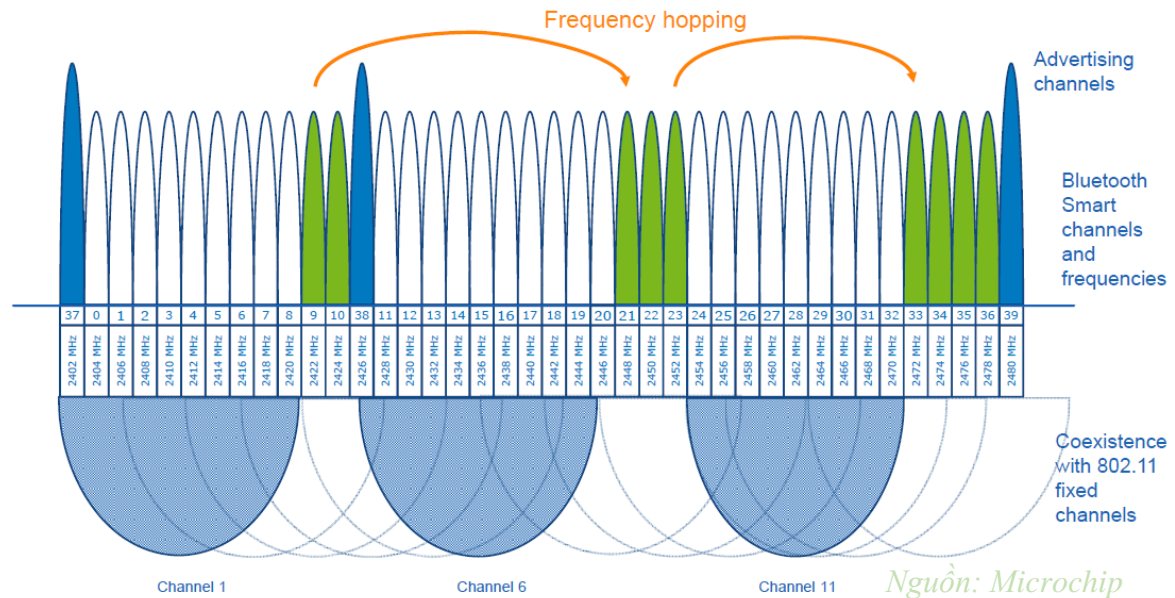


Hình 3. Nhảy tần thích ứng của Link Layer(LL)

Biểu đồ cho thấy nhảy tần cung cấp một phương pháp mạnh mẽ để duy trì kết nối khi có nhiều hoặc nhiều thiết bị khác trong dải vô tuyến.

Cơ chế nhảy tần thích ứng được lớp liên kết sử dụng để hoán đổi lại một gói nhất định từ một kênh xấu đã biết thành một kênh tốt đã biết để giảm nhiễu từ các thiết bị khác sử dụng Wifi.

Ví dụ: giả sử rằng một thiết bị BLE ở cùng khu vực với một số mạng Wi-Fi trên các kênh 1, 6 và 11. Thiết bị BLE sẽ đánh dấu các kênh 0-8, 11-20 và 24-32 là các kênh không hợp lệ. Điều này có nghĩa là khi hai thiết bị đang giao tiếp, chúng sẽ chuyển động qua các kênh và ánh xạ lại các kênh này thành một tập hợp các kênh tốt như được minh họa:



Hình 4. Nhảy tần thích ứng tránh xung đột với Wifi

1.3.3.5. Host Controller Interface(HCI)

Lớp giao diện bộ điều khiển máy chủ (HCI) là một lớp mỏng truyền các lệnh và sự kiện giữa các phần tử máy chủ và bộ điều khiển của ngăn xếp giao thức Bluetooth. Trong một ứng dụng bộ xử lý mạng thuần, lớp HCI được thực hiện thông qua một giao thức truyền tải như SPI hoặc UART.

Trong các dự án MCU không dây nhúng, chẳng hạn như dự án đơn giản, lớp HCI được thực hiện thông qua các lệnh gọi hàm và lệnh gọi lại trong MCU không dây. Tất cả các lệnh và sự kiện được thảo luận giao tiếp với bộ điều khiển, chẳng hạn như ATT, GAP, v.v., cuối cùng sẽ gọi một API HCI để chuyển từ các lớp trên của ngăn xếp giao thức qua lớp HCI tới bộ điều khiển. Tương tự như vậy, bộ điều khiển gửi dữ liệu và sự kiện đã nhận đến máy chủ và các lớp trên thông qua HCI.

Cũng như các lệnh Bluetooth BLE HCI tiêu chuẩn, một số lệnh dành riêng cho nhà cung cấp phần mở rộng HCI có sẵn để mở rộng một số chức năng của bộ điều khiển để ứng dụng sử dụng.

1.3.3.6. Logical Link Control & Adaptation Protocol(L2CAP)

Lớp L2CAP cung cấp các dịch vụ đóng gói dữ liệu cho các lớp trên, cho phép truyền dữ liệu từ đầu đến cuối một cách logic.

L2CAP cung cấp dịch vụ hướng kết nối và phi kết nối cho các tầng giao thức bên trên.

L2CAP có khả năng phân kênh (multiplexing), phân đoạn (segmentation) và tái tổ hợp (reassembly operation).

L2CAP cho phép các giao thức ở các tầng trên và các ứng dụng truyền và nhận dữ liệu.

Mỗi gói dữ liệu của L2CAP tối đa 64 Kbytes.

1.3.3.7. Security Manager(SMP)

Security Manager(SMP) xác định các phương thức ghép nối và phân phối khóa, đồng thời cung cấp các chức năng cho các lớp khác của ngăn xếp giao thức để kết nối và trao đổi dữ liệu với thiết bị khác một cách an toàn. Cung cấp các ứng dụng chạy qua ngăn xếp Bluetooth Low Energy với các loại dịch vụ sau:

- + Device Authentication.
- + Device Authorization.
- + Data Integrity.
- + Data Confidentiality.
- + Data Privacy.

1.3.3.8. Attribute Protocol(ATT)

Attribute Protocol(ATT) là một lớp cấp thấp xác định cách truyền dữ liệu. Nó xác định các thuộc tính khám phá thiết bị, đọc và ghi trên một thiết bị khác. Mặt khác, Generic Attribute Profile(GATT) được xây dựng phía trên ATT để cung cấp các dịch vụ cấp cao cho nhà sản xuất triển khai BLE. Các dịch vụ này về cơ bản được sử dụng để quản lý quá trình truyền dữ liệu một cách có hệ thống hơn.

1.3.3.9. Generic Attribute Protocol(GATT)

a. Tổng quan

GATT sử dụng Giao thức thuộc tính (ATT) làm cơ chế vận chuyển, cũng như phương tiện tổ chức dữ liệu của bạn thành các bit hoặc thuộc tính dễ dàng truyền.

GATT định nghĩa các hoạt động này và các hoạt động cơ bản khác theo cách thức tiêu chuẩn, được hiểu chung. Điều quan trọng là phải hiểu GATT, vì hầu hết các triển khai BLE đều cung cấp API GATT cho các ứng dụng muốn sử dụng chức năng này.

b. Universally Unique Identifier(UUID)

Universally Unique Identifier(UUID) là số 128-bit(16 bytes) duy nhất trên toàn cầu được sử dụng để xác định cấu hình, dịch vụ và kiểu dữ liệu trong cấu hình Thuộc tính chung (GATT). Các định dạng rút gọn chỉ có thể sử dụng với các cấu hình GATT được xác định bởi Bluetooth SIG.

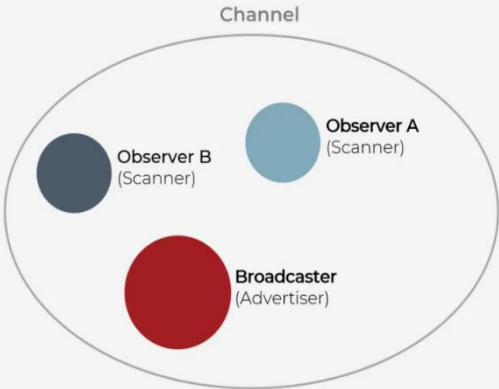
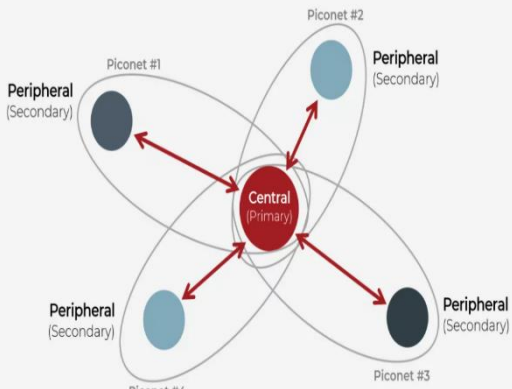
Để tối ưu rút ngắn UUID 128-bit, thông số kỹ thuật Bluetooth Low Energy(BLE) bổ sung hỗ trợ cho các UUID 16-bit rút gọn.

1.3.3.10. Generic Access Profile(GAP)

a. Tổng quan

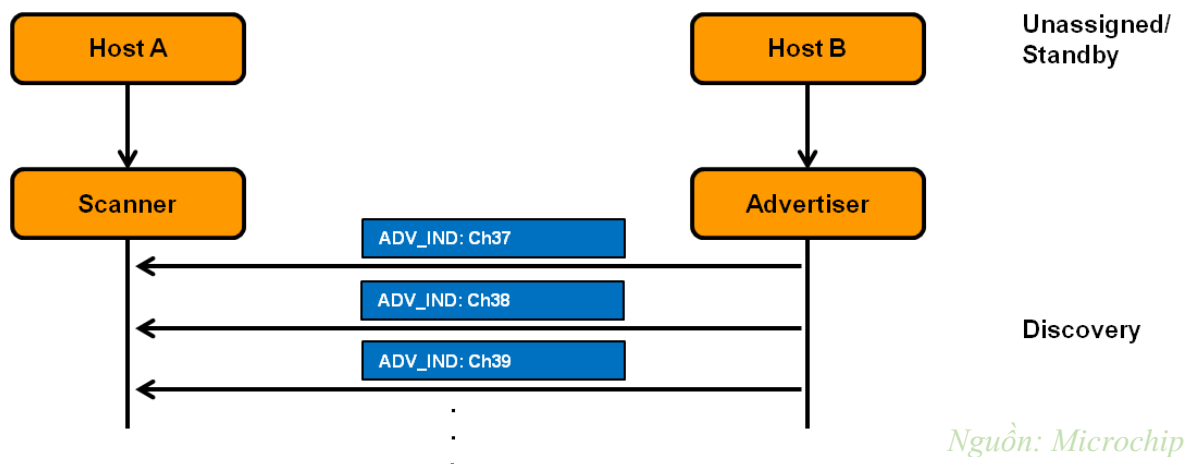
Generic Access Profile(GAP) xác định cấu trúc liên kết chung của ngăn xếp mạng Bluetooth Low Energy(BLE). Có hai cơ chế mà thiết bị BLE có thể sử dụng để giao tiếp với các thiết bị khác: Broadcasting hoặc Connecting. Các cơ chế này tuân theo các nguyên tắc về Generic Access Profile (GAP). GAP xác định các thiết bị hỗ trợ BLE có thể tự phát sóng quảng bá hoặc tự giao tiếp trực tiếp với nhau.

b. Vai trò

Vai trò GAP	Không kết nối	Định hướng kết nối
Chủ động	Broadcaster <ul style="list-style-type: none"> Đóng vai trò là 1 thiết bị quảng cáo. Chỉ truyền gói tin. Định kỳ gửi các gói dữ liệu quảng cáo. Không cho phép kết nối. 	Peripheral <ul style="list-style-type: none"> Là 1 thiết bị Client (Slave). Chấp nhận yêu cầu kết nối. Cho phép kết nối với 1 thiết bị duy nhất Server (Central).
Thụ động	Observer <ul style="list-style-type: none"> Đóng vai trò là 1 thiết bị quét. Chỉ nhận gói tin. Tìm kiếm các gói dữ liệu quảng cáo. Không bắt đầu kết nối. 	Central <ul style="list-style-type: none"> Là 1 thiết bị khởi xướng kết nối – Server (Master). Tìm kiếm các gói quảng cáo để khởi xướng kết nối. Cho phép nhiều kết nối từ Peripheral (tối đa 7 Slave).
Dữ liệu	<ul style="list-style-type: none"> Công cộng. Quảng cáo 1 chiều tới thiết bị quét. Tải trọng gói quảng cáo 31 byte. Tải trọng phản hồi quét 31 byte. 	<ul style="list-style-type: none"> Trước khi kết nối giống như vai trò hoạt động Broad-caster/Observer. Sau khi kết nối dữ liệu thì giống như giao thức GATT, riêng tư, giao tiếp 2 chiều.
Biểu đồ		

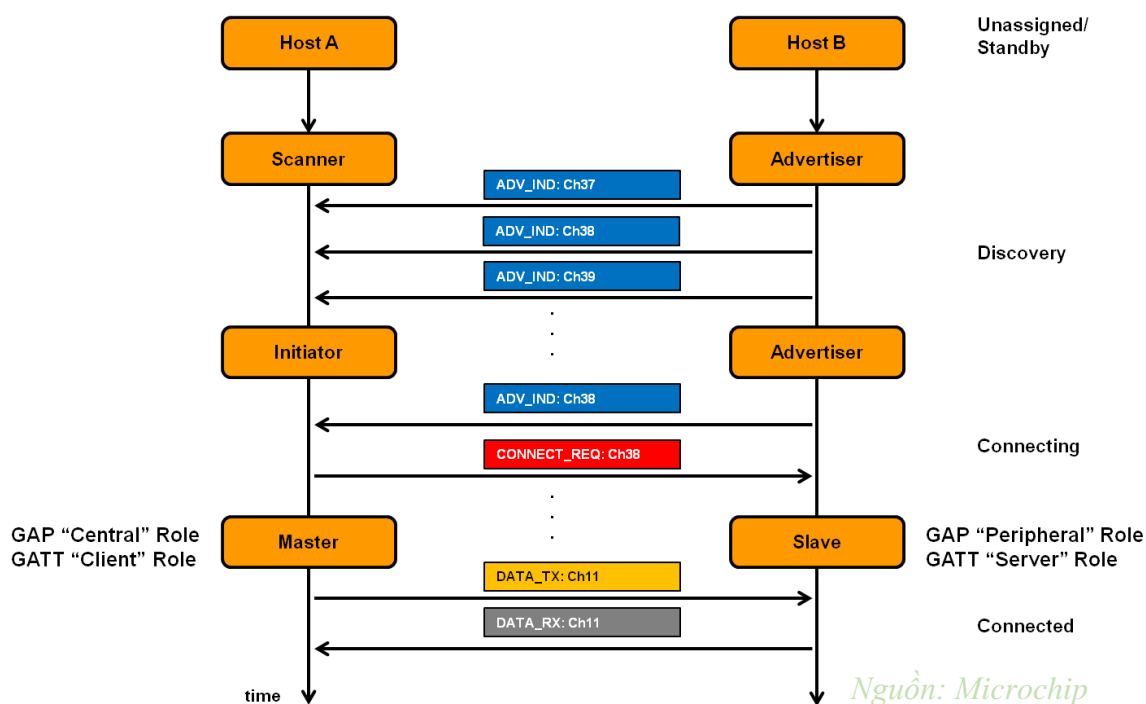
Bảng 1. Vai trò Generic Access Profile(GAP)

Mô hình không kết nối: Sơ đồ mô tả hai thiết bị BLE, ban đầu trong trạng thái chờ không có kết nối. Sau đó Host B bước trạng thái quảng cáo và gửi các gói tin, còn Host A tiến vào trạng thái khám phá quét các thiết bị xung quanh. Tất cả các máy quét đều nhận được gói tin quảng cáo khi trong trạng thái này.



Hình 6. Mô hình không kết nối Generic Access Profile(GAP)

Trong mô hình có kết nối thì sau khi nhận gói tin quảng cáo từ Host B thì Host A lọc và phân tích gói tin. Sau đó Host A trở thành thiết bị khởi xướng và quyết định bắt đầu kết nối với Host B, và cuối cùng Host B chấp nhận yêu cầu kết nối và sau đó được đánh đổi vai trò cho nhau Host A thành Master còn Host B thành Slave.



Hình 7. Mô hình định hướng kết nối Generic Access Profile(GAP)

1.3.4. BLE Advertisements

1.3.4.1. Tổng quan

Quảng cáo BLE là một trong những khía cạnh quan trọng nhất của Bluetooth Low Energy. Hiểu cách sử dụng quảng cáo đúng cách có thể giúp bạn giảm mức tiêu thụ điện năng, tăng tốc kết nối và cải thiện độ tin cậy. Quảng cáo BLE cũng là chìa khóa để báo hiệu, đã trở nên phổ biến hơn để tạo vị trí và theo dõi. Chúng ta sẽ xem xét cách chúng hoạt động và cách sử dụng chúng.

1.3.4.2. Kiến thức cơ bản về quảng cáo BLE

Thiết bị Bluetooth có thể gửi các gói quảng cáo(PDU) để phát dữ liệu hoặc cho phép các thiết bị Bluetooth khác tìm thấy và kết nối.

Generic Access Profile(GAP) là phần liên quan tới BLE Advertisement giúp xác định cấu trúc liên kết chung, cũng như vai trò phát sóng quảng bá không kết nối hoặc hay định hướng kết nối.

1.3.4.3. Physical BLE Advertisements

Bluetooth Low Energy(BLE) chia sẻ một số điểm tương đồng với Bluetooth Classic. Cả hai đều sử dụng phổ tần 2.4GHz. Basic Rate (BR) và BLE đều sử dụng điều chế GFSK ở tốc độ 1Mbps, nhưng chỉ số điều chế của chúng khác nhau. Tốc độ dữ liệu nâng cao (EDR) sử dụng một điều chế hoàn toàn khác với GFSK. Bluetooth cổ điển có 79 kênh so với 40 kênh của BLE. Các kênh cũng có khoảng cách khác nhau. Cả hai sự khác biệt này làm cho BLE và Classic trở nên khác biệt và không tương thích với nhau, vì vậy chúng không thể giao tiếp.

	BLE	CLASSIC	
	BLE	BR	EDR
Modulation	GFSK 0.45 đến 0.55; 0.5(Modulation ổn định).	GFSK 0.28 đến 0.35	DQPSK/8DPSK
Data Rate	1 Mbps, 2 Mbps(Bluetooth 5.0)	1Mbps	2Mbps, 3Mbps
Channels	40	79	79
Spacing	2MHz	1MHz	1MHz

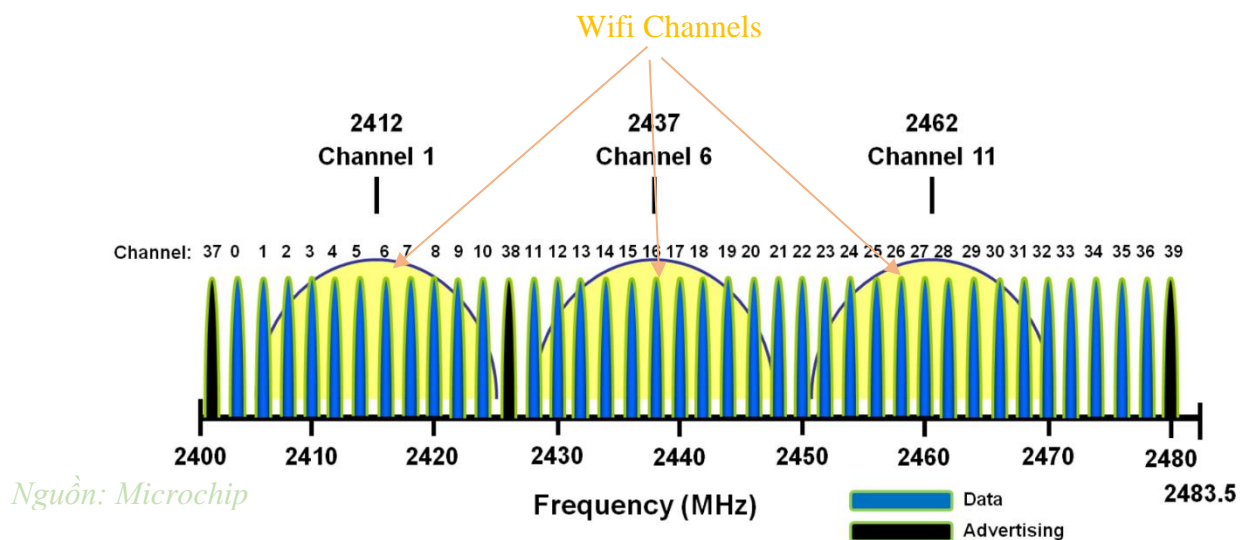
Bảng 2. So sánh tầng vật lý giữa BLE & Bluetooth Classic

Các kênh 37, 38 và 39 chỉ được sử dụng để gửi các gói quảng cáo. Phần còn lại được sử dụng để trao đổi dữ liệu trong quá trình kết nối. Chúng ta quan tâm đến những gì đang xảy ra trong 3 kênh này và đó là những gì em sẽ đề cập ở đây. Điều quan trọng cần lưu ý là với thông số kỹ thuật Bluetooth mới nhất, Tiện ích mở rộng quảng cáo cũng cho phép sử dụng các kênh khác để quảng cáo, nhưng trước tiên chúng ta sẽ tập trung vào 3 kênh chính.

Trong quá trình quảng cáo BLE, một thiết bị ngoại vi BLE truyền cùng một gói tin trên 3 kênh quảng cáo, cái này đến kênh khác. Thiết bị trung tâm quét các thiết bị hoặc đèn hiệu sẽ lắng nghe các kênh đó để tìm các gói quảng cáo, giúp nó phát hiện ra các thiết bị lân cận.

Các kênh 37, 38 và 39 được trải rộng trên phổ tần 2.4GHz có chủ đích. 37 và 39 là kênh đầu tiên và cuối cùng trong dải, trong khi 38 nằm ở giữa. Nếu bất kỳ kênh quảng cáo đơn lẻ nào bị chặn, các kênh khác có thể sẽ không còn hoạt động vì chúng cách nhau một vài MHz bằng thông.

Điều này đặc biệt đúng vì hầu hết các thiết bị khác can thiệp vào BLE là băng tần hẹp. Cụ thể, kênh 38 được đặt giữa các kênh Wi-Fi 1 và 6 nên nó tránh được tín hiệu Wi-Fi. Khoảng cách rộng giữa các kênh quảng cáo giúp BLE quản lý tốt hơn việc gây nhiễu từ Wi-Fi, Bluetooth cổ điển, Lò vi sóng... để đảm bảo rằng quảng cáo thành công.

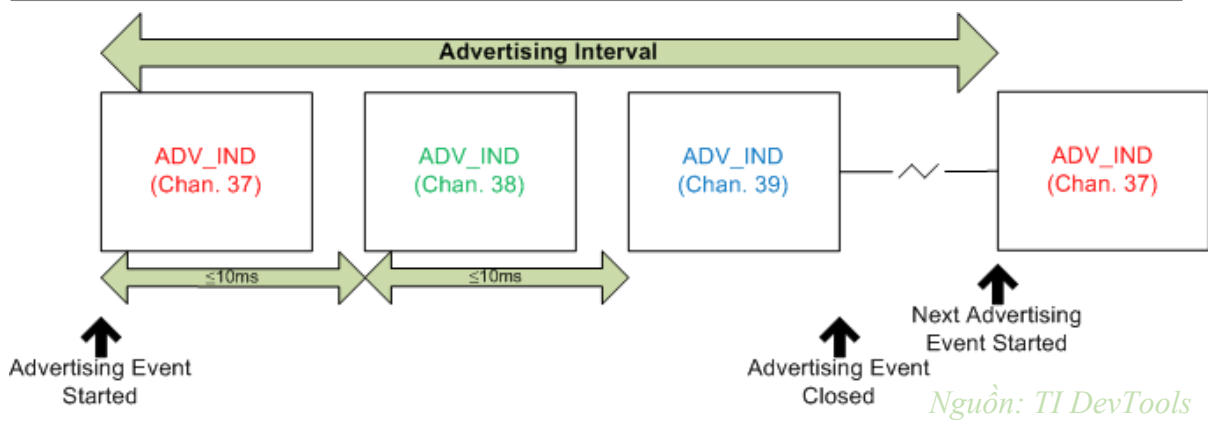


Hình 8. Dải tần số các kênh Bluetooth và Wifi

1.3.4.4. BLE Advertisements Interval

Khi thiết bị ngoại vi BLE ở chế độ quảng cáo, các gói quảng cáo được gửi định kỳ trên mỗi kênh quảng cáo. Khoảng thời gian giữa tập gói có cả khoảng thời gian cố định và khoảng thời gian trễ ngẫu nhiên. Khoảng thời gian được xác định giữa bộ 3 gói (và 3 kênh hầu như luôn được sử dụng).

Có thể đặt khoảng thời gian cố định từ 20ms đến 10,24 giây, trong các bước 0,625ms. Độ trễ ngẫu nhiên là một giá trị giả ngẫu nhiên từ 0ms đến 10ms được tự động thêm vào. Sự ngẫu nhiên này giúp giảm khả năng va chạm giữa các quảng cáo của các thiết bị khác nhau (nếu chúng giảm với cùng một tỷ lệ, chúng có thể dễ gây nhiễu hơn). Đây chỉ là một trong những cách mà BLE sử dụng để cải thiện trách gây nhiễu.



Hình 9. BLE Advertisements Interval

1.3.4.5. Các loại quảng cáo khác nhau

Dữ liệu quảng cáo bao gồm tối đa 31 byte dữ liệu người dùng có thể định cấu hình. Thêm 31 byte có thể được gửi dưới dạng phản hồi quét cho một yêu cầu quét. Có tám loại quảng cáo khác nhau trong đó có loại cuối bảng đã được thêm vào Bluetooth 5. Những PDU đó được gọi là PDU quảng cáo mở rộng. Các gói bắt đầu bằng ADV_ được truyền trên các kênh quảng cáo chính(37, 38, 39). Các gói bắt đầu bằng AUX_ được truyền trên các kênh quảng cáo phụ(0 - 36).

PDU quảng cáo	Miêu tả	Độ dài dữ liệu quảng cáo tối đa	Cho phép yêu cầu quét	Cho phép kết nối
ADV_IND	Được sử dụng để gửi quảng cáo vô hướng có thể kết nối.	31 byte	Có	Có
ADV_DIRECT_IND	Được sử dụng để gửi quảng cáo định hướng, có thể kết nối	-	Không	Có
ADV_SCAN_IND	Được sử dụng để quảng cáo vô hướng có thể quét	31 byte	Có	Không
ADV_NON-CONN_IND	Được sử dụng để gửi quảng cáo vô hướng không thể kết nối.	31 byte	Không	Không
ADV_EXT_IND	Được sử dụng để chỉ ra rằng một quảng cáo sẽ được gửi trên một kênh quảng cáo phụ.	Không cho phép dữ liệu quảng cáo.	Không	Không
AUX_ADV_IND	Được sử dụng để quảng cáo trực tiếp có thể kết nối trên một kênh quảng cáo phụ.	254 byte	Có	Có

AUX_SYNC_IND	Được sử dụng cho các quảng cáo định kỳ trên các kênh quảng cáo thứ cấp.	254 byte	Không	Không
AUX_CHAIN_IND	Được sử dụng để kết hợp các gói quảng cáo, cho phép dữ liệu quảng cáo mở rộng.	254 byte	-	-

Bảng 3. Những loại quảng cáo PDU

1.3.4.6. Advertisements Packet Structure

a. Advertising Packet – Preamble



Preamble là phần mở đầu mang giá trị 1 byte, được sử dụng để đồng bộ hóa và ước tính thời gian để Receiver có thể điều biên độ, tần số cho phù hợp. Preamble luôn mang giá trị 10101010(0xAA) cho các gói được phát phần mở đầu.

b. Advertising Packet – Access Address



Giá trị của trường Access Address được cố định cho một gói quảng cáo, nó không thật sự là một địa chỉ nhưng nó là một mã dùng để chỉ ra gói tin này là của loại quảng cáo nào.

Kích thước của trường Access Address là 4 byte.

c. Advertising Packet – Protocol Data Unit



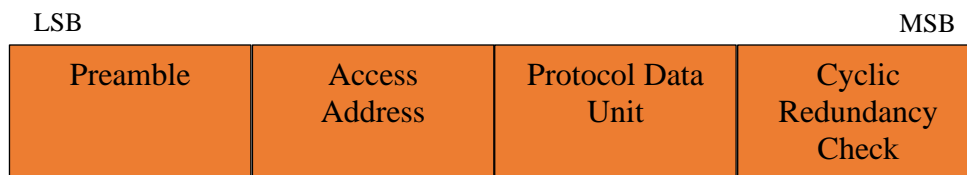
Protocol Data Unit(PDU) hay còn gọi là dữ liệu giao thức, đơn vị kích thước của trường này có thể khác nhau từ 2 byte đến 39 byte.

d. Advertising Packet – CRC

Cyclic Redundancy
Check

Cyclic Redundancy Check(CRC) hay còn gọi là chu kỳ kiểm tra, kích thước 24 bit được sử dụng để phát hiện bất kỳ sự thay đổi ngẫu nhiên nào trong các giá trị bit nếu CRC kiểm tra không thành công, có nghĩa là nếu gói tin bị hỏng thì sau đó toàn bộ gói tin đó bị loại bỏ.

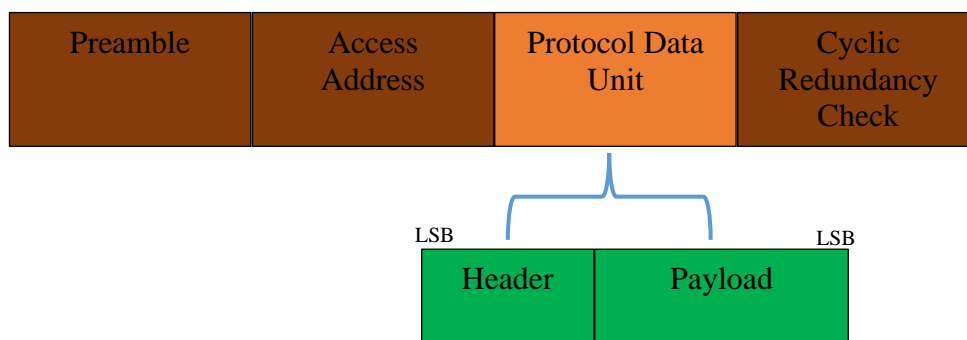
e. Advertising Packet – Link Layer



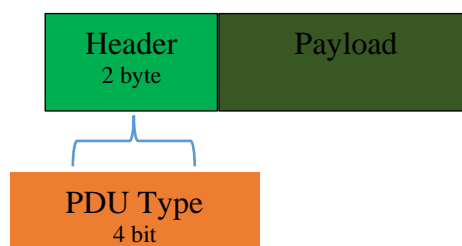
Đây là gói quảng cáo Bluetooth hoàn chỉnh sẽ trông giống như này, lưu ý rằng các gói được truyền đi với lsb đầu tiên và Preamble được truyền đầu tiên.

Chúng tiếp lấy trường Protocol Data Unit(PDU) để phân tích tiếp, các trường còn lại được điền tự động bởi ngân xếp BLE.

f. Protocol Data Unit(PDU)



PDU được chia nhỏ hơn nữa thành các trường Header(2 byte) và Payload có thể dài tới 37 byte.



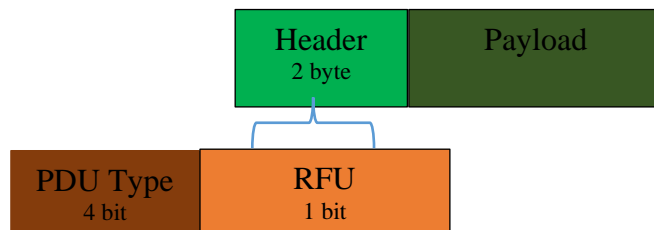
Header gồm có 6 trường:

+ PDU Type: Cho biết đó có phải là một PDU quảng cáo hay không.

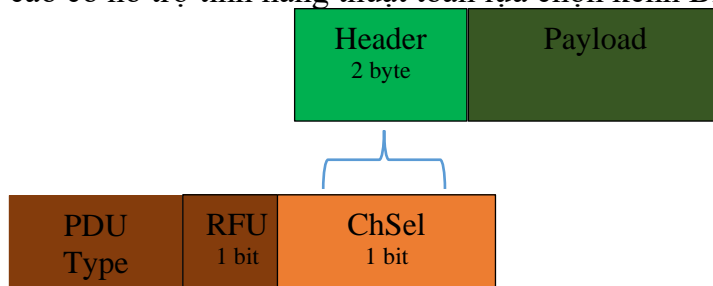
PDU Type	Packet Name	Description
0b0000	ADV_IND	Sự kiện quảng cáo vô hướng có thể kết nối.
0b0010	ADV_NONCONN_IND	Sự kiện quảng cáo vô hướng không thể kết nối.
0b0110	ADV_SCAN_IND	Sự kiện quảng cáo vô hướng có thể quét được.

Bảng 4. PDU Type quảng bá dữ liệu truyền phát.

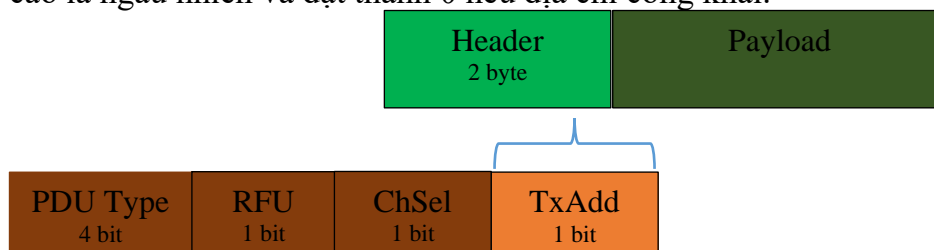
+ PDU Header – RFU: Được dự trữ để sử dụng trong tương lai, kích thước 1 bit.



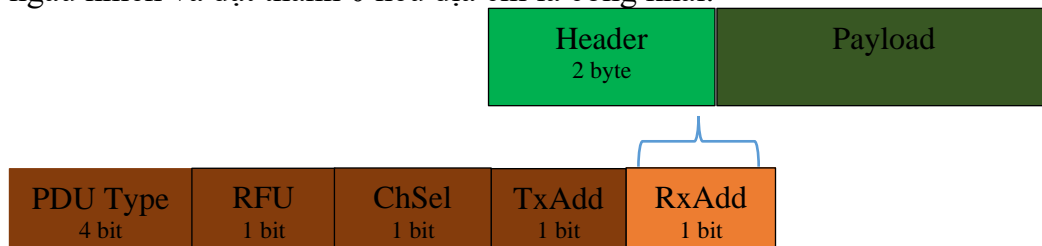
+ PDU Header – Channel Selection: Bit này sẽ được đặt thành 1 nếu nhà quảng cáo có hỗ trợ tính năng thuật toán lựa chọn kênh BLE số 2.



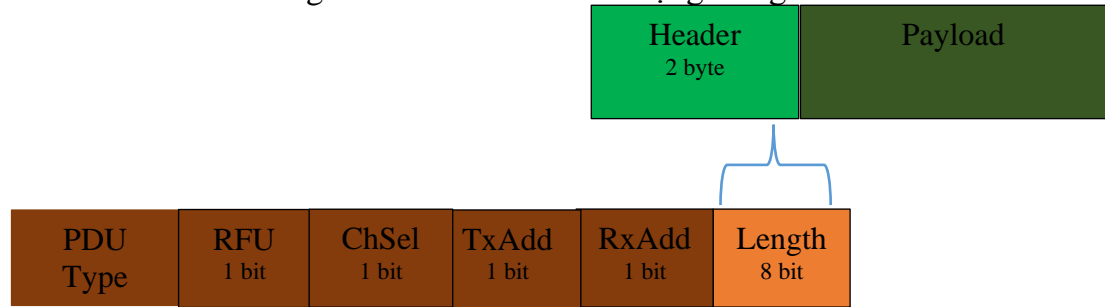
+ PDU Header – TxAdd: Bit này được đặt thành 1 nếu địa chỉ của nhà quảng cáo là ngẫu nhiên và đặt thành 0 nếu địa chỉ công khai.



+ PDU Header – RxAdd: Bit này được đặt thành 1 nếu địa chỉ thiết bị đích là ngẫu nhiên và đặt thành 0 nếu địa chỉ là công khai.

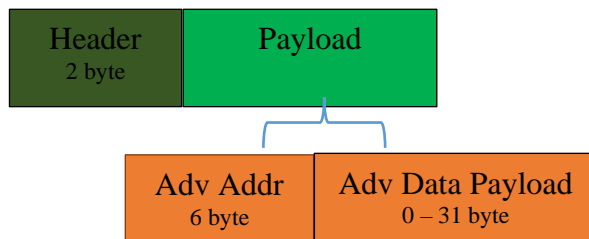


+ PDU Header – Length: Chứa chiều dài tải trọng của gói tin.

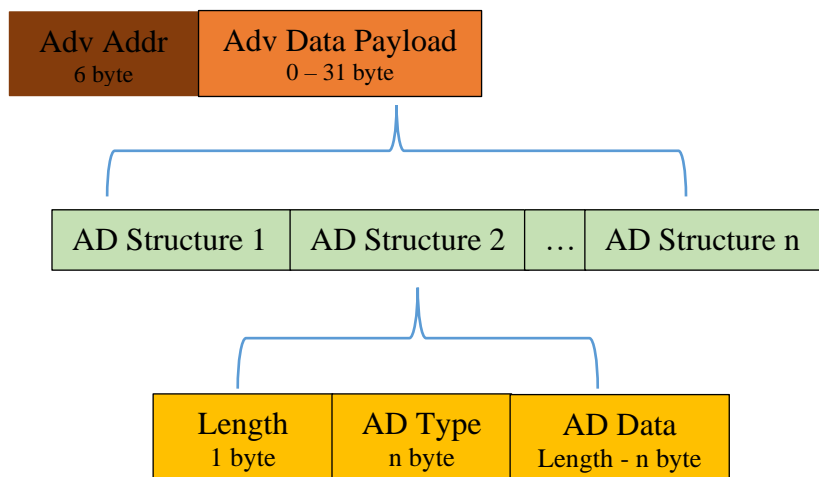


Payload gồm 2 trường:

+ PDU Payload: Payload của gói quảng cáo và kích thước tối đa của Payload phụ thuộc vào loại PDU được sử dụng. Dữ liệu quảng cáo bao gồm thông tin mà nhà quảng cáo muốn chuyển tiếp đến thiết bị Observer / Central device. Địa chỉ thiết bị quảng cáo (MAC) có độ dài 6 byte (dùng để chứa địa chỉ công khai hoặc địa chỉ ngẫu nhiên). Tiếp theo là Adv Data Payload của thiết bị phát quảng cáo có độ dài trường từ 0 – 31 byte, dùng để chứa dữ liệu quảng cáo theo chỉ định của máy chủ.



+ Adv Data Payload AD Structures: Ba trường đầu tiên gồm Length có độ dài 1 byte và Length chỉ định độ dài của AD Structure và không bao gồm 1 byte độ dài của Length. Tiếp theo là AD Type mà việc này chỉ định loại dữ liệu được chứa trong cấu trúc và cuối cùng là trường AD Data chứa dữ liệu thật sự nhưng được xác định bởi AD Type. Danh sách tất cả các AD Type có sẵn và có thể tìm thấy trên Bluetooth SIG.



+ Một số AD Type thường gặp:

Flags: 0x01 (cờ).

Service UUID: 0x03 (Danh sách đầy đủ các UUID lớp dịch vụ 16 bit).

Complete Local Name: 0x09 (Tên thiết bị quảng cáo đầy đủ).

Shortened Local Name: 0x08 (Tên thiết bị quảng cáo rút gọn).

Manufacturer specific data: 0xFF (Dữ liệu dành riêng cho nhà sản xuất).

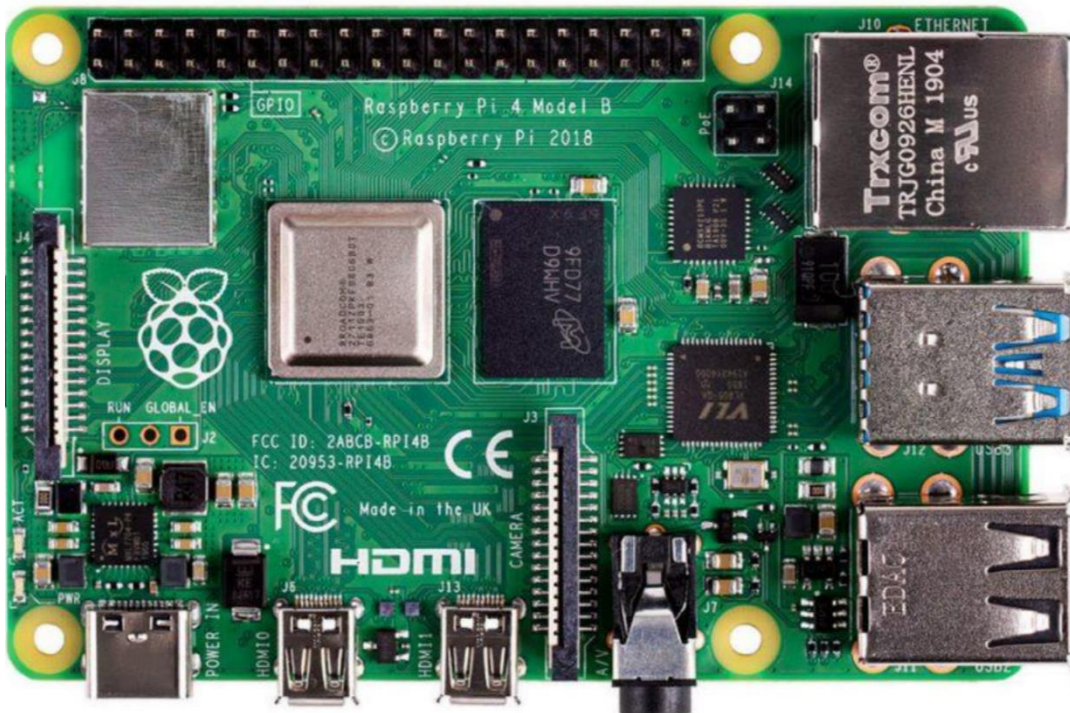
1.4. Raspberry Pi 4B

1.4.1. Tổng quan

Raspberry Pi là từ để chỉ các máy tính có một board mạch (hay còn gọi là máy tính nhúng) kích thước chỉ bằng một thẻ tín dụng, được phát triển tại Anh bởi Raspberry Pi Foundation. Raspberry Pi chủ yếu sử dụng các hệ điều hành dựa trên nhân Linux.

1.4.2. Phần cứng

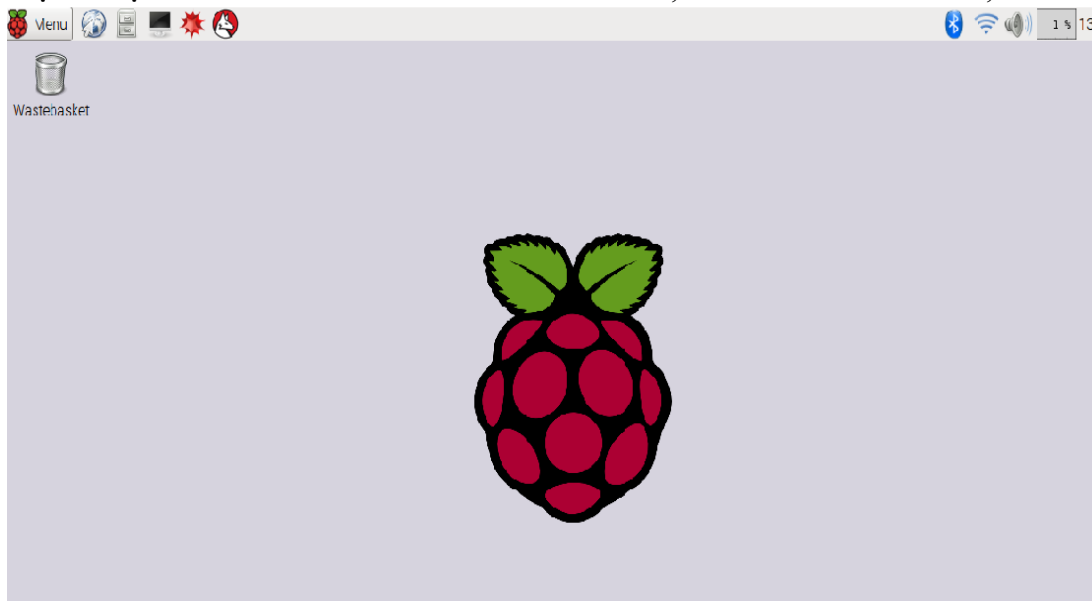
Raspberry Pi 4B được trang bị vi xử lý Broadcom BCM 2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz, RAM 8GB và đặc biệt hỗ trợ chuẩn Wifi 802.11n cùng với Bluetooth 5.0.



Hình 10. Raspberry Pi 4B

1.4.3. Hệ điều hành tương thích

Raspberry Pi có rất nhiều hệ điều hành hỗ trợ có nhân là Linux, trong số đó có Raspbian là hệ điều hành chính thức của Raspberry Pi Foundation. Ngoài ra có thêm một số hệ điều hành tiêu biểu như Ubuntu Mate, Window 10 IoT Core, OSMC...



Hình 11. Hệ điều hành Raspbian

1.5. Lập trình ứng dụng trên Android Studio

1.5.1. Thư viện hỗ trợ Bluetooth trên Android Studio

1.5.1.1. Tổng quan

Nền tảng Android bao gồm hỗ trợ ngăn xếp mạng Bluetooth, cho phép thiết bị trao đổi dữ liệu không dây với các thiết bị Bluetooth khác. Khung ứng dụng cung cấp quyền truy cập vào chức năng Bluetooth thông qua các API Bluetooth. Các API này cho phép các ứng dụng kết nối với các thiết bị Bluetooth khác, cho phép các tính năng không dây như point to point, quét dữ liệu quảng cáo hoặc phát quảng cáo bằng Bluetooth.

Android cung cấp ngăn xếp Bluetooth mặc định hỗ trợ cả Bluetooth cổ điển và Bluetooth năng lượng thấp. Sử dụng Bluetooth, các thiết bị Android có thể tạo mạng khu vực cá nhân để gửi và nhận dữ liệu với các thiết bị Bluetooth gần đó.

Trong Android 4.3 trở lên, ngăn xếp Bluetooth Android cung cấp khả năng triển khai Bluetooth Low Energy (BLE). Để tận dụng hoàn toàn các API BLE, hãy làm theo yêu cầu HCI Bluetooth của Android. Các thiết bị Android có chipset đủ điều kiện có thể triển khai Bluetooth Cổ điển hoặc Bluetooth Cổ điển và BLE. BLE không tương thích ngược với các chipset Bluetooth cũ hơn.

Trong Android 8.0, ngăn xếp Bluetooth gốc hoàn toàn đủ điều kiện cho Bluetooth 5. Để sử dụng các tính năng Bluetooth 5 khả dụng, thiết bị cần có chipset đủ điều kiện Bluetooth 5.

1.5.1.2. Class Interface

android.bluetooth: Các lớp và giao diện bạn cần phải tạo để kết nối Bluetooth.

BluetoothAdapter: Đại diện cho bộ điều hợp Bluetooth cục bộ (đài Bluetooth). Đây BluetoothAdapter là điểm đầu vào cho tất cả các tương tác

Bluetooth. Bằng cách sử dụng này, bạn có thể khám phá các thiết bị Bluetooth khác, truy vấn danh sách các thiết bị được liên kết (được ghép nối), khởi tạo BluetoothDevice bằng địa chỉ MAC đã biết và tạo một BluetoothSocket để nghe liên lạc từ các thiết bị khác.

BluetoothDevice: Đại diện cho một thiết bị Bluetooth từ xa. Sử dụng quyền này để yêu cầu kết nối với một thiết bị từ xa thông qua một BluetoothSocket truy vấn thông tin về thiết bị như tên, địa chỉ, lớp và trạng thái liên kết.

BluetoothAdapter.LeScanCallback: áo cáo gọi lại một thiết bị BLE được tìm thấy trong quá trình quét thiết bị do **BluetoothAdapter.startLeScan** khởi xướng.

ParcelUuid: Lớp này là một trình bao bọc UUID có thể thay thế được, xung quanh nó là một đại diện bất biến của một mã định danh duy nhất phổ quát 128-bit.

BluetoothManager: Người quản lý cấp cao được sử dụng để lấy một phiên bản của một BluetoothAdapter và để tiến hành quản lý Bluetooth tổng thể.

PackageManager: Lớp để truy xuất các loại thông tin liên quan đến các gói ứng dụng hiện đang được cài đặt trên thiết bị.

UUID: Một lớp đại diện cho một định danh duy nhất toàn cầu không thể thay đổi (UUID). UUID đại diện cho một giá trị 128 bit.

1.5.2. Tìm hiểu về cơ sở dữ liệu SQLite trong Android Studio.



Hình 12. Cơ sở dữ liệu SQLite

1.5.2.1. Tổng quan

SQLite là hệ quản trị cơ sở dữ liệu (DBMS) quan hệ tương tự như Mysql, ... Đặc điểm nổi bật của SQLite so với các DBMS khác là gọn, nhẹ, đơn giản, đặt biệt không cần mô hình Server - Client, không cần cài đặt, cấu hình hay khởi động nên không có khái niệm user, password hay quyền hạn trong SQLite Database. Dữ liệu cũng được lưu ở một file duy nhất.

SQLite thường không được sử dụng với các hệ thống lớn nhưng với những hệ thống ở quy mô vừa và nhỏ thì SQLite không thua các DBMS khác về chức năng hay tốc độ. Vì không cần cài đặt hay cấu hình nên SQLite được sử dụng nhiều trong việc phát triển, thử nghiệm ... vì tránh được những rắc rối trong quá trình cài đặt.

SQLiteOpenHelper là một lớp tích hợp sẵn của gói android.database.sqlite.SQLiteDatabase. Nó là một lớp trợ giúp để quản lý việc tạo cơ sở dữ liệu SQLite và quản lý phiên bản. Lớp trợ giúp quản lý việc tạo cơ sở dữ liệu, xử lý các thao tác với cơ sở dữ liệu và cả quản lý phiên bản. Chúng ta cần tạo một lớp con mở rộng từ lớp SQLiteOpenHelper cho các thao tác với cơ sở dữ liệu. Trong lớp này, chúng ta sẽ thực hiện hai phương thức ghi đè onCreate () và onUpgrade (). Các lớp này đảm nhận việc mở cơ sở dữ liệu nếu nó tồn tại, tạo nó nếu nó không tồn tại và nâng cấp nó khi cần thiết.

1.5.2.2. Class Interface

- Một số phương thức từ lớp SQLiteOpenHelper:

+Phương thức onCreate()

onCreate () được gọi khi cơ sở dữ liệu được tạo lần đầu tiên. Nó chỉ được gọi một lần trong toàn bộ vòng đời ứng dụng. Nó sẽ được gọi bất cứ khi nào có lệnh gọi đầu tiên đến hàm getReadableDatabase () hoặc getWritableDatabase (). Các hàm này có sẵn trong lớp siêu SQLiteOpenHelper.

+ Phương thức onUpgrade()

Phương thức onUpgrade() được gọi khi cơ sở dữ liệu cần được nâng cấp. Nó được gọi khi tệp cơ sở dữ liệu đã tồn tại và chúng tôi muốn nâng cấp phiên bản của nó.

- Một số câu lệnh truy vấn và tạo Database cơ bản:

CREATE: Tạo một bảng mới, một View của một bảng hoặc một đối tượng khác trong Database.

ALTER: Sửa đổi một số đối tượng cơ sở dữ liệu đang tồn tại.

DROP: Xóa cả một bảng, một View của một bảng hoặc đối tượng khác trong Database.

INSERT: Tạo một bản ghi.

UPDATE: Sửa đổi các bản ghi.

DELETE: Xóa các bản ghi.

SELECT: Lấy các bản ghi cụ thể từ một hoặc nhiều bản.

1.6. Thư viện BlueZ

1.6.1. Tổng quan

BlueZ là ngăn xếp Bluetooth chính thức của Linux. BlueZ cung cấp các module, hỗ trợ cho các giao thức và lớp Bluetooth cốt lõi.

1.6.2. Tính năng, đặc điểm

BlueZ linh hoạt, hiệu quả và sử dụng một module để triển khai dự án..., BlueZ cung cấp tính năng thú vị:

- + An toàn xử lý đa đối xứng.
- + Xử lý dữ liệu đa luồng.
- + Hỗ trợ nhiều thiết bị Bluetooth.
- + Giao diện Socket tiêu chuẩn cho tất cả các class.
- + Hỗ trợ bảo mật thiết bị và dịch vụ.

1.6.3. API BlueZ Advertisement

array{string} ServiceUUIDs: Danh sách các UUID để đưa vào trường “Service UUID” của dữ liệu quảng cáo.

array{string} SolicitUUIDs: Mảng UUID để đưa vào “Service Solicitation” dữ liệu quảng cáo.

array{string} Includes: Danh sách các tính năng được đưa vào quảng cáo gói tin.

string LocalName: Tên của thiết bị do mình đặt tên sẽ được sử dụng trong quảng cáo.

unit_16_t Timeout: Thời gian chờ quảng cáo.

unit32 MinInterval [Experimental]: Khoảng thời gian quảng cáo tối thiểu được sử dụng bởi bộ quảng cáo [20ms – 10,485 s].

unit32 MaxInterval [Experimental]: Khoảng thời gian quảng cáo tối đa được sử dụng bởi bộ quảng cáo [20ms – 10,485 s].

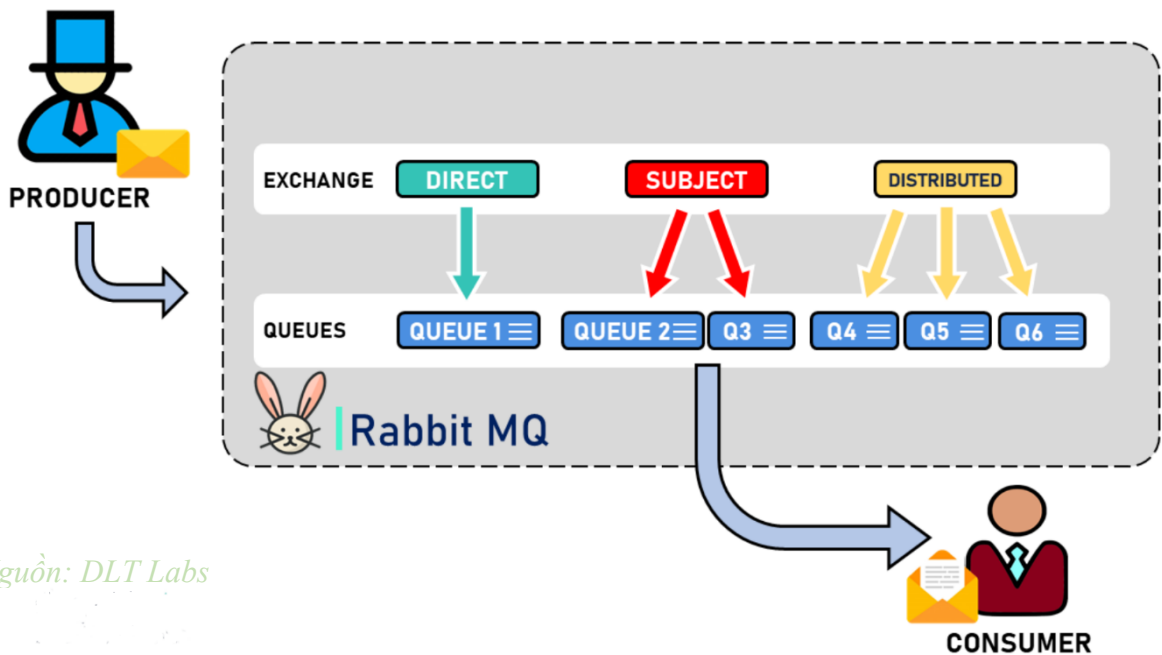
int16 TxPower [Experimental]: Công suất truyền yêu cầu của bộ quảng cáo, giá trị trong khoảng [-127 đến +20] dBm.

Dict ServiceData: Các yếu tố service bao gồm: UUID để liên kết đến lớp dữ liệu.

dict Data [Experimental]: AD Type để đưa vào quảng cáo dữ liệu. Key là loại quảng cáo có giá trị dữ liệu dưới dạng byte array.

dict ManufacturerData: Các trường dữ liệu của nhà sản xuất cần đưa vào dữ liệu quảng cáo. Key là ID của UUID để kết hợp với dữ liệu.

1.7. RabbitMQ



Hình 13. Mô hình RabbitMQ

1.7.1. Tổng quan

Rabbitmq là một AMQP message broker hay còn gọi là phần mềm quản lý hàng đợi message. Nói đơn giản, đây là phần mềm định nghĩa hàng đợi một ứng dụng khác có thể kết nối để bỏ message vào và gửi message dựa trên nó.

Message broker là một chương trình trung gian được thiết kế để validating, transforming và routing message. Chúng phục vụ các nhu cầu giao tiếp giữa các ứng dụng với nhau.

Với Message broker, ứng dụng nguồn (producer) gửi một message đến một server process mà nó có thể cung cấp việc sắp xếp dữ liệu, routing(định tuyến), message translation, persistence và delivery tất cả các điểm đến thích hợp(consumer).

1.7.2. Thư viện Pika

1.7.2.1 IO and Event Looping

Vì AMQP là một giao thức RPC hai chiều nơi máy khách có thể gửi yêu cầu đến máy chủ và máy chủ có thể gửi yêu cầu đến máy khách, Pika triển khai hoặc mở rộng các vòng IO trong mỗi bộ điều hợp kết nối không đồng bộ của nó. Các vòng lặp IO này là các phương pháp chặn vòng lặp và lắng nghe các sự kiện. Mỗi bộ điều hợp không đồng bộ tuân theo cùng một tiêu chuẩn để gọi vòng lặp IO. Vòng lặp IO được tạo khi bộ điều hợp kết nối được tạo.

1.7.2.2. Continuation-Passing Style

Giao diện không đồng bộ với Pika được thực hiện bằng cách chuyển các phương thức gọi lại khi một sự kiện nhất định hoàn tất. Ví dụ: nếu định khai báo một hàng đợi, thì truyền vào một phương thức sẽ được gọi khi máy chủ RabbitMQ trả về một phản hồi `Queue.DeclareOk`.

1.7.2.3. TCP Backpressure

Kể từ RabbitMQ 2.0, Channel.Flow phía client đã bị xóa. Thay vào đó, broker RabbitMQ sử dụng TCP Backpressure để làm chậm ứng dụng client nếu nó gửi thư quá nhanh. Nếu đã chuyển **backpressure_detection** vào các thông số kết nối, Pika sẽ cố gắng giúp xử lý tình huống này bằng cách cung cấp một cơ chế thông báo nếu Pika nhận thấy có quá nhiều khung hình chưa được phân phối. Bằng cách đăng ký một hàm gọi lại với **add_backpressure_callback** phương thức của bất kỳ bộ điều hợp kết nối nào, thì chương trình sẽ được gọi khi Pika thấy rằng tồn đọng gấp 10 lần kích thước khung hình trung bình mà server đang gửi đã bị vượt quá. Có thể điều chỉnh giá trị hệ số thông báo bằng cách gọi **set_backpressure_multiplier** phương thức chuyển bất kỳ giá trị số nguyên nào.

1.7.2.4. Class và Module của Pika

class pika.adapters.blocking_connection.BlockingConnection(): tạo ra một lớp trên core không đồng bộ của Pika cung cấp các phương thức sẽ chặn cho đến khi phản hồi dự kiến được trở lại.

ConnectionParameters(): Dùng để chỉ định tất cả các tham số kết nối cần thiết để kết nối với RabbitMQ, ConnectionParameters cung cấp các thuộc tính để điều chỉnh mỗi tùy chọn kết nối có thể có.

class pika.connection.Connection(): Đây là lớp cốt lõi để triển khai đến RabbitMQ. Lớp này không nên được gọi trực tiếp mà phải thông qua việc sử dụng bộ điều hợp như SelectConnection hoặc BlockingConnection.

class pika.channel.Channel(): Channel là phương thức giao tiếp chính để tương tác với RabbitMQ. Không nên trực tiếp gọi tạo đối tượng channel trong chương trình, mà nên tạo channel bằng cách gọi phương thức channel() để kết nối hoạt động.

queue_declare(): Khai báo hàng đợi, phương thức này tạo hoặc kiểm tra một hàng đợi. Khi tạo một hàng đợi mới, client có thể chỉ định các thuộc tính khác nhau để kiểm tra thử hàng đợi và cũng như nội dung và mức độ chia sẻ của hàng đợi.

basic_publish(): Truyền cho phương thức channel, khóa định tuyến và nội dung.

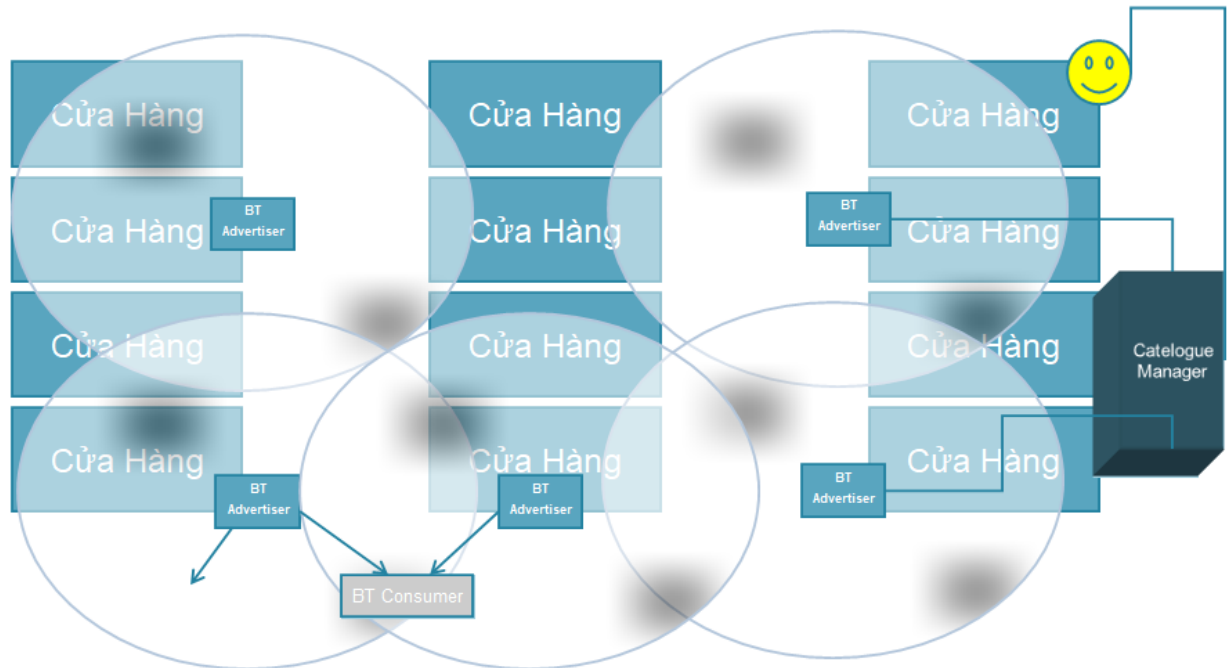
connection.close(): Đóng kết nối.

basic_consume(): Gửi lệnh AMQP 0-9-1 Basic. Consume tới broker và liên kết tin nhắn cho consumer_tag với lệnh consumer callback. Nếu không nhận consumer_tag thì tag sẽ được tạo tự động cho bạn. Sau đó trả tag cho Consumer.

Chương 2. Thiết kế và cài đặt giải pháp

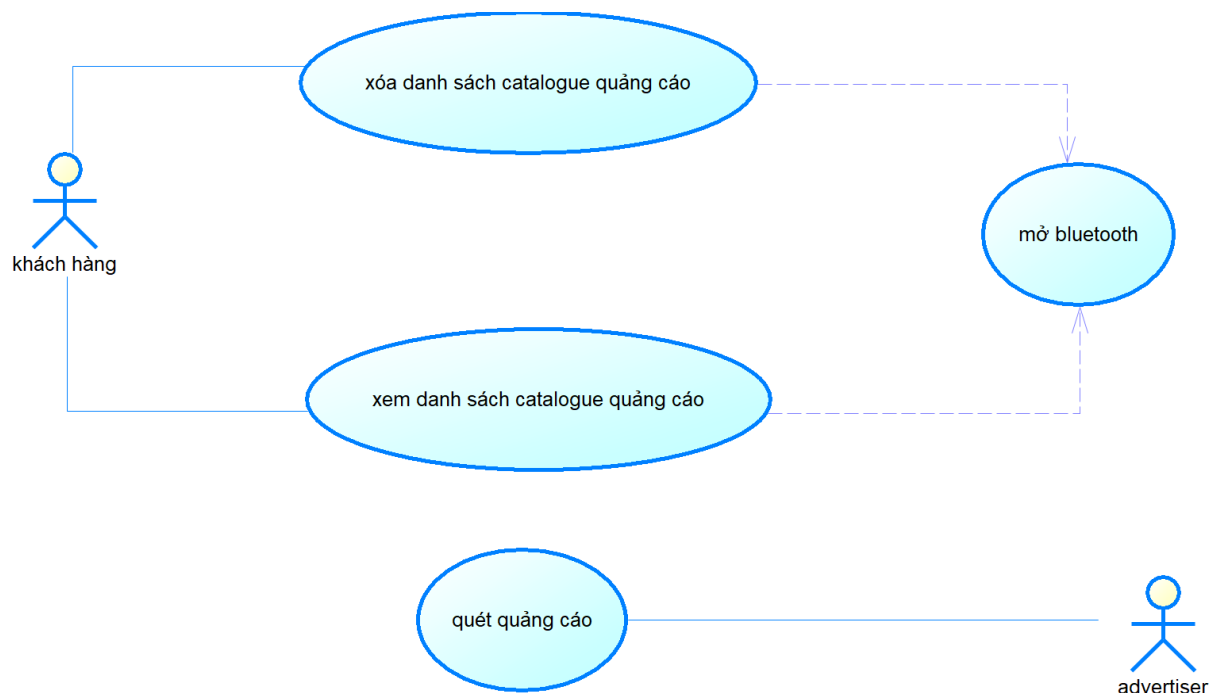
2.1. Kiến trúc hệ thống:

2.1.1. Mô hình tổng thể các thành phần trong hệ thống



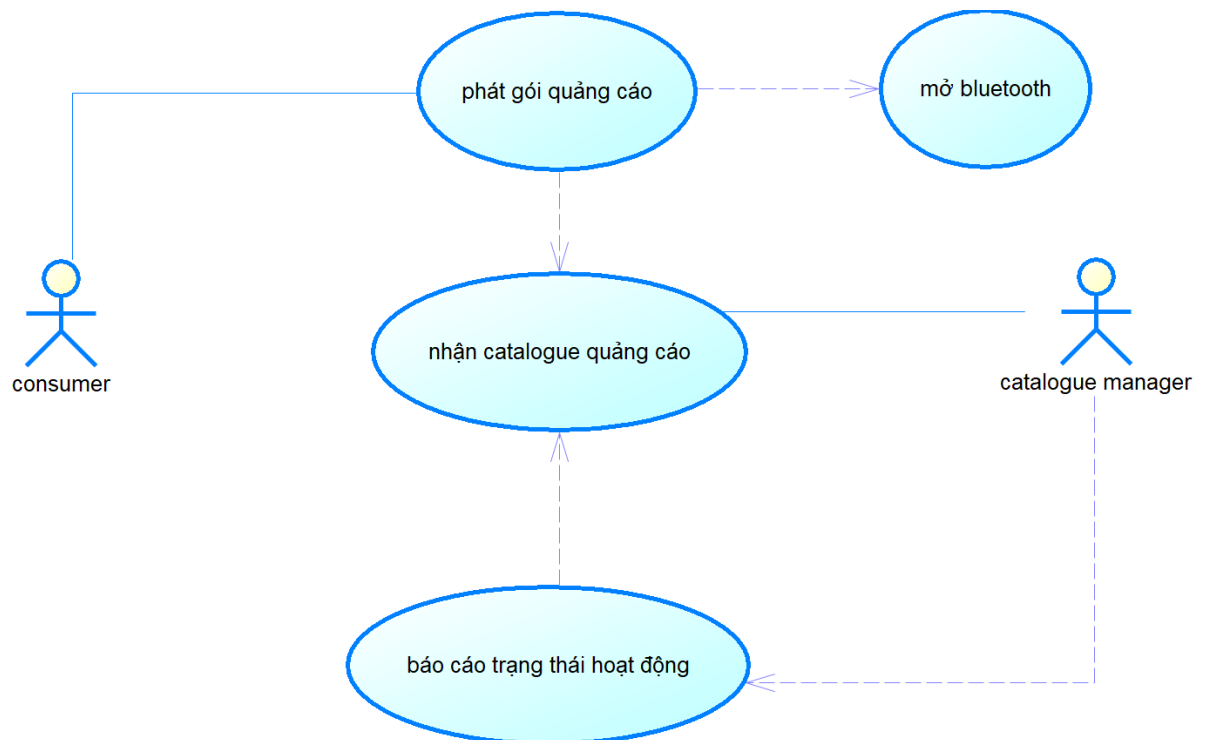
Hình 14. Mô hình hệ thống quảng cáo sản phẩm dựa trên công nghệ Bluetooth

2.1.2. Mô hình Use case Consumer



Hình 15. Mô hình Use case Consumer

2.1.3. Mô hình Use case Advertiser



Hình 16. Mô hình Use case Advertiser

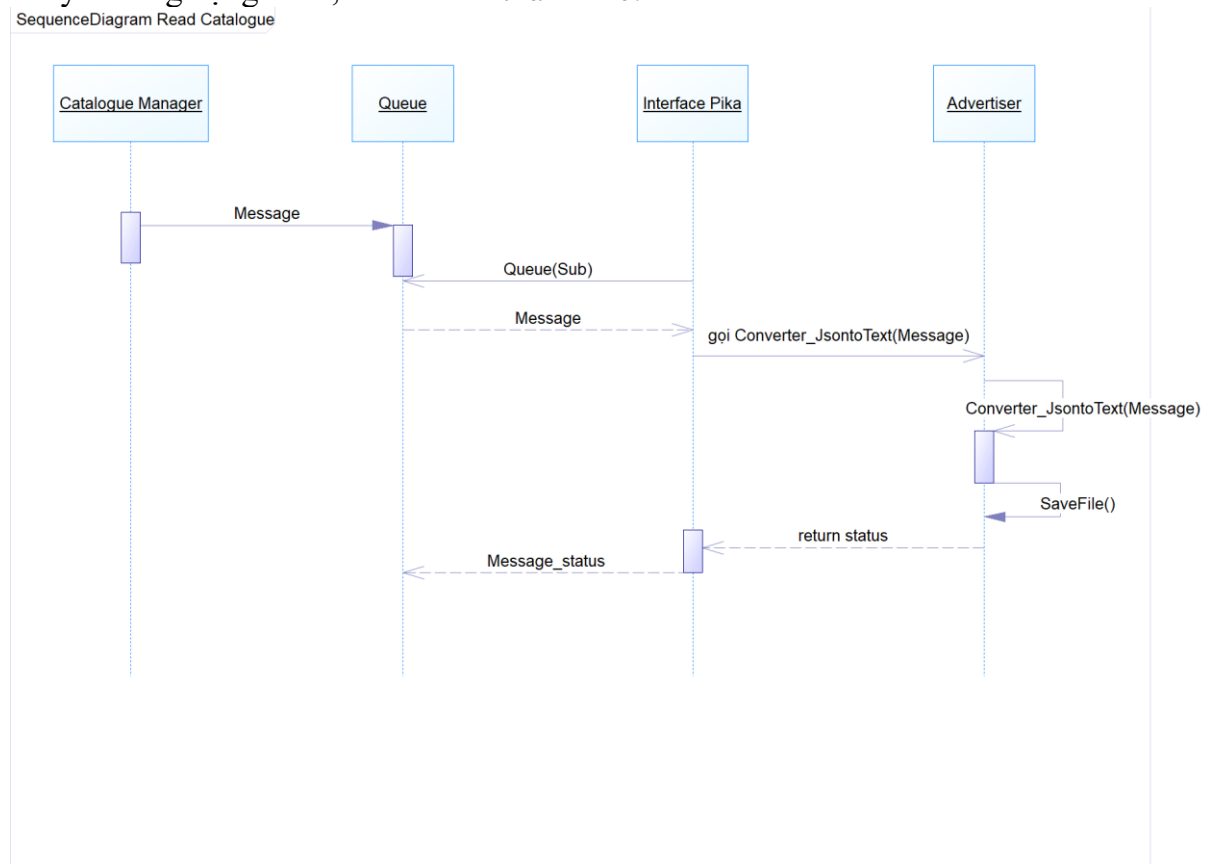
2.1.4. Cơ sở dữ liệu quan hệ SQLite

Tên bảng	Userdetails	
Tên cột	Kiểu dữ liệu	Chú thích
Profile	String	Khóa chính
ID	String	
AdvCount	String	
Off_set	String	
MF	String	
DataAdv	String	
Rssi	String	

Bảng 5. Cấu hình cơ sở dữ liệu quan hệ SQLite

2.1.5. Mô hình Sequence nhận catalogue từ RabbitMQ

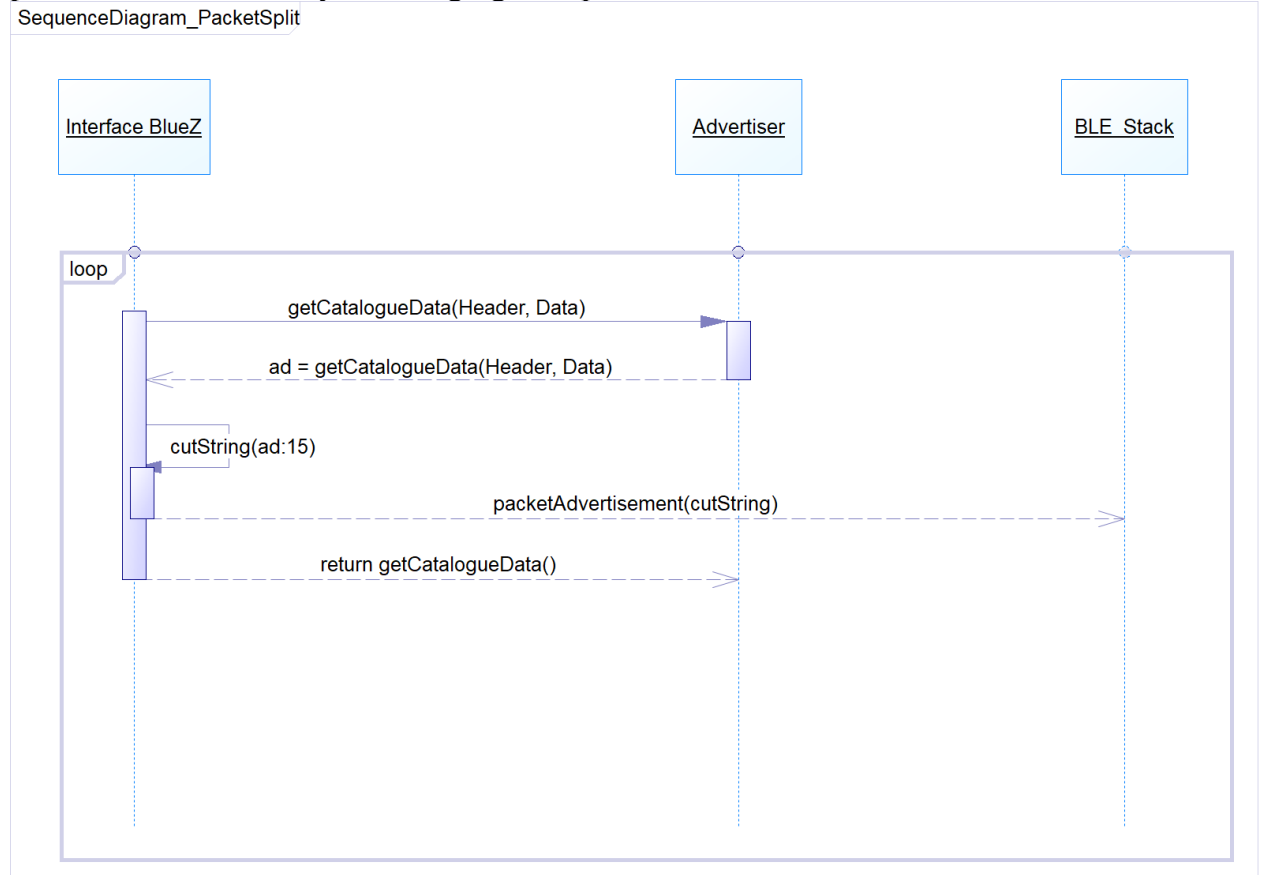
Có chức năng chính sẽ bắt message trên hàng chờ của RabbitMQ, xong mang message về chuyển Json thành dạng Text và đánh Header cho catalogue vừa được chuyển sang dạng Text, sau đó lưu thành file.



Hình 17. Mô hình Sequence nhận catalogue từ RabbitMQ

2.1.6. Mô hình Sequence phân tách dữ liệu quảng cáo và truyền vào ngăn xếp BLE

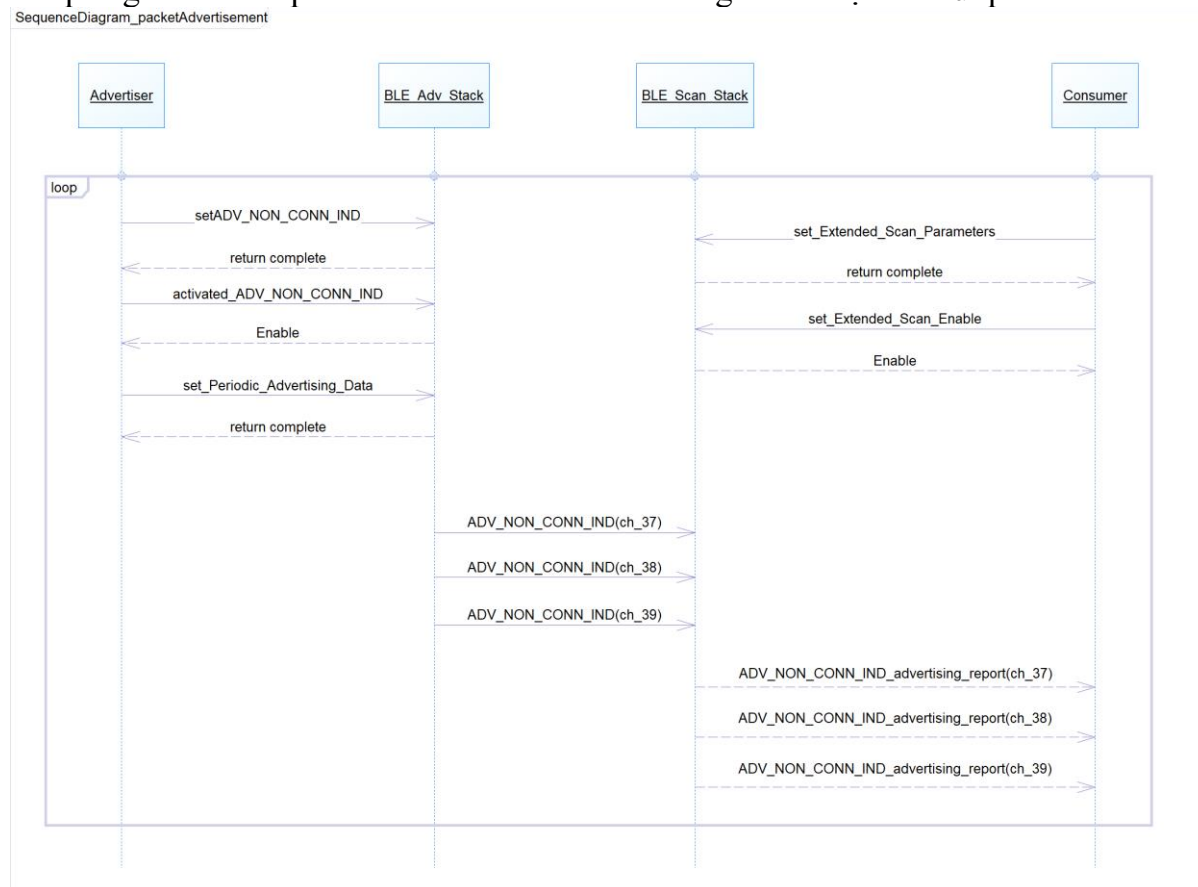
Chức năng chính đọc file catalogue dạng text vào trong chương trình và Header ở đầu dòng, có chức năng phân tách bao nhiêu ký tự và sau đó khi kết thúc chương trình thì Header này sẽ được cập nhật để lần gọi phân tách gói tiếp theo. Sau đó dữ liệu được phân tách sẽ được chuyển xuống ngăn xếp Bluetooth.



Hình 18. Mô hình Sequence phân tách dữ liệu quảng cáo

2.1.7. Mô hình Sequence phát gói tin quảng cáo cho Consumer

Sau khi dữ liệu quảng cáo được phân tách và được đóng thành packet và chờ quảng cáo. Ở dưới ngăn xếp Bluetooth sẽ chuẩn bị xác nhận về AD Type, để phát gói tin quảng cáo. Và ở phía bên kia thì Consumer cũng chuẩn bị về kiểu quét của mình.



Hình 19. Mô hình Sequence phát gói tin quảng cáo cho Consumer

2.1.8. Mô hình Sequence lọc gói tin quảng cáo và ghi dữ liệu vào SQLite

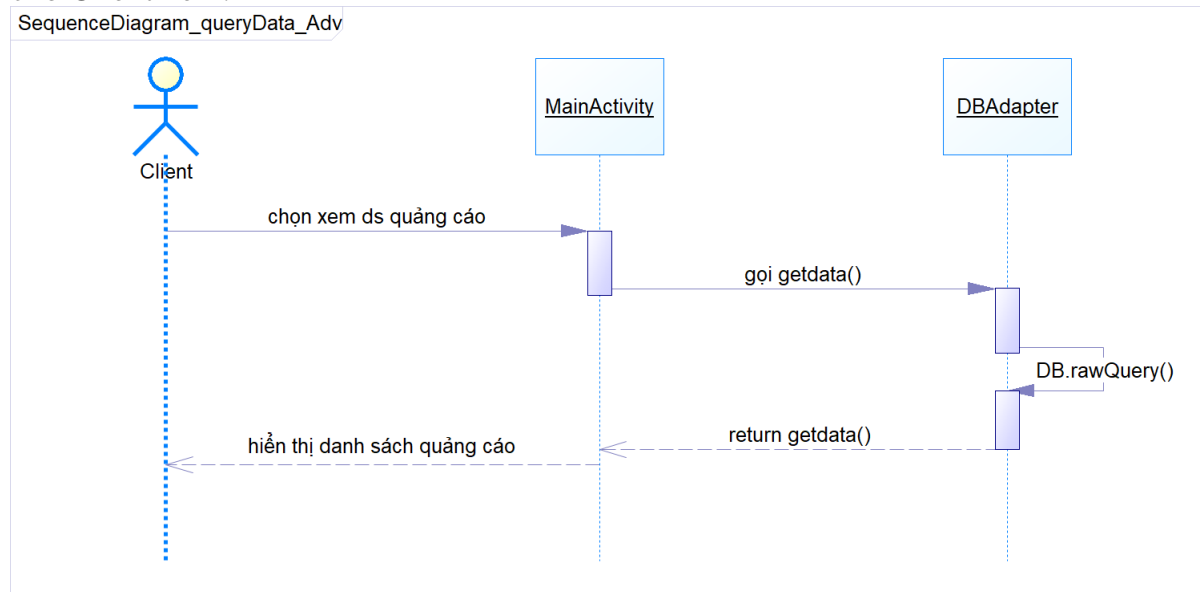
Sau khi class BluetoothAdapter quét và nhận được dữ liệu thì, gửi cho class AdRecord để nhận dạng Flags cũng như các Ad Type để chia chính xác từng byte trong gói tin quảng cáo thành từng thành phần như UUID, MAC Address, rssi , Ad Data. Sau đó đưa về cho MainActivity để đối chiếu xem uuid tại gói tin quảng cáo đó có trùng khớp không, nếu giống nhau thì truy vấn và ghi vào cơ sở dữ liệu.



Hình 20. Mô hình Sequence lọc packet quảng cáo

2.1.9. Mô hình Sequence xem danh sách catalogue quảng cáo

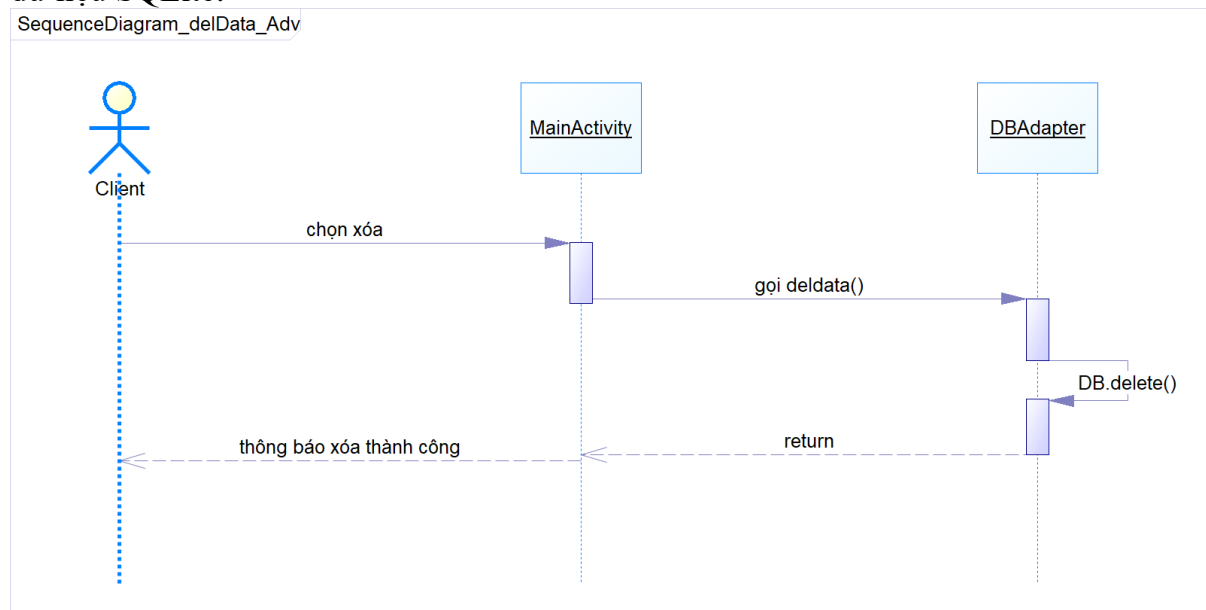
Chức năng chính là truy vấn từ cơ sở dữ liệu SQLite các sản phẩm quảng cáo cho Client xem.



Hình 21. Mô hình Sequence xem danh sách catalogue quảng cáo

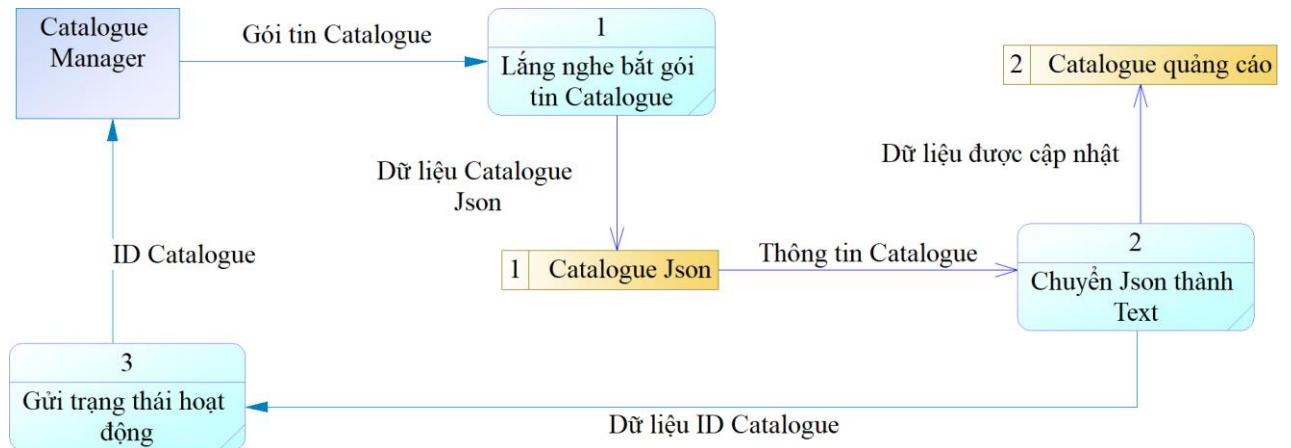
2.1.10. Mô hình Sequence xóa catalogue quảng cáo

Chức năng chính là dùng gọi truy vấn xóa các sản phẩm quảng cáo trong cơ sở dữ liệu SQLite.



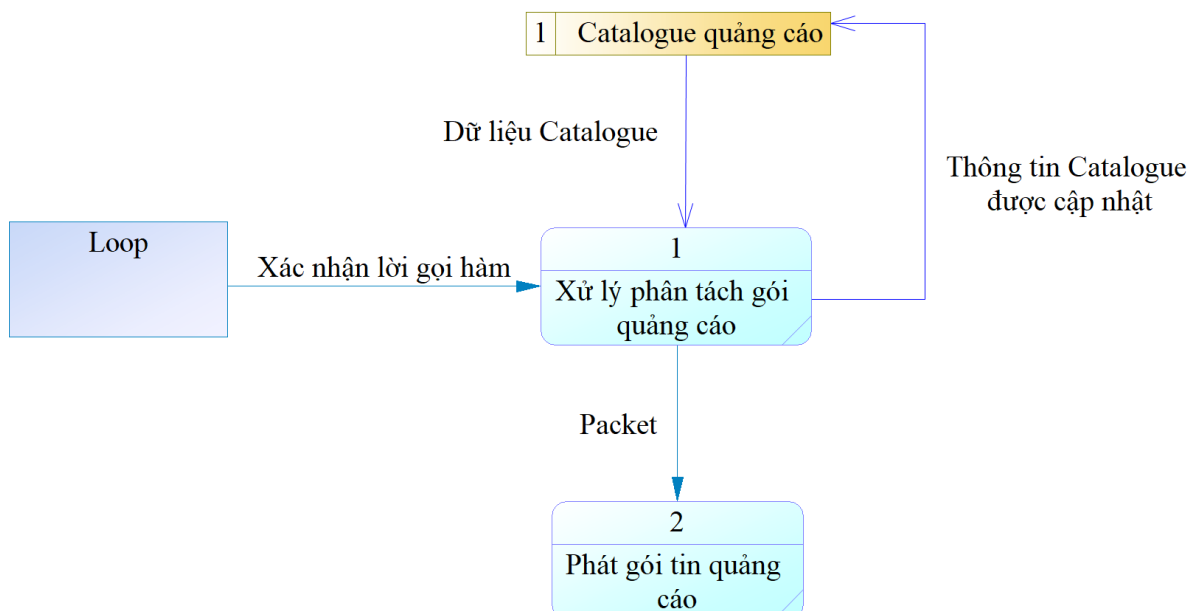
Hình 22. Mô hình Sequence xóa catalogue quảng cáo

2.1.11. Data Flow Diagram nhận message từ RabbitMQ



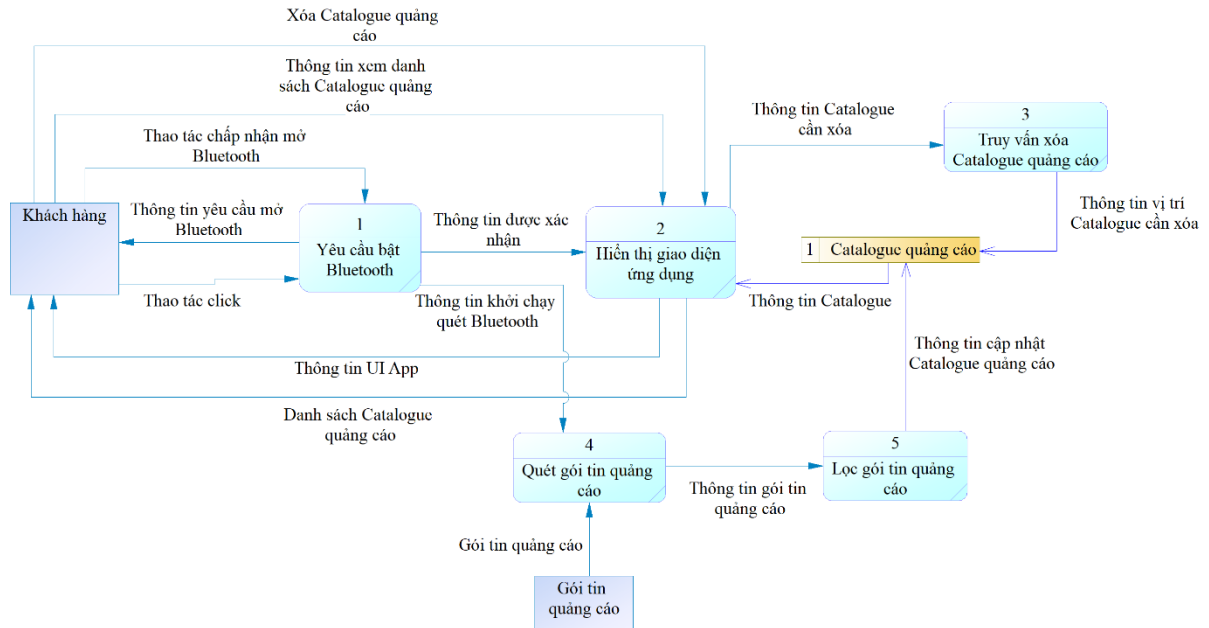
Hình 23. Data Flow Diagram nhận message từ RabbitMQ

2.1.12. Data Flow Diagram Advertiser



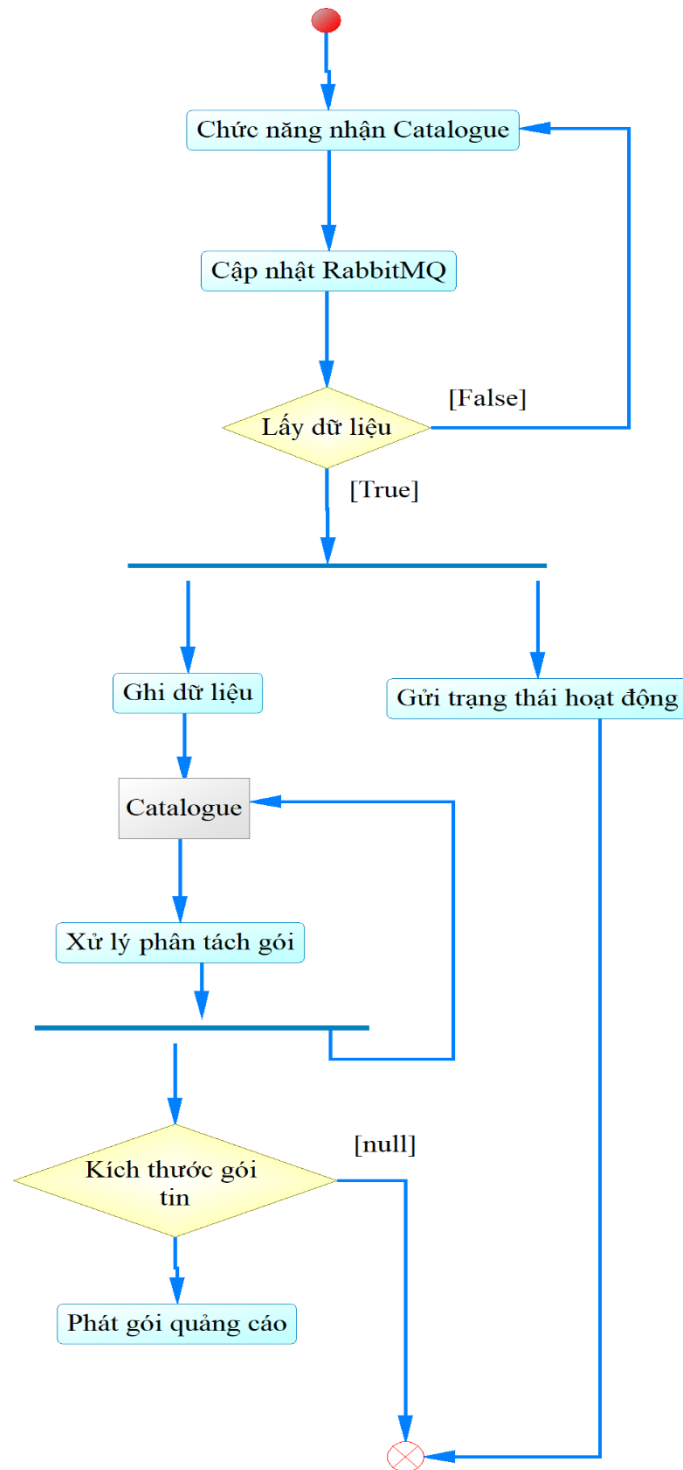
Hình 24. Data Flow Diagram Advertiser

2.1.13. Data Flow Diagram Consumer



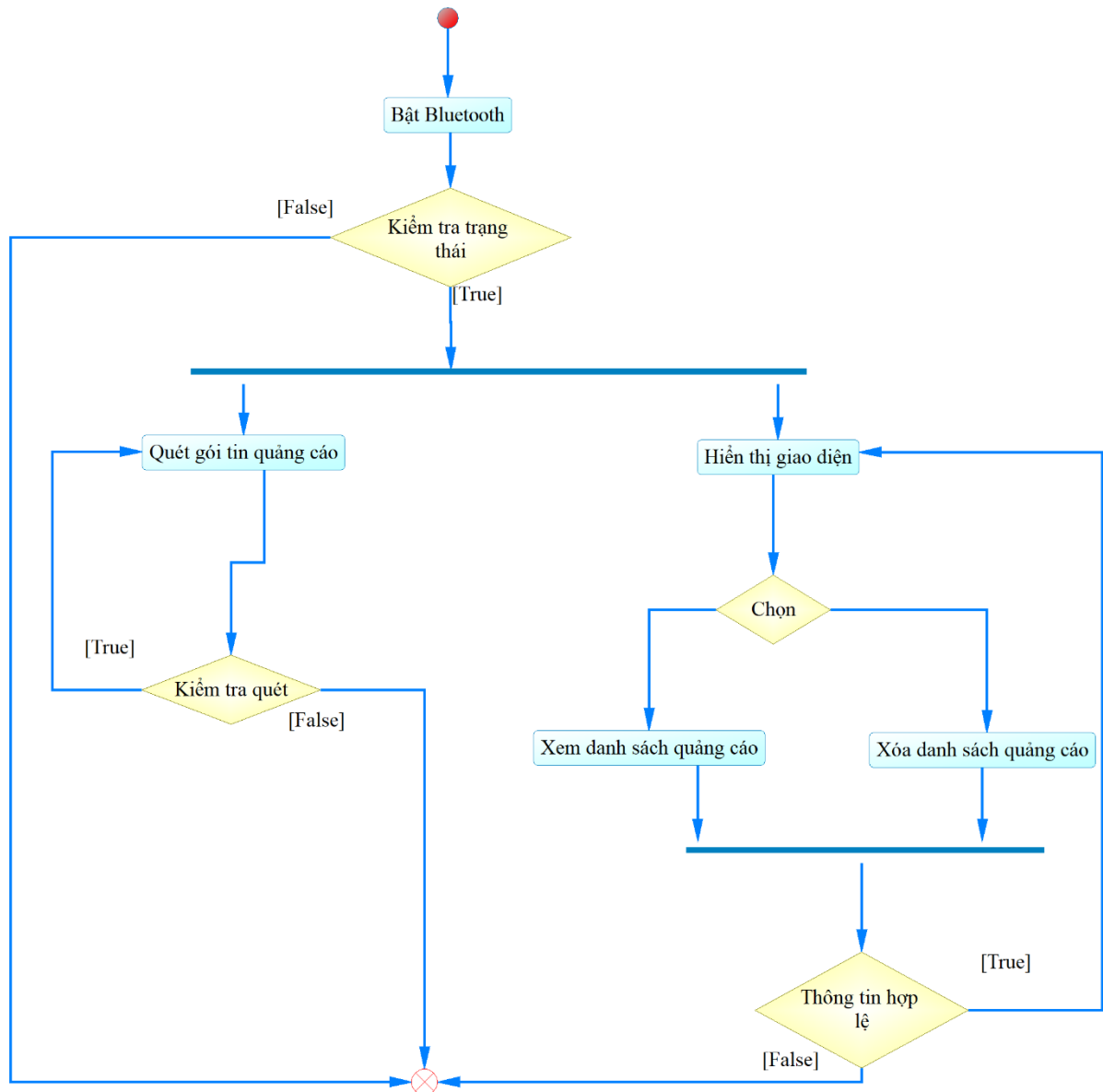
Hình 25. Data Flow Diagram Consumer

2.1.14. Activity Diagram Advertiser



Hình 26. Activity Diagram Advertiser

2.1.15. Activity Diagram Consumer



Hình 27. Diagram Activity Consumer

2.2. Thiết kế giải thuật xử lý

2.2.1. Phương pháp sử dụng các byte làm cờ và các byte độn (Flag byte with byte stuffing)

Phương pháp sử dụng các cờ bắt đầu và kết thúc kết hợp với các byte độn để phân biệt được các điểm bắt đầu và kết thúc của một khung dữ liệu.

Sử dụng các cờ hiệu để xác định vị trí của khung dữ liệu và bắt đầu một khung dữ liệu mới bằng hai byte cờ hiệu liên tiếp.

FLAG	Header	Payload field	Trailer	FLAG
------	--------	---------------	---------	------

Phương pháp định khung cho **local_name** Bluetooth truyền trên Bluetooth dựa trên phương pháp Flag byte with byte stuffing:

+ Sử dụng ký tự đặc biệt “{” để làm cờ bắt đầu (1 byte), và ký tự đặc biệt “}” làm cờ kết thúc (1 byte). Từng catalogue được phân tách nhau bởi các cờ “{“ “}”.

+ Sử dụng các byte độn “#” để sử dụng làm tiền đề phân tách các trường để dựa vào đó phân gói dữ liệu quảng cáo dựa vào giải thuật Maximum Transmission Unit(MTU).

+ Cấu trúc **local_name** có dạng: {ID_Store#DataLen#Offset#MF}.

2.2.2. Trên Raspberry Pi 4B (Advertiser)

- Lập trình phần mềm sử dụng thư viện Pika nhận message trên hàng đợi RabbitMQ về Advertiser (Raspberry).

- Địa chỉ của Server RabbitMQ là 'funnybunny-ui.7perldata.win'. và chờ nhận message trên hàng đợi queue = 'huy'. Sau khi nhận được message thì lưu catalogue quảng cáo về file và tiếp tục lắng nghe tiếp bằng việc gọi lại hàm start consuming().

```
connection = pika.BlockingConnection(pika.ConnectionParameters(host='funnybunny-
ui.7perldata.win'))
channel = connection.channel()
channel.queue_declare(queue='huy')

def callback(ch, method, properties, body):

    str = body.decode('UTF-8')

    print("str: ",str)
    path_w = 'myfile.txt'
    with open(path_w, mode = 'w') as f:
        f.write(str)
        f.close()
    connection.close()
    channel.basic_consume(queue='huy', on_message_callback=callback,
auto_ack=True)
    print('Waiting for messages...')
    channel.start_consuming()
```

- Tiếp theo gửi lại message lên hàng đợi 'status' với nội dung là ID của catalogue mà phía Catalogue Manager vừa gửi, nhằm mục đích kiểm tra xem Advertiser còn hoạt động không.

```
connection = pika.BlockingConnection(
    pika.ConnectionParameters(host='funnybunny-ui.7perldata.win'))
channel = connection.channel()

channel.queue_declare(queue='status')

channel.basic_publish(exchange='', routing_key='status', body=
store.id)
print("Sent status Advertiser!")
connection.close()
```

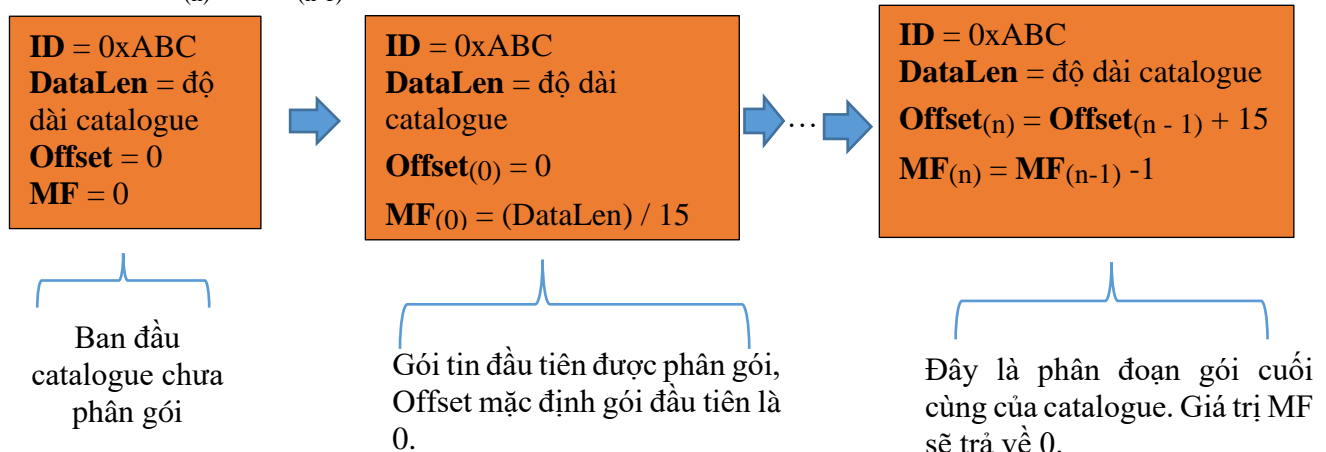
2.2.3. Giải thuật Maximum Transmission Unit(MTU)

+ Áp dụng giải thuật Maximum Transmission Unit(MTU) để chia nhỏ dữ liệu catalogue phù hợp với kích thước gói tin Bluetooth.

$$+ \text{Offset}_{(n)} = \text{Offset}_{(n-1)} + 15$$

MTU = 15 ký tự (kích thước gói tin Bluetooth khả dụng được định dạng Utf-8 mà gói tin có thể nhận).

$$+ \text{MF}_{(n)} = \text{MF}_{(n-1)} - 1$$



- Lập trình dựa trên API BlueZ để phát những gói quảng cáo catalogue dựa trên công nghệ Bluetooth đến Consumer.

+ Để cập nhật lại sau mỗi lần phân gói quảng cáo thì thêm biến HeaderOut để ghi về file catalogue và sau đó lại tiếp tục dựa vào việc phân gói lần trước để cập nhật gói hiện tại.

```
HeaderOpen = Catalogue.find('{')
HeaderClose = Catalogue.find('}')
HeaderIn = Catalogue[HeaderOpen:HeaderClose+1]

lst = []
for pos,char in enumerate(HeaderIn):
    if(char == '#'):
        lst.append(pos)

ID = HeaderIn[1 : lst[0]]

AdvCount = HeaderIn[lst[0] + 1 : lst[1]]

Offset = int(HeaderIn[lst[1] + 1 : lst[2]])

MF = int(HeaderIn[lst[2] + 1 : len(HeaderIn) - 1])

i = 0
AdvLen = 15
split_str = [Catalogue[i : i+AdvLen] for i in range(HeaderClose + 1, len(Catalogue),
AdvLen)]
print('split_str:', split_str[0])

if (MF == 0):
    Catalogue_W = ""
    HeaderOut = ""
else:
    HeaderOut = '{' + ID + '#' + AdvCount + '#' + str(Offset + AdvLen) + '#' + str(MF - 1) + '}'
    Catalogue_W = Catalogue[HeaderClose + 16 : len(Catalogue) + 1]
```

+ Để phát quảng cáo những gói tin sau khi được phân chia gói thì gọi API của BlueZ. UUID-16bit có thể vào Bluetooth SIG để có thể chọn mã định danh UUID phù hợp với AD Type. Tiếp theo add_local_name dùng để đặt tên tạm thời thiết bị Bluetooth. Include_tx_power là thiết đặt công suất phát. Cuối cùng là dữ liệu catalogue được phân gói và chờ phát quảng cáo.

```
self.add_service_uuid('1809')
self.add_local_name(HeaderIn)
self.include_tx_power = True
self.add_service_data('1809', split_str[0].encode('utf-8'))
```

2.2.4. Cấu trúc gói tin phát quảng cáo

Ở giai đoạn này, từng mẫu tin được đóng gói thành 1 packet hoàn chỉnh do Link Layer phụ trách, sau đó chuyển xuống Physical Layer (Stack Bluetooth) phát gói tin quảng cáo.

Từng gói tin Adv Data sẽ được giải thuật MTU phân gói ra từ catalogue, sau đó được cập nhật giá trị Header của từng gói tin nhằm tạo cho mỗi gói tin quảng cáo có private key để tránh việc ghi quá nhiều gói tin quảng cáo trùng nhau vào database của ứng dụng quét quảng cáo (Consumer).

Từng Header được giải thuật MTU cập nhật giá trị(ID; DataLen; Offset; MF) nhưng Header phải tuân theo quy tắc giải thuật Flag byte with byte stuffing.

Mô tả từng phân đoạn quảng cáo được chia nhỏ ra ghi vào trong gói packet như sau:

Dữ liệu catalogue chưa được phân đoạn.

```
{1ABC#256#0#17}Bách hóa xanh Mậu Thân
69A, Xuân Khánh, Ninh Kiều, Cần Thơ
Bắp bò xuất khổ
180000
150000
1
kg
giảm giá 15%
20/11/2021 02:38:18
19/12/2021 22:38:18
Coca-cola Zero
10000
5000
1
chai
giảm còn 15k 1 chai.
20/11/2021 02:38:18
19/11/2021 22:38:18
```

Phân đoạn packet đầu tiên		...	Phân đoạn packet cuối cùng	
Header	Adv Data(15 ký tự)		Header	Adv Data(15 ký tự)
{1ABC#256#0#17}	‘Bách hóa xanh M’		{1ABC#1#255#0}	‘8’

Bảng 6. Phân đoạn gói đầu tiên – gói cuối cùng

Channel	Access Address	Adv PDU Type	Adv PDU Header				Adv Addr	Adv Data Payload
			Type	TxAdd	RxAdd	PDU-Length		
0x25	0x8E89BED6	ADV_NON_CONN_IND	2	0	0	30	0xB0B448CF6A03	02 01 06 03 03 09 18 16 09 7B 31 41 42 43 23 32 35 36 23 30 23 31 37 7D

Hình 28. Cấu trúc quảng cáo được phát với packet Header đầu tiên

Chanel 37 loại quảng cáo Complete local name(0x09) sẽ truyền Header đầu tiên.

AD Structure	Adv Data Pay-load(Hex)	Adv Data Pay-load(String)	Chú thích
1	02		Length
	01		Flags
	06		Type_uuid128_inc
2	03		Length
	03		AD Type = 0x03 ->UUID-16bit
	09		Value = 0x1809
	18		
	16		Length
	09		AD Type = Complete local name
3	7B	{	
	31	'1'	
	41	'A'	
	42	'B'	
	43	'C'	
	23	'#'	
	32	'2'	
	36	'5'	
	36	'6'	
	23	'#'	
	30	'0'	
	23	'#'	
	31	'1'	

	37	‘7’	
	7D	‘}’	

Bảng 7. Quảng cáo PDU payload data Header gói đầu tiên.

Channel	Access Address	Adv PDU Type	Adv PDU Header				Adv Addr	Adv Data Payload
			Type	TxAdd	RxAdd	PDU-Length		
0x26	0x8E89BED6	ADV_NON_CONN_IND	2	0	0	31	0xB0B448CF6A03	02 01 06 03 03 09 18 18 FF 42 C3 A1 63 68 20 68 C3 B3 61 20 78 61 6E 68 20 4D

Hình 29. Cấu trúc quảng cáo được phát với packet Data đầu tiên

Channel 38 loại quảng cáo dành riêng cho nhà sản xuất (0xFF) sẽ truyền Data đầu tiên.

AD Structure	Adv Data Pay-load(Hex)	Adv Data Pay-load(String ASCII converter utf-8)	Chú thích
1	02		Length
	01		Flags
	06		Type_uuid128_inc
2	03		Length
	03		AD Type = 0x03 ->UUID-16bit
	09		Value = 0x1809
	18		
	18		Length
	FF		Dữ liệu riêng do nhà sản xuất (Bluetooth SIG)
3	42	‘B’	
	C3	‘á’	
	A1		
	63	‘c’	
	68	‘h’	
	20	‘ ’	
	68	‘h’	
	C3	‘ó’	

	B3		
	61	‘a’	
	20	‘ ’	
	78	‘x’	
	61	‘a’	
	6E	‘n’	
	68	‘h’	
	20	‘ ’	
	4D	‘M’	

Bảng 8. Quảng cáo PDU payload data với Data gói đầu tiên.

Channel	Access Address	Adv PDU Type	Adv PDU Header				Adv Addr	Adv Data Payload
			Type	TxAdd	RxAdd	PDU-Length		
0x25	0x8E89BED6	ADV_NON_CONN_IND	2	0	0	29	0xB0B448CF6A03	02 01 06 03 03 09 18 15 09 7B 31 41 42 43 23 31 23 32 35 35 23 30 7D

Hình 30. Cấu trúc quảng cáo được phát với packet Header cuối cùng

Channel 37 loại quảng cáo Complete local name(0x09) sẽ truyền Header cuối cùng.

AD Structure	Adv Data Payload(Hex)	Adv Data Payload(String)	Chú thích
1	02		Length
	01		Flags
	06		Type_uuid128_inc
2	03		Length
	03		AD Type = 0x03 ->UUID-16bit
	09		Value = 0x1809
	18		
	15		Length
	09		AD Type = Complete local name
3	7B	‘{‘	
	31	‘1’	
	41	‘A’	

	42	‘B’	
	43	‘C’	
	23	‘#’	
	31	‘2’	
	23	‘5’	
	32	‘6’	
	35	‘#’	
	35	‘0’	
	23	‘#’	
	30	‘1’	
	7D	‘}’	

Bảng 9. Quảng cáo PDU payload data Header gói cuối cùng.

Channel	Access Address	Adv PDU Type	Adv PDU Header				Adv Addr	Adv Data Payload
			Type	TxAdd	RxAdd	PDU-Length		
0x26	0x8E89BED6	ADV_NON_CONN_IND	2	0	0	16	0xB0B448CF6A03	02 01 06 03 03 09 18 02 FF 38

Hình 31. Cấu trúc quảng cáo được phát với packet Data cuối cùng

Channel 38 loại quảng cáo dành riêng cho cho nhà sản xuất (0xFF) sẽ truyền Data cuối cùng.

AD Structure	Adv Data Pay-load(Hex)	Adv Data Pay-load(String ASCII converter utf-8)	Chú thích
1	02		Length
	01		Flags
	06		Type_uuid128_inc
2	03		Length
	03		AD Type = 0x03 ->UUID-16bit
	09		Value = 0x1809
	18		
	02		Length

	FF		Dữ liệu riêng do nhà sản xuất (Bluetooth SIG)
3	38	‘8’	

Bảng 10. Quảng cáo PDU payload data với Data gói cuối cùng.

2.2.5. Trên Android Studio lập trình giao diện mức thấp

2.2.5.1. Quét và lọc gói tin quảng cáo Bluetooth

Trên MainActivity.java

- Đầu tiên kiểm tra người dùng có bật Bluetooth không, nếu chưa thì đòi người dùng kích hoạt Bluetooth lên.

```
if (mBluetoothAdapter == null || !mBluetoothAdapter.isEnabled()) {

    Intent enableBtIntent = new Intent(Bluetooth-
Adapter.ACTION_REQUEST_ENABLE);
    startActivity(enableBtIntent);
    finish();
    return;
}
```

- Tiến hành quét dịch vụ quảng cáo trong vòng 6 giây, những gói tin quảng cáo catalogue có chọn lọc, những địa chỉ UUID-128bit trùng khớp hoặc danh UUID-16bit dạng rút gọn 0x1809.

```
private static final int UUID_SERVICE_THERMOMETER =
0x1809;
public static final ParcelUuid THERM_SERVICE = Par-
celUuid.fromString("00001809-0000-1000-8000-
00805f9b34fb");
```

```
private void startScan() {

    mBluetoothAdapter.startLeScan(new UUID[]
{ THERM_SEVICE.getUuid() }, this);
    setProgressBarIndeterminateVisibility(true);
    // Scan 6s
    mHandler.postDelayed(mStopRunnable, 6000);
}
```

- Sau khi hết thời gian quét thì đưa về ở chế độ dừng quét dịch vụ quảng cáo, và dừng trong vòng 0,1 giây.

```
private void stopScan() {
    mBluetoothAdapter.stopLeScan(this);
    setProgressBarIndeterminateVisibility(false);
    //Delay 0,1s
    mHandler.postDelayed(mStartRunnable, 100);
}
```

- Để duy trì quá trình quét trong vòng 6 giây và tạm dừng 0,1 giây liên tục thì gọi hàm run() để tạo vòng lặp.

```
protected void onPause() {
    super.onPause();
    mHandler.removeCallbacks(mStopRunnable);
    mHandler.removeCallbacks(mStartRunnable);
    mBluetoothAdapter.stopLeScan(this);
}

private Runnable mStopRunnable = new Runnable() {
    @Override
    public void run() {
        stopScan();
    }
};

private Runnable mStartRunnable = new Runnable() {
    @Override
    public void run() {
        startScan();
    }
};
```

- Nhập dữ liệu gói tin quảng cáo sau khi phân tích xong từ class AdRecord.

```
for(AdRecord packet : records) {
    if (packet.getType() == AdRecord.TYPE_SERVICEDATA
        && AdRecord.getServiceDataUuid(packet) ==
        UUID_SERVICE_THERMOMETER) {
        byte[] data = AdRecord.getServiceData(packet);
        assert data != null;
        DataAdv = new String(data);
    }
}
```

- Tổng hợp lại gói tin quảng cáo dựa trên Header(Local_Name) giải thuật MTU phân gói, và phân luồng từng cột để truyền qua class DBAdapter để insert vào cơ sở dữ liệu SQLite.

- Công thức tính khoảng cách từ điện thoại đến thiết bị quảng cáo(Advertiser) dựa vào Rssi của Bluetooth: $10^{((-69 - (Rssi_value)) / (10 * 2))}$.

```
int HeaderOpen = (device.getName()).indexOf("{");
int HeaderClose = (device.getName()).indexOf("}");
char[] HeaderIn = (device.getName()).toCharArray();
int[] list = new int[3];
int index = 0;

for(int i = 0; i < HeaderIn.length; i++){
    if(HeaderIn[i] == '#'){
        list[index++] = i;
    }
}
```

```

        Radius = Math.pow(10, ((-69 - Double.valueOf(rssi)) / (10*2)));
        NumberFormat formatter = new DecimalFormat("##.##");

        ID = (device.getName()).substring(HeaderOpen+1, list[0]);

        AdvCount = (device.getName()).substring(list[0] + 1, list[1]);

        Off_set = (device.getName()).substring(list[1] + 1, list[2]);

        MF = (device.getName()).substring(list[2] + 1, HeaderClose);

        DB.insertdata(device.getName(), ID, AdvCount, Off_set, MF, DataAdv, String.valueOf(formatter.format(Radius)));
    
```

- Hiển thị những catalogue được tổng hợp gói tin đầy đủ ra ngoài màn hình.

```

Cursor res = DB.getdata();

if(res.getCount()==0){
    Toast.makeText(MainActivity.this, "No Entry Exists", Toast.LENGTH_SHORT).show();
    return;
}
StringBuffer buffer = new StringBuffer();
while(res.moveToNext()){
    buffer.append("Profile :"+res.getString(0)+"\n");
    buffer.append("ID :"+res.getString(1)+"\n");
    buffer.append("AdvCount :"+res.getString(2)+"\n\n");
    buffer.append("Off_set :"+res.getString(3)+"\n\n");
    buffer.append("MF :"+res.getString(4)+"\n\n");
    buffer.append("DataAdv :"+res.getString(5)+"\n\n");
    buffer.append("Bán kính (Rssi) :"+res.getString(6)+"\n\n");
}
    
```

Trên AdRecord.java

+ Danh sách một số các AD Type thường gặp.

```

public static final int TYPE_FLAGS = 0x1;
public static final int TYPE_UUID16_INC = 0x2;
public static final int TYPE_UUID16 = 0x3;
public static final int TYPE_UUID32_INC = 0x4;
    
```

```
public static final int TYPE_UUID32 = 0x5;
public static final int TYPE_UUID128_INC = 0x6;
public static final int TYPE_UUID128 = 0x7;
public static final int TYPE_NAME_SHORT = 0x8;
public static final int TYPE_NAME = 0x9;
public static final int TYPE_TRANSMITPOWER = 0xA;
public static final int TYPE_CONNINTERVAL = 0x12;
public static final int TYPE_SERVICEDATA = 0x16;
```

+ Phân tích cú pháp dữ liệu quảng cáo từ các AD Type thường gặp.

```
public static int getServiceDataUuid(AdRecord service-
Data) {
    if (serviceData.mType != TYPE_SERVICEDATA) return -1;

    byte[] raw = serviceData.mData;
    int uuid = (raw[1] & 0xFF) << 8;
    uuid += (raw[0] & 0xFF);

    return uuid;
}
```

+ Trả kết quả sau khi phân tích xong cho class MainActivity.java

```
public static byte[] getServiceData(AdRecord serviceData)
{
    if (serviceData.mType != TYPE_SERVICEDATA) return
null;

    byte[] raw = serviceData.mData;
    return Arrays.copyOfRange(raw, 2, raw.length);
}
```


Trên DBAdapter.java
+ Tạo cơ sở dữ liệu SQLite.

```
public DBAdapter(Context context) {
    super(context, "Catalogue.db", null, 1);
}
```

+ Tạo bảng, khóa chính, các cột trong cơ sở dữ liệu SQLite.

```
public void onCreate(SQLiteDatabase DB) {
    DB.execSQL("create Table Userdetails(Profile TEXT
primary key, ID TEXT, AdvCount TEXT, Off_set TEXT, MF
TEXT, DataAdv TEXT, Rssi TEXT)");
}
```

+ Insert dữ liệu từ class MainActivity gửi đến.

```
public Boolean insertdata(String Profile, String ID,
String AdvCount, String Off_set, String MF, String
DataAdv, String Rssi)
{
    SQLiteDatabase DB = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("Profile", Profile);
    contentValues.put("ID", ID);
    contentValues.put("AdvCount", AdvCount);
    contentValues.put("Off_set", Off_set);
    contentValues.put("MF", MF);
    contentValues.put("DataAdv", DataAdv);
    contentValues.put("Rssi", Rssi);
    long result=DB.insert("Userdetails", null, con-
tentValues);
    if(result==-1){
        return false;
    }else{
        return true;
    }
}
```

+ Lọc các gói tin quảng cáo truyền bị lỗi hoặc truyền bị trùng gói tin, sau đó kiểm tra xem vị trí gói tin đầu tiên với vị trí gói tin tiếp theo có bị sai giá trị của cột MF theo thứ tự giảm dần không. Nếu lỗi xóa hàng đó tại vị trí Profile +1.

```
int SumlistProfile = listProfile.size();
    if(SumlistProfile < 2){
        return;
    }
    else if ((Integer.parseInt(listOff_set.get(0)) == 0) &
        ((Integer.parseInt(listAdvCount.get(0))) / 15) == (Integer.par-
seInt(listMF.get(0)))) {
        int FirstMF = Integer.par-
seInt(listMF.get(0));
```

```

        int SecondMF = Integer.parseInt(listMF.get(1));

        String SecondProfile = listProfile.get(1);

        if(FirstMF <= SecondMF){
            Cursor cursordelDup =
            DB.rawQuery("Select * from Userdetails WHERE Profile = ? ", new String[] {SecondProfile});

            if (cursordelDup.getCount() > 0){
                long resultdel = DB.delete("Userdetails", "Profile=?", new String[]{SecondProfile});
            }
        }

        Cursor cursorinsert = DB.rawQuery("Select * from Userdetails WHERE Profile = ? ", new String[] {FirstProfile});

        contentValues.put("AdvCount", OutAdvCount);
        contentValues.put("Off_set", OutOff_set);
        contentValues.put("MF", OutMF);
        contentValues.put("DataAdv", OutDataAdv);
        contentValues.put("Rssi", OutRssi);
    }
}

```

+ Ngược lại nếu gói tin thỏa điều kiện thì gộp dữ liệu quảng cáo, cập nhật giá trị cột và xóa vị trí Profile + 1 đó đi.

```

else if((FirstMF - 1) == SecondMF){

    String OutAdvCount = String.valueOf( (Integer.parseInt(listAdvCount.get(0)) - (listDataAdv.get(1)).length()) );
    String OutOff_set = "0";
    String OutMF = String.valueOf(Integer.parseInt(listMF.get(0)) - 1);
    String OutDataAdv = listDataAdv.get(0) + listDataAdv.get(1);
    String OutRssi = listRssi.get(1);
    String FirstID = listID.get(0);
    String FirstProfile = listProfile.get(0);
    Cursor cursordel = DB.rawQuery("Select * from Userdetails WHERE Profile = ? ", new String[] {SecondProfile});
}

```

```

        if (cursordel.getCount() > 0) {
            long resultdel = DB.delete("Userdetails", "Profile=?", new
String[]{SecondProfile});
        }
        else{
            Log.i(TAG, "ERROR cursordel....");
        }
    }

```

+ Truy vấn những gói tin quảng cáo bị phân gói cùng một ID catalogue để liên kết lại những gói đó để tổng hợp lại thành một catalogue hoàn chỉnh để gửi đến khách hàng.

```

Cursor cursor = DB.rawQuery("SELECT * FROM Userdetails
WHERE ID = (SELECT ID FROM Userdetails GROUP BY ID ORDER
BY COUNT(*) DESC LIMIT 1) ORDER BY CAST(Off_set AS INTE-
GER) ASC;", null);

```

+ Truy vấn cơ sở dữ liệu những sản phẩm được tổng hợp xong và truyền cho class MainActivity để hiển thị ra màn hình điện thoại.

```

public Cursor getdata ()
{
    SQLiteDatabase DB = this.getWritableDatabase();
    Cursor cursor = DB.rawQuery("Select * from Userde-
tails", null);
    return cursor;
}

```

2.3. Cài đặt giải pháp

2.3.1. Cài đặt và quản trị hệ điều hành Raspbian trên Raspberry Pi

B1. Chuẩn bị thẻ nhớ microSD có dung lượng tối thiểu 8GB và đã được định dạng.

B2. Download hệ điều hành Raspbian về máy tính.

B3. Download phần mềm balenaEtcher giải nén hệ điều hành Raspbian vào thẻ microSD.

B4. Mở terminal gõ lệnh: `sudo raspi -config`, lúc này màn hình hiển thị hộp thoại Raspberry Software Configuration Tool -> Interfacing Options -> SSH để có thể mở port 22 để cho phép truy cập từ xa vào Raspberry Pi bằng dòng lệnh và mở port 5900 để truy cập từ xa bằng giao diện GUI.

B5. Truy cập từ xa ssh từ máy trạm: `ssh pi@IP` của Raspberry và nhập mật khẩu là raspberry (mặc định).

2.3.2. Thiết lập môi trường phát triển ứng dụng và cài đặt thư viện BlueZ

B1. Khi kết nối Raspberry Pi bằng SSH, tiếp gõ lệnh: `sudo apt update` để cập nhật.

B2. Cài đặt thư viện BlueZ trên Raspbian:

```
cd ~  
wget http://www.kernel.org/pub/linux/bluetooth/bluez-5.37.tar.xz  
tar xvf bluez-5.37.tar.xz
```

B3. Kiểm tra xem thông tin Bluetooth: `sudo service bluetooth status`.

B4. Thiết đặt interval cho device bluetooth trên Raspberry Pi ở mức phát 200 ms. Vào các file `adv_min_interval`, `adv_max_interval` tại đường dẫn `cd /sys/kernel/debug/bluetooth/hci0`

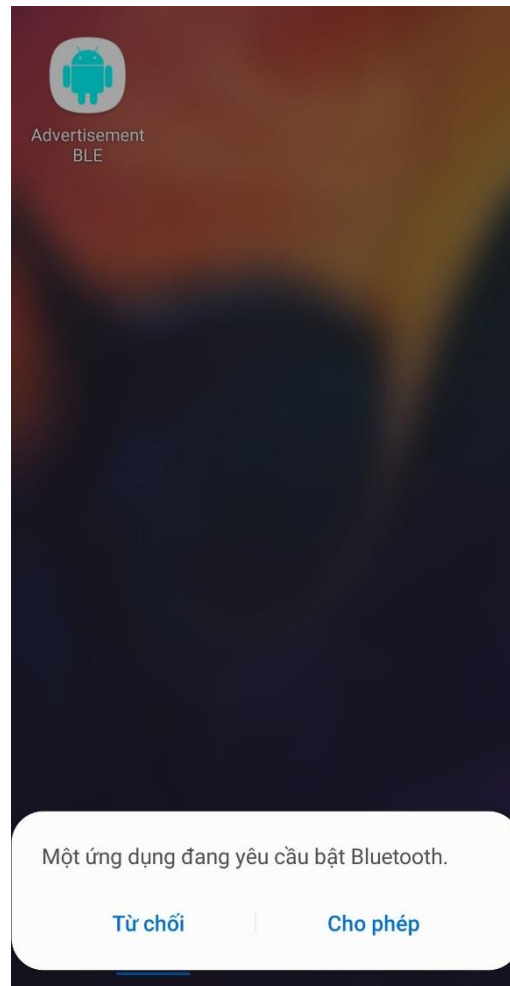
2.3.3. Cấp quyền cho ứng dụng Android

Vào Project dự án của mình `/app/manifests/AndroidManifest.xml`

```
<uses-feature android:name="android.hardware.bluetooth_le" android:required="true"/>  
  
<uses-permission android:name="android.permission.BLUETOOTH"/>  
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>  
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT" />  
  
<uses-permission android:name="android.permission.BLUETOOTH_SCAN" />  
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

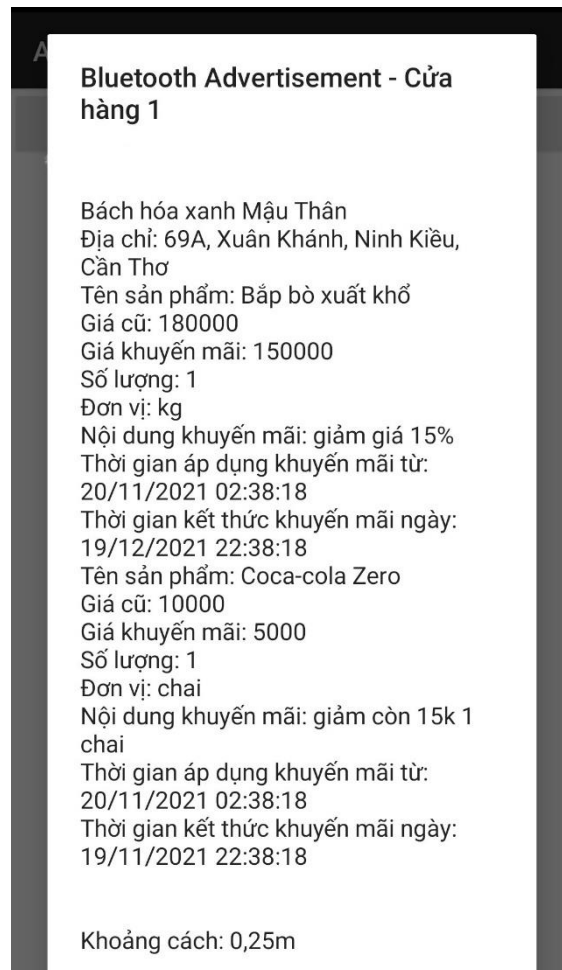
2.3.4. Khởi chạy ứng dụng trên Smartphone

Mở ứng dụng, nếu Bluetooth chưa được bật trên Smartphone thì 1 thông báo hiện ra yêu cầu cho phép bật Bluetooth.



Hình 32. Yêu cầu cho phép ứng dụng bật Bluetooth

Bấm vào “View Data” để xem các catalogue quảng cáo từ các nhãn hàng.



Hình 33. Xem danh sách các sản phẩm quảng cáo

2.3.5. Kiểm tra kết quả

2.3.5.1. Dữ liệu Catalogue Manager truyền qua RabbitMQ

Catalogue Manager (Web Server) gửi catalogue quảng cáo lên trên RabbitMQ dữ liệu dưới dạng json.

```
{
  "store_id": "1",
  "store_name": "Bách hóa xanh Mậu Thân",
  "store_address": "69A, Xuân Khánh, Ninh Kiều, Cần Thơ",
  "catalogs": [
    {
      "cat_id": "1",
      "cat_name": "food",
      "products": [
        {
          "id": "5",
          "name": "Bắp bò xuất khẩu",
          "old_price": "180000",
          "sale_price": "150000",
          "quantity": "1",
          "unit": "kg",
          "saleContent": "giảm giá 15%",
          "saleDateFrom": 1637304039,
          "saleDateTo": 1637822439
        }
      ]
    },
    {
      "cat_id": "2",
      "cat_name": "drink",
      "products": [
        {
          "id": "7",
          "name": "Coca-cola Zero",
          "old_price": "10000",
          "sale_price": "5000",
          "quantity": "1",
          "unit": "chai.",
          "saleContent": "giảm còn 15k 1 chai",
          "saleDateFrom": 1637304039,
          "saleDateTo": 1637822439
        }
      ]
    }
  ]
}
```

2.3.5.2. Chuyển dữ liệu phát gói quảng cáo Bluetooth

Sau khi nhận message trên hàng đợi RabbitMQ về và tự động chuyển đổi dữ liệu json về định dạng file text.

```

Bách hóa xanh Mậu Thân
69A, Xuân Khánh, Ninh Kiều, Cần Thơ
Bắp bò xuất khổ
180000
150000
1
kg
giảm giá 15%
20/11/2021 02:38:18
19/12/2021 22:38:18
Coca-cola Zero
10000
5000
1
chai
giảm còn 15k 1 chai.
20/11/2021 02:38:18
19/11/2021 22:38:18

```

Catalogue quảng cáo được đánh Header để phần mềm phát quảng cáo đọc và phân gói tin truyền để phát quảng cáo Bluetooth cho phù hợp với kích thước gói tin.

```

{1ABC#256#0#17}Bách hóa xanh Mậu Thân
69A, Xuân Khánh, Ninh Kiều, Cần Thơ
Bắp bò xuất khổ
180000
150000
1
kg
giảm giá 15%
20/11/2021 02:38:18
19/12/2021 22:38:18
Coca-cola Zero
10000
5000
1
chai
giảm còn 15k 1 chai.
20/11/2021 02:38:18
19/11/2021 22:38:18

```


2.3.5.2. Database trên SQLite

Trường hợp Smartphone quét quảng cáo Bluetooth và nhận về đầy đủ các gói quảng cáo do Advertiser phát ra.

Database Structure Browse Data Edit Pragmas Execute SQL							
Table: Userdetails		Filter in any column					
Profile	ID	AdvCount	Off_set	MF	DataAdv	Rssi	
Filter	Filter	Filter	Filter	Filter	Filter	Filter	
1 {1ABC#256#0#17}	1ABC	1	0	0	Bách hóa xanh Mậu Thân...	0,32	

Hình 34. Smartphone nhận đầy đủ các gói quảng cáo

Trường hợp Smartphone quét quảng cáo Bluetooth và nhận về không đầy đủ các gói quảng cáo do Advertiser phát ra.

Table: Userdetails							
Filter in any column							
Profile	ID	AdvCount	Off_set	MF	DataAdv	Rssi	
Filter	Filter	Filter	Filter	Filter	Filter	Filter	
1 {1ABC#256#0#17}	1ABC	241	0	16	Bách hóa xanh Mậ...	0,35	
2 {1ABC#256#45#14}	1ABC	256	45	14	Kiều, Cần Thơ...	0,35	
3 {1ABC#256#90#11}	1ABC	256	90	11	...	0,4	
4 {1ABC#256#120#9}	1ABC	256	120	9	:38:18...	0,4	
5 {1ABC#256#135#8}	1ABC	256	135	8	21 22:38:18...	0,35	
6 {1ABC#256#150#7}	1ABC	256	150	7	a-cola Zero...	0,32	
7 {1ABC#256#180#5}	1ABC	256	180	5	giảm còn 15k 1	0,28	
8 {1ABC#256#210#3}	1ABC	256	210	3	1 02:38:18...	0,25	

Hình 35. Smartphone không nhận đầy đủ các gói quảng cáo

Chương 3. Kiểm thử và đánh giá

3.1. Kiểm thử

Trên Advertiser (Raspberry Pi):

Trên hệ thống phát dữ liệu quảng cáo(Raspberry Pi) có hai chương trình chạy cùng lúc, nhằm lấy dữ liệu catalogue từ trên RabbitMQ về cho chương trình thứ nhất chuyển dữ liệu json thành file text và trả kết quả đã nhận thành công file json về cho Catalogue Manager, cả hai chương trình đều hoạt động tốt không bị xung đột, đọc và ghi dữ liệu chính xác không xảy ra tình trạng thất thoát dữ liệu, thời gian lấy message catalogue và gửi message phản hồi về đều hoàn thành trong chu kỳ 3 giây không bị trễ. Đồng thời chương trình thứ hai sẽ đọc dữ liệu từ file text lên và phân đoạn đánh Header, để khi đưa dữ liệu vào Bluetooth Stack không bị vượt qua kích thước của gói tin quảng cáo, vì thế không xảy ra tình trạng gói tin quảng cáo.

Trên Consumer(Phần mềm quét quảng cáo):

Phần mềm quét các gói quảng cáo khoảng 6 giây và 0,1 giây để ngừng quét hoạt động đúng theo yêu cầu, quá trình bắt gói quảng cáo để lọc UUID chính xác

và giúp giảm thời gian quét quảng cáo cho các dịch vụ UUID khác. Quá trình ghi, cập nhật và xóa trong cơ sở dữ liệu SQLite không xảy ra truy vấn lỗi.

Thời gian để một khách hàng nhận đủ các gói tin quảng cáo Bluetooth.

Mẫu tin quảng cáo	
ID	1ABC
DataLen	428
Offset	0
MF	28

Chu kỳ phát 1 Catalogue	Advertisement Interval	Thời gian phát 1 gói quảng cáo	Số gói tin quảng cáo nhận được	Thời gian nhận các gói tin của 1 chu kỳ phát Catalogue	Tổng số gói tin được phân đoạn
Lần 1	32ms	1s	4 packet	28s	28 packet
Lần 2			7 packet		
Lần 3			8 packet		
Lần 4			11 packet		
Lần 5			11 packet		
Lần 6			14 packet		
Lần 7			15 packet		
Lần 8			16 packet		
Lần 9			19 packet		
Lần 10			23 packet		
Lần 11			23 packet		
Lần 12			27 packet		
Lần 13			28 packet		
Tổng thời gian để khách hàng nhận đủ một catalogue: $28 * 13 = 364$ giây					
Lần 1	300ms	2s	11 packet	56s	28 packet
Lần 2			17 packet		
Lần 3			24 packet		
Lần 4			28 packet		
Tổng thời gian để khách hàng nhận đủ một catalogue: $56 * 4 = 224$ giây					
Lần 1	200ms	2s	21	56s	28 packet
Lần 2			28		
Tổng thời gian để khách hàng nhận đủ một catalogue: $56 * 2 = 112$ giây					
Lần 1	200ms	1s	12	28s	28 packet
Lần 2			17		
Lần 3			20		
Lần 4			26		
Lần 5			28		
Tổng thời gian để khách hàng nhận đủ một catalogue: $28 * 5 = 140$ giây					

Bảng 11. So sánh thời gian chu kỳ tổng hợp gói tin catalogue

3.2. Đánh giá

Mục tiêu cải tiến việc quảng cáo trực tiếp bằng phát tờ rơi và đánh poster thay vào đó bằng việc phát quảng cáo sản phẩm, dịch vụ trực tuyến bằng công nghệ Bluetooth, điều này giúp cho các người bán và người mua dễ dàng tiếp cận hơn. Việc truyền và nhận dữ liệu giữa hai thiết bị Advertiser và Consumer ổn định không gặp vấn đề bất cập cũng như giữa Advertiser và Catalogue Manager.

Có thể biến rất nhiều Raspberry Pi thành một hệ thống phát quảng cáo phủ sóng Bluetooth cho toàn thành phố, khu đô thị, và với tình hình dịch Covid-19 vẫn còn xuất hiện thì việc truyền thông tin đến người dân hoặc những khu vực có mức độ lây nhiễm cao cần hạn chế ra vào.

PHẦN III. KẾT LUẬN

1. Kết quả đạt được

Với việc phát triển “Hệ thống quảng cáo sản phẩm dựa trên công nghệ Bluetooth – Mobile App” giúp người tiêu dùng tiếp cận quảng cáo theo xu hướng mới và các công ty, doanh nghiệp các cửa hàng sẽ được tiếp cận với một phương pháp quảng cáo các sản phẩm, dịch vụ mới.

Cải tiến phương pháp quảng cáo Bluetooth cũ bị giới hạn dữ liệu truyền, thì với cách nhận dữ liệu từ Catalogue Manager về và chia nhỏ thành nhiều gói tin, mỗi lần phát là một gói có dữ liệu khác nhau.

Với việc quảng cáo bằng Bluetooth thì chúng ta loại bỏ sức người vào một việc lặp đi lặp lại nhiều lần phát tờ rơi...

Giúp Marketing sản phẩm, dịch vụ nhanh nhất tới khách hàng mà không bị giới hạn dữ liệu nhập vào, có thể thiết đặt thời gian địa điểm quảng cáo, và khách hàng đầu cuối làm sao họ nhận quảng cáo nhanh nhất và không cần thao tác nhấn hoặc chấp nhận kết nối từ thiết bị quảng cáo Bluetooth.

Cung cấp khả năng lưu trữ cho khách hàng để có thể xem lại những thông tin sản phẩm cũng như có thể lưu lại những mã voucher từ cửa hàng.

2. Hạn chế

Do áp dụng giải thuật Maximum Transmission Unit (MTU) để chia nhỏ dữ liệu catalogue, và cơ chế quảng bá không nối kết giúp cho các Hacker có thể giả mạo các gói tin quảng cáo Bluetooth để truyền cho nạn nhân nhằm thay đổi nội dung quảng cáo.

3. Hướng phát triển

Tìm hiểu sâu hơn về ngăn Stack Bluetooth và các dạng kết nối cũng như xây dựng bảo mật và an toàn hơn cho hệ thống.

Cải tiến sử dụng Bluetooth 5.x cho việc truyền các gói quảng cáo với dung lượng có thể lên tới 1650 byte.

Mở rộng mô hình và nhân rộng lên những trạm phát Advertiser để quảng cáo sản phẩm, dịch vụ phủ sóng khu dân cư, khu đô thị và toàn thành phố.

Dựa vào mạng tầm gần Bluetooth chúng ta có thể kết hợp với nhiều cục phát quảng cáo (Advertiser) thành một mạng ngang hàng (Mesh) để có thể giúp những người dân quanh khu vực phủ sóng có thể tìm kiếm người già hoặc trẻ em thất lạc. Đặc biệt với tình hình dịch Covid-19 hiện nay thì có thể cung cấp cho ta thông tin những khu vực phủ sóng Bluetooth biết những vùng nguy hiểm có khả năng lây bệnh cao.

TÀI LIỆU THAM KHẢO

- [1] TS. Ngô Bá Hùng, Ths. Đoàn Hoà Minh, giáo trình Lập trình thiết bị di động, NXB Đại học Cần Thơ, 2016.
- [2] BLE Advertising Primer:
<https://www.argenox.com/library/bluetooth-low-energy/ble-advertising-primer/>
- [3] Tài liệu tổng quang về Bluetooth LE:
<https://microchipdeveloper.com/wireless:ble-gap-overview>
- [4] Tài liệu tham khảo về thư viện BlueZ:
<https://github.com/bluez/bluez/tree/master/doc>
- [5] Tài liệu tham khảo lập trình Bluetooth trên nền tảng Android:
<https://developer.android.com/reference/classes>
- [6] Tài liệu tham khảo cơ sở dữ liệu quan hệ SQLite:
<https://developer.android.com/reference/android/database/sqlite>