

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA CÔNG NGHỆ THÔNG TIN  
BỘ MÔN MẠNG MÁY TÍNH VÀ VIỄN THÔNG**

**ĐÀO QUÝ THÁI AN - TRẦN THỊ MỸ HẠNH**

**TÌM HIỂU VỀ CÔNG NGHỆ BLUETOOTH  
VÀ VIẾT ÚNG DỤNG MINH HỌA**

**LUẬN VĂN CỬ NHÂN TIN HỌC**

**Tp.HCM, 7/2005**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA CÔNG NGHỆ THÔNG TIN  
BỘ MÔN MẠNG MÁY TÍNH VÀ VIỄN THÔNG**

**ĐÀO QUÝ THÁI AN                    0112421  
TRẦN THỊ MỸ HẠNH                0112345**

**TÌM HIỂU VỀ CÔNG NGHỆ BLUETOOTH  
VÀ VIẾT ỨNG DỤNG MINH HỌA**

**GIÁO VIÊN HƯỚNG DẪN :**

**Thạc sĩ : HUỲNH THỦY BẢO TRÂN**

**NIÊN KHÓA 2001 - 2005**

## **NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

# NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

## LỜI CÁM ƠN

Chúng em xin bày tỏ lòng biết ơn chân thành nhất đến Cô Huỳnh Thụy Bảo Trân, người đã tận tâm hướng dẫn, giúp đỡ chúng em trong suốt thời gian thực hiện luận văn này.

Chúng con xin gửi tất cả lòng biết ơn sâu sắc và sự kính trọng đến ông bà, cha mẹ, cùng toàn thể gia đình, những người đã nuôi dạy chúng con trưởng thành đến ngày hôm nay.

Chúng em cũng xin chân thành cảm ơn quý Thầy cô trong Khoa Công nghệ thông tin, trường Đại học Khoa học Tự nhiên TP.Hồ Chí Minh đã tận tình giảng dạy, hướng dẫn, giúp đỡ và tạo điều kiện cho chúng em thực hiện tốt luận văn này.

Xin chân thành cảm ơn sự giúp đỡ, động viên và chỉ bảo rất nhiệt tình của các anh chị và tất cả các bạn, những người đã giúp chúng tôi có đủ nghị lực và ý chí để hoàn thành luận văn này.

Mặc dù đã cố gắng hết sức, song chắc chắn luận văn không khỏi những thiếu sót. Chúng em rất mong nhận được sự thông cảm và chỉ bảo tận tình của quý Thầy Cô và các bạn.

TP.HCM, 7/2005

Nhóm sinh viên thực hiện  
Đào Quý Thái An – Trần Thị Mỹ Hạnh



# LỜI NÓI ĐẦU

Ngày nay, xã hội phát triển mạnh mẽ, kỹ thuật ngày càng hiện đại nên nhu cầu về trao đổi thông tin, giải trí, nhu cầu về điều khiển thiết bị từ xa,...ngày càng cao. Và những hệ thống dây cáp phức tạp lại không thể đáp ứng tốt nhu cầu này, nhất là ở những khu vực chật hẹp, những nơi xa xôi, trên các phương tiện vận chuyển,...Vì thế công nghệ không dây đã ra đời và đang phát triển mạnh mẽ, tạo rất nhiều thuận lợi cho con người trong đời sống hằng ngày. Kỹ thuật không dây phục vụ rất nhiều nhu cầu khác nhau của con người, từ nhu cầu làm việc, học tập đến các nhu cầu giải trí như chơi game, xem phim, nghe nhạc, v.v...Với các nhu cầu đa dạng và phức tạp đó, kỹ thuật không dây đã đưa ra nhiều chuẩn với các đặc điểm kỹ thuật khác nhau để có thể phù hợp với từng nhu cầu, mục đích và khả năng của người sử dụng như IrDA, WLAN với chuẩn 802.11, ZigBee, OpenAir, UWB, Bluetooth,...

Mỗi chuẩn kỹ thuật đều có những ưu, khuyết điểm riêng của nó, và Bluetooth đang dần nổi lên là kỹ thuật không dây tầm ngắn có nhiều ưu điểm, rất thuận lợi cho những thiết bị di động. Với một tổ chức nghiên cứu đông đảo, hiện đại và số lượng nhà sản xuất hỗ trợ kỹ thuật Bluetooth vào sản phẩm của họ ngày càng tăng, Bluetooth đang dần lan rộng ra khắp thế giới, xâm nhập vào mọi lĩnh vực của thiết bị điện tử và trong tương lai mọi thiết bị điện tử đều có thể được hỗ trợ kỹ thuật này.

Xuất phát từ các lý do trên, chúng em đã thực hiện đề tài “**TÌM HIỂU CÔNG NGHỆ BLUETOOTH VÀ VIẾT ỨNG DỤNG MINH HỌA**”. Trong đề tài này, chúng em tìm hiểu về kỹ thuật không dây Bluetooth và xây dựng một chương trình truyền phonebook qua Bluetooth giữa các điện thoại Nokia sử dụng hệ điều hành Symbian Series 60 với nhau và với máy tính.

Mục tiêu của đề tài là tìm hiểu công nghệ Bluetooth và xây dựng một ứng dụng thông qua Bluetooth của các điện thoại Nokia Series 60 và máy tính để minh họa hoạt động của kỹ thuật này ... Các nội dung chính của đề tài bao gồm:

- Tìm hiểu về hoạt động của kỹ thuật Bluetooth.
- Tìm hiểu vấn đề bảo mật, virus và các cách tấn công vào điện thoại di động thông qua Bluetooth.
- Tìm hiểu về hệ điều hành Symbian và series 60.
- Xây dựng ứng dụng chạy trên điện thoại di động Nokia series 60 có tích hợp Bluetooth để: trao đổi phonebook giữa hai điện thoại di động với nhau, và giữa điện thoại di động và máy tính.

Nội dung của luận văn được chia làm 3 phần và 10 chương:

## **PHẦN I: BLUETOOTH**

**Chương 1. Giới thiệu tổng quan về Bluetooth:** Giới thiệu khái quát về công nghệ Bluetooth như khái niệm, lịch sử phát triển, các đặc điểm và một số ứng dụng hiện nay của Bluetooth.

**Chương 2. Kỹ thuật Bluetooth:** Mô tả chi tiết các kỹ thuật Bluetooth như: các khái niệm dùng trong công nghệ, sóng radio trong Bluetooth, tầng giao thức, đặc điểm kỹ thuật của Bluetooth và so sánh Bluetooth với một vài công nghệ không dây khác.

**Chương 3. Vấn đề về an toàn và bảo mật trong Bluetooth:** Phân tích các vấn đề về an toàn bảo mật, hacking, virus, và các giải pháp bảo mật trong Bluetooth.

**Chương 4. Các ưu nhược điểm và tương lai của Bluetooth:** Trình bày về các ưu khuyết điểm của Bluetooth và tương lai của công nghệ này.

## **PHẦN II: HỆ ĐIỀU HÀNH SYMBIAN**

**Chương 5. Tổng quan về hệ điều hành Symbian và thế hệ Series 60:** Giới thiệu tổng quan về hệ điều hành Symbian cũng như kiến trúc hệ thống của nó. Giới thiệu Series 60, một platform trên các điện thoại di động thông minh của hãng Nokia dùng Symbian, lập trình ứng dụng trên Symbian và lập trình C++ cho Symbian.

**Chương 6. Lập trình C++ trên Symbian :** Trình bày một số vấn đề về lập trình C++ trên Symbian.

**Chương 7. Bluetooth và Symbian : Lập trình sử dụng giao tiếp Bluetooth trên Symbian với C++:** các vấn đề về lập trình giao tiếp Bluetooth.

### **PHẦN III: XÂY DỰNG ỨNG DỤNG MINH HỌA SỬ DỤNG CÔNG NGHỆ BLUETOOTH**

**Chương 8. Phân tích và thiết kế ứng dụng trao đổi phonebook qua Bluetooth:** Phân tích và thiết kế chương trình ứng dụng phonebook.

**Chương 9. Cài đặt và thử nghiệm:** tiến hành cài đặt và thử nghiệm ứng dụng.

**Chương 10. Tổng kết.**

KHOA CNTT

## MỤC LỤC

<b>Phần 1 LÝ THUYẾT VỀ BLUETOOTH .....</b>	<b>11</b>
<b>Chương 1 GIỚI THIỆU TỔNG QUAN VỀ BLUETOOTH.....</b>	<b>12</b>
1.1. Khái niệm Bluetooth .....	12
1.2. Lịch sử, hình thành và phát triển của Bluetooth .....	12
1.2.1. Lịch sử tên Bluetooth:.....	12
1.2.2. Hình thành và phát triển của Bluetooth:.....	12
1.3. Các đặc điểm của Bluetooth. ....	14
1.4. Ứng dụng của Bluetooth. ....	15
1.4.1. Thiết bị thông minh.....	15
1.4.2. Thiết bị truyền thanh.....	16
1.4.3. Thiết bị truyền dữ liệu.....	17
1.4.4. Các ứng dụng nhúng. ....	18
1.4.5. Một số ứng dụng khác.....	20
<b>Chương 2 KỸ THUẬT BLUETOOTH .....</b>	<b>21</b>
2.1. Các khái niệm dùng trong công nghệ Bluetooth.....	21
2.1.1. Master Unit : .....	21
2.1.1. Slaver Unit : .....	21
2.1.2. Piconet: .....	22
2.1.3. Scatternet: .....	23
2.1.4. Kết nối theo kiểu ad hoc: .....	25
2.1.5. Định nghĩa các liên kết vật lý trong Bluetooth: .....	26
2.1.6. Trạng thái của thiết bị Bluetooth: .....	26
2.1.7. Các chế độ kết nối:.....	27
2.2. Bluetooth Radio. ....	27
2.2.1. Ad Hoc Radio Connectivity.....	27
2.2.2. Kiến trúc của hệ thống Bluetooth Radio.....	28
2.2.2.1. Radio Spectrum-Dãy sóng vô tuyến: .....	28
2.2.2.2. Interference Immunity – Sự chống nhiễu: .....	29
2.2.2.3. Multiple Access Scheme_Phối hợp đa truy cập: .....	30
2.3. Kĩ thuật trai phô nhảy tần trong công nghệ Bluetooth.....	32
2.3.1. Khái niệm trai phô trong công nghệ không dây :.....	32
2.3.2. Kĩ thuật nhảy tần số trong công nghệ Bluetooth : .....	32

2.4. Cách thức hoạt động của Bluetooth .....	35
2.4.1. Cơ chế truyền và sửa lỗi : .....	35
2.4.2. Quá trình hình thành Piconet .....	36
2.4.3. Quá trình hình thành Scatternet .....	38
2.5. Các tầng giao thức trong Bluetooth. ....	39
2.5.1. Bluetooth Radio: .....	40
2.5.2. BaseBand: .....	42
2.5.2.1. Network topology .....	42
2.5.2.2. Liên kết SCO và ACL.....	44
2.5.2.3. Địa chỉ thiết bị.....	44
2.5.2.4. Định dạng gói tin .....	45
2.5.2.5. Quản lý trạng thái.....	45
2.5.2.6. Thiết lập kết nối .....	46
2.5.2.7. Các chế độ kết nối:.....	47
2.5.2.8. Những chức năng khác của Baseband .....	47
2.5.3. Link Manager Protocol: .....	48
2.5.4. Host Controller Interface: .....	48
2.5.4.1. Những thành phần chức năng của HCI.....	48
2.5.4.2. Các lệnh HCI.....	50
2.5.4.3. Các sự kiện, mã lỗi, luồng dữ liệu HCI .....	50
2.5.4.4. Host Controller Transport Layer.....	51
2.5.5. Logical link control and adaption protocol (L2CAP): .....	51
2.5.5.1. Những yêu cầu chức năng của L2CAP .....	51
2.5.5.2. Những đặc điểm khác của L2CAP.....	52
2.5.6. RFCOMM Protocol: .....	53
2.5.7. Service Discovery Protocol: .....	54
2.5.7.1. Thiết lập giao thức SDP .....	54
2.5.7.2. Các dịch vụ SDP .....	55
2.5.7.3. Tìm kiếm dịch vụ .....	55
2.5.7.4. Data element .....	56
2.6. Bluetooth Profiles: .....	57
2.6.1. 4 profile tổng quát trong đặc tả Bluetooth v1.1: .....	59
2.6.2. Model-Oriented Profiles .....	60
2.6.3. Một số Profiles khác. ....	62

2.7. Vấn đề sử dụng năng lượng trong Bluetooth.....	64
2.7.1. Giới thiệu.....	64
2.7.2. Việc sử dụng và quản lý năng lượng trong công nghệ Bluetooth.....	65
2.7.2.1. Tổng quan: .....	65
2.7.2.2. Các chế độ năng lượng.....	66
2.8. So sánh Bluetooth với các kỹ thuật không dây khác : Hồng ngoại, Wi-fi (802.11b wireless).....	71
2.8.1. So sánh Bluetooth với Wi-Fi .....	71
2.8.2. So sánh Bluetooth với IrDA: .....	74
<b>Chương 3 VẤN ĐỀ AN TOÀN VÀ BẢO MẬT TRONG BLUETOOTH..</b>	<b>77</b>
3.1. Sơ lược về vấn đề bảo mật trong các chuẩn không dây.....	77
3.1.1. Sơ lược chuẩn bảo mật mạng không dây trong 802.11.....	77
3.1.2. Chuẩn bảo mật WEP trong IEEE 802.11.....	77
3.1.3. Những vấn đề nảy sinh trong an ninh mạng không dây .....	79
3.2. Qui trình bảo mật trong Bluetooth : .....	81
3.2.1. An toàn bảo mật trong Bluetooth:.....	81
3.2.1.1. Phần mô tả về an toàn bảo mật: .....	82
3.2.1.2. Nhìn sơ về bảo mật Bluetooth: .....	84
3.2.2. Hacking:.....	94
3.2.2.1. Impersonation attack by inserting/replacing data .....	94
3.2.2.2. Bluejacking .....	94
3.2.2.3. Bluetooth Wardriving .....	95
3.2.2.4. Nokia 6310i Bluetooth OBEX Message DoS .....	96
3.2.2.5. Brute-Force attack.....	96
3.2.2.6. Denial-of-Service attack on the device .....	97
3.2.2.7. Disclosure of keys.....	97
3.2.2.8. Unit key attacks.....	98
3.2.2.9. Backdoor attack .....	98
3.2.2.10. Pairing attack .....	98
3.2.2.11. BlueStumbling = BlueSnarfing.....	99
3.2.2.12. BlueBug attack.....	100
3.2.2.13. PSM Scanning.....	100
3.2.2.14. On-line PIN cracking .....	100

3.2.2.15. A man-in-the-middle attack using Bluetooth in a WLAN interworking environment.....	100
3.2.2.16. Off-line encryption key (via Kc).....	101
3.2.2.17. Attack on the Bluetooth Key Stream Generator .....	101
3.2.2.18. Replay attacks .....	101
3.2.2.19. Man-in-the-middle attack.....	101
3.2.2.20. Denial-of-Service attack on the Bluetooth network.....	101
3.2.3. Virus:.....	102
3.2.3.1. Appdisabler.B .....	102
3.2.3.2. Cabir.Dropper .....	104
3.2.3.3. Cabir – A.....	106
3.2.3.4. Cabir – B .....	107
3.2.3.5. Cabir.Y.....	109
3.2.3.6. Commwarrior.A .....	109
3.2.3.7. Dampig.A.....	112
3.2.3.8. Doomboot.A.....	113
3.2.3.9. Drever – A.....	114
3.2.3.10. Drever – C .....	115
3.2.3.11. Fontal.A .....	116
3.2.3.12. Hobbes.A .....	117
3.2.3.13. Lasco.A .....	119
3.2.3.14. Locknut – B.....	121
3.2.3.15. Mabir.A .....	121
3.2.3.16. MGDroppe.R .....	123
3.2.3.17. Mosquito Trojan.....	125
3.2.3.18. Skulls – A .....	126
3.2.3.19. Skulls- B.....	128
3.3. Các giải pháp an toàn bảo mật khi sử dụng công nghệ mạng Bluetooth. ....	129
3.3.1. Những mẹo an toàn cho thiết bị Bluetooth: .....	129
3.3.2. Phòng chống virus trên mobile phone?.....	129
<b>Chương 4 CÁC ƯU NHƯỢC ĐIỂM VÀ TƯƠNG LAI CỦA BLUETOOTH. ....</b>	<b>131</b>
4.1. Ưu điểm .....	131

4.2. Khuyết điểm.....	131
4.3. Tầm ứng dụng và tương lai của Bluetooth.....	132
4.3.1. Các phiên bản kỹ thuật của Bluetooth: .....	132
4.3.2. Những ứng dụng Bluetooth: .....	136
<b>Phần 2 HỆ ĐIỀU HÀNH SYMBIAN .....</b>	<b>141</b>
<b>Chương 5 TỔNG QUAN VỀ HỆ ĐIỀU HÀNH SYMBIAN VÀ THẾ HỆ SERIES 60 .....</b>	<b>142</b>
5.1. Khái niệm về hệ điều hành Symbian. ....	142
5.2. Lịch sử phát triển. ....	143
5.3. Kiến trúc Tổng quan của hệ điều hành Symbian .....	146
5.3.1. Nhân hệ điều hành - Kernel .....	147
5.3.2. Middleware .....	148
5.3.3. Application Engine .....	148
5.3.4. User Interface framework .....	148
5.3.5. Kỹ thuật đồng bộ - Synchronization technology .....	148
5.3.6. Java virtual machine implementation.....	149
5.4. Giới thiệu về thế hệ Series 60 .....	149
5.5. Lập trình ứng dụng cho Symbian.....	151
5.5.1. Các ngôn ngữ lập trình.....	151
5.5.2. Các bộ công cụ phát triển ứng dụng – SDK (Software Development Kit) và các môi trường phát triển tích hợp – IDE (Integrated Development Enviroment) cho lập trình C++.....	152
<b>Chương 6 LẬP TRÌNH C++ TRÊN SYMBIAN. ....</b>	<b>154</b>
6.1. Các kiểu dữ liệu cơ bản.....	154
6.2. Kiểu dữ liệu chuỗi và descriptor trên Symbian. ....	155
6.3. Các qui ước trong lập trình Symbian C++.....	160
6.3.1. Qui ước về đặt tên lớp.....	160
6.3.2. Qui ước đặt tên dữ liệu : .....	160
6.3.3. Qui ước đặt tên hàm: .....	161
6.4. Quản lý lỗi trên Symbian. ....	162
6.4.1. Cơ chế bắt lỗi trên Symbian.....	162
6.4.2. Hàm Leave.....	163
6.5. Một số vấn đề về quản lý bộ nhớ trong lập trình Symbian C++ :.....	164

6.5.1. Cơ chế Cleanup Stack .....	164
6.5.2. Khởi tạo 2 pha (Two - phase constructor) .....	166
6.5.3. Khởi tạo đối tượng với NewL() và NewLC() .....	168
<b>Chương 7 BLUETOOTH VÀ SYMBIAN: LẬP TRÌNH SỬ DỤNG</b>	
<b>GIAO TIẾP BLUETOOTH TRÊN SYMBIAN VỚI C++.....</b>	<b>170</b>
7.1. Giới thiệu .....	170
7.1.1. Các ứng dụng Bluetooth trên các thiết bị sử dụng hệ điều hành Symbian: .....	170
7.1.2. Các công cụ phát triển và ví dụ:.....	170
7.2. Tổng quan về Bluetooth API: .....	171
7.2.1. Các nhóm hàm Bluetooth API: .....	172
7.2.2. Quan hệ giữa các nhóm hàm API: .....	173
7.3. Một vài kiểu dữ liệu Bluetooth thông dụng .....	174
7.4. Bluetooth Sockets. ....	176
7.4.1. Mở và cấu hình Bluetooth Socket :.....	176
7.4.2. Xây dựng Bluetooth Socket Server : Lắng nghe và chấp nhận kết nối từ thiết bị là Client : .....	178
7.4.3. Xây dựng Bluetooth Socket Client : Tìm kiếm và kết nối tới thiết bị là Server. ....	181
7.4.3.1. Chọn thiết bị để kết nối tới : .....	181
7.4.3.2. Truy vấn thông tin về thiết bị xung quanh:.....	181
7.4.3.3. Truy vấn về dịch vụ được cung cấp trên thiết bị Server : .....	184
7.4.3.4. Kết nối với thiết bị đã được chọn và thực hiện trao đổi dữ liệu: 184	
7.4.4. Trao đổi dữ liệu thông qua Bluetooth socket :.....	186
7.5. Bluetooth Service Discovery Database:.....	187
7.5.1. Kết nối vào Bluetooth Service Discovery Database : .....	187
7.5.2. Đăng ký một dịch vụ vào Service Database : .....	188
7.5.3. Thiết lập các thuộc tính trong một Service Record: .....	190
7.6. Bluetooth Service Discovery Agent: .....	191
7.6.1. Truy vấn các dịch vụ trên thiết bị khác với Bluetooth Service Discovery Agent: .....	192
7.6.2. Tìm kiếm các thuộc tính dịch vụ: .....	193
7.6.3. Tạo ra đối tượng để quản lý các kết quả truy vấn:.....	194

7.7. Bluetooth security manager: .....	195
7.7.1. Tổng quan .....	195
7.7.2. Kết nối vào Bluetooth Security Manager.....	196
7.7.3. Thiết lập các chế độ bảo mật : .....	197
7.8. Bluetooth Device Selection UI. ....	198
7.9. Xây dựng ứng dụng Bluetooth trên Symbian OS với Series 60 SDK .....	201
7.9.1. Sự khác nhau về Bluetooth trên thiết bị ảo và thiết bị thật. ....	201
7.9.2. Các yêu cầu về phần cứng và phần mềm cho việc phát triển ứng dụng Bluetooth với Series 60 SDK :.....	202
7.9.3. Cài đặt và cấu hình thiết bị USB Bluetooth.....	203
<b>Phần 3 XÂY DỰNG ỨNG DỤNG MINH HỌA SỬ DỤNG CÔNG NGHỆ BLUETOOTH .....</b>	<b>205</b>
<b>Chương 8 PHÂN TÍCH VÀ THIẾT KẾ ỨNG DỤNG TRAO ĐỔI PHONEBOOK.....</b>	<b>206</b>
8.1. Giới thiệu .....	206
8.2. Phân tích và xác định yêu cầu.....	206
8.3. Qui trình kết nối và gửi nhận dữ liệu.....	207
8.4. Xây dựng phần ứng dụng trên điện thoại.....	209
8.4.1. Phần Server.....	211
8.4.2. Phần Client.....	214
8.5. Xây dựng phần ứng dụng PbkExchange trên máy tính .....	218
8.5.1. Kết nối vào cổng COM : .....	218
8.5.2. Quảng bá dịch vụ .....	219
8.5.3. Chấp nhận kết nối .....	219
8.5.4. Thực hiện truyền và nhận dữ liệu : .....	219
<b>Chương 9 CÀI ĐẶT VÀ THỬ NGHIỆM .....</b>	<b>221</b>
9.1. Cài đặt: .....	221
9.2. Thử nghiệm.....	221
<b>Chương 10 TỔNG KẾT.....</b>	<b>222</b>
<b>PHỤ LỤC A : Một số thuật ngữ sử dụng trong luận văn .....</b>	<b>223</b>
<b>PHỤ LỤC B : Hướng dẫn sử dụng chương trình PbkExchange .....</b>	<b>227</b>
1.     Sử dụng ứng dụng PbkExchange trên điện thoại : .....	227
2.     Sử dụng ứng dụng PbkExchange trên máy tính : .....	232

<b>PHỤ LỤC C : Xây dựng ứng dụng HelloWorld trên Symbian với Series 60 SDK v1.2 .....</b>	<b>236</b>
1. Cài đặt các chương trình cần thiết : .....	236
2. Tạo Project .....	236
3. Cấu trúc thư mục của ứng dụng HelloWorld .....	238
4. Mở một project đã có : .....	239
5. Xây dựng và biên dịch ứng dụng .....	239
6. Tạo file cài đặt cho ứng dụng HelloWorld: .....	240
7. Cài đặt ứng dụng trên thiết bị thật: .....	243
<b>Tài liệu tham khảo .....</b>	<b>243</b>

## Danh sách các hình

Hình 1-1 Tai nghe Bluetooth .....	16
Hình 1-2 Thiết bị truyền dữ liệu .....	17
Hình 1-3 USB Bluetooth.....	17
Hình 1-4 Máy ảnh điều khiển bằng điện thoại di động.....	18
Hình 1-5 Màn hình hiển thị theo giao diện dành cho các cuộc điện thoại.....	19
Hình 1-6 Bluetooth Car Kit .....	19
Hình 1-7 Máy chụp hình kỹ thuật số có hỗ trợ Bluetooth để truyền hình ảnh .....	19
Hình 1-8 Đồng hồ có hỗ trợ Bluetooth để nghe nhạc mp3 .....	20
Hình 2-1 Một Piconet trong thực tế .....	22
Hình 2-2 Piconet gồm 1 Slave .....	23
Hình 2-3 Piconet gồm nhiều Slave .....	23
Hình 2-4 Một Scatternet gồm 2 Piconet .....	24
Hình 2-5 Sự hình thành một Scatternet theo cách 1 .....	24
Hình 2-6 Sự hình thành một Scatternet theo cách 2 .....	25
Hình 2-7 Kĩ thuật trai phỏ nhảy tần số.....	32
Hình 2-8 Các Packet truyền trên các tần số khác nhau .....	33
Hình 2-9 Các Packet truyền trên khe thời gian .....	33
Hình 2-10 Cấu trúc gói tin Bluetooth .....	34
Hình 2-11 Access code .....	34
Hình 2-12 Cấu tạo một packet .....	35
Hình 2-13 Mô hình piconet .....	36
Hình 2-14 Quá trình truy vấn tạo kết nối .....	37
Hình 2-15 Truy vấn tạo kết nối giữa các thiết bị trong thực tế .....	38
Hình 2-16 Minh họa một Scatternet .....	39
Hình 2-17 Bluetooth Protocol Stack .....	40
Hình 2-18 Các tầng nghi thức Bluetooth .....	40
Hình 2-19 Frequency hopping .....	41
Hình 2-20 Piconet .....	43
Hình 2-21 Scatternet .....	44
Hình 2-22 Định dạng gói tin Bluetooth .....	45
Hình 2-23 Host Controller Interface .....	49

Hình 2-24 Host controller transport layer .....	50
Hình 2-25 Bluetooth v1.1 profiles .....	59
Hình 2-26 TCS profile .....	60
Hình 2-27 Networking Profiles .....	61
Hình 2-28 Headset Profile .....	61
Hình 2-29 LAN Access .....	61
Hình 2-30 File Transfer Profile .....	62
Hình 2-31 Object Push Profile .....	62
Hình 2-32 Hands Free Profile .....	64
Hình 2-33 Human Interface Device Profile .....	64
Hình 2-34 Hold Mode Interaction .....	68
Hình 2-35 Sniff Mode Interaction .....	69
Hình 3-1 Hai phương pháp truy cập mạng WLAN .....	78
Hình 3-2 Khoá WEP tĩnh được chia sẻ cho AP và các Client trong mạng .....	78
Hình 3-3 Mạng WLAN và các thiết bị xâm nhập .....	79
Hình 3-4 Card mạng với khoá mã WEP bên trong .....	80
Hình 3-5 Các Rogue AP tấn công mạng bằng cách giả danh một AP hợp pháp .....	80
Hình 3-6 Quá trình thiết lập kênh truyền .....	86
Hình 3-7 Bluetooth Key Generation from PIN .....	89
Hình 3-8 Bluetooth Authentication .....	91
Hình 3-9 Bluetooth Encryption Process .....	92
Hình 3-10 Màn hình điện thoại nhiễm Cabir.D .....	105
Hình 3-11 Tin nhắn MMS có kèm sâu Comwarrior .....	111
Hình 3-12 Màn hình cài đặt Doomboot.A .....	114
Hình 3-13 Màn hình yêu cầu cài đặt .....	118
Hình 3-14 Màn hình ngay sau khi cài đặt xong .....	118
Hình 3-15 Màn hình yêu cầu cài đặt sâu Lasco.A .....	120
Hình 3-16 Mosquito Trojan .....	125
Hình 3-17 Troj/Skulls-A .....	127
Hình 3-18 Troj/Skulls-A .....	128
Hình 4-1 Những thiết bị ứng dụng Bluetooth .....	136
Hình 5-1 Các nhà sản xuất được Symbian cấp phép (5/2005) .....	146
Hình 5-2 Kiến trúc hệ điều hành Symbian .....	147
Hình 5-3 Bàn phím của Series 60 .....	150
Hình 6-1 Mô hình đối tượng TPtrC và TPtr .....	156
Hình 6-2 Mô hình đối tượng TBufC và TBuf .....	157
Hình 6-3 Mô hình đối tượng HBufC .....	157
Hình 6-4 Sơ đồ hệ thống các descriptor .....	160
Hình 7-1 Kiến trúc Bluetooth Stack .....	171
Hình 7-2 Quan hệ giữa các nhóm hàm Bluetooth API .....	173
Hình 7-3 Bluetooth Data Element Types .....	175
Hình 7-4 Bluetooth Sockets .....	177
Hình 7-5 Các bước khởi tạo Bluetooth Socket Server .....	179
Hình 7-6 Sự khác biệt giữa chồng giao thức Bluetooth trên thiết bị thật và trên máy ảo .....	202
Hình 7-7 Virtual Bluetooth COM port tạo ra trên máy tính .....	204
Hình 7-8 Cấu hình Bluetooth COM port cho thiết bị giả lập .....	204
Hình 8-1 Qui trình kết nối và gửi nhận dữ liệu .....	208
Hình 8-2 Sơ đồ lớp của phần ứng dụng trên điện thoại .....	209
Hình 8-3 Mô tả chức năng các lớp của phần ứng dụng trên điện thoại .....	210

Hình 8-4 Sơ đồ lớp của phần ứng dụng trên điện thoại (Server) .....	211
Hình 8-5 Quảng bá dịch vụ của Server .....	212
Hình 8-6 Nhận dữ liệu từ Client .....	213
Hình 8-7 Truyền dữ liệu phonebook tới client .....	214
Hình 8-8 Sơ đồ lớp của phần ứng dụng trên điện thoại (Client) .....	215
Hình 8-9 Sơ đồ tìm kiếm thiết bị .....	216
Hình 8-10 Sơ đồ UML truy vấn dịch vụ trên thiết bị .....	217
Hình B - 1 Giao diện ứng dụng trên điện thoại.....	228
Hình B - 2 Khởi tạo điện thoại là server .....	229
Hình B - 3 Trạng thái lắng nghe .....	229
Hình B - 4 Xác nhận yêu cầu kết nối .....	230
Hình B - 5 Menu sau khi kết nối thành công .....	230
Hình B - 6 Lựa chọn các contact để truyền .....	231
Hình B - 7 Sử dụng ứng dụng PbkExchange .....	231
Hình B - 8 Khởi tạo điện thoại là client .....	232
Hình B - 9 Lựa chọn thiết bị để kết nối .....	232
Hình B - 10 Giao diện ứng dụng PbkExchange trên máy tính. ....	233
Hình B - 11 Combo Box lựa chọn công COM .....	233
Hình B - 12 File dữ liệu .....	233
Hình B - 13 Listbox chứa phonebook hiện hành .....	234
Hình B - 14 Thông tin sơ lược của một contact .....	234
Hình B - 15 Dialog NewContact.....	235
Hình B - 16 Textbox Log.....	235
Hình C - 1 Tạo Project symbian mới trên visual C++ .....	237
Hình C - 2 Thông tin project mới tạo ra .....	238
Hình C - 3 Cấu trúc thư mục của ứng dụng HelloWorld .....	238
Hình C - 4 Mở một project đã có .....	239
Hình C - 5 Mở một project đã có .....	239
Hình C - 6 Chạy ứng dụng HelloWorld .....	240
Hình C - 7 Ứng dụng HelloWorld .....	240
Hình C - 8 Biên dịch ứng dụng cho hệ thống ARMI .....	241
Hình C - 9 Biên dịch ứng dụng cho hệ thống ARMI .....	241
Hình C - 10 Tạo file cài đặt .....	242

## Danh sách các bảng

Bảng 1-1 So sánh Wifi và Bluetooth .....	74
Bảng 1-2 So sánh IrDA và Bluetooth .....	76
Bảng 3-1 Mô tả các hàm quảng bá dịch vụ .....	213
Bảng 3-2 Mô tả các hàm tìm thiết bị .....	216
Bảng 3-3 Mô tả các hàm truy vấn dịch vụ .....	218
Bảng 3-4 Tham số hàm ReadFile và WriteFile .....	220

## **Phần 1 LÝ THUYẾT VỀ BLUETOOTH**

- ❖ **Chương 1. Giới thiệu tổng quan về Bluetooth.**
- ❖ **Chương 2. Kỹ thuật Bluetooth.**
- ❖ **Chương 3. Vấn đề về an toàn và bảo mật trong Bluetooth.**
- ❖ **Chương 4. Các ưu nhược điểm và tương lai của Bluetooth.**

KHOA CNTT

# **Chương 1 GIỚI THIỆU TỔNG QUAN VỀ BLUETOOTH**

## **1.1. Khái niệm Bluetooth.**

- \_ Bluetooth là công nghệ không dây cho phép các thiết bị điện, 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 số 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 cable giữa máy tính và các thiết bị truyền thông cá nhân, kết nối vô tuyến giữa các thiết bị điện tử lại với nhau một cách thuận lợi với giá thành rẻ.
- \_ Khi được kích hoạt, Bluetooth có thể tự động định vị những thiết bị khác có chung công nghệ trong vùng xung quanh và bắt đầu kết nối với chúng. Nó được định hướng sử dụng cho việc truyền dữ liệu lẵn tiếng nói.

## **1.2. Lịch sử, hình thành và phát triển của Bluetooth.**

### **1.2.1. Lịch sử tên Bluetooth:**

Bluetooth là tên của nhà vua Đan Mạch- Harald I Bluetooth (Danish King Harald Blåtand) (910-985). Harald Bluetooth đã hợp nhất Đan Mạch và Norway. Ngày nay Bluetooth là biểu tượng của sự thống nhất giữa Computer và Telecom, giữa công nghệ máy tính và công nghệ truyền thông đa phương tiện.

### **1.2.2. Hình thành và phát triển của Bluetooth:**

- \_ **Năm 1994:** Lần đầu tiên hãng Ericsson đưa ra một đề án nhằm hợp nhất liên lạc giữa các loại thiết bị điện tử khác nhau mà không cần phải dùng đến các sợi cáp nối cồng kềnh, phức tạp. Đây thực chất là một mạng vô tuyến không dây cự ly ngắn chỉ dùng một vi mạch cỡ 9mm có thể chuyển các tín hiệu sóng vô tuyến điều khiển thay thế cho các sợi dây cáp điều khiển rối rắm.

- **Năm 1998:** 5 công ty lớn trên thế giới gồm Ericsson, Nokia, IBM, Intel và Toshiba đã liên kết, hợp tác thiết kế và triển khai phát triển một chuẩn công nghệ kết nối không dây mới mang tên **BLUETOOTH** nhằm kết nối các thiết bị vi điện tử lại với nhau dùng sóng vô tuyến.
- **Đến ngày 20/5/1998:** nhóm nghiên cứu Special Interest Group - SIG chính thức được thành lập với mục đích phát triển công nghệ Bluetooth trên thị trường viễn thông. Bất kỳ công ty nào có kế hoạch sử dụng công nghệ Bluetooth đều có thể tham gia vào.
- **Tháng 7/1999:** các chuyên gia trong SIG đã đưa ra thuyết minh kỹ thuật Bluetooth phiên bản 1.0.
- **Năm 2000 :** SIG đã bổ sung thêm 4 thành viên mới là 3Com, Lucent Technologies, Microsoft và Motorola. Công nghệ Bluetooth đã được cấp dấu chứng nhận kỹ thuật ngay trong lần ra mắt đầu tiên.
- **Năm 2001:** Bluetooth 1.1 ra đời cùng với bộ Bluetooth software development kit-XTNDAccess Blue-SDK, đánh dấu bước phát triển chưa từng có của công nghệ Bluetooth trên nhiều lĩnh vực khác nhau với sự quan tâm của nhiều nhà sản xuất mới. Bluetooth được bình chọn là công nghệ vô tuyến tốt nhất trong năm.
- **Tháng 7/2002,** Bluetooth SIG thiết lập cơ quan đầu não toàn cầu tại Overland Park, Kansas, USA. Năm 2002 đánh dấu sự ra đời các thế hệ máy tính Apple hỗ trợ Bluetooth. Sau đó không lâu Bluetooth cũng được thiết lập trên máy Macintosh với hệ điều hành MAC OS S. Bluetooth cho phép chia sẻ tập tin giữa các máy MAC, đồng bộ hóa và chia sẻ thông tin liên lạc giữa các máy Palm, truy cập internet thông qua điện thoại di động có hỗ trợ Bluetooth (Nokia, Ericsson, Motorola...).
- **Tháng 5/2003,** CSR (Cambridge Silicon Radio) cho ra đời 1 chip Bluetooth mới với khả năng tích hợp dễ dàng và giá cả hợp lý hơn. Điều này góp phần cho sự ra đời thẻ hệ Motherboard tích hợp Bluetooth, giảm sự chênh lệch giá cả giữa những mainboard,

cellphone có và không có Bluetooth. Tháng 11/2003 dòng sản phẩm Bluetooth 1.2 ra đời.

- **Năm 2004**, các công ty điện thoại di động tiếp tục khai thác thị trường sôi nổi này bằng cách cho ra đời các thế hệ điện thoại di động đời mới hỗ trợ Bluetooth (N7610, N6820, N6230). Motorola cho ra sản phẩm Bluetooth đầu tay của mình. Các sản phẩm Bluetooth tiếp tục ra đời và được và được xúc tiến mạnh mẽ qua chương trình “Operation Blueshock” International Consumer Electronics Show (CES) tại Las Vegas ngày 9/1/2004.
- **6-1-2004**, trong hội nghị Bluetooth CES (Consumer Electronics Show) ở Las Vegas, tổ chức Bluetooth SIG thông báo số thành viên của mình đã đạt con số 3000, trở thành tổ chức có số thành viên đông đảo thuộc nhiều lĩnh vực công nghệ: từ máy móc tự động đến thiết bị y tế, PC đến điện thoại di động, tất cả đều sử dụng kỹ thuật không dây tầm ngắn trong sản phẩm của họ
- Bluetooth hiện đang có tốc độ phát triển khá nhanh với khả năng ứng dụng ngày càng đa dạng, theo tính toán của công ty nghiên cứu thị trường Frost & Sullivan, trong năm 2001 có 4.2 triệu sản phẩm sử dụng công nghệ Bluetooth được đưa ra thị trường, con số này sẽ tăng lên 1.01 tỷ vào năm 2006.
- Những năm gần đây, Bluetooth được coi là thị trường năng động và sôi nổi nhất trong lĩnh vực truyền thông. Với sự ra đời của công nghệ Bluetooth thì ta có thể lạc quan nói rằng, thời kỳ kết nối bằng dây hữu tuyến giữa các thiết bị đã đến hồi kết thúc, thay vào đó là khả năng kết nối không dây thông minh và trong suốt, điều này sẽ là hiện thực chỉ trong một tương lai gần mà thôi.

### 1.3. Các đặc điểm của Bluetooth.

- Tiêu thụ năng lượng thấp, cho phép ứng dụng được trong nhiều loại thiết bị, bao gồm cả các thiết bị cầm tay và điện thoại di động
- Giá thành hạ (Giá một chip Bluetooth đang giảm dần, và có thể xuống dưới mức 5\$ một đơn vị).

- Khoảng cách giao tiếp cho phép :
  - Khoảng cách giữa hai thiết bị đầu cuối có thể lên đến 10m ngoài trời, và 5m trong tòa nhà.
  - Khoảng cách thiết bị đầu cuối và Access point có thể lên tới 100m ngoài trời và 30m trong tòa nhà.
- Bluetooth sử dụng băng tần không đăng ký 2.4Ghz trên dãy băng tần ISM. Tốc độ truyền dữ liệu có thể đạt tối đa 1Mbps (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 khác thông qua các chuẩn “Bluetooth profiles”, do đó có thể độc lập về phần cứng cũng như hệ điều hành sử dụng.
- Bluetooth được dùng trong giao tiếp dữ liệu tiếng nói: có 3 kênh để truyền tiếng nói, 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 and encryption)
- Tính tương thích cao, được nhiều nhà sản xuất phần cứng cũng như phần mềm hỗ trợ.

## 1.4. Ứng dụng của Bluetooth.

### 1.4.1. Thiết bị thông minh.

Gồm có các loại điện thoại di động, PDA, PC, cellphone, smartphone...

Điện thoại di động: Sony Ericsson P800, P900, Nokia 3650, 7610, 7650... Công nghệ Bluetooth gắn sẵn trên thiết bị di động nên không cần dùng cáp. Có thể kết nối với tai nghe Bluetooth, camera kỹ thuật số hay máy tính, cho phép người dùng xem tivi, chụp ảnh, quay phim, nghe MF3, FM, duyệt web và email từ điện thoại...



Hình 1-1 Nokia 6600 - PalmTungsten

Palm Tungsten W: một trung tâm dữ liệu cầm tay, cung cấp một sự kết hợp tinh vi của công nghệ thu điện tử không dây, thông điệp SMS, các chức năng của điện thoại, các ứng dụng kinh doanh và phần mềm quản lý thông tin cá nhân của Palm. Với 3 băng tần 900-1800-1900 MHz, Palm Tungsten W được chế tạo với một trong những sóng vô tuyến nhanh nhất hiện nay cho các mạng GSM/GPRS, vì thế có thể dùng nó như một chiếc điện thoại với tai nghe Bluetooth. Plam Tungsten W không sử dụng SIM, và có thể dùng với bất kỳ nhà cung cấp dịch vụ nào.

#### 1.4.2. Thiết bị truyền thanh.

Gồm các loại tai nghe (headset), loa và các trạm thu âm thanh...



Hình 1-1 Tai nghe Bluetooth

Công ty Logitech chuyên sản xuất thiết bị ngoại vi cho máy tính PC vừa giới thiệu loại tai nghe Bluetooth di động. So với các tai nghe bluetooth khác trên thị trường, tai nghe HS02 hỗ trợ tiêu chuẩn Bluetooth phiên bản 1.2 và có loa nghe lớn 2cm.

Tiêu chuẩn Bluetooth 1.2 giảm thời gian kết nối và tiêu thụ điện năng khi nối với thiết bị Bluetooth 1.2 khác. Ngoài ra khả năng sử dụng công nghệ

tần số tiêu chuẩn 1.2 giúp tránh xung nhiễu từ các thiết bị tần số 2,4Ghz khác như mạng không dây WiFi và các điện thoại không dây.

Với hỗ trợ Bluetooth 1.2, âm thanh nghe qua tai nghe to, như trên loa.

Tai nghe cung cấp tần số trả về lớn và có âm lượng tối đa lớn hơn cần thiết để sử dụng trong trường hợp nhất định.

#### 1.4.3. Thiết bị truyền dữ liệu.

Gồm chuột, bàn phím, joystick, camera, bút kỹ thuật số, máy in, LAN access point...



Hình 1-2 Thiết bị truyền dữ liệu

Modem Zoom dùng để kết nối Internet hoặc mạng cục bộ bằng điện thoại. Nó có 2 ngõ giao tiếp với PC hay PDA: ngõ không dây Bluetooth class 1, bán kính hoạt động 100m; ngõ RS232 qua cổng COM. Tốc độ 56Kbps.



Hình 1-3 USB Bluetooth

Bluetooth MDU 0001USB là thiết bị kết nối không dây sử dụng công nghệ Bluetooth class 2, vùng phủ sóng bán kính 10m; nối với PC qua USB 1.1. Tuy nhỏ như đầu ngón tay nhưng thiết bị được tích hợp gần như tất cả các chuẩn giao tiếp hiện có, ví dụ: RS232, FTP, Dial-up, Fax, OBEX (chuẩn đồng bộ hóa dữ liệu cho PDA)..., nên khi lắp MDU 0001USB vào thì vô hình trung PC của bạn biến thành một đài phát sóng. Ngược lại, PC này cũng có thể dò

tìm và kết nối đến tất cả máy tính, PDA đang trong vùng phủ sóng. Cắm thiết bị, cài đặt driver, khởi động lại máy là tất cả các máy tính trong bán kính 10m có thể kết nối, trao đổi dữ liệu với nhau.



**Máy ảnh điều khiển bằng điện thoại di động**

Hình 1-4 Máy ảnh điều khiển bằng điện thoại di động

Sản phẩm của Sony Ericsson có tên ROB-1ROB-1 được điều khiển không dây qua bàn phím joystick hoặc nhờ màn hình cảm ứng trên điện thoại P900 hoặc P910. Người sử dụng nhìn trên màn hình điện thoại những gì đang có trên ống kính máy ảnh, và chụp các hình ảnh trên điện thoại như một máy ảnh thông thường.

Với khả năng lăn tròn chung quanh với khoảng cách 50m cách người sử dụng, ROB-1 cùng lúc truyền hình ảnh trên điện thoại di động.

Thiết bị có ba bánh xe và có dạng hình cầu kết hợp với công nghệ máy ảnh thông minh giúp di chuyển nhanh nhẹn và cơ động với góc nhìn rộng.

Có đường kính 11cm, ROB-1 có thể di chuyển về phía trước, sau, nhìn quanh các góc, quanh một điểm hay nghiêng ống kính một góc 70 độ lên trên và 20 độ xuống dưới. Có đèn chiếu phía trước giúp chụp trong bóng tối. Bộ nhớ lớn giúp chụp một số ảnh trước khi lưu vào điện thoại, hoặc truyền tới một máy tính PC qua cổng USB.

#### **1.4.4. Các ứng dụng nhúng.**

Điều khiển nguồn năng lượng trong xe hơi, các loại nhạc cụ, trong công nghiệp, y tế...

Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa



Hình 1-5 Màn hình hiển thị theo giao diện dành cho các cuộc điện thoại.

Kể từ nay, những khách hàng của chiếc sedan Lexus LS430 và xe thể thao hai cầu LX470 sẽ không cần phải dừng lại trên đường để nhận hay tiến hành cuộc gọi điện thoại di động. Các thao tác được thực hiện đơn giản nhờ một nút bấm trên tay lái hoặc qua màn hình kỹ thuật số.



Hình 1-6 Bluetooth Car Kit



Hình 1-7 Máy chụp hình kỹ thuật số có hỗ trợ Bluetooth để truyền hình ảnh

#### 1.4.5. Một số ứng dụng khác.

Do số lượng công ty tham gia vào tổ chức SIG ngày càng nhiều, vì vậy, số lượng các loại sản phẩm được tích hợp công nghệ Bluetooth được tung ra thị trường ngày càng nhiều, bao gồm cả các thiết bị dân dụng như tủ lạnh, lò vi sóng, máy điều hòa nhiệt độ, các loại đồ chơi...



- ▶ Đồ chơi điều khiển bằng điện thoại di động.

Đây là một phụ kiện hoàn toàn mang tính giải trí. Bộ *CAR-100 Bluetooth Car Kit* do Sony Ericsson sản xuất hoàn toàn chỉ là một chiếc ôtô điện tử to bằng bao diêm. Nó có chiều dài 7 cm, được điều khiển bằng cách bấm các phím trên một chiếc điện thoại Sony Ericsson có hỗ trợ Bluetooth. Nó được nạp điện bằng cách nối ngay vào khe cắm ở đuôi điện thoại. Ôtô có thể chạy liên 1 tiếng với cự ly hoạt động là 10 m. Thời gian nạp lại điện cũng đòi hỏi ít nhất 1 tiếng.



Hình 1-8 Đồng hồ có hỗ trợ Bluetooth để nghe nhạc mp3

## **Chương 2 KỸ THUẬT BLUETOOTH.**

### **2.1. Các khái niệm dùng trong công nghệ Bluetooth.**

#### **2.1.1. Master Unit :**

Là thiết bị duy nhất trong 1 Piconet, Master thiết lập đồng hồ đếm xung và kiểu bước nhảy (hopping) để đồng bộ tất cả các thiết bị trong cùng piconet mà nó đang quản lý, thường là thiết bị đầu tiên chuyển đổi dữ liệu. Master cũng quyết định số kênh truyền thông. Mỗi Piconet có một kiểu hopping duy nhất.

#### **2.1.1. Slave Unit :**

Là tất cả các thiết bị còn lại trong piconet, một thiết bị không là Master thì phải là Slave. Tối đa 7 Slave dạng Active và 255 Slave dạng Parked (Inactive) trong 1 Piconet.

Có 3 dạng Slave trong một Piconet:

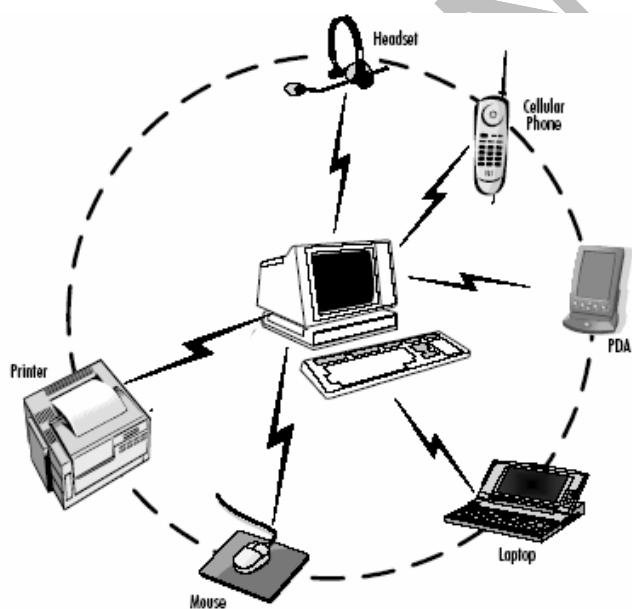
- Active: Slave hoạt động, có khả năng trao đổi thông tin với Master và các Slave Active khác trong Piconet. Các thiết bị ở trạng thái này được phân biệt thông qua 1 địa chỉ MAC (Media Access Control) hay AMA (Active Member Address ) - đó là con số gồm 3 bit. Nên trong 1 Piconet có tối đa 8 thiết bị ở trạng thái này (1 cho Master và 7 cho Slave).
- Standby: Standby là một dạng inactive, thiết bị trong trạng thái này không trao đổi dữ liệu, sóng radio không có tác động lên, công suất giảm đến tối thiểu để tiết kiệm năng lượng, thiết bị không có khả năng dò được bất cứ mã truy cập nào. Có thể coi là những thiết bị trong nằm ngoài vùng kiểm soát của Master.
- Parked: là một dạng inactive, chỉ 1 thiết bị trong 1 Piconet thường xuyên được đồng bộ với Piconet, nhưng không có 1 địa chỉ MAC. Chúng như ở trạng thái "ngủ" và sẽ được Master gọi dậy bằng tín hiệu "beacon" (tín hiệu báo hiệu). Các thiết bị ở trạng thái Parked được đánh địa chỉ thông qua địa chỉ PMA (Packed Member Address). Đây là con số 8 bits để phân biệt các packed Slave với nhau và có tối đa 255 thiết bị ở trạng thái này trong 1 Piconet.

### 2.1.2. Piconet:

Picotnet là tập hợp các thiết bị được kết nối thông qua kỹ thuật Bluetooth theo mô hình Ad-Hoc (đây là kiểu mạng được thiết lập cho nhu cầu truyền dữ liệu hiện hành và tức thời, tốc độ nhanh và kết nối sẽ tự động huỷ sau khi truyền xong). Trong 1 Piconet thì chỉ có 1 thiết bị là Master. Đây thường là thiết bị đầu tiên tạo kết nối, nó có vai trò quyết định số kênh truyền thông và thực hiện đồng bộ giữa các thành phần trong Piconet, các thiết bị còn lại là Slave. Đó là các thiết bị gửi yêu cầu đến Master.

Lưu ý rằng, 2 Slave muốn thực hiện liên lạc phải thông qua Master bởi chúng không bao giờ kết nối trực tiếp được với nhau, Master sẽ đồng bộ các Slave về thời gian và tần số. Trong 1 Piconet có tối đa 7 Slave đang hoạt động tại 1 thời điểm.

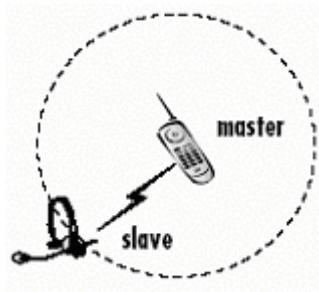
- Minh họa một Piconet:



Hình 2-1 Một Piconet trong thực tế.

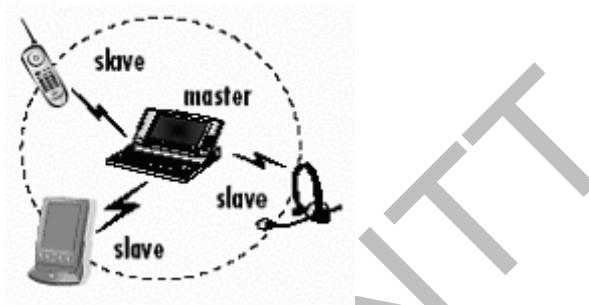
❖ Các mô hình Piconet :

► Piconet chỉ có 1 Slave :



Hình 2-2 Piconet gồm 1 Slave.

► Piconet gồm nhiều Slave :



Hình 2-3 Piconet gồm nhiều Slave.

❖ Cách hình thành một Piconet:

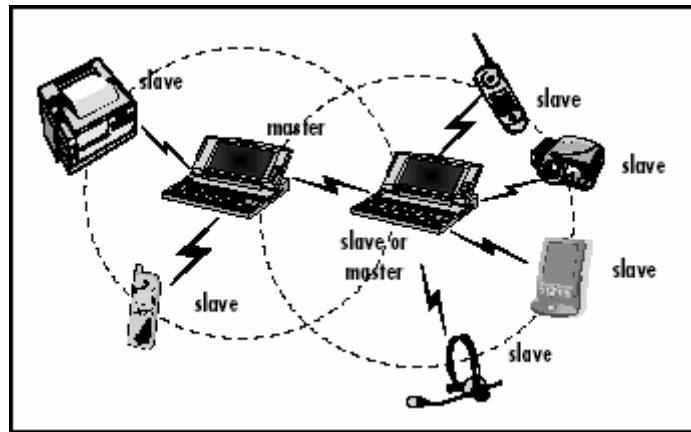
Một piconet bắt đầu với 2 thiết bị kết nối với nhau, như laptop PC với 1 Mobilephone. Giới hạn 8 thiết bị trong 1 Piconet (3 bit MAC cho mỗi thiết bị). Tất cả các thiết bị Bluetooth đều ngang hàng và mang chức năng xác định. Tuy nhiên khi thành lập 1 Piconet, 1 thiết bị sẽ đóng vai Master để đồng bộ về tần số và thời gian truyền phát, và các thiết bị khác làm Slave.

#### 2.1.3. Scatternet:

- \_ Là 2 hay nhiều Piconet độc lập và không đồng bộ, các Piconet này kết hợp lại truyền thông với nhau.
- \_ Lưu ý:
  - Một thiết bị có thể vừa là Master của Piconet này, vừa là Slave của Piconet khác.
  - Vai trò của 1 thiết bị trong Piconet là không cố định, có nghĩa là nó có thể thay đổi từ Master thành Slave và ngược lại, từ Slave thành Master. Ví dụ nếu Master không đủ khả năng cung cấp tài nguyên phục vụ cho Piconet của mình thì nó sẽ chuyển quyền

cho 1 Slave khác giàu tài nguyên hơn, mạnh hơn, bởi vì trong 1 piconet thì Clock và kiểu Hopping đã được đồng bộ nhau sẵn.

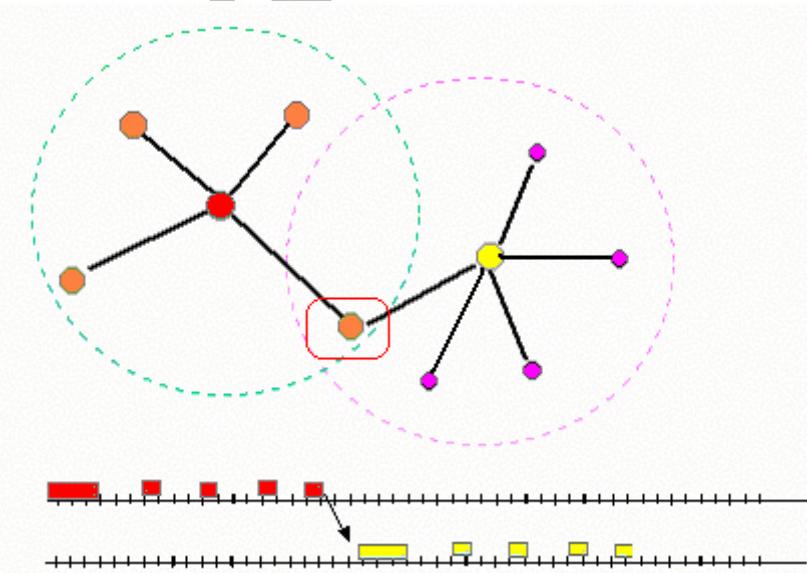
\_ Ví dụ một Scatternet :



Hình 2-4 Một Scatternet gồm 2 Piconet.

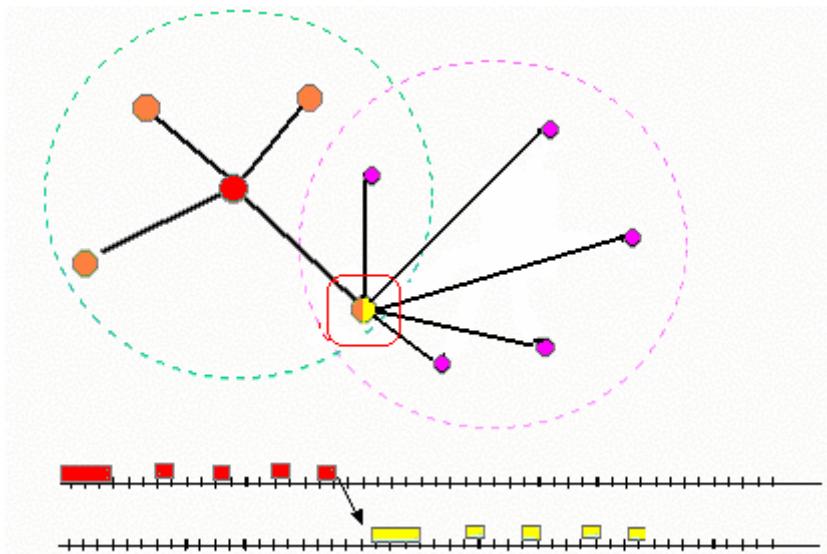
\_ Có 2 cách hình thành một Scatternet:

- ▶ Cách 1: Piconet này cử ra 1 Slave làm Slave của Piconet kia (các Piconet là độc lập với nhau và không đồng bộ). Slave này sẽ phân chia các time slots (TS), một vài TS ở Piconet này, vài TS ở Piconet kia.



Hình 2-5 Sự hình thành một Scatternet theo cách 1.

- ▶ Cách 2: Một Slave trong Piconet này trở thành 1 Master trong 1 Piconet khác. Cũng bằng cách chia các TS như trên cách 1. Cách này cho phép 2 Piconet đồng bộ nhau về clock (xung nhịp) và kiểu hopping (kiểu nhảy tần số). Vì 1 Slave đóng vai trò Master trong 1 Piconet mới, sẽ mang theo clock và hopping của Piconet cũ, đồng bộ cho các Slave trong Piconet mới mà nó làm Master.



Hình 2-6 Sự hình thành một Scatternet theo cách 2.

- Khi có nhiều Piconet độc lập, có thể bị nhiễu trên một số kênh, những packet đó sẽ bị mất và được truyền lại. Nếu tín hiệu là tiếng nói (tín hiệu thoại), chúng sẽ bị bỏ qua.

#### 2.1.4. Kết nối theo kiểu ad hoc:

Không có sự phân biệt giữa các radio units; nghĩa là không có sự phân biệt dựa vào vị trí hay khoảng cách. Kết nối ad hoc dựa vào sự liên lạc giữa các điểm, không cần thiết bị hỗ trợ kết nối giữa các thiết bị di động, không cần mạch điều khiển trung tâm cho các unit dựa vào để thiết lập kết nối. Trong Bluetooth, nó giống như một số lượng lớn các kết nối ad hoc cùng tồn tại trong một vùng mà không cần bất kỳ một sự sắp xếp nào, các network độc lập cùng tồn tại chồng chéo lên nhau.

### 2.1.5. Định nghĩa các liên kết vật lý trong Bluetooth:

- Asynchronous connectionless (ACL): được thiết lập cho việc truyền dữ liệu, những gói dữ liệu cơ bản (primarily packet data). Là một kết nối point-to-multipoint giữa Master và tất cả các Slave tham gia trong piconet. Chỉ tồn tại duy nhất một kết nối ACL. Chúng hỗ trợ những kết nối chuyển mạch gói (packet-switched connection) đối xứng và không đối xứng. Những gói tin đa khe dùng ACL link và có thể đạt tới khả năng truyền tối đa 723 kbps ở một hướng và 57.6 kbps ở hướng khác. Master điều khiển độ rộng băng tầng của ACL link và sẽ quyết định xem trong một piconet một slave có thể dùng băng tầng rộng bao nhiêu. Những gói tin broadcast truyền bằng ACL link, từ master đến tất cả các slave. Hầu hết các gói tin ACL đều có thể truyền lại.
- Synchronous connection-oriented (SCO): hỗ trợ kết nối đối xứng, chuyển mạch mạch (circuit-switched), point-to-point giữa một Master và một Slave trong 1 piconet. Kết nối SCO chủ yếu dùng để truyền dữ liệu tiếng nói. Hai khe thời gian liên tiếp đã được chỉ định trước sẽ được dành riêng cho SCO link. Dữ liệu truyền theo SCO link có tốc độ 64kbps. Master có thể hỗ trợ tối đa 3 kết nối SCO đồng thời. SCO packet không chứa CRC (Cyclic Redundancy Check) và không bao giờ truyền lại. Liên kết SCO được thiết lập chỉ sau khi 1 liên kết ACL đầu tiên được thiết lập.

### 2.1.6. Trang thái của thiết bị Bluetooth:

Có 4 trạng thái chính của 1 thiết bị Bluetooth trong 1 piconet:

- Inquiring device (inquiry mode): thiết bị đang phát tín hiệu tìm thiết bị Bluetooth khác.
- Inquiry scanning device (inquiry scan mode): thiết bị nhận tín hiệu inquiry của thiết bị đang thực hiện inquiring và trả lời.
- Paging device (page mode): thiết bị phát tín hiệu yêu cầu kết nối với thiết bị đã inquiry từ trước.
- Page scanning device (page scan mode): thiết bị nhận yêu cầu kết nối từ paging device và trả lời.

### 2.1.7. Các chế độ kết nối:

- Active mode: trong chế độ này, thiết bị Bluetooth tham gia vào hoạt động của mạng. Thiết bị master sẽ điều phối lưu lượng và đồng bộ hóa cho các thiết bị slave.
- Sniff mode: là 1 chế độ tiết kiệm năng lượng của thiết bị đang ở trạng thái active. Ở Sniff mode, thiết bị slave lắng nghe tín hiệu từ mạng với tần số giảm hay nói cách khác là giảm công suất. Tần số này phụ thuộc vào tham số của ứng dụng. Đây là chế độ ít tiết kiệm năng lượng nhất trong 3 chế độ tiết kiệm năng lượng.
- Hold mode: là 1 chế độ tiết kiệm năng lượng của thiết bị đang ở trạng thái active. Master có thể đặt chế độ Hold mode cho slave của mình. Các thiết bị có thể trao đổi dữ liệu ngay lập tức ngay khi thoát khỏi chế độ Hold mode. Đây là chế độ tiết kiệm năng lượng trung bình trong 3 chế độ tiết kiệm năng lượng.
- Park mode: là chế độ tiết kiệm năng lượng của thiết bị vẫn còn trong mạng nhưng không tham gia vào quá trình trao đổi dữ liệu (inactive). Thiết bị ở chế độ Park mode bỏ địa chỉ MAC, chỉ lắng nghe tín hiệu đồng bộ hóa và thông điệp broadcast của Master. Đây là chế độ tiết kiệm năng lượng nhất trong 3 chế độ tiết kiệm năng lượng.

## 2.2. Bluetooth Radio.

### 2.2.1. Ad Hoc Radio Connectivity

Phần lớn hệ thống radio trong thương mại sử dụng ngày nay đều được dựa vào cấu trúc tế bào radio. Một mạng mobile thiết lập cơ sở hạ tầng bằng những sợi cáp kim loại theo dạng xương sống, dùng một hoặc nhiều trạm cơ sở đặt ở những vị trí chiến lược để sóng có thể phủ hết các tế bào; thiết bị sử dụng là những điện thoại có khả năng di chuyển, hoặc nói chung là những terminal di động, để sử dụng mobile network; những terminal này duy trì một kết nối với mạng thông qua một radio link đến các trạm cơ sở. Đây là liên kết chẽ giữa trạm cơ sở và terminal. Khi một terminal đăng ký với mạng, nó sẽ giữ một kênh điều khiển, và kết nối sẽ được thiết lập hoặc giải phóng theo nghi thức của kênh đó. Truy xuất kênh, chia kênh, điều khiển lưu thông và những sự can

thiệp khác đều được điều khiển một cách gọn gàng bởi các trạm cơ sở. Chẳng hạn theo quy ước của hệ thống radio thì những hệ thống điện thoại công cộng như là Global System for Mobile Communications (GSM), D-AMPS, và IS-95 [1–3], nhưng cũng có những hệ thống tư nhân như hệ thống mạng cục bộ không dây (WLAN) dựa trên 802.11 hoặc HIPERLAN I và HIPERLAN II [4–6], và hệ thống cordless như Digital Enhanced Cordless Telecommunications (DECT) và Personal Handyphone System (PHS) [7,8].

Trái lại, trong hệ thống ad hoc thật sự thì không hề có sự khác biệt giữa các radio unit; tức là không hề có điểm khác biệt giữa các trạm cơ sở và terminal. Liên kết ad hoc tùy thuộc vào sự liên lạc giữa các thiết bị. Không có cơ sở hạ tầng là dây cáp kim loại hỗ trợ kết nối giữa các unit di động, không có thiết bị kiểm soát trung tâm cho các unit dựa vào để tạo các quan hệ nối liền với nhau, cũng không có hỗ trợ việc sắp xếp truyền thông.Thêm vào đó, ở đây không có sự can thiệp của người điều hành. Có thể mường tượng kịch bản của Bluetooth như thế này, nó có vẻ như là một số lượng lớn các kết nối ad hoc cùng tồn tại ở cùng một vùng mà không có bất cứ sự phối hợp lẫn nhau nào. Đối với những ứng dụng Bluetooth, có nhiều mạng độc lập chồng chéo lên nhau trên cùng một vùng.

Hệ thống ad hoc radio chỉ được dùng trong vài trường hợp như hệ thống walky-talky dùng bởi quân đội, cảnh sát, cứu hỏa, và những đội cứu hộ nói chung.Tuy nhiên, hệ thống Bluetooth là hệ thống ad hoc radio thương mại đầu tiên được dùng một cách rộng rãi và với quy mô lớn nơi công cộng.

### **2.2.2. Kiến trúc của hệ thống Bluetooth Radio**

#### **2.2.2.1. Radio Spectrum-Dãy sóng vô tuyến:**

- \_ Thứ nhất việc chọn lựa dãy sóng vô tuyến phải được xác định mà không có người điều hành tác động. Dãy sóng phải được dùng nơi công cộng mà không cần phải đăng ký. Thứ hai, dãy sóng phải sẵn sàng để dùng ở trên toàn thế giới. Những ứng dụng Bluetooth đầu tiên đặt mục tiêu là những doanh nghiệp đi du lịch, những người phải kết nối thiết bị di động của họ ở bất cứ nơi nào họ đến. May thay có một tần số vô tuyến không phải đăng ký

luôn sẵn dùng trên toàn cầu. Đó là tần số Industrial, Scientific, Medical (ISM), vào khoảng 2,45 GHz và trước đây được dành riêng cho một số nhóm chuyên nghiệp nhưng gần đây thì đã được mở rộng trên toàn thế giới cho mục đích thương mại. Ở Mỹ, băng tần này đi từ 2400 đến 2483.5 MHz, và những điều lệ FCC (Federal Communications Commission) phần 15 được áp dụng. Ở phần lớn châu Âu, một băng tần giống nhau được dùng theo điều lệ ETS-300328. Ở Nhật, gần đây băng tần từ 2400 đến 2500 MHz được phép dùng cho những ứng dụng thương mại và hòa hợp với giải pháp của thế giới. Tóm lại, ở hầu hết các quốc gia trên thế giới, tần số miễn phí sẵn dùng từ 2400 MHz đến 2483,5 MHz, và những nỗ lực cho sự hòa hợp đang được tiến hành để dây sóng vô tuyến này thật sự sẵn dùng trên toàn thế giới.

- \_ Những quy định không giống nhau ở những nơi khác nhau trên thế giới. Tuy nhiên mục tiêu của họ là làm sao để bất kỳ người sử dụng nào cũng có quyền sử dụng tần số vô tuyến đó một cách công bằng. Những quy luật nói chung quy định rõ sự phân bố của những tín hiệu được truyền đi và mức năng lượng tối đa được phép truyền. Do đó, đối với một hệ thống có thể hoạt động trên toàn cầu thì khái niệm tần số vô tuyến được phép dùng phải là phần giao của các luật lệ.

#### **2.2.2.2. Interference Immunity – Sự chống nhiễu:**

- \_ Do băng tần miễn phí có thể được sử dụng bởi bất cứ một thiết bị phát nào, do đó việc chống nhiễu là vấn đề rất quan trọng. Phạm vi và khả năng nhiễu trong tần số ISM 2.45 GHz là không thể dự đoán trước được, bởi có rất nhiều thiết bị phát sử dụng sóng vô tuyến ở trong băng tần này, đó có thể là thiết bị Bluetooth, thiết bị Wifi, ... và thậm chí cả lò vi sóng và một vài thiết bị phát sáng khác cũng phát ra sóng trong băng tần này.
- \_ Sự chống nhiễu có được thực hiện nhờ vào việc ngăn chặn hoặc tránh đi. Ngăn chặn bằng cách dàn trải những chuỗi hoặc mã (coding or direct-sequence spreading).

- \_ Sự ngăn chặn có thể được thực hiện bằng cách viết code hoặc chia tần số thành các dãy liên tục. Tuy nhiên, phạm vi các dãy tần động của các tín hiệu được can thiệp trong một môi trường sóng đặc biệt, liên tục có thể rất rộng. Phân chia theo thời gian có thể là một lựa chọn nếu như xảy ra sự gián đoạn trong các nhịp tần số của sự phân chia theo thời gian. Việc phân chia trên tần số có khả năng hơn. Trong khi tần số 2.45 GHz có thể cung cấp băng thông khoảng 80 MHz và băng thông của hầu hết các hệ thống radio đều bị giới hạn, một số phần quang phổ của sóng radio có thể được sử dụng mà không gặp bất cứ trở ngại nào. Việc lọc trên các vùng băng tần sẽ giúp ngăn nhiễu ở những phần khác của dãy sóng radio. Bộ lọc ngăn chặn có thể dễ dàng đạt đến tần số 50 dB hoặc hơn nữa.

#### **2.2.2.3. Multiple Access Scheme\_Phối hợp đa truy cập:**

- \_ Việc lựa chọn sự phối hợp đa truy cập cho một hệ thống vô tuyến ad hoc được điều khiển bởi những luật lệ của dãy tầng ISM và thiếu sự phối hợp (lack of coordination)
- \_ Đa truy cập phân chia theo tần số (FDMA) đã thu hút những hệ thống ad hoc do kênh trực giao chỉ trả lời đúng tần số của máy tạo dao động tương ứng trên các băng tần khác nhau. Phối hợp với việc phân chia kênh truyền một cách thích ứng và năng động thì việc nhiễu có thể tránh khỏi. Đáng tiếc FDMA cơ bản lại không đáp ứng hết nhu cầu lan rộng có trong dãy ISM.
- \_ Đa truy cập phân chia theo thời gian (TDMA) đòi hỏi sự đồng bộ về thời gian vô cùng khắc khe ở kênh trực giao. Đối với nhiều liên kết ad hoc được sắp xếp ở một chỗ, việc duy trì sự tham chiếu khung thời gian trở nên khá cồng kềnh.
- \_ Đa truy cập phân chia theo mã (CDMA) tỏ ra là đặc tính tốt nhất cho hệ thống vô tuyến ad hoc khi nó quy định sự phân bổ và đề cập đến những hệ thống rác.
- \_ Direct sequence (DS)-CDMA không thu hút bằng vì vấn đề gần xa, nó đòi hỏi kiểm soát năng lượng lẫn nhau hoặc tăng thêm xử lý thừa.Thêm vào đó, như TDMA, kênh trực giao DS-CDMA cũng quy định việc tham chiếu

khung thời gian. Cuối cùng, đối với những user cao cấp thì những loại chip khá đắt đã được dùng đến nhưng không thu hút lắm vì băng thông rộng (tránh nhiễu) và sự tiêu thụ hiện tại ngày càng tăng.

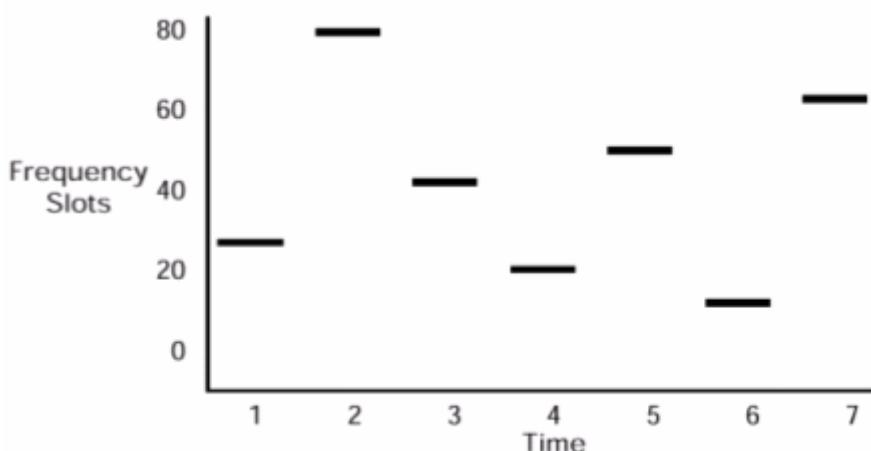
- \_ Nhảy tần số (FH)-CDMA kết hợp một số những đặc tính để trở thành chọn lựa tốt nhất cho hệ thống vô tuyến ad hoc. Trung bình một tín hiệu có thể trải ra trên một dãy tần số lớn, nhưng ngay lúc đó chỉ có một dải băng thông nhỏ được sử dụng, tránh được hầu hết khả năng nhiễu trong dãy ISM. Bước nhảy của sóng mang là trực giao, và việc nhiễu trên những bước sóng kế nhau có thể bị ngăn chặn bởi bộ lọc. Việc phối hợp những bước sóng có thể sẽ không trực giao( dù sao việc phối hợp lẩn nhau giữa các bước sóng không được cho phép theo luật FCC ), nhưng băng thông hẹp và việc nhiễu khi người dùng chung (co-user) chỉ bị xem như là gián đoạn ngắn trong việc truyền tin, một việc có thể được khắc phục bằng giải pháp dùng những kỹ thuật ở tầng cao hơn.
- \_ Bluetooth dựa vào kỹ thuật FH-CDMA- các packet được truyền trên những tần số khác nhau. Trong dãy tầng ISM 2.45 GHz, định nghĩa một bộ 79 bước nhảy, mỗi bước nhảy cách nhau 1MHz. Việc truyền nhận sử dụng các khe thời gian. Chiều dài 1 khe thời gian thông thường là  $625\mu s$ . Một số lớn những cách phối hợp bước nhảy được tạo ra ngẫu nhiên nhưng chỉ cách phối hợp đặc biệt được định nghĩa bởi một unit gọi là master mới kiểm soát kênh nhảy tần số. Một đồng hồ của master unit cũng định nghĩa một chu kỳ bước nhảy. Tất cả những unit khác đều gọi là slave, chúng dùng sự đồng nhất của master để chọn bước nhảy giống nhau và cộng thêm khoảng thời gian gián đoạn vào đồng hồ tương ứng của chúng để đồng bộ hóa việc nhảy tần số. Trong lĩnh vực thời gian, các kênh được chia thành những slot. Một slot tương ứng với một khoảng thời gian tối thiểu là  $625 \mu s$ . Để thực hiện đơn giản, truyền tin song công được thực hiện bằng cách áp dụng time-division duplex (TDD). Điều này có nghĩa là một unit sẽ lần lượt phát và nhận. Chia cắt việc phát và nhận thực sự ngăn chặn được nhiễu xuyên âm giữa quá trình phát và nhận trong máy thu phát vô tuyến. Từ khi việc phát

và nhận đặt ở những time slot khác nhau thì chúng cũng được đặt ở những bước nhảy khác nhau.

### 2.3. Kỹ thuật trai phô nhảy tần số trong công nghệ Bluetooth.

#### 2.3.1. Khái niệm trai phô trong công nghệ không dây:

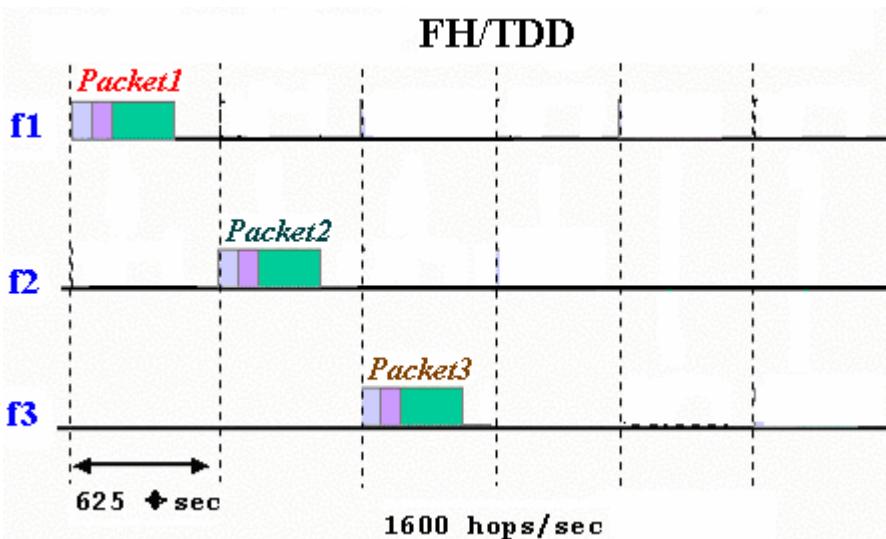
- \_ Trong truyền thông bằng sóng radio cổ điển, người ta chỉ dùng một tần số để truyền dữ liệu, nhưng khả năng mất dữ liệu là rất lớn do tần số này có thể bị nhiễu, mặt khác tốc độ truyền sẽ không cao.
- \_ Truyền thông trai phô là kỹ thuật truyền tín hiệu sử dụng nhiều tần số cùng 1 lúc (DSSS-Direct Sequence Spread Spectrum) hoặc luân phiên (FHSS-Frequency Hopping Spread Spectrum) để tăng khả năng chống nhiễu, bảo mật và tốc độ truyền dữ liệu.
- \_ Trai phô nhảy tần số là kỹ thuật phân chia giải băng tần thành một tập hợp các kênh hẹp và thực hiện việc truyền tín hiệu trên các kênh đó bằng việc nhảy tuần tự qua các kênh theo một thứ tự nào đó.



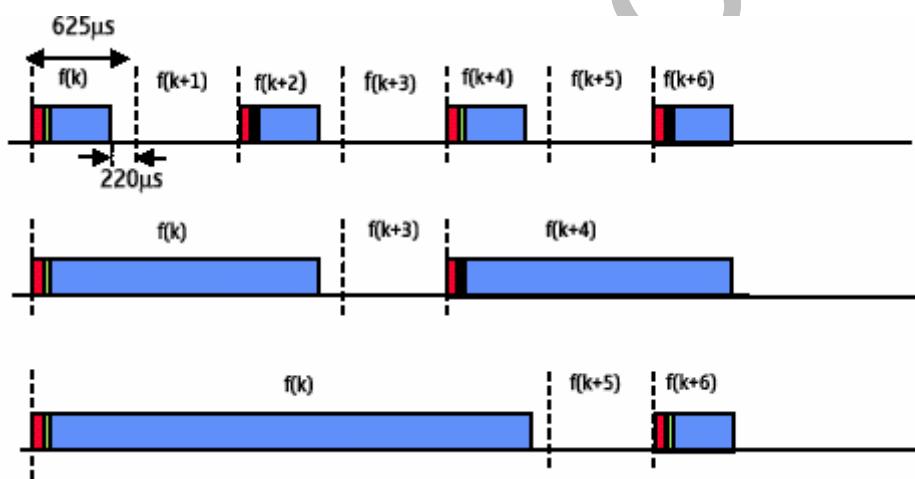
Hình 2-7 Kỹ thuật trai phô nhảy tần số.

#### 2.3.2. Kỹ thuật nhảy tần số trong công nghệ Bluetooth :

- \_ Việc truyền dữ liệu trong Bluetooth được thực hiện bằng sử dụng kỹ thuật nhảy tần số, có nghĩa là các packet được truyền trên những tần số khác nhau. Giải băng tần ISM 2.4Ghz được chia thành 79 kênh, với tốc độ nhảy là 1600 lần trong một giây, điều đó có thể tránh được nhiễu tốt và chiều dài của các packet ngắn lại, tăng tốc độ truyền thông.



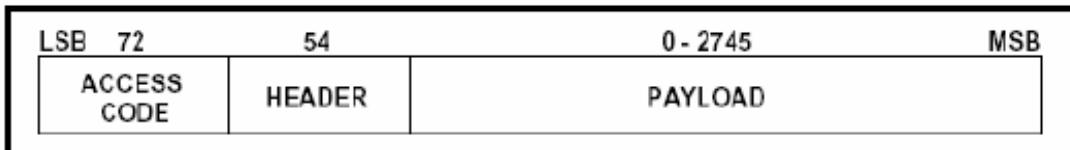
Hình 2-8 Các Packet truyền trên các tần số khác nhau.



Hình 2-9 Các Packet truyền trên khe thời gian.

- Việc truyền nhận sử dụng các khe thời gian. Chiều dài 1 khe thời gian thông thường là 625 $\mu$ s. Một packet thường nằm trong 1 khe đơn, nhưng cũng có thể mở rộng ra 3 hay 5 khe. Với các packet đa khe, yêu cầu tần số phải không đổi cho đến khi toàn bộ packet gửi xong.

- Sử dụng packet đa khe, tốc độ truyền dữ liệu cao hơn nhờ phần header của mỗi packet chỉ đòi hỏi 1 lần  $220\mu s$  (là thời gian chuyển đổi sau mỗi packet). Có thể hiểu ngắn gọn là thời gian truyền 3 packets đơn khe sẽ lớn hơn thời gian truyền 1 packet 3-khe. Bù lại, trong môi trường có nhiều tín hiệu truyền, các packet dài chiếm nhiều timeslot dễ bị nhiễu hơn, do đó dễ bị mất hơn.
- Mỗi packet chứa 3 phần :Access Code (Mã truy cập), Header, Payload.



Hình 2-10 Cấu trúc gói tin Bluetooth

- Kích thước của Access Code và Header là cố định.

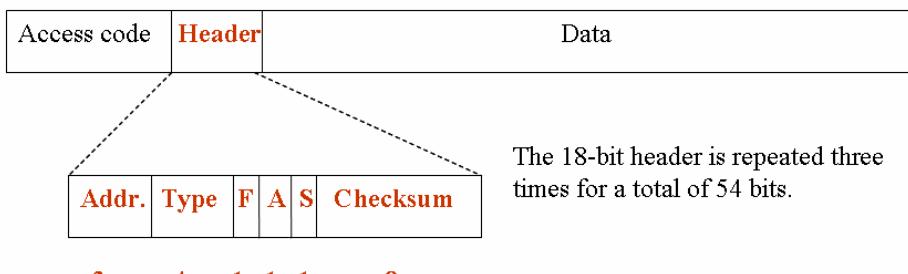
\* **Access code:** Gồm 72 bits, dùng trong việc đồng bộ dữ liệu, định danh, báo hiệu.



Hình 2-11 Access code

\* **Header:**

Bits: 72            54            0-2744



Hình 2-12 Cấu tạo một packet.

- Trong Header có 54 bits, trong đó:
  - + 3 bits được dùng trong việc định địa chỉ, do đó có tối đa 7 Active slave.
  - + 4 bits tiếp theo cho biết loại packet (một số không dùng đến).
  - + 1 bit điều khiển luồng.
  - + 1-bit ARQ : cho biết packet là Broadcast không có ACK.
  - + 1-bit Sequencing : lọc bỏ những packet trùng do truyền lại.
  - + 8 bits HEC : kiểm tra tính toàn vẹn của header.
- Tổng cộng có 18 bits, các bit đó được mã hóa với 1/3 FEC ( Forward Error Correction) để có được 54 bit.
- \* **Payload**: phần chứa dữ liệu truyền đi, có thể thay đổi từ 0 tới 2744 bit/packet. Payload có thể là dữ liệu Voice hoặc data.

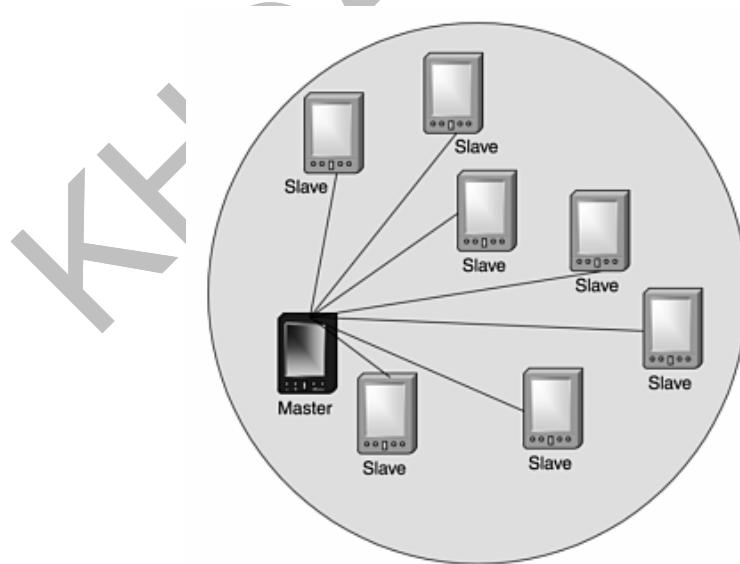
## 2.4. Cách thức hoạt động của Bluetooth.

### 2.4.1. Cơ chế truyền và sửa lỗi :

- Kỹ thuật Bluetooth thực sự là rất phức tạp. Nó dùng kỹ thuật nhảy tần số trong các timeslot (TS), được thiết kế để làm việc trong môi trường nhiều tần số radio, Bluetooth dùng chiến lược nhảy tần để tạo nên sức mạnh liên kết truyền thông và truyền thông thông minh. Cứ mỗi lần gửi hay nhận một packet xong, Bluetooth lại nhảy sang một tần số mới, như thế sẽ tránh được nhiễu từ các tín hiệu khác.
- So sánh với các hệ thống khác làm việc trong cùng băng tần, sóng radio của Bluetooth nhảy tần nhanh và dùng packet ngắn hơn. Vì nhảy nhanh và packet ngắn sẽ làm giảm va chạm với sóng từ lò vi sóng và các phương tiện gây nhiễu khác trong khí quyển.
- Có 3 phương pháp được sử dụng trong việc kiểm tra tính đúng đắn của dữ liệu truyền đi:
  - ▶ Forwad Error Corrrection: thêm 1 số bit kiểm tra vào phần Header hay Payload của packet.

- ▶ Automatic Repeat Request: dữ liệu sẽ được truyền lại cho tới khi bên nhận gửi thông báo là đã nhận đúng.
- ▶ Cyclic Redundancy Check: mã CRC thêm vào các packet để kiểm chứng liệu Payload có đúng không.
- \_ Bluetooth dùng kỹ thuật sửa lỗi tiền FEC (Forward Error Correction) để sửa sai do nhiễu tự nhiên khi truyền khoảng cách xa. FEC cho phép phát hiện lỗi, biết sửa sai và truyền đi tiếp (khác với kỹ thuật BEC-Backward Error Control chỉ phát hiện, không biết sửa, yêu cầu truyền lại).
- \_ Giao thức băng tần cơ sở (Baseband) của Bluetooth là sự kết hợp giữa chuyển mạch và chuyển đổi packet. Các khe thời gian có thể được dành riêng cho các packet phục vụ đồng bộ. Thực hiện bước nhảy tần cho mỗi packet được truyền đi. Một packet trên danh nghĩa sẽ chiếm 1 timeslot, nhưng nó có thể mở rộng chiếm đến 3 hay 5 timeslot.
- \_ Bluetooth hỗ trợ 1 kênh dữ liệu bất đồng bộ, hay 3 kênh tín hiệu thoại đồng bộ nhau cùng một lúc, hay 1 kênh hỗ trợ cùng lúc dữ liệu bất đồng bộ và tín hiệu đồng bộ.

#### 2.4.2. Quá trình hình thành Piconet



Hình 2-13 Mô hình piconet

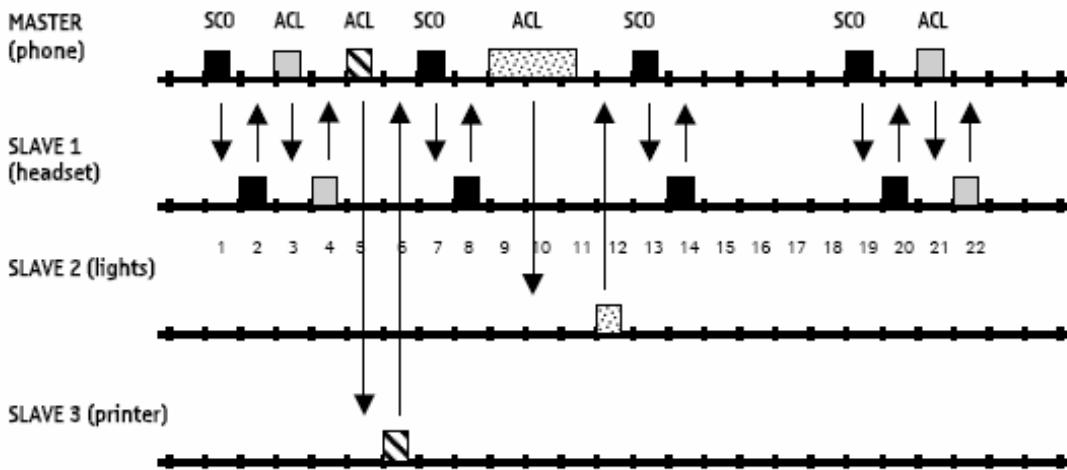
- \_ Một Piconet được tạo bằng 4 cách:
  - ▶ Có Master rồi, Master thực hiện Paging để kết nối với 1 Slave.

- ▶ Một Unit (Master hay Slave) lắng nghe tín hiệu (code) mà thiết bị của nó truy cập được.
  - ▶ Khi có sự chuyển đổi vai trò giữa Master và Slave.
  - ▶ Khi có một Unit chuyển sang trạng thái Active
- \_ Để thiết lập một kết nối mới, tiến trình INQUIRY hay PAGE sẽ bắt đầu. Tiến trình Inquiry cho phép 1 Unit phát hiện các Unit khác trong tầm hoạt động cùng với địa chỉ và đồng hồ của chúng.
- \_ Tiến trình Paging mới thực sự là tạo kết nối. Kết nối chỉ thực hiện giữa những thiết bị mang địa chỉ Bluetooth. Unit nào thiết lập kết nối sẽ phải thực hiện tiến trình paging và tự động trở thành Master của kết nối.
- \_ Trong tiến trình paging, có thể áp dụng vài chiến lược paging. Có một chiến lược paging bắt buộc tất cả các thiết bị Bluetooth đều phải hỗ trợ, chiến lược dùng khi các Unit gặp nhau lần đầu tiên, và trong trường hợp tiến trình paging theo ngay sau tiến trình inquiry. Hai Unit sau khi kết nối nhau dùng chiến lược bắt buộc này, sau đó có thể chọn chiến lược paging khác.
- \_ Sau thủ tục Paging (PAGE), Master thăm dò Slave bằng cách gửi packet POLL thăm dò hay packet NULL rỗng theo như Slave yêu cầu.
- \_ Chỉ có Master gửi tín hiệu POLL cho Slave, ngược lại không có.
- \_ Các vai trò của thiết bị trong Piconet là:
- ▶ Stand by : Không làm gì cả.
  - ▶ Inquiry : Tìm thiết bị trong vùng lân cận.
  - ▶ Paging : Kết nối với 1 thiết bị cụ thể.
  - ▶ Connecting : Nhận nhiệm vụ.



Hình 2-14 Quá trình truy vấn tạo kết nối.

- \_ Mô hình truy vấn các thiết bị trong thực tế:

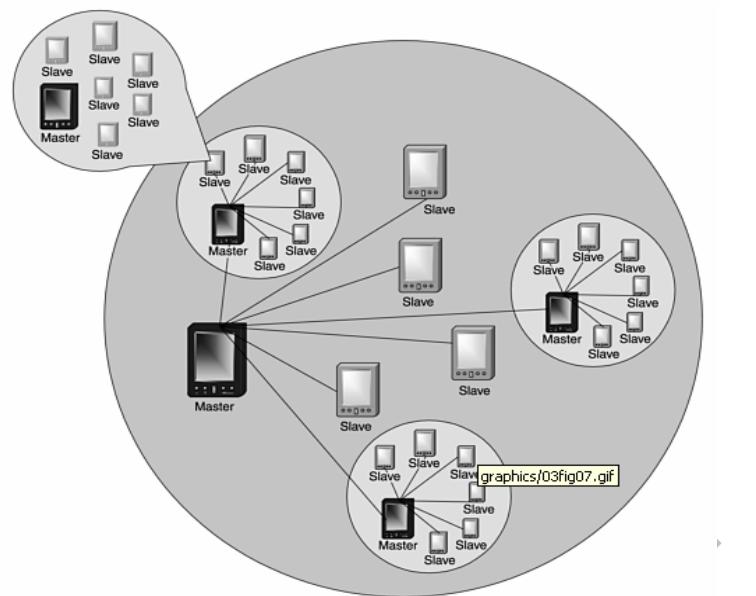


Hình 2-15 Truy vấn tạo kết nối giữa các thiết bị trong thực tế.

- \_ Khi thiết bị tạo paging muốn tạo các kết nối ở các tầng trên, nó sẽ gửi yêu cầu kết nối host theo nghị thức LMP (Link Management Protocol). Khi Unit quản lý host này nhận được thông điệp, nó thông báo cho host biết về kết nối mới. Thiết bị từ xa có thể chấp nhận (gửi thông điệp chấp nhận theo nghị thức LMP) hoặc không chấp nhận kết nối (gửi thông điệp không chấp nhận theo nghị thức LMP).
- \_ Khi thiết bị không yêu cầu bất kỳ thủ tục thiết lập liên kết từ xa nào cả, nó sẽ gửi thông điệp "thiết lập hoàn thành". Thiết bị này vẫn nhận được yêu cầu từ các thiết bị khác. Khi một thiết bị khác đã sẵn sàng tạo liên kết, nó cũng gửi thông điệp "thiết lập hoàn thành". Sau đó 2 thiết bị có thể trao đổi packet trên kênh logic khác với LMP.

#### 2.4.3. Quá trình hình thành Scatternet

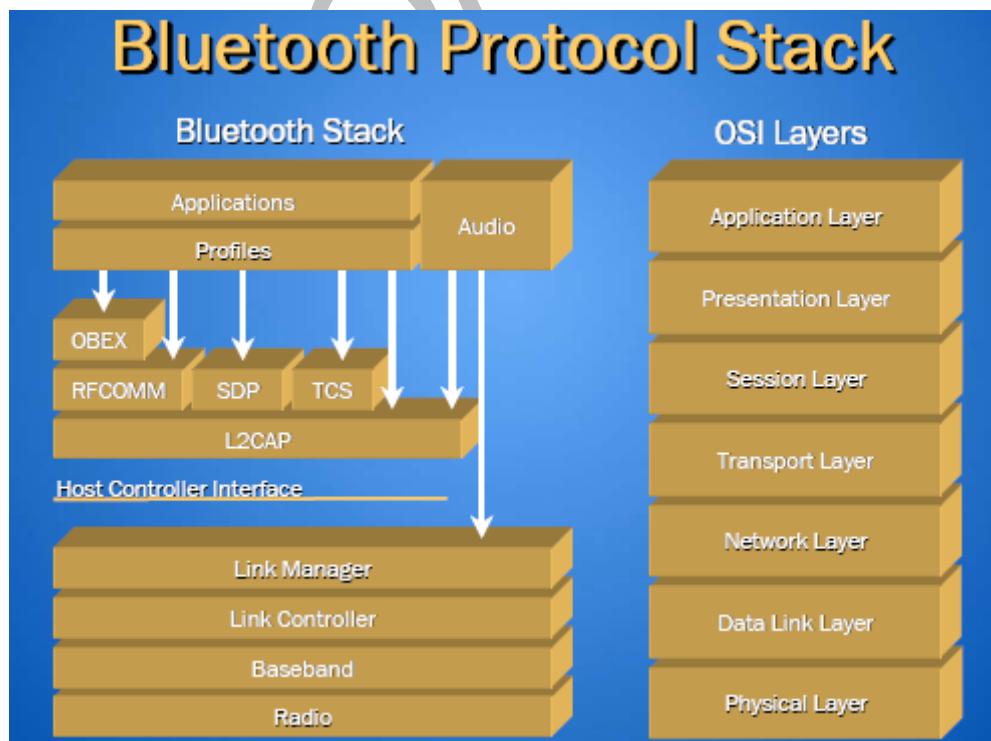
Một Master hay Slave của Piconet này có thể thành Slave của Piconet khác nếu bị Master của piconet khác thực hiện tiến trình paging với nó. Có nghĩa là bất kỳ unit nào cũng có thể tạo 1 Piconet mới bằng cách paging một unit đã là thành viên của một Piconet nào đó. Ngược lại, bất kỳ unit nào tham gia trong 1 Piconet, đều có thể thực hiện paging lên Master hay Slave của Piconet khác. Điều này có thể dẫn đến việc chuyển đổi vai trò giữa Master và Slave trong kết nối mới này.



Hình 2-16 Minh họa một Scatternet.

Các kết nối bên trong một Piconet được thiết lập thông qua các unit chia sẻ, unit này thuộc về 2 hay nhiều Piconet, nó dùng kỹ thuật phân chia thời gian để chuyển đổi qua lại giữa các Piconet.

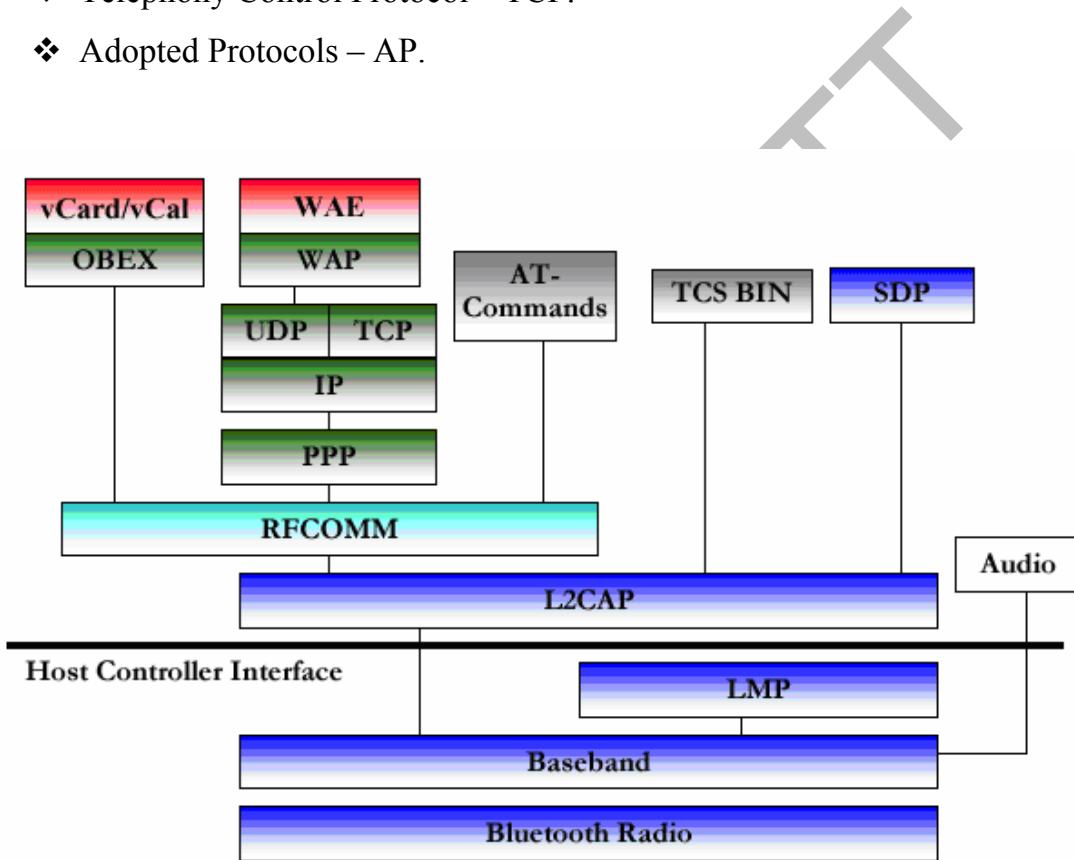
## 2.5. Các tầng giao thức trong Bluetooth.



Hình 2-17 Bluetooth Protocol Stack

Các giao thức cốt lõi trong Bluetooth:

- ❖ Bluetooth Radio
- ❖ Baseband.
- ❖ Link Manager Protocol – LMP.
- ❖ Logical Link Control and Adaptation Protocol – L2CAP.
- ❖ Radio Frequency Communication – RFCOMM.
- ❖ Service Discovery Protocol – SDP.
- ❖ Telephony Control Protocol – TCP.
- ❖ Adopted Protocols – AP.



Hình 2-18 Các tầng nghi thức Bluetooth.

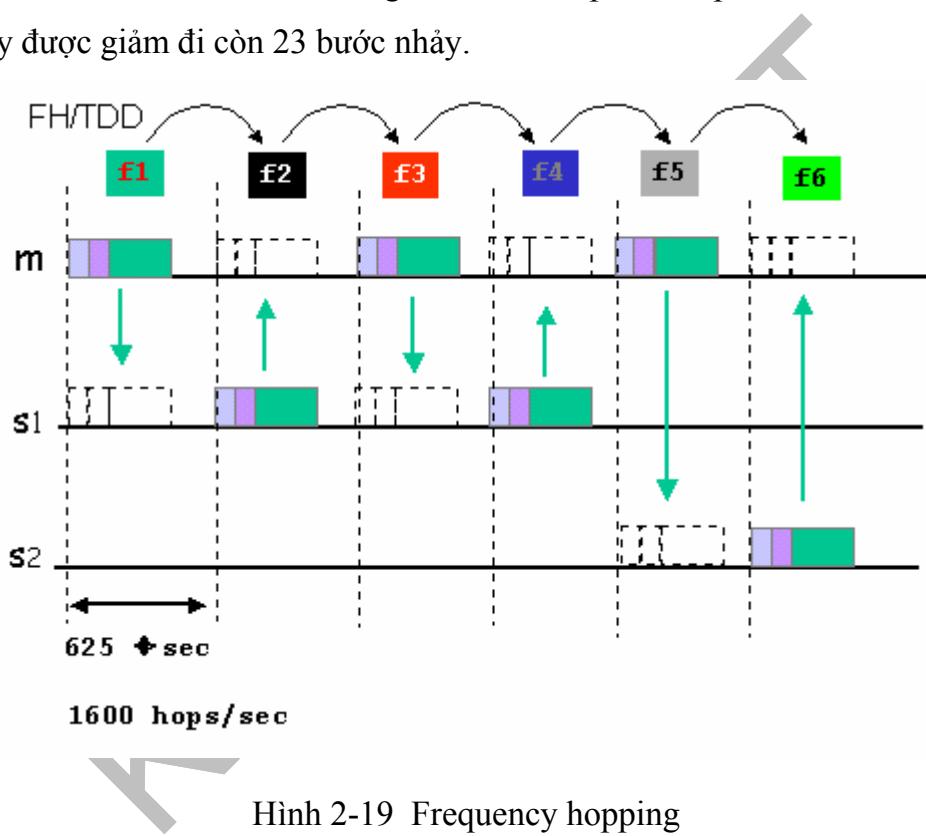
### 2.5.1. Bluetooth Radio:

Tầng Bluetooth radio là tầng thấp nhất được định nghĩa trong đặc tả Bluetooth. Nó định nghĩa những yêu cầu cho bộ phận thu phát sóng hoạt động ở tần số 2.4GHz ISM (Industrial, Scientific, and Medical). Băng tần ISM là

băng tần không cần đăng kí được dành riêng để dùng cho các thiết bị không dây trong công nghiệp, khoa học và y tế.

Nhờ giao tiếp bằng sóng radio mà dữ liệu Bluetooth có thể xuyên qua các vật thể rắn và phi kim.

Sóng radio của Bluetooth được truyền đi bằng cách nhảy tần số (frequency hopping), có nghĩa là mọi packet được truyền trên những tần số khác nhau. Tốc độ nhảy nhanh giúp tránh nhiễu tốt. Hầu hết các nước dùng 79 bước nhảy, mỗi bước nhảy cách nhau 1MHz, bắt đầu ở 2.402GHz và kết thúc ở 2.480GHz. Ở một vài nước, chẳng hạn như Pháp, Nhật, phạm vi của dải băng tần này được giảm đi còn 23 bước nhảy.



Hình 2-19 Frequency hopping

Bluetooth được thiết kế để hoạt động ở mức năng lượng rất thấp. Đặc tả đưa ra 3 mức năng lượng từ 1mW tới 100 mW

- Mức năng lượng 1 (100mW): Được thiết kế cho những thiết bị có phạm vi hoạt động rộng (~100m)
- Mức năng lượng 2 (2.5mW): Cho những thiết bị có phạm vi hoạt động thông thường (~10m)
- Mức năng lượng 3 (1mW): Cho những thiết bị có phạm vi hoạt động ngắn (~10cm)

Những thiết bị có khả năng điều khiển mức năng lượng có thể tối ưu hóa năng lượng bằng cách dùng những lệnh LMP (Link Manager Protocol).

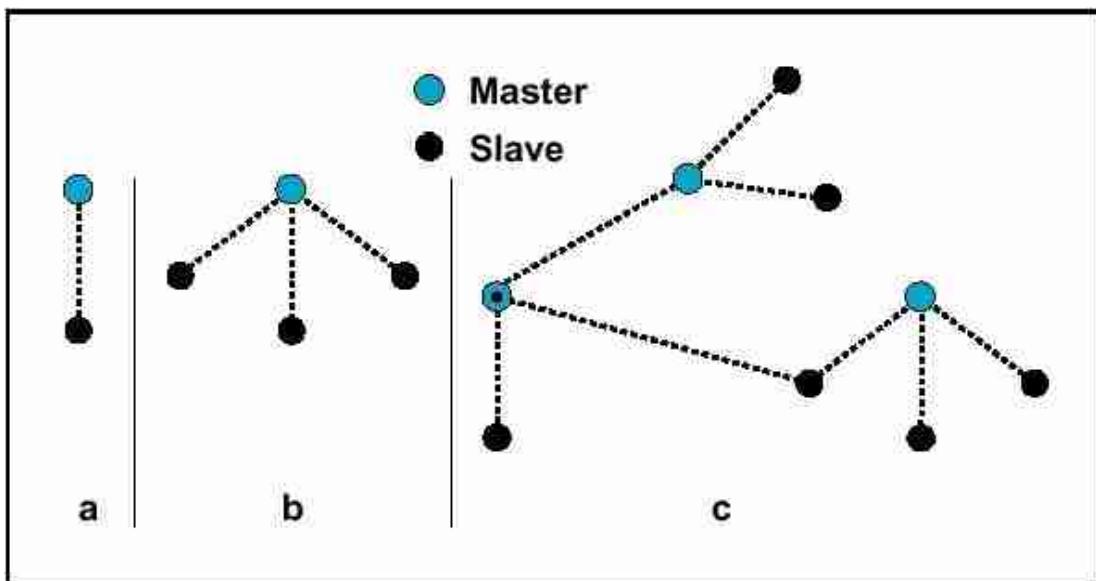
### 2.5.2. BaseBand:

Baseband protocol nằm ở tầng vật lý của Bluetooth. Nó quản lý những kênh truyền và liên kết vật lý tách biệt khỏi những dịch vụ khác như sửa lỗi, chọn bước nhảy và bảo mật. Tầng Baseband nằm bên trên tầng radio trong ch่อง giao thức của Bluetooth. Baseband protocol được cài đặt như là một Link Controller. Nó cùng với Link Manager thực hiện những công việc ở mức thấp như kết nối, quản lý năng lượng. Tầng Baseband cũng quản lý những kết nối đồng bộ và không đồng bộ, quản lý các gói tin, thực hiện tìm kiếm và yêu cầu kết nối đến các thiết bị Bluetooth khác.

#### 2.5.2.1. Network topology

Hai hoặc nhiều thiết bị kết nối với nhau tạo thành một **piconet**. Các thiết bị kết nối theo kiểu ad-hoc nghĩa là kiểu mạng được thiết lập chỉ cho nhu cầu truyền dữ liệu hiện hành và tức thời, sau khi dữ liệu truyền xong, mạng sẽ tự hủy. Trong một piconet, một thiết bị đóng vai trò là Master (thường là thiết bị đầu tiên tạo kết nối), các thiết bị sau đó đóng vai trò là Slave. Một piconet chỉ có duy nhất 1 Master, Master thiết lập đồng hồ đếm xung để đồng bộ các thiết bị trong cùng piconet mà nó đóng vai trò là Master. Master cũng quyết định số kênh truyền thông. Tất cả các thiết bị còn lại trong piconet, nếu không là Master thì phải là Slave. Chú ý: không cho phép truyền thông trực tiếp giữa Slave – Slave.

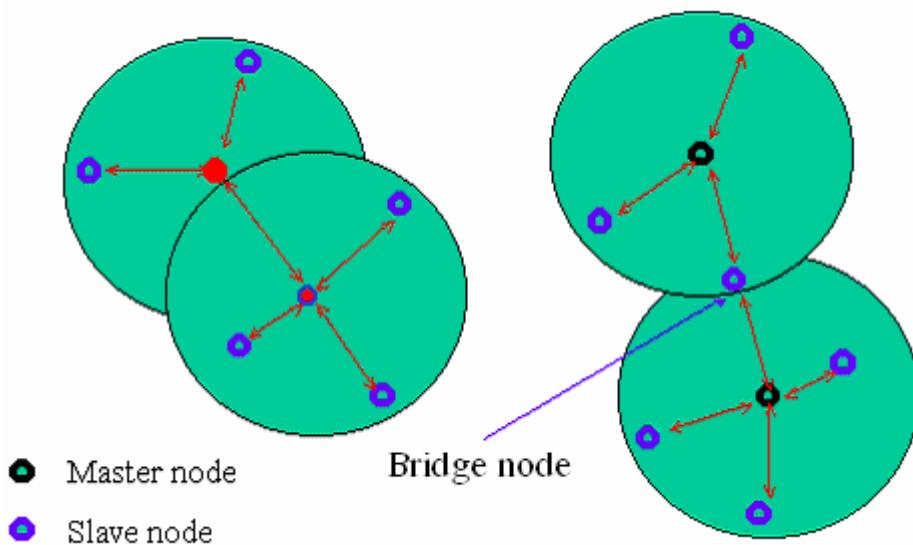
Vai trò Master trong 1 piconet không cố định, ví dụ khi Master không đủ tài nguyên phục vụ cho piconet, nó sẽ giao quyền lại cho một Slave “giàu có” hơn làm Master, còn nó làm Slave.



Hình 2-20 Piconet

Khi có 2 hay nhiều piconet kết hợp lại truyền thông với nhau, ta có một **scatternet**. Có 2 loại scatternet:

- Một Slave trong piconet này cũng là Slave trong piconet kia. Khi này các piconet độc lập với nhau và không đồng bộ. Khi có nhiều piconet độc lập, có thể bị nhiễu trên một số kênh, một số packet sẽ bị mất và được truyền lại. Nếu tín hiệu là tiếng nói (tín hiệu thoại), chúng sẽ bị bỏ qua.
- Một Slave trong piconet này là Master trong piconet khác. Khi này 2 piconet đồng bộ nhau về clock (xung nhịp) và hopping (khoảng nhảy tần số) vì Slave đóng vai trò Master trong piconet mới sẽ mang theo clock và hopping của piconet cũ, đồng bộ cho các Slave trong piconet mới mà nó làm Master.



Hình 2-21 Scatternet

### 2.5.2.2. Liên kết SCO và ACL

Tầng Baseband quản lý 2 dạng kết nối:

**SCO link** (Synchronous Connection Oriented) là một kết nối đối xứng point-to-point giữa một Master và một Slave trong 1 piconet. Kết nối SCO chủ yếu dùng để truyền dữ liệu tiếng nói. Master có thể hỗ trợ tối đa 3 kết nối SCO đồng thời. SCO packet không chứa CRC (Cyclic Redundancy Check) và không bao giờ truyền lại. Liên kết SCO được thiết lập chỉ sau khi 1 liên kết ACL đầu tiên được thiết lập.

**ACL Link** (Asynchronous Connectionless Link) là một kết nối point-to-multipoint giữa Master và tất cả các Slave tham gia trong piconet. Chỉ tồn tại duy nhất một kết nối ACL. Hầu hết các ACL packet đều có thể truyền lại.

### 2.5.2.3. Địa chỉ thiết bị

Có 4 loại địa chỉ khác nhau có thể gán cho một thiết bị Bluetooth:

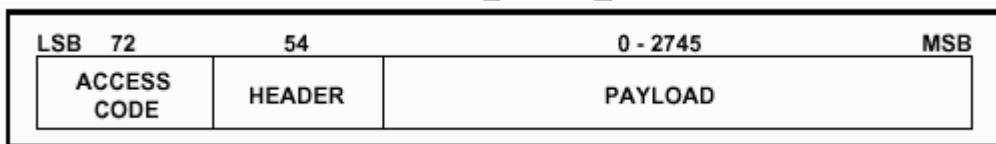
**BD\_ADDR, AM\_ADDR, PM\_ADDR, AR\_ADDR.**

- **BD\_ADDR: Bluetooth Device Address.** Là 48 bit địa chỉ MAC theo tiêu chuẩn IEEE quy định (Giống như địa chỉ MAC trên mỗi card mạng), xác định duy nhất 1 thiết bị Bluetooth trên toàn cầu, trong đó 3 byte cho nhà sản xuất thiết bị và 3 byte cho sản phẩm.

- **AM\_ADDR: Active Member Address.** Nó còn gọi là địa chỉ MAC (Media Access Control) của một thiết bị Bluetooth. Nó là một con số 3 bit dùng để phân biệt giữa các active slave tham gia trong 1 piconet.  $2^3 = 8$  nên có tối đa 7 Slave active trong 1 piconet, còn 000 là địa chỉ Broadcast (truyền đến tất cả các thành viên trong piconet). Địa chỉ này chỉ tồn tại khi Slave ở trạng thái active.
- **PM\_ADDR: Parked Member Address.** Là một con số 8 bit, phân biệt các parked Slave. Do đó có tối đa 255 thiết bị ở trạng thái parked. Địa chỉ này chỉ tồn tại khi Slave ở trạng thái parked.
- **AR\_ADDR: Access Request Address.** Địa chỉ này được dùng bởi parked Slave để xác định nơi mà nó được phép gửi thông điệp yêu cầu truy cập tới.

#### 2.5.2.4. Định dạng gói tin

Mỗi gói tin bao gồm 3 phần là **Access code** (72 bits), **header** (54 bits) và **payload** (0-2745 bits)



Hình 2-22 Định dạng gói tin Bluetooth

**Access code:** Dùng để đồng bộ hóa, dùng trong quá trình tìm kiếm thiết bị và yêu cầu kết nối. Có 3 loại khác nhau của Access code: Channel Access Code (CAC), Device Access Code (DAC) and Inquiry Access Code (IAC). CAC dùng để xác định một piconet duy nhất, DAC dùng để thực hiện yêu cầu kết nối, IAC dùng để thực hiện tìm kiếm thiết bị.

**Header:** Chứa một số thông tin về packet như thứ tự của packet, địa chỉ đích, kiểm lỗi, v.v...

#### 2.5.2.5. Quản lý trạng thái

Có 4 trạng thái chính của một thiết bị Bluetooth trong một piconet:

- **Inquiring device (inquiry mode):** Thiết bị đang phát tín hiệu tìm những thiết bị Bluetooth khác.

- **Inquiry scanning device (inquiry scan mode):** Thiết bị nhận tín hiệu inquiry của inquiry device và trả lời
- **Paging device (page mode):** Thiết bị phát tín hiệu yêu cầu kết nối với thiết bị đã inquiry từ trước.
- **Page scanning device (page scan mode):** Thiết bị nhận yêu cầu kết nối từ paging device và trả lời.

#### 2.5.2.6. Thiết lập kết nối

##### 2.5.2.6.1. Hình thành piconet

Một piconet được tạo bằng 4 cách:

- Có Master rồi, Master thực hiện paging để kết nối với 1 Slave
- Một unit (Master hay Slave) lắng nghe tín hiệu mà thiết bị của nó truy cập được (scanning)
- Khi có sự chuyển đổi vai trò giữa Master và Slave
- Khi có một unit chuyển sang trạng thái active

Để thiết lập một kết nối mới, tiến trình Inquiry và Paging sẽ bắt đầu. Tiến trình Inquiry cho phép 1 unit phát hiện các units trong tầm hoạt động cùng với địa chỉ và đồng hồ của chúng. Sau khi Inquiry, thiết bị thực hiện tiếp tiến trình Paging để thiết lập kết nối, sau khi được page scanning device chấp nhận kết nối mới thực sự được thiết lập.

Unit nào thiết lập kết nối sẽ phải thực hiện tiến trình paging và tự động trở thành Master của kết nối.

Sau thủ tục paging (PAGE), Master thăm dò Slave bằng cách gửi packet POLL thăm dò hay packet NULL rỗng.

Chỉ có Master gửi tín hiệu POLL cho Slave, ngược lại không có. Khi thiết bị tạo paging muốn tạo các kết nối ở các tầng trên LM (link manager), nó sẽ gửi yêu cầu kết nối host theo nghị thức LMP (Link Manager Protocol). Khi unit quản lý host này nhận được thông điệp, nó thông báo cho host biết về kết nối mới. Thiết bị từ xa có thể chấp nhận (gửi thông điệp chấp nhận theo nghị thức LMP) hoặc không chấp nhận kết nối (gửi thông điệp không chấp nhận theo nghị thức LMP). Sau đó 2 thiết bị có thể trao đổi dữ liệu với nhau.

### 2.5.2.6.2. Hình thành scatternet

Một Master/Slave của piconet này có thể thành Slave của piconet khác nếu bị Master của piconet khác thực hiện tiến trình paging với nó. Có nghĩa là bất kỳ unit nào cũng có thể tạo 1 piconet mới bằng cách paging một unit đã là thành viên của một piconet nào đó.

Ngược lại, bất kỳ unit nào tham gia trong 1 piconet, đều có thể thực hiện paging lên Master/Slave của piconet khác. Điều này có thể dẫn đến việc chuyển đổi vai trò giữa Master và Slave trong kết nối mới này.

### 2.5.2.7. Các chế độ kết nối:

- **Active mode:** Trong chế độ này, thiết bị Bluetooth tham gia vào hoạt động của mạng. Thiết bị master sẽ điều phối lưu lượng và đồng bộ hóa cho các thiết bị slave.
- **Sniff mode:** là 1 chế độ tiết kiệm năng lượng của thiết bị đang ở trạng thái active. Ở Sniff mode thiết bị slave lắng nghe tín hiệu từ mạng với tần số giảm hay nói cách khác là giảm công suất. Tần số này phụ thuộc vào tham số của ứng dụng. Đây là chế độ ít tiết kiệm năng lượng nhất trong 3 chế độ tiết kiệm năng lượng.
- **Hold mode:** là 1 chế độ tiết kiệm năng lượng của thiết bị đang ở trạng thái active. Master có thể đặt chế độ Hold mode cho slave của mình. Các thiết bị có thể trao đổi dữ liệu ngay lập tức khi thoát khỏi chế độ Hold mode. Đây là chế độ tiết kiệm năng lượng trung bình trong 3 chế độ tiết kiệm năng lượng
- **Park mode:** là 1 chế độ tiết kiệm năng lượng của thiết bị vẫn còn trong mạng nhưng không tham gia trong qua trình trao đổi dữ liệu (inactive). Thiết bị ở chế độ Park mode bỏ địa chỉ MAC, chỉ lắng nghe tín hiệu đồng bộ hóa và thông điệp broadcast của Master. Đây là chế độ tiết kiệm năng lượng nhất trong 3 chế độ tiết kiệm năng lượng.

### 2.5.2.8. Những chức năng khác của Baseband

- Sửa lỗi
- Quản lý lưu lượng dữ liệu: Baseband dùng cấu trúc dữ liệu FIFO trong việc truyền và nhận dữ liệu

- Đồng bộ hóa
- Bảo mật

### 2.5.3. Link Manager Protocol:

Link Manager (LM) thực hiện việc thiết lập kênh truyền, xác nhận hợp lệ, cấu hình kênh truyền. Nó tìm kiếm những LM khác và giao tiếp với chúng thông qua Link Manager Protocol. Để thực hiện được vai trò của mình, LM dùng những dịch vụ do tầng Link Controller bên dưới cung cấp.

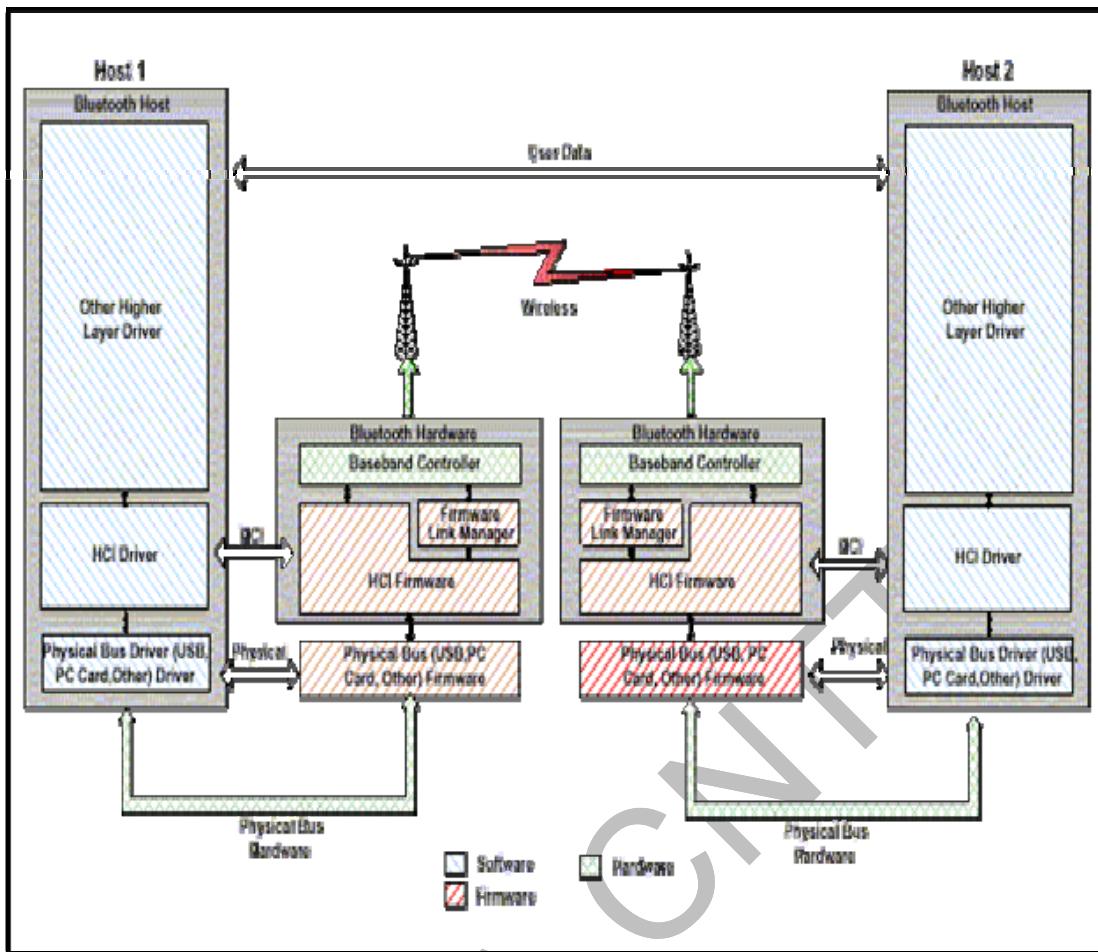
Về cơ bản, các lệnh LMP bao gồm các PDU (Protocol Data Unit – Xem thêm trong phần SDP bên dưới) được gửi từ thiết bị này sang thiết bị khác.

### 2.5.4. Host Controller Interface:

HCI cung cấp một giao diện cho phép các tầng bên trên điều khiển Baseband Controller và Link Manager, đồng thời cho phép truy cập đến trạng thái của phần cứng và các thanh ghi điều khiển. Về bản chất, giao diện này cung cấp một phương thức duy nhất để truy cập đến những khả năng của băng tần cơ sở. HCI tồn tại trong 3 phần: Host – Transport layer – Host controller. Mỗi phần đóng một vai trò khác nhau trong hệ thống HCI.

#### 2.5.4.1. Những thành phần chức năng của HCI

Về mặt chức năng, HCI được chia thành 3 phần riêng biệt là HCI firmware, HCI driver và Host controller transport layer. Hình sau mô tả mô hình hoạt động của các thành phần HCI.

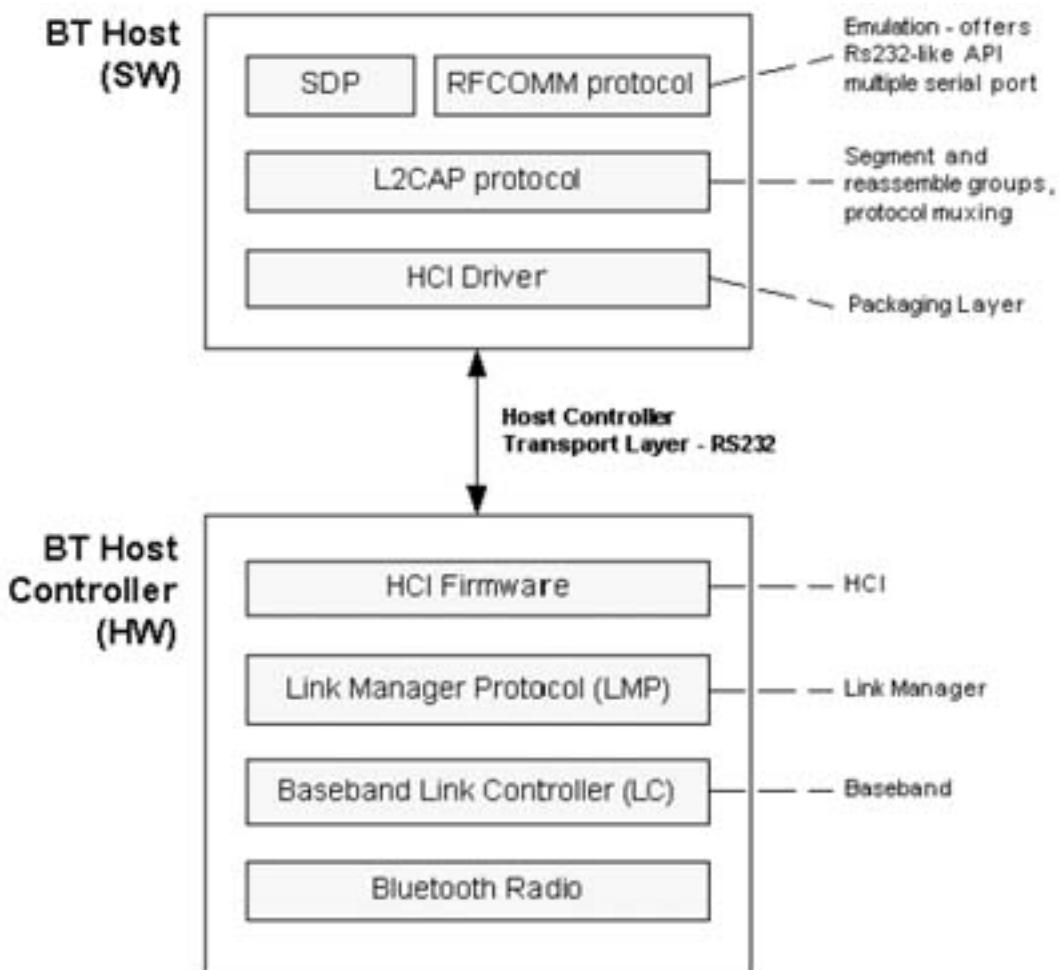


Hình 2-23 Host Controller Interface

**HCI firmware:** nằm ở Host Controller (tức là nằm ở phần cứng của Bluetooth). HCI firmware cung cấp các lệnh HCI cho phần cứng Bluetooth bằng cách truy cập các lệnh ở tầng Baseband, Link Manager.

**HCI driver:** Năm ở phần Host (tức là phần mềm). Khi có sự kiện xảy ra, một HCI event sẽ được gửi đến Host và Host sẽ phân tích gói tin nhận được để xác định xem sự kiện nào đã xảy ra, sau đó nó sẽ chuyển các gói tin lên các tầng bên trên.

**Host controller transport layer:** HCI driver và firmware giao tiếp với nhau thông qua Host controller transport layer. Có nhiều loại transport layer như: **USB**, **UART** và **RS232**. Nhờ vào Host controller transport layer mà phần cứng và phần mềm có thể trao đổi dữ liệu mà không cần biết về cách thức dữ liệu được trao đổi



Hình 2-24 Host controller transport layer

#### 2.5.4.2. Các lệnh HCI

HCI cung cấp các lệnh cho phép truy cập tới phần cứng Bluetooth. Các lệnh HCI Link cho phép phần Host khả năng điều khiển việc kết nối đến các thiết bị Bluetooth khác. Những lệnh này cần đến Link Manager để trao đổi các lệnh LMP với các thiết bị Bluetooth khác.

#### 2.5.4.3. Các sự kiện, mã lỗi, luồng dữ liệu HCI

##### Luồng dữ liệu

Luồng dữ liệu từ Host đến Host Controller được điều khiển để ngăn ngừa việc bộ đệm của Host Controller bị tràn bởi dữ liệu ACL gửi đến các thiết bị khác mà không có phản hồi. Phần Host sẽ quản lý bộ đệm của Host Controller.

##### Các sự kiện HCI

Có nhiều sự kiện được định nghĩa cho tầng HCI. Các sự kiện này cung cấp một phương thức để trả về các tham số và dữ liệu gắn với mỗi sự kiện. Cho đến nay đã có 32 sự kiện HCI được cài đặt.

### **Mã lỗi HCI**

Một số lượng lớn các mã lỗi đã được định nghĩa cho tầng HCI. Khi một lệnh thất bại, các mã lỗi sẽ được trả về để cho biết nguyên nhân phát sinh lỗi. Có 35 mã lỗi HCI khác nhau đã được định nghĩa.

#### **2.5.4.4. Host Controller Transport Layer**

##### **UART Transport Layer**

Mục tiêu của HCI UART Transport Layer là cho phép dùng Bluetooth HCI thông qua giao diện serial giữa 2 UART. UART là viết tắt của từ Universal Asynchronous Receiver – Transmitter. UART dùng để truyền và nhận tín hiệu thông qua giao tiếp serial không đồng bộ.

##### **RS232 Transport Layer**

Mục tiêu của HCI RS232 Transport Layer là cho phép dùng Bluetooth HCI thông qua giao diện RS232 giữa Bluetooth Host và Bluetooth Host Controller. RS232 là một chuẩn công nghiệp về truyền nhận dữ liệu thông qua cổng serial.

##### **USB Transport Layer**

Mục tiêu của HCI Universal Serial Bus (USB) Transport Layer là cho phép dùng giao diện USB cho phần cứng Bluetooth.

#### **2.5.5. Logical link control and adaption protocol (L2CAP):**

L2CAP nằm trên giao thức băng tần cơ sở (Baseband protocol) và nằm ở tầng Data Link. L2CAP cung cấp những dịch vụ hướng kết nối (connection-oriented) và phi kết nối (connectionless) cho những tầng giao thức bên trên. L2CAP có khả năng phân kênh (multiplexing), phân đoạn (segmentation), tái tổ hợp (reassembly operation). L2CAP cho phép những giao thức ở tầng cao hơn và những ứng dụng truyền, nhận những dữ liệu. Mỗi gói dữ liệu L2CAP tối đa 64 kilobytes.

#### **2.5.5.1. Những yêu cầu chức năng của L2CAP**

### **Phân kênh giao thức (Protocol Multiplexing)**

L2CAP phải hỗ trợ phân kênh giao thức bởi vì Baseband Protocol không hỗ trợ việc xác định các giao thức ở tầng cao hơn. L2CAP phải có khả năng phân biệt những giao thức ở tầng bên trên như Service Discovery Protocol, RFCOMM, Telephony Control.

### **Phân đoạn và tái tổ hợp**

So với những phương tiện truyền thông dùng dây khác thì những gói dữ liệu được định nghĩa bởi Baseband Protocol bị giới hạn kích thước. Những gói tin lớn phải được L2CAP chia nhỏ thành nhiều gói tin Baseband trước khi được truyền đi. Tương tự, những gói tin Baseband nhận được sẽ được tái tổ hợp thành một gói tin duy nhất kèm theo việc kiểm tra toàn vẹn dữ liệu. Chức năng phân đoạn và tái tổ hợp thật sự cần thiết để hỗ trợ những giao thức dùng những gói tin lớn hơn gói tin được hỗ trợ bởi Baseband.

#### **2.5.5.2. Những đặc điểm khác của L2CAP**

##### **Định dạng gói tin**

Các gói tin L2CAP được truyền dẫn dựa trên “kênh” (channel). Một kênh đại diện cho một luồng dữ liệu. Các kênh có thể là hướng kết nối (connection-oriented) hoặc phi kết nối (connectionless). Tất cả các gói tin được lưu trữ dưới dạng Little Endian.

##### **Các tùy chọn tham số cấu hình**

Các tùy chọn là cơ chế để mở rộng khả năng điều phối các yêu cầu kết nối. Các tùy chọn được truyền đi dưới dạng một tập hợp những thành phần bao gồm kiểu tùy chọn, độ dài tùy chọn, và dữ liệu.

##### **Các dịch vụ**

Nhiều dịch vụ được cung cấp bởi L2CAP. Chúng bao gồm các phần:

- Connection: Thiết lập, cấu hình, hủy kết nối
- Data: Đọc, ghi
- Group: Tạo, đóng, thêm thành viên, hủy thành viên
- Information: Ping, lấy thông tin
- Connection-less traffic: Cho phép, ngăn cấm

### 2.5.6. RFCOMM Protocol:

Giao thức RFCOMM cho phép giả lập cổng serial thông qua giao thức L2CAP. Giao thức này dựa trên chuẩn ETSI TS 07.10. Chỉ có một phần của chuẩn TS 07.10 được dùng và được chỉnh sửa cho phù hợp với Bluetooth.

RFCOMM hỗ trợ tối đa 60 kết nối cùng một lúc giữa 2 thiết bị Bluetooth. Số kết nối tối đa tùy thuộc vào nhà sản xuất. Đối với RFCOMM, một kết nối bao gồm 2 ứng dụng chạy trên 2 thiết bị riêng biệt (2 thiết bị đầu cuối).

**Loại thiết bị:** Về cơ bản, RFCOMM cung cấp cho 2 loại thiết bị:

- Loại thiết bị 1 là những đầu cuối như máy tính hay máy in.
- Loại thiết bị 2 là những thành phần dùng để truyền dữ liệu, chẳng hạn modem.

**Tín hiệu điều khiển:** RFCOMM giả lập 9 mạch của chuẩn RS232, 9 mạch đó là:

Pin Circuit Name
102 Signal Common
103 Transmit Data (TD)
104 Received Data (RD)
105 Request to Send (RTS)
106 Clear to Send (CTS)
107 Data Set Ready (DSR)
108 Data Terminal Ready (DTR)
109 Data Carrier Detect (CD)

### 125 Ring Indicator (RI)

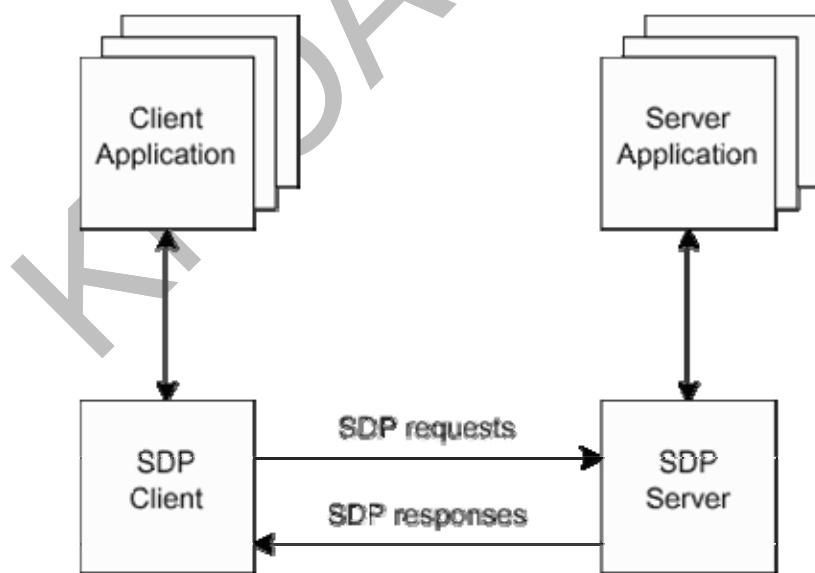
**Nhiều cổng nối tiếp giả lập:** 2 thiết bị Bluetooth dùng RFCOMM trong giao tiếp giữa chúng có thể mở nhiều cổng nối tiếp (serial port). RFCOMM hỗ trợ tối đa 60 cổng, tuy nhiên số cổng có thể dùng trong một thiết bị tùy thuộc vào nhà sản xuất.

#### 2.5.7. Service Discovery Protocol:

SDP cho phép các ứng dụng tìm kiếm những dịch vụ và thuộc tính của các dịch vụ có trong một thiết bị Bluetooth. SDP. Điều này rất cần thiết bởi vì các dịch vụ mà một thiết bị Bluetooth cung cấp sẽ thay đổi tùy theo mỗi thiết bị.

##### 2.5.7.1. Thiết lập giao thức SDP

SDP là một giao thức đơn giản với những yêu cầu tối thiểu về việc truyền dẫn bên dưới. SDP dùng mô hình request / response với mỗi giao tác bao gồm một request protocol data unit (PDU) và một response PDU. Client sẽ gửi yêu cầu đến server, và server sẽ trả lời ngược lại client.



**Định dạng PDU:** Mỗi PDU bao gồm 1 PDU header, sau đó là các tham số PDU. Phần header bao gồm 3 trường:

- PDU ID: Xác định loại PDU cho biết ý nghĩa của nó và nó chứa loại tham số nào.
- Transaction ID: Dùng để xác định duy nhất một request PDU và dùng để ánh xạ giữa response PDU và request PDU.
- Parameter length: Cho biết độ dài (tính bằng byte) của tất cả tham số chứa trong PDU.

**Partial response và continuation state:** Vài request PDU có thể yêu cầu các phản hồi có kích thước lớn hơn 1 response PDU. Trong trường hợp này, SDP server sẽ phát sinh các partial response chứa một phần của phản hồi, đồng thời kèm theo mỗi partial response là một continuation state cho biết phản hồi đó còn nhiều phần nữa.

**Quản lý lỗi:** Trong trường hợp request PDU gửi đến server bị lỗi thì server sẽ phản hồi bằng một error PDU.

#### 2.5.7.2. Các dịch vụ SDP

**Service record:** Tất cả các thông tin về một dịch vụ được chứa trong một service record. Service record chứa một danh sách các thuộc tính của dịch vụ (service attribute)

**Service attribute:** Mỗi service attribute mô tả một thuộc tính của dịch vụ. Mỗi service attribute bao gồm 2 thành phần: attribute ID và attribute value. Attribute ID là một số nguyên không dấu 16 bit xác định duy nhất một thuộc tính trong một service record. Attribute value có độ dài không cố định chứa giá trị của thuộc tính. Trong giao thức SDP, attribute value được thể hiện bằng một phần tử dữ liệu (data element).

**Service class:** Mỗi dịch vụ là một thể hiện của một lớp dịch vụ (service class). Lớp dịch vụ cung cấp các định nghĩa cho tất cả thuộc tính chứa trong service record. Mỗi định nghĩa thuộc tính cho biết giá trị của attribute ID, mục đích sử dụng của attribute value, định dạng của attribute value. Mỗi lớp dịch vụ được gán một con số định danh duy nhất, được gọi là UUID (Universal Unique Identifier).

#### 2.5.7.3. Tìm kiếm dịch vụ

Nhiệm vụ chính của SDP là cho phép một thiết bị Bluetooth tìm kiếm xem các thiết bị Bluetooth khác có cung cấp những dịch vụ nào. SDP cho phép làm điều này bằng nhiều cách: **Searching**, tìm kiếm một dịch vụ cụ thể, hoặc **Browsing**, lấy những dịch vụ đang được cung cấp.

#### 2.5.7.3.1. Tìm kiếm dịch vụ cụ thể (Searching for Service)

Phiên tìm kiếm dịch vụ cho phép một client tìm được Service record cụ thể trên server dựa trên các giá trị của thuộc tính trong những record này.

Client không có khả năng tìm service record dựa trên các thuộc tính có giá trị tùy tiện. Nói đúng hơn là Client chỉ có thể tìm các thuộc tính dựa trên Universally Unique Identifiers (UUIDs). Một mẫu tìm kiếm dịch vụ thường so sánh với một danh sách các UUIDs (những thuộc tính của dịch vụ) để tìm ra dịch vụ mà nó cần.

#### 2.5.7.3.2. Duyệt dịch vụ (Browsing for service)

Tiến trình này lấy tất cả các dịch vụ mà nó được phép duyệt. Trong SDP, Client dựa trên một thuộc tính được tất cả các lớp dịch vụ chia sẻ. Thuộc tính này được gọi là BrowseGroupList. Giá trị của các thuộc tính này gồm một danh sách các UUIDs, mỗi UUID đại diện cho một BrowseGroup dùng cho mục đích duyệt service.

Khi Client duyệt lướt qua các dịch vụ của SDP Server, nó tạo một mẫu tìm dịch vụ chứa các UUID đại diện cho BrowseGroup. Tất cả các dịch vụ được duyệt có giá trị UUID giống với giá trị của thuộc tính trong BrowseGroupList.

#### 2.5.7.4. Data element

Trong giao thức SDP, một thuộc tính được xem như là một phần tử dữ liệu (data element). Một phần tử dữ liệu bao gồm 2 trường: trường header và trường dữ liệu. Trường header gồm 2 phần: phần mô tả kiểu và phần mô tả kích thước. Trường dữ liệu có kích thước và kiểu như đã được mô tả trong phần header.

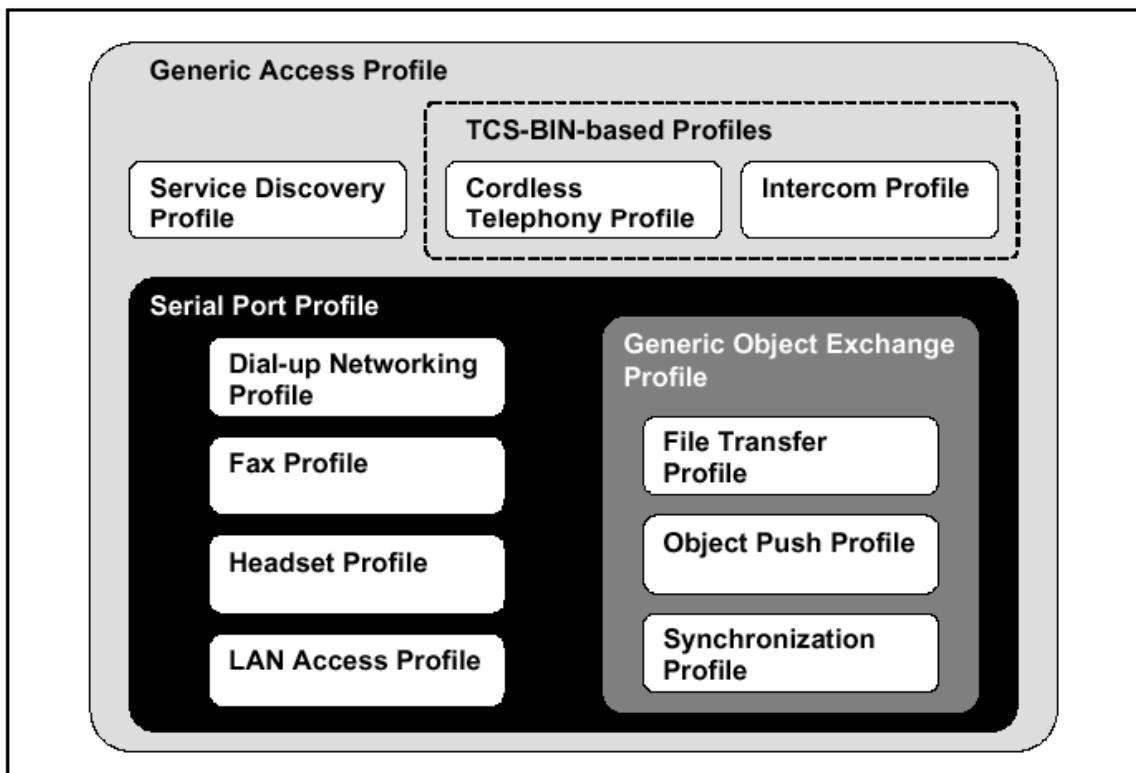
## 2.6. Bluetooth Profiles:

- \_ Tổ chức SIG (The Bluetooth Special Interest Group ) đã định nghĩa một số mô hình sử dụng công nghệ Bluetooth. Họ vạch ra những ứng dụng chính về Bluetooth và những thiết bị trong tương lai, ví dụ như sự đồng bộ hóa giữa thiết bị cầm tay và PC, và kết nối không dây với Internet bằng một điện thoại di động hoặc một cordless modem.
- \_ Profile chỉ định giải pháp khả thi cho những chức năng đã được miêu tả trong các mô hình sử dụng đã được cung cấp, đồng thời nó cũng định nghĩa những protocol và những đặc trưng của mỗi protocol hỗ trợ cho mô hình sử dụng riêng biệt. Một số profile phụ thuộc vào những profile khác. Ví dụ, 3 profile (File Transfer Profile, Object Push Profile, và Synchronization Profile) phụ thuộc vào Generic Object Exchange Profile. Tất cả profile phụ thuộc vào Generic Access Profile...
- \_ Những sản phẩm Bluetooth hỗ trợ những bộ profile khác nhau, và để hỗ trợ một bộ profile nào đó thì những điểm đặc trưng bắt buộc của profile đó phải được thực hiện đầy đủ.
- \_ Những profile sau được Bluetooth SIG định nghĩa và thông qua:
  - Advanced Audio Distribution Profile (A2DP)
  - Audio/Video Remote Control Profile (AVRCP)
  - Basic Imaging Profile (BIP)
  - Basic Printing Profile (BPP)
  - Common ISDN Access Profile (CIP)
  - Cordless Telephony Profile (CTP)
  - Dial-up Networking Profile (DUN)
  - Fax Profile (FAX)
  - File Transfer Profile (FTP)
  - General Audio/Video Distribution Profile (GAVDP)
  - Generic Access Profile (GAP)
  - Generic Object Exchange Profile (GOEP)
  - Hands Free Profile (HFP)

- Hard Copy Cable Replacement Profile (HCRP)
- Headset Profile (HSP)
- Human Interface Device Profile (HID)
- Intercom Profile (ICP)
- Object Push Profile (OPP)
- Personal Area Networking Profile (PAN)
- Serial Port Profile (SPP)
- Service Discovery Application Profile (SDAP)
- SIM Access Profile (SAP)
- Synchronisation Profile (SYNCH)
- Video Distribution Profile (VDP)

Những profile còn lại vẫn chưa hoàn thành, nhưng đã được Bluetooth SIG đề xuất là:

- Handsfree Profile 1.5 (HFP 1.5)
- Unrestricted Digital Information (UDI)
- Wireless application Protocol over BT (WAP)
- Extended Service discovery profile (ESDP)
- Local Positioning Profile (LPP)
- Video Conferencing Profile (VCP)
- Device ID (DID) : cho phép thiết bị được nhận dạng theo bǎn kýt thuật, nhà sản xuất, sản phẩm, phiên bản sản phẩm,...



Hình 2-25 Bluetooth v1.1 profiles

#### 2.6.1. 4 profile tổng quát trong đặc tả Bluetooth v1.1:

Generic Access Profile: định nghĩa những thủ tục chung có liên quan tới việc phát hiện những thiết bị Bluetooth (idle mode procedures) và những khía cạnh quản lý các kết nối đến những thiết bị này devices (connecting mode procedures). Nó cũng định nghĩa những phương thức liên quan tới việc sử dụng những cấp độ bảo mật khác nhau.Thêm vào đó, profile này cũng chứa những thủ tục định dạng phổ biến của những tham số có thể được dùng trên giao diện người dùng. Mỗi thiết bị Bluetooth đều có hỗ trợ Generic Access Profile.

Service Discovery Application Profile: định nghĩa những tính năng và thủ tục cho một ứng dụng trong thiết bị Bluetooth để phát hiện ra những service của thiết bị Bluetooth khác.

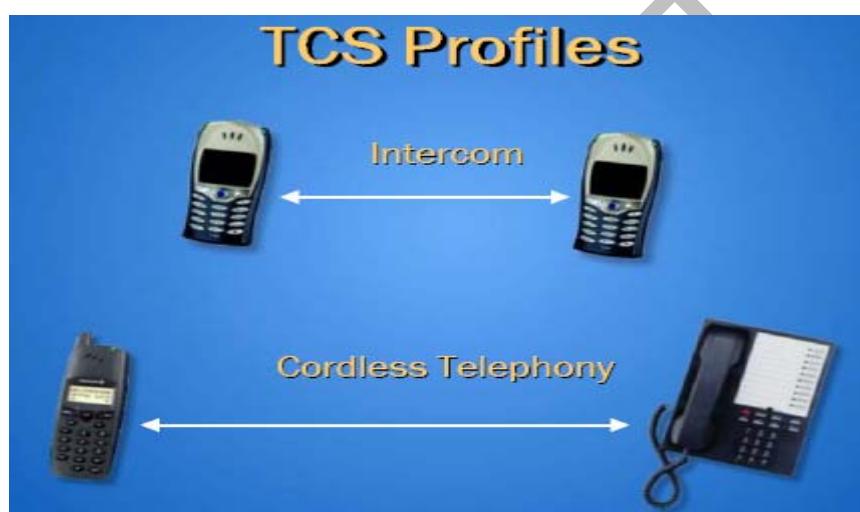
Serial Port Profile: định nghĩa thủ tục cần thiết của thiết bị Bluetooth để thiết lập những kết nối emulated serial cable sử dụng RFCOMM giữa hai thiết bị ngang hàng.

Generic Object Exchange Profile: định nghĩa những giao thức và thủ tục sẽ được dùng bởi những ứng dụng cần có năng lực trao đổi đối tượng(object

exchange capabilities ). Khả năng có thể xảy ra là sự đồng bộ hoá, file transfer, và object push.

### 2.6.2. Model-Oriented Profiles

Cordless Telephony Profile và Intercom Profile: định nghĩa những tính năng và thủ tục cần cho thao tác giữa các phần giữa những unit hoạt động trong mô hình "three-in-one phone" (một điện thoại có thể được dùng như là một cordless phone, một walkie-talkie, và một cellular phone). The Cordless Telephony Profile được dùng khi một điện thoại kết nối với một trạm cơ sở của một mạng điện thoại cố định thông qua Bluetooth và Intercom Profile thực hiện cái gọi là sử dụng "walkie-talkie" giữa những điện thoại Bluetooth.



Hình 2-26 TCS profile

Dial-Up Networking Profile: mô tả cách sử dụng một cellular phone hoặc một modem cạnh một computer như là một wireless modem để nhận dữ liệu, kết nối đến dial-up Internet access server, hoặc sử dụng dial-up service khác.



Hình 2-27 Networking Profiles

Fax Profile: định nghĩa cách một computer có thể sử dụng một Bluetooth cellular phone hoặc modem như là wireless fax modem để gửi hoặc nhận fax.

Headset Profile: định nghĩa những yêu cầu cần thiết cho một thiết bị Bluetooth hỗ trợ sử dụng headset. Wireless headsets có thể được dùng với cellular phones và laptops.



Hình 2-28 Headset Profile

LAN Access Profile: định nghĩa cách một thiết bị Bluetooth có thể truy cập dịch vụ của một mạng cục bộ sử dụng PPP (Point-To-Point Protocol) thông qua RFCOMM (giao thức Bluetooth - cạnh tranh với tín hiệu RS-232 )



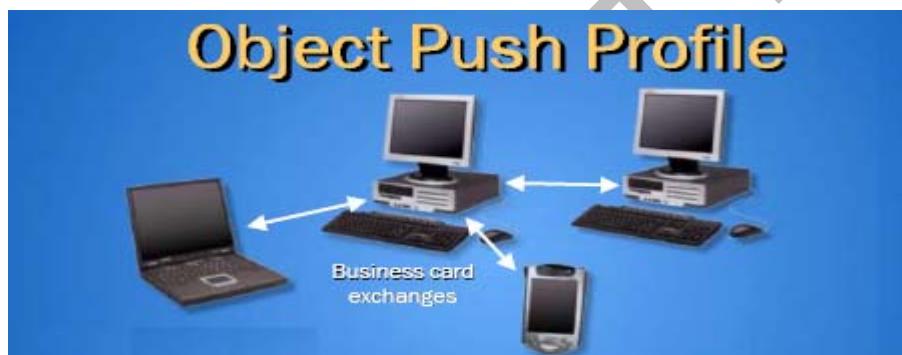
Hình 2-29 LAN Access

File Transfer Profile: cho phép người sử dụng duyệt và hiệu chỉnh những tập tin và thư mục(object) trong hệ thống tập tin của thiết bị Bluetooth khác và chuyển giao object giữa 2 thiết bị Bluetooth. Những thiết bị phổ biến nhất là PC, notebook và PDA.



Hình 2-30 File Transfer Profile

Object Push Profile: cho phép người sử dụng push, pull, và trao đổi những object đơn giản như business card giữa 2 thiết bị Bluetooth như PC, PDA và điện thoại di động.



Hình 2-31 Object Push Profile

Synchronization Profile: cho phép trao đổi dữ liệu về thông tin cá nhân (PIM) giữa 2 thiết bị tự động đồng bộ hóa dữ liệu(ví dụ: thành phần calendar hay phonebook). Đồng bộ hóa được dùng giữa những thiết bị notebook, PDA và điện thoại di động

### 2.6.3. Một số Profiles khác.

Đặc tả ban đầu của Bluetooth gồm 13 profile như trên. Để bảo đảm thao tác giữa các phần trong ứng dụng, những nhóm làm việc trong tổ chức SIG đã định thêm những profile mới. 12 profile thêm vào đã được công bố.

- Generic Audio/Video Distribution Profile (GAVDP): định nghĩa những phần chung của các giao thức và ứng dụng dùng phân phối nội dung audio/video sử dụng kênh ACL.

- Advanced Audio Distribution Profile (A2DP): định rõ sự phân bố nội dung audio về chất lượng cao (high quality), mono hoặc stereo trên kênh ACL.
- Audio/Video Remote Control Profile (AVRCP): định nghĩa việc truyền tín hiệu điều khiển từ một user đã được kích hoạt (user-activated A/V) đến một thiết bị Bluetooth từ xa.
- Basic Imaging Profile (BIP): là một OBEX-based profile cho phép thiết bị chọn được kích cỡ và mã của dữ liệu hình ảnh để trao đổi.
- Basic Printing Profile (BPP): là một OBEX-based profile cho phép in những e-mail dạng text, những thông điệp ngắn và định dạng những tài liệu từ thiết bị di động
- Hardcopy Cable Replacement Profile (HCRP): là một profile không quan trọng lăm dùng để in và quét bất cứ loại tư liệu nào. HCRP được thực hiện ngay lập tức ở trên L2CAP tránh liên quan tới OBEX, RFCOMM, hoặc PAN.
- Bluetooth Extended Service Discovery Profile (ESDP): cho Universal Plug và PlayTM (UPnP) là một profile dùng để phát hiện các thiết bị khác dùng dịch vụ UPnP và truy tìm thông tin về dịch vụ.
- Hands-Free Profile (HFP): định rõ trường hợp một điện thoại di động được dùng chung với một thiết bị hands-free (như một dụng cụ trang bị cho xe hơi-car kit). HFP cung cấp những phương tiện không dây cho cả điều khiển từ xa và kết nối bằng giọng nói.



Hình 2-32 Hands Free Profile

- Human Interface Device Profile (HID): định ra việc sử dụng bàn phím không dây ( wireless keyboard ), thiết bị trỏ (pointing device), thiết bị chơi game (gaming device), và thiết bị điều khiển màn hình (remote monitoring device).



Hình 2-33 Human Interface Device Profile

- Common ISDN Access Profile: định rõ làm sao những ứng dụng truy cập ISDN thông qua Bluetooth.
- Personal Area Networking Profile (PAN): định nghĩa IP cho những mạng cá nhân. PAN cũng hỗ trợ cho những điểm truy cập mạng ( network access point ) như LAN or GSM.
- SIM Access Profile (SAP): định ra làm sao truy cập SIM card thông qua liên kết Bluetooth.

## 2.7. Vấn đề sử dụng năng lượng trong Bluetooth.

### 2.7.1. Giới thiệu.

- Năng lượng là vấn đề cực kỳ quan trọng đối với thiết bị không dây vì những thiết bị này chỉ có thể sử dụng năng lượng từ pin, và điều này làm phát sinh những vấn đề liên quan như thời gian sử dụng pin, thời gian dự phòng và kích thước vật lý.
- Khi kết nối bằng Bluetooth thì ta phải cần năng lượng để duy trì kết nối, năng lượng để điều khiển bộ vi xử lý thực hiện chòng nghi thức Bluetooth và năng lượng để khuếch đại tín hiệu âm thanh đến cấp độ người sử dụng có

thể nghe được. Và những thiết bị di động nhỏ thì không thể sử dụng loại pin lớn nên tiêu thụ ít năng lượng là vấn đề quan tâm hàng đầu.

- \_ Chương trình quản lý năng lượng (power-managed application) là một ứng dụng cho phép thiết bị thực hiện chế độ ngủ(sleep mode) ở những giai đoạn đáng kể trong quy trình hoạt động. Sleep mode không làm tốn năng lượng của thiết bị, thật ra thì điều này không đúng lắm vì vẫn có vài chức năng luôn cần năng lượng, tuy nhiên vẫn ít hơn khi thiết bị thật sự “thức giấc” (awake), nói chung quản lý năng lượng sẽ là quản lý thời gian bỏ phí.
- \_ Một đặc điểm thêm nữa của việc quản lý năng lượng ở cấp độ ứng dụng là không ảnh hưởng xấu đến sự thực thi ứng dụng và việc lưu giữ năng lượng bằng trình ứng dụng không phụ thuộc vào kỹ thuật bên dưới ngay cả khi phần cứng được cải tiến để giảm thiểu sử dụng năng lượng.
- \_ Kỹ thuật Bluetooth thực hiện việc quản lý năng lượng đồng thời ở mức phần cứng (hardware) và phần mềm (software). Mật hạn chế là thời gian đáp ứng (response time) của các ứng dụng tăng lên và nếu như không dùng đúng thì việc quản lý năng lượng sẽ làm cho trình ứng dụng không còn đáp ứng nhanh nữa. Bluetooth cung cấp một số chế độ năng lượng thấp và một chế độ thích hợp với những loại ứng dụng khác nhau.
- \_ Trước khi chọn power management mode để sử dụng, độ trễ lớn nhất và mô hình radio traffic được mong chờ của ứng dụng phải được tính toán trước.

### **2.7.2. Việc sử dụng và quản lý năng lượng trong công nghệ Bluetooth**

#### **2.7.2.1. Tổng quan:**

- \_ Bluetooth cung cấp 3 chế độ có năng lượng thấp (low power mode) cho những lập trình viên sử dụng là hold, sniff, và park. Mỗi chế độ đều có những đặc điểm riêng và thuận lợi cho những lớp khác nhau của ứng dụng.
- \_ Hold mode thì thuận lợi cho những ứng dụng dự báo và điều khiển thời gian cho lần truyền dữ liệu kế tiếp. Khi mà khoảng thời gian giữa 2 lần truyền được thương lượng một cách độc lập bởi lần tiếp theo thì chế độ này vô cùng

thích hợp để ứng dụng giám sát thường xuyên kết nối và có thể tăng hoặc giảm “thời gian ngủ” (sleep time) cho phù hợp.

- \_ Hold mode không thể tự biến mất và do đó không nên dùng cho những ứng dụng có nhu cầu hard latency.
- \_ Sniff mode cho phép một thiết bị Bluetooth-enabled lưu trữ năng lượng bằng cách giảm đi số slot mà master có thể truyền, bằng cách đó có thể giảm số slot mà slave phải nhận. Chế độ này có vẻ thuyết phục hơn so với hold mode khi nó có thể toát ra bất kỳ lúc nào. Slave sẽ lắng nghe một cách định kỳ số slot và điều này làm cho sniff mode đặc biệt thuận lợi hơn đối với những ứng dụng mà dữ liệu đòi hỏi được truyền ở những khoảng thời gian cách đều. Ứng dụng không thích hợp với sniff mode là những loại cần truyền lượng dữ liệu lớn một cách liên tục và điều này bắt buộc thiết bị phải giữ nguyên tình trạng awake.
- \_ Park mode là chế độ cho phép lưu giữ năng lượng ở mức tối đa. Chế độ này thuận lợi nhất đối với những ứng dụng có mô hình lưu lượng sóng vô tuyến (radio traffic) không thể dự đoán trước và độ trễ của việc thiết lập kết nối được giới hạn bởi những hạn định cao hơn (upper limit). Ví dụ ở Headset profile, liên kết RFCOMM phải được unparked càng sớm càng tốt khi có một yêu cầu cần được gửi đi thông qua Audio Gateway để đến headset.
- \_ Các chế độ low power của Bluetooth khác nhau trong việc hỗ trợ quản lý năng lượng và do đó không có chế độ nào thật sự tốt nhất để sử dụng. Để xác định chế độ low power được dùng thì phải dựa vào dãy các nhân tố phụ thuộc vào loại ứng dụng và những nhu cầu của nó.

Những nhân tố chính là:

- Ứng dụng sử dụng việc quản lý năng lượng có tiện lợi không.
- Độ trễ tối đa mà ứng dụng có thể chấp nhận.
- Mô hình radio traffic được mong chờ: nhẫu nhiên(random), định kỳ(periodic), truyền loạt(bursty),...

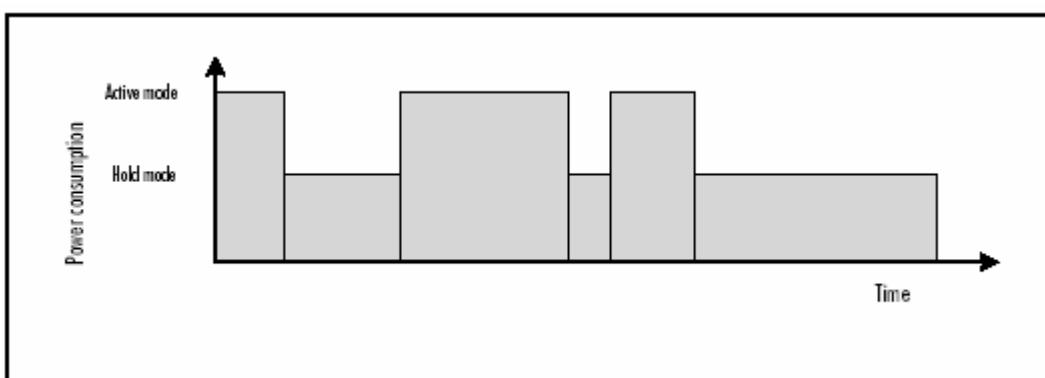
#### 2.7.2.2. Các chế độ năng lượng.

### 2.7.2.2.1. Active mode

- Trong chế độ Active, thiết bị tham gia hoạt động trên kênh sóng radio. Master sắp xếp các quá trình truyền phát dữ liệu, các gói tin được chuyển phát trên những băng tần được xác định và Slave phải lắng nghe các gói tin ở những khe thời gian được dành riêng cho chúng. Chế độ này là một tiêu chuẩn kỹ thuật để so sánh với hiệu năng của những chế độ năng lượng thấp bởi vì nó không những tiêu tốn hầu hết năng lượng mà còn có thông lượng dữ liệu truyền phát lớn nhất. Sự tiêu thụ năng lượng của thiết bị phụ thuộc nhiều vào nhà sản xuất thiết bị và ứng dụng đang chạy trên nó.
- Những ứng dụng mà thích hợp với chế độ Active thì sẽ không có lợi hoặc không thể sử dụng bất kỳ chế độ năng lượng thấp nào khác (Hold, Park, Sniff). Một ứng dụng có nhu cầu tần số dữ liệu truyền phát cao thì khó có thể tiết kiệm năng lượng bởi vì nó cần năng lượng cho máy truyền phát sóng radio cho phần lớn chu kỳ hoạt động. Tương tự những ứng dụng yêu cầu độ trễ thấp cũng không thích hợp để sử dụng những chế độ năng lượng thấp.

### 2.7.2.2. Hold mode

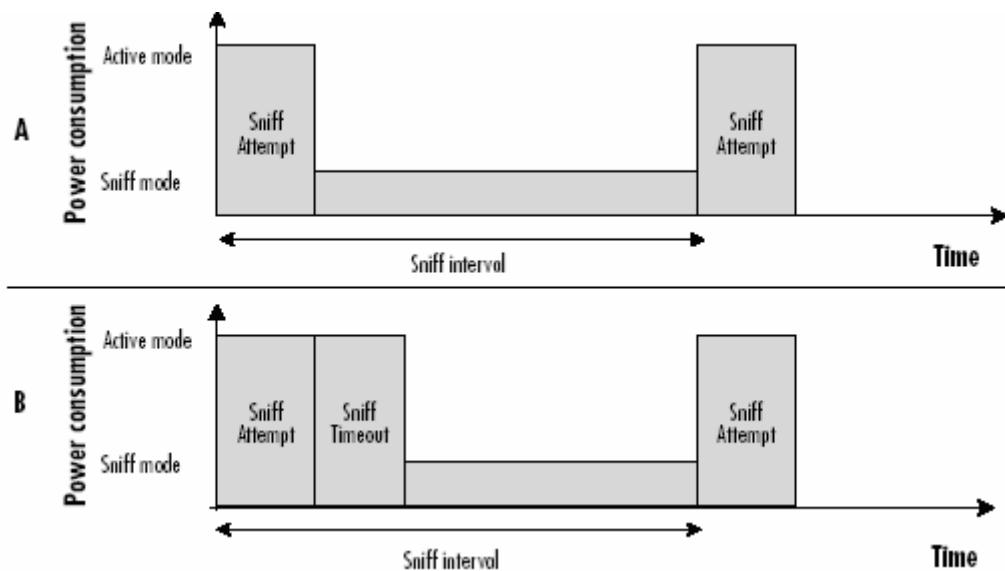
- Đây là chế độ đơn giản nhất trong những chế độ năng lượng thấp của Bluetooth. Master và Slave sẽ thỏa thuận với nhau trong suốt thời gian mà thiết bị Slave ở trong chế độ này. Khi một kết nối thiết lập trong chế độ này, nó không hỗ trợ những gói dữ liệu trên kết nối đó và có thể tiết kiệm năng lượng, lắng nghe định kỳ một khoảng thời gian lâu hơn hoặc cũng có thể tham gia vào một Piconet mới. Điều quan trọng là thời gian Hold sẽ được thỏa thuận trước mỗi khi chế độ Hold được thiết lập.



Hình 2-34 Hold Mode Interaction

- Hình trên cho thấy sự tương tác giữa những thiết bị sử dụng chế độ Hold. Một khía cạnh quan trọng hơn của chế độ Hold là mỗi lần chế độ này được thiết lập nó sẽ không bị hủy bỏ, và khoảng thời gian Hold phải kết thúc trước khi sự truyền thông có thể tái kích hoạt trở lại.
- Vậy những ứng dụng nào thì đạt hiệu quả khi sử dụng chế độ Hold? Nếu ứng dụng của bạn có thể quyết định hoặc điều khiển thời gian truyền phát dữ liệu ở lần kế tiếp thì ứng dụng có thể sử dụng chế độ Hold cho việc quản lý năng lượng. Một ví dụ là hệ thống phân phát e-mail không dây. E-mail không phải là một phương tiện truyền thông đồng bộ và những thông điệp được phân phát đến đích sau vài giây hoặc đến vài giờ. Quan trọng hơn, người sử dụng không biết được sự phân phát e-mail có thể xảy ra ngay lập tức và do đó bỏ qua độ trì hoãn nhỏ cho việc kéo dài thời gian sử dụng năng lượng của thiết bị.
- Một khía cạnh riêng biệt khác của chế độ Hold là sử dụng liên kết SCO mà không cần gửi trao đổi các gói dữ liệu. Hơn nữa nếu ứng dụng không quan trọng chất lượng audio lắm, nó có thể sử dụng ít hơn số khe thời gian do đó giảm được năng lượng. Ví dụ kiểm tra sự hoạt động của những thiết bị phát ra âm thanh (chỉ cần có liên kết SCO hoạt động không cần sử dụng liên kết ACL). Bằng cách đặt liên kết ACL trong chế độ Hold cho những khoảng thời gian vừa phải, và giảm chất lượng của liên kết SCO, ứng dụng có thể tiết kiệm năng lượng hơn.
- Nay giờ chúng ta hãy xem xét qua những ứng dụng mà không thích hợp cho việc sử dụng chế độ Hold. Chế độ Hold không thích hợp cho những ứng dụng yêu cầu thời gian phản hồi nhanh và khuôn mẫu lưu thông không thể đoán biết trước. Ví dụ như thiết bị cảm biến, truy cập Web thông qua liên kết không dây (trình duyệt Web không đoán biết được khuôn mẫu lưu thông của ứng dụng). Nhớ rằng khi chế độ Hold được thiết lập, nó không thể bị huỷ bỏ cho đến khi thời gian Hold thỏa thuận kết thúc.

#### 2.7.2.2.3. Sniffmode



Hình 2-35 Sniff Mode Interaction

- Chế độ năng lượng thấp này tiết kiệm năng lượng bằng cách giảm số lượng khe thời gian mà Master bắt đầu quá trình truyền phát dữ liệu và do đó cũng giảm số khe thời gian mà Slave phải lắng nghe. Tsniiff là khoảng thời gian giữa những khe thời gian được thỏa thuận giữa Master và Slave khi chế độ Sniff được thiết lập. Khi Slave lắng nghe trên kênh truyền, nó làm việc trong những khe Nsniff attempt, sau đó có thể giảm năng lượng cho đến cuối khoảng thời gian Sniff hiện thời. Thời gian tiếp nhận gói dữ liệu cuối cùng dành cho Slave rất quan trọng, vì vậy Slave phải lắng nghe trong khoảng thời gian Nsniff timeout ngắn nhất sau khi gói tin cuối cùng được nhận xong.
- Hình A cho thấy số lượng khe thời gian mà Slave phải lắng nghe. Trong trường hợp này Slave chỉ lắng nghe trong khoảng thời gian Nsniff attempt. Điều này xảy ra nếu Slave nhận được gói tin cuối cùng khi có nhiều hơn những khe Nsniff timeout trong Sniff attempt. Slave chỉ lắng nghe trong phần lớn khoảng thời gian Sniff attempt, sau đó giảm năng lượng.
- Hình B cho thấy Slave đang lắng nghe trong một khoảng thời gian mở rộng. Trong trường hợp này Slave lắng nghe khe Nsniff attempt, sau đó nhận một gói tin và lắng nghe thêm những khe thời gian Nsniff timeout.

Điều này cho thấy Slave phải lắng nghe thêm những khe thời gian Nsniff timeout nếu gói tin được nhận khi có ít hơn những khe Nsniff timeout ở bên trái khoảng thời gian Sniff attempt. Nếu Slave tiếp tục nhận những gói tin, nó sẽ lắng nghe tiếp tục những khe Nsniff timeout sau khi gói tin cuối cùng được nhận, vì vậy nếu Master vẫn giữ nguyên quá trình truyền phát thì Slave vẫn tiếp tục hoạt động.

- \_ Slave có thể thay đổi hoạt động của nó chỉ từ những khe Nsniff attempt thông qua những khe (Nsniff attempt + Nsniff timeout) và thậm chí tiếp tục hoạt động mà không cần thỏa thuận lại một vài tham số. Bằng cách chọn lựa những giá trị thích hợp cho khoảng thời gian Sniff và số lượng khe mà Slave phải lắng nghe, đạt được hiệu quả tiết kiệm năng lượng mà không ảnh hưởng bất lợi đến hiệu năng của ứng dụng.
- \_ Chế độ Sniff thì linh hoạt hơn chế độ Hold bởi vì Master hoặc Slave có thể giải phóng chế độ này. Bởi vì chế độ Sniff đòi hỏi thiết bị Slave thay đổi trạng thái hoạt động một cách định kỳ nên nó thích hợp cho những ứng dụng có sự truyền phát dữ liệu cách đều nhau.
- \_ Chế độ này thì không thích hợp cho những ứng dụng đòi hỏi thường xuyên truyền phát dữ liệu lớn. Đối với những ứng dụng, thời gian truyền phát dữ liệu rất quan trọng, bởi vì chúng cần nhiều thời gian nên không thể giảm năng lượng trong thời gian dài.

#### 2.7.2.2.4. Park mode

- \_ Chế độ Park là một chế độ năng lượng thấp cho phép tiết kiệm năng lượng nhất. Tuy nhiên trong khi ở chế độ Park, thiết bị không thể truyền hoặc nhận dữ liệu và không có liên kết SCO được thiết lập. Trong chế độ này, Slave không tham gia vào Piconet, tuy nhiên nó vẫn đồng bộ với kênh truyền trong Piconet. Chế độ này có thêm một thuận lợi là cho phép Master hỗ trợ hơn 7 thiết bị Slave bằng cách đưa những thiết bị còn lại vào trạng thái Park trong khi những thiết bị khác đang hoạt động trong trạng thái Active. Slave trong chế độ Park hoạt động một cách định kỳ để tái đồng bộ với kênh truyền và lắng nghe những thông điệp broadcast. Để làm được điều này, Master hỗ trợ cấu trúc tín hiệu phức tạp để liên lạc với Slave trong chế độ

Park. Tuy nhiên cấu trúc tín hiệu có thể thay đổi, sau đó Master dùng thông điệp broadcast để thông báo những thay đổi cho những Slave trong chế độ Park.

- Khi thiết kế ứng dụng, chúng ta phải chọn khoảng thời gian tín hiệu chính xác để tiết kiệm năng lượng trong khi duy trì thời gian hồi đáp có thể chấp nhận. Thời gian phản hồi chịu ảnh hưởng bởi Slave cần bao lâu để yêu cầu Unpark, hoặc Master cần bao lâu để Unpark cho Slave. Cả 2 trường hợp trên đều bị chi phối bởi thời gian tín hiệu Park.
- Nếu Slave trong chế độ Park mất sự đồng bộ, nó sẽ ngừng hồi đáp đến Master và có thể hoàn toàn mất kết nối. Sau đó Master sẽ khôi phục kết nối bằng cách gửi tín hiệu Paging đến Slave, rồi lại đặt nó vào chế độ Park lần nữa. Rõ ràng đây là sự hao phí vô ích. Vì vậy những thiết bị trong chế độ Park trong phần lớn thời gian hoạt động nên có những khoảng thời gian báo hiệu để mà nếu Slave bị nhỡ một tín hiệu, nó có thể được tái đồng bộ ở lần kế tiếp. Nói chung, để Master có thể gửi dữ liệu đến Slave thì trước tiên Slave phải được Unpark.
- Một ví dụ ứng dụng sử dụng chế độ Park: máy tính xách tay Bluetooth dùng trình duyệt Web không dây. Người sử dụng có thể mở nhiều trang Web, nhưng tại một thời điểm đang đọc một trang nào đó thì các trang khác sẽ chuyển sang trạng thái Park.
- Mang những bộ cảm biến thì không thích hợp sử dụng chế độ Park bởi vì trong cách bộ cảm biến gửi dữ liệu, yêu cầu phải hồi đáp ngay lập tức, không cho phép có độ trễ.

## **2.8. So sánh Bluetooth với các kỹ thuật không dây khác : Hồng ngoại, Wi-fi (802.11b wireless).**

### **2.8.1. So sánh Bluetooth với Wi-Fi**

Wi-Fi là chuẩn do IEEE (Institute of Electrical and Electronics Engineers) phát triển, đã trở thành một chuẩn rất phổ biến trong kết nối không dây. Wi-Fi là

chuẩn áp dụng cho mạng LAN không dây kiểu Ethernet[3], hoạt động trong vùng sóng radio 2.4 Ghz. Tốc độ truyền dữ liệu nói chung là 1 Mbps hoặc 2 Mbps với 802.11 và 5.5 Mbps hoặc 11 Mbps với 802.11b, mặc dù tốc độ có thể lên đến khoảng 20 Mbps với 802.11b. Wi-Fi hỗ trợ mạng đa điểm (multipoint networking) những kiểu truyền dữ liệu như các gói tin broadcast, multicast, unicast. Chuẩn thông thường là một access point (AP) đến 10-20 trạm (station), nhưng địa chỉ MAC có trong mỗi thiết bị đã cấp một số ảo vô tận để thiết bị có thể tham gia vào một mạng nhất định. Carrier Sense Multiple Access với Collision Avoidance (CSMA/CA) đã được dùng để điều khiển các kênh thông thường và ngăn xung đột. Tuy nhiên vì thế mà khá đắt và tốn nhiều năng lượng hơn.

	<b>Wi-fi</b>	<b>Bluetooth</b>
<b>Sử dụng điển hình (Typical usage)</b>	Phiên bản không dây của chuẩn Ethernet (wireless Ethernet), chỉ thay thế cáp cho truy cập mạng LAN. Truy cập mạng không dây với khoảng cách dài.	Thay thế cáp cá nhân (wireless USB) cho nhiều ứng dụng khác nhau. Truy cập mạng không dây với khoảng cách trung bình.
<b>Băng thông</b>	11 Mbps, chia sẻ. 2 đến 3 Mbps với WEP.	1 Mbps, chia sẻ. Version 1.1 và 1.2 là 723.1 Kbps, version 2.0 là 2.1 Mbps, thấp hơn khi bị nhiễu.
<b>Nhiều</b>	Các thiết bị sử dụng sóng radio khác, các vật liệu xây dựng, trang thiết bị.	Các thiết bị sử dụng sóng radio khác, các vật liệu xây dựng, trang thiết bị.
<b>Bảo mật</b>	Không an toàn nếu	Bảo mật thấp. Liên kết

	<p>không bảo vệ tốt. Cần giải quyết những trực trặc của mạng, truy cập bất hợp pháp, hi-jacking, "Đánh hơi mạng" (hay còn gọi là dò tìm lỗ hổng mạng thông qua việc dò tuẫn tự các gói tin ở cổng mạng khác nhau), đánh cắp phiên làm việc và truy cập trái phép.</p> <p>Liên kết mức độ WEP dễ bị bẻ gãy. Tin cậy vào những truy cập ở cấp độ ứng dụng và sự mật hoá.</p>	<p>được thiết lập ở mức độ sự thẩm định quyền (authentication). Khó khăn hơn cho traffic sniffing. Vẫn còn phụ thuộc vào sự thẩm định quyền ở cấp độ ứng dụng và sự mật hoá.</p>
<b>Tiêu thụ năng lượng</b>	<p>Khá cao.</p> <p>Thời gian sử dụng pin rất ngắn do tiêu thụ nhiều năng lượng và duy trì kết nối.</p>	<p>Thấp.</p> <p>Có 3 chế độ năng lượng thấp (Hold mode, Sniff mode, Park mode) giúp tăng thời gian sử dụng pin.</p>
<b>Khoảng cách (ngoài trời)</b>	<p>200 m - 11 Mbps.</p> <p>500 m - 1 Mbps.</p>	30 m-100 m.
<b>Khoảng cách (trong nhà)</b>	<p>40 m - 11 Mbps.</p> <p>100 m - 1 Mbps.</p>	10 m-30 m.
<b>Số kênh</b>	<p>11 DSSS.</p> <p>79—FHSS.</p>	79
<b>Năng lượng truyền</b>	20 dBm—FHSS.	20 dBm

<b>tối đa</b>	30 dBm—DSSS.	
<b>Tần số</b>	2.4GHz -b/g 5.8GHz - a	2.4GHz
<b>Giá thành</b>	Cao	Thấp
<b>Kết nối theo đường thẳng</b>	Không	Không
<b>Thiết bị hỗ trợ</b>	Hỗ trợ trong một số laptop hiện đại, PDA: đài external H/W card, Notebook computer, desktop computer, server.	Hỗ trợ trong laptop hiện đại, nhiều điện thoại di động, PDA, thiết bị điện tử, thiết bị tự động trong công nghiệp và văn phòng.
<b>Vị trí sử dụng</b>	Ở trong tầm hoạt động của các thiết bị WLAN, thường là trong các tòa nhà.	Bất cứ nơi nào có ít nhất 2 thiết bị Bluetooth.
<b>Ngày bắt đầu phát triển</b>	1990	1998
<b>Số thiết bị có thể truy cập đồng thời</b>	Nhiều, chia sẻ. IP&P2P.	Tối đa 8, chia sẻ. P2P.

Bảng 1-1 So sánh Wifi và Bluetooth

### 2.8.2. So sánh Bluetooth với IrDA:

IrDA (Infrared Data Association) là một tổ chức thương mại phi lợi nhuận với hơn 160 công ty thành viên về thiết bị máy tính, viễn thông, phần mềm, adapter... Kết nối hồng ngoại là kỹ thuật không dây sử dụng tia hồng ngoại để truyền dữ liệu, là chuẩn kết nối theo kiểu ad-hoc trong khoảng cách 1m. IrDA đã được khai thác từ rất lâu và được tích hợp vào nhiều thiết bị cá

nhận như điện thoại di động, PC, PDA, printer...với số lượng thiết bị ngày càng tăng. Ngày nay, các nhà sản xuất đã tích hợp cả IrDA và Bluetooth vào thiết bị của họ để việc kết nối không dây trong khoảng cách ngắn hiệu quả hơn.

	<b>IrDA</b>	<b>Bluetooth</b>
<b>Sử dụng điển hình (Typical usage)</b>	Kỹ thuật không dây dùng tia hồng ngoại để truyền dữ liệu Giao tiếp point-to-point hoặc point-to-multipoint khoảng cách ngắn	Thay thế cáp cá nhân Truy cập mạng không dây với khoảng cách trung bình
<b>Băng thông</b>	4Mbps - 16Mbps	1 Mbps, chia sẻ Version 1.1 và 1.2 là 723.1 Kbps, version 2.0 là 2.1 Mbps, thấp hơn khi bị nhiễu.
<b>Nhiều</b>	Bị ảnh hưởng bởi độ trong sạch của ánh sáng.	Các thiết bị sử dụng sóng radio khác, các vật liệu xây dựng, trang thiết bị.
<b>Bảo mật</b>	Khoảng cách ngắn + truyền thẳng => bảo mật ở cấp độ thấp nhưng độ an toàn cao. Phần còn lại phụ thuộc vào ứng dụng.	Bảo mật thấp. Liên kết được thiết lập ở mức độ sự thẩm định quyền (authentication). Khó khăn hơn cho traffic sniffing. Vẫn còn phụ thuộc vào sự thẩm định quyền ở cấp độ ứng dụng và sự mật hóa.
<b>Tiêu thụ năng lượng</b>	Rất thấp Không cần phải duy trì	Thấp Có 3 chế độ năng lượng

	kết nối nên thời gian sử dụng pin rất dài.	thấp (Hold mode, Sniff mode, Park mode) giúp tăng thời gian sử dụng pin.
<b>Khoảng cách</b>	10cm – 1m.	Người dùng cần phải ở gần một access point, khoảng 10m. Nhưng với thiết bị đặc biệt thì khoảng cách này được tăng lên.
<b>Giá thành</b>	Rất thấp	Thấp
<b>Kết nối theo đường thẳng</b>	Bắt buộc.  Tầm hoạt động trong một hình nón có mà góc ở chóp là $30^{\circ}$ và không thể xuyên qua vật cản.  IrDA không thực sự ổn định và không phải bất cứ thiết bị nào cùng hỗ trợ chuẩn IrDA cũng có thể kết nối với nhau.	Không.  Có thể xuyên qua vật cản.  Thiết bị nếu đã hỗ trợ Bluetooth thì chắc chắn kết nối với nhau được.
<b>Thiết bị hỗ trợ</b>	Trong rất nhiều điện thoại di động, PC, PDA, modem, camera và rất nhiều thiết bị điện tử khác trong y tế và công nghiệp.	Hỗ trợ trong laptop hiện đại, nhiều điện thoại di động, PDA, thiết bị điện tử, thiết bị tự động trong công nghiệp và văn phòng
<b>Số thiết bị có thể truy cập đồng thời</b>	Nhiều, chia sẻ	Tối đa 8, chia sẻ

Bảng 1-2 So sánh IrDA và Bluetooth

## **Chương 3 VẤN ĐỀ AN TOÀN VÀ BẢO MẬT TRONG BLUETOOTH.**

### **3.1. Sơ lược về vấn đề bảo mật trong các chuẩn không dây.**

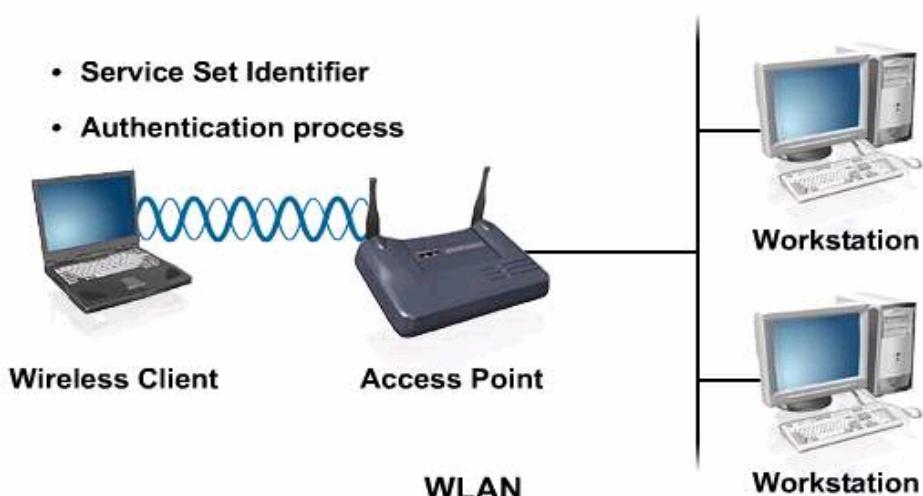
#### **3.1.1. Sơ lược chuẩn bảo mật mạng không dây trong 802.11**

Kỹ thuật kết nối Bluetooth cũng là kỹ thuật kết nối không dây thông thường, do đó các vấn đề bảo mật cốt lõi của mạng không dây cũng là những vấn đề chính trong bảo mật mạng Bluetooth. Việc giới thiệu sơ lược về các chuẩn bảo mật mạng không dây truyền thống giúp ta có cái nhìn tổng quát về qui trình bảo mật trong kỹ thuật Bluetooth.

#### **3.1.2. Chuẩn bảo mật WEP trong IEEE 802.11**

Định hướng ban đầu của mạng Wireless LAN (WLAN) trong an ninh mạng là sử dụng SSID (System Set Identifier) và xác thực điều khiển thông qua địa chỉ MAC của Client, với ý tưởng SSID được sử dụng giống như một từ khoá dùng chung cho các Access Point (AP) và các Client.

Nếu Client sử dụng SSID không giống với SSID của AP thì Client đó không có khả năng truy cập vào mạng LAN thông qua AP. Ngoài ra, WLAN còn hỗ trợ việc lọc địa chỉ MAC để điều khiển mức truy cập mạng. Các bảng thiết lập bằng tay trên AP cho phép hay ngăn cấm các Client truy cập qua AP vào mạng.



### Hình 3-1 Hai phương pháp truy cập mạng WLAN

Tuy nhiên, khi mạng WLAN phát triển, được ứng dụng nhiều thì có nhiều vấn đề về an ninh mạng phát sinh và trở thành mối quan tâm đặc biệt khi triển khai mạng WLAN và việc sử dụng SSID và địa chỉ MAC không đảm bảo được an ninh mạng.

Tiêu chuẩn 802.11 định nghĩa khả năng bảo mật WEP (Wired Equivalency Privacy) cho mạng WLAN sử dụng các khoá mã hoá 40 bits cho thuật toán mã hoá RC4, đây được xem là thuật toán đối xứng vì nó sử dụng khoá liên kết để mã hoá và giải mã plaintext Protocol Data Unit (PDU).

Khi sử dụng phương thức bảo mật này, một AP và các Wireless Client dùng chung các khoá WEP tĩnh. Khoá mã này được kiểm tra trong quá trình xác thực (Authentication), nếu khoá không tương thích thì Client không được liên kết với AP và do đó không thể truy cập được vào mạng. Khoá mã tĩnh dùng chung có khả năng dò tìm và bị lấy cắp, khi đó việc mã hoá không còn ý nghĩa với vấn đề an ninh mạng nữa.



Hình 3-2 Khoá WEP tĩnh được chia sẻ cho AP và các Client trong mạng.

Cisco hỗ trợ sử dụng tới 4 khoá mã hoá WEP có độ dài lên đến 128 bits trong một AP để tăng cường mức độ an ninh bảo mật của mạng. Tương ứng với khoá mã WEP, có 2 phương thức xác thực, đó là phương thức xác thực sử dụng

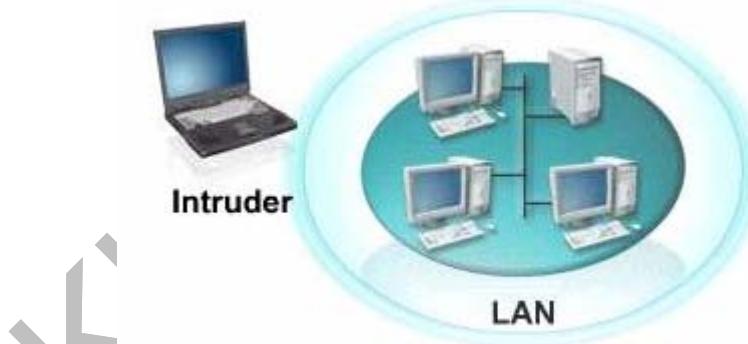
khoá mã chia sẻ dùng chung (Share Key Authentication) và xác thực mở (Open Authentication).

Xác thực sử dụng khoá mã dùng chung (shared key) cùng mục đích an ninh giống như SSID ban đầu, nhưng khi đó sẽ hạn chế khả năng linh hoạt của mạng WLAN. Trong khi đó xác thực sử dụng khoá mã mở lại được ưu dùng hơn, nhưng cũng bộc lộ một số nhược điểm.

### 3.1.3. Những vấn đề nảy sinh trong an ninh mạng không dây

SSID là một chuỗi ký tự 32 bits, ban đầu nó được xem là một cách bảo mật nhưng khi mạng WLAN phát triển thì nó không còn được coi là phương thức bảo mật nữa. Vì khi sử dụng phương thức xác nhận mở, 802.11 cho phép các Wireless Client sử dụng giá trị SSID trắng (giá trị NULL) để liên kết với AP trong quá trình tạo liên kết và xác thực.

Các nguy cơ đe dọa an ninh mạng từ phía ngoài do sử dụng môi trường truyền dẫn là không khí ở tần số “free” nên bất kỳ thiết bị không dây nào nằm trong vùng phát sóng của AP cũng nhận được thông tin từ AP truyền đến.



Hình 3-3 Mạng WLAN và các thiết bị xâm nhập

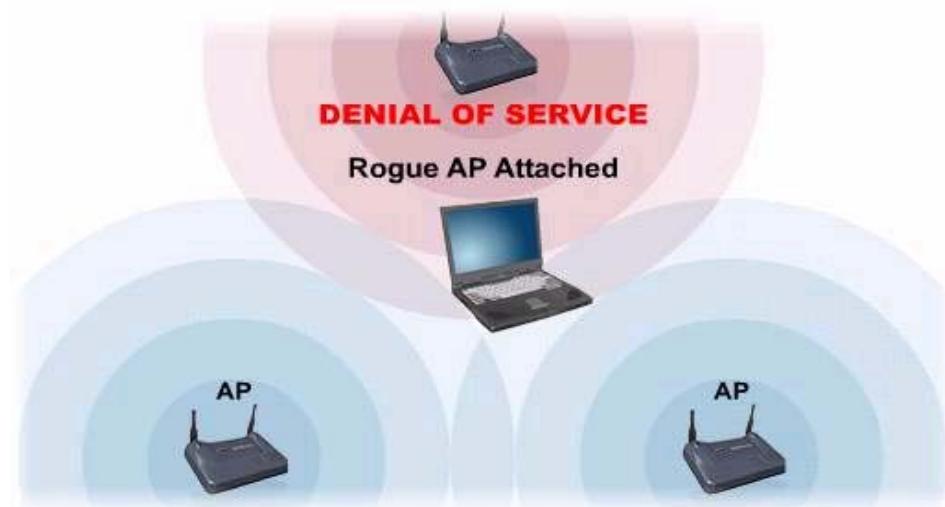
Nếu các khoá mã WEP được chứa trong card mạng không dây (Wireless Card) thì khi bị đánh cắp, các Client có được Card mạng đó có thể truy cập mạng mà không bị phát hiện từ phía người quản trị mạng, hay AP. Giả sử có phát hiện được thì phải thay đổi khoá mã WEP, điều này trở nên phức tạp với những mạng có số lượng người dùng lớn.



Hình 3-4 Card mạng với khoá mã WEP bên trong.

Đối với vấn đề xác thực, chuẩn 802.11 chỉ xác định phương thức xác định một chiều (one-way), từ phía AP đối với Client chứ chưa có chiều xác thực ngược lại từ Client đến AP (Rogue AP). Một khía cạnh khác các khoá mã sử dụng trong khi mã hoá dữ liệu là các khoá mã tĩnh, không có cách tạo mã và quản lý các khoá mã đó. Vì vậy nếu có thể thay đổi thường xuyên các khoá mã sẽ an toàn hơn cho các khoá mã không bị đánh cắp hoặc phát hiện ra.

“Rogue AP” (AP giả mạo) có thể được dùng để tấn công mạng khi được sử dụng và đặt trong vùng gần với vùng phủ sóng của mạng WLAN. Các Client khi di chuyển đến gần Rogue AP sẽ tự động liên kết với AP giả mạo đó và cung cấp các thông tin của mạng WLAN cho Rogue AP.



Hình 3-5 Các Rogue AP tấn công mạng bằng cách giả danh một AP hợp pháp.

Mặt khác, chuẩn 802.11 không hỗ trợ các phương pháp xác thực người dùng truy nhập từ xa vào mạng hiện tại như các giao thức xác thực RADIUS, LDPA, ... nên hạn chế khả năng quản lý an ninh mạng một cách tập trung.

### **3.2. Qui trình bảo mật trong Bluetooth :**

#### **3.2.1. An toàn bảo mật trong Bluetooth:**

- Trong công nghệ hoặc những mặt khác thì vấn đề an toàn tuyệt đối có lẽ không bao giờ được đảm bảo. Chúng sẽ càng ngày càng phát triển và quan trọng đối với bất kỳ kỹ thuật nào. Bluetooth SIG đã đưa ra những cải tiến về bảo mật nhằm tăng tính vững chắc cho tiến trình pairing đồng thời đảm sự riêng tư khi kết nối đã được thiết lập, có găng luôn đi trước một bước để đảm bảo thiết bị không bị tấn công.
- Bluetooth có nhiều khía cạnh về bảo mật cần giải quyết. Đối với mục tiêu là mật mã hóa và thẩm định quyền, Bluetooth Special Interest Group đã tạo ra 4 yếu tố để bảo mật. Nhưng mức độ an toàn của chúng không được tốt lắm, vì những đặc tả về an toàn của nó đã khiến cho nhiều thiết bị Bluetooth có thể được truy cập tự do mà không qua một rào cản nào cả.
- Bluetooth sử dụng môi trường wireless do đó nảy sinh một số vấn đề bảo mật của chuẩn wireless. Đây là lĩnh vực con người đang khám phá và cũng là nơi có thể làm nhiều tín hiệu bạn sử dụng. Bluetooth đang cố gắng giải quyết những vấn đề này bằng cách sử dụng hệ thống nhảy tần số. Khi 2 thiết bị Bluetooth kết nối và đồng bộ với nhau chúng sẽ nhảy 79 bước trên tần số 2.4 GHz. Những phiên bản cũ của Bluetooth có rắc rối với việc sử dụng tần số do một số nước hạn chế bước nhảy là 23. Thiết bị 23 bước nhảy không thể giao tiếp với thiết bị 79 bước nhảy. Tuy nhiên sau các thỏa thuận của Bluetooth Special Interest Group, Bluetooth đã sử dụng 79 bước nhảy ở tất cả các nước.
- Bảo mật Bluetooth cũng phải đối mặt với những vấn đề phổ biến. Bluetooth sử dụng 4 yếu tố khác nhau để duy trì sự bảo mật. Đầu tiên là địa chỉ thiết bị Bluetooth do Institute of Electrical and Electronics Engineers

(IEEE) định nghĩa, với 48-bit duy nhất cho mỗi thiết bị Bluetooth. Thứ hai, Private Authentication Key là một số ngẫu nhiên 128-bit. Thứ ba, Private Encryption Key có từ 8-128-bit dùng để mật mã hoá. Cuối cùng là một số ngẫu nhiên do chính thiết bị tạo ra.

- \_ Khi 2 thiết bị muốn kết nối với nhau, một số ngẫu nhiên (link key) được tạo ra, và nếu thiết bị không đồng ý với điều đó, chúng sẽ không thể kết nối. Đó có thể là vấn đề hàng đầu nếu thiết bị không chờ đủ lâu để kết nối và link key không được tạo ra. Vấn đề khác của những phiên bản trước của Bluetooth là nếu thiết bị slave thực hiện thuật toán tạo khóa nhanh hơn master, cả hai sẽ đều coi mình là master và không thể kết nối được.
- \_ Một vấn đề khác của Bluetooth là bảo mật không là điều bắt buộc. Có 3 mức độ trong vấn đề bảo mật chung (Generic Security) của Bluetooth. Cấp 1 là không bảo mật (non-secure), nghĩa là mọi thiết bị đều có thể giao tiếp với thiết bị Bluetooth này. Cấp 2 là bảo mật theo mức dịch vụ (service-level enforced security), thiết bị sẽ kết nối sau đó mới xác thực. Cấp 3 là bảo mật theo mức liên kết (link-level enforced security), nó sẽ không kết nối đến thiết bị trừ khi đã được xác thực. Vấn đề chính của việc bảo mật ở cấp độ này là có một số thiết bị Bluetooth đã được kích hoạt theo chế độ mặc định và việc bảo mật bị vô hiệu hóa. “Một số thiết bị Bluetooth được lưu hành với các yếu tố bảo mật đã bị vô hiệu hóa, cho phép những thiết bị Bluetooth khác truy cập vào, theo RSA Security.” (Judge, 2002).
- \_ Có 2 loại cấp độ (level) truy cập vào thiết bị Bluetooth. Các dịch vụ (Services) trong một thiết bị Bluetooth cũng có 3 cấp độ. Có một số dịch vụ đòi hỏi sự xác thực (authentication) và quyền hạn (authorization), một số chỉ cần sự xác thực, và một số thì không cần gì cả (open services). Có 2 cấp độ bảo mật ở mức thiết bị. Thiết bị un-trusted cần sự xác thực trong khi thiết bị trusted thì không cần.

### **3.2.1.1. Phần mô tả về an toàn bảo mật:**

- \_ Những nguy hiểm đều được kể thừa từ kỹ thuật không dây, trong đó có một số giống như mạng có dây, một số thì trầm trọng hơn do kết nối không

dây và một số mới xuất hiện. Có lẽ hầu hết những nguy hiểm quan trọng đều do kỹ thuật này lấy sự giao tiếp trong không khí, một môi trường mờ, làm cơ sở.

– Những nguy hiểm đặc biệt và những yếu điểm của mạng không dây và thiết bị cầm tay bao gồm:

- Tất cả những yếu điểm tồn tại trong mạng thông thường đều có trong mạng không dây.
- Những người xấu có thể giành được quyền truy cập bất hợp pháp vào một chi nhánh mạng thông qua kỹ thuật kết nối không dây, một đường vòng để tránh bất kỳ firewall nào.
- Những thông tin nhạy cảm không được mã hóa (hoặc mã hóa bằng kỹ thuật đơn giản) được truyền đi giữa hai thiết bị không dây có thể bị ngăn chặn và lộ ra.
- Tấn công DoS có thể được thực hiện ở kết nối không dây hoặc trên thiết bị.
- Kẻ xấu có thể đánh cắp được những đặc điểm nhận dạng của người dùng hợp pháp và giả mạo họ để truy cập vào hệ thống mạng nội bộ (internal) hoặc bên ngoài (external).
- Dữ liệu quý có thể bị hỏng trong quá trình đồng bộ sai.
- Kẻ xấu có thể can thiệp vào thông tin cá nhân của người dùng và theo dõi được mọi hoạt động của họ.
- Kẻ xấu có thể thu lợi bất chính bằng cách sử dụng những thiết bị không hợp pháp (ví dụ như thiết bị client và access point) truy cập vào thông tin quý.
- Thiết bị cầm tay rất dễ bị mất và lộ thông tin mật.
- Dữ liệu có thể bị lấy mà không hề bị phát hiện do cấu hình thiết bị không đúng cách.
- Virus hoặc những đoạn code nguy hiểm có thể làm hỏng dữ liệu trên thiết bị không dây và sau đó được đưa vào kết nối mạng có dây.

- Kẻ xấu có thể thông qua mạng không dây kết nối đến những tổ chức hoặc những chi nhánh từ đó bắt đầu tấn công mà không để lại dấu vết nào.
- Những kẻ xâm nhập từ bên ngoài có thể chiếm được quyền điều khiển và quản lý mạng, từ đó họ có thể vô hiệu hóa hoặc phá vỡ mọi hoạt động.
- Kẻ xấu có thể dùng “nhóm thứ 3”, những dịch vụ mạng không dây không đáng tin cậy để giành quyền truy cập những tài nguyên của một chi nhánh hay tổ chức khác.
- Tấn công nội bộ có thể thực hiện được thông qua những đường truyền đặc biệt.

### **3.2.1.2. Nhìn sơ về bảo mật Bluetooth:**

Chuẩn không dây trên khắp thế giới đã phát triển và có nhiều định dạng khác nhau để giải quyết vấn đề an toàn cho người sử dụng. Kỹ thuật không dây Bluetooth là một trong những chuẩn không dây mới nhất. Bluetooth là lựa chọn thích hợp nhất cho mạng cá nhân (Personal Area Networks PANs)

Những người phát triển sản phẩm dựa vào kỹ thuật không dây Bluetooth có nhiều chọn lựa cho việc thực hiện vấn đề bảo mật.

Trong các thiết bị Bluetooth có 4 yếu tố dùng để duy trì sự an toàn ở cấp độ liên kết:

- BD\_ADDR (Bluetooth device address): một địa chỉ 48 bit duy nhất dành cho mỗi thiết bị Bluetooth do IEEE (Institute of Electrical and Electronics Engineers) qui định.
- Private authentication key: một số ngẫu nhiên dài 128 bit dùng cho việc xác nhận người sử dụng.
- Private encryption key: số dài 8-128 bit dùng để mã hóa.
- RAND: một số ngẫu nhiên hoặc giả ngẫu nhiên 128 bit, được thay đổi thường xuyên bởi chính thiết bị Bluetooth.

Trong Bluetooth Generic Access Profile, có 3 chế độ bảo mật khi truy cập Bluetooth giữa 2 thiết bị:

- Security Mode 1: không bảo mật
- Security Mode 2: bảo mật thi hành ở cấp độ dịch vụ
- Security Mode 3: bảo mật thi hành ở cấp độ liên kết

Sự khác biệt giữa mode 2 và mode 3 là ở mode 3 quy trình bảo mật được khởi động trước khi kênh truyền được thiết lập.

Các nhà sản xuất sẽ tự chọn chế độ bảo mật cho sản phẩm của mình. Thiết bị và dịch vụ cũng có những mức độ bảo mật khác nhau.

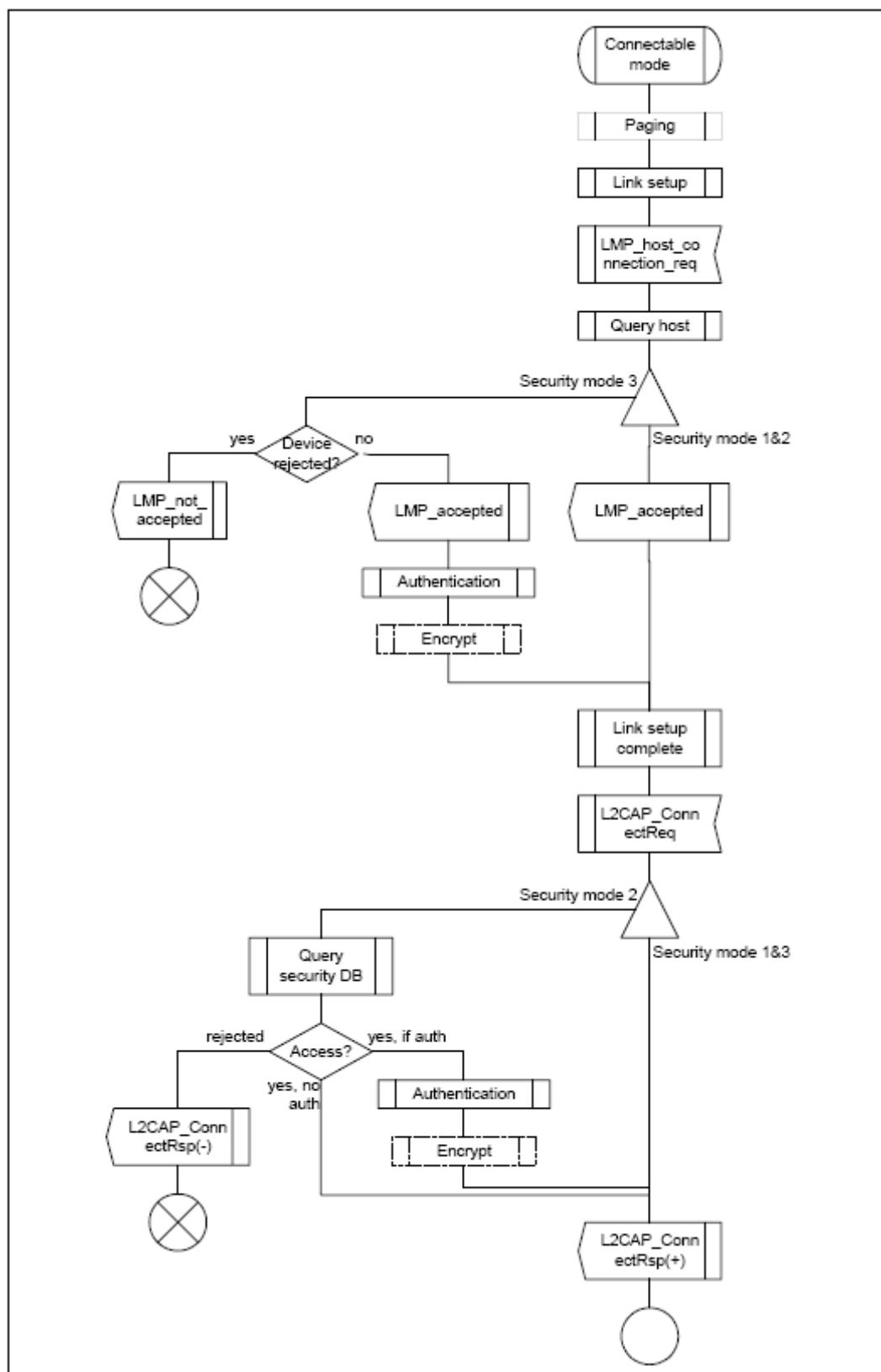
Thiết bị có 2 mức độ là "trusted device" and "untrusted device". Một thiết bị khi kết nối với thiết bị trusted sẽ được truy cập vào mọi dịch vụ mà không bị hạn chế.

Dịch vụ có 3 mức độ: dịch vụ yêu cầu cấp phép và xác nhận, dịch vụ chỉ yêu cầu xác nhận và dịch vụ “mở” đối với tất cả thiết bị.

Thuật toán mã hóa trong Bluetooth cũng chắc chắn. Thông thường trong một cặp thì việc truyền thông cũng rất an toàn như giữa chuột và bàn phím với PC, một điện thoại di động đồng bộ với PC và một PDA dùng điện thoại di động như một modem.

Mô hình dưới đây miêu tả quá trình thiết lập kênh truyền, khởi đầu việc xác nhận người dùng.

## Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa



Hình 3-6 Quá trình thiết lập kênh truyền

Khi xác nhận một thiết bị Bluetooth, nó sẽ thực hiện theo Bluetooth Link Manager Protocol và chế độ pairing hiện thời, tùy thuộc vào cấp độ bảo mật được sử dụng.

Mỗi thiết bị Bluetooth chỉ hoạt động ở một chế độ trong một thời điểm riêng biệt.

### 3.2.1.2.1. Security Mode 1: không bảo mật (Nonsecure mode)

Ở chế độ này một thiết bị sẽ không phải thực hiện bất kỳ quy trình bảo mật nào, các hoạt động bảo mật (xác nhận và mã hóa) hoàn toàn bị bỏ qua (không bao giờ gửi send LMP\_au\_rand, LMP\_in\_rand hoặc LMP\_encryption\_mode\_req). Kết quả là thiết bị Bluetooth ở chế độ 1 cho phép các thiết bị Bluetooth khác kết nối với nó. Chế độ này áp dụng cho những ứng dụng không yêu cầu bảo mật như trao đổi business card. Ở cấp độ này thiết bị Bluetooth không bao giờ thực hiện bất kỳ biện pháp bảo mật nào như không bao giờ gửi send LMP\_au\_rand, LMP\_in\_rand hoặc LMP\_encryption\_mode\_req.

### 3.2.1.2.2. Security Mode 2: bảo mật thi hành ở cấp độ dịch vụ (Service-level enforced security mode)

Thiết bị Bluetooth sẽ không thực hiện bất kỳ biện pháp an toàn nào trước khi thiết lập kênh truyền ở cấp độ Logical Link Control và Adaptation Protocol (nhận được L2CAP\_ConnectReq) hoặc tiến trình thiết lập kênh truyền được bản thân nó thực hiện. L2CAP nằm ở tầng data link và cung cấp dịch vụ kết nối có định hướng và phi kết nối ở những tầng cao hơn. Quá trình bảo mật có được thực hiện hay không đều tùy thuộc vào yêu cầu của kênh truyền hoặc dịch vụ.

Ở cấp độ bảo mật này, một người quản lý bảo mật (như lý thuyết trong đặc điểm Bluetooth) điều khiển truy cập vào dịch vụ và thiết bị. Quản lý bảo mật tập trung bao gồm kiểm soát việc điều khiển truy cập với các giao thức khác và người dùng thiết bị. Đôi với các ứng dụng có yêu cầu bảo mật khác nhau được sử dụng song song thì ta có thể thay đổi việc kiểm soát an toàn và các mức độ tin cậy để hạn chế truy cập. Do đó nó có thể cho phép truy cập vào

dịch vụ này mà không được truy cập vào dịch vụ khác. Trong cấp độ này rõ ràng khái niệm cấp phép (authorization) -cho phép thiết bị A được truy cập vào dịch vụ X hay không- đã được áp dụng.

Thiết bị Bluetooth ở cấp độ này sẽ phân loại yêu cầu an toàn của dịch vụ nó sử dụng theo những đặc điểm sau:

- Yêu cầu phân quyền (Authorization required)
- Yêu cầu xác nhận (Authentication required)
- Yêu cầu mã hóa (Encryption required)

Ghi chú: khi dịch vụ không yêu cầu bất kỳ biện pháp an toàn nào thì cấp độ này giống với cấp độ 1.

### 3.2.1.2.3. Security Mode 3: bảo mật thi hành ở cấp độ liên kết (Link-level enforced security mode)

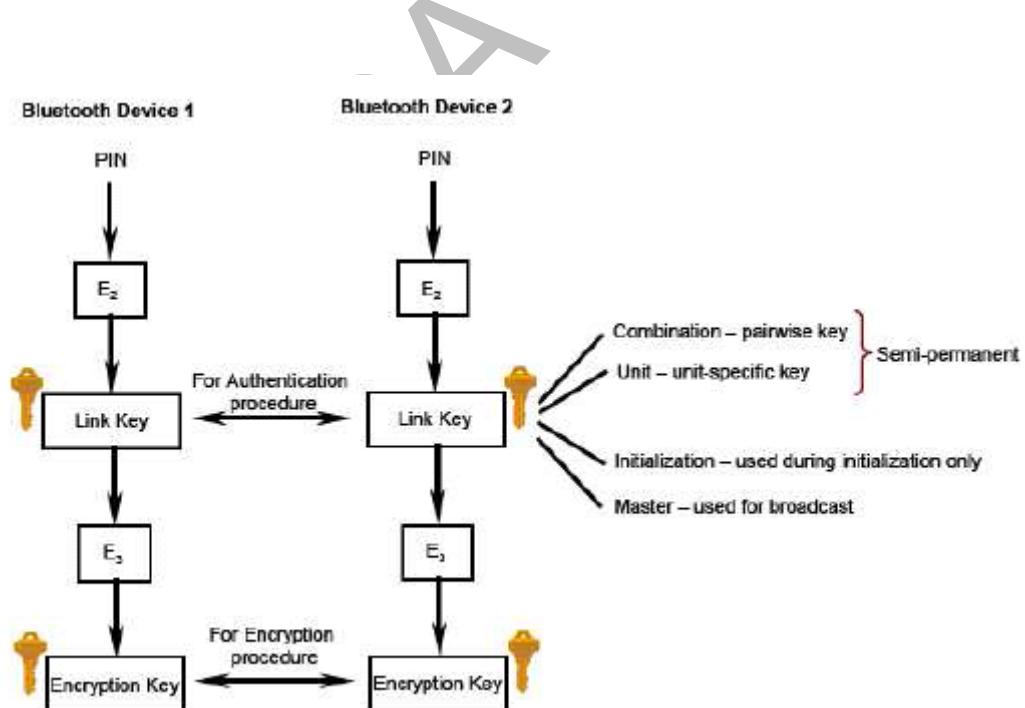
Thiết bị Bluetooth sẽ thực hiện quy trình bảo mật trước khi kênh truyền được thiết lập (nó gửi LMP\_link\_setup\_complete). Đây là cơ chế bảo mật “gắn liền”, và nó không nhận thấy bất kỳ biện pháp bảo mật ở cấp độ ứng dụng nào. Chế độ này hỗ trợ việc xác nhận đúng (authentication), một chiều hay hai chiều, và mã hóa. Những điều này tùy thuộc vào một link key bí mật dùng giữa một cặp thiết bị. Để tạo ra key này, một quy trình pairing được thực hiện khi hai thiết bị giao tiếp trong lần đầu tiên. Ở cấp độ này, thiết bị Bluetooth có thể bác bỏ yêu cầu kết nối máy chủ (LMP\_host\_connection\_req, đáp lại bằng LMP\_not\_accepted) tùy thuộc vào cài đặt của máy chủ.

### 3.2.1.2.4. Tạo Bluetooth key từ số PIN (Bluetooth Key Generation from PIN)

– PIN (Personal Identification Number) là một mã do người dùng chọn ngẫu nhiên 4 ký số hoặc nhiều hơn, PIN code dùng trong thiết bị Bluetooth có thể thay đổi từ 1-16 byte, dùng để kết hợp với một thiết bị khác để bảo đảm an toàn cho quá trình pairing. Một số ứng dụng thường chọn số PIN 4 byte, tuy nhiên nên chọn số PIN dài hơn do vấn đề an toàn. Người dùng được khuyên là nên chọn số PIN có từ 8 ký số trở lên để bảo đảm và chỉ nên đưa số PIN

cho những người và thiết bị “được tín nhiệm” để pairing. Không có số PIN hoặc số PIN không giống nhau thì quá trình pairing không xảy ra.

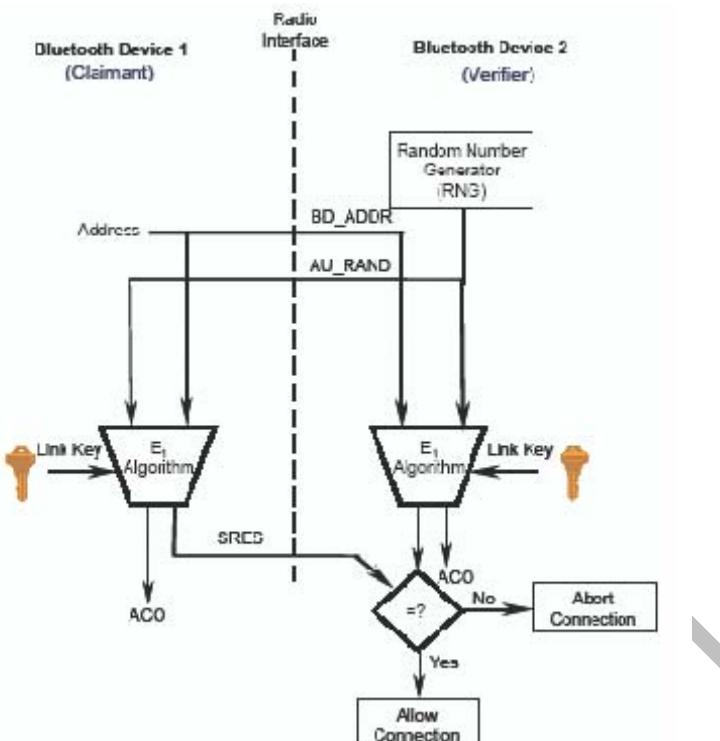
- \_ Về mặt lý thuyết, hacker có thể giám sát và ghi nhận mọi hành động trong dãy tần số và dùng máy tính để tìm ra số PIN đã được trao đổi. Điều này yêu cầu một thiết bị đặc biệt và một kiến thức toàn diện về hệ thống Bluetooth. Sử dụng số PIN có từ 8 ký số trở lên sẽ làm hacker tốn hàng năm để tìm ra còn dùng 4 ký số thì họ chỉ mất vài giờ để truy ra số PIN.
- \_ Link key được tạo ra trong suốt quá trình khởi tạo, khi hai thiết bị Bluetooth đang liên lạc với nhau, gọi là “associated” hoặc “bonded”. Bằng đặc điểm kỹ thuật Bluetooth, hai thiết bị giao tiếp với nhau ngay lập tức sẽ tạo ra link key trong quá trình khởi tạo, ngay khi người dùng đưa số PIN nhận diện vào cả hai thiết bị. Nhập số PIN, kết nối thiết bị và tạo ra link key được miêu tả trong hình 1-42. Sau khi quá trình khởi tạo hoàn thành, các thiết bị xác nhận một cách tự động và “trong suốt”, đồng thời thực hiện mã hóa. Nó có thể tạo ra link key dùng cho các phương thức trao đổi key ở tầng cao hơn và sau đó nhập link key vào Bluetooth module.



Hình 3-7 Bluetooth Key Generation from PIN

Xác thực trong Bluetooth (Bluetooth Authentication):

- \_ Quá trình xác nhận trong Bluetooth nằm trong sự phối hợp “challenge-response”. Hai thiết bị tương tác nhau trong một thủ tục xác nhận sẽ được xem như là một bên yêu cầu (claimant) và một bên xác minh (verifier). Thiết bị Bluetooth làm nhiệm vụ verifier phải xác nhận tính hợp lệ trong “nhân dạng” của thiết bị kia. Thiết bị claimant phải có gắng chứng tỏ nhân dạng của mình. Giao thức challenge-response xác nhận tính hợp lệ của các thiết bị bằng cách kiểm tra thông tin của secret key (link key của Bluetooth). Nguyên tắc phối hợp kiểm tra challenge-response được miêu tả trong hình 1-43. Như đã miêu tả, một trong những thiết bị Bluetooth (claimant) cố gắng bắt và kết nối với thiết bị kia (verifier).
  - \_ Các bước trong tiến trình xác nhận diễn ra như sau:
    1. Thiết bị claimant truyền địa chỉ 48 bit của nó (BD\_ADDR) đến verifier
    2. Thiết bị verifier truyền một challenge ngẫu nhiên 128 bit (AU RAND) đến claimant.
    3. Verifier dùng thuật toán E1 và sử dụng địa chỉ, link key, và challenge làm đầu vào để tính toán một câu trả lời xác nhận (authentication response). Thiết bị claimant cũng thực hiện cùng thao tác đó.
    4. Thiết bị claimant trả kết quả vừa tính được (SRES) cho verifier.
    - 5.Verifier sẽ so sánh SRES của claimant với SRES mà nó tính được.
    - 6.Nếu hai giá trị SRES 32 bit này bằng nhau thì verifier sẽ tiếp tục thiết lập kết nối.

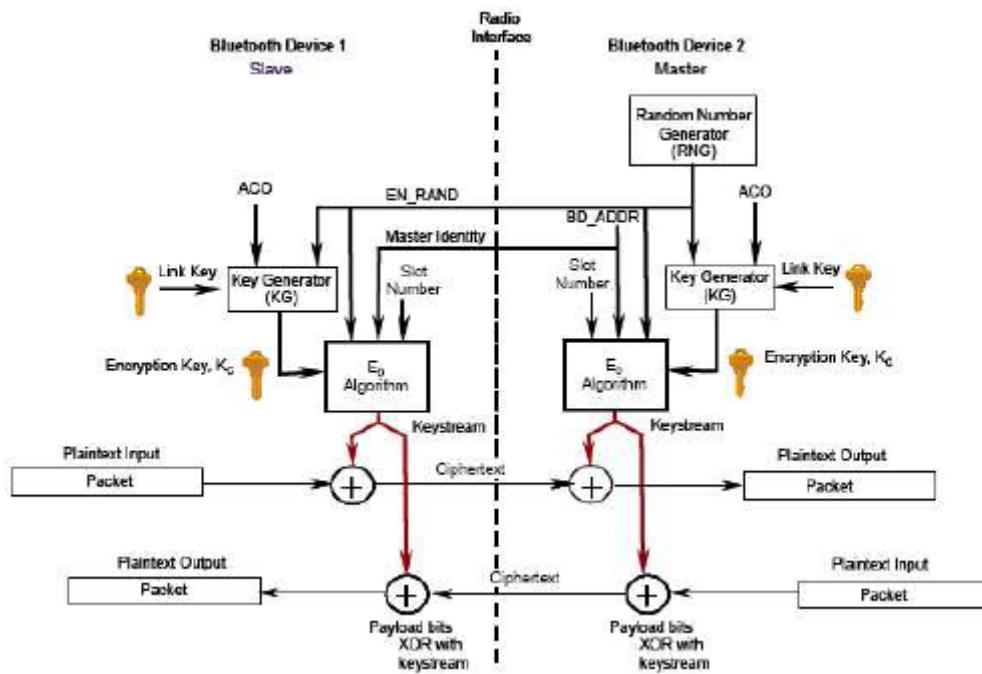


Hình 3-8 Bluetooth Authentication

### 3.2.1.2.5. Tiết trình mã hóa trong Bluetooth (Bluetooth Encryption Process):

Đặc tả Bluetooth cũng cho phép 3 chế độ mã hóa khác nhau để hỗ trợ cho sự an toàn của dịch vụ.

- Chế độ mã hóa 1: không thực hiện mã hóa khi truyền thông.
- No encryption is performed on any traffic.
- Chế độ mã hóa 2: truyền thông đại chúng (broadcast) thì không cần bảo vệ (không mã hóa), nhưng truyền cho cá nhân phải mã hóa theo link key riêng biệt.
- Chế độ mã hóa 3: tất cả mọi sự truyền thông đều phải được mã hóa theo link key của master.



Hình 3-9 Bluetooth Encryption Process

### 3.2.1.2.6. Những vấn đề trong an toàn bảo mật của chuẩn Bluetooth (Problems with the Bluetooth Standard Security)

- Mặt mạnh của phép tạo ngẫu nhiên challenge-response không hề được biết đến: RNG (Random Number Generator) thường dùng số cố định hoặc những số thay đổi theo chu kỳ, điều này làm giảm hiệu quả của phép xác nhận đúng.
- Cho phép những số PIN ngắn: sử dụng những số PIN đơn giản để tạo ra các link key và encryption key nên dễ dàng bị đoán ra. Tăng độ dài số PIN sẽ làm tăng độ an toàn. Nhưng mọi người lại có xu hướng chọn số PIN ngắn.
- Việc tạo và phân phối số PIN không đơn giản: thiết lập số PIN trong một mạng Bluetooth rộng lớn có nhiều người sử dụng rất khó khăn và thường xảy ra các vấn đề về an toàn bảo mật.
- Độ dài của encryption key có thể bị “thương lượng”: tổ chức Bluetooth SIG cần phát triển thêm quy trình phát sinh key khởi tạo mạnh mẽ hơn.
- Unit key có thể dùng lại được và trở thành công khai một khi được sử dụng: Một unit key là một link key được tạo ra bởi chính nó và được sử

dụng như là một link key với bất kỳ thiết bị nào. Unit key chỉ sử dụng an toàn khi tất cả các thiết bị paired với cùng unit key có độ tin tưởng tuyệt đối. Từ phiên bản Bluetooth 1.2 trở về sau đã không còn sử dụng unit key nhưng do tính kế thừa nên unit key vẫn chưa hoàn toàn bị loại bỏ ra khỏi các chi tiết kỹ thuật.

- Master key bị dùng chung: Nhóm Bluetooth SIG cần phát triển một phương pháp truyền khóa đại chúng tốt hơn.
- Không xác nhận người sử dụng: chỉ cung cấp cách xác nhận thiết bị. Xác nhận người sử dụng chỉ có thể thực hiện ở bảo mật cấp độ ứng dụng.
- Việc thử xác nhận được lặp đi lặp lại nhiều lần: Bluetooth SIG cần phát triển một giới hạn để ngăn chặn số yêu cầu quá nhiều. Đặc điểm kỹ thuật Bluetooth cần đưa ra một khoảng thời gian hạn định (time-out) giữa hai lần thử và được tăng lên theo số mũ.
- Thuật toán stream cipher E0 rất yếu kém: bắt nguồn từ phép tổng hợp stream cipher (summation combiner stream cipher) được Massey và Rueppel đưa ra vào giữa những năm 1980. Hầu hết tất cả các cuộc tấn công lớn vào loại stream ciphers này đều liên quan đến sự tấn công dựa vào việc đoán những khóa đơn giản. Gần đây việc giải các mật mã càng chỉ rõ tính yếu kém của the E0 cipher.
- Chiều dài các key có thể bị “thương lượng”: Một thỏa thuận chung toàn cầu về chiều dài tối thiểu của key cần phải được thiết lập.
- Sự phân bổ unit key có thể dẫn đầu về eavesdropping: một kẻ xâm chiếm (được quyền truy cập bất hợp pháp) có thể “thỏa hiệp” biện pháp bảo mật giữa 2 người sử dụng khác nếu kẻ đó kết nối với một trong 2 người này. Điều này là bởi vì link key (unit key), lấy được từ thông tin chung, đã bị lộ.
- Sự riêng tư có thể bị xâm phạm nếu địa chỉ thiết bị Bluetooth (BD\_ADDR) bị lộ ra và bị liên kết với một người “đặc biệt”, khi đó mọi hành động của người sử dụng thiết bị sẽ bị ghi nhận và không còn sự riêng tư.

- \_ Cách xác nhận thiết bị là shared-key challenge-response đơn giản: phương pháp xác nhận challenge-response chỉ một chiều là mục tiêu chính trong các cuộc tấn công man-in-the-middle. Nên yêu cầu xác nhận lẫn nhau để tăng tính tin cậy: cả người sử dụng lẫn mạng đều hợp pháp.
- \_ End-to-end security không được thi hành: chỉ thực hiện việc xác thực và mã hóa ở các liên kết riêng biệt. Các phần mềm ứng dụng ở tầng trên của Bluetooth cần phải được phát triển thêm.
- \_ Dịch vụ bảo mật bị hạn chế: không có kiểm định (audit), thực hiện (nonrepudiation), và những dịch vụ khác. Nếu cần thiết thì những điều này sẽ được thực hiện ở những vị trí đặc biệt trong một mạng Bluetooth.

### **3.2.2. Hacking:**

#### **3.2.2.1. Impersonation attack by inserting/replacing data**

Khi không thực hiện mã hóa thì tấn công kiểu này rất dễ đạt được bằng cách sửa CRC check data sau khi đã thay đổi dữ liệu. Nếu trong hệ thống có mã hóa thì rất khó do hacker phải tìm hiểu cấu trúc của gói dữ liệu để việc thay đổi có hiệu quả như mong muốn.

#### **3.2.2.2. Bluejacking**

Là kiểu gửi tin nhắn nặc danh ở những nơi công cộng bằng cách lợi dụng tiến trình pairing của kỹ thuật Bluetooth. Kẻ quấy rối gửi tin nhắn vào lúc khởi động giai đoạn “bắt tay” vì phần “name” - hiện tên thiết bị muốn kết nối có thể dài đến 248 ký tự. Thực ra mục đích của các nhà sản xuất là muốn thể hiện thông tin của thiết bị kết nối rõ ràng hơn để người dùng thấy yên tâm thực hiện trao đổi, cập nhật và đồng bộ dữ liệu. Nhưng đặc điểm này đã bị kẻ xấu lợi dụng để gửi những tin nhắn nặc danh cho các thiết bị Bluetooth đang hoạt động trong vùng xung quanh (10m).

Ảnh hưởng:

- Làm người chủ thiết bị khó chịu, hoang mang lo lắng.
- Không hề ảnh hưởng đến vấn đề an toàn, Nó không dòi hỏi hoặc thay đổi bất kỳ dữ liệu nào trên thiết bị.

- Khả năng mở rộng của ý tưởng này là gửi vCard' với một tên thông thường như “Home” hoặc “Work” nhằm cố gắng viết đè lên thông tin đã có sẵn trong máy người nhận.

### 3.2.2.3. Bluetooth Wardriving

Bản đồ tự nhiên về mọi vị trí của người sử dụng đang mở thiết bị Bluetooth. Mỗi thiết bị Bluetooth có một địa chỉ 48 bit tự do duy nhất dành cho broadcast và nó đã để lại dấu vết của người sử dụng. Để tránh bị theo dõi, thiết bị Bluetooth đã dùng một chế độ nặc danh (anonymity mode). Ở chế độ này hệ điều hành thiết bị thường xuyên cập nhật địa chỉ thiết bị của họ bằng cách chọn một số ngẫu nhiên. Các kiểu tấn công theo dấu vết gồm:

- **subsubsectionInquiry attack:** tấn công vào một hay nhiều thiết bị Bluetooth trong vùng phủ sóng. Chỉ xảy ra khi người sử dụng để thiết bị ở chế độ “nhìn thấy được” (discoverable mode). Thiết bị tấn công có thể vẽ được bản đồ các thiết bị Bluetooth xung quanh bằng cách thường xuyên gửi các thông điệp yêu cầu (inquiry messages) và thường xuyên nắm giữ danh sách tất cả thiết bị đã bị phát hiện.
- **subsubsectionTraffic monitoring attack:** tấn công ngay cả khi thiết bị nạn nhân ở không ở chế độ “nhìn thấy được”. Kẻ tấn công thường là giám sát các cuộc truyền thông giữa hai thiết bị “trusted” đối với nạn nhân. Những thiết bị này khi giao tiếp sẽ sử dụng một CAC đặc trưng. CAC này được tính ra từ địa chỉ của thiết bị master trong piconet. Hơn nữa toàn bộ địa chỉ của thiết bị được gửi đi trong gói FHS, cho phép một attacker xác định được “nhân dạng” của một thiết bị. Nhưng FHS chỉ được sử dụng khi thiết lập kết nối.
- **subsubsectionPaging attack:** tấn công kiểu này cho phép attacker xác định rõ khi nào một thiết bị với BD\_ADDR hoặc DAC đã nhận biết đang hiện diện trong vùng phủ sóng nhưng chỉ thực hiện được khi thiết bị đang kết nối. Thiết bị tấn công page với thiết bị đích, đợi nhận gói tin ID nhưng sau đó không phản hồi lại. Nếu nhận được ID thì attacker biết được là thiết bị đó đang hiện diện. Còn thiết bị đích

chỉ đợi tin phản hồi trong một thời gian “time out” nhất định và việc xảy ra sẽ không báo cáo lên tầng ứng dụng.

- **subsubsectionFrequency hopping attack:** hệ thống nhảy tần số trong Bluetooth được thực hiện bằng cách lặp lại tuần tự các bước nhảy. Lược đồ bước nhảy được tính toán từ những thông số khác nhau được nhập vào như địa chỉ và đồng hồ của master. Trong trạng thái kết nối thì LAP và tối thiểu 4 bit của UAP của thiết bị master sẽ được sử dụng. Trong trạng thái page thì LAP/UAP của thiết bị paged được sử dụng. Do đó về mặt lý thuyết có thể lấy thông tin của LAP và 4 bit trong UAP dựa vào lược đồ bước nhảy của đối tượng.
- **subsubsectionUser-friendlyname attack:** một thiết bị Bluetooth có thể đề nghị một cái tên thân thiện (user-friendly name) bất cứ lúc nào sau khi đã thực hiện thành công tiến trình paging. Và lệnh yêu cầu này có thể sử dụng để theo dõi dấu vết.

#### 3.2.2.4. Nokia 6310i Bluetooth OBEX Message DoS

Nokia 6310i có một khe hở cho phép từ chối dịch vụ từ xa. Điều này đã được phát hiện khi một thông điệp Bluetooth OBEX không hợp lệ do attacker gửi tới làm mất tính sẵn sàng của điện thoại.

Ảnh hưởng: Nhỏ vì khi ấy điện thoại bị shutdown mà không mất dữ liệu.

#### 3.2.2.5. Brute-Force attack

Tấn công Brute-force trên địa chỉ BD\_ADDR (MAC address) của thiết bị khi không ở chế độ “có thể nhìn thấy”. Một số nhà sản xuất đã khẳng định rằng việc này phải mất một thời gian lâu (khoảng 11 giờ). Tuy nhiên phiên bản đa tiêu trình của @stake’s RedFang có thể dùng cùng một lúc 8 thiết bị USB Bluetooth để giảm thời gian từ 11 giờ xuống 90 phút.

Ảnh hưởng:

- Có thể mất nhiều thời gian để phát hiện một BD\_ADDR chính xác.

- Một khi BD\_ADDR đã bị phát hiện thì một cuộc tấn công dạng Bluesnarf có thể được thiết lập trong khi người chủ thiết bị vẫn nghĩ họ vẫn an toàn bởi vì thiết bị đã đặt ở chế độ hidden.

### 3.2.2.6. Denial-of-Service attack on the device

Tấn công DoS (Denial of Service) là phương pháp tấn công phổ biến vào các trang web trên Internet và mạng, và bây giờ là một tùy chọn tấn công vào thiết bị đang mở Bluetooth. Phương pháp này rất đơn giản, chỉ là kẻ tấn công dùng máy tính có mở Bluetooth kết hợp với một phần mềm đặc biệt yêu cầu thiết bị của nạn nhân phải liên tục trả lời những yêu cầu làm cho pin hao nhanh chóng, đồng thời phải duy trì yêu cầu kết nối bất hợp pháp nên thiết bị tạm thời bị vô hiệu hóa.

Tấn công DoS thực hiện trên bất kỳ thiết bị Bluetooth trong tình trạng “có thể tìm ra” (discoverable) nhưng đối với “hacker cao cấp” thì có thể phát hiện được thiết bị Bluetooth “không thể tìm ra” (non-discoverable). Vì thế, nhóm Bluetooth SIG đang cố gắng tạo ra những biện pháp bảo mật hơn để trong tương lai những thiết bị “không thể phát hiện ra” sẽ không bị “nhìn xuyên thấu” như thế.

#### Ảnh hưởng:

- DoS chỉ cho phép hacker tạm thời quấy nhiễu một ai đó chứ không cho phép truy cập vào dữ liệu hoặc dịch vụ, nên không có bất kỳ thông tin nào bị sử dụng hoặc bị đánh cắp.
- Ngày nay tấn công DoS vào thiết bị Bluetooth chỉ còn được thực hiện trong phòng thí nghiệm kiểm tra như một thủ tục tối thiểu và bình thường của kỹ thuật không dây Bluetooth.

### 3.2.2.7. Disclosure of keys

Một thiết bị Bluetooth gắn với máy tính có thể trao đổi nhầm với người có mục đích lấy trộm link key.

Một USB plug hoặc PCMCIA (Personal Computer Memory Card International Association) có thể bị lấy ra khỏi máy tính của người chủ và đưa

vào máy của “đối thủ” và một hay nhiều key bị đọc trộm mà chủ nhân không hề biết.

Những phần mềm xấu (Trojan horse) trá hình thành một chương trình bình thường để gửi cơ sở dữ liệu của key về cho những kẻ xấu muốn truy cập. Nếu đoạn mã nguy hiểm này được kèm trong một con virus hay worm thì cuộc tấn công này sẽ nhanh chóng lan rộng ra trên số lớn thiết bị. Một khi link key của máy tính và điện thoại (và BD\_ADDR của máy tính) bị lộ thì kẻ thù có thể kết nối bí mật vào điện thoại di động với vai trò của máy tính và sử dụng bất kỳ dịch vụ nào trên điện thoại đó thông qua Bluetooth.

### **3.2.2.8. Unit key attacks**

Một thiết bị dùng unit key thì chỉ sử dụng duy nhất một key cho tất cả các liên kết an toàn của nó. Do đó nó chia sẻ key này cho tất cả những thiết bị khác mà nó tin tưởng. Vì thế một thiết bị “trusted” (đã có unit key) có thể nghe trộm những thông điệp xác nhận ban đầu giữa hai thiết bị hoặc bất kỳ cuộc trao đổi nào giữa các thiết bị này. Nó còn có thể giả dạng để phân phát unit key.

Rủi ro tiềm tàng với unit key đã được Bluetooth SIG phát hiện ra. Lúc đầu unit key được sử dụng để giảm nhu cầu bộ nhớ ở những thiết bị hạn chế và còn được giữ lại vì lý do tương thích của chuẩn.

### **3.2.2.9. Backdoor attack**

Backdoor attack bao gồm thiết lập một mối quan hệ tin tưởng thông qua cơ chế pairing, nhưng phải bảo đảm rằng nó không xuất hiện nữa trên danh sách các thiết bị đã paired của thiết bị đích. Bằng cách này trừ khi người sử dụng thật sự chú ý đến thiết bị của họ đúng lúc thiết lập kết nối, nếu không họ sẽ không chắc được thông báo chuyện xảy ra và kẻ tấn công tiếp tục sử dụng bất cứ tài nguyên nào mà một thiết bị trusted được phép truy cập bao gồm dữ liệu, dịch vụ Internet, WAP và GPRS mà chủ nhân không hề hay biết. Khi Backdoor đã được thực hiện thì tấn công theo Bluesnarf sẽ hoạt động được trên thiết bị mà trước đây đã từ chối truy cập, và không hề bị những hạn chế của Bluesnarf ảnh hưởng.

### **3.2.2.10. Pairing attack**

Đặc điểm kỹ thuật của Bluetooth 1.1 dễ bị ảnh hưởng từ các cuộc tấn công trong quá trình pairing. Pairing attack chỉ thực hiện được khi attacker có mặt ngay thời điểm pairing, vốn chỉ xảy ra một lần giữa một cặp thiết bị. Nếu quá trình pairing được thực hiện ngay nơi công cộng như lúc kết nối với access point, máy in... thì nguy cơ cao hơn.

### 3.2.2.11. BlueStumbling = BlueSnarfing

Bluesnarfing cho phép kết nối vào thiết bị mà không hề cảnh báo cho chủ thiết bị và giành quyền truy cập vào những vùng hạn chế của dữ liệu như phonebook (và bất kỳ image cũng như dữ liệu liên kết với nó), calendar, realtime clock, business card, properties, change log, IMEI (International Mobile Equipment Identity, “nhân dạng” duy nhất của điện thoại trong mạng mobile, và sẽ bị sử dụng ở điện thoại “nhái”). Tấn công thường chỉ thực hiện khi thiết bị đang ở chế độ “nhìn thấy được” (“discoverable” hoặc “visible”). Bluesnarfing có lẽ khai thác một khe hở do quá trình mặc định password của pairing (thường chỉ 4 ký tự), nó bị đoán ra đồng thời thiết bị Bluetooth được bật lên và chế độ nhìn thấy là “all”. Không cần những thiết bị đặc biệt, hacker có thể tấn công thiết bị trong khoảng cách 10 m với một phần mềm đặc biệt (tuy nhiên với “Khẩu súng trường” BlueSniper, do John Hering và các đồng sự chế tạo có gắn ống ngắm và ăngten, nối với laptop Bluetooth hoặc PDA đặt trong ba lô có khả năng thu nhận dữ liệu từ ĐTDĐ cách nó 1,8 km). Nhưng chỉ những thiết bị Bluetooth đời cũ khi bật Bluetooth mới dễ nhạy cảm với bluesnarfing.

Cũng có thể gọi tấn công kiểu này là OBEX Pull Attack: OBEX cho phép bạn trong một số trường hợp có thể nặc danh để kéo (PULL) những mục chọn giữa hai thiết bị.

#### Ảnh hưởng:

- Một số thiết bị cầm tay của Nokia, Ericsson & Sony Ericsson và nhiều điện thoại thông dụng đều nhạy cảm với kiểu tấn công này.
- Phụ thuộc rất nhiều vào việc thực thi của OBEX/Bluetooth stack.

- Thông tin bị lấy có thể quan trọng như calendar, real time clock, business card, properties,
- Change log, IMEI.
- Có nhiều thiết bị có yếu điểm này.

### 3.2.2.12. BlueBug attack

BlueBug attack tạo một kết nối serial profile đến thiết bị, bằng cách đó có thể lấy được toàn bộ quyền truy cập vào tập lệnh AT (AT command set), sau đó có thể khai thác để sử dụng shelf tool như PPP cho mạng và gnokii cho message, quản lý contact, nghe lén những cuộc trò chuyện điện thoại, làm lệch hướng hoặc thực hiện cuộc gọi tới những số trả tiền cước cao, gửi và đọc sms message, kết nối Internet...thực hiện voice call thông qua mạng GSM đến mọi nơi trên thế giới. Thiết lập việc chuyển hướng cuộc gọi làm những cuộc gọi đến người chủ bị chặn đứng, cung cấp những kênh gọi có đích đến đắt tiền...

Như mọi cuộc tấn công khác, hacker phải đứng trong phạm vi 10 m gần điện thoại.

### 3.2.2.13. PSM Scanning

Không phải tất cả cổng PSM (Protocol/Service Multiplexer ports) đều được đăng ký với SDP địa phương (Service Discovery Protocol). Vì thế nếu chúng ta bỏ qua cơ sở dữ liệu của SDP và cố gắng liên tục kết nối với PSM chúng ta có thể định vị được một cổng “ẩn”.

Ảnh hưởng: Ý tưởng này thường tạo nên Backdoor attack.

### 3.2.2.14. On-line PIN cracking

- Tấn công chỉ thực hiện được khi tìm ra số PIN đã dùng trước đó của thiết bị (cùng một số PIN cho mỗi lần kết nối).
- Mỗi lần cần phải thay đổi địa chỉ Bluetooth và số PIN khác nhau.
- Những đặc điểm kỹ thuật không cung cấp giải pháp cho yếu điểm này.

### 3.2.2.15. A man-in-the-middle attack using Bluetooth in a WLAN interworking environment

Một man-in-the-middle attack có thể thực hiện được trên liên kết Bluetooth trong môi trường mạng WLAN environment. Attacker sẽ nhử nạn nhân kết nối vào một access point WLAN “nguy hiểm”. Chúng không cần biết Bluetooth link key và có thể lặp lại cách tấn công này nhiều lần với cùng một nạn nhân trên bất kỳ mạng WLAN nào.

### **3.2.2.16. Off-line encryption key (via Kc)**

Mở rộng từ Kinit recovery attack.

### **3.2.2.17. Attack on the Bluetooth Key Stream Generator**

Phá vỡ tính an toàn của sự mã hóa, tấn công vào Linear Feedback Shift Register Work (sự cố gắng của khoảng  $2^{67,58}$  phép tính).

### **3.2.2.18. Replay attacks**

Hacker có thể ghi lại việc truyền thông trên cả 79 kênh của Bluetooth và sau đó tính toán ra trình tự bước nhảy và thực hiện lại cả cuộc truyền tin đó.

### **3.2.2.19. Man-in-the-middle attack**

Can thiệp vào truyền thông trong quá trình pairing.

### **3.2.2.20. Denial-of-Service attack on the Bluetooth network**

Không khả thi lăm vì phải làm tắc nghẽn cả dãy tầng ISM

Ngoài ra còn một số cách tấn công khác :

- Off-line PIN (via Kinit) recovery
- Reflection Attack
- Impersonate original sending/receiving unit

### **Kết luận :**

Sự gia tăng nhanh chóng của các thiết bị Bluetooth làm cho việc truyền thông không dây trở nên dễ dàng hơn và các nhóm Bluetooth muốn bạn tin rằng kỹ thuật này an toàn trước hacker. Tuy nhiên với “khẩu súng trường” BlueSniper, những thành viên của Flexilis (nhóm chuyên gia về kỹ thuật không dây ở Los Angeles) có thể quét và tấn công vào các thiết bị trong vòng một dặm ( $\approx 1.6$  km). Phiên bản đầu tiên của khẩu súng này do John Hering và các

đồng sự chế tạo có gắn ống ngắm và ăngten, nối với laptop Bluetooth hoặc PDA đặt trong ba lô đã được trình diễn tại hội nghị về hacker và bảo mật "Black Hat and DefCon" tại Las Vegas (Mỹ) năm 2004. So với phiên bản cũ, khẩu súng mới này có vẻ chuyên nghiệp hơn, lớn hơn, mạnh hơn, bền hơn và anten thu sóng mạnh gấp hai lần kiểu cũ. Nó cũng có một máy tính nhỏ để lọc những thứ cần thiết trước khi đưa vào laptop tập hợp dữ liệu lại. Và làm thiết bị này theo John Hering thì không khó lầm, chỉ mất khoảng vài trăm USD và một buổi chiều.

Gần đây nhất một báo cáo của 2 nhà nghiên cứu về an toàn bảo mật người Israel (có một người đang là nghiên cứu sinh) đã gây sốc khi họ có thể giành quyền điều khiển Bluetooth-tích hợp vào điện thoại di động, ngay cả khi tính năng an toàn của handset đã được bật lên. Điều này được thực hiện dựa trên kỹ thuật tấn công Ollie Whitehouse of @Stake đã miêu tả năm rồi. Điều khác biệt và rất quan trọng là kỹ thuật cũ đòi hỏi hacker phải lắng nghe quá trình pairing giữa 2 thiết bị, còn kỹ thuật mới cho phép hacker buộc 2 thiết bị phải lặp lại quá trình pairing này, theo cách đó hacker có cơ hội xác định được số PIN dùng để bảo vệ kết nối chỉ trong khoảng từ 0.06 đến 0.3 giây. Sau đó họ có thể sử dụng số PIN này để kết nối vào Bluetooth handset mà không cần sự cho phép. Và một khi kết nối được thiết lập, kẻ tấn công có thể thực hiện yêu cầu trên thiết bị, lấy thông tin và lắng nghe việc truyền dữ liệu giữa thiết bị này với các thiết bị khác.

Họ giả dạng một trong 2 thiết bị, gửi một thông điệp đến thiết bị kia yêu cầu phải quên link key. Điều này thúc giục thiết bị đó hủy key và sau đó cả 2 thiết bị bắt đầu thực hiện lại tiến trình pairing.

### 3.2.3. Virus:

#### 3.2.3.1. *Appdisabler.B*

##### 3.2.3.1.1. Thông tin

Tên: Appdisabler.B

Loại virus: Trojan

Ngày giờ ngăn chặn sự xâm nhập: 17/5/2005

### 3.2.3.1.2. Mô tả chi tiết thông tin virus

Appdisabler.B là một loại thuộc Trojan được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60.

Appdisabler.B được đóng gói trong file Freetalktime.sis.

Khi cài đặt, Appdisabler.B sẽ thay thế phần thực thi chính của các ứng dụng khác bằng cách ghi đè lên những file chính.

Những file bị vô hiệu hóa:

AD7650

AnswRec

BlackList

BlueJackX

callcheater

CallManager

Camcoder

camerafx

ETICamcorder

ETIMovieAlbum

ETIPlayer

extendedrecorder

FaceWarp

FExplorer

FSCaller

Hair

HantroCP

irremote

Jelly

KPCaMain

Launcher

logoMan

MIDIED

mmp

Mp3Go

Mp3Player

photoacute

PhotoEditor

Photographer

PhotoSafe

PhotoSMS

PVPlayer

RallyProContest

realplayer

RingMaster

SmartAnswer

SmartMovie

SmsMachine

Sounder

sSaver

SystemExplorer

UltraMP3

UVSMStyle

WILDSKIN

Có thể quét virus bằng cách dùng trình quản lý ứng dụng để gỡ bỏ

Freetalktime.sis và cài đặt lại những ứng dụng đã bị tổn hại.

### **3.2.3.2. *Cabir.Dropper***

#### **3.2.3.2.1. Thông tin**

Tên: Cabir.Dropper

SymbOS/Cabir.Dropper,

Norton AntiVirus 2004 Professional.sis

Loại virus: Worm

Ngày giờ ngăn chặn sự xâm nhập: 17/5/2005

### 3.2.3.2.2. Mô tả chi tiết thông tin virus

Cabir.Dropper là một dạng file cài đặt của hệ điều hành Symbian và nó sẽ cài Cabir.B, Cabir.C và Cabir.D vào thiết bị đồng thời vô hiệu hóa ứng dụng điều khiển Bluetooth. File gốc của Cabir.Dropper tên là Norton AntiVirus 2004 Professional.sis.

Cabir.Dropper sẽ cài những biến thể của Cabir vào những nơi khác nhau trong hệ thống file của thiết bị. Một số Cabir sẽ thay thế các ứng dụng, cho nên khi người dùng cài đặt vào sẽ bị thay bằng Cabir.D đồng thời không hiển thị biểu tượng của ứng dụng đó.



Hình 3-10 Màn hình điện thoại nhiễm Cabir.D

Nếu người dùng nhấp vào biểu tượng này thì Cabir.D sẽ được kích hoạt và tự lây lan sang các thiết bị khác dưới dạng file ([YUAN].SIS).

Cabir.Dropper cũng tự cài đặt thành phần giúp kích hoạt Cabir.D khi thiết bị khởi động lại, nhưng ở đây có lỗi là thành phần này lại trả vào thư mục không được cài vào hệ thống.

Khi Cabir.Dropper được cài vào hệ thống, nó sẽ cài file vào những thư mục sau:

```
\images\  
\sounds\digital
```

```
\system\apps  
\system\install  
\system\recogs  
\system\apps\btui  
\system\apps\fexplorer  
\system\apps\file  
\system\apps\freakbtui  
\system\apps\smartfileman  
\system\apps\smartmovie  
\system\apps\systemexplorer  
\system\apps\[yuan]
```

Một số Cabir được cài vào thư mục cài mặc định của các ứng dụng như FExplorer, SmartFileMan, Smartmovie and SystemExplorer.

Có thể quét virus bằng cách xóa những file worm, sau đó dùng chương trình quản lý ứng dụng để gỡ bỏ Norton AntiVirus 2004 Professional.sis.

Nếu điện thoại bạn bị nhiễm Cabir và không cài được file thông qua Bluetooth, bạn có thể tải chương trình quét virus F-Secure Mobile Anti-Virus trực tiếp vào máy.

### **3.2.3.3. *Cabir – A***

#### **3.2.3.3.1. Thông tin:**

Tên: Symb/Cabir-A

Loại virus: Worm (Sâu)

Các bí danh: Cabir

Epoc.Cabir

EPOC/Cabir.A

Worm.Symbian.Cabir.a

Symbian/Cabir.b

#### **3.2.3.3.2. Mô tả thông tin chi tiết:**

Symb/Cabir-A là worm, biết đến như là malware, được viết dành riêng cho điện thoại di động dòng máy đang chạy hệ điều hành Symbian Series 60.

Đây là virus đầu tiên được phát tán từ website của một nhóm tin tặc có tên 29A. Virus này lây từ các địa chỉ website mà người sử dụng có thể tải về máy nhạc chuông, trò chơi... hoặc từ một điện thoại bị nhiễm khác. "Cabir" tự giả dạng như là một công cụ của hệ điều hành Symbian với tên gọi "Caribe Security Manager" và được gửi đi dưới dạng một file Caribe.SIS của hệ điều hành. Gói tin chứa 3 thành phần caribe.app, flo.mdl và caribe.rsc. Nếu bấm Yes, file này sẽ được nhận và lưu vào Inbox trong tin nhắn. Tiếp tục click vào file này, nó sẽ cài đặt vào ĐTDĐ như một ứng dụng thông thường và sau đó sẽ lây nhiễm. Các thành phần được cài đặt vào \system\apps\directory trên thiết bị. Khi bị nhiễm, ĐTDĐ sẽ hiện chữ Cabire mỗi khi bật máy và Cabire sẽ thông qua cổng Bluetooth của ĐTDĐ bị nhiễm liên tục tìm kiếm các ĐTDĐ khác đang bật Bluetooth để lây lan sang. Cabire cũng có thể lây lan thông qua một file đính kèm gửi qua e-mail ĐTDĐ. Cabire làm hao pin rất nhanh vì nó liên tục kích hoạt Bluetooth và tìm kiếm các ĐTDĐ khác trong bán kính vài chục mét cũng có cổng Bluetooth để lây lan sang.

Hiện nay Cabir chỉ tấn công điện thoại di động sử dụng hệ điều hành Symbian series 60 và người sử dụng có thể chủ động không nhận và cài đặt malware này vào máy. Để phòng tránh không nên mở tính năng Bluetooth nơi đông người, đặc biệt là các quán cà phê tập trung nhiều loại máy Symbian. Nếu thấy file Caribe.sis được gửi đến máy, hãy từ chối và tắt Bluetooth để không tiếp tục được "mời" nhận.

Mục tiêu của Cabir không phải là tấn công mà chủ yếu là để phô trương thanh thế, chứng minh khả năng thiết bị cầm tay cũng có thể bị virus tấn công. Cabir không chứa mã độc hại với khả năng phá huỷ file, song nếu người sử dụng làm theo những yêu cầu của virus, nó sẽ khiến pin điện thoại của họ cạn kiệt năng lượng.

### **3.2.3.4. Cabir – B**

#### **3.2.3.4.1. Thông tin:**

Tên gọi: Symb/Cabir-B

Loại Virus: Worm (Sâu)

Ngày giờ ngăn chặn được sự xâm nhập của Symb/Cabir: 30/11/2004  
14:31:33 (GMT)

### 3.2.3.4.2. Mô tả chi tiết thông tin virus:

- \_ Symb/Cabir-B là một loại thuộc họ virus Worm (sâu) được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60.
- \_ Giống Symb/Cabir-A, một khi Symb/Cabir-B được kích hoạt, chúng sẽ cố gắng thử gửi chính bản sao đến thiết bị có kích hoạt Bluetooth được tìm thấy trong tầm hoạt động của máy bị nhiễm virus.
- \_ File camtimer của virus Symb/Cabir-B có thể được cài bởi một loại Trojan có tên là Troj/Skulls-B (được trình bày ở phần trên)
- \_ Symb/Cabir-B cài 2 file có tên camtimer.rsc và camtimer.app mà 2 file này là thành phần của một ứng dụng camera timer (định giờ máy ảnh) vô hại.
- \_ Symb/Cabir-B được mở ra (bung hay giải nén) từ gói Symbian SIS có tên là camtimer.sis. Gói này chứa các thành phần con được giải nén và chép vào thư mục ./System/Apps, ./System/CARIBESECURITYMANAGER và ./System/Recogs:

Các file được hình thành trong các thư mục:

```
./system/apps/CamTimer/camtimer.rsc
./system/apps/CamTimer/camtimer.app
./system/apps/caribe/flo.mdl
./system/apps/caribe/caribe.rsc
./system/apps/caribe/caribe.app
./system/CARIBESECURITYMANAGER/caribe.rsc
./system/CARIBESECURITYMANAGER/caribe.app
./system/CARIBESECURITYMANAGER/CAMTIMER.sis
./system/RECOGS/flo.mdl
```

Riêng file Flo.mdl là một file DLL (Dynamic Linked Library: Thư viện liên kết động) dùng kỹ thuật EZBoot để có gắng chạy file ứng dụng (application) caribe.app mỗi khi thiết bị được mở lên.

### **3.2.3.5. Cabir.Y**

#### **3.2.3.5.1. Thông tin**

Tên: Cabir.Y

Bí danh: SymbOS/Cabir.Y, EPOC/Cabir.Y, Worm.Symbian.Cabir.Y

Loại virus: Worm

Ngày giờ ngăn chặn sự xâm nhập: 13/12/2004

#### **3.2.3.5.2. Mô tả chi tiết thông tin virus**

Cabir Y là một biến thể nhỏ hơn của Cabir.B và khác ở chỗ Cabir.Y lan truyền bằng file symTEE.SIS trong khi Cabir.B là file Caribe.sis.

F-Secure Mobile Anti-Virus có thể quét virus bằng cách xóa những file worm, sau đó bạn có thể xóa thư mục C:\SYSTEM\SystemShareddatas\JBguan-all-by-symteeq\

Nếu điện thoại bạn bị nhiễm Cabir và không cài được file thông qua Bluetooth, bạn có thể tải chương trình quét virus F-Secure Mobile Anti-Virus trực tiếp vào máy.

### **3.2.3.6. Commwarrior.A**

#### **3.2.3.6.1. Thông tin**

Tên: Commwarrior

SymbOS/ Commwarrior.A

Loại virus: Worm

Nguồn gốc: Russia

Ngày giờ ngăn chặn sự xâm nhập: 7/3/2005

#### **3.2.3.6.2. Mô tả chi tiết thông tin virus**

Comwarrior là worm (sâu virus) được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60. Nó có khả năng lây lan thông qua Bluetooth và tin nhắn MMS.

Comwarrior sẽ tạo một bản sao là file SIS với tên bất kỳ, có chứa file thực thi chính của worm commwarrior.exe và thành phần khởi động commrec.mdl. File SIS chứa phần tự kích hoạt cho virus.

Khi Comwarrior nhiễm vào máy, nó sẽ tự động tìm những điện thoại khác trong tầm hoạt động và đang mở Bluetooth để gửi bản sao là file .SIS đến các điện thoại đó. Những file này có tên bất kỳ nên người sử dụng khó tránh bị lây nhiễm.Thêm vào đó khi lây qua đường Bluetooth, Comwarrior sẽ đọc tên người và số điện thoại trong phonebook, sau đó gửi tin nhắn MMS có chứa file SIS có chứa sâu đến những địa chỉ đó.

Comwarrior chứa những file text:

CommWarrior v1.0 (c) 2005 by e10d0r

ATMOS03KAMA HEAT!

Có thể diệt Comwarrior bằng cách cài F-Secure Mobile Anti-Virus.

Nếu điện thoại bạn bị nhiễm Cabir và không cài được file thông qua Bluetooth, bạn có thể tải chương trình quét virus F-Secure Mobile Anti-Virus trực tiếp vào máy.

Sau khi quét virus trong điện thoại, bạn có thể xóa những thư mục trống và gỡ bỏ file SIS chứa Comwarrior.

### Lây qua Bluetooth:

Cơ chế hoạt động của Comwarrior khác với Cabir. Cabir chỉ chốt vào một điện thoại mà nó tìm thấy đầu tiên trong tầm ảnh hưởng, và chỉ lây sang một điện thoại khác khi hệ thống được khởi động lại. Comwarrior thì sẽ tìm thiết bị mới ngay sau khi đã gửi bản sao cho thiết bị đầu tiên. Do đó nó có thể lây lan qua tất cả mọi thiết bị có thể tiếp xúc được. Vì vậy tốc độ lây của Comwarrior nhanh hơn Cabir rất nhiều. Comwarrior chỉ lây qua Bluetooth từ 08:00 đến 23:59, tùy thuộc vào đồng hồ của điện thoại nhiễm.

### Lây qua MMS

Comwarrior lây qua MMS bằng cách gửi tin nhắn MMS có kèm file commw.sis chứa virus đến người sử dụng khác.



Hình 3-11 Tin nhắn MMS có kèm sâu Comwarrior

Tên file SIS kèm theo luôn giống nhau, không giống như khi lây qua Bluetooth (tên file SIS luôn thay đổi sau mỗi lần gửi).

Comwarrior thường dùng những dòng sau trong tin nhắn MMS:

Norton AntiVirus Released now for mobile, install it!  
Dr.Web New Dr.Web antivirus for Symbian OS. Try it!  
MatrixRemover Matrix has you. Remove matrix!  
3DGame 3DGame from me. It is FREE !  
MS-DOS MS-DOS emulator for SymbianOS. Nokia series 60 only. Try it!  
PocketPCemu PocketPC \*REAL\* emulator for Symbian OS! Nokia only.  
Nokia ringtoner Nokia RingtoneManager for all models.  
Security update #12 Significant security update. See [www.symbian.com](http://www.symbian.com)  
Display driver Real True Color mobile display driver!  
Audio driver Live3D driver with polyphonic virtual speakers!  
Symbian security update See security news at [www.symbian.com](http://www.symbian.com)  
SymbianOS update OS service pack #1 from Symbian inc.  
Happy Birthday! Happy Birthday! It is present for you!  
Free SEX! Free \*SEX\* software for you!  
Virtual SEX Virtual SEX mobile engine from Russian hackers!  
Porno images Porno images collection with nice viewer!  
Internet Accelerator Internet accelerator, SSL security update #7.  
WWW Cracker Helps to \*CRACK\* WWW sites like hotmail.com

Internet Cracker It is \*EASY\* to \*CRACK\* provider accounts!

PowerSave Inspector Save you battery and \*MONEY\*!

3DNow! 3DNow!(tm) mobile emulator for \*GAMES\*.

Desktop manager Official Symbian desctop manager.

CheckDisk \*FREE\* CheckDisk for SymbianOS released!MobiComm

### Lây nhiễm:

Comwarrior được cài đặt vào những thư mục:

\system\apps\CommWarrior\commwarrior.exe

\system\apps\CommWarrior\commrec.mdl

Khi thực thi nó chép những file sau:

\system\updates\commrec.mdl

\system\updates\commwarrior.exe

Và tạo thành bản sao:

\system\updates\commw.sis

Rồi bắt đầu lây qua MMS

Comwarrior chỉ lây qua MMS từ 00:00 đến 06:59 tùy thuộc vào đồng hồ điện thoại.

#### 3.2.3.7. Dampig.A

##### 3.2.3.7.1. Thông tin

Tên: Dampig.A

Bí danh: SymbOS/Dampig.A, FSCaller crack trojan

Loại virus: Trojan

Ngày giờ ngăn chặn sự xâm nhập: 13/12/2004

##### 3.2.3.7.2. Mô tả chi tiết thông tin virus

Dampig.A là một file SIS, giả dạng bản crack của ứng dụng FSCaller 3.2 (Fscaller3.2Crack7610.sis hoặc vir.sis). Dampig.A vô hiệu hóa Bluetooth, quản

Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa

lý file hệ thống, ứng dụng tin nhắn và phone book, đồng thời cài đặt một số sâu Cabir vào máy.

Dampig.A cũng làm sai lạc thông tin gỡ cài đặt nên không thể gỡ bỏ nó mà chưa quét virus trước.

Danh sách ứng dụng vẫn còn nên người dùng có thể dùng điện thoại tải chương trình quét virus để diệt mà không cần công cụ đặt biệt nào.

Có thể xóa những ứng dụng mà bạn nghi ngờ hoặc dùng F-Secure Mobile Anti-Virus để quét. Sau khi xóa file worm bạn có thể gỡ bỏ file Fscaller3.2Crack7610.sis bằng trình quản lý ứng dụng.

Nếu điện thoại bạn bị nhiễm Cabir và không cài được file thông qua Bluetooth, bạn có thể tải chương trình quét virus F-Secure Mobile Anti-Virus trực tiếp vào máy.

#### **Những ứng dụng bị vô hiệu hóa:**

Bluetooth UI

Camera

FExplorer

Messaging

Phonebook

SmartFileManager

Smartmovie

SystemExplorer

UltraMP3

#### **3.2.3.8. Doomboot.A**

##### **3.2.3.8.1. Thông tin**

Tên: Doomboot.A

Bí danh: SymbOS/ Doomboot.A

Loại virus: Worm

Ngày giờ ngăn chặn sự xâm nhập: 7/3/2005

##### **3.2.3.8.2. Mô tả chi tiết thông tin virus**

Doomboot.A là một trojan làm sai hệ thống nhị phân và cài Commwarrior.B vào máy, làm thiết bị bị lỗi khi được khởi động lại.

Doomboot.A giả dạng file crack của Doom 2 (DFT\_S60\_v1.0.sis). Nếu người dùng cài vào máy thì vẫn không nhận được thông báo hay biểu tượng nào đồng thời Commwarrior.B lại chạy ẩn nén họ không cách nào biết được máy mình đã nhiễm virus.

Commwarrior.B cài đặt bởi Doomboo sẽ tự động kích hoạt và lây nhiễm. Do Commwarrior.B luôn kích hoạt Bluetooth nên điện thoại rất mau hết pin và nếu đây là Doomboot.A thì máy sẽ không thể khởi động lại sau khi hết pin.



Hình 3-12 Màn hình cài đặt Doomboot.A

Nếu máy bị nhiễm Doomboot.A thì nhất định không thể khởi động lại đồng thời phải tiến hành quét virus.

F-Secure Mobile Anti-Virus đều diệt được Doomboot.A và Commwarrior.B.

Nếu điện thoại bạn bị nhiễm Cabir và không cài được file thông qua Bluetooth, bạn có thể tải chương trình quét virus F-Secure Mobile Anti-Virus trực tiếp vào máy

### 3.2.3.9. *Drever – A*

#### 3.2.3.9.1. Thông tin:

- Tên gọi: Troj/Drever-A
- Hiệu ứng lè (Side effects): Tắt các chương trình hoặc ứng dụng diệt virus
- Biệt hiệu (Aliases):
  - Trojan.SymbOS.Drever.A
  - SymbOS/Drever.a!mdl
  - SymbOS\_DREVER.A

Ngày giờ ngăn chặn được sự xâm nhập của Troj/Drever-A: 24/3/2005  
14:46:02

### **3.2.3.9.2. Mô tả chi tiết thông tin virus:**

Troj/Drever-A là một loại thuộc Trojan được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60.

Troj/Drever-A xuất hiện dưới dạng một gói cài đặt (installation file) tên Antivirus.sis. Nếu file này được cài đặt lên thiết bị (hoặc điện thoại di động), tiến trình cài đặt sẽ cố gắng thử ghi đè lên các file có liên quan đến các chương trình hoặc ứng dụng diệt virus như sau:

C:\system\recogs\AVBoot.mdl  
C:\system\recogs\kl\_antivirus.mdl.

Điều này có thể ngăn ngừa hoặc làm tê liệt hoạt động của các chương trình như *Simworks and Kaspersky labs anti-virus* trong việc khởi động của các chương trình anti-virus này.

### **3.2.3.10. Drever – C**

#### **3.2.3.10.1. Thông tin:**

Tên gọi: Troj/Drever-C

Loại virus: Trojan

Hiệu ứng lè (Side effects): Tắt các chương trình hoặc ứng dụng diệt virus

Biệt hiệu (Aliases): Trojan.SymbOS.Drever.c

SymbOS/Drever.c!sis

SymbOS\_DREVER.C

Ngày giờ ngăn chặn được sự xâm nhập của Troj/Drever-C:  
25/03/2005 12:58:15 (GMT)

### 3.2.3.10.2. Mô tả chi tiết thông tin virus:

Troj/Drever-C là một loại thuộc Trojan được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60.

Troj/Drever-C xuất hiện dưới dạng một gói cài đặt (installation file) tên New\_bases\_and\_crack\_for\_antiviruses.sis. Nếu file này được cài đặt lên thiết bị (hoặc điện thoại di động), tiến trình cài đặt sẽ cố gắng thử ghi đè lên các file có liên quan đến các chương trình hoặc ứng dụng diệt virus như sau:

C:\system\recogs\AVBoot.mdl  
C:\system\recogs\kl\_antivirus.mdl  
C:\system\recogs\fsrec.mdl

Điều này có thể ngăn ngừa hoặc làm tê liệt hoạt động của các chương trình như *Simworks*, *Kaspersky labs* and *F-Secure anti-virus* trong việc khởi động của các chương trình anti-virus này.

### 3.2.3.11. Fontal.A

#### 3.2.3.11.1. Thông tin

Tên: Fontal.A

Bí danh: SymbOS/ Fontal.A

Loại virus: Trojan.

Ngày giờ ngăn chặn sự xâm nhập: 6/4/2005

#### 3.2.3.11.2. Mô tả chi tiết thông tin virus

Fontal.A là file cài đặt (Kill Saddam By OID500.sis) làm hư file Font của thiết bị, làm thiết bị bị lỗi trong lần khởi động lại kết tiếp.

Nếu điện thoại bị nhiễm Fontal.A, không được để nó khởi động lại trước khi diệt virus, nếu không điện thoại sẽ bị mắc kẹt ở quy trình khởi động và không sử dụng được nữa. Thêm vào đó việc làm hỏng file font làm ảnh hưởng đến trình quản lý ứng dụng, dẫn đến không chương trình nào cài vào được trước khi diệt virus.

Sau khi đã quét xong virus, bạn có thể xóa các thư mục trống còn lại và gỡ bỏ file SIS chứa virus.

### Lây nhiễm:

Khi file SIS được cài đặt, nó sẽ chép file vào những thư mục sau:

```
\system\apps\appmngr\appmngr.app  
\system\apps\kill_sadam\kill_sadam.app  
\system\apps\fonts\kill_sadam_font.gdr
```

### 3.2.3.12. Hobbes.A

#### 3.2.3.12.1. Thông tin

Tên: Hobbes.A

Bí danh: SymbOS/ Hobbes.A

Loại virus: Trojan

Ngày giờ ngăn chặn sự xâm nhập: 14/3/2005

#### 3.2.3.12.2. Mô tả chi tiết thông tin virus

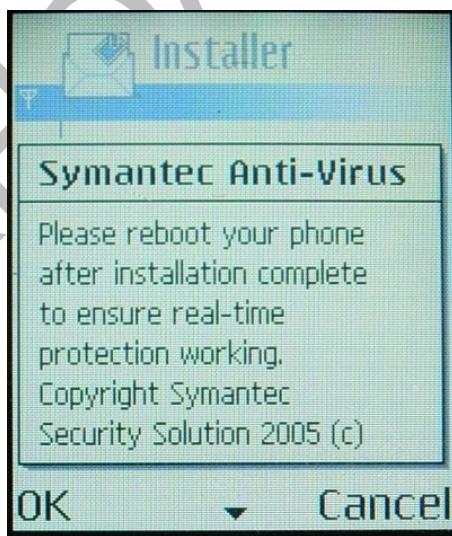
Hobbes.A là một file thực thi (Symantec.sis) làm hỏng hệ thống nhị phân, là nguyên nhân khiến cho quá trình tải ứng dụng về sẽ làm “sụp đổ” những điện thoại đời cũ dùng hệ điều hành Symbian.

Hobbes.A chỉ tấn công vào điện thoại dùng hệ điều hành Symbian 6.1, nghĩa là những điện thoại khác sẽ vô sự.



Hình 3-13 Màn hình yêu cầu cài đặt

Hobbes.A giả làm một bản sao của Symantec Anti-Virus cho điện thoại Symbian. Khi được cài đặt nó sẽ cố cài một phiên bản Fexplorer sai lạc để vô hiệu hóa trình quản lý file Fexplorer của máy nhưng lại sai thư mục nên không thực hiện được. Nó cũng cài một số phần vào ổ C và E trong đó có một phần làm hỏng quá trình boot máy của phiên bản cũ của hệ điều hành Symbian. Sau khi cài đặt nó sẽ yêu cầu thiết bị khởi động lại.



Hình 3-14 Màn hình ngay sau khi cài đặt xong

Hệ thống nhị phân của thiết bị sai nên ứng dụng hệ thống của hệ điều hành bị lỗi khi khởi động, không ứng dụng hệ thống nào được thực hiện. Nghĩa là

những ứng dụng cho điện thoại thông minh bị vô hiệu hóa hoàn toàn, chỉ gọi và nghe cuộc gọi là thực hiện bình thường.

Do đó người sử dụng không được khởi động lại thiết bị khi bị nhiễm virus vì Hobbes.A chỉ hoạt động khi khởi động lại, và khi đó chỉ cần dùng trình quản lý ứng dụng gỡ bỏ Symantec.sis là xong.

Khi đã lỡ khởi động lại thì người sử dụng nên:

1. Lấy memory card ra khỏi điện thoại và khởi động lại lần nữa.
2. Cài đặt chương trình quản lý file cho điện thoại.
3. Xóa file \system\recogs\recAutoExec.mdl trong memory card.
4. Gỡ bỏ Symantec.sis bằng trình quản lý ứng dụng.

### **3.2.3.13. Lasco.A**

#### **3.2.3.13.1. Thông tin**

Tên: Lasco.A

Bí danh: SymbOS/ Lasco.A, EPOC/ Lasco.A

Loại virus: Worm

Ngày giờ ngăn chặn sự xâm nhập: 10/1/2005

#### **3.2.3.13.2. Mô tả chi tiết thông tin virus**

Lasco.A là một worm sử dụng Bluetooth và file SIS để lây lan trên điện thoại di động sử dụng hệ điều hành Symbian Series 60.

Lasco.A tạo một bản sao (velasco, trong đó chứa phần thực thi chính velasco.app, nhận diện hệ thống marcos.mdl và resource file velasco.rsc) và gởi file này vào thư mục message inbox của điện thoại thông qua Bluetooth. Khi người dùng nhấp vào file này và cài đặt thì worm sẽ được kích hoạt và bắt đầu tìm kiếm thiết bị để lây thông qua Bluetooth.

Khi sâu Lasco tìm được thiết bị Bluetooth khác nó sẽ gởi bản sao của file velasco.sis đến thiết bị đó đến khi nào thiết bị này ra khỏi tầm sóng của nó. Giống như Cabir.H, Lasco.A có khả năng tìm thiết bị mới sau khi thiết bị đầu tiên ra khỏi vùng phủ sóng của nó.

Thêm vào đó, việc gởi bản sao qua Bluetooth được lặp lại bằng cách chèn sâu Lasco.A vào mọi file SIS khác trên thiết bị và gởi đến các thiết bị khác. Khi người dùng chọn file SIS bị lây nhiễm này, nó sẽ yêu cầu cài Velasco.



Hình 3-15 Màn hình yêu cầu cài đặt sâu Lasco.A

Nên nhớ rằng những file SIS bị nhiễm sâu sẽ không tự động gởi đi, nên cách duy nhất để bị nhiễm Lasco.A trên file SIS bị nhiễm (không phải là Velasco.SIS) là chép và cài đặt chúng.

Lasco.A được dựa vào source tương tự Cabir.H và rất giống nhau. Sự khác nhau chính là “thói quen” lây nhiễm của chúng.

Khi file velasco.sis file được cài đặt, nó sẽ chép các phần vào các thư mục sau:

```
c:\system\apps\velasco\velasco.rsc  
c:\system\apps\velasco\velasco.app  
c:\system\apps\velasco\flo.mdl
```

Khi velasco.app được thực thi, nó sẽ chép những file sau:

```
flo.mdl to c:\system\recogs  
velasco.app to c:\system\symbiansecuredata\velasco\  
velasco.rsc to c:\system\symbiansecuredata\velasco\
```

Lasco chỉ lây qua thiết bị hỗ trợ Bluetooth và phải là đang ở chế độ tìm thấy được (discoverable mode).

Nên để Bluetooth của thiết bị ở chế độ ẩn (hidden) để tránh bị nhiễm, nhưng khi đã bị nhiễm thì sâu sẽ cố lây nhiễm sang thiết bị khác dù người sử dụng đã tắt Bluetooth của thiết bị.

Có thể diệt sâu Lasco.A bằng F-Secure Mobile Anti-Virus, sau đó bạn xóa thư mục: c:\system\symbiansecuredata\velasco\

Nếu điện thoại bạn bị nhiễm Lasco.A và không cài được file thông qua Bluetooth, bạn có thể tải chương trình quét virus F-Secure Mobile Anti-Virus trực tiếp vào máy

### **3.2.3.14. *Locknut – B***

#### **3.2.3.14.1. Thông tin:**

Tên: Troj/Locknut-B

Loại virus: Trojan

Hiệu ứng lè: bỏ nhiều malware.

để lại những file không bị nhiễm trên máy.

#### **3.2.3.14.2. Mô tả thông tin chi tiết:**

- \_ Troj/Locknut-B là 1 Trojan trên thiết bị điện thoại di động được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60.
- \_ Trojan có thể đóng gói cài đặt file MMFPatch.sis.
- \_ Đôi khi các Trojan có thể đóng gói với worm Symb/Cabir-A.
- \_ Khi cài đặt file lần đầu tiên trên thiết bị, nó sẽ vài lần hiển thị những cảnh báo bảo mật. Có thể văn bản “MMFPatch” được hiển thị trong lúc cài đặt.
- \_ Troj/Locknut-B cài đặt nhóm file:

C:\system\apps\gavno\gavnoreturn.app

C:\system\apps\gavno\gavnoreturn.Rsc

C:\system\apps\gavno\gavnoreturn\_caption.Rsc

### **3.2.3.15. *Mabir.A***

### 3.2.3.15.1. Thông tin

Tên: Mabir.A

Bí danh: SymbOS/Mabir.A

Loại virus: Worm

Ngày giờ ngăn chặn sự xâm nhập: 13/12/2004

### 3.2.3.15.2. Mô tả chi tiết thông tin virus

Mabir.A là một loại worm được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60, có thể lây nhiễm qua Bluetooth và tin nhắn MMS.

Khi Mabir.A lây vào một điện thoại, nó sẽ bắt đầu tìm kiếm điện thoại khác trong tầm hoạt động của Bluetooth và gởi bản sao trong file caribe.sis (chứa caribe.app, caribe.rsc and flo.mdl.) đến điện thoại đó. Sau khi điện thoại này ra khỏi vùng phủ sóng nó vẫn sẽ tiếp tục gởi cho điện thoại này.

Bên cạnh việc lây lan qua Bluetooth, Mabir.A cũng lắng nghe tin nhắn MMS được gởi đến và trả lời những tin nhắn này bằng những tin MMS có kèm Mabir trong info.sis. Những tin này chỉ chứa file info.sis và không hề có bất kỳ dòng văn bản nào.

F-Secure Mobile Anti-Virus có thể quét virus bằng cách xóa những file worm. Nếu điện thoại bạn bị nhiễm Cabir và không cài được file thông qua Bluetooth, bạn có thể tải chương trình quét virus F-Secure Mobile Anti-Virus trực tiếp vào máy.

Sau khi diệt virus, bạn hãy xóa những thư mục rỗng và gỡ bỏ file SIS chứa Mabir.A (caribe.sis hoặc info.sis)

Khi file Mabir file được cài đặt, nó sẽ chép các phần vào các thư mục sau:

```
\system\apps\Caribe\Caribe.app  
\system\apps\Caribe\Caribe.rsc  
\system\apps\Caribe\flo.mdl
```

Khi Mabir.exe được thực thi, nó sẽ chép những file sau:

Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa

\system\symbiansecuredata\caribesecuritymanager\Caribe.app

\system\symbiansecuredata\caribesecuritymanager\Caribe.rsc

Và tạo file sis trong MMS:

\system\symbiansecuredata\caribesecuritymanager\Info.sis

### **3.2.3.16. MGDrop.A**

#### **3.2.3.16.1. Thông tin**

Tên: MGDrop.A

Bí danh: SymbOS/MGDrop, Metal Gear trojan

Loại virus: Trojan

Ngày giờ ngăn chặn sự xâm nhập: 13/12/2004

#### **3.2.3.16.2. Mô tả chi tiết thông tin virus**

MGDrop là một file cài đặt (Metal\_gear.sis), có thẻ vô hiệu hóa những ứng dụng quản lý file phổ biến và phần mềm Anti-Virus, đồng thời cài đặt Cabir.G vào điện thoại.

Cabir.G được tự động kích hoạt khi MGDrop được cài đặt và bắt đầu lây nhiễm. Khi Cabir.G lây nhiễm từ điện thoại nhiễm MGDrop, những file SIS mà nó gởi đi chỉ chứa Cabir.G chứ không có MGDrop. Tuy nhiên MGDrop cũng cài đặt Cabir.G vào thư mục khác như SEXXXX.SIS, nó không hiện lên trên danh sách các ứng dụng của điện thoại.

MGDrop cố gắng vô hiệu hóa F-Secure Mobile Anti-Virus bằng cách chép đè lên những file của chương trình bằng những file giả. Tuy nhiên F-Secure Mobile Anti-Virus vẫn có khả năng diệt Cabir.G trong MGDrop. Anti-Virus sẽ dò tìm file SIS nhiễm sâu ngăn nó cài đặt miễn là Anti-Virus được đặt ở chế độ realtime scan mode như mặc định.

Sau Cabir.G gởi kèm theo MGDrop vẫn diệt được bằng Cabir.Gen. Vì thế MGDrop vẫn bị diệt và ngăn chặn mà không cần cập nhật dữ liệu của Anti-Virus.

#### **Diệt virus với 2 điện thoại series 60**

Để diệt hoàn toàn MGDropster trong điện thoại nhiễm (DT1), bạn cần được một điện thoại series 60 khác, không bị nhiễm virus giúp đỡ và phải xóa sạch memory card của điện thoại đó (DT2).

Lấy bộ công cụ F-Skulls từ <ftp://ftp.f-secure.com/anti-virus/tools/f-skulls.zip> hoặc lấy trực tiếp vào điện thoại từ <http://www.europe.f-secure.com/tools/f-skulls.sis>

1. Cài đặt F-Skulls.sis vào DT2.
2. Đặt memory card với F-Skulls vào DT1.
3. Khởi động DT1, danh sách ứng dụng sẽ làm việc.
4. Đến trình quản lý ứng dụng gỡ bỏ file SIS mà bạn dùng để cài Skull.
5. Chép và cài F-Secure Mobile Anti-Virus để gỡ bất kỳ Cabirs kèm theo Skulls từ

<http://www.europe.f-secure.com/estore/avmobile.shtml> hoặc <http://mobile.f-secure.com/>

6. Cài đặt trình quản lý file EFilemanager vào DT1 từ <http://www.psiloc.com/>

7. Đưa memory card của DT1 vào vào DT1 và xóa những file

E:\System\Apps\SystemExplorer\SystemExplorer.app

E:\System\Apps\smartfileman\smartfileman.app

E:\System\Apps\file\file.app

E:\System\Apps\Anti-Virus\Anti-Virus.app

E:\System\Apps\Anti-Virus\FsAVUpdater.app

E:\System\Apps\AppInst\Appinst.aif

E:\System\Apps\AppInst\Appinst.app

E:\System\Apps\cabirfix\cabirfix.app

E:\System\Apps\Decabir\DECABIR.APP

E:\System\Apps\Disinfect\Disinfect.app

E:\System\Apps\FExplorer\FExplorer.app

#### **Những ứng dụng bị vô hiệu hóa:**

Simworks Anti-Virus

F-Secure Mobile Anti-Virus

Application installer

Cabirfix

Decabir

F-Cabir

FExplorer

File manager

Smart file manager

System Explorer

### 3.2.3.17. *Mosquito Trojan*

#### 3.2.3.17.1. Thông tin:



Hình 3-16 Mosquito Trojan

Hiệu ứng lè: Người dùng sẽ phải bất ngờ khi thanh toán khoản bill rất cao do trojan lén gửi các tin nhắn dạng text SMS đắt tiền mà người dùng không hay biết đến các số premium rate.

Trojan này hiện đã được cảnh báo ở nhiều website cũng như các mạng peer-to-peer

#### 3.2.3.17.2. Mô tả chi tiết thông tin virus:

- \_ Troj/Drever-A là một loại thuộc Trojan được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60.

- Ân sâu bên trong Trojan horse là dòng thông báo sau:

This version has been cracked by SODDOM BIN LOADER No rights reserved. Pirate copies are illegal and offenders will have lotz of phun!!!

- Trojan xuất hiện dưới dạng cài trang như là một phiên bản crack của trò chơi “Mosquito” (Bắn muỗi)(như trong hình vẽ) mà game này có thể cài đặt dễ dàng trên các điện thoại thông minh (smartphone) hiện đại.

### 3.2.3.18. *Skulls – A*

#### 3.2.3.18.1. Thông tin

- Tên: Troj/Skulls-A
- Loại Virus: trojan
- Cách thức lan truyền: wed downloads.
- Hiệu ứng lè (side effects): thay đổi, sửa đổi dữ liệu trên máy.

#### 3.2.3.18.2. Mô tả chi tiết thông tin virus

- Troj/Skulls-A là một loại thuộc họ trojan được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60. Trojan này thường được đính kèm theo những phần mềm shareware dùng cho các máy tính dùng hệ điều hành Symbian, thông thường là những file mở rộng kiểu.sis(Extended Theme.sis installation file).
- Khi những file.sis này được cài đặt trên máy, nó sẽ tạo ra một số lượng file trên vùng Ram của máy (thông thường là ổ c:\), đồng thời cũng sẽ tạo ra trên Rom (ổ z:\) những file tương tự, khi đó tất cả các ứng dụng hệ thống trên máy sẽ bị thay thế bởi các gói chương trình do file.sis tạo ra.
- Những chương trình trong hệ điều hành Symbian bao gồm một số ít file với phần đuôi mở rộng là.app, những thông tin của chương trình thì thường được đặt trong những file với phần đuôi là.aif. File AIF chứa

đựng icon của chương trình và một con trỏ trỏ đến file chương trình chính.

- \_ Troj/Skulls-A sẽ tạo ra một file AIF mới chứa đựng icon trông giống như đầu lâu người, và icon này không trỏ đến file chương trình chính. Troj/Skulls-A sẽ vô hiệu hóa các chức năng của máy, mặc dù máy vẫn có thể còn gọi được.



Hình 3-17 Trojan/Skulls-A

- \_ Trojan/Skulls-A chứa một file text, file text này sẽ được hiện thị trong suốt quá trình cài đặt: “Extended Theme is an advanced Theme Manager for 7610. It uses to manage, edit, & create themes using your 7610. Tee-222 takes no responsibility for any kind of results caused by this app. Install at your own risk. Developed by Tee-222 2004.”



### Hình 3-18 Trojan/Skulls-A

File cài đặt.sis bao gồm rất nhiều file con. Những file trong thư mục Libs là những file nén RAR chứa đựng những thông tin, thông điệp do Trojan tạo ra, chẳng hạn như thông điệp sau:

"What is T-VIRUS?

T-VIRUS is not a type of virus, instead it is a system file, specially designed & created for you.

T-VIRUS crashes the main system of your phone, i guess it is the right time for you to go to your service center, or buy a new phone.

Newer & higher version of T-VIRUS, coming soon.

If you have Cabir, feel free to send it to me, I'll appreciate it very much."

#### 3.2.3.19. *Skulls-B*

##### 3.2.3.19.1. Thông tin:

Tên: Trojan/Skulls-B

Loại Virus: trojan

Hiệu ứng lè (side effects): để lại những file bị không bị nhiễm trên máy.

Cách thức lan truyền: wed downloads.

##### 3.2.3.19.2. Mô tả chi tiết thông tin virus:

Trojan/Skulls-B là một loại thuộc họ trojan được viết dành riêng cho dòng máy chạy trên nền hệ điều hành Symbian Series 60. Trojan này thường được đính kèm theo những phần mềm shareware dùng cho các máy tính dùng hệ điều hành Symbian, thông thường là những file icons.sis(Extended Theme.sis installation file). Cách thức hoạt động của Trojan/Skulls-B cũng tương tự như Trojan/Skulls-A.

Trojan/Skulls-B thực hiện cài đặt **Symb/Cabir-B** vào máy.

### 3.3. Các giải pháp an toàn bảo mật khi sử dụng công nghệ mạng Bluetooth.

#### 3.3.1. Những mẹo an toàn cho thiết bị Bluetooth:

- Chỉ mở Bluetooth khi bạn cần thiết
- Giữ thiết bị ở chế độ “không phát hiện ra”(hidden)
- Sử dụng số PIN dài và khó đoán ra khi pairing thiết bị
- Loại bỏ tất cả những yêu cầu pairing không bảo đảm
- Khi nhận lời mời kết nối nên yêu cầu PIN code.
- Thỉnh thoảng nên kiểm tra danh sách các thiết bị đã paired để chắc chắn là không có thiết bị lạ nào trong danh sách này.
- Điện thoại của bạn nên thường xuyên cập nhật phiên bản mới nhất của chương trình.
- Nếu thiết bị đó dễ bị bluesnarfing hoặc bluebugging, họ có thể cài phần mềm để khắc phục nhược điểm này.
- Nên mã hóa khi thiết lập kết nối Bluetooth với máy tính của bạn.

#### 3.3.2. Phòng chống virus trên mobile phone?

Virus trên mobile phone vẫn còn khá mới mẻ. Do đó, các phần mềm phòng chống virus trên mobile phone chưa có nhiều và chưa phổ biến như phần mềm phòng chống virus trên computer. Hơn nữa, do đây là công nghệ mới nên hầu như các hãng sản xuất phần mềm phòng chống virus cũng chỉ mới cho phép người dùng sử dụng bản trial và các nhà sản xuất điện thoại hầu như không hỗ trợ cho khách hàng trong việc diệt virus.

Phần mềm phòng chống và diệt các components của virus trên mobile phone phổ biến hiện nay là F- Secure Mobile Anti-Virus (bản trial có tại link: [www.f-secure.com/products/fsmavs60](http://www.f-secure.com/products/fsmavs60))

Khi bị nhiễm Cabir thì cách đơn giản nhất là tải chương trình miễn phí CabirFix - diệt virus Cabir tại <http://www.jamanda.com/> do Hãng Jamada phát

triển. Tuy nhiên, CabirFix không "diệt tận gốc" mà chỉ xóa biểu tượng và vài file phát hiện được, bạn cần install một ứng dụng quản lý tập tin và xoá các tập tin sau:

```
:\system\apps\caribe\caribe.rsc  
c:\system\apps\caribe\caribe.app  
c:\system\apps\caribe\flo.mdl  
c:\system\recogs\flo.mdl  
c:\system\Symbiansecuredata\caribesecuritymanager\caribe.app  
c:\system\Symbiansecuredata\caribesecuritymanager\caribe.rsc
```

Nếu thật rành về hệ thống, bạn có thể tìm kiếm những file và thư mục lây nhiễm Cabir để xóa, tuy nhiên cách này tương đối nguy hiểm. Tốt nhất là đem máy đến các trung tâm dịch vụ để sửa lỗi và diệt virus này, hoặc cài lại hệ điều hành Symbian mới (có thể mất dữ liệu và danh bạ). Ứng dụng quản lý tập tin có thể được download free từ :

<http://my-Symbian.com/7650/applications/applications.php?tag=2&fldAuto=88>

Virus trên điện thoại di động vẫn còn khá mới mẻ ở Việt Nam nhưng trong tương lai gần, nó cũng sẽ trở nên phổ biến vì công nghệ Bluetooth có khá nhiều tiện ích hay. Tuy nhiên, nếu chưa có biện pháp hỗ trợ cho việc bảo vệ chiếc mobile phone của mình (phần mềm phòng chống virus), hầu hết các chuyên gia đều khuyên rằng: “Tốt nhất bạn nên tắt chế độ Bluetooth đi”.

## **Chương 4 CÁC ƯU NHƯỢC ĐIỂM VÀ TƯƠNG LAI CỦA BLUETOOTH.**

### **4.1. Ưu điểm**

- Truyền dữ liệu giữa các thiết bị không cần cáp trong khoảng cách trung bình (10m, có thể xa hơn với thiết bị đặc biệt).
- Sử dụng sóng radio ở băng tần không cần đăng ký 2.4GHz ISM (Industrial, Scientific, Medical).
- Có khả năng xuyên qua vật thể rắn và phi kim, không cần phải truyền thẳng (line-of-sight).
- Khả năng kết nối point-point, point-multipoint.
- Bluetooth sử dụng cùng một chuẩn giao thức nên mọi thiết bị Bluetooth đều có thể làm việc với nhau.
- Sử dụng ít năng lượng, thích hợp với các thiết bị di động có nguồn năng lượng hạn chế.
- Sử dụng “frequency hopping” giúp giảm độ tốn đà.
- Có khả năng hỗ trợ 3 kênh thoại và 1 kênh dữ liệu.
- Có khả năng bảo mật từ 8 → 128bit.
- Thiết bị nhỏ gọn, số lượng thiết bị hỗ trợ Bluetooth ngày càng nhiều và đa dạng.
- Giá thành thiết bị rẻ, truyền dữ liệu miễn phí.
- Thiết lập kết nối dễ dàng và nhanh chóng, không cần access point.
- Sử dụng được ở bất cứ nơi nào.
- Được đỡ đầu bởi 9 tập đoàn khổng lồ, và ngày càng có nhiều tổ chức tham gia vào=>Bluetooth ngày càng được phát triển hoàn thiện và mạnh mẽ hơn.

### **4.2. Khuyết điểm**

- Do sử dụng mô hình adhoc → không thể thiết lập các ứng dụng thời gian thực.

- Khoảng cách kết nối còn ngắn so với các công nghệ mạng không dây khác.
- Số thiết bị active, pack cùng lúc trong một piconect còn hạn chế.
- Tốc độ truyền của Bluetooth không cao.
- Bị nhiễu bởi một số thiết bị sử dụng sóng radio khác, các trang thiết bị khác.
- Bảo mật còn thấp.

#### **4.3. Tầm ứng dụng và tương lai của Bluetooth.**

Bluetooth là thành quả nghiên cứu của nhiều công ty và được phát triển bởi nhóm SIG (Bluetooth Special Interest Group), tổ chức chính phát triển Bluetooth. Hiện nay tổ chức này đã có khoảng 3000 công ty thành viên, trở thành tổ chức có số thành viên đông đảo thuộc nhiều lĩnh vực công nghệ: từ máy móc tự động đến thiết bị y tế, PC đến điện thoại di động, tất cả đều sử dụng kỹ thuật không dây tầm ngắn trong sản phẩm của họ. Tuy nhiên có những đẳng cấp khác nhau giữa các công ty trong tổ chức này, điều này phụ thuộc vào sự quan tâm cũng như trình độ phát triển Bluetooth tại công ty đó. Những mức độ này có thể là: nhà sáng chế (promoter), nhà cộng tác Gold hoặc Silver (associate), adopter.

Bluetooth SIG đã đưa ra mục tiêu cải tiến trong vòng ba năm tới: giảm năng lượng sử dụng, tăng cường sự an toàn, tăng khoảng cách kết nối, hỗ trợ đa kết nối và tăng độ rộng băng thông. Điều này không chỉ giúp các nhà sản xuất có thể hoạch định chiến lược cho sản phẩm của họ mà còn nâng cao vai trò của Bluetooth trên lĩnh vực wireless trên thế giới.

##### **4.3.1. Các phiên bản kỹ thuật của Bluetooth:**

###### **Bluetooth 1.0 and 1.0B**

Versions 1.0 và 1.0B có nhiều vấn đề và các nhà sản xuất khác nhau làm sản phẩm của họ không làm việc với nhau được. 1.0 and 1.0B cũng bắt buộc truyền BD\_ADDR trong quá trình handshaking, tình trạng nặc danh không

thực hiện được ở mức giao thức là sự thất bại chủ yếu của những dịch vụ định sử dụng trong môi trường Bluetooth.

### **Bluetooth 1.1**

Version 1.1 đã sửa nhiều lỗi trong 1.0B và hỗ trợ thêm kênh không mã hóa (non-encrypted channel).

### **Bluetooth 1.2**

Version này vẫn có thể tương hợp với 1.1. Những đặc điểm nâng cao hơn là :

- *AFH (Adaptive Frequency Hopping)*: chống nhiễu tốt hơn bằng cách nhảy tầng số.
- Tốc độ truyền cao (*Higher transmission speeds*).
- eSCO (*extended Synchronous Connections*): cải tiến chất lượng âm thanh của đường truyền audio bằng cách truyền lại những gói hồng.
- RSSI (*Received Signal Strength Indicator*).
- HCI (*Host Controller Interface*) hỗ trợ cho 3-wire *UART*.
- HCI truy cập thông tin thời gian cho các ứng dụng Bluetooth.
- Nền tảng dây tầng cơ sở (baseband platform) lấy từ Ericsson Techonology Licensing.
- Khả năng Scatternet: thực hiện nhiều piconet cùng một lúc, giúp các piconet kết nối được với nhau.
- Hoàn thiện QoS.
- Kết nối nhanh hơn so với 1.1.

Ngày 8-11-2004, nhằm mục đích giúp các nhà sản xuất có kế hoạch cho sản phẩm tương lai, Bluetooth SIG đã tiết lộ kế hoạch ba năm, gồm một loạt những nâng cao trong đặc điểm kỹ thuật của Bluetooth nhằm nâng cao hiệu suất, sự an toàn, tiêu thụ năng lượng và tính tiện lợi. Điều này sẽ giúp Bluetooth giữ được vị trí trong lĩnh vực kết nối cá nhân.

### **2004 – Hiệu suất và năng lượng tiêu thụ (Performance and Power Consumption): Bluetooth version 2.0 + EDR**

Version này có thể tương hợp với 1.x. Nâng cao chính là nâng tốc độ truyền EDR (*Enhanced Data Rate*) lên 2.1 Mbit/s. Điều này cho phép :

- Tốc độ truyền tăng gấp 3 lần (có trường hợp gấp 10 lần).
- Sử dụng năng lượng ít hơn do chu trình thực thi giảm.
- Đơn giản hóa “kịch bản” multi-link tạo nhiều băng thông hơn.
- Cải thiện việc thực hiện BER (Bit Error Rate).

Kỹ thuật Bluetooth có thể có ích trong VOIP (Voice over IP). Khi VOIP phổ biến hơn thì các công ty không cần các đường dây điện thoại như hiện nay. Khi đó Bluetooth có thể dùng trong việc liên lạc giữa điện thoại bàn (cordless phone) và một máy tính nghe bằng VOIP và với một PCI card hồng ngoại là cơ sở của cordless phone. Cordless phone khi đó chỉ cần một cradle để nạp điện. Sử dụng Bluetooth như thế làm cho cordless phone vẫn còn được sử dụng trong thời gian dài. Tháng 5-2005 SIG đã loan báo họ sẽ làm việc với nhà sản xuất UWB để phát triển một kỹ thuật Bluetooth trong đó cho phép dùng kỹ thuật UWB và truyền với tốc độ UWB. Điều này cho phép kỹ thuật Bluetooth được dùng để thực hiện việc trao đổi dữ liệu tốc độ cao, cần thiết cho wireless VOIP, những ứng dụng music và video.

## **2005 – Chất lượng dịch vụ, bảo mật và tiêu thụ năng lượng (Quality of Service (QoS), Security and Power Consumption)**

Trong 2005, Bluetooth SIG sẽ kiểm tra và đưa ra một phiên bản kỹ thuật mới nhằm nâng cao tính tiện lợi trong trường hợp multi-device, hoàn thiện toàn diện vấn đề an toàn bảo mật và cải tiến “đột ngột” việc sử dụng năng lượng, thậm chí có thể làm cho thiết bị Bluetooth chỉ sử dụng một bộ pin trong nhiều năm. QoS cải thiện vấn đề địa chỉ thiết bị. Đây là một nhu cầu để nhiều thiết bị Bluetooth có thể kết nối và hoạt động cùng một lúc với nhau mà không bị trễ hay nhiễu. Với QoS, thiết bị có thể giao tiếp với nhau, thực hiện việc trao đổi dữ liệu một cách tốt đẹp.

Thêm vào đó việc cải thiện năng lượng sử dụng, việc tăng số lượng thiết bị tối đa trong một piconet từ 7 slave và 1 master lên 255 thiết bị bao gồm master sẽ làm kỹ thuật Bluetooth phù hợp hơn trong viễn cảnh tương lai như hệ thống an ninh gia đình và những ứng dụng tự động trong công nghiệp. Kỹ thuật ngày càng tạo thêm những tiện ích cho người sử dụng trong những hệ thống dùng thiết bị Bluetooth khác, ví dụ như một người có thể giải trừ hệ thống an ninh trong một căn nhà chỉ bằng cách nhấn vào một nút trên điện thoại di động, hoặc một người quản lý nhà máy có thể quản lý tiến trình sản xuất thông qua việc kết nối với một PC.

### **2006 – Multi-cast, Security and Performance:**

Bluetooth SIG dự định trong 2006 sẽ tiếp tục cải tiến những đặc điểm kỹ thuật để hoàn thiện tính tiện lợi, sự an toàn và hiệu suất.

Khả năng multi-cast sẽ cho phép cùng một mẫu tin được gửi đến nhiều thiết bị cùng một lúc, đồng thời cho phép cải tiến tính thích hợp cũng như năng lượng sử dụng trong các ứng dụng như các trò chơi có nhiều người chơi (multiplayer gaming), nhiều headphone và speaker âm thanh nổi.

Đặc biệt là đối với mối nguy hiểm từ xa thì những cải tiến cho sự riêng tư cũng sẽ làm cho thiết bị đang ở chế độ không thể nhận ra (non-discoverable) không bị định vị bởi tất cả kỹ thuật tiên tiến nhất, multi-year attack. Cải tiến hiệu suất làm tăng phạm vi hoạt động của thiết bị Bluetooth lên 100 m dù tiêu thụ năng lượng rất ít.

- Bluetooth Version 2.0 + EDR cũng đã được giới thiệu sau 6 tháng đầu. Những sản phẩm của EDR được trông đợi trong sáu hoặc chín tháng tới. Hai phiên bản kỹ thuật kế cũng được hy vọng hoàn thành công đoạn kiểm tra với cùng thời gian như thế và được sản xuất vào cuối mỗi năm.

#### 4.3.2. Những ứng dụng Bluetooth:



Hình 4-1 Những thiết bị ứng dụng Bluetooth

Bluetooth hiện nay chỉ mới được thử và xác nhận trong kỹ thuật mạng không dây tầm ngắn, dùng trong không gian của mạng cá nhân với những thiết bị như điện thoại di động, PC, PDA, headset, và hệ thống tự động (automotive hands-free system). Nhưng trong tương lai gần, với kế hoạch 3 năm của SIG, Bluetooth sẽ thâm nhập vào những lĩnh vực mới như kỹ thuật cảm biến, ứng dụng âm thanh, multi-player gaming,...

Những công ty ngoài lĩnh vực điện thoại cũng đang bắt đầu nghiên cứu và thiết kế những ứng dụng cho kỹ thuật không dây. Ngành công nghiệp máy tính là nhóm kinh doanh thứ hai thu lợi từ Bluetooth. Máy tính ngày nay có thể kết nối Internet thông qua mạng không dây Bluetooth.

Các ứng dụng Bluetooth ngày càng lan rộng khắp ngành công nghiệp máy tính và truyền thông, thị trường thiết bị di động cá nhân, và các ngành công nghiệp khác.

Các điện thoại di động kết hợp Bluetooth đã được bán với số lượng lớn và có thể kết nối với máy tính, PDA, các thiết bị cầm tay. BMW là hãng xe đầu tiên cài đặt kỹ thuật Bluetooth vào xe hơi của họ (Bluetooth car kit), ở các dòng xe 3 Series, 5 Series, 7 Series và X5. Đến đây là các hãng khác như Toyota Prius, Lexus, Lincoln. Bluetooth car kit cho phép người dùng với điện thoại di động có hỗ trợ chức năng Bluetooth thực hiện cuộc gọi mà không cần rời mắt khỏi đường chạy trong khi điện thoại không có trên người họ như ở vali chặng hạn.

Thị trường Bluetooth cho thiết bị thu phát cầm tay (handset) vẫn còn nhỏ nhưng đang ngày phát triển. Theo báo cáo của SIG, 6 triệu thiết bị Bluetooth handset bán trong 2004 nhưng tới 2009 dự đoán là 87.5 triệu thiết bị điện thoại di động thông minh (smartphone), hoặc 70% smartphone bán ra có chip Bluetooth.

Phiên bản mới nhất là Bluetooth Version 2.0 + EDR (Enhanced Data Rate), đáp ứng nhu cầu truyền âm thanh, hình ảnh kỹ thuật số và in laser. Broadcom, CSR và RF Micro Devices đều đã kiểm tra chuẩn EDR từ giữa 2004, core chip đã hoàn thành từ 11-2004 nhưng một số nhà sản xuất chỉ dự tính bắt đầu tung ra những sản phẩm đầu tiên hỗ trợ phiên bản này cho người tiêu dùng vào tháng 1-2005 và thực sự tung ra hàng loạt vào giữa năm 2005. Tất nhiên chúng đều tương hợp với những phiên bản trước đó.

### **Các ứng dụng Bluetooth:**

- Thiết lập mạng không dây giữa laptop và desktop, hoặc giữa những desktop ở những nơi không thể tạo mạng có dây.
- Nối các thiết bị Bluetooth ngoại vi như máy in, chuột và bàn phím.
- Truyền file (hình ảnh, nhạc, mp3...) giữa điện thoại di động, PDA và máy tính thông qua OBEX.
- Các máy nghe nhạc mp3 và máy chụp hình hay quay phim kỹ thuật số có tích hợp Bluetooth trao đổi file với máy tính.
- Car kits và Bluetooth headset cho điện thoại di động.

- Những ứng dụng cho y tế (Advanced Medical Electronics Corporation) trên một số dụng cụ.
- GPS receiver chuyển giao dữ liệu NMEA thông qua Bluetooth.

#### **Mobile Commerce**

Ngành thương mại về lưu động có một tiềm năng rất lớn trong các ứng dụng Bluetooth như máy bán hàng tự động, bãi đậu xe, nơi bảo quản sửa chữa xe ôtô, nơi bán thức ăn, nơi vui chơi giải trí đều có khả năng sử dụng Bluetooth. Có một số nơi ở châu Âu và Á đã dùng như thế.

#### **Automotive Industry**

Bluetooth và xe ôtô không thể kết nối trực tiếp với nhau tuy nhiên có thể dùng trong các ứng dụng cho ôtô như kết nối Internet, nhận lệnh bằng giọng nói, liên lạc với hệ thống bên ngoài...

Ngoài ra Bluetooth còn được sử dụng trong mạng không dây để download thông tin và giải trí hoặc những mục đích thực tế như kích hoạt cửa garage, đèn điện trong nhà, hệ thống sưởi ấm,...

#### **The Bluetooth shopping centre**

Tháng 12-2004 một trung tâm dịch vụ khách hàng dựa vào đặc điểm Bluetooth (bluepulse) đã được mở ra ở Sydney. Bluepulse là một ứng dụng đơn giản dành cho điện thoại di động, cho phép mọi người trong trung tâm mua sắm dùng điện thoại di động của họ lấy thông tin có ích ở xung quanh họ. Bluepulse có thể được truy cập thông qua mạng Bluetooth địa phương hay một mạng thông thường.

Broadway Shopping Centre trong trung tâm Sydney là trung tâm mua sắm dùng Bluetooth đầu tiên cho phép hệ thống bán lẻ của Broadway Centre giao tiếp với người tiêu dùng “bluepulse” khi họ bước vào trung tâm bằng cách cho họ truy cập những thông tin có giá trị liên quan và thực hiện lời mời chào thông qua điện thoại di động của họ. Australia sẽ là nơi thí nghiệm mô hình

bluepulse. Dự định cuối 2005 sẽ có khoảng 20 trung tâm mua sắm dạng này trên toàn Australia.

Bluepulse đưa đến những ích lợi như:

- Một dịch vụ thông minh đưa ra hướng dẫn trong thế giới thực: đưa ra lời hướng dẫn chi tiết đến từng met về vị trí của một cửa hàng đặc biệt, máy ATM, máy bán thuốc lá,...
- Gửi SMS và MMS miễn phí đến những thành viên khác trong trung tâm mua sắm.
- Một lịch biểu bao gồm thời gian chiếu phim, phim, đoạn phim quảng cáo và mua vé xem phim bằng điện thoại.
- Cung cấp bản đồ vị trí của những người bạn đang có mặt trong trung tâm.
- Cập nhật tức thì những quảng cáo hoặc những ưu đãi trong trung tâm.
- Xem danh sách mua sắm có trong máy tính ở nhà, chỉ việc nhập danh sách này vào tài khoản bluepulse online và có thể xem nó mọi lúc khi bạn đang đi shopping.

Bạn cũng có thể sử dụng bluepulse thông qua mạng truyền thông thường khi ở bên ngoài mạng Bluetooth của trung tâm mua sắm. Bluepulse giống như một bộ máy tìm kiếm lớn nhất của toàn thế giới trong túi của bạn, tìm kiếm thông tin ngay lập tức và gửi kết quả ngay đến màn hình điện thoại di động của bạn. Giám đốc điều hành bluepulse, Ben Keighran, tin tưởng rằng trong một tương lai không xa, ở bất kỳ nơi nào, mọi người có thể nhận thông tin có liên quan xung quanh họ trên màn hình điện thoại di động mà không hề phụ thuộc vào mạng mà họ đang kết nối cũng như loại thiết bị mà họ sử dụng. Ví dụ trong một sân thi đấu thể thao đó là một đoạn phim quay chậm, hoặc trong một festival, buổi hòa nhạc thì đó là mục lục chương trình, tìm bạn trong đám đông, ...

 **Bluetooth sẽ xâm nhập vào thị trường điều khiển từ xa, như điều khiển TV thay vì dùng hồng ngoại.**

Trong thị trường này, Bluetooth được xem là mạnh mẽ hơn và thuận lợi hơn rất nhiều do:

- Tia hồng ngoại, vốn thường được dùng trong điều khiển từ xa, phải định hướng nên gây rất nhiều khó khăn. Trong khi đó Bluetooth sử dụng sóng vô tuyến thì không cần định hướng, thậm chí có thể xuyên qua được một số vật cản nên tiện hơn rất nhiều.
- Bluetooth tạo liên lạc hai chiều nên có thể tạo ra những thiết bị điều khiển từ xa thông minh (intelligent remote control) như có một màn hình nhỏ trên thiết bị điều khiển.

Bluetooth SIG đang cố gắng nghiên cứu cải tiến, làm cho kỹ thuật ngày càng hoàn thiện hơn để các nhà sản xuất thành viên có thể tiếp tục đưa vào sản phẩm của mình. Và với kế hoạch ba năm như trên, trong tương lai có thể mọi thiết bị điện tử trên thế giới đều được “không dây hóa” và giao tiếp với nhau. Bluetooth sẽ được trang bị ở 3 môi trường: nhà, văn phòng và on-the-go (trang bị trên xe hơi, quần áo, mắt kính, bút,... ví dụ áo ski jacket trang bị Bluetooth bên trong và kết nối với Bluetooth trong mũ bảo hiểm mô tô).

## **Phần 2 HỆ ĐIỀU HÀNH SYMBIAN**

Để minh họa cho việc sử dụng công nghệ Bluetooth, chúng em xây dựng một ứng dụng nhỏ có sử dụng công nghệ Bluetooth. Chúng em thực hiện việc xây dựng ứng dụng trên điện thoại di động thuộc Series 60, sử dụng bộ công cụ phát triển Series 60 SDK v1.2 với môi trường Visual C++ 6.0.

Do ứng dụng được xây dựng trên điện thoại sử dụng hệ điều hành Symbian, cụ thể là series 60 ,vì vậy, trong phần này chúng em sẽ trình bày những vấn đề sau :

- ❖ **Chương 5. Tổng quan về hệ điều hành Symbian và thế hệ Series 60.**
- ❖ **Chương 6. Lập trình C++ trên Symbian.**
- ❖ **Chương 7. Bluetooth và Symbian : Lập trình sử dụng giao tiếp Bluetooth trên Symbian với C++.**

## **Chương 5 TỔNG QUAN VỀ HỆ ĐIỀU HÀNH SYMBIAN VÀ THẾ HỆ SERIES 60**

### **5.1. Khái niệm về hệ điều hành Symbian.**

- Symbian là hệ điều hành được sử dụng rộng rãi trên các thiết bị di động hiện nay. Bắt nguồn từ hệ điều hành EPOC (Electronic Pocket Communication) phát triển ban đầu bởi công ty Psion, ngày nay Symbian được hỗ trợ và phát triển bởi hàng loạt các công ty hàng đầu trong lĩnh vực truyền thông như : Sony Ericson, Nokia, Motorola (đã ra đời năm 2003 ), Psion, Panasonic, Siemens, Samsung...
- Symbian OS được thiết kế đặc biệt cho các thế hệ điện thoại di động 2G, 2.5G, 3G với nhu cầu về khả năng lưu trữ và chia sẻ dữ liệu.
- Đặc điểm của hệ điều hành Symbian :
  - Tích hợp hệ thống điện thoại di động đa chế độ (Intergated Multimode Mobile Telephony) : Symbian OS tích hợp sức mạnh của tính toán với hệ thống điện thoại di động, mang đến các tiện ích của các dịch vụ dữ liệu.
  - Môi trường ứng dụng mở (Open application environment): Hệ điều hành Symbian cho phép các điện thoại di động trở thành nền tảng (platform) cho sự phát triển của các ứng dụng và các dịch vụ ứng dụng, với nhiều loại ngôn ngữ phát triển khác nhau.
  - Các thành phần và các chuẩn mở (Open Standards and interoperability) : Được cài đặt mềm dẻo và từng phần (modular), Symbian OS cung cấp một tập nền các hàm API và các kỹ thuật được chia sẻ giữa tất cả các điện thoại dùng Symbian.
  - Đa nhiệm (Multi-tasking): Nhiều ứng dụng có thể chạy cùng một lúc, các services của hệ thống như telephony, networking

middleware, application engines chạy trên các tiến trình riêng biệt.

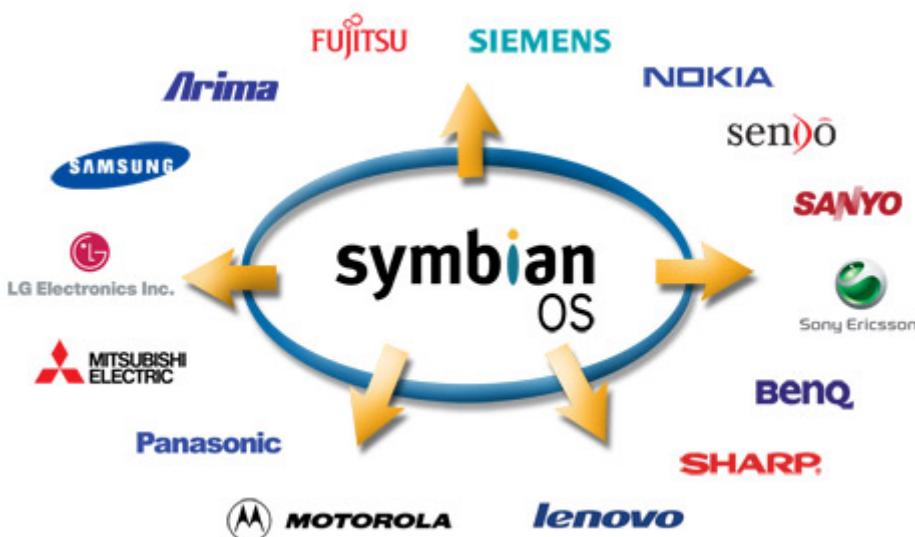
- Hướng đối tượng một cách đầy đủ ( Fully Object-Oriented and component base) : Hệ điều hành Symbian được thiết kế ngay từ đầu với mục đích hướng tới các thiết bị di động, sử dụng các tiến bộ của kỹ thuật hướng đối tượng hướng tới một kiến trúc thành phần phức tạp (flexible component based architecture ).
- Giao diện người dùng được thiết kế linh động (Flexible user interface design) : Cho phép các nhà sản xuất có thể tùy biến giao diện đồ họa của thiết bị. Việc phát triển ứng dụng sử dụng cùng một nền tảng hệ điều hành cho phép các ứng dụng của các nhà phát triển khác có thể dễ dàng được sử dụng trên các loại thiết bị của các nhà sản xuất khác nhau.
- Bảo mật : Cho phép trao đổi dữ liệu an toàn.
- Mạnh mẽ ( Robustness) : Symbian OS quản lý các truy cập dữ liệu của người dùng, đảm bảo sự toàn vẹn của dữ liệu, ngay cả khi có sự trao đổi thông tin không an toàn cũng như khi tài nguyên như bộ nhớ (memory), bộ phận lưu trữ (storage), hoặc năng lượng bị cạn kiệt.

## 5.2. Lịch sử phát triển.

- Nguồn gốc của hệ điều hành Symbian có từ buổi đầu của những thiết bị cầm tay. David Potter, một giảng viên vật lý cùng các cộng sự thành lập công ty Psion chuyên nghiên cứu và phát triển các thiết bị có dung lượng bộ nhớ thấp và hệ điều hành cho chúng.
- **Năm 1988** Psion công bố hệ điều hành SIBO (sixteen bit organizer), tiền thân của Symbian, đã đưa lại những thành công nhất định cho Psion. Máy tính đầu tiên sử dụng SIBO là MC laptop.
- **Năm 1991** Psion cho ra đời Series 3, màn hình được thiết kế vừa vặn cho một máy tính bỏ túi.

- **Năm 1996** đánh dấu sự ra đời của Series 3c với khả năng giao tiếp hồng ngoại
- **Năm 1998** với Series 3mx tốc độ xử lý được cải thiện, hơn hẳn các thế hệ trước đó và 3 tính năng nổi trội sau: Quản lý tốt nguồn năng lượng, hỗ trợ các hiệu ứng về ánh sáng cho các ứng dụng, vận hành dễ dàng trên nhiều loại máy tính khác nhau. Vào giữa cuối những năm 1990 Psion tung ra một hệ điều hành mới 32 bit với EPOC kernel release 1, hỗ trợ màn hình cảm ứng, sử dụng đa phương tiện, mở rộng nhiều khả năng giao tiếp. Hệ điều hành mới này được thiết kế theo hướng đối tượng, cho phép dễ mang chuyển cho những kiến trúc và thiết bị khác nhau. Nó bắt đầu cho nhiều tính năng cơ sở của hệ điều hành Symbian về sau.
- Do những tính năng vượt trội trên, **tháng 6 năm 1998** Psion và các nhà lãnh đạo của những công ty mobile lớn như Nokia, Ericsson, Motorola liên kết với nhau cùng nhau phát triển EPOC đánh dấu sự ra đời của hệ điều hành Symbian (Symbian OS).
- **Năm 1999** Matsushita (Panasonic) gia nhập vào hội những nhà phát triển Symbian OS. Cũng trong năm nay, Symbian được tạp chí Red\_Herring bầu chọn là công ty triển vọng nhất trong năm (Best long-term potential).
- **Năm 2000** chiếc điện thoại sử dụng Symbian đầu tiên ra đời và được tung ra thị trường, đó là Ericsson R380, vào năm đó Sony và Sanyo được cấp bản quyền Symbian OS.
- **Năm 2001** Fujitsu, Siemens được cấp bản quyền Symbian OS, và Symbian OS v6.1 cho điện thoại di động (2.5 G) được công bố, điện thoại 2.5 G đầu tiên sử dụng Symbian OS là Nokia 7650. Dòng điện thoại Nokia 9210 Communicator được tung ra thị trường.
- **Năm 2002** Symbian OS v7.0 được công bố tại 3GSM World Congress. Sendo được cấp bản quyền Symbian OS. Siemens trở thành cổ đông của Symbian, Sony Ericsson gia nhập Symbian trở thành cổ đông và được cấp phép bản quyền.

- **Năm 2003** Symbian OS v7.0 được công bố rộng rãi, Nokia 6600 là điện thoại đầu tiên sử dụng Symbian 7.0 và tiếp theo là hàng loạt những thành công khác với hệ điều hành mới này trên các thiết bị di động: Foma 2102v, Motorola A920, Nokia 7700, Sendo X, Siemens SX1, Sony Ericsson P900..... Cũng trong năm nay, Samsung trở thành cổ đông của Symbian.
- **Năm 2004 :** Hàng loạt các điện thoại sử dụng Symbian được tung ra thị trường : Panasonic X700, Motorola A1000, Nokia 6260, 6630, 9500, 7610 và N\_Gage QD, Samsung SGH D-710, Sony Ericsson P910, FOMA F900iT, F900iC, và F900 iES.
- Cũng trong Năm 2004, có sự gia nhập và được Symbian cấp phép của nhiều công ty sản xuất điện thoại di động : Lenovo, là tập đoàn IT lớn nhất của Trung Quốc, công ty Arima, LG Electronic, Sharp.
- Phiên bản Symbian OS v8.0 được công bố.
- **Năm 2005 :**
  - + Kết thúc quý 1 năm 2005, tổng số điện thoại di động sử dụng hệ điều hành Symbian được bán ra trên toàn thế giới đạt đến con số 32 triệu, trong đó, chỉ tính trong quý 1, số lượng bán ra cũng đã đạt đến 6.75 triệu chiếc, tăng 180% so với cùng kì năm 2004 ( Quý 1 năm 2004 đạt 2.4 triệu chiếc ).
  - + 2/2005 : phiên bản hệ điều hành Symbian OS v9. được công bố, dự tính nửa cuối năm 2005 sẽ có sản phẩm sử dụng hệ điều hành Symbian v9.
  - + 2/2005 Series 60 3<sup>rd</sup> Edition được công bố.
  - + Tính đến tháng 5/2005 : có 48 điện thoại smartphone sử dụng hệ điều hành Symbian có mặt trên thị trường, trong đó có 12 điện thoại thiết kế cho mạng điện thoại 3G. Có tổng số 14 nhà sản xuất được Symbian cấp phép, đó là : Arima, BenQ, Fujitsu, LG, Lenovo, Mitsubishi, Motorola, Nokia, Panasonic, Sendo, Sharp, Siemens, Samsung và Sony Ericsson.

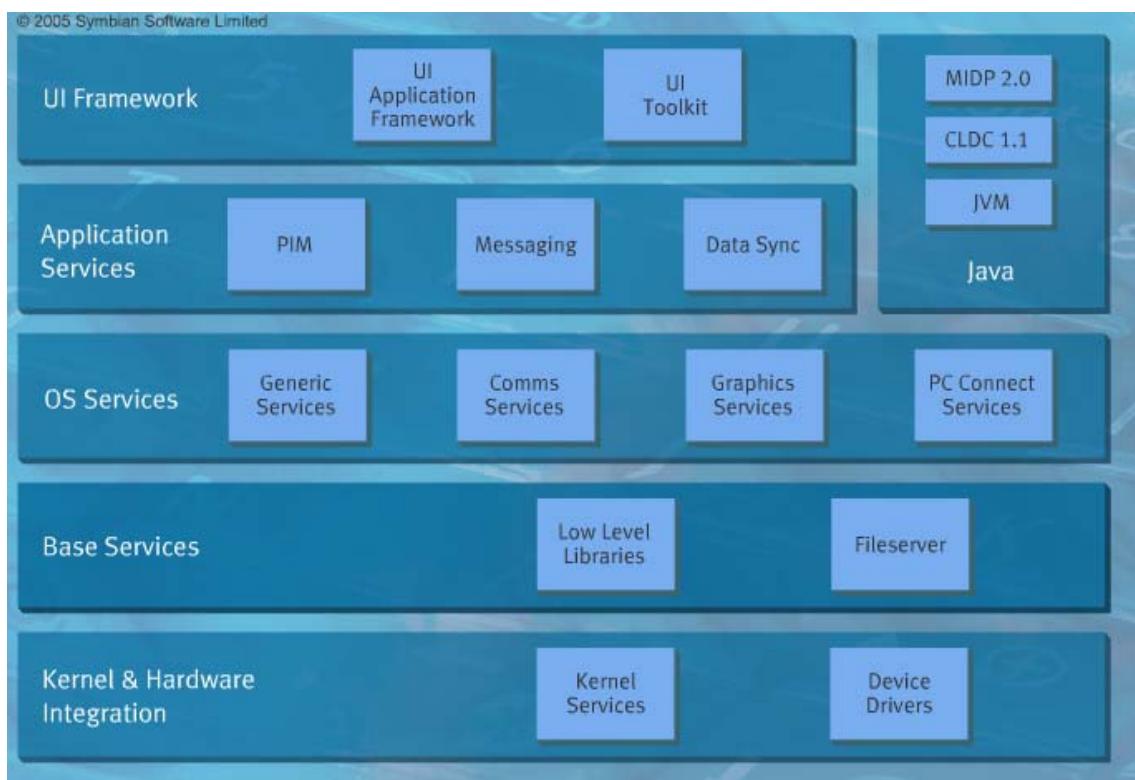


Hình 5-1 Các nhà sản xuất được Symbian cấp phép (5/2005)

### 5.3. Kiến trúc Tổng quan của hệ điều hành Symbian.

Về tổng quan, hệ điều hành Symbian được chia thành các thành phần sau:

- + Lõi của hệ điều hành, gọi là kernel
- + Một tập hợp các middleware cho các dịch vụ hệ thống.
- + Một tập hợp các quản lý tài nguyên, gọi là application engines.
- + Một khung chương trình để thiết kế giao diện người dùng gọi là : User Interface framework
- + Các phương pháp để đồng bộ hóa với các máy khác ( Synchronization technology).
- + Cài đặt máy ảo Java (Java Virtual Machine Implementation).



Hình 5-2 Kiến trúc hệ điều hành Symbian

### 5.3.1. Nhân hệ điều hành - Kernel

Là một chương trình chạy suốt từ khi mở đến khi tắt máy làm nhiệm vụ quản lý các dịch vụ (services) hỗ trợ người dùng và quản lý các bảng dữ liệu (data tables) của Symbian OS. Kernel của Symbian OS là một lõi điều hành hệ thống gồm: Tập hợp những drivers, bảng dữ liệu (data tables) và một số chương trình cho phép người dùng giao tiếp với phần cứng, do đó kernel cần được trỏ nén nhỏ và hiệu quả. Những chức năng hoạt động thường xuyên thì được đặt trong kernel, những chức năng khác được đặt trong middleware hay application engine. Thiết kế này làm cho kernel thêm chặt chẽ và module hóa kiến trúc và hoạt động của Symbian OS.

Ngay từ lúc hình thành Symbian OS đã được thiết kế hướng đối tượng với một cấu trúc nhỏ gọn và hiệu quả. Đây là một OS 32 bit hỗ trợ multitasking và multithreading. Cấu trúc module của nó hỗ trợ giao tiếp component, cho phép thêm các component để phù hợp với những thiết bị và những công nghệ khác.

### 5.3.2. Middleware

Middleware là tập hợp các thư viện, kho dữ liệu và chương trình thực hiện các chức năng hệ thống nhưng không cần thiết đặt trong kernel như: Quản lý dữ liệu, giao tiếp, đồ họa.

Bằng việc tạo một tầng mới (middleware), các nhà thiết kế Symbian OS dễ dàng thiết kế các dịch vụ hệ thống mới hoặc nâng cấp các dịch vụ cũ mà không cần phải viết lại kernel. Middleware tồn tại để hỗ trợ nhiều dịch vụ hệ thống như: Hệ thống cửa sổ, giao tiếp mạng, cổng serial và hồng ngoại, Bluetooth, quản lý đa phương tiện và cơ sở dữ liệu.

### 5.3.3. Application Engine

Application Engine cung cấp khả năng truy cập đến những tài nguyên không quan trọng của hệ thống. Các application engine quản lý những dữ liệu và dịch vụ không liên quan đến hệ thống. Các ứng dụng mức người dùng tương tác với application engine và application engine sẽ tương tác với middleware.

### 5.3.4. User Interface framework

Từ khi Symbian OS được sử dụng nhiều trên những thiết bị cầm tay thì vấn đề thiết kế giao diện người dùng là cực kì quan trọng. Giao diện phải được thiết kế sao cho dễ sử dụng, dễ thay đổi và dễ lập trình. Hơn thế nữa do nhiều thiết bị được thiết kế khác nhau nên giao diện phải phù hợp với từng thiết bị. Vì những ràng buộc trên các nhà thiết kế Symbian OS đã xây dựng một framework giao diện người dùng như là một phần lõi của hệ điều hành.

Hiện tại có 2 loại Framework giao diện người dùng đó là Uikon và Standard Eikon. Uikon là một framework lớn gồm tất cả các thiết kế dùng để tham khảo. Standard Eikon chứa những module thường được sử dụng.

### 5.3.5. Kỹ thuật đồng bộ - Synchronization technology

Đây là kỹ thuật cho phép đồng bộ hóa dữ liệu với những máy tính khác. Kỹ thuật này gồm 3 phần:

- Quản lý kết nối: Một tiến trình do người dùng khởi tạo chạy trên Symbian, phát hiện và đồng bộ hoá kết nối khi có một thiết bị khác yêu cầu kết nối.
- Thực hiện dịch vụ khi kết nối: Gồm các dịch vụ khác nhau như duyệt tập tin, sao lưu, phục hồi.
- Chuyển định dạng tập tin: Chuyển đổi các định dạng tập tin khác nhau giữa các ứng dụng như rich text sang html

#### 5.3.6. Java virtual machine implementation

Symbian OS hỗ trợ Java đầy đủ với bộ J2ME (Java 2 Micro Edition) framework.

#### 5.4. Giới thiệu về thế hệ Series 60.

\* Trong phạm vi của luận văn, để xây dựng một ứng dụng minh họa cho việc sử dụng công nghệ Bluetooth, chúng em chọn xây dựng ứng dụng trên dòng điện thoại Symbian Series 60. Do đó, chúng em xin được giới thiệu sơ lược về platform này.

\* Series 60 platform là platform hệ điều hành hàng đầu cho các điện thoại smartphone hiện nay, được phát triển bởi Nokia, xây dựng dựa trên nền hệ điều hành Symbian. Hiện nay, Series 60 platform được sử dụng bởi các công ty sản xuất điện thoại di động hàng đầu trên thế giới bao gồm : LG Electronic, Nokia, Lenovo, Panasonic, Samsung, Sendo và Siemens.

\* Một vài đặc điểm của Series 60 platform :

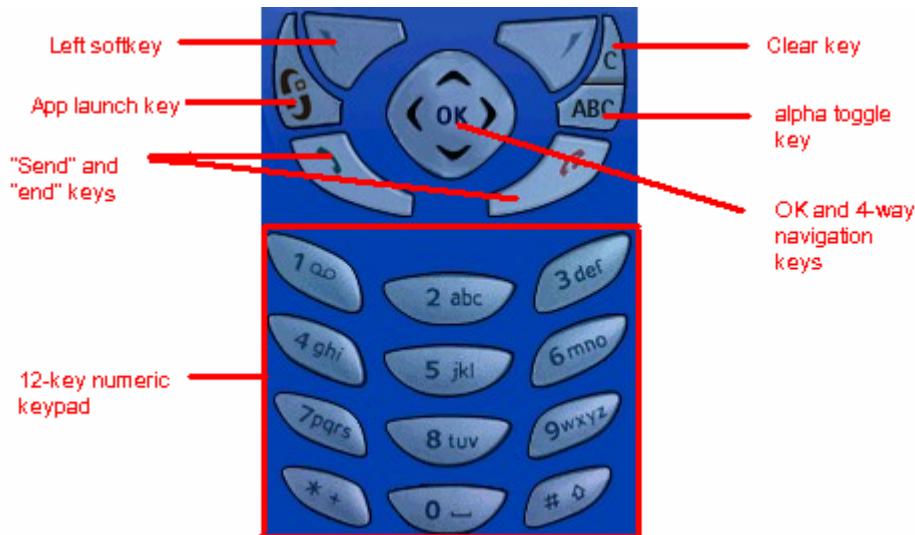
+ Màn hình của thiết bị với độ phân giải 176 x 208 ( với phiên bản Series 60 2nd Feature pack 3 hỗ trợ độ phân giải lớn hơn).

+ Hỗ trợ các ứng dụng được xây dựng bằng Symbian C++ và Java(J2ME/MIDP 1.0, 2.0).

+ Được thiết kế để có thể sử dụng dễ dàng và nhanh chóng.

+ Cung cấp một khung ứng dụng( Application framework) linh hoạt và mạnh mẽ.

+ Bàn phím gồm có 1 bàn phím 12 phím (12-key numeric keypad), 2 phím để nghe và kết thúc cuộc gọi (“send” and “end” keys), 2 softkey, 4 phím di chuyển (4-way navigation key), 1 phím OK, 1 phím để vào giao diện ứng dụng (app launch key), 1 phím xóa (clear key) và 1 phím để chuyển qua lại giữa các chế độ nhập liệu (alpha toggle key).



Hình 5-3 Bàn phím của Series 60.

\* Các phiên bản và sản phẩm của Series 60 platform :

+ Series 60 Version 0.9 : Dựa trên hệ điều hành Symbian 6.1.

Sản phẩm : Nokia 7650

+ Series 60 1st Edition : Dựa trên hệ điều hành Symbian 6.1

Các sản phẩm sử dụng platform này gồm có : Nokia 3620, 3650, 3660, N-Gage, N-Gage QD, Samsung SGH D700, Sendo X, Sendo X2, Siemens SX1....

+ Series 60 2nd Edition (Version 2.0) : Dựa trên hệ điều hành Symbian 7.0 : có thêm các tính năng mới như hỗ trợ J2ME/MIDP 2.0 và theme.

Các sản phẩm dựa trên platform này có : Nokia 6600, Panasonic X700, Samsung SGH-D710...

+ Series 60 2nd Edition (Version 2.1): Dựa trên hệ điều hành Symbian 7.0.

Các sản phẩm dựa trên platform này gồm có: Nokia 6620, 7610, 6260, 6670, 3230...

+ **Series 60 2nd Edition (Version 2.6):** Dựa trên hệ điều hành Symbian 8.0a.

Các sản phẩm : Nokia 6630, 6638, 6680, 6681, 6682...

+ **Series 60 2nd Edition (Version 2.8):** Dựa trên hệ điều hành Symbian 8.1a.

Các sản phẩm : Nokia N70, Nokia N90

+ **Series 60 3rd Edition (Version 3.0):** Dựa trên hệ điều hành Symbian 9.1

Các sản phẩm : Nokia N91.

## 5.5. Lập trình ứng dụng cho Symbian.

### 5.5.1. Các ngôn ngữ lập trình.

Ứng dụng trên Symbian có thể được viết bằng một trong các ngôn ngữ lập trình sau :

- C++: Symbian được viết bằng C++, nên đây được xem là ngôn ngữ lập trình chính, thư viện hỗ trợ nhiều nhất, có thể lập trình các server hay điều khiển thiết bị.

- Java: PersonalJava và JavaPhone được hỗ trợ trên Symbian 6.0, 6.1 nhưng không còn được hỗ trợ trên Symbian 7.0. Trên Symbian 7.0 sử dụng J2ME, cụ thể là MIDP (Mobile Information Device Profile), cung cấp các Java API cho lập trình Java, nó chạy trên CLDC (Connected Limited Device Configuration) và sử dụng KVM (Kilobyte Virtual Machine), một máy ảo Java cho các thiết bị nhỏ. Phiên bản hiện tại là MIDP 2.0.

- Asembler: thường được sử dụng để xây dựng các chương trình cấp rất thấp chặng hạn bộ điều phối active scheduler, thường không hỗ trợ cho lập trình viên tự do.

- C: không còn được sử dụng để viết chương trình trên Symbian nhưng Symbian vẫn hỗ trợ để chuyển đổi các ứng dụng trước kia viết bằng C để có thể chạy trên hệ điều hành Symbian.
- OPL: là ngôn ngữ tựa Basic, hiện nay trên Symbian 7.0 không còn hỗ trợ nữa. Muốn sử dụng các ứng dụng viết bằng OPL trên Symbian 7.0 và các phiên bản về sau phải sử dụng một chương trình nền gọi là Booster.
- Các ngôn ngữ hỗ trợ lập trình Web hay theo các giao thức không dây như JavaScript hay WMLScript.

Đối với các nhà phát triển ứng dụng, chủ yếu họ sử dụng C++ hoặc Java để xây dựng ứng dụng trên Symbian, và thường thì các điện thoại cũng chỉ hỗ trợ cho họ hai loại ngôn ngữ này.

Trong luận văn này, chúng em sử dụng ngôn ngữ lập trình là C++ khi xây dựng ứng dụng, do đó, chúng em chỉ xin giới thiệu sơ lược về lập trình C++ trên Symbian, cụ thể là với nền hệ thống Series 60.

### **5.5.2. Các bộ công cụ phát triển ứng dụng – SDK (Software Development Kit) và các môi trường phát triển tích hợp – IDE (Integrated Development Environment) cho lập trình C++.**

#### **\* SDK :**

Nokia cung cấp các bộ công cụ phát triển phần mềm SDK dành cho dòng điện thoại series 60 sau:

- Với Series 60 phiên bản 1.0 dựa trên Symbian 6.1 có các bộ SDK sau :
  - \* Series 60 SDK 1.2 hỗ trợ IDE Borland C++ BuilderX hay Microsoft Visual C++ (v6.0 or .NET).
  - \* Series 60 SDK 1.2 hỗ trợ IDE MetroWerks CodeWarrior cho hệ điều hành Symbian.
- Với Series 60 phiên bản 2.0 dựa trên hệ điều hành Symbian 7.0 có các bộ SDK sau :
  - \* Series 60 SDK 2.1 hỗ trợ IDE Borland C++ BuilderX hay Microsoft Visual C++ (v6.0 or .NET).

\* Series 60 SDK 2.1 hỗ trợ IDE MetroWerks CodeWarrior cho hệ điều hành Symbian.

Các bộ SDK trên được cung cấp miễn phí tại trang web của Nokia :  
[www.forum.nokia.com](http://www.forum.nokia.com)

Chú ý: + Các bộ SDK chỉ hỗ trợ Windows NT4, Windows 2000 hoặc các phiên bản Windows mới hơn.

+ Các công cụ biên dịch thường phải chạy trên nền Perl nên trước khi cài đặt các bộ Symbian SDK, phải cài đặt các bản Active Perl. Đôi khi các phiên bản còn yêu cầu phải cài đặt môi trường thực thi Java (Java runtime environment).

+ Các lập trình viên thường phát triển ứng dụng cho nhiều nền hệ thống phần mềm nên đôi khi họ cài đặt nhiều bộ SDK khác nhau cùng lúc. Lúc này, các lập trình viên phải chú ý đến một biến môi trường có tên là EPOCROOT, được dùng để xác định bộ công cụ SDK hiện thời đang hoạt động. Để chuyển đổi hoạt động giữa các bộ SDK, ta có thể đặt lại giá trị cho biến EPOCROOT bằng cách sử dụng công cụ EpochSwitch được cung cấp khi cài đặt bộ SDK, hoặc bằng câu lệnh : >devices -setdefault <nền hệ thống>.

\* IDE : Để tiện cho việc lập trình, biên dịch và kiểm lỗi, ngoài các bộ SDK được Nokia cung cấp miễn phí, chúng ta cũng cần phải có một trong các môi trường phát triển tích hợp-IDE sau :

- Microsoft Visual C++ 6.0
- Microsoft Visual C++ .NET
- Borland C++ Builder 6.0 Nokia Edition
- Borland C++BuilderX
- Metrowerks CodeWarrior.

Việc sử dụng IDE nào là tùy thuộc vào sự hỗ trợ của bộ SDK sử dụng và sự quen thuộc của các lập trình viên.

## **Chương 6 LẬP TRÌNH C++ TRÊN SYMBIAN.**

Symbian được xây dựng bằng C++, vì vậy việc xây dựng ứng dụng trên Symbian bằng C++ cũng rất quen thuộc với các lập trình C truyền thống. Tuy nhiên, C++ trên Symbian có những điểm khác so với C++ trên môi trường PC.

### **6.1. Các kiểu dữ liệu cơ bản.**

- TAny : tương tự kiểu dữ liệu void trong C chuẩn
- Tbool : Tương tự như kiểu dữ liệu BOOL trong C chuẩn, nhận 1 trong hai giá trị : ETrue, và EFalse.
- TChar : là một số nguyên không dấu 32 bit mô tả một kí tự.
- TInt8, TUint8 : Số nguyên 8 bit có dấu và không dấu
- TInt16 : số nguyên có dấu 16 bit.
- TUint16 : số nguyên không dấu 16 bit.
- TInt32 , TUint32 : số nguyên có dấu và không dấu 32 bit.
- TInt64 : là số nguyên 64 bit được tạo ra bởi 2 số nguyên không dấu 32 bit.
- TReal32 : số dấu chấm động 32 bit, tương tự kiểu float trong C chuẩn.
- TReal, TReal64 : số dấu chấm động 64 bit, tương tự kiểu double trong C chuẩn.
- TText8 : mô tả một kí tự 8 bit, tương tự unsigned char.
- TText16 : mô tả một kí tự 16 bit, tương tự unsigned short int.
- TText : Tùy thuộc vào tình huống sử dụng :
  - \* Nếu là non\_unicode TText là TText8.
  - \* Nếu là unicode TText là TText16.

Đối với các số nguyên, kiểu dữ liệu TInt được dùng thông dụng nhất. Đối với các kiểu dữ liệu số chấm động, ta nên hạn chế dùng vì tốc độ tính toán trên số chấm động chậm hơn các kiểu dữ liệu khác. Đối với số chấm động, kiểu dữ liệu TReal được dùng thông dụng nhất.

## 6.2. Kiểu dữ liệu chuỗi và descriptor trên Symbian.

Trên Symbian chuỗi được biết và cài đặt dưới các descriptor thay vì string như trên C/C++ chuẩn hay Java. Chúng được cài đặt qua các lớp khác nhau mang lại sự tiện lợi như trên C++ chuẩn hay Java. Nhưng khác với C++ chuẩn và Java vốn dùng trên PC, chuỗi trên Symbian đã được quản lý theo phong cách mới để phù hợp với bộ nhớ nhỏ trên điện thoại Symbian. Ngoài ra descriptor có thể dùng để lưu trữ các dữ liệu nhị phân.

Các loại descriptor : Trên Symbian, chuỗi được cài đặt trong các loại descriptor sau :

- + Abstract descriptor.
- + Pointer descriptor.
- + Buffer descriptor.
- + Heap descriptor.
- + Literal descriptor

▪ **Abstract descriptor:** Symbian cung cấp 2 lớp TDesC và TDes để biểu diễn chuỗi.

- TDesC là một descriptor hằng, không thể thay đổi nội dung được. Nó có một địa chỉ và một chiều dài. Với TDesC, chúng ta có thể thao tác chuỗi qua các hàm mà nó cung cấp nhưng không thể thay đổi được dữ liệu.

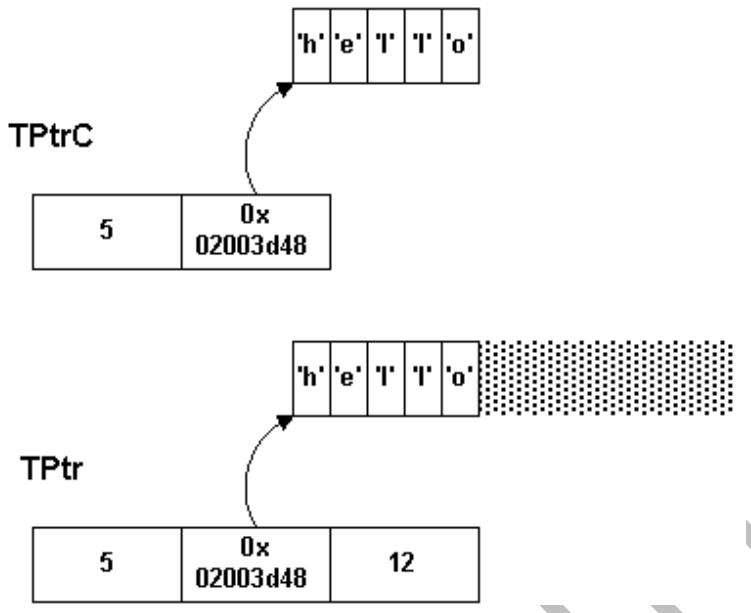
- TDes là một descriptor có thể sửa đổi dữ liệu được. Ngoài các thuộc tính kế thừa từ TDesC, nó có một độ dài tối đa cho phép dữ liệu được mở rộng, nói, cắt trong giới hạn của chiều dài tối đa. TDes cung cấp đầy đủ các hàm thao tác chuỗi, kể cả các hàm sửa đổi chuỗi mà trên TDesC không có.

Hai lớp này cung cấp nhiều hàm để thao tác với chuỗi, tất cả các descriptor còn lại đều kế thừa từ abstract descriptor qua 2 lớp này.

▪ **Pointer descriptor** : Đây là một loại descriptor mà dữ liệu của nó do một descriptor khác lưu giữ, có thể là trên heap, stack hay trên ROM. Gồm có hai loại như sau :

- **TPtrC** kế thừa từ TDesC, nó chỉ có chiều dài và địa chỉ, nên chỉ mất 2 từ nhớ 32 bit (8 byte).

- TPtr kế thừa từ TDes, nó được dùng để mô tả một vùng đệm (buffer) trên heap với một thuộc tính độ dài tối đa được thêm vào.



Hình 6-1 Mô hình đối tượng TPtrC và TPtr

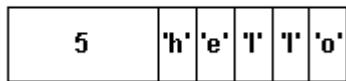
TPtrC và TPtr gần giống như `conts char*` và `char*` trên C, nhưng trong bản thân nó đã có chiều dài, không cần phải quét tìm ký tự kết thúc như trên C.

Sử dụng poiter descriptor :

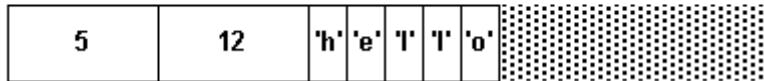
```
_LIT(KExample,"Example");
TPtrC examPtr(KExample);
TPtr examPtr(KExample);
```

- **Buffer descriptor** : Cũng gồm có 2 lớp là TBufC<n> và TBuf<n>, chứa dữ liệu ngay trong chúng, lưu trữ trong ngăn xếp stack, tương tự như `char[]` trong C, đối số n ở đây chính là khai báo chiều dài tối đa của chuỗi.

### TBufC<5>



### TBuf<12>



Hình 6-2 Mô hình đối tượng TBufC và TBuf

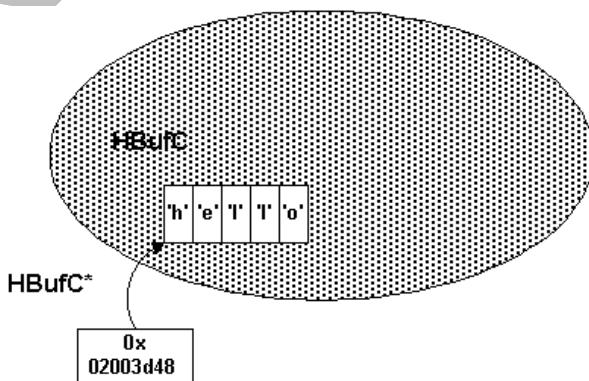
Các descriptor này sử dụng cơ chế mẫu (template) trong C++ với tham số là một số nguyên để mô tả chiều dài.

Sử dụng Buffer Descriptor :

```
_LIT(KExample,"Example");
TBufC <7> examStack(KExample);
hoặc
TBuf <7> examStack(KExample);
```

Lưu ý: do kích thước stack cấp cho một ứng dụng là rất nhỏ nên chuỗi cấp trên stack cũng chỉ nên cấp cho các chuỗi nhỏ, thường dưới 128 byte, nếu lớn hơn nên cấp trên heap.

- **Heap descriptor** : Lớp cài đặt : HBufC.



Hình 6-3 Mô hình đối tượng HBufC

HBufC chứa dữ liệu của chúng trong các ô trên heap. Điều này gần giống với (char\*) malloc (length+1) trong C. Giống như trên C, loại này được dùng khi chúng ta không biết độ dài là bao nhiêu. Các buffer descriptor lưu trữ trên heap, chúng thường được tham khảo thông qua con trỏ HBufC\* thay vì trực tiếp HBufC.

Sử dụng Heap descriptor :

```
_LIT(KExample,"Example");  
HBufC* examHeap = KExample().AllocLC();
```

Hay :

```
HBufC* iHelloBufC = HBufC::NewL(20);  
*iHelloBufC = KExample;
```

▪ **Literal descriptor** : Đây là một loại descriptor hằng, gần giống như static char[] trên C. Chúng thường được dùng dưới dạng các macro gồm 3 dạng : \_LIT, \_L, và \_S. **Literal descriptor** được dùng để tạo ra các hằng descriptor chứa các chuỗi.

- **Macro \_LIT** : Khai báo như sau:

```
_LIT(KHello,"Hello World");
```

Với khai báo như vậy, KHello được xem là tên 1 đối tượng TLITC16, nó sẽ chỉ đến một đoạn dữ liệu nhị phân của chương trình chứa nội dung cần lưu trữ, trong trường hợp này là chuỗi "Hello World". Lúc này thông qua tên đối tượng TLITC16 là KHello, ta có thể hoàn toàn sử dụng nó như là một descriptor hằng.

- **Macro \_L** : Khai báo như sau :

```
_L("Hello World");
```

Cũng giống như ở trên, với khai báo \_L, một vùng dữ liệu trên file chương trình được dùng để chứa chuỗi khai báo, nhưng khác với ở trên, chúng không có tên, không có gì để nắm giữ, điều khiển chúng. Trong trường hợp này, hệ thống sẽ tạo một pointer descriptor là TPtrC trên stack tạm thời đảm nhận việc kiểm soát và xử lý chuỗi dữ liệu này. Hiện nay, Symbian đã đề xuất

bỏ kiểu khai báo này do tốn thêm stack cho đối tượng tạm TPtr và chi phí khởi tạo nó. Tuy nhiên nhược điểm là giảm code lại khỏi phải đặt tên nên chúng vẫn thường được dùng với mục đích test. Ví dụ, một khi ứng dụng lỗi, để test xem có phải hàm này gây ra hay không, chúng ta có thể làm theo cách sau: ngay sau hàm nghi ngờ, chúng ta đặt hàm sau:

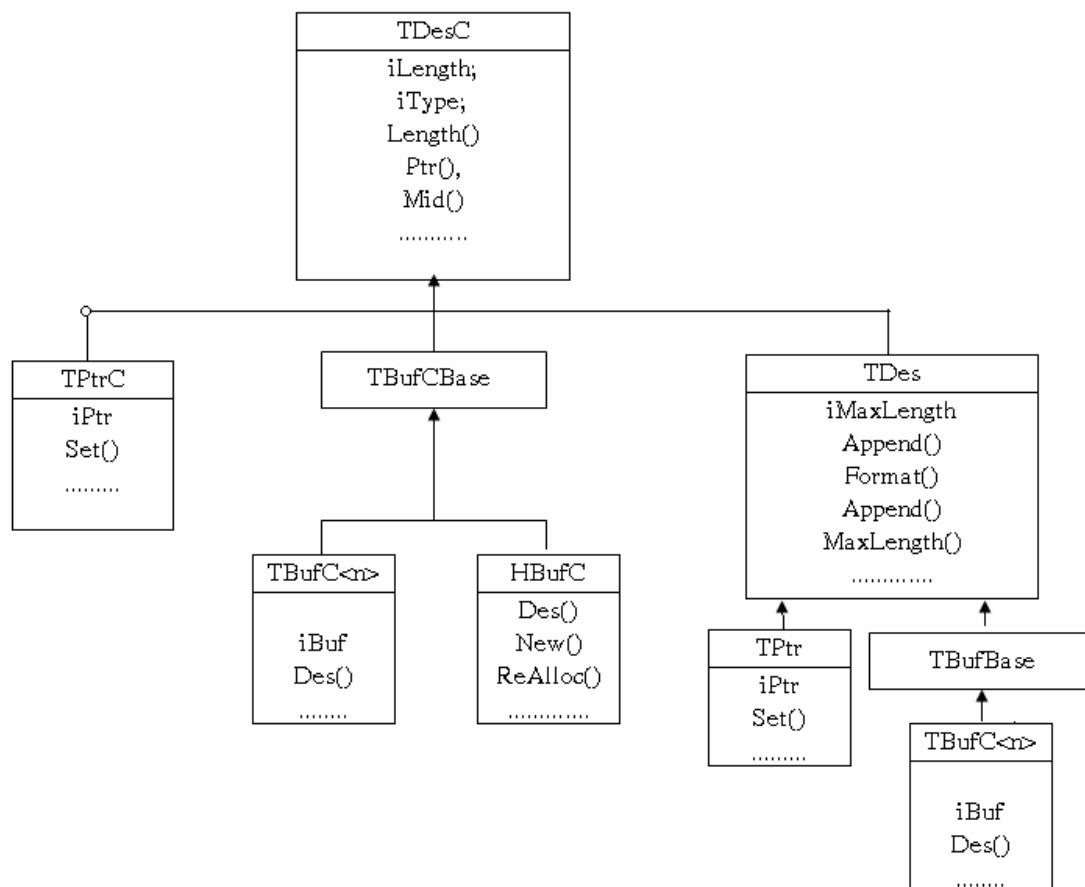
```
User::Infoprint(_L("Error here !"));
```

Thay vì phải viết :

```
_LIT(KMsg,"Error here!");
```

```
User::Infoprint(KMsg);
```

- **Macro \_S** : cũng gần giống như macro \_L, tuy nhiên nó không yêu cầu tạo đối tượng tạm TPtr mà sẽ cho phép sử dụng chuỗi trực tiếp như trên C.
- **Hệ thống các lớp descriptor:**



Hình 6-4 Sơ đồ hệ thống các descriptor

### 6.3. Các qui ước trong lập trình Symbian C++.

Symbian đưa ra một số qui ước trong lập trình C++ trên Symbian, các qui ước này giúp cho việc lập trình thuận lợi hơn, rõ ràng hơn, tránh sai sót và dễ nâng cấp sau này.

#### 6.3.1. Qui ước về đặt tên lớp.

Lớp trong Symbian dùng một kí tự đầu tiên để chỉ tính chất của lớp đó như sau:

- **Lớp T:** lớp đơn giản (tựa như typedef) thường sử dụng cho số integer, kiểu boolean và các cấu trúc đơn giản khác. Nó không có constructor và destructor và có thể được lưu trên stack. Ví dụ: TInt, TBool, TPoint,...
- **Lớp C:** Lớp có constructor và destructor và tất cả đều là dẫn xuất từ CBase. Các đối tượng được tạo bằng new và luôn được lưu trữ trên heap. Ví dụ: CConsoleBase, CActive,...
- **Lớp R:** Bất cứ lớp nào thao tác với các tài nguyên (file, time, v.v...) đều bắt đầu bằng chữ R (viết tắt của từ Resource).. Nó có thể được lưu trên heap. Ví dụ: RFile, RTimer, RWindow,... .Lớp thường sử dụng hàm close() để giải phóng tài nguyên mà nó quản lý.
- **Lớp M (Mix-ins):** Lớp ảo (abstract) giống như interface trong Java, nó chỉ bao gồm các phương thức ảo rỗng và không có dữ liệu. Ví dụ: MGraphicsDeviceMap, MGameViewCmdHandler,...

Việc phân biệt giữa các lớp T, C và R là rất quan trọng, nó ảnh hưởng tới việc cấp phát, quản lý và giải phóng bộ nhớ khi sử dụng.

#### 6.3.2. Qui ước đặt tên dữ liệu :

Tương tự như qui ước đặt tên lớp, Symbian cũng dùng chữ cái đầu để phân biệt các loại dữ liệu:

- **Hằng liệt kê (Enumerated constant):** Bắt đầu với ký tự E, nó đại diện cho một giá trị hằng trong một dãy liệt kê. Nó có thể là một phần của lớp T. Ví dụ: (ETrue,EFalse) hay EMonday là một thành phần của TdayOfWeek.

- Hằng (constant): Bắt đầu với ký tự K, thường được dùng trong các khai báo #define hay các giá trị hằng do Symbian quy định. Ví dụ: KMaxFileName hay KErrNone.
- Biến thành viên (member variable): Bắt đầu với chữ cái i (instance), được dùng khi sử dụng các biến động là thành viên của một lớp. Đây là quy ước quan trọng, dùng cho việc hủy vùng nhớ trên heap. Ví dụ: iDevice, iX, ...
- Tham số (argument): Bắt đầu bằng chữ a (argument), được dùng khi các biến làm tham số. Ví dụ: aDevice, aX, ...
- Macro: Không có quy ước đặc biệt. Tất cả đều viết hoa và dùng dấu gạch dưới để phân tách từ. Ví dụ: IMPORT\_C, \_TEST\_INVARIANT, \_ASSERT\_ALWAYS, v.v...
- Biến tự động (automatic): là các biến mà việc quản lý vùng nhớ do Symbian thực hiện tự động. Người dùng không cần cấp phát vùng nhớ khi khai báo và không hủy khi sử dụng xong. Ví dụ : device, size, x, y ....Chữ cái đầu của các biến này nên viết thường.
- Biến toàn cục (global): nên viết hoa chữ cái đầu nhưng để tránh nhầm lẫn nên bắt đầu tên bằng chữ cái “g”. Tuy nhiên trên Symbian không khuyến khích dùng biến toàn cục.

### 6.3.3. Qui ước đặt tên hàm:

Tên hàm thường bắt đầu bằng ký tự hoa, khác với 2 qui ước trên, trong tên hàm, kí tự cuối đóng vai trò quan trọng.

- Hàm “none-leaving”: Là các hàm kết thúc an toàn, không gây lỗi. Ví dụ: Draw() hay Intersects().
- Hàm Leaving : Kết thúc bằng ký tự L. Một hàm “leaving” là một hàm có cấp phát bộ nhớ, mở file, v.v..., nói chung là thực hiện những thao tác liên quan đến tài nguyên mà có thể không thành công do thiếu tài nguyên hoặc do những điều kiện về môi trường. Với loại hàm này, ngay cả khi gặp lỗi, chúng ta vẫn có thể quản lý được. Ví dụ: DrawL() hay RunL().

- Hàm LC: Kết thúc với cặp ký tự LC. Các hàm này khai báo một đối tượng mới, cấp phát vùng nhớ cho nó, đặt nó lên cleanup stack (ngăn xếp chứa các đối tượng cần xóa khi có ngắt xảy ra), sau khi gọi hàm này, lập trình viên cần phải gọi Cleanup:PopAndDestroy() để giải phóng vùng nhớ đã cấp phát cho đối tượng. Ví dụ: AllocLC(), CreateLC(), OpenLC() hay NewLC().
- Các hàm thiết lập : Các hàm này thường bắt đầu bằng chữ Set và dùng để gán giá trị cho một thuộc tính của đối tượng. Ví dụ : SetSize(), SetDevice(), SetTextL() .....
- Hàm lấy dữ liệu: dùng để truy xuất dữ liệu của đối tượng, ví dụ : Size(), Device(), GetTextL()...

## 6.4. Quản lý lỗi trên Symbian.

### 6.4.1. Cơ chế bắt lỗi trên Symbian.

Trên môi trường máy tính để bàn, Java và C++ quản lý các lỗi chương trình thông qua các exception với các cơ chế như throw hoặc try-catch. Nhưng lúc Symbian được thiết kế thì cơ chế exception chưa được giới thiệu trên C++ , hơn nữa sau này khi được giới thiệu thì nó cũng tỏ ra không thích hợp với môi trường hạn chế về xử lý cũng như tài nguyên như Symbian, bởi vì nó làm tăng đáng kể kích thước mã biên dịch và tốn nhiều RAM. Vì vậy, Symbian đưa ra một cơ chế quản lý lỗi cho riêng mình được biết tới tên gọi là “leave”.

Cơ chế hoạt động của “leave” như sau : khi xảy ra một lỗi nào đó (như thiếu bộ nhớ để cấp phát, thiếu vùng nhớ để ghi, lỗi trong truyền thông hay thiếu năng lượng cho các tài nguyên,...) thì hàm đang hoạt động sẽ bị ngắt lại, hệ điều hành tiến hành quét các chỉ thị, các hàm sẽ thực thi sau nó và quyền điều khiển sẽ được chuyển đến phần chỉ thị xử lý lỗi đầu tiên mà nó tìm thấy trong các hàm này.

Hành động ngắt hàm bị lỗi và chuyển đến phần xử lý lỗi được gọi là “leave” (thoát). Một “leave” là một cuộc gọi đến hàm User::Leave(), nó sẽ gây ra một ngoại lệ (exception) và lập tức tìm gọi phần xử lý lỗi gọi là “trap

harness”. Tất cả các hàm có thể “leave” thì đều có tên kết thúc bằng chữ cái L. “Leave” trong C++ trên Symbian giống như “throw” và “trap harness” giống như “try-catch” trên C++ và Java trong môi trường máy tính để bàn.

Symbian cung cấp 2 macro “trap harness” là TRAP và TRAPD. Bất cứ khi nào xảy ra lỗi dẫn đến leave tại hàm được đặt làm tham số của các macro “trap harness” thì ngay lập tức quyền điều khiển hoạt động chương trình sẽ được trả về cho các macro này. Các macro này sẽ trả về mã lỗi gắn với hàm gây lỗi. Hai macro TRAP và TRAPD là giống nhau, chỉ khác là ở TRAP biến chứa mã lỗi trả về phải khai báo trước.

```
//; Macro TRAPD
TRAPD(error,someFuncL());
if (error!=KErrNone)
{
//Xử lý lỗi
}
```

```
//; Macro TRAP
 TInt error;
TRAP(error,someFuncL());
if (error!=KErrNone)
{
//Xử lý lỗi
}
```

#### 6.4.2. Hàm Leave.

Theo qui ước trong đặt tên hàm của Symbian, hàm Leave là hàm được kết thúc bằng kí tự L, ví dụ như NewL(), RunL()... Việc biết một hàm có phải là một hàm Leave hay không rất quan trọng, bởi vì Symbian là một môi trường rất hạn hẹp về tài nguyên, vì vậy, lỗi thiếu tài nguyên hay xảy ra, nếu một hàm leave không được xử lý kĩ thì rất dễ gây ra một lỗi rất lớn, đó là làm “lủng” bộ nhớ (memory leak).

Một hàm có thể Leave nếu nó :

- + Thực hiện các tác vụ có liên quan đến tài nguyên.
- + Gọi hàm có thể leave mà không được gọi kèm với các trap harness như TRAP hay TRAPD để xử lý lỗi nếu xảy ra.
- + Có gọi một trong các hàm hệ thống đảm nhận leave như User::Leave() hay User::LeaveIfError(),...
- + Có dùng toán tử new(Leave).

### **6.5. Một số vấn đề về quản lý bộ nhớ trong lập trình Symbian C++ :**

Vấn đề về quản lý bộ nhớ trong lập trình Symbian được coi là một vấn đề vô cùng quan trọng, bởi vì bộ nhớ của các thiết bị di động rất giới hạn, thường chỉ có khoảng 4MB RAM hoặc không quá 16 MB RAM, là vùng nhớ của hệ điều hành và để nạp các chương trình ứng dụng. Mặt khác, do các thiết bị có thể rất ít khi khởi động lại, vì vậy sẽ rất dễ xảy ra tình trạng thiếu bộ nhớ nếu không được quản lý tốt.

Tình trạng thiếu bộ nhớ thường xảy ra do do các lập trình viên quên giải phóng vùng nhớ đã cấp phát trên heap cho một đối tượng. Trên môi trường máy tính để bàn, việc thiếu bộ nhớ rất ít xảy ra do bản thân hệ thống đã sở hữu ít nhất 64 MB RAM và bộ nhớ ảo swapping có thể lên đến 4GB, mặt khác, các vùng nhớ sẽ được giải phóng khi chương trình kết thúc hoặc khi khởi động lại máy tính, hoặc nếu với lập trình bằng Java, bộ dọn rác Garbage Collector sẽ đảm nhận việc dọn dẹp bộ nhớ khi chương trình kết thúc. Nhưng, trên Symbian, việc dọn dẹp, giải phóng phóng các vùng nhớ được phó mặc cho các lập trình viên.

Một trường hợp lỗi bộ nhớ khác mà cũng rất dễ mắc phải, đó là khi gọi hàm xảy ra “leave” mà không được xử lý, sẽ dẫn đến một lỗi nghiêm trọng : gây ra “lủng” bộ nhớ (memory leak). Symbian đưa ra một cơ chế để có thể quản lý loại lỗi này, đó là cơ chế Cleanup Stack.

#### **6.5.1. Cơ chế Cleanup Stack**

Khi một hàm Leave xảy ra “leave”, điều khiển sẽ được chuyển đến phần xử lý lỗi, lúc này vùng stack dành cho hàm có leave này sẽ được giải phóng,

các biến khai báo cục bộ trong hàm này sẽ bị xóa đi. Đối với các biến khai báo kiểu T trên stack thì không sao nhưng đối với các biến kiểu C khai báo trên heap hay các biến kiểu R thì đây là vấn đề nghiêm trọng. Bởi lẽ theo đúng quy trình thực thi, nếu không có gì xảy ra thì vào cuối hàm, chúng ta sẽ hủy vùng nhớ đối tượng trên heap qua toán tử delete hay gọi hàm Close() cho các biến kiểu R nhưng nếu giữa chừng hàm bị ngắt trước khi ta gọi các hàm hủy này thì rõ ràng các đối tượng này sẽ không được giải phóng hoàn toàn, tạo ra lỗ hổng trên bộ nhớ. Ví dụ :

```
void UnsafeFunctionL()
{
    CExClass* test = CExClass::NewL(); //Hàm có thẻ leave
    test->FunctionMayLeaveL();
    delete test;
}
```

Trong khi gọi hàm *UnsafeFunctionL()*, khi thực hiện đến hàm *FunctionMayLeaveL()* thì xảy ra leave, lúc này, hàm *UnsafeFunctionL()* sẽ bị ngắt ngay, stack bị xóa, biến *test* bị xóa nhưng vùng nhớ cấp phát cho nó trên heap qua lời gọi hàm *CExClass::NewL()* thì vẫn còn tồn tại mà không được ai quản lý, do được cấp phát mà không được giải phóng nó sẽ gây ra một lỗ hổng bộ nhớ (memory leak). Để giải quyết vấn đề đó, Symbian đưa ra một khái niệm mới : đó là **CleanupStack**.

**Cleanup stack** là một vùng nhớ được dành riêng dùng để lưu các nội dung của con trỏ trên stack (địa chỉ một vùng nhớ trên heap). Cleanup stack dùng để phục hồi địa chỉ của vùng nhớ heap trên stack khi stack có sự cố do một hàm xảy ra leave.

Ví dụ : Sửa lại hàm trên sử dụng cleanup stack:

```
void UnsafeFunctionL()
{
    CExClass* test = CExClass::NewL(); //Hàm có thẻ leave
```

```
CleanupStack::push(test);
test->FunctionMayLeaveL();
CleanupStack::Pop(test);
delete test;
}
```

Lúc này nếu có leave xảy ra, do địa chỉ vùng nhớ test đã được đưa lên cleanup stack với hàm CleanupStack::push(test), sau khi hàm FunctionMayLeaveL() xảy ra leave, địa chỉ vùng nhớ trên heap của biến test được khôi phục thông qua hàm CleanupStack::pop(test), và vùng nhớ được giải phóng bình thường. Lỗi “memory leak” được khắc phục.

### 6.5.2. Khởi tạo 2 pha (Two - phase constructor)

Cleanup stack giúp chúng ta giải quyết vấn đề giải phóng vùng nhớ khi leave xảy ra, nhưng có một tình huống mà cleanup stack không giải quyết được, đó là khi biến thành viên của lớp này là một đối tượng của lớp khác và biến này được khởi tạo trong constructor của lớp. Khi đó, có thể xảy ra trường hợp như sau : Các hàm cấp phát tài nguyên cho các biến thành viên vẫn được thực hiện, trong khi đó, việc cấp phát tài nguyên cho đối tượng cha không thể thực hiện được, lúc đó sẽ xảy ra lỗi mà chúng ta không thể kiểm soát được : vùng nhớ của các biến thành viên sẽ không được giải phóng và xảy ra tình trạng “lủng” bộ nhớ.

Để giải quyết tình trạng đó, Symbian đưa ra một luật, đó là : constructor không được phép leave. Để thực hiện việc cấp phát tài nguyên cho các biến thành viên và cho đối tượng cha được thực hiện thông qua việc khởi tạo 2 pha (Two phase constructor) như sau :

- Pha 1: Phần constructor đơn giản, không leave. Phần này sẽ được gọi liền ngay sau khi toán tử new được gọi.
- Pha 2: Một hàm khác sẽ đảm nhận việc hoàn tất khởi tạo các biến thành viên, trên Symbian thường đặt tên là ConstructL(), hàm này có thể leave.

Ví dụ về khởi tạo hai pha:

```
Class CClassB:public CBase
{
public:
    ~CClassB();
    CClassB();
    void    ConstructL();
    CClassX *iX;
};

CClassB::CClassB() // Khởi tạo pha 1, không leave
{
}

CClassB::~CClassB()
{
    delete iX;
}

void CClassB::CConstructL() // Khởi tạo pha 2 : có thể leave
{
    iX = new (ELeave) CClassX;
}
```

Lúc đó, khi sử dụng lớp CClassB sẽ thực hiện như sau :

```
CClassB *iY = new (ELeave) CClassB; //Khởi tạo pha 1
CleanupStack::PushL(iY);
iY->ConstructL(); // khởi tạo pha 2
// Sử dụng iY
.....
// Hủy đối tượng iY:
CleanupStack::Pop(iY);
```

### 6.5.3. Khởi tạo đối tượng với NewL() và NewLC()

Việc khởi tạo hai pha như trên khá an toàn, nhưng lại khá rắc rối và dễ quên do phải thực hiện khởi tạo đối tượng phải thực hiện hai pha với 4 hàm. Do đó, Symbian đưa ra thêm một khái niệm khác nhằm làm giảm sự rắc rối và dài dòng trong khởi tạo đối tượng : đó là bao bọc contructor trong hai hàm static **NewL()** và **NewLC()**.

- **NewL()**: Dùng để cấp phát vùng nhớ cho các thành viên trong một lớp.
- **NewLC()**: Dùng để cấp phát vùng nhớ cho các con trỏ không phải là thành viên

Ví dụ : Lớp CExam :

```
Class CExam:public CBase
{
    ...
    static CExam * NewL();
    static CExam* NewLC();
    .....
};

// Cài đặt của hai hàm NewL() và NewLC() :
CExam* CExam::NewLC()
{
    CExam* me = new (Eleave) CExam();//pha 1
    CleanupStack:: push (me);
    exam->ContructL();           //Pha 2
    return me;
}

CExam* CExam::NewL()
{
    CExam* me = CExam::NewLC();
    CleanupStack:: pop (me);
```

Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa

```
return me;  
}
```

Khi đó việc sử dụng lớp CExam được thực hiện như sau :

```
// Khởi tạo đối tượng  
CExam *iY = CExam::NewL();  
CleanupStack::PushL(iY)  
// .... sử dụng iY  
.....  
// Hủy đối tượng :  
CleanupStack::PopAndDestroy(iY)
```

Hoặc :

```
// Khởi tạo đối tượng  
CExam *iY = CExam::NewLC();  
// .... sử dụng iY  
.....  
// Hủy đối tượng :  
CleanupStack::PopAndDestroy(iY)
```

## **Chương 7 BLUETOOTH VÀ SYMBIAN: LẬP TRÌNH SỬ DỤNG GIAO TIẾP BLUETOOTH TRÊN SYMBIAN VỚI C++.**

### **7.1. Giới thiệu.**

#### **7.1.1. Các ứng dụng Bluetooth trên các thiết bị sử dụng hệ điều hành Symbian:**

Hệ điều hành Symbian (kể từ phiên bản Symbian 6.1 trở đi) cung cấp các phần mềm hỗ trợ Bluetooth. Các nhà phát triển ứng dụng có thể phát triển các ứng dụng có sử dụng Bluetooth thông qua các hàm APIs được cung cấp sẵn, ví dụ như trên các nền phát triển phần mềm (platform) như : Series 60 Developer platform 1.0, Series 60 Developer platform 2<sup>nd</sup> Edition, Series 80 Developer platform 2.0, Series 90 Developer platform 2.0... Các ứng dụng có thể là các ứng dụng đơn giản point-to-point như chat application, cho đến các ứng dụng phức tạp như các game nhiều người chơi...

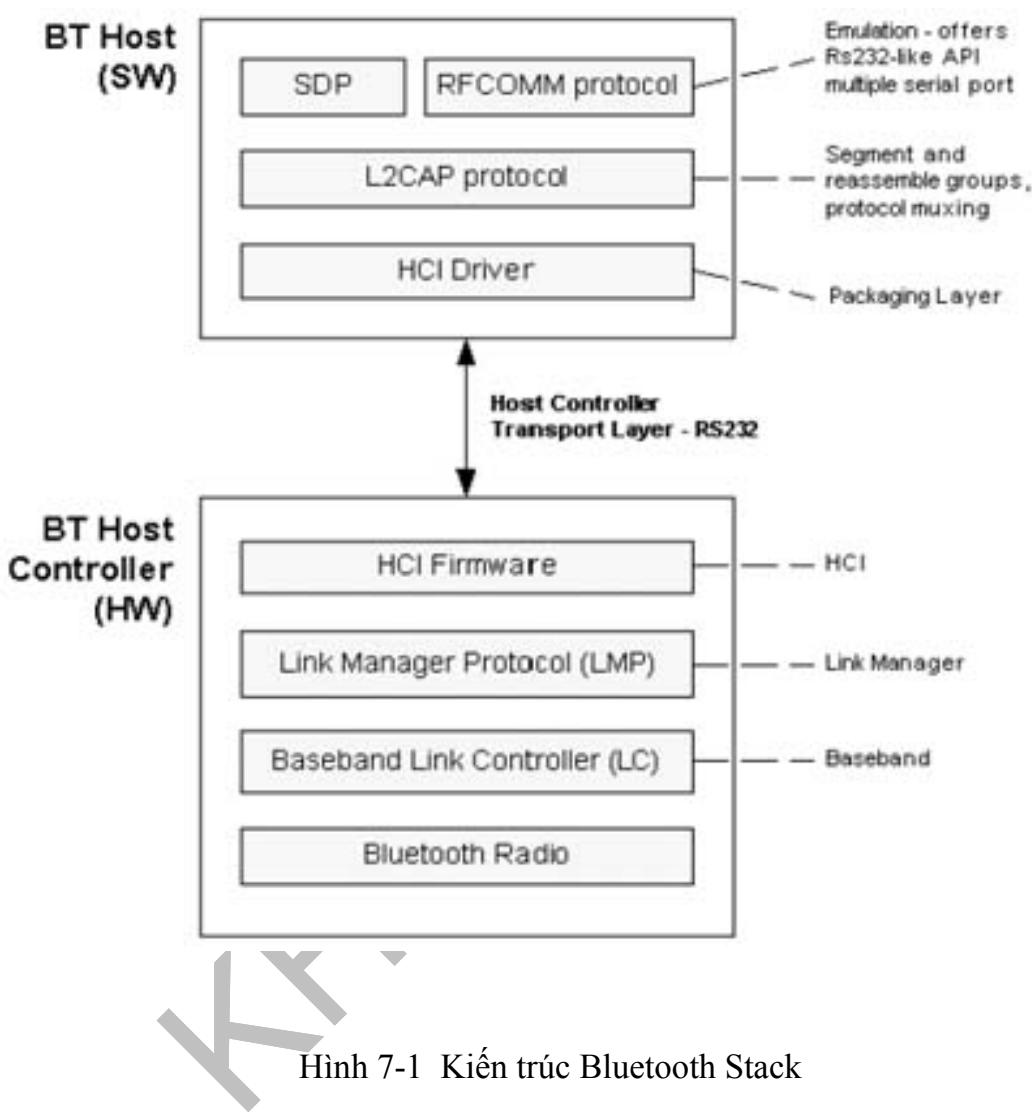
Các liên kết giữa một thiết bị và nhiều thiết bị khác cũng có thể cùng lúc xảy ra, điều này cho phép xây dựng các ứng dụng point-to-multipoint.

#### **7.1.2. Các công cụ phát triển và ví dụ:**

Để có thể xây dựng các ứng dụng Bluetooth cho Symbian bằng C++, cần phải download và cài đặt bộ SDK phù hợp với Developer Platform mà lập trình viên muốn phát triển ứng dụng. Các SDK có thể được download miễn phí từ trang web của Nokia. Bộ SDK có chứa tất cả các chi tiết cần thiết cho việc phát triển ứng dụng ( bao gồm các tài liệu, tham khảo API, các công cụ, các trình giả lập, trình phiên dịch, ví dụ ....), tuy nhiên, cũng cần một môi trường phát triển tích hợp (IDE ) để viết code.

## 7.2. Tổng quan về Bluetooth API:

Cũng giống như những công nghệ giao tiếp khác, Bluetooth bao gồm một tập hợp nhiều thành phần, được gọi là stack (chồng giao thức). Chồng giao thức được mô tả trong sơ đồ sau :



Hình 7-1 Kiến trúc Bluetooth Stack

- \_ Các thành phần của Bluetooth Host Controller thông thường được cài đặt trong phần cứng của thiết bị. Các ứng dụng không thể trực tiếp truy xuất tới tầng này.
- \_ Các thành phần của Bluetooth Host cho phép các ứng dụng có thể gửi và nhận dữ liệu thông qua các liên kết Bluetooth hoặc định cấu hình cho liên kết. Các thành phần trong Bluetooth Host được mô tả sau:

- RFCOMM : cho phép các ứng dụng coi các liên kết Bluetooth như là các liên kết trao đổi thông tin qua cổng Serial. Hay nói cách khác, giao thức này cho phép giả lập cổng Serial trong việc truyền nhận dữ liệu qua Bluetooth.
- L2CAP (Logical Link Control and Adaption Protocol): cho phép quản lý các liên kết, quản lý việc phân nhỏ các gói tin gửi đi và sắp xếp lại các gói tin nhận được.
- SDP (Service Discovery Protocol) : Được sử dụng để xác định, truy vấn các dịch vụ Bluetooth được cung cấp hoặc có trên thiết bị. Các ứng dụng thường sử dụng đến nó khi bắt đầu thiết lập liên kết Bluetooth với các thiết bị Bluetooth khác.
- HCI (Host Controller Interface) : cho phép các thành phần khác trong Bluetooth Host giao tiếp với phần cứng.
- Các hàm Bluetooth APIs của Symbian OS cho phép các ứng dụng sử dụng các dịch vụ của RFCOMM, L2CAP, SDP truy cập ở mức giới hạn dịch vụ của HCI.

### 7.2.1. Các nhóm hàm Bluetooth API:

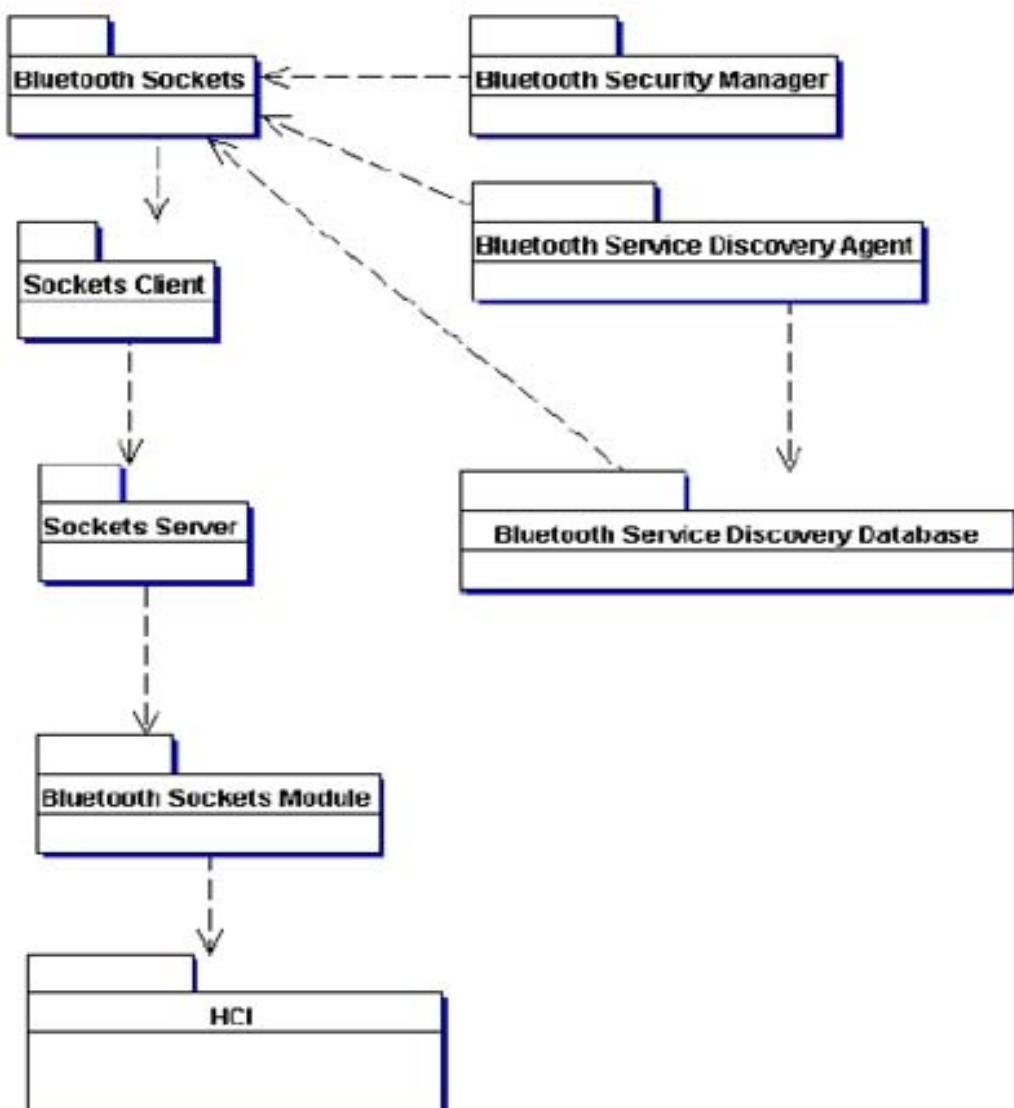
Các hàm Bluetooth API có thể chia thành những nhóm sau đây:

- **Bluetooth Sockets:** Cho phép truy cập đến các dịch vụ của L2CAP và RFCOMM thông qua giao diện socket của TCP/IP.
- **Bluetooth Service Discovery Database:** Sử dụng các dịch vụ của SDP. Các dịch vụ trên thiết bị sử dụng chúng để ghi nhận các thuộc tính của nó, nhờ đó, các thiết bị khác có thể tìm thấy nó và xác định xem chúng có dùng được không.
- **Bluetooth Service Discovery Agent:** Sử dụng các dịch vụ của SDP. Nó cho phép người dùng tìm những dịch vụ và thuộc tính của dịch vụ trên thiết bị Bluetooth khác.
- **Bluetooth Security Manager:** cho phép các dịch vụ thiết lập các yêu cầu về an toàn và bảo mật mà các kết nối tới cần phải đáp ứng.

- **Bluetooth Device Selection UI:** Cung cấp một hàm API dùng để thể hiện một dialog yêu cầu người dùng chọn thiết bị muốn giao tiếp.

### 7.2.2. Quan hệ giữa các nhóm hàm API:

Mỗi quan hệ giữa các nhóm hàm Bluetooth API được thể hiện trong hình bên dưới. Trong đó các hàm Bluetooth API của Socket đóng vai trò cơ bản nhất, các nhóm hàm API khác đều dựa vào nó để giao tiếp với các thiết bị khác:



Hình 7-2 Quan hệ giữa các nhóm hàm Bluetooth API.

### 7.3. Một vài kiểu dữ liệu Bluetooth thông dụng.

- **Bluetooth device address:** Mỗi thiết bị Bluetooth được xác định bằng một số nguyên 48bit mô tả địa chỉ thiết bị, trong Symbian, nó được mô tả trong lớp **TBTDevAddr**, định nghĩa trong file header “bbtypes.h”
- **Universally Unique Identifier (UUID) :** UUID là một số nguyên 128 bit được định nghĩa sẵn đại diện cho một dịch vụ Bluetooth . UUID được quản lý bởi lớp **TUUID**. Lớp này có thể quản lý dữ liệu UUID đầy đủ (128 bit) hoặc quản lý ở dạng rút gọn. Lớp TUUID được định nghĩa trong file header “bbtypes.h”
- **Service record.** Mỗi dịch vụ Bluetooth được đại diện bởi một UUID và được lưu trữ bởi một Bluetooth Service Record. Một service record chứa một UUID và một tập các thuộc tính cung cấp thông tin về dịch vụ mà nó lưu trữ. Symbian cung cấp các hàm Bluetooth SDP Database API cho phép thao tác với service record.
- **Service class and Profiles :** Service classes được diễn tả bởi các số nguyên UUID. Dùng để quản lý các dịch vụ được cung cấp bởi thiết bị.
- **Service profile handle :** Tất cả service record được lưu trữ trong SDP Database. SDP Database xác định các service record khác nhau bằng một con số 32 bit còn gọi là Service Record Handle. Một Service Record Handle được thể hiện bởi lớp **TSdpServRecordHandle**, được khai báo trong file header “btsdp.h”
- **Service Attribute ID :** Mỗi service record chứa một tập hợp các thuộc tính và mỗi thuộc tính được xác định bởi một con số định danh. Con số này được thể hiện bởi lớp **TSdpAttributeID**. Trong thực tế đây không phải là lớp mà là một số nguyên 16 bit. Kiểu dữ liệu này được định nghĩa trong file “btsdp.h”. Symbian dùng attribute ID để xác định các thuộc tính và giá trị của chúng trong SDP database.
- **SDP database:** Mỗi thiết bị Bluetooth có một cơ sở dữ liệu cục bộ chứa tất cả các dịch vụ mà nó cung cấp. Cơ sở dữ liệu này được thể

hiện bởi lớp **RSdpDatabase**. Lớp này được định nghĩa trong file header “btmdp.h”

- **Attribute ranges:** Sự khác nhau giữa **CSdpSearchPattern** và **CSdpAttrIdMatchList** là **CSdpAttrIdMatchList** chấp nhận một cấu trúc **TAttrRange** làm tham số. Cấu trúc này cho phép xác định một dải các attribute ID cần tìm kiếm.
- **Data elements :** Bảng sau mô tả các kiểu dữ liệu bluetooth data element, tất cả các kiểu dữ liệu này được khai báo trong file header “btmdp.h”:

Bluetooth Data Element Type	Symbian OS Representation
Nil	CSdpAttrValueNIL
Unsigned Integer	CSdpAttrValueUInt
Signed Twos-Complement Integer	CSdpAttrValueInt
UUID (Universally Unique Identifier)	CSdpAttrValueUUID
Text String	CSdpAttrValueString
Boolean	CSdpAttrValueBoolean
Data Element Sequence (list)	CSdpAttrValueDES
Data Element Alternative (list)	CSdpAttrValueDEA
URL	CSdpAttrValueURL

Hình 7-3 Bluetooth Data Element Types

- **Service search patterns:** Lớp **CSdpSearchPattern** cho phép xác định các dịch vụ cần tìm kiếm bằng cách tạo ra một danh sách các **TUUID** cần tìm. Lớp này cung cấp các hàm cho phép thao tác với danh sách các TUUID mà nó đang giữ như AddL, Remove, Find, v.v...
- **Attribute search patterns:** Cũng tương tự như service search pattern, Symbian cũng cung cấp các hàm cho phép xác định các thuộc tính cần tìm kiếm trong một service record. Lớp

**CSdpAttrIdMatchList** thực hiện điều này bằng cách tạo ra một danh sách attribute ID cần tìm kiếm.

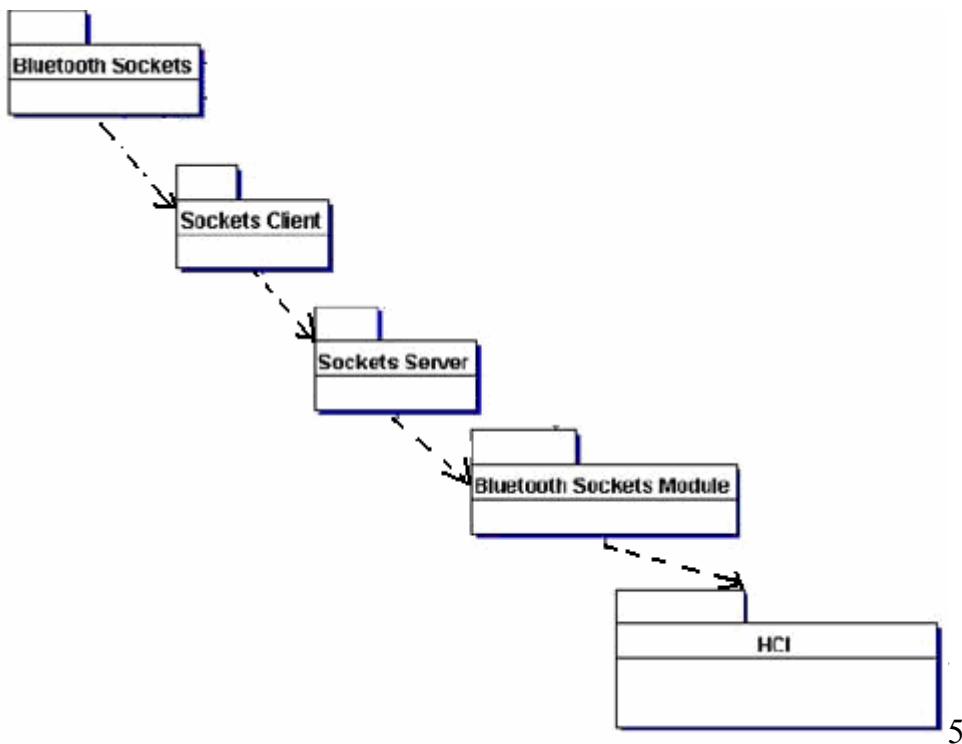
#### 7.4. Bluetooth Sockets.

Trong Symbian, Bluetooth Sockets được dùng để tìm kiếm các thiết bị Bluetooth khác và để truyền và nhận dữ liệu thông qua Bluetooth. Trong quá trình trao đổi thông tin qua Bluetooth, có hai vai trò được thiết lập : khởi tạo kết nối – Initiator , và chấp nhận kết nối – Receiver. Bên chấp nhận kết nối – Receiver khởi tạo Bluetooth và lắng nghe các yêu cầu kết nối từ Initiator. Một khi kết nối được thiết lập, cả hai đều có vai trò như nhau, đều có khả năng gửi, nhận dữ liệu, hoặc chấm dứt kết nối.

Các hàm API của Bluetooth Sockets hỗ trợ giao tiếp với cả hai tầng giao thức L2CAP và RFCOMM.Các hàm Bluetooth Sockets API dựa trên các Socket Client Side API, cung cấp các hàm API chuẩn cho phép một thiết bị hoạt động như Client có thể tạo kết nối với một thiết bị Bluetooth khác. Ngoài ra, nó cũng có thể hoạt động như là một Server, cho phép các thiết bị khác kết nối vào. Khi kết nối được thiết lập, thiết bị có thể thực hiện gửi và nhận dữ liệu trước khi kết thúc kết nối. Bluetooth Socket API được thêm vào các kiểu dữ liệu và các hằng số phù hợp cho phép các hàm Socket API có thể được sử dụng với Bluetooth.

Bluetooth Socket API có 5 khái niệm cơ bản : địa chỉ của socket, tìm kiếm thiết bị ( remote device inquiry), tập lệnh và các tùy chọn của RFCOMM, tập lệnh của L2CAP, và tập lệnh của HCI.

##### 7.4.1. Mở và cấu hình Bluetooth Socket :



Hình 7-4 Bluetooth Sockets

Mọi Bluetooth Socket đều phải thông qua Socket Server để có thể giao tiếp với các Bluetooth socket khác, vì vậy, trước khi mở một Socket, ta phải kết nối vào socket server của thiết bị. Lớp RSocketServ dùng để giao tiếp với socket server của thiết bị. Để kết nối vào socket server, ta khai báo một đối tượng kiểu RSocketServ và gọi hàm Connect:

```
RSocketServ socksrv;  
err = socksrv.Connect();  
User::LeaveIfError(err);
```

Symbian OS cung cấp lớp RSocket để mở và cấu hình các loại socket. Tùy theo Protocol sử dụng mà ta mở loại socket khác nhau. Để mở socket , ta sử dụng hàm Open() sau:

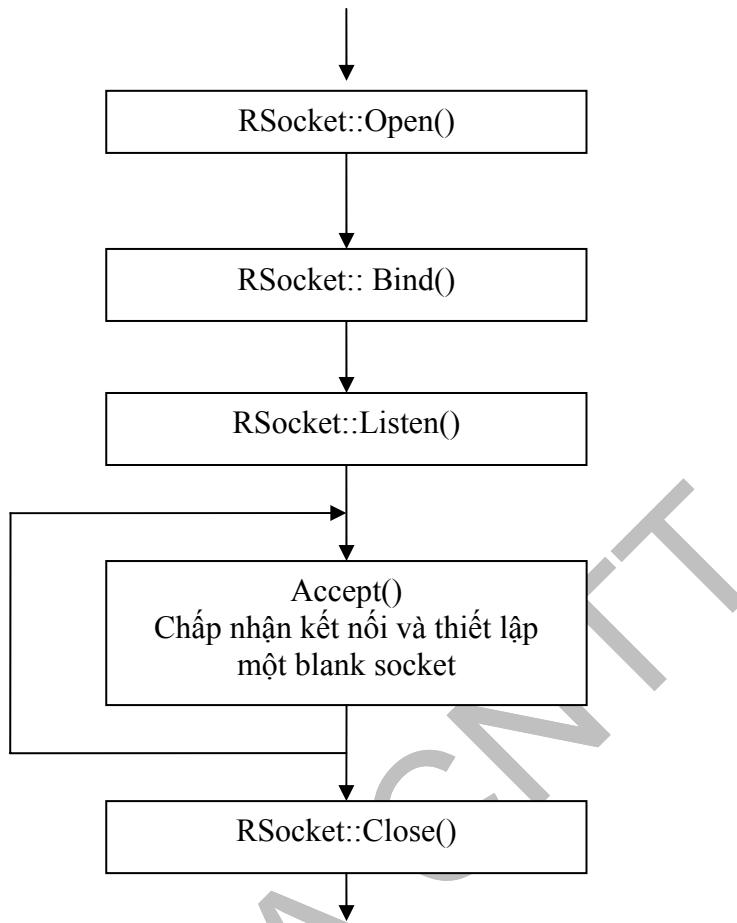
```
TInt Open(RSocketServ& aServer,  
          TUint addrFamily,  
          TUint sockType,  
          TUint protocol );
```

Trong đó :

- \* RSocketServ& aServer : Là Socket server kết nối tới.
- \* Tuint addrFamily : Kiểu địa chỉ của kết nối thực hiện, nếu là kết nối Bluetooth thì đó là KBTAddrFamily, hoặc nếu là TCP/IP thì đó là KafInet.
- \* Tuint sockType: Xác định loại socket sử dụng, ví dụ : nếu là TCP thì đó là KSockStream.
- \* Tuint protocol :Xác định loại protocol sử dụng
  - \* Đối với Bluetooth, nếu sử dụng giao thức RFCOMM thì sockType là KsockStream và protocol là KRCOMM; còn nếu sử dụng giao thức L2CAP thì sockType là KsockSeqPaket và protocol là KL2CAP.

#### **7.4.2. Xây dựng Bluetooth Socket Server : Lắng nghe và chấp nhận kết nối từ thiết bị là Client :**

Để có thể thiết lập một kết nối Bluetooth giữa hai thiết bị, một trong hai thiết bị phải được thiết lập, khởi tạo Socket Bluetooth ở trạng thái lắng nghe, và sau đó chấp nhận kết nối khi có yêu cầu kết nối tới từ thiết bị khác. Các bước để tạo Bluetooth Socket Server được thực hiện như sau :



Hình 7-5 Các bước khởi tạo Bluetooth Socket Server

Kết nối vào Socket server trên thiết bị và chọn protocol cần sử dụng.

Mở một socket tương ứng với protocol sử dụng.

Tạo ra một đối tượng địa chỉ Bluetooth Socket thuộc lớp TBTSockAddr và thiết lập cổng (port) của nó là server channel (đối với giao thức L2CAP và RFCOMM có sự khác biệt), và sau đó kết buộc socket vào địa chỉ đó bằng hàm RSocket::Bind().

Đưa socket vừa tạo ra vào trạng thái lắng nghe các kết nối tới bằng hàm: RSocket::Listen();

Tạo ra một socket trống (blank socket) và truyền tới cho socket listen thông qua hàm RSocket::Accept(RSocket &acceptSocket,...). Khi hàm này kết thúc, socket được truyền trong tham số của hàm sẽ kết nối với thiết bị khác và có thể được dùng để truyền và nhận dữ liệu. Socket lắng nghe lúc đó vẫn tồn tại

và sẵn sàng để có thể được truyền vào một socket trống khác và tạo ra một kết nối khác.

Khi ứng dụng đóng vai trò là Server hay Receiver kết thúc, nó cần phải đóng socket listen, cũng như tất cả các socket đã kết nối.

Đoạn code sau minh họa việc tạo ra Bluetooth socket server và lắng nghe kết nối:

```
// 1. Kết nối tới socket server secion
RSocketServ socketServ;
socketServ.Connect();
TProtocolDesc pInfo;
_LIT(KL2Cap, "L2CAP"); // hoặc là RFCOMM tùy giao thức sử dụng

// 2. Mở socket để lắng nghe
RSocket listen;
listen.Open(socketServ,KL2Cap);

// 3. Khởi tạo đối tượng địa chỉ bluetooth socket
TBTSockAddr addr;
addr.SetPort(KListeningPSM);

// Kết buộc socket lắng nghe
User::LeaveIfError(listen.Bind(addr));

// 4. Bắt đầu lắng nghe các yêu cầu kết nối từ các thiết bị khác.
User::LeaveIfError(listen.Listen(2));

// 5. Chờ đợi và thực hiện kết nối khi có yêu cầu kết nối
RSocket accept;
TRequestStatus status;
User::LeaveIfError(accept.Open(socketServ));
```

```
listen.Accept(accept,status);  
User::WaitForRequest(status);
```

// Nếu status == KerrNone : accept socket đã kết nối thành công.

Trước khi chấp nhận một kết nối tới, cần phải thiết lập các yêu cầu về an toàn và bảo mật đối với các kết nối (xem phần Bluetooth Security Manager), và cần phải quảng bá các dịch vụ có thể sử dụng của thiết bị để các thiết bị khác sử dụng khi kết nối ( Bluetooth Service Discovery Database ) .

#### **7.4.3. Xây dựng Bluetooth Socket Client : Tìm kiếm và kết nối tới thiết bị là Server.**

Để một ứng dụng trên thiết bị Bluetooth có thể kết nối tới một thiết bị Bluetooth khác, nó cần phải thực hiện các bước sau : đầu tiên, cần phải xác định được thiết bị server mà người sử dụng muốn kết nối tới, tiếp đó, ứng dụng cần xác định được xem dịch vụ mà nó cần có ở server hay không, cuối cùng, ứng dụng thực hiện kết nối với thiết bị server, và thực hiện việc trao đổi dữ liệu nếu kết nối thành công.

##### **7.4.3.1. Chọn thiết bị để kết nối tới :**

Ứng dụng có thể xác định được thiết bị mà nó cần kết nối tới bằng một số cách sau :

- + Kết nối cứng (hard-wired) mặc định đã được thiết lập trước.
- + Kết nối với các thiết bị đã được lưu thông tin từ trước.
- + Chọn lựa bởi người dùng thông qua Bluetooth Device Selection UI.
- + Hoặc được xác định qua lập trình : ứng dụng tự động xác định thiết bị mà nó muốn kết nối.

##### **7.4.3.2. Truy vấn thông tin về thiết bị xung quanh:**

+ Mỗi thiết bị Bluetooth có một địa chỉ 48 bit duy nhất được xây dựng bên trong phần cứng của nó, và có thể có một tên thiết bị để thể hiện tới người dùng.

- + Địa chỉ và tên thiết bị được thực hiện truy vấn thông qua lớp socket RHostResolver của Symbian. Trong khi truy vấn, một lớp địa chỉ socket đặc biệt được sử dụng, đó là lớp TInquirySockAddr, chứa đựng địa chỉ Bluetooth, mã truy cập, các dịch vụ (services) và các lớp thiết bị, được cung cấp trong khi thực hiện truy vấn.

**b.1. Thực hiện truy vấn địa chỉ của thiết bị kết nối tới được thực hiện qua các bước :**

- + Kết nối tới Socket server (RSocketServ) và chọn loại giao thức sử dụng bằng cách gọi hàm RSockeServ::FindProtocol(). Do các truy vấn về địa chỉ và tên thiết bị được cung cấp bởi chồng giao thức BTLinkManager, do đó chọn giao thức này.
  - + Tạo ra và thiết lập một đối tượng RHostResolver.
  - + Thiết lập tham số TInquirySockAddr cho quá trình truy vấn: đối với truy vấn về địa chỉ, cờ KHostResInquiry cần phải được thiết lập thông qua hàm TInquirySockAddr::SetAction(KHostResInquiry);
- Sau đó, việc truy vấn có thể được bắt đầu bằng lời gọi hàm:

```
RHostResolver::GetByAddress(const TSockAddr &anAddr, TNameEntry  
&aResult);  
RHostResolver::GetByAddress(const TSockAddr &anAddr, TNameEntry  
&aResult, TResquestStatus&aStatus);
```

- + Sau khi kết thúc hàm GetByAddress(), tham số TnameEntry &aResult sẽ chứa địa chỉ và các lớp của thiết bị đầu tiên tìm được, hoặc là không xác định nếu không tìm thấy thiết bị nào. Tham số aStatus chứa mã lỗi trả về, là KErrNone nếu thành công.
- + Để tiếp tục truy vấn các thiết bị khác nếu có, sử dụng hàm RHostResolver::Next() cho tới khi KerrHostResNoMoreResults được trả về.

Đoạn code sau mô tả cách thực hiện việc truy vấn địa chỉ và tên của các thiết bị Bluetooth xung quanh:

```
// 1. Kết nối socket server section
RSocketServ socketServ;
socketServ.Connect();
TProtocolDesc pInfo;
_LIT(KL2Cap, "BTLinkManager");
User::LeaveIfError(socketServ.FindProtocol(KL2Cap,pInfo));

// 2 Tạo ra và khởi tạo đối tượng RHostResolver
RHostResolver hr;
User::LeaveIfError(hr.Open(socketServ,pInfo.iAddrFamily,pInfo.iProtocol));

// 3. Thiết lập truy vấn và thực hiện truy vấn.
TInquirySockAddr addr;
TNameEntry name;
addr.SetIAC(KGIAC);
addr.setAction(KHostResInquiry);
TRequestStatus status;
hr.GetByAddress(addr, entry, status); // thông tin tra về được lưu trong entry
User::WaitForRequest(status);

// 4. Xử lý các thông tin về địa chỉ thiết bị được trả về trong entry
.....
```

### **b.2. Truy vấn tên của thiết bị:**

Để truy vấn tên của thiết bị khác, ta thực hiện tương tự như việc truy vấn địa chỉ của thiết bị như trên với thiết lập đối tượng TInquirySockAddr là KHostResName qua hàm TInquirySockAddr::SetAction(KhosResName);

Ví dụ:

```
// Thực hiện truy vấn tên thiết bị xung quanh:  
addr.SetAction(KHostResName);  
hr.GetByAddress(addr, entry, stat);  
User::WaitForRequest(stat);  
TPtrC deviceName;  
if (stat == KErrNone)  
    deviceName.Set(entry().iName);
```

Việc truy vấn địa chỉ và tên của thiết bị xung quanh có thể thực hiện cùng một lúc bằng cách thiết lập đối tượng *TInquirySockAddr* với hàm *SetAction* như sau :

*TInquirySockAddr::SetAction(KhostResName | KhostResEntry).*

#### **7.4.3.3. Truy vấn về dịch vụ được cung cấp trên thiết bị Server :**

Trên một thiết bị Bluetooth có thể cung cấp nhiều dịch vụ Bluetooth , bảng thông tin về dịch vụ cung cấp đó có thể có từ lớp của thiết bị (class of device). Lớp của thiết bị được lấy thông qua lời gọi hàm : *TInquirySockAddr::MajorClassOfDevice()* sau khi thực hiện truy vấn địa chỉ của thiết bị.

Trong những tình huống cụ thể, việc xác định thông tin về dịch vụ của lớp thiết bị không đầy đủ để có thể xác định sẽ chọn thiết bị nào, trong trường hợp đó, việc truy vấn tìm kiếm dịch vụ có thể thực hiện đối với từng thiết bị tìm thấy để có thể chọn đúng thiết bị cần.Việc truy vấn dịch vụ SDP (Service Discovery Database ) được thực hiện thông qua Bluetooth Service Discovery Agen API (xem chi tiết trong phần sau).

#### **7.4.3.4. Kết nối với thiết bị đã được chọn và thực hiện trao đổi dữ liệu:**

Khi mà đã xác định được thiết bị và dịch vụ được cung cấp, chúng ta có thể thực hiện kết nối với thiết bị và dịch vụ trên thiết bị đó, và sử dụng dịch vụ được cung cấp.

Việc kết nối với thiết bị được thực hiện thông qua hàm *Connect()* của lớp *RSocket* với giao thức là L2CAP hoặc RFCOMM. Đối với Bluetooth

socket L2CAP, “port” sử dụng là Protocol/Service Multiplexer (PSM) , đối với Bluetooth Socket RFCOMM thì “port” sử dụng là server channel.

+ Hàm Connect() được sử dụng :

```
void RSocket::Connect(TSockAddr& anAddr,  
                      TRequestStatus &aRequest)
```

Tham số : anAddr : địa chỉ socket của thiết bị.

aRequestStatus : Chứa mã lỗi trả về.

Địa chỉ của socket được thể hiện bởi lớp TBTSockAddr vốn kế thừa từ lớp TSockAddr. Mỗi thiết bị Bluetooth có một địa chỉ 48 bit được thể hiện bởi lớp TBTDevAddr, lớp TBTSockAddr có hàm SetBTAddr() dùng để gán địa chỉ thiết bị Bluetooth cho đối tượng của lớp này và hàm SetPort() để chọn một Channel cụ thể.

```
void SetBTAddr(const TBTDevAddr& aRemote);  
void SetPort(TUInt aPort);
```

Mỗi client phải có các thành phần sau:

**SocketServer (RSocketServ):** Dùng để giao tiếp với socket server của thiết bị.

**SendingSocket:** Dùng để kết nối, nhận và gửi dữ liệu đến server socket

Đoạn code sau mô tả việc thực hiện kết nối :

```
RSocketServ SocketServer;  
TInt err;  
err = SocketServer.Connect();  
User::LeaveIfError(err);  
RSocket SendingSocket;  
err = SendingSocket.Open(SocketServer,  
                         KBTAddrFamily,  
                         KSockStream,  
                         KRFCOMM);  
User::LeaveIfError(err);  
TBTSockAddr addr;
```

```
TRquestStatus status;  
addr.SetBTAddr(...);  
addr.SetPort(...);  
SendingSocket.Connect(addr, status);  
// Nhận và gửi dữ liệu ở đây .....
```

#### 7.4.4. Trao đổi dữ liệu thông qua Bluetooth socket :

Sau khi thiết lập kết nối thành công, việc nhận và gửi dữ liệu với Bluetooth socket cũng tương tự như với bất kỳ loại socket nào khác trên Symbian (IrDA socket, Internet socket).

\* Để nhận dữ liệu ta có thể dùng các hàm Read(), Recv() , hoặc ReceiveOneOrMore():

```
- void Read( TDes8& aDesc, TRquestStatus& aStatus );  
- void Recv( TDes8& aDesc,  
             TUint flags,  
             TRquestStatus& aStatus);  
- void Recv( TDes8& aDesc,  
             TUint flags,  
             TRquestStatus& aStatus,  
             TSockXfrLength& aLen);  
- void RecvOneOrMore( TDes8& aDesc,  
                      TUint flags,  
                      TRquestStatus& aStatus,  
                      TSockXfrLength& aLen);
```

Trong đó:

- + TDes8& aDesc : buffer chứa dữ liệu nhận được.
- + TRquestStatus& aStatus : Sau khi hàm kết thúc, biến này chứa mã lỗi trả về. Nếu không có lỗi, giá trị của aStatus là KerrNone.
  - + TUint flags : các thông tin về Protocol, I/O.

+ TsockXfrLength& aLen : Chiều dài dữ liệu nhận được.

\* Để gửi dữ liệu, ta dùng hàm Write():

```
+ void Write(const TDesC8& aDesc, TRequestStatus& aStatus);  
+ void Send(const TDesC8& aDesc,  
           TUint someFlags,  
           TRequestStatus& aStatus);  
+ void Send(const TDesC8& aDesc,  
           TUint someFlags,  
           TRequestStatus& aStatus,  
           TSockXfrLength& aLen);
```

Trong đó:

- + TDesC8& aDesc : Dữ liệu cần gửi đi.
- + TRequestStatus& aStatus : Sau khi hàm kết thúc, mã lỗi được trả về qua biến này.
- + TUint flags : các thông tin về Protocol, I/O.
- + TsockXfrLength& aLen : Chiều dài dữ liệu gửi đi.

## 7.5. Bluetooth Service Discovery Database:

Mỗi dịch vụ Bluetooth được lưu trong một record trong SDP database, nhờ đó, các thiết bị khác có thể biết được trên thiết bị Bluetooth đó có những dịch vụ nào. Bluetooth Service Discovery Database cho phép các dịch vụ trên nội bộ thiết bị có thể đưa các đặc tính của nó vào trong Bluetooth Service Database, nhờ đó, các thiết bị Bluetooth khác có thể phát hiện được là dịch vụ đó được hỗ trợ trên thiết bị đó.

### 7.5.1. Kết nối vào Bluetooth Service Discovery Database :

Để có thể sử dụng Service Discovery Database, client phải thực hiện các bước sau :

- + Tạo một phiên làm việc (session) với đối tượng database **RSdp** và mở một kết nối
  - + Tạo ra một phiên làm việc con (subsession) tới đối tượng Database **RsdpDatabase** và mở kết nối. Một client có thể có nhiều phiên làm việc con đồng thời.
  - + Đóng subsession và session khi đã sử dụng xong.

Đoạn code sau minh họa cách thức kết nối tới Service Database :

```
//1. Tạo và mở một phiên làm việc đến database
```

```
RSdp sdp;
```

```
User::LeaveIfError(sdp.Connect());
```

```
// 2. Create and open a subsession
```

```
RSdpDatabase sdpSubSession;
```

```
User::LeaveIfError(sdpSubSession.Open(sdp));
```

```
...
```

```
// 3. Đóng các kết nối đã mở.
```

```
sdpSubSession.Close();
```

```
sdp.Close();
```

### 7.5.2. Đăng kí một dịch vụ vào Service Database :

Sau khi Service Discovery Database được mở ra, một service record có thể được tạo ra. Việc đó được thực hiện bằng việc cung cấp một giá trị UUID (Bluetooth Universally Unique Identifier) cho lớp dịch vụ (service class) của record. Service class có thể là một giá trị UUID hoặc một list các giá trị UUID ở dạng DES.

Các bước đăng kí một dịch vụ vào Database :

- + Tạo ra một record dịch vụ trống thuộc đối tượng **TSdpServRecordHandle**, là handle của một service record.
  - + Nếu thuộc tính của service class là một giá trị đơn UUID, tạo ra và thiết lập một đối tượng **TUUID**. Nếu thuộc tính của service class là một danh

sách các giá trị UUID, tạo ra một đối tượng danh sách thuộc tính thuộc lớp CSdpAttrValueDES hoặc lớp CSdpAttrValueDEA , và gọi hàm MsdpElementBuilder::BuildUUIDL() đối với từng thuộc tính để đưa vào danh sách thuộc tính.

+ Gọi hàm RSdpDatabase::CreateServiceRecordL() ở trong một subsession đã mở. Hàm trả về một handle trỏ tới service record mới tạo.

\* Đoạn code sau mô tả cách tạo một service record với một đối tượng dịch vụ đơn (single service class) :

```
/* Giả sử sdpSession là một phiên làm việc với đối tượng Database đã được mở thành công, sdpsubsession : phiên làm việc con */
```

```
// 1. Tạo ra một Record Handle trống
```

```
TSdpServRecordHandle recordHandle = 0;
```

```
// 2.Tạo ra đối tượng service class đơn và thiết lập giá trị của nó
```

```
TUUID uuid(0x20000);
```

```
// 3. Đưa record mới tạo vào Database .
```

```
sdpsubSession.CreateServiceRecordL(uuid, recordHandle);
```

\* Đối với danh sách thuộc tính, ta dùng hàm StartListL() để báo hiệu việc bắt đầu thao tác với list, hàm BuildxxxL() để thêm dữ liệu vào danh sách, tùy vào loại dữ liệu mà sử dụng các hàm BuildxxxL() khác nhau, như hàm BuildURLL(), BuildDESL(), BuildIntL(), BuildUUIDL()....., Sau đó, ta dùng hàm EndList() để đóng danh sách lại.

\* Giả sử ứng dụng cần đưa vào service record một thuộc tính là danh sách các giao thức (protocol) mà ứng dụng hỗ trợ, đoạn chương trình sau minh họa việc tạo một danh sách gồm 2 protocol. Protocol đầu tiên là L2CAP. Protocol thứ 2 là RFCOMM kèm theo một số nguyên đại diện cho kênh truyền (channel), và đưa vào Database.

```
/ /1. Tạo ra một record handle trống :
```

```
TSdpServRecordHandle recordHandle =0;
```

```
//2. Tạo một đối tượng CSdpAttrValueDES
CSdpAttrValueDES* protocolDescriptorList =
    CSdpAttrValueDES::NewDESL(NULL);
CleanupStack::PushL(protocolDescriptorList);
// kênh sử dụng:
Tbuf8<1> channel;
Channel.Append((Tchar)port); /* với port là số
nguyên TInt cho biết port sử dụng.*/
//3 . Tạo danh sách thuộc tính
protocolDescriptorList
->StartListL()
->BuildDESL()
->StartListL()
->BuildUUIDL(KL2CAP)
->EndListL()

->BuildDESL()
->StartListL()
->BuildUUIDL(KRFCOMM)
->BuildUintL(channel)
->EndListL()
->EndListL();
//4. Đưa record vào Database :
sdpSubSession.CreateServiceRecord(*protocolDescriptorList
, recordhandle);
CleanUpStack::Pop(); // protocolDescriptorList
```

### 7.5.3. Thiết lập các thuộc tính trong một Service Record:

Các thuộc tính trong một Service Record có thể được thiết lập hoặc thay đổi bằng cách gọi hàm :

```
RSdpDatabase::UpdateAttributeL(TSdpServRecordHandle aHandle,
TSdpAttributeID aAttrID, XXX & aValue)
```

Tùy thuộc vào loại thuộc tính mà XXX có thể là :CSdpAttrValue, TUint, TDesC16, TdesC8

Trong đó :

aHandle : Handle của service record cần cập nhật

aAttrID : Attribute ID cần cập nhật trên Service Record

aValue : Giá trị của thuộc tính

\* Để xóa một service record ra khỏi database, ta dùng hàm :

*RSdpDatabase::DeleteRecordL(TSdpServRecordHandle aHandle)*

Với aHandle : là handle tới Service record cần xóa.

## 7.6. Bluetooth Service Discovery Agent:

Bluetooth Service Discovery Agent cho phép các lập trình viên có thể xác định được các dịch vụ Bluetooth , các thuộc tính của các dịch vụ có tồn tại trên các thiết bị Bluetooth xung quanh.Các hàm API Service Discovery Agent là một trong hai loại hàm API cho phép sử dụng giao thức Bluetooth Service Discovery. Loại hàm thứ hai là Bluetooth Service Discovery Database cho phép các dịch vụ có trên thiết bị có thể đưa các thuộc tính của nó vào trong Database về dịch vụ có trên thiết bị đó.

Lớp chính cho phép giao thức Bluetooth service discovery truy vấn tới thiết bị khác là lớp **CSdpAgent**. Có hai truy vấn cơ bản có thể thực hiện sử dụng lớp này đó là:

+ Để có thể lấy được các dịch vụ trên thiết bị khác, truy vấn các lớp của dịch vụ mà bạn muốn lấy bằng cách sử dụng đối tượng **CSdpSearchPattern**.

+ Để lấy các thuộc tính của một dịch vụ cụ thể, thiết lập danh sách các thuộc tính lấy về bằng cách sử dụng đối tượng **CSdpAttrIdMatchList**.

Việc sử dụng lớp **CSdpAgent** phải được cài đặt với interface **MSdpAgentNotifier** để nhận các đáp ứng từ truy vấn.

Các lớp **CSdpAgent**, **CSdpSearchPattern**, **CSdpAttrIdMatchList**, **MSdpAgentNotifier** đều được khai báo trong file header “btsdp.h” và được liên kết với thư viện “bluetooth.lib”.

### 7.6.1. Truy vấn các dịch vụ trên thiết bị khác với Bluetooth Service

#### Discovery Agent:

Bluetooth Service Discovery Agent được dùng để thực hiện các truy vấn về các dịch vụ Bluetooth sẵn có trên một thiết bị cụ thể. Nó thường được sử dụng sau khi một thiết bị Bluetooth phù hợp ở trong phạm vi cho phép được xác định thông qua các hàm Bluetooth Socket API (xem lại phần 7.4).

Sau khi kết thúc tìm kiếm dịch vụ, kết quả trả về là các record handle của các dịch vụ thuộc lớp hoặc các lớp đã được định nghĩa (các số UUID). Trước khi một ứng dụng có thể bắt đầu tìm kiếm các dịch vụ được cung cấp trên một thiết bị cụ thể, nó phải tạo ra một lớp cài đặt của lớp **MSdpAgentNotifier**. Kết quả sau khi thực hiện truy vấn được trả về thông qua các hàm callbacks của cài đặt giao diện **MSdpAgentNotifier**.

Các bước thực hiện tìm kiếm dịch vụ được thực hiện như sau:

**Bước 1:** Tạo ra một đối tượng CSdpAgent và cung cấp cho nó một cài đặt của lớp **MSdpAgentNotifier** đã được tạo ra trước, và chỉ định địa chỉ của thiết bị Bluetooth mà ứng dụng muốn truy vấn.

**Bước 2 :**Tạo ra một đối tượng **CSdpSearchPattern** và chỉ định các lớp dịch vụ cần tìm kiếm. Các lớp dịch vụ đó có thể được đưa vào thông qua hàm **CSdpSearchPattern::Add(const TUUUID &aUUID);**

**Bước 3 :** Thiết lập kiểu tìm kiếm trên đối tượng CSdpAgent sử dụng thông qua hàm **:CSdpAgent::SetRecordFilterL()** nhằm thiết lập một danh sách các dịch vụ mà ứng dụng quan tâm.

**Bước 4:** Gọi hàm **CSdpAgent::NextRecordRequestL()** để lấy các kết quả tìm kiếm cho tới khi hết, hoặc đã tìm được kết quả cần tìm. Khi gọi hàm này, quá trình tìm kiếm được thực hiện, khi tìm kiếm được một dịch vụ, hàm **MSdpAgentNotifier::NextRecordRequestComplete()** sẽ được gọi và thông báo tới ứng dụng thông qua lớp được cài đặt giao diện **MSdpAgentNotifer** của ứng dụng.

Đoạn code sau minh họa các bước trên :

Giả sử : rcvr là lớp được cài đặt giao diện lớp **MSdpAgentNotifer**, và devAddr là địa chỉ của thiết bị đang thực hiện truy vấn tới.

```
/* 1 : Tạo ra một đối tượng thuộc lớp CSdpAgent.*/
CSdpAgent *agent = CSdpAgent::NewLC(rcvr, devAddr);
/* 2: Tạo ra một mẫu tìm kiếm (search pattern) thuộc lớp
CSdpSearchPattern và đặt một service classes vào */
CSdpSearchPattern* list = CSdpSearchPattern::NewL();
List->AddL(0x0100);
/* 3: set the search pattern on the agent */
agent->SetRecordFilterL(*list);
/*4: Bắt đầu quá trình tìm kiếm , kết quả tìm kiếm được
trả về qua hàm callbacks của rcvr */
agent->NextRecordRequestL();
```

### 7.6.2. Tìm kiếm các thuộc tính dịch vụ:

Dữ liệu của các service record chứa trong các thuộc tính, mỗi thuộc tính có một ID được qui định trước, một loại thuộc tính, và một giá trị của thuộc tính đó.

Như việc tìm kiếm dịch vụ, kết quả của tìm kiếm các thuộc tính dịch vụ được trả về thông qua các hàm callback của lớp giao diện **MSdpAgentNotifier** mà ứng dụng phải cài đặt.

Các bước sau mô tả quá trình tìm kiếm thuộc tính dịch vụ:

1: Tạo ra một đối tượng **CSdpAttrIdMatchList** qui định các thuộc tính cần lấy về ( gọi là match list ).

2: Thêm các IDs của thuộc tính vào match list sử dụng hàm : **CSdpAttrIdMatchList::AddL(TAttrRange aRange)**. Các IDs đóng gói trong kiểu dữ liệu **TAttrRange**. Để bỏ các thuộc tính ra khỏi match list sử dụng hàm **CSdpAttrIdMatchList::RemoveL(TAttrRange aRange)**.

3: Bắt đầu thực hiện truy vấn sử dụng hàm: **CSdpAgent::AttributeRequestL()** , tùy vào nhu cầu sử dụng mà có nhiều hàm overloaded có thể sử dụng.

Ví dụ về truy vấn các thuộc tính dịch vụ:

( Giả sử agent là một đối tượng CSdpAgent, và serviceHandle là handle của service record).

```
// 1. Tạo ra một match list
CSdpAttrIdMatchList *matchList
=CSdpAttrIdMatchList::NewL();
CleanupStack::PushL(matchList);

//2. Thêm vào ID của thuộc tính cần lấy :
matchList->AddL(TAttrRange(0x102));

//3. Đặt matchList vào agent
agent-> AttributeRequestL(serviceHandle,
*matchList);

CleanupStack::PopAndDestroy(); //matchList
```

### 7.6.3. Tạo ra đối tượng để quản lý các kết quả truy vấn:

Các ứng dụng thực hiện các truy vấn về dịch vụ và thuộc tính qua đối tượng **CSdpAgent** phải được cài đặt giao diện **MSdpAgentNotifier** để quản lý đáp ứng trả về.

+ Quản lý kết quả truy vấn dịch vụ:

Khi việc truy vấn dịch vụ kết thúc, ứng dụng được hệ điều hành thông báo tới thông qua hàm **NextRecordRequestComplete()**:

```
virtual void NextRecordRequestComplete(TInt aError,
TSdpServRecordHandle aHandle, TInt aTotalRecordsCount)
```

Trong đó : aError : chứa mã lỗi trả về.

aHandle : Handle tới service record của dịch vụ tìm được.

aTotalRecordsCount :tổng số lượng các record thỏa điều kiện tìm kiếm.

+ Quản lý truy vấn thuộc tính:

Khi truy vấn thuộc tính kết thúc, mỗi thuộc tính được trả về với ID của thuộc tính sử dụng hàm : **AttributeRequestResult()**.

```
virtual void AttributeRequestResult(TSdpServRecordHandle
aHandle, TSdpAttributeID aAttrID, CSdpAttrValue*
aAttributeValue)
```

Trong đó :

aHandle : Service record thực hiện truy vấn

aAttrID : chứa ID của thuộc tính.

aAttrValue : chứa giá trị của thuộc tính.

Khi đã trả về hết các thuộc tính, hàm **AttributeRequestComplete()** được gọi.

```
virtual void AttributeRequestComplete( SdpServRecordHandle aHandle, TInt  
aError )= 0 ;
```

Trong đó : aHandle : Service record thực hiện truy vấn.

aError : KErrNone hoặc một lỗi SDP.

## 7.7. Bluetooth security manager:

### 7.7.1. Tổng quan

Bluetooth Security Manager (BSM) cho phép ứng dụng Bluetooth thiết lập các chế độ bảo mật mà một kết nối phải đáp ứng. Các thiết lập bảo mật chỉ đơn giản là có authentication, có authorization và có encryption hay không. Authentication nghĩa là cả 2 ứng dụng phải nhập cùng 1 khóa để có thể giao tiếp, authorization nghĩa là ứng dụng sẽ hỏi xem người dùng có chấp nhận kết nối hay không, encryption nghĩa là dữ liệu được mã hóa khi truyền nhận. Các thiết lập bảo mật này được áp dụng cho một server, một protocol, một kênh truyền cụ thể. Bluetooth security manager bảo đảm rằng kết nối từ bên ngoài vào phải đáp ứng các yêu cầu bảo mật.

Trong Symbian, Bluetooth Security Manager (BSM) được cài đặt như một server. Các hàm API của BSM cung cấp cho client lớp **TBTServiceSecurity** để đóng gói các cấu hình bảo mật. Lớp **TBTServiceSecurity** được định nghĩa trong file header “btmanclient.h”.

Trước khi sử dụng BSM, ứng dụng phải mở một phiên làm việc (session) tới server sử dụng lớp **RBTMan** và sau đó mở một subsession sử dụng **RBTSecuritySettings** để đăng ký hoặc bỏ đăng ký dịch vụ với server của BSM.

Sử dụng đối tượng Security Setting thuộc lớp **TBTServiceSecurity** dùng để thiết lập khi sử dụng authentication, autorisation, và hoặc cần encryption.

### 7.7.2. Kết nối vào Bluetooth Security Manager.

Để có thể sử dụng Bluetooth Security Manager, ứng dụng phải thực hiện các bước sau:

**1 :** Tạo ra một phiên làm việc mới và kết nối với BSM server thông qua lớp **RBTMan**. Lớp **RBTMan** cung cấp kết nối tới BSM server, lớp này được định nghĩa trong file header “btmanclient.h” như sau :

```
class RBTMan : public RSessionBase
{
public:
    IMPORT_C RBTMan();
    IMPORT_C TInt Connect();
    IMPORT_C TVersion Version() const;
};
```

**2:** Tạo ra một phiên làm việc con( subsession) tới BSM server sử dụng lớp **RBTSecuritySettings**, và mở ra một phiên làm việc mới. Một ứng dụng có thể có nhiều phiên làm việc con nếu cần.

**3:** Đóng subsession và session khi sử dụng xong : sử dụng phương thức: **RBTMan::Close()**, và **RBTSecuritySettings::Close()**.

Đoạn code sau mô tả cách kết nối tới Bluetooth Security Manager :

```
/* 1. Tạo và mở một phiên làm việc mới và kết nối tới BSM server thông qua
lớp RBTMan */
RBTMan secMan;
User::LeaveIfError(secMan.Connect());
/* 2. Tạo và mở một subsession */
RBTSecuritySettings secmanSubSession;
User::LeaveIfError(secmanSubSession.Open(secMan));
...
/* 3. Đóng session và subsession khi sử dụng xong*/
```

```
secmanSubSession.Close();  
secMan.Close();
```

### 7.7.3. Thiết lập các chế độ bảo mật :

Trong tất cả các thiết bị Bluetooth , việc thiết lập các yêu cầu về bảo mật và an toàn đối với một dịch vụ chỉ đơn giản ở việc chỉ định có sử dụng hay không các chế độ bảo mật là authentication, authorisation, hay encryption. Để thiết lập các yêu cầu về bảo mật, thực hiện các bước sau :

- \* Tạo ra một đối tượng thuộc lớp **TBTServiceSecurity** để thiết lập các yêu cầu về an toàn,bảo mật. Khởi tạo đối tượng với giao thức Bluetooth mà dịch vụ sử dụng, port của dịch vụ, một định danh duy nhất của dịch vụ.
- \* Thiết lập trong đối tượng **TBTServiceSecurity** là cần chế độ an toàn và bảo mật nào : authentication, authorisation, hay encryption.
- \* Gọi hàm **RBTSecuritySetting::RegisterService()** để đăng kí các thiết lập với BSM server.

Đoạn code ví dụ sau thiết lập các an toàn và bảo mật của dịch vụ là authentication, authorisation, và encryption :

```
/* Giả sử secmanSubSession là một subsession đã mở như ở  
trên, và channel là port mà một dịch vụ RFCOMM đang chạy  
trên đó*/  
  
// 1. Tạo ra đối tượng TBTServiceSecurity  
TUid serviceUID;  
serviceUID.iUid = 0X10008AD0;  
TBTServiceSecurity secSettings(serviceUID, KSolBtRFCOMM,  
channel);  
  
// 2.Thiết lập các tùy chọn về an toàn, bảo mật.  
secSettings.SetAuthentication(ETrue);  
secSettings.SetAuthorisation(ETrue);  
secSettings.SetEncryption(ETrue);
```

```
// 3. Đăng ký dịch vụ vào BMS server.  
TRequestStatus status;  
secmanSubSession.RegisterService(secSettings, status);  
User::WaitForRequest(status);
```

## 7.8. Bluetooth Device Selection UI.

Khi có nhiều thiết bị Bluetooth phù hợp tìm kiếm được xung quanh, ứng dụng cần phải hiển thị chúng tới người dùng và để người dùng chọn thiết bị thích hợp để thực hiện kết nối. Symbian cung cấp một chức năng cho phép thực hiện tìm kiếm các thiết bị Bluetooth và hiển thị chúng cho người dùng. Việc này được thực hiện thông qua lớp **RNotifier**. Lớp này được định nghĩa trong file header “e32std.h”.

Lớp **TBTDeviceSelectionParams** cho phép một ứng dụng truyền vào các tham số khởi tạo cho quá trình chọn lựa

```
class TBTDeviceSelectionParams  
{  
public:  
    IMPORT_C TBTDeviceSelectionParams();  
    IMPORT_C void SetUUID(const TUUUID& aUUID);  
    IMPORT_C void SetDeviceClass(  
        TBTDeviceClass aClass);  
    IMPORT_C const TUUUID& UUID();  
    IMPORT_C TBTDeviceClass DeviceClass();  
    IMPORT_C TBool IsValidDeviceClass();  
    IMPORT_C TBool IsValidUUID();  
    ...  
};
```

Bằng cách dùng hàm **SetDeviceClass()** , ứng dụng có thể giới hạn số lượng những thiết bị cần tìm kiếm.

Ứng dụng cũng có thể xác định các UUID của dịch vụ mà nó cần tìm. Điều này sẽ làm cho quá trình tìm kiếm chỉ tìm kiếm những thiết bị có hỗ trợ dịch vụ mà ứng dụng yêu cầu. Việc này thực hiện thông qua hàm SetUUID(). Để tìm kiếm tất cả các thiết bị, ứng dụng nên để đối tượng **TBTDeviceSelectionParams** ở giá trị mặc định.

Sau khi người dùng chọn thiết bị muốn giao tiếp, thông tin về thiết bị mà người dùng lựa chọn sẽ được trả về cho ứng dụng thông qua đối tượng của lớp **TBTDeviceResponseParams**. Lớp này được định nghĩa trong file header “btextnotifiers.h” . Định nghĩa của nó như sau:

```
class TBTDeviceResponseParams
{
public:
    IMPORT_C TBTDeviceResponseParams();
    IMPORT_C void SetDeviceAddress(
        const TBTDevAddr& aBDAddr);
    IMPORT_C void SetDeviceName(
        const TDesC& aName);
    IMPORT_C void SetDeviceClass(
        TBTDeviceClass aClass);
    IMPORT_C const TBTDevAddr& BDaddr() const;
    IMPORT_C const TDesC& DeviceName() const;
    IMPORT_C TBTDeviceClass DeviceClass();
    IMPORT_C TBool IsValidBDAddr() const;
    IMPORT_C TBool IsValidDeviceName() const;
    IMPORT_C TBool IsValidDeviceClass();
    ...
};
```

Hàm IsValidxxxx() được dùng để bảo đảm rằng các thông tin liên quan đã được thiết lập trong lớp. Ví dụ, nếu hàm IsValidDBAddr() trả về giá trị true, ứng dụng biết được rằng địa chỉ của thiết bị Bluetooth đã được thiết lập trong lớp **TBTDeviceResponseParams**. Nếu hàm này trả về giá trị false có nghĩa là

chưa có dữ liệu về địa chỉ thiết bị Bluetooth trong lớp này. Ứng dụng có thể nhận được địa chỉ thiết bị thông qua hàm BDAddr() và tên của thiết bị thông qua hàm DeviceName().

Symbian cung cấp một tập các gói dữ liệu đóng gói các lớp vừa trình bày bên trên:

**TBTDeviceSelectionParamsPckg**

**TBTDeviceResponseParamsPckg**

Nói tóm lại, quy trình để thực hiện tìm kiếm thiết bị được mô tả như sau:

```
// Kết nối vào server trước khi có thể sử dụng
RNotifier not;
User::LeaveIfError(not.Connect());
TBTDeviceSelectionParamsPckg selectionFilter;
// Thực hiện tìm kiếm và hiển thị cho người dùng
not.StartNotifierAndGetResponse(
    status,
    KDeviceSelectionNotifierUid,
    selectionFilter,
    aResponse
);

// Đợi người dùng chọn thiết bị cần giao tiếp
User::WaitForRequest(status);
if (status.Int() == KErrNone)
{
    if (aResponse().IsValidDeviceName())
    {
        success = ETrue;
    }
    else
    {
        iReporter.Error(_L("Failed to connect"));
    }
}
```

```
    }
    else
    {
        iReporter.Error(_L("No device selected"));
    }
    not.CancelNotifier(KDeviceSelectionNotifierUid);
    not.Close();
    return success;
}
```

## 7.9. Xây dựng ứng dụng Bluetooth trên Symbian OS với Series 60 SDK

Ứng dụng Bluetooth trên Symbian OS cũng được tạo ra như bất kì một ứng dụng nào khác được xây dựng với Symbian C++. Bộ công cụ phát triển ứng dụng Series 60 SDK và Series 80 Developer Platform 2.0 SDK cung cấp một tập các hàm Bluetooth APIs cho việc xây dựng các ứng dụng Bluetooth với Symbian C++, và một bộ giả lập thiết bị để test ứng dụng. Chúng ta có thể test ứng dụng với các trường hợp sau :

- 1 : Tìm kiếm thiết bị Bluetooth xung quanh.
- 2 : Truy vấn các dịch vụ Bluetooth được hỗ trợ.
- 3 : Thiết lập một kết nối tới thiết bị khác.
- 4 : Gửi và nhận dữ liệu.

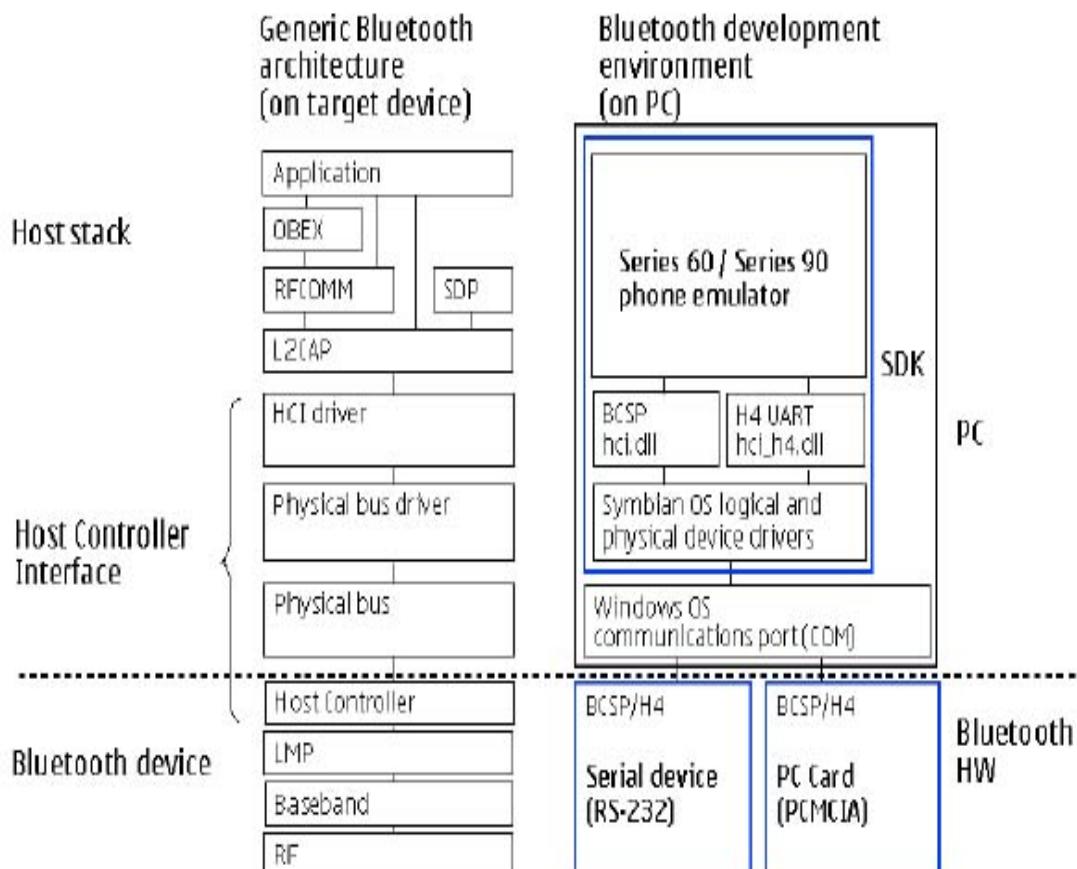
Trong khuôn khổ của luận văn, chúng em chỉ xin giới thiệu cách xây dựng ứng dụng Bluetooth với bộ công cụ phát triển ứng dụng Series 60 SDK.

### 7.9.1. Sự khác nhau về Bluetooth trên thiết bị ảo và thiết bị thật.

Môi trường phát triển ứng dụng ( bộ giả lập thiết bị trên PC) và môi trường thiết bị thật khác nhau. Thể hiện ở các điểm sau :

- \* Hiệu năng của thiết bị ảo và thiết bị thật khác nhau.
- \* Chỗ giao thức Bluetooth và các cài đặt bên dưới của thiết bị ảo và thiết bị thật khác nhau như thể hiện trong hình bên dưới đây.
- \* Thiết bị ảo không nạp tất cả các dịch vụ Bluetooth của nó khi khởi động ,trong khi thiết bị thật nạp thì có. Nếu muốn một thiết bị khác

muốn yêu cầu kết nối vào máy ảo thì trước đó máy ảo phải khởi tạo các dịch vụ cần thiết. Để làm được điều này xin xem thêm các ví dụ về Bluetooth đi kèm với bộ SDK.



Hình 7-6 Sự khác biệt giữa chồng giao thức Bluetooth trên thiết bị thật và trên máy ảo

Do đó, ứng dụng trước khi hoàn thành cần phải được test cẩn thận trên một thiết bị thật.

### 7.9.2. Các yêu cầu về phần cứng và phần mềm cho việc phát triển ứng dụng Bluetooth với Series 60 SDK :

\* Để có thể phát triển ứng dụng Bluetooth cho series 60, bạn cần phải có một máy tính để bàn sử dụng hệ điều hành Windows 2000 hoặc Windows XP, cùng với bộ Series 60 SDK được cài đặt.

\* Về phần cứng Bluetooth, bộ giả lập hỗ trợ phần cứng Bluetooth với phiên bản của Host Controller Interface (HCI) là BCSP và H4 UART của, với tốc độ baud-rate là 115.2 kbps, giao tiếp thông qua giao diện cổng COM của Windows, vì vậy, chỉ có thể sử dụng các phần cứng Bluetooth tương thích với BCSP và H4 UART mà thôi. Bộ giả lập đã chứa đựng các chồng giao thức Bluetooth , vì vậy, thiết bị Bluetooth không cần thêm bất cứ một phần mềm điều khiển nào nữa.

\* Thiết bị phần cứng Bluetooth với giao diện USB không được bộ SDK hỗ trợ, vì vậy, để có thể sử dụng USB Bluetooth với máy ảo, cần phải có driver chuyển đổi USB Bluetooth để phù hợp với máy ảo. Hiện nay chỉ có driver của hãng Cyberabi ([www.cyberabi.com](http://www.cyberabi.com)) cho phép sự chuyển đổi đó :

- Nếu bạn sử dụng Series 60 SDK v1.2, bạn cần có driver DTL\_X .
- Nếu bạn sử dụng Series 60 SDK v2.1, bạn cần phải sử dụng driver BH4\_X.

\* Trong luận văn này, chúng em chỉ xin hướng dẫn cách sử dụng với thiết bị phần cứng Bluetooth là USB Bluetooth Dongle.

### 7.9.3. Cài đặt và cấu hình thiết bị USB Bluetooth.

+ Để cấu hình Bluetooth USB để có thể giả lập thiết bị Bluetooth với máy ảo, trước hết bạn cần phải có Driver DTL\_X hoặc BH4\_X của Cyberabi.

+ Gỡ bỏ tất cả các driver của thiết bị Bluetooth USB đã cài đặt trên máy tính, gỡ thiết bị USB Bluetooth ra khỏi máy tính, khởi động lại máy tính.

+ Gắn USB Bluetooth vào cổng USB của máy tính. Khi đó máy tính sẽ nhận ra thiết bị Bluetooth USB mới cắm vào và yêu cầu chọn driver cho thiết bị đó. Ta chọn đường dẫn đến thư mục có chứa driver DTL\_X hoặc BH4\_X.

+ Sau khi cài driver cho thiết bị Bluetooth xong, một COM port mới là được tạo ra. Đây chính là Bluetooth virtual COM port. Ta sẽ sử dụng COM port này cho máy ảo. Khi đó, trong Device Manager của Windows, ta sẽ thấy được một cổng COM mới tạo ra như sau :



Driver BH4-X

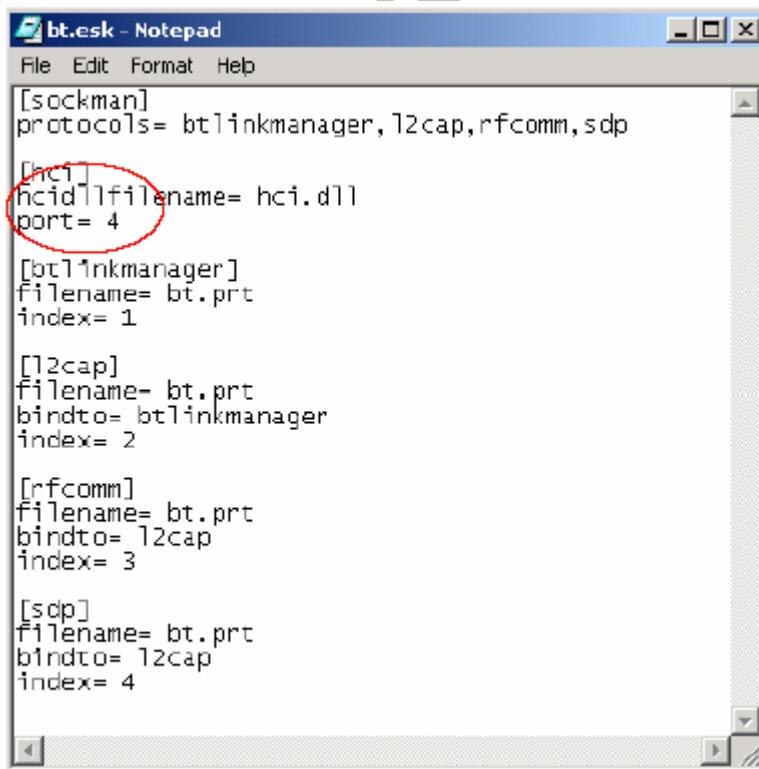
Driver DTL\_X

Hình 7-7 Virtual Bluetooth COM port tạo ra trên máy tính.

- + Sau khi cổng Bluetooth COM được tạo ra, thực hiện cấu hình Bluetooth cho thiết bị giả lập như sau :

Với bộ SDK v1.2 : Mở file ....\Epoc32\Wins\c\system\data\bt.esk

Và thay đổi cấu hình file bt.esk như sau :



Hình 7-8 Cấu hình Bluetooth COM port cho thiết bị giả lập

Thay con số ở vị trí port thành con số của Bluetooth port mà ta vừa tạo ra. Lưu ý là con số mà ta nhập vào đây phải nhỏ hơn con số của Bluetooth port một đơn vị, nghĩa là nếu Bluetooth port là COM4 thì trong file bt.esk ta thay port thành 3.

Lưu ý là thiết bị giả lập chỉ cho phép sử dụng với các cổng COM từ COM2 tới COM6, nếu sau khi cài driver DTL\_X hoặc BH4\_X mà tạo ra cổng COMx với  $x < 1$  hoặc  $x > 6$ , ta phải thiết lập lại số của cổng cho phù hợp.

Bây giờ, thiết bị giả lập đã sẵn sàng cho việc xây dựng và kiểm thử các ứng dụng Bluetooth.

## **Phần 3 XÂY DỰNG ỨNG DỤNG MINH HỌA**

### **SỬ DỤNG CÔNG NGHỆ BLUETOOTH**

Xây dựng chương trình trao đổi phonebook giữa điện thoại di động với máy tính, và với điện thoại di động thông qua Bluetooth.

- ❖ **Chương 8. Phân tích và thiết kế ứng dụng trao đổi phonebook qua Bluetooth.**
- ❖ **Chương 9. Cài đặt và thử nghiệm.**
- ❖ **Chương 10. Tổng kết**

## **Chương 8 PHÂN TÍCH VÀ THIẾT KẾ ỨNG DỤNG TRAO ĐỔI PHONEBOOK**

### **8.1. Giới thiệu**

Ứng dụng PbkExchange được xây dựng nhằm hỗ trợ cho việc quản lý và trao đổi danh bạ điện thoại của người dùng điện thoại di động. Người dùng có thẻ thông Bluetooth để trao đổi danh bạ điện thoại giữa hai điện thoại hoặc để lưu và hoặc phục hồi danh bạ điện thoại bằng cách lưu ra máy tính.

Ứng dụng PbkExchange được phát triển cho các điện thoại sử dụng hệ điều hành Symbian trên nền hệ thống Series 60 và đã được cài đặt thử nghiệm thực tế trên điện thoại Nokia 6600, Nokia 3230. Do sử dụng bộ công cụ phát triển ứng dụng là Series 60 SDK v1.2 hỗ trợ các điện thoại thuộc Series 60 trên nền hệ điều hành Symbian 6.1, vì vậy chương trình PbkExchange này cũng có thể chạy trên các loại điện thoại thuộc series 60 khác như Nokia 7610, Nokia 3360, N\_Gage...

### **8.2. Phân tích và xác định yêu cầu**

Mục đích chính của ứng dụng là thực hiện các kết nối và trao đổi dữ liệu qua Bluetooth, cho phép người dùng trao đổi phonebook (số danh bạ trên điện thoại) giữa hai điện thoại di động thuộc Series 60, hoặc trao đổi phonebook giữa điện thoại và máy tính để lưu trữ trên máy tính.

Ứng dụng PbkExchange gồm hai phần : phần ứng dụng chạy trên điện thoại và phần ứng dụng chạy trên máy tính.

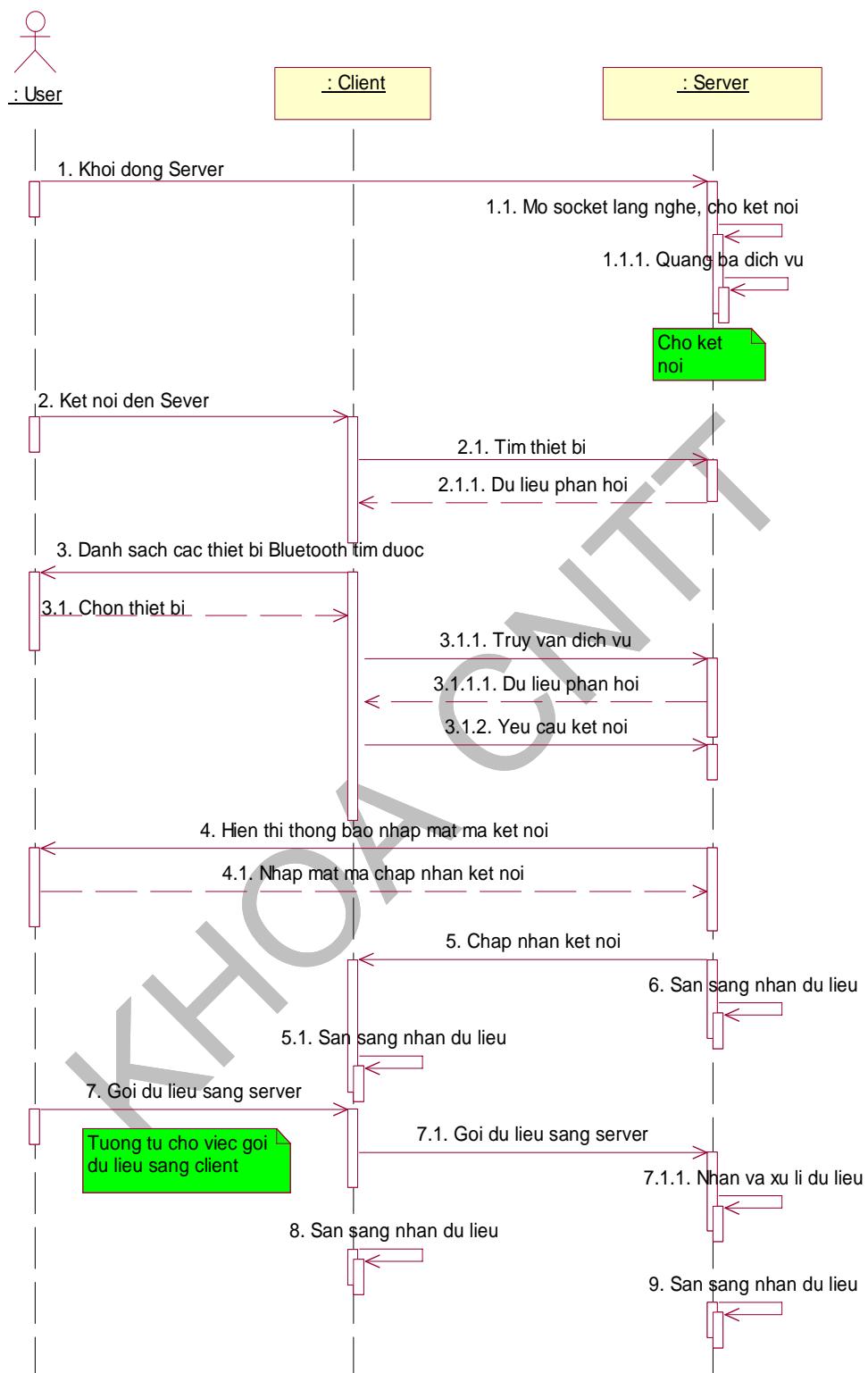
Các yêu cầu của ứng dụng:

- + Thực hiện các kết nối Bluetooth.
- + Trao đổi số danh bạ giữa điện thoại và máy tính
- + Trao đổi số danh bạ giữa hai điện thoại.
- + Thực hiện các thao tác quản lý số danh bạ : thêm, xóa, sửa các phần tử.

### **8.3. Qui trình kết nối và gửi nhận dữ liệu**

Trong một phiên kết nối Bluetooth điểm nối điểm giữa hai thiết bị Bluetooth thông qua giao thức Serial port , thường sẽ có một thiết bị khởi tạo Bluetooth trước và thực hiện quảng bá dịch vụ Bluetooth Serial Port tới thiết bị Bluetooth khác, sau đó lắng nghe các yêu cầu kết nối từ thiết bị khác, ta gọi đó là Server, thiết bị còn lại sẽ thực hiện việc tìm kiếm các thiết bị Bluetooth xung quanh, chọn thiết bị cần kết nối và gửi yêu cầu kết nối tới thiết bị là server, ta gọi thiết bị đó là client.

Qui trình thực hiện kết nối và truyền nhận dữ liệu giữa hai thiết bị được thể hiện như sơ đồ UML sau :



Hình 8-1 Qui trình kết nối và gửi nhận dữ liệu

#### 8.4. Xây dựng phần ứng dụng trên điện thoại

\* Phần ứng dụng PbkExchange trên điện thoại có hai vai trò : Server và Client :

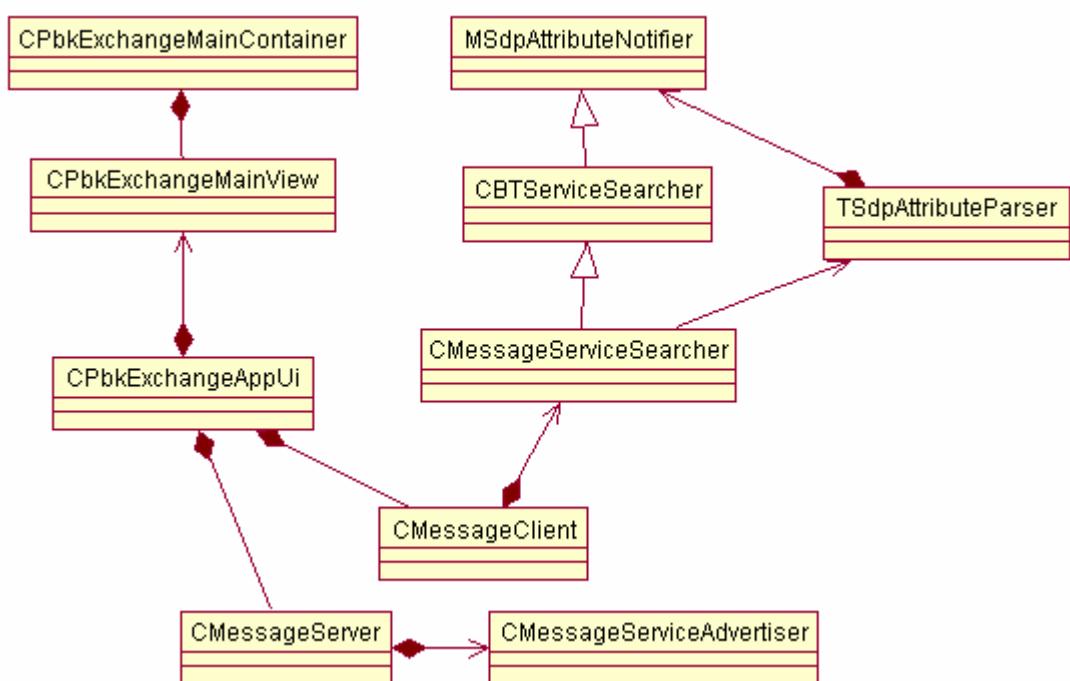
+ Khi thực hiện kết nối với máy tính : điện thoại đóng vai trò client : nó sẽ gửi yêu cầu kết nối tới máy tính, và máy tính đóng vai trò Server.

+ Khi thực hiện kết nối giữa hai điện thoại : một điện thoại sẽ đóng vai trò server : nó sẽ khởi tạo Bluetooth và lắng nghe yêu cầu kết nối từ điện thoại khác, một điện thoại sẽ đóng vai trò client : gửi yêu cầu kết nối tới server.

\* Các chức năng của phần ứng dụng trên điện thoại :

- Cho phép thực hiện các thao tác trên phonebook của điện thoại: thêm xóa, sửa các contact.
- Thực hiện khởi tạo và thiết lập các kết nối Bluetooth
- Sau khi đã thực hiện kết nối thành công, cả server và client đều có khả năng gửi và nhận dữ liệu : thực hiện việc trao đổi phonebook thông qua Bluetooth.
- Xử lý dữ liệu nhận được.

Các lớp chính của PbkExchange phần trên điện thoại :



Hình 8-2 Sơ đồ lớp của phần ứng dụng trên điện thoại.

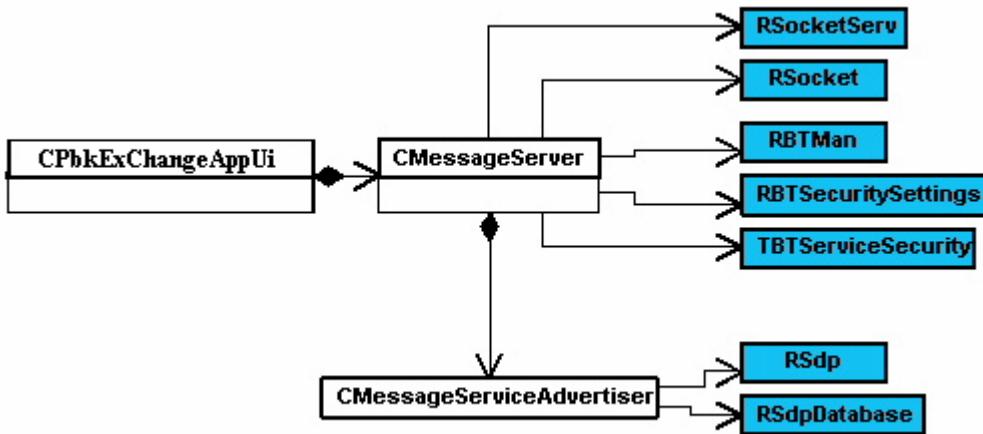
STT	Tên lớp	Chức năng
1	CPbkExchangeAppUi	Đây là lớp xử lý chính của chương trình, là lớp nhận và xử lý các sự kiện từ người dùng, lớp này cũng có nhiệm vụ giao tiếp với các lớp gửi và nhận dữ liệu qua Bluetooth , thao tác với lớp RFile, xử lý dữ liệu nhận được.
2	CMessageClient	Thực hiện tìm kiếm thiết bị server, gửi yêu cầu kết nối, quản lý kết nối, nhận và gửi dữ liệu khi đóng vai trò là client.
3	CMessageServer	Thực hiện khởi tạo Bluetooth, chấp nhận kết nối và thực hiện các thao tác trao đổi dữ liệu, quản lý kết nối.
4	CPbkExchangeMainView	Quản lý các menu và phần giao diện của ứng dụng.
5	CPbkExchangeMainContainer	Hỗ trợ cho lớp CPbkExchangeMainView trong việc quản lý giao diện của ứng dụng.
6	CMessageServiceSearcher, CBTServiceSearcher	Tìm thiết bị và dịch vụ Bluetooth, lấy Port hỗ trợ CMessageClient thực hiện kết nối
7	MSdpAttributeNotifier, TSdpAttributeParser	Hỗ trợ việc lấy, phân tích thuộc tính các record của dịch vụ
8	CMessageServiceAdvertiser	Hỗ trợ lớp CMessageServer trong việc quảng bá dịch vụ Bluetooth Serial Port.

Hình 8-3 Mô tả chức năng các lớp của phần ứng dụng trên điện thoại.

#### 8.4.1. Phần Server

Khi một điện thoại đóng vai trò server, nó lắng nghe và chấp nhận kết nối từ client khác : điện thoại đó sẽ phải khởi tạo Bluetooth trước và thực hiện quảng bá dịch vụ Bluetooth Serial Port tới các thiết bị khác.

Sơ đồ lớp khi thiết bị đóng vai trò server:



Hình 8-4 Sơ đồ lớp của phần ứng dụng trên điện thoại (Server)

\* Lớp **CPbkExchangeAppUi** là lớp thể hiện giao diện người dùng của ứng dụng, nó sử dụng một thể hiện của lớp **CMessageServer** để :

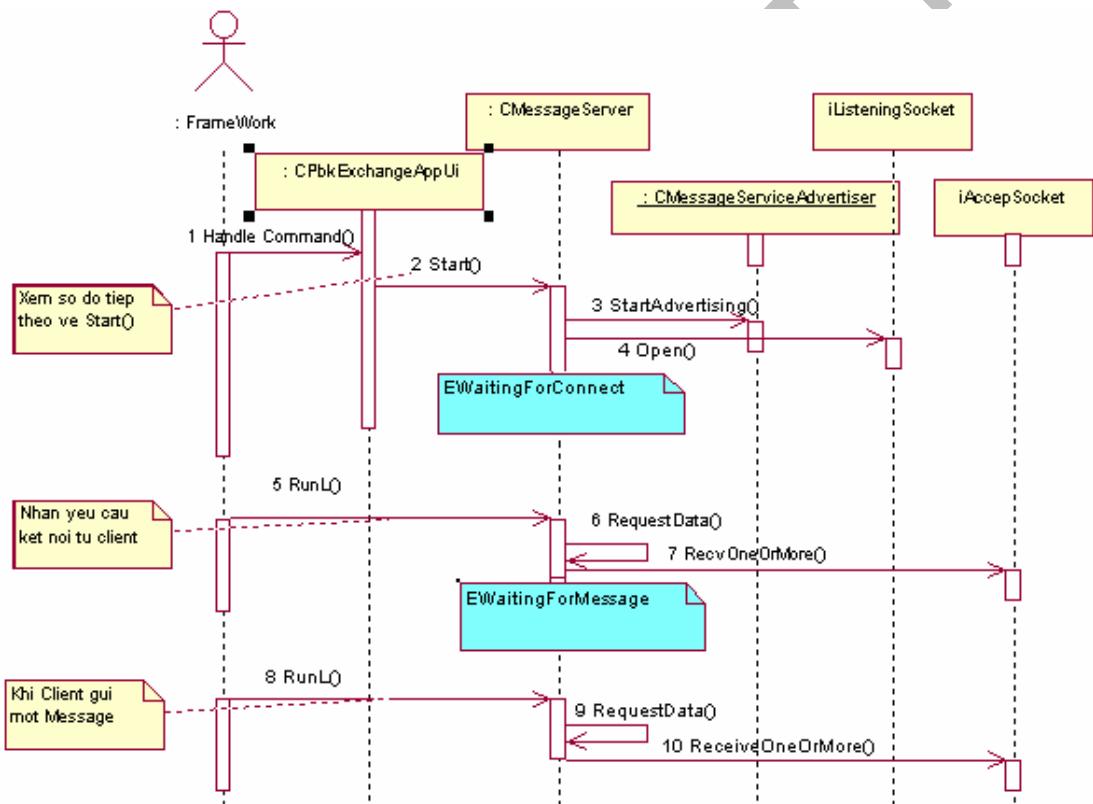
- + Khởi tạo dịch vụ Bluetooth Serial Port
- + Cho phép sự kết nối của các thiết bị Bluetooth khác.
- + Quảng bá dịch vụ Bluetooth Serial Port.
- + Chấp nhận một kết nối socket.
- + Thực hiện các thao tác gửi và nhận dữ liệu qua Bluetooth.

\* Lớp **CMessageServer** sử dụng một số lớp của hệ điều hành Symbian sau :

- + Lớp **RSocketServ** : Để giao tiếp với socket server của thiết bị.
- + Lớp **RSocket** : Cung cấp các hàm cho việc tạo socket, đọc và ghi dữ liệu qua socket. Có hai thể hiện của lớp **RSocket** :
  - **iListeningSocket** : lắng nghe các yêu cầu kết nối từ client.
  - **iAcceptSocket** : Socket kết nối với client : thực hiện việc truyền và nhận dữ liệu thông qua socket này.

- + Lớp RBTMan và RBTSecuritySettings : tạo ra một phiên làm việc (session) tới Bluetooth Security Manager (BSM), dùng để đăng kí và hủy bỏ các đăng kí các dịch vụ với BSM. Các thiết lập về an toàn của một dịch vụ được chứa trong cấu trúc TBTServiceSecurity.
- + Lớp **CMassageServiceAdvertiser** được lớp CMassageServer sử dụng để quảng bá dịch vụ Bluetooth Serial Port. Lớp này sử dụng hai lớp của Symbian là RSdp và RSdpDatabase để tạo một phiên làm việc tới Bluetooth SDP Database. Việc thực hiện quảng bá dịch vụ Bluetooth Serial Port được thực hiện bằng cách tạo một record thích hợp trong SDP Database.

\* Sơ đồ UML sau thể hiện quá trình quảng bá dịch vụ của server :



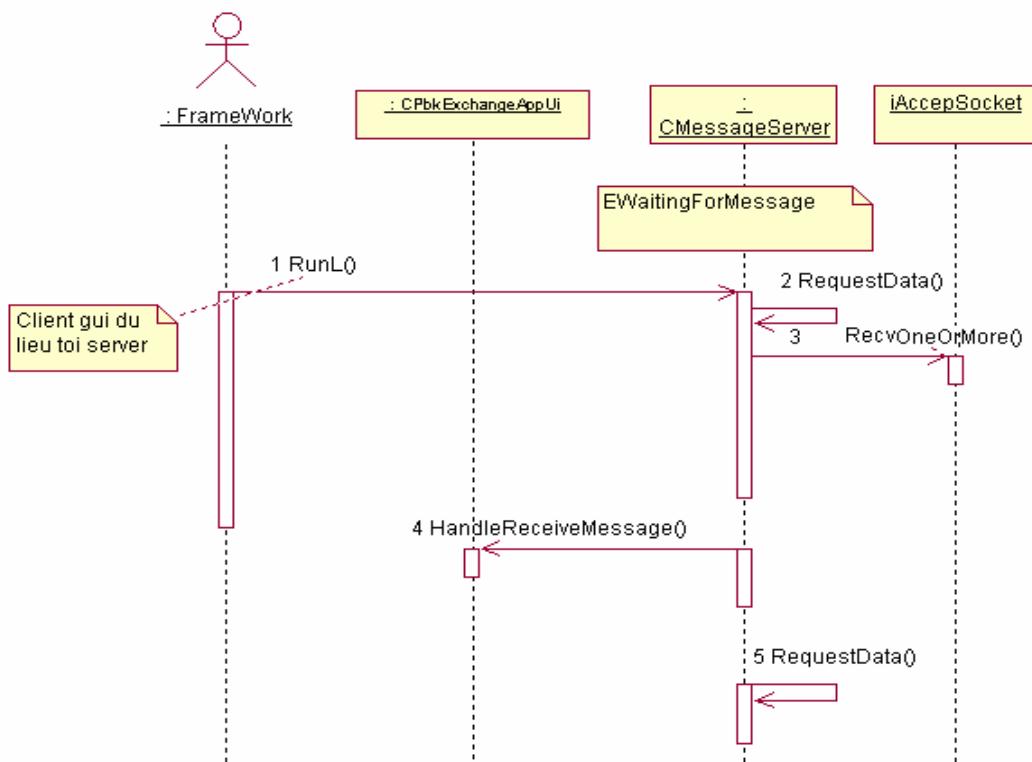
Hình 8-5 Quảng bá dịch vụ của Server

Hàm	Mô tả
1 - 2	Người dùng chọn vào menu <b>StartListen</b> trên thiết bị, lúc này hàm HandleCommandL của lớp CPbkExchangeAppUi được gọi, và tiếp đó hàm <b>StartL()</b> của lớp <b>CMassageServer</b> được gọi.

	Hàm <b>StartL()</b> sẽ làm nhiệm vụ tìm kiếm một port để lắng nghe, thiết lập các chế độ an toàn trên đó, và mở port đó để lắng nghe.
3 - 4	StartL() gọi hàm <b>StartAdvertisingL()</b> của lớp CMessageAdvertiser để quảng bá dịch vụ Bluetooth Serial Port tới các thiết bị khác. Biến trạng thái iState của CMessageServer mang giá trị : EWaitingForConnect
5-7	Thiết lập một kết nối với thiết bị client khi nhận được yêu cầu kết nối. Hàm <b>RunL()</b> của <b>CMessageServer</b> được gọi, biến trạng thái iState chuyển từ EWaitingForConnect thành EWaitingForMessage
8-10	Nhận dữ liệu từ Client, hàm <b>requestData()</b> , và hàm <b>ReceiveOneOrMore()</b> được gọi để nhận dữ liệu khi được gửi tới.

Bảng 3-1 Mô tả các hàm quảng bá dịch vụ

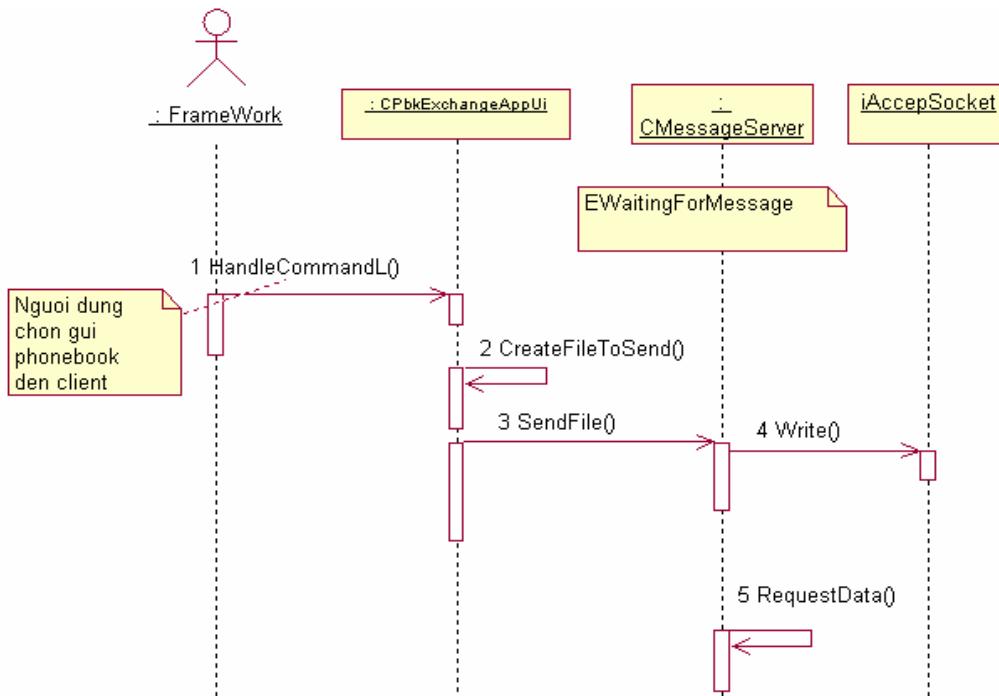
\* Sơ đồ UML sau thể hiện việc nhận dữ liệu từ Client :



Hình 8-6 Nhận dữ liệu từ Client

Sau khi đã thực hiện kết nối với client, Server luôn ở trạng thái sẵn sàng nhận dữ liệu từ client. Khi client thực hiện truyền dữ liệu, hàm RequestData() trên server được gọi, và hàm RecvOneOrMore() được gọi để nhận dữ liệu từ socket. Dữ liệu nhận được được truyền qua cho lớp CPbkExchangeAppUi xử lý.

\* Thực hiện truyền Phonebook tới Client :

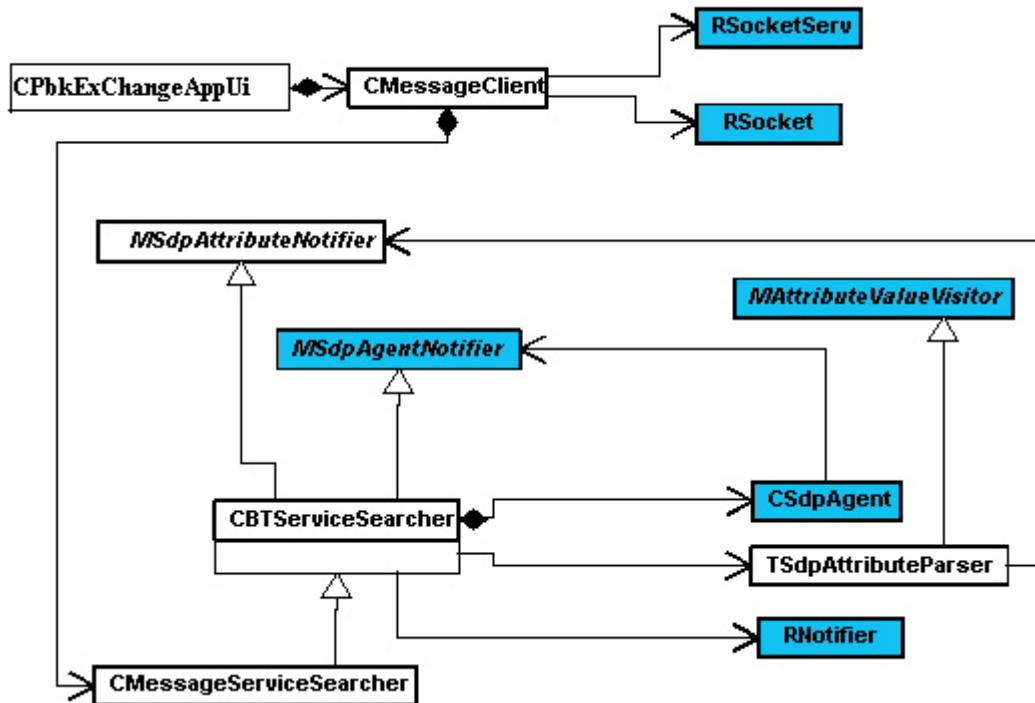


Hình 8-7 Truyền dữ liệu phonebook tới client

#### 8.4.2. Phần Client

Ứng dụng trên điện thoại sẽ đóng vai trò Client khi nó gửi yêu cầu kết nối tới máy tính hoặc tới điện thoại khác. Lúc này, nó sẽ phải thực hiện việc tìm kiếm thiết bị Bluetooth xung quanh, chọn thiết bị kết nối tới và thực hiện kết nối.

Sơ đồ lớp thể hiện khi thiết bị đóng vai trò là Client :

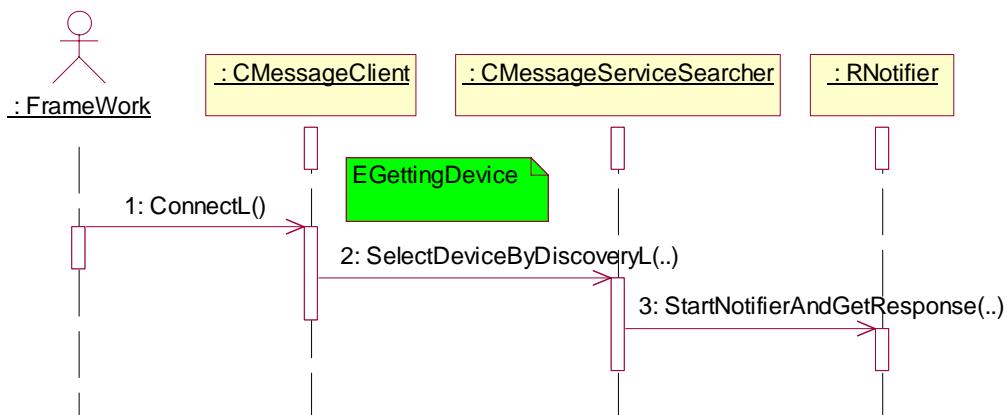


Hình 8-8 Sơ đồ lớp của phần ứng dụng trên điện thoại (Client)

\* Lớp **CPbkExchangeAppUi** là lớp thể hiện giao diện người dùng của ứng dụng, nó sử dụng một thể hiện của lớp **CMessageClient** để thực hiện kết nối Bluetooth với Server và thực hiện các trao đổi dữ liệu thông qua Bluetooth với Server. Lớp **CMessageClient** sử dụng lớp **RSocketServ** và **RSocket** để mở một socket để truyền và nhận dữ liệu.

Trước khi có thể mở một socket, đối tượng **CMessageClient** phải thực hiện việc tìm kiếm thiết bị Bluetooth server, truy vấn dịch vụ có trên server xem có cung cấp dịch vụ là Bluetooth Serial Port không và thực hiện kết nối với Server.

\* Thực hiện tìm kiếm thiết bị:

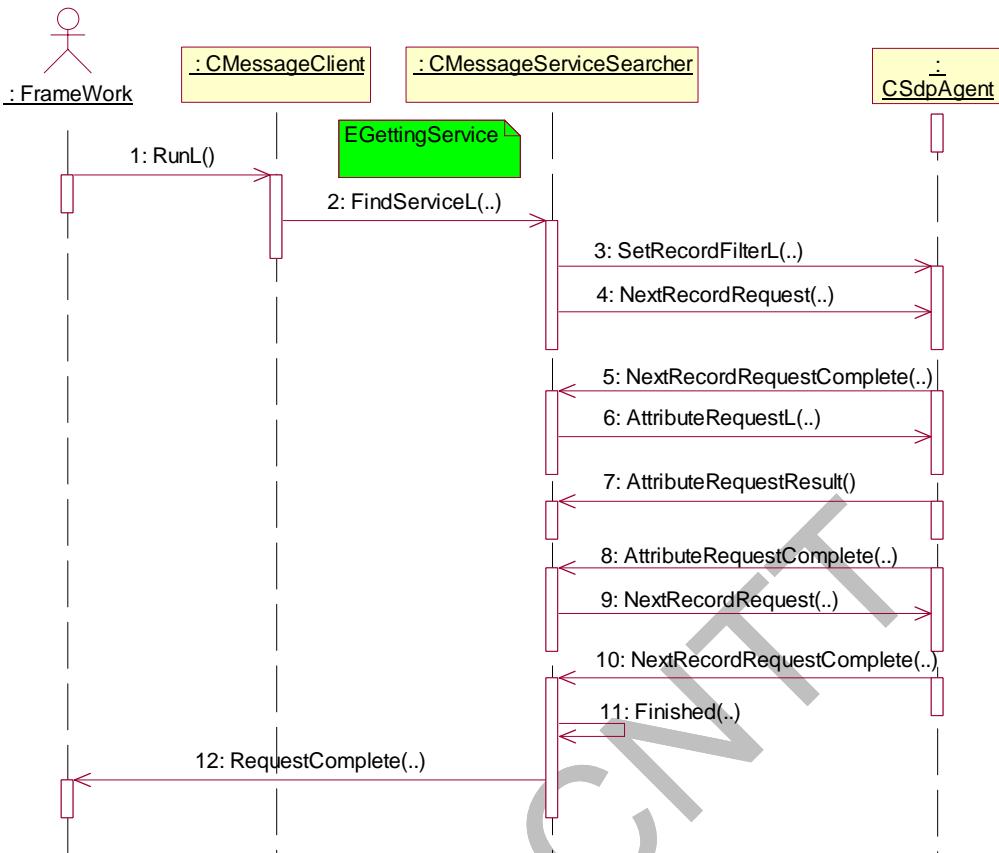


Hình 8-9 Sơ đồ tìm kiếm thiết bị

Hàm	Mô tả
1 – 3	Hàm ConnectL của đối tượng CMessageClient được gọi, nó thiết lập biến trạng thái (iState) là EGettingDevice và gọi hàm SelectDeviceByDiscoveryL của đối tượng CMessageServiceSearcher. Và hàm này gọi hàm StartNotifierAndGetResponse của đối tượng RNotifier để tìm và chọn thiết bị mà nó nhận được. Sau khi người dùng chọn thiết bị xong hàm RunL của đối tượng CMessageClient sẽ được gọi.

Bảng 3-2 Mô tả các hàm tìm thiết bị

\* Thực hiện truy vấn dịch vụ



Hình 8-10 Sơ đồ UML truy vấn dịch vụ trên thiết bị

Hàm	Mô tả
1 – 2	Sau khi người dùng chọn thiết bị muốn kết nối thì hàm RunL sẽ được gọi, lúc này biến trạng thái iState là EGettingService, và gọi hàm FindServiceL của đối tượng CMessageServiceSearcher để tìm Serial Port service record trong SDP database của thiết bị muốn kết nối .
3 – 4	FindServiceL thiết lập một bộ lọc để chỉ nhận những Serial Port service record bằng cách gọi hàm SetRecordFilter của đối tượng CSdpAgent. Sau đó NextRecordRequest được gọi để tìm Serial Port record trong SDP database của thiết bị muốn kết nối.
5 – 6	Khi một record được tìm thấy, đối tượng CSdpAgent gọi hàm callback NextRecordRequestComplete, hàm này gọi hàm NextRecordRequestCompleteL của đối tượng

	CMessageServiceSearcher, đến lượt mình hàm này lại gọi hàm AttributeRequestL của đối tượng CSdpAgent để yêu cầu thuộc tính đầu tiên của record.
7	Khi đối tượng CSdpAgent nhận được một thuộc tính nó gọi hàm callback AttributeRequestResult, đến lượt mình hàm này gọi hàm AttributeRequestResultL của đối tượng CMessageServiceSearcher. Một bộ phân tích sẽ phân tích các thuộc tính này. Nếu những thuộc tính này là Protocol Descriptor List, bộ phân tích sẽ lấy channel (port) và gán nó vào biến thành viên iPort của đối tượng CMessageServiceSearcher.
8 – 12	Khi tất cả các thuộc tính được tìm thấy, hàm AttributeRequestCompleteL sẽ được gọi và hàm này sẽ gọi hàm NextRecordRequest trong trường hợp có record Serial Port service khác trong SDP database. Nếu không có thêm record nào thì đối tượng CSdpAgent sẽ gọi hàm NextRecordRequestComplete và cờ EoF sẽ được thiết lập. Tiếp theo, hàm Finished sẽ được gọi và hàm này gọi hàm RequestComplete, hoàn tất hàm này hàm RunL của đối tượng CMessageClient sẽ được gọi.

Bảng 3-3 Mô tả các hàm truy vấn dịch vụ

## 8.5. Xây dựng phần ứng dụng PbkExchange trên máy tính

Ứng dụng PbkExchange trên máy tính đóng vai trò Server trong quá trình kết nối và trao đổi dữ liệu. Ứng dụng trên Server sẽ khởi tạo Bluetooth, quảng bá dịch vụ Bluetooth Serial Port tới các thiết bị khác, và chấp nhận kết nối với client khi nhận được yêu cầu kết nối. Khi đã kết nối thành công, quá trình trao đổi dữ liệu có thể được diễn ra.

Các bước xây dựng server:

### 8.5.1. Kết nối vào cổng COM :

Thay vì mở Socket để lắng nghe chờ kết nối như trên di động thì trên máy tính ta chỉ cần mở một cổng COM đã được driver của thiết bị định sẵn, là

cổng COM mà thiết bị Bluetooth sẽ lắng nghe kết nối từ client. Việc này được thực hiện một cách đơn giản như sau:

```
sprintf(sCommPortBuf, "\\\.\%s", sCommPort);
m_hBluetoothHandle = CreateFile(sCommPortBuf,
    GENERIC_READ | GENERIC_WRITE,
    FILE_SHARE_READ | FILE_SHARE_WRITE,
    NULL, // no security
    OPEN_EXISTING,
    0, // not overlapped I/O
    NULL);
```

sCommPort là kiểu chuỗi, ví dụ “COM5”

m\_hBluetoothHandle kiểu HANDLE

Hàm CreateFile mở cổng COM với port là 5 và trả về Handle của cổng COM.

### 8.5.2. Quảng bá dịch vụ

Việc quảng bá dịch vụ Bluetooth Serial Port được driver của thiết bị Bluetooth thực hiện, ứng dụng Bluetooth chỉ cần kết nối vào COM port mà driver đã chỉ định sẵn là có thể chấp nhận các yêu cầu kết nối nào từ client.

### 8.5.3. Chấp nhận kết nối

Khi có Client yêu cầu kết nối thì Server sẽ tự động chấp nhận kết nối

### 8.5.4. Thực hiện truyền và nhận dữ liệu :

Việc truyền và nhận dữ liệu của ứng dụng trên máy tính thực ra là việc đọc và ghi dữ liệu ra cổng COM mà ứng dụng kết nối vào. Để làm việc đó, ta sử dụng các hàm đọc và ghi dữ liệu sau :

```
BOOL ReadFile(
    HANDLE hFile,
    LPVOID lpBuffer,
    DWORD nNumberOfBytesToRead,
    LPDWORD lpNumberOfBytesRead,
```

```
LPOVERLAPPED lpOverlapped
```

```
);
```

```
BOOL WriteFile(  
    HANDLE hFile,  
    LPCVOID lpBuffer,  
    DWORD nNumberOfBytesToWrite,  
    LPDWORD lpNumberOfBytesWritten,  
    LPOVERLAPPED lpOverlapped  
);
```

Tên tham số	Mô tả
hFile	HANDLE của cổng COM đã được mở với hàm CreateFile như trên (m_hBluetoothHandle)
lpBuffer	Con trỏ trả về buffer chứa dữ liệu để truyền đi hoặc nhận được.
nNumberOfBytesToRead, nNumberOfBytesToWrite	Số byte dữ liệu sẽ nhận hoặc truyền
lpNumberOfBytesRead, lpNumberOfBytesWritten	Số byte thực tế nhận hoặc truyền đi được.
lpOverlapped	Là một con trỏ trả về cấu trúc OVERLAPPED, do cổng COM được mở với hàm CreateFile với tham số thứ 6 bằng 0 (không sử dụng Overlapped) do đó tham số này bằng NULL.

Bảng 3-4 Tham số hàm ReadFile và WriteFile

## **Chương 9 CÀI ĐẶT VÀ THỬ NGHIỆM**

### **9.1. Cài đặt:**

\* Ứng dụng PbkExchange được xây dựng với môi trường phát triển ứng dụng sau:

- Môi trường cài đặt ứng dụng : Windows XP Professional Service pack 2
- Môi trường lập trình : Microsoft Visual C++ 6.0
- Bộ công cụ phát triển ứng dụng Series 60 SDK v1.2 hỗ trợ Microsoft Visual C++ 6.0
- Công cụ Rational Rose

\* Phần cứng Bluetooth :

- Phần ứng dụng trên máy tính : Bluetooth USB.
- Phần ứng dụng trên điện thoại :
  - Giả lập máy ảo với Bluetooth là USB Bluetooth , và phần mềm DTL\_X driver.
  - Kiểm thử với Nokia 6600, Nokia 3230

### **9.2. Thử nghiệm**

Ứng dụng PbkExchange được thử nghiệm trên máy ảo của Series 60 SDK v1.2 và trên điện thoại Nokia 6600 và Nokia 3230

➡ Thử nghiệm trên máy ảo :

- Thử nghiệm kết nối với Nokia 6600 :
  - Kết nối tốt, khá ổn định
  - Gửi nhận dữ liệu tốt.
  - Tốc độ xử lý dữ liệu nhanh.
- Hiển thị hình ảnh (hình nền) : khá nhanh, gần như tức thời.

➡ Thử nghiệm trên điện thoại Nokia 6600, và Nokia 3230 :

- Kết nối với PC :
  - Tốc độ kết nối nhanh, ổn định

- Gửi nhận dữ liệu tốt.
- Tốc độ xử lý dữ liệu chậm hơn máy ảo.
- Kết nối giữa hai điện thoại :
  - Tốc độ kết nối : khá nhanh, ổn định
  - Gửi nhận dữ liệu : tốt.
  - Tốc độ xử lý dữ liệu chậm hơn máy ảo
- Hiển thị hình ảnh (hình nền) : chậm hơn máy ảo, khoảng 2 giây.

Chương trình chạy tốt trên Nokia 6600, Nokia 3230 và được xây dựng bằng bộ Series 60 SDK v1.2, do đó có thể chạy trên các loại điện thoại thuộc Series 60 khác.

## Chương 10 TỔNG KẾT

Với mong muốn tìm hiểu công nghệ, kĩ thuật mới và được sự phân công và hướng dẫn của cô Huỳnh Thụy Bảo Trân, chúng em đã hoàn thành luận văn cử nhân công nghệ thông tin với đề tài “Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa”.

Sau khi thực hiện đề tài, chúng em đã đạt được một số kết quả sau :

- Tìm hiểu được công nghệ Bluetooth, một công nghệ không dây đang phát triển rất mạnh và có tầm ứng dụng rộng rãi hiện nay, năm được cách thức hoạt động, các đặc điểm kĩ thuật và khả năng của công nghệ Bluetooth.Thêm vào đó, trong quá trình tìm hiểu về Bluetooth, chúng em cũng nắm được một số kĩ thuật mạng không dây khác.
- Tìm hiểu được một hệ điều hành dành cho điện thoại di động thông minh phổ biến nhất hiện nay, đó là hệ điều hành Symbian, biết được sơ lược về cấu trúc của hệ điều hành Symbian, và cách xây dựng ứng dụng trên Symbian. Hiện nay, Symbian đang ngày càng phát triển mạnh mẽ và mở ra một môi trường lập trình mới đầy tiềm năng cho các lập trình viên : lập trình ứng dụng cho điện thoại thông minh.

- Để minh họa cho việc sử dụng công nghệ Bluetooth, chúng em đã xây dựng được một ứng dụng thực hiện việc trao đổi phonebook giữa hai điện thoại di động Series 60, và trao đổi phonebook giữa điện thoại di động Series 60 và máy tính. Qua đó, chúng em nắm được việc xây dựng ứng dụng sử dụng giao tiếp Bluetooth giữa các thiết bị là điện thoại Symbian và máy tính.

Mặc dù đã cố gắng hết sức, nhưng do thời gian có hạn, và việc tìm hiểu công nghệ mới cũng gặp phải nhiều khó khăn do không có nhiều tài liệu và thời gian tìm hiểu, vì vậy, ứng dụng của chúng em chỉ mang tính minh họa cho việc sử dụng công nghệ mà thôi.

## PHỤ LỤC A : Một số thuật ngữ sử dụng trong luận văn

- ISM (Industrial, Scientific, Medical): dãy tầng 2.40- 2.48 GHz, dãy băng tầng không cần đăng ký được dành riêng để dùng cho các thiết bị không dây trong công nghiệp, khoa học, y tế.
- SIG (Special Interest Group): nhóm nghiên cứu SIG chính thức được thành lập với mục đích phát triển công nghệ Bluetooth trên thị trường viễn thông. Bất kỳ công ty nào có kế hoạch sử dụng công nghệ Bluetooth đều có thể tham gia vào.
- CSR (Cambridge Silicon Radio): một nơi nghiên cứu chế tạo chip Bluetooth.
- CES (Consumer Electronics Show): hội nghị về các sản phẩm điện tử tổ chức ở Las Vegas, Mỹ.
- OBEX (OBject EXchange protocol): chuẩn đồng bộ hóa dữ liệu cho PDA.
- MAC (Media Access Control): điều khiển truy cập truyền thông.
- AMA (Active Member Address) : địa chỉ 3 bit dành cho thiết bị đang hoạt động trong piconet.
- PMA (Packed Member Address) : con số 8 bits để phân biệt các packed Slave với nhau và có tối đa 255 thiết bị ở trạng thái này trong 1 Piconet

- ACL (Asynchronous connectionless) : phi kết nối bất đồng bộ, dành cho truyền dữ liệu.
- SCO (Synchronous connection-oriented): kết nối đồng bộ có định hướng.
- CRC (Cyclic Redundancy Check): gói kiểm lỗi theo chu kỳ.
- FCC( Federal Communications Commission): điều lệ quy định việc sử dụng sóng radio trên thế giới.
- FDMA (Frequency Division Multiple Access): đa truy cập phân chia theo tần số.
- TDMA (Time Division Multiple Access): đa truy cập phân chia theo thời gian.
- CDMA (Code Division Multiple Access): đa truy cập phân chia theo mã.
- DS-CDMA (Direct sequence - Code Division Multiple Access): chuỗi quản lý CDMA.
- TS (timeslot) : khe thời gian.
- FEC (Forward Error Correction) : sửa lỗi trước.
- LMP (Link Management Protocol) : giao thức quản lý kết nối.
- Device Access Code (DAC): mã truy cập thiết bị.
- Channel Access Code (CAC): mã truy cập kênh truyền.
- Inquiry Access Code (IAC): mã truy cập quá trình inquiry.
- HCI (Host Controller Interface): giao diện điều khiển máy chủ.
- SDP (Service Discovery Protocol): giao thức tìm kiếm dịch vụ.
- PDU (protocol data unit): một định dạng gói tin trong SDP.
- UUID (Universal Unique Identifier): số định danh duy nhất dành cho mỗi dịch vụ.
- A2DP (Advanced Audio Distribution Profile)
- AVRCP (Audio/Video Remote Control Profile)
- BIP (Basic Imaging Profile)
- BPP(Basic Printing Profile)
- CIP(Common ISDN Access Profile)
- CTP (Cordless Telephony Profile)
- DUN (Dial-up Networking Profile)
- FAX (Fax Profile)

- FTP (File Transfer Profile)
- GAVDP (General Audio/Video Distribution Profile)
- GAP (Generic Access Profile)
- GOEP (Generic Object Exchange Profile)
- HFP (Hands Free Profile)
- HCRP (Hard Copy Cable Replacement Profile)
- HSP (Headset Profile)
- HID (Human Interface Device Profile)
- ICP (Intercom Profile)
- OPP (Object Push Profile)
- PAN (Personal Area Networking Profile)
- SPP (Serial Port Profile)
- SDAP (Service Discovery Application Profile)
- SAP (SIM Access Profile)
- SYNCH (Synchronisation Profile)
- VDP (Video Distribution Profile)
- HFP 1.5 (Handsfree Profile 1.5)
- UDI (Unrestricted Digital Information)
- WAP (Wireless application Protocol over BT)
- ESDP (Extended Service discovery profile)
- LPP (Local Positioning Profile)
- VCP (Video Conferencing Profile)
- DID (Device ID)
- PPP (Point-To-Point Protocol)
- IEEE (Institute of Electrical and Electronics Engineers)
- AP (access point)
- CSMA/CA (Carrier Sense Multiple Access với Collision Avoidance)
- IrDA (Infrared Data Association)
- WLAN (Wireless LAN)
- SSID (System Set Identifier)
- BD\_ADDR (Bluetooth device address hoặc MAC address)
- AU RAND: thông điệp mời ngẫu nhiên 128 bit

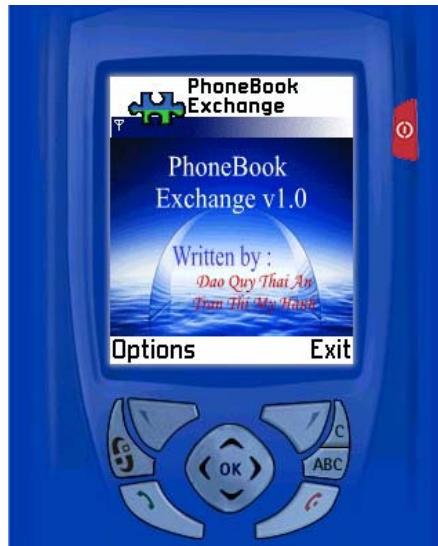
- RNG (Random Number Generator): quy trình tạo số ngẫu nhiên
- FHS: Frequency Hop Synchronization: đồng bộ nhảy tần số.
- LAP: Lower Address Part: phần địa chỉ 24 bit thấp trong BD\_ADDR
- UAP: Upper Address Part: phần địa chỉ 8 bit cao trong BD\_ADDR
- NAP field: non-significant address part: phần địa chỉ 16 bit không quan trọng trong BD\_ADDR
- DoS (Denial of Service): từ chối dịch vụ
- PCMCIA (Personal Computer Memory Card International Association): tổ chức quốc tế về bộ nhớ trong máy tính cá nhân.
- MMS messages: là một loại tin nhắn đa phương tiện được trao đổi giữa các điện thoại Symbian và những điện thoại khác có hỗ trợ MMS.
- Claimant: thiết bị yêu cầu
- Verifier: thiết bị xác minh
- IMEI (International Mobile Equipment Identity)
- AT (Attention) command set: tập lệnh AT, một tiêu chuẩn đối với phần mềm điều khiển modem do hãng Hayes Microcomputer Products soạn thảo và được đưa ra lần đầu tiên dùng với modem Smartmodems.
- PSM (Protocol/Service Multiplexer ports): cổng đa thành phần dành cho dịch vụ hoặc giao thức.
- AFH (*Adaptive Frequency Hopping*): phương pháp chống nhiễu tốt hơn bằng cách nhảy tầng số.
- eSCO (*extended Synchronous Connections*): kết nối đồng bộ mở rộng.
- RSSI (*Received Signal Strength Indicator*): thông báo độ mạnh của tín hiệu nhận được.
- **API** (Application Programming Interface): API là tập các chức năng được cung cấp bởi một hệ thống và tập hợp các API này thể hiện chức năng của hệ thống đó.
- **Cleanup stack**: Một ngăn xếp đặc biệt lưu giữ các biến tự động (biến cục bộ) là các con trỏ trả về vùng nhớ heap để có thể giải phóng các vùng nhớ này khi hàm chứa biến tự động bị thoát do lỗi môi trường
- **Descriptor** : Kiểu dữ liệu trên Symbian, có thể được dùng để biểu diễn chuỗi hoặc dữ liệu nhị phân.

- **EPOC** (Electronic Pocket Communication): Thể hệ cũ của hệ điều hành Symbian. Epoc vẫn còn được sử dụng trong một số phần trên hệ điều hành Symbian như trên Emulator hay cấu trúc thư mục.
- **J2ME** (Java 2 Platform, Micro Edition): Bản phân phối của nền hệ thống Java nhắm đến các thiết bị gia dụng, thiết bị truyền thông nhỏ. Công nghệ J2ME gồm máy ảo Java và tập các API được thiết kế cho môi trường trên các thiết bị này.
- **Leave**: Khả năng ngắt hoạt động tại hàm nơi đó xảy ra lỗi môi trường và chuyển đến phần xử lý lỗi. Các hàm có thể leave có tên hàm kết thúc bằng chữ cái L.
- **Nền hệ thống (Platform)**: Một tập công nghệ được xem như là nền tảng cho các ứng dụng thế giới thực hoặc cho các nền hệ thống cao hơn. Hệ điều hành Symbian là nền hệ thống cho hệ thống giao diện như Series 60 hay UIQ và sự kết hợp Symbian và hệ thống giao diện tạo nền hệ thống cho ứng dụng Symbian.
- **UID** (Unique Identifier) - **Định danh**: số xác định duy nhất cho một loại chương trình hay phân biệt giữa các ứng dụng trong hệ điều hành Symbian. Các giá trị định danh này là duy nhất trên toàn thiết bị dùng hệ điều hành Symbian.

## PHỤ LỤC B : Hướng dẫn sử dụng chương trình PbkExchange

### 1. Sử dụng ứng dụng PbkExchange trên điện thoại :

Giao diện chính của ứng dụng trên điện thoại :



Hình B - 1 Giao diện ứng dụng trên điện thoại.

- Khi thực hiện trao đổi phonebook giữa hai điện thoại, một điện thoại phải đóng vai trò Server : khởi tạo và quảng bá dịch vụ Bluetooth tới các thiết bị khác, một điện thoại đóng vai trò Client: gửi yêu cầu kết nối tới server.
- Khi thực hiện trao đổi phonebook với máy tính, điện thoại đóng vai trò client : gửi yêu cầu kết nối tới máy tính.

\* **Sử dụng Server :**

Để sử dụng chương trình, trước hết dịch vụ Bluetooth trên thiết bị phải được bật lên và thiết lập ở chế độ mà các thiết bị Bluetooth khác có thể tìm thấy được.

Trên menu chính của chương trình chọn vào Start Listen để khởi tạo server, khi đó, dịch vụ Bluetooth Serial Port sẽ được khởi tạo và quảng bá dịch vụ tới các thiết bị khác.



Hình B - 2 Khởi tạo điện thoại là server

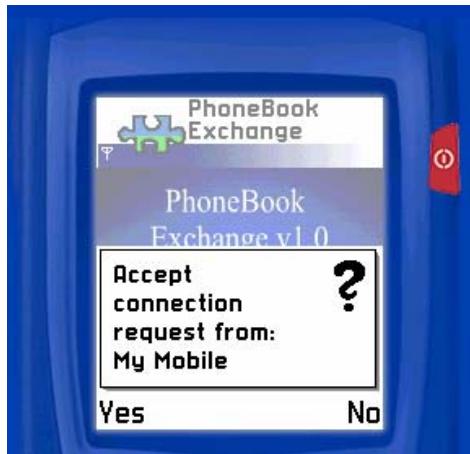
Sau đó, Server sẽ ở trạng thái lắng nghe yêu cầu kết nối từ các điện thoại khác, khi đó menu của ứng dụng sẽ chuyển thành :



Hình B - 3 Trạng thái lắng nghe

Ở đây, người dùng có thể đưa ứng dụng thoát khỏi trạng thái lắng nghe kết nối bằng cách chọn vào mục “Stop Listen”.

Khi có một điện thoại khác gửi yêu cầu kết nối tới, nếu thiết bị đó chưa được cấp phép truy cập vào server, một thông báo sẽ hiện lên yêu cầu người dùng xác nhận sự truy cập :



Hình B - 4 Xác nhận yêu cầu kết nối

Khi người dùng đồng ý kết nối, việc kết nối sẽ được thực hiện. Nếu kết nối thành công, việc trao đổi dữ liệu đã sẵn sàng được thực hiện. Khi đó, menu của ứng dụng sẽ chuyển thành :



Hình B - 5 Menu sau khi kết nối thành công

Người dùng có thể chọn thực hiện các chế độ trao đổi phonebook là : truyền toàn bộ phonebook tới máy được kết nối (chọn mục : Send PhoneBook) hoặc chọn chế độ là chọn các contact sẽ truyền (chọn mục : Select Send Item).

Khi người dùng chọn mục Select Send Item, một dialog sẽ hiện ra cho phép người dùng chọn các contact sẽ truyền. Việc truyền dữ liệu sẽ diễn ra sau khi người dùng nhấn OK.



Hình B - 6 Lựa chọn các contact để truyền

Khi nhận được dữ liệu từ máy kết nối tới, một thông báo sẽ hiện ra cho người dùng xác nhận xem có thực hiện thêm các contact mới nhận vào phonebook hay không. Việc thêm các contact vào phonebook sẽ diễn ra nếu người dùng chọn Yes, nếu chọn No, dữ liệu nhận được sẽ bị hủy bỏ.



Hình B - 7 Sử dụng ứng dụng PbkExchange

Để chấm dứt kết nối, người dùng chọn vào mục “Disconnect” trên menu, khi đó, ứng dụng sẽ thực hiện chấm dứt kết nối và chuyển về menu ban đầu như lúc mới khởi tạo.

\* **Sử dụng Client:**

Sau khi khởi tạo, tại menu của ứng dụng, người dùng chọn vào mục “Connect To Device” :



Hình B - 8 Khởi tạo điện thoại là client

Khi đó, nếu Bluetooth chưa được bật lên, một thông báo sẽ xuất hiện xác nhận xem người dùng có muốn bật Bluetooth lên không. Nếu Bluetooth được bật lên, tiếp đó, ứng dụng sẽ thực hiện việc tìm kiếm thiết bị Bluetooth xung quanh, người dùng sẽ chọn thiết bị để thực hiện kết nối.

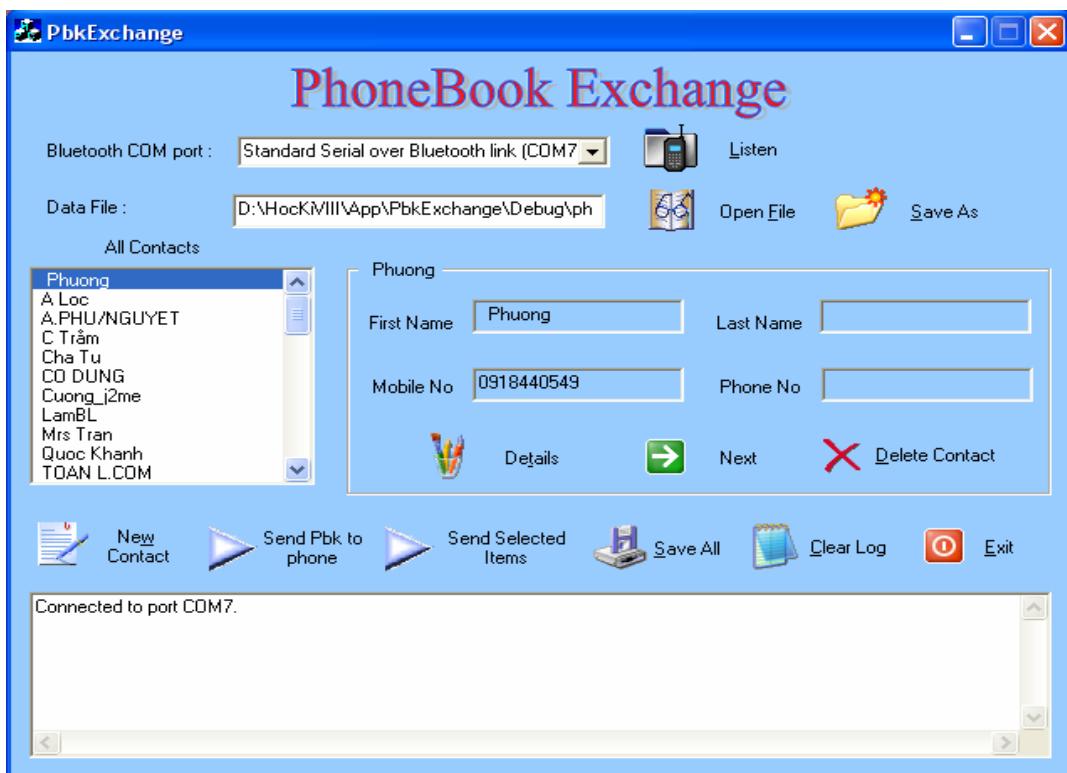


Hình B - 9 Lựa chọn thiết bị để kết nối

Khi đã kết nối thành công, việc trao đổi phonebook được thực hiện giống như đối với Server.

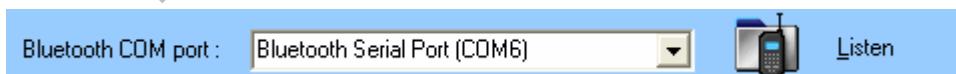
## 2. Sử dụng ứng dụng PbkExchange trên máy tính :

Giao diện của ứng dụng trên điện thoại như sau :



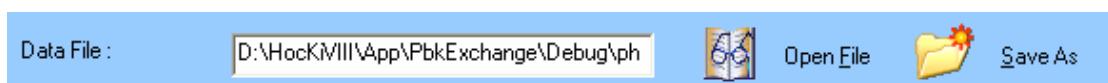
Hình B - 10 Giao diện ứng dụng PbkExchange trên máy tính.

Để có thể kết nối với điện thoại, người dùng phải chọn đúng cổng COM mà driver của USB Bluetooth chỉ định để lắng nghe kết nối từ điện thoại. Các cổng COM có trên máy được liệt kê trong Combo box để cho người dùng chọn, hoặc người dùng có thể nhập vào bằng bàn phím cổng COM để lắng nghe với định dạng : COMx( với x là số hiệu cổng, ví dụ : COM1, COM2 ...). Khi đã chọn đúng cổng, người dùng nhấn vào nút nhấn Listen để kết nối vào cổng COM đó, khi đó, Driver của USB Bluetooth sẽ tự động thiết lập và quảng bá dịch vụ Bluetooth Serial Port và lắng nghe các yêu cầu kết nối từ điện thoại.



Hình B - 11 Combo Box lựa chọn cổng COM

Dữ liệu về phonebook được lưu trong file có đường dẫn trong textbox “Data File”.



Hình B - 12 File dữ liệu

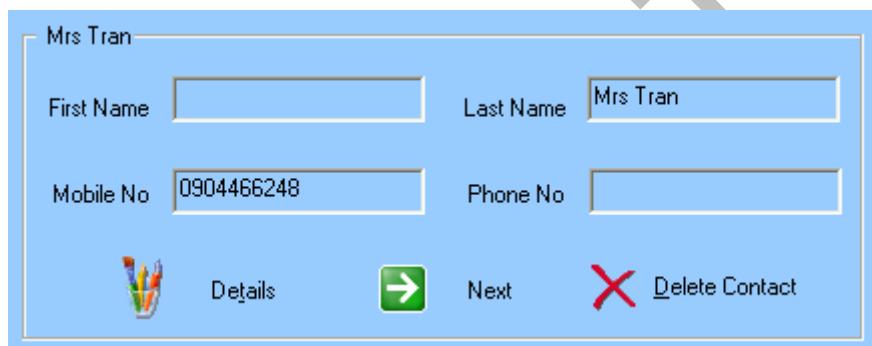
Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa

Các contact của phonebook hiện hành (đang xử lý) được thể hiện trong Listbox :



Hình B - 13 Listbox chứa phonebook hiện hành

Khung chứa sau khi hiện thông tin vắn tắt về contact đang được chọn trong listbox trên :



Hình B - 14 Thông tin sơ lược của một contact

Người dùng có thể xem thông tin chi tiết của contact này bằng cách nhấp vào nút Detail, hoặc double-click vào contact trên listbox.

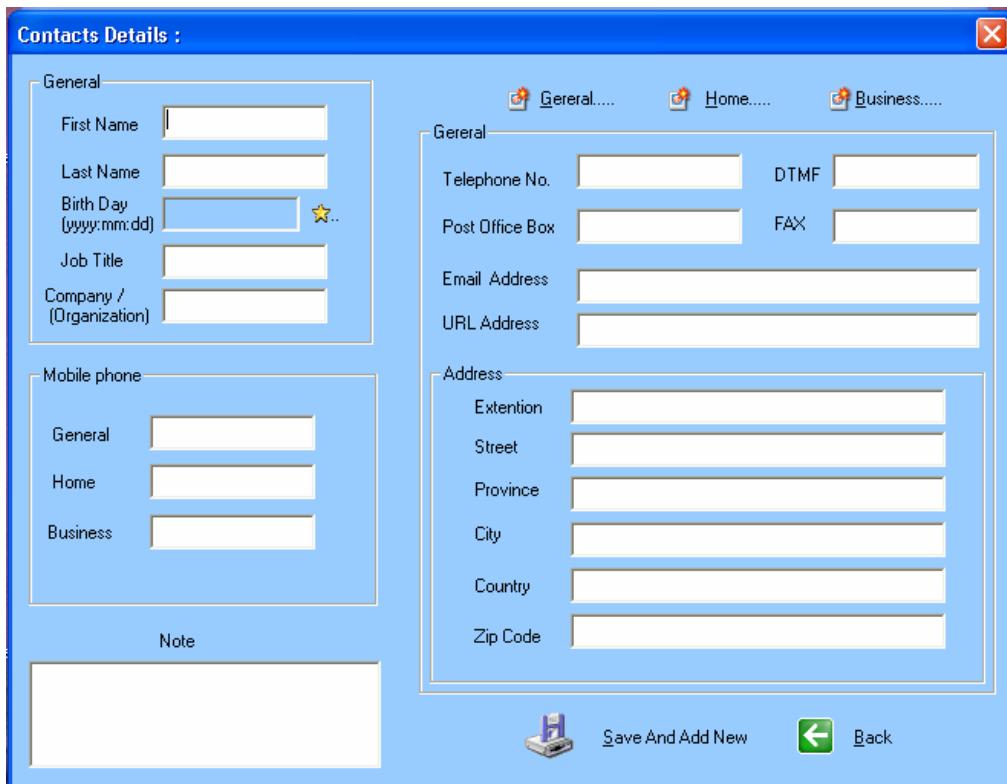
Sau khi đã kết nối với điện thoại thành công, người dùng có thể thực hiện truyền dữ liệu sang điện thoại, có thể truyền tất cả các contact có trong listbox hoặc chọn các contact sẽ truyền.



Nút nhấp "Send Pbk to phone" sẽ thực hiện truyền toàn bộ phonebook qua điện thoại, nút "Send Selected Items" sẽ thực hiện truyền các contact được chọn trên listbox qua điện thoại.

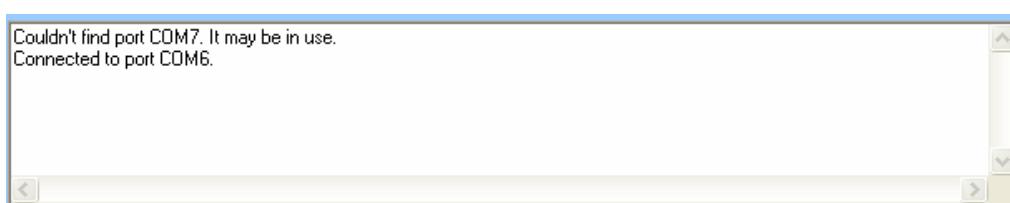
Việc truyền dữ liệu chỉ được thực hiện khi có một điện thoại kết nối thành công với máy tính.

Ngoài ra, chương trình còn cho phép người dùng tạo ra các contact trên máy tính để truyền qua điện thoại. Khi nhấn vào nút “New Contact”, một dialog sẽ hiện ra cho phép nhập các thông tin của một contact mới :



Hình B - 15 Dialog NewContact

Một số chức năng khác của ứng dụng như : lưu file, nạp file, xóa contact ra khỏi phonebook, hiển thị một vài thông tin liên quan đến chương trình trong textbox “log” như có kết nối thành công hay không, cổng COM có sẵn sàng hay không, các thông tin về quá trình trao đổi dữ liệu.



Hình B - 16 Textbox Log

## PHỤ LỤC C : Xây dựng ứng dụng HelloWorld trên Symbian với Series 60 SDK v1.2

Trong phần này, chúng ta sẽ xây dựng một ứng dụng HelloWorld xuất ra màn hình một lời chào HelloWorld nhằm minh họa cho việc xây dựng ứng dụng với Series 60 SDK và tạo file cài đặt (.sis) của ứng dụng.

### 1. Cài đặt các chương trình cần thiết :

- Microsoft Visual C++ 6.0 ( cần cài thêm service pack, ít nhất là service pack 3)
- Cài đặt môi trường thực thi Java, ở đây, do dùng bộ SDK v1.2 nên ta dùng Java Runtime Environment 1.3.1 để phù hợp với các công cụ của bộ SDK.
- Cài đặt Perl (Perl được dùng để chạy các Tool cho Symbian như biên dịch, tạo file .sis hay các Tool tiện ích khác...)
- Cài đặt bộ Series 60 SDK v1.2.
- Cài đặt Application Wizard và MmpClick đi kèm theo bộ SDK theo hướng dẫn cài nằm ở hai thư mục tương ứng là:  
\Symbian\6.1\Series60\Series60Tools\applicationwizard và  
\Symbian\6.1\Series60\Series60Tools\mmpclick

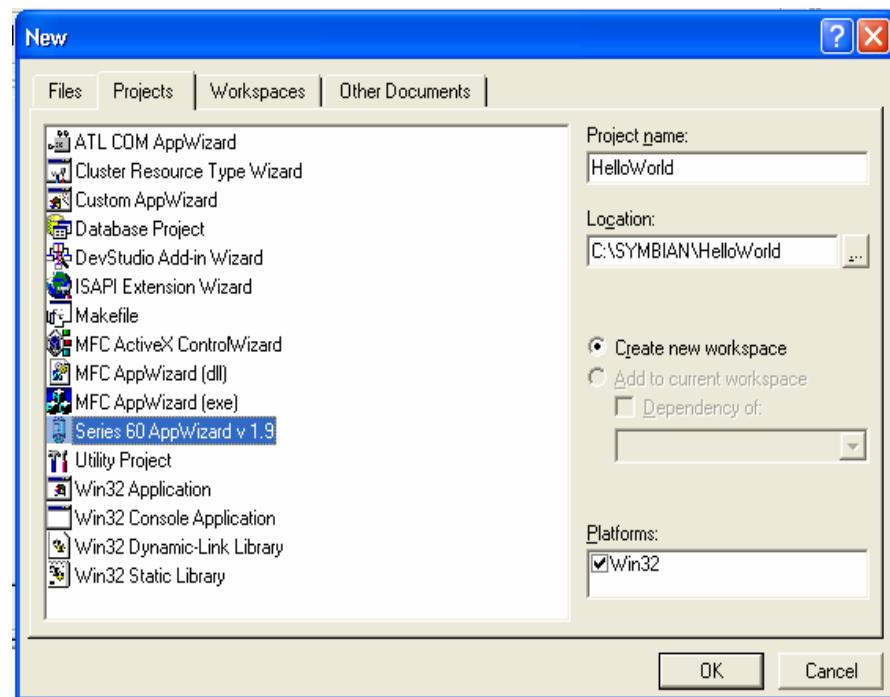
### 2. Tạo Project

Sau khi cài đặt thành công bộ SDK và các công cụ như Application Wizard và MmpClick, ta có thể tạo một project mới cho ứng dụng trên Symbian một cách dễ dàng trong Visual C++ 6.0.

Trong Visual C++ 6.0 :

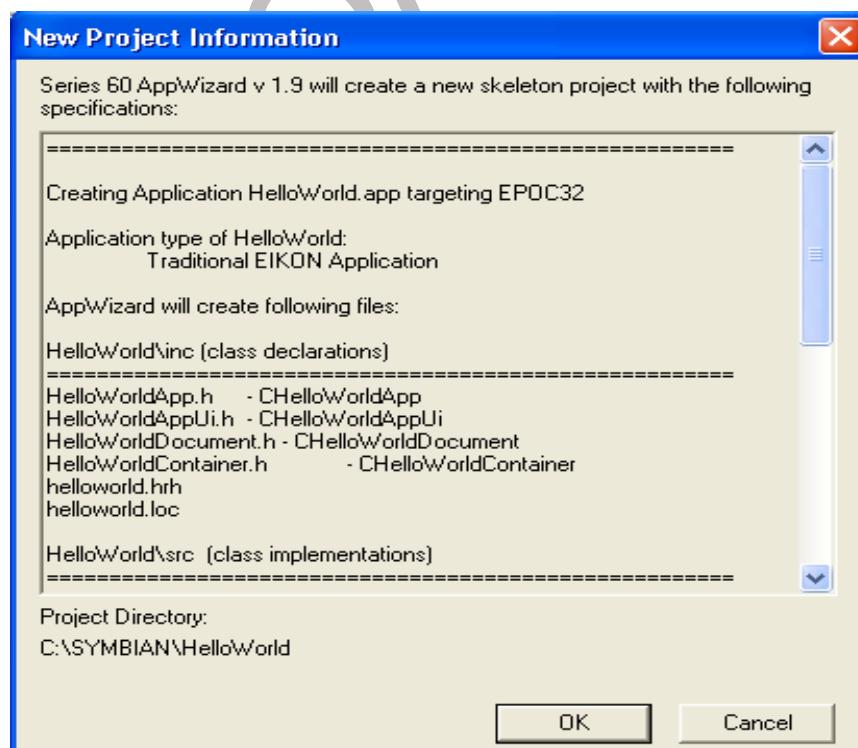
- Chọn : File-> New
- Chọn Series 60 AppWizard v1.9

- Nhập tên Project và nơi tạo project, nhấn next để tiếp tục.



Hình C - 1 Tạo Project symbian mới trên visual C++

Tiếp đó, ta để nguyên các thiết lập mặc định, nhấn Finish để kết thúc Wizard, khi đó, một màn hình sẽ xuất hiện thông báo các thông tin về ứng dụng sẽ tạo :

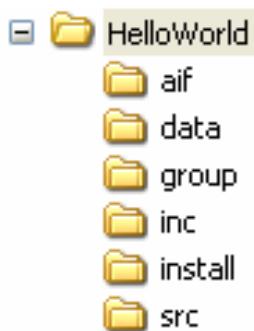


### Hình C - 2 Thông tin project mới tạo ra

Nhấn OK để xác nhận, khi đó, một project mới xây dựng ứng dụng cho Symbian đã được tạo ra.

### 3. Cấu trúc thư mục của ứng dụng HelloWorld

Sau khi kết thúc Wizard tạo ứng dụng, một thư mục mới của ứng dụng được tạo ra có cấu trúc như sau :

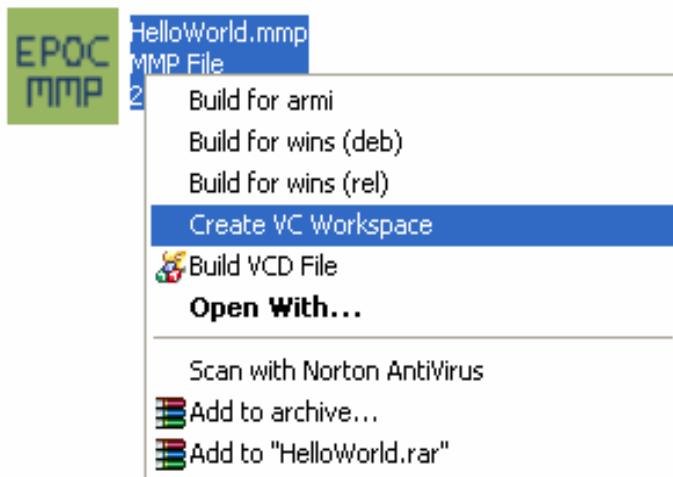


Hình C - 3 Cấu trúc thư mục của ứng dụng HelloWorld

- Thư mục group : Thư mục dự án, chứa file dự án: HelloWorld.mmp, bld.inf.
- Thư mục inc : chứa các file Header của các lớp và chứa các file khai báo tài nguyên.
- Thư mục src : là thư mục mã nguồn, chứa các file cài đặt của chương trình.
- Thư mục data : Thư mục dữ liệu: chứa dữ liệu cần cho chương trình ứng dụng.
- Thư mục aif : Thư mục thông tin ứng dụng: chứa file tài nguyên .rss để tạo file .aif và các hình ảnh, tài nguyên phục vụ cho ứng dụng. Tập hợp các hình này được lưu trữ trong một file .mbm (multi bitmap).
- Thư mục install : chứa các file .pkg, các thành phần cài đặt bổ sung cho ứng dụng như hình nền, file dữ liệu....

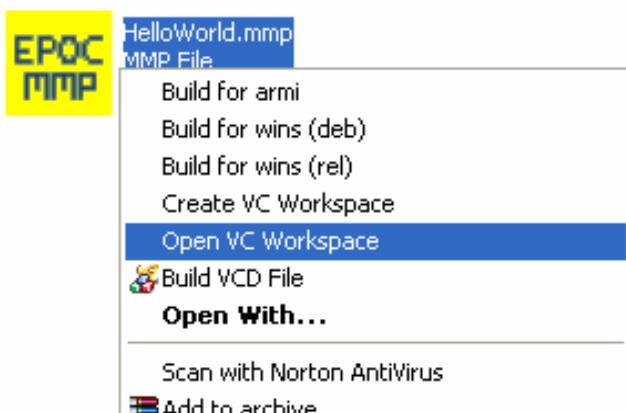
#### 4. Mở một project đã có :

Để mở một project đã có, ta vào thư mục group của project, click phải vào file .mmp, chọn Create VC Workspace :



Hình C - 4 Mở một project đã có

Khi đó, project sẽ được tạo một workspace để có thể mở trong Visual C, và để mở project, ta click phải vào file .mmp và chọn “Open VC Workspace”



Hình C - 5 Mở một project đã có

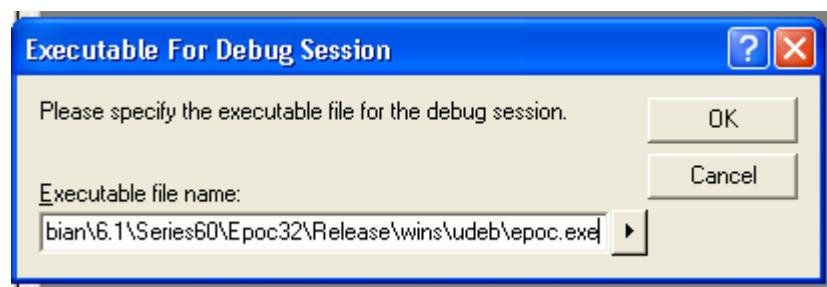
#### 5. Xây dựng và biên dịch ứng dụng

Ứng dụng có thể được xây dựng và kiểm lỗi ngay trong môi trường VC như các ứng dụng Visual C khác, để có thể biên dịch ứng dụng, có thể chọn từ menu Build hoặc nhấn F7.

Để chạy ứng dụng, chọn Execute từ menu Build hoặc nhấn F5, khi đó visual C sẽ yêu cầu chọn đến file thực thi của ứng dụng, ở đây, ta chọn đường dẫn tới máy ảo của Series 60 :

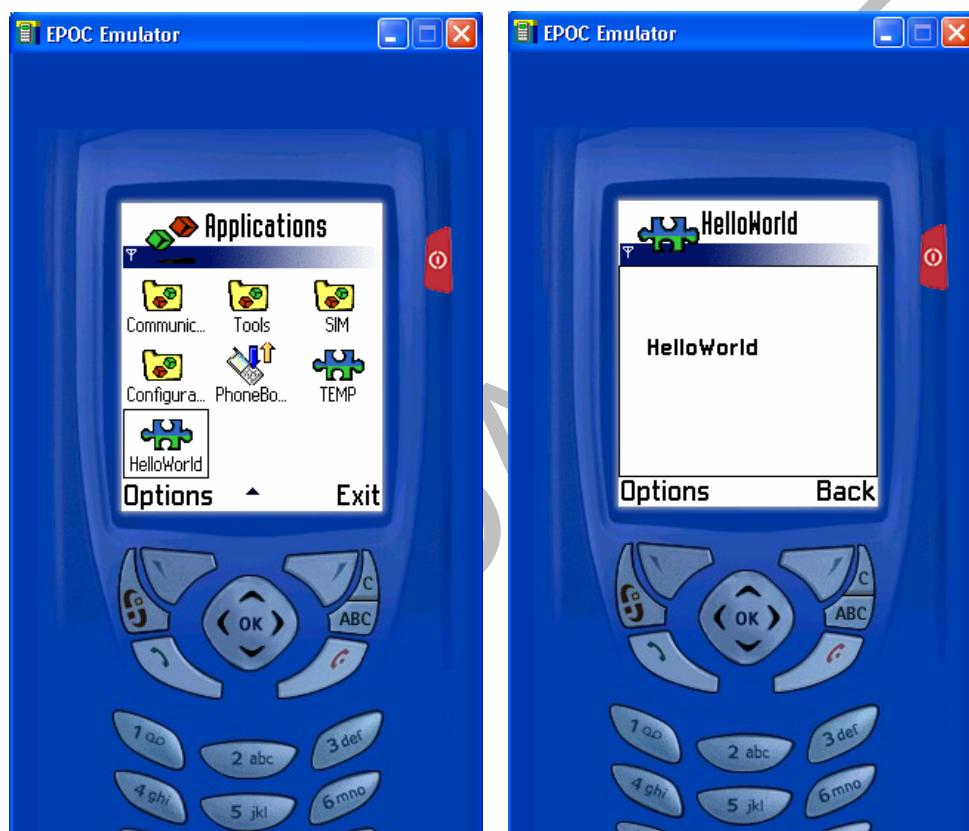
Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa

[nơi cài đặt SDK]\Epoc32\Release\wins\udeb\epoc.exe :



Hình C - 6 Chạy ứng dụng HelloWorld

Khi đó, máy ảo của Series 60 sẽ được chạy, ta sử dụng các phím di chuyển để chuyển tới ứng dụng mới tạo và chọn mở ứng dụng :



Hình C - 7 Ứng dụng HelloWorld

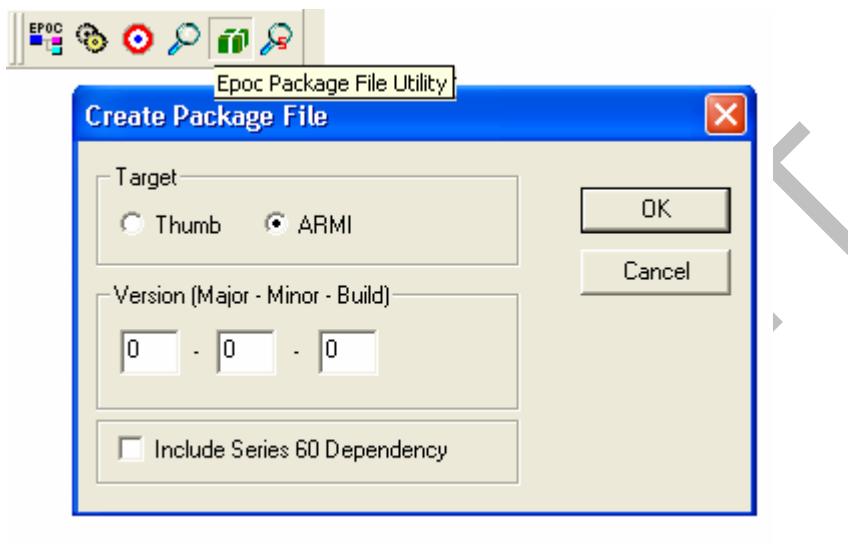
Khi đó, chúng ta đã hoàn thành ứng dụng HelloWorld.

#### 6. Tạo file cài đặt cho ứng dụng HelloWorld:

Ứng dụng sau khi kiểm thử thành công trên thiết bị ảo, để có thể cài đặt ứng dụng trên điện thoại thật, chúng ta phải tạo file cài đặt .sis cho ứng dụng.

Để tạo file cài đặt cho ứng dụng, chúng ta có thể tạo từ dòng lệnh bằng cách sử dụng lệnh makesis của Symbian, hoặc sử dụng công cụ mmpclick của bộ SDK như sau :

+ Trước hết phải biên dịch ứng dụng với thiết lập dành cho hệ thống AMRI như sau: Trên thanh công cụ của EPOC, chọn mục “Epoch Package File Utility”, chọn : target → ARMI, và biên dịch lại ứng dụng bằng cách click vào mục “Epoch Build Utility” trên thanh công cụ của EPOC.



Hình C - 8 Biên dịch ứng dụng cho hệ thống ARMI

Hoặc có thể biên dịch cho hệ thống ARMI bằng cách click phải vào file .mmp và chọn mục “Build for armi” như sau :

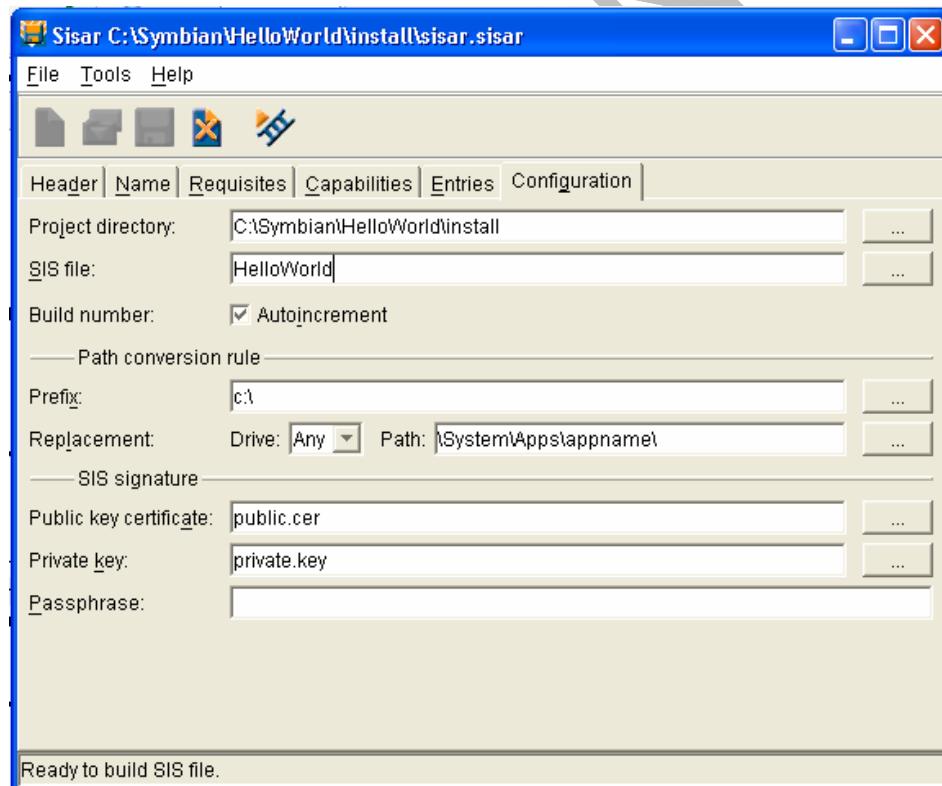


Hình C - 9 Biên dịch ứng dụng cho hệ thống ARMI

Khi đó, một màn hình console sẽ xuất hiện cho biết tình trạng biên dịch và thông báo lỗi nếu có của ứng dụng cho hệ thống ARMI.

+ Sau khi đã biên dịch ứng dụng dành cho hệ thống ARMI, ta sử dụng công cụ Sisar của bộ SDK để tạo file cài đặt cho ứng dụng như sau:

- Mở Sisar : C:\Symbian\6.1\Shared\EPOC32\Tools\sisar\sisar.jar.
- Chọn File->New Project và thiết lập các cấu hình cho project Sisar mới :
  - Từ Tool→ Import PKG file : chọn tới file HelloWorld.pkg trong thư mục install của ứng dụng.
  - Trong Tab : Configuration, chọn chỉ định đến thư mục để lưu project và file .sis sẽ tạo ra.



Hình C - 10 Tạo file cài đặt

- Khi đó, để tạo file .sis, chọn : Tool→Build SIS file, sau đó, một file .sis sẽ được tạo ra trong thư mục project chỉ định ở trên.

## 7. Cài đặt ứng dụng trên thiết bị thật:

Sau khi tạo được file .sis, ứng dụng đã sẵn sàng được cài đặt trên điện thoại thật. Để cài đặt ứng dụng có thẻ sử dụng chức năng cài đặt trên PC của bộ PC Suit hoặc truyền file .sis vào điện thoại (bằng Bluetooth , hóng ngoại, hoặc thông qua Cable) và sử dụng chức năng cài đặt ứng dụng có trên điện thoại.

## Tài liệu tham khảo

### Tài liệu viết:

- [ 1] Đặng Minh Thắng, Chu Nguyên Tú, *Xây dựng hệ thống điều khiển máy tính từ xa sử dụng công nghệ Bluetooth*, Luận văn cử nhân tin học, Đại học Khoa học Tự nhiên TP.Hồ Chí Minh, 2004.
- [ 2] Jan Beutel, Oliver Kasten, Matthias Ringwald, Frank Siegemund, Lothar Thiele, *Bluetooth Smart Nodes for Mobile Ad-hoc Networks*, Swiss National Science Foundation, 2003
- [ 3] Nupur Mittal, *Bluetooth Technology Models and Future*, Exforsys Inc, 2005
- [ 4] Charlie White, *Bluetooth: Past, Present and Future*, CEN talks with Mike Foley, Executive Director, Bluetooth SIG, Digital Media Online, 2005
- [ 5] Peter Judge, *Why Bluetooth version 2 matters*, Techworld, 2005
- [ 6] Forum Nokia – *Bluetooth Technology Overview* – Nokia, 2003
- [ 7] David Kammer, Gordon McNutt, Brian Senese, Jennifer Bray, *The Short Range Interconnect Solution Application Developer's Guide*, Synpress , 2002.
- [ 8] Atmel Corporation– *The Bluetooth™ Wireless Technology White Paper*, 2000
- [ 9] Sil Janssens, *Preliminary study: BLUETOOTH SECURITY*, 2004
- [ 10] Jahanzeb Khan, Anis Khwaja, *Building Secure Wireless Networks with 802.11*, Wiley Publishing, Inc., Indianapolis, Indiana, 2003

- [ 11] Steven Vittitoe, *Bluetooth Security*, SANS Institute 2004,
- [ 12] By Michelle Man, *BLUETOOTH AND WI-FI*, Socket Communications, Inc., 2002
- [ 13] Pico Communications, *A Comparison of Bluetooth™ and Wi-Fi™ (802.11b)*, Pico Communications, 2001.
- [ 14] Dr. Peter Driessen, *Bluetooth Evaluation Project*, BlueSuit Research, 2001.
- [ 15] Eric Holland, *Understanding Your Wireless Options*, Sensors online, 2004
- [ 16] Bluetooth – *Specification of the Bluetooth System*, Bluetooth, 2004.
- [ 17] Forum Nokia – *Developer Platform 1.0 for Series 60: Getting Started with C++ Application Development* - Nokia, 2003
- [ 18] Forum Nokia – *Series 60 Application Framework Handbook* - Nokia, 2002
- [ 19] Forum Nokia – *Introduction to Series 60 Applications for C++ Developers* - Nokia, 2002
- [ 20] Forum Nokia – *Setting Up and Using Bluetooth Hardware with Development Tools* - Nokia, 2004
- [ 21] Forum Nokia – *Designing Bluetooth Applications in C++* - Nokia, 2004
- [ 22] Nokia – *Series 60 SDK Help* – Nokia Series 60 SDK 1.2
- [ 23] Forum Nokia – *Using Contact APIs* – Nokia, 2004

Website:

- [ 24] Symbian, <http://www.symbian.com>
- [ 25] Forum Nokia, <http://www.forum.nokia.com>
- [ 26] Palo wireless, <http://www.palowireless.com>

Tìm hiểu công nghệ Bluetooth và viết ứng dụng minh họa

- [ 27] <http://www.bluetooth.com>
- [ 28] <http://www.bluetooth.org>
- [ 29] <http://www.zedge.no>
- [ 30] <http://www.securityfocus.com>
- [ 31] The Codeproject, <http://www.codeproject.com>
- [ 32] The Codeguru, <http://www.codeguru.com>
- [ 33] SourceForge, <http://www.sourceforge.net>
- [ 34] Experts Exchange, <http://www.experts-exchange.com>
- [ 35] The NewLC, <http://www.newlc.com>
- [ 36] Forum GSM , [www.gsm.com.vn/forum/](http://www.gsm.com.vn/forum/)
- [ 37] Series 60, [www.Series60.com](http://www.Series60.com)