Bluetooth Scatternet Formation and Internetworking
with 802.11 and GPRS

by

Satyajit Chakrabarti

B.Tech., Institute of Engineering and Management, University of Kalyani, 2002

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
Master of Science

in
THE FACULTY OF GRADUATE STUDIES
(Department of Computer Science)

We accept this thesis as conforming
to the required standard

The University of British Columbia
September 2004
© Satyajit Chakrabarti, 2004
Library Authorization

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Name of Author (please print)  
SATYAJIT CHAKRABARTI

Date (dd/mm/yyyy)  
27/09/2004

Title of Thesis:  
Bluetooth Scatternet Formation and Internetworking with 802.11 and GPRS.

Degree:  
Master of Science  
Year:  
2004

Department of  
Computer Science

The University of British Columbia  
Vancouver, BC  Canada
Abstract

Mobile Ad-Hoc networks (MANET) using Bluetooth technology has gained immense popularity among the networking community. Nowadays electronic devices like cell phones come equipped with Bluetooth. Whereas Bluetooth technology has certain obvious advantages like low power consumption and reliable connection but it suffers some inherent problems including low area of operation, limit of 7 slave devices per master device. 802.11 technology on the other hand, has a wider area of operation and therefore very useful as Access points and higher bandwidth in the order of 11 Mbps in 802.11b. But 802.11 has a higher power consumption than Bluetooth. Cellular networks have a much wider coverage in geographical area than 802.11. Complementing Bluetooth with 802.11 and cellular network technology like GPRS would solve the shortcomings of these three technologies.

This thesis is composed of four projects. It introduces RCBTEE (A Remotely Controlled Bluetooth Enabled Environment) which presents a simple Bluetooth Scatternet formation algorithm with remote control applications and a single access point connected to a content server. Then EquiBlue is presented. EquiBlue is a load balanced Bluetooth Scatternet formation algorithm. Then BlueFi is introduced. BlueFi presents a conceptual software layer in the protocol stacks of Bluetooth and 802.11 to realize interoperability in devices with both Bluetooth and 802.11 interfaces. At the end, BlueMobile, a novel protocol for handoff's in Bluetooth, 802.11 and GPRS networks is presented. A thorough analysis of all the four projects is done and detailed simulation results are provided. The results underscore the importance of achieving convergence in Bluetooth networks with other networks like 802.11 and GPRS.
Contents

Abstract ii

Contents iii

List of Tables vi

List of Figures vii

Acknowledgements x

Dedication xi

1 Introduction 1
  1.1 Motivation .................................................. 2
  1.2 Thesis contributions ....................................... 4
  1.3 Thesis Outline ............................................. 6

2 Background 7
  2.1 Bluetooth Technology ....................................... 7
  2.2 Bluetooth Protocol Stack ................................... 8
    2.2.1 Radio Layer ........................................... 9
    2.2.2 Baseband ............................................. 10
    2.2.3 Link Manager Protocol (LMP) ......................... 11
    2.2.4 Logical Link Control and Adaptation Protocol (L2CAP) ... 12
5.6 Discussions ........................................... 71

6 BlueMobile - A Mobile IP based Handoff system for Bluetooth, 802.11 and GPRS links 72

6.1 Introduction ......................................... 72
6.2 Related Work ....................................... 74
6.3 Mobile IP ........................................... 74
   6.3.1 Mobile IP Architecture ....................... 74
   6.3.2 Mobile IP Adaptation for Bluetooth, 802.11 AND GPRS ..... 76
6.4 Mobile IP Handoff Algorithm between Bluetooth, 802.11 & GPRS 78
6.5 Analysis of the Algorithm ......................... 82
6.6 Discussions ....................................... 87

7 Conclusions .......................................... 88

7.1 Summary ........................................... 88
7.2 Future Work ....................................... 90

Bibliography ........................................... 92

Appendix A ............................................ 97
List of Tables

2.1 Bluetooth Power Modes ........................................ 14

3.1 Comparison of RCBTEE with other Bluetooth Scatternet algorithms 34

4.1 Comparison of EquiBlue with other Scatternet Algorithms .... 52

5.1 API's available for the LSC for use ................................. 62

5.2 Summary of Results of BlueFi ...................................... 65

5.3 Comparison of Bluetooth and 802.11b ............................... 66

5.4 Comparison of Wireless standards .................................. 67

6.1 Results of BlueMobile ............................................... 86
# List of Figures

2.1 A Tree Structured Scatternet ........................................... 8
2.2 The Bluetooth Protocol Stack ........................................... 9
2.3 Bluetooth Packet Format [20] ........................................... 9
2.5 Bluetooth Packet Header [20] ........................................... 10
2.6 Bluetooth Link Controller State Diagram [20] ....................... 11
2.7 Overview of the Lower software layers of the Bluetooth stack [20] ... 12
2.8 GPRS Network Architecture ............................................ 17

3.1 The architecture of the Bluetooth Remote control environment ... 19
3.2 Piconet formation .......................................................... 20
3.3 Scatternet formation with multiple piconets. ......................... 21
3.4 A Tree Structured Scatternet ............................................ 22
3.5 RCBTEE Step 1 ............................................................. 23
3.6 RCBTEE Step 2 ............................................................. 23
3.7 RCBTEE Step 3 ............................................................. 24
3.8 RCBTEE Step 4 ............................................................. 24
3.9 RCBTEE Step 5 ............................................................. 25
3.10 RCBTEE Step 6 ............................................................ 25
3.11 RCBTEE Scatternet Simulation ......................................... 31
3.12 RCBTEE Remote Control Centre ..................................... 33
6.3 MH moves from GPRS to Bluetooth ........................................ 80
6.4 Overlapped Bluetooth and 802.11 zone ................................. 82
Acknowledgements

I would like to express my gratitude to my supervisor, Dr. Son Vuong, for his guidance, inspiration and encouragement. I am also grateful to Dr. Victor Leung for being my second reader and for his useful comments that have improved this thesis.

I am thankful to Prof. Norm Hutchinson for his encouragement and feedback on my projects. I would like to thank my Professors - Prof. Mike Feeley, Prof. Karon Maclean, Prof. Joanna McGrenere, Prof. Raymond Ng, Prof. Mark Greenstreet, Prof. Kellogg Booth, Prof. Alan Wagner and Prof. Gail Murphy.

I gratefully acknowledge the financial support from the Department of Computer Science at UBC in the forms of TA and RA as well as support from CITR Network of Centers of Excellence, the Advanced Systems Institute of BC, and the industrial partner Proface INDE.

I would like to thank the co-authors in various publications - Liyun Wu, Anirban Sinha and Rajashree Paul, as well as my lab mates - Sukanta Pramanik, Christian Chita, Juan Li, Kamran Malik, Xiao Juan Cai, Gary Huang, Ken Deeter, Mohammad Alam, and Sergio G. Valenzuela.

I would also like to acknowledge the love my parents, my grandma, my sister Sanghamitra and brother-in-law Debashis gave and for their consistent support during my graduate studies at UBC.

Satyajit Chakrabarti

The University of British Columbia
September 2004
To my mom, dad and grandma

Always there for me, no matter what
Chapter 1

Introduction

Bluetooth [1], [2], [3], [4] is a promising new short range wireless technology, which enables portable devices to form wireless ad hoc networks and PAN (personal area networks) and is based on a frequency hopping physical layer. This implies that hosts communicate after having discovered each other by synchronizing their frequency hopping patterns. This also means that even if all nodes are within direct communication range of each other, only those nodes which are synchronized with the transmitter can hear the transmission. To support any-to-any communication, nodes must be synchronized so that the pairs of nodes (which can communicate with each other) together form a connected graph.

Bluetooth has become another important platform for ad hoc networking, apart from 802.11. Ad hoc networking over Bluetooth can lead to many useful applications. For example, in a conference room, a special announcement can be broadcast to the Bluetooth enabled mobile phones and hand-held computers through an ad hoc network.

The area of ad hoc networking has gathered much research interests in the past years. Many studies have concentrated on the routing issues of ad hoc networks...
These studies usually assume that any two in-range nodes can communicate with each other. Therefore, an ad hoc network can be modeled as a graph such that the in-range nodes are adjacent. For example, simulation-based studies [6], [7] of ad hoc routing protocols have been conducted with a link-layer model based on or similar to the IEEE 802.11b standard.

A Bluetooth network is composed of piconets. Each piconet contains one master and up to seven slaves. Piconets can be connected into a Scatternet by sharing slaves. As shown by Miklos et al. [8] and Zurbes [9], the configuration of a scatternet has great effects on the performance of the network. For instance, when a scatternet contains more piconets, the rate of packet collisions increases.

1.1 Motivation

While much research has been done on the formation of Scatternet in Bluetooth networks, other works have hitherto not investigated the load balance in Scatternets. In Scatternets, sometimes lower processing power devices like cell phones can become Master devices, thus proving to be a bottleneck in the Scatternet in terms of processing power. Also, sometimes due to formation of large Scatternets with many piconets causes packet collisions due to same frequency hopping spectrum in neighboring piconets in a Scatternet.

Since Bluetooth hotspots and PANs (personal area networks) are an emerging area, no investigation has hitherto been made into remote querying and remote controlling devices in a Bluetooth hotspot. With WAP and WXML and XHTML becoming very popular tools in web browsing from mobile devices, we thought it would be interesting to try to port our remote control and querying system to web enable it so that it could be used with mobile devices as well.
Following are the previous works that have investigated handoff's: Kansal et al. [14] introduce a Handoff scheme for Bluetooth devices to allow mobility of devices in Bluetooth public access (BluePAC) environments. The paper by Mishra et al. [18] does a thorough analysis of the handoff procedure in the 802.11 MAC layer. Kastell et al. [19] presents security issues involved in hybrid handover procedures. Pack et al. [15] uses a predictive authentication for fast handoff's in 802.1x mode. However, none of the previous works with Bluetooth handoff's have investigated handoff with other wireless technologies like 802.11 and GPRS. With more and more handheld devices being shipped with inbuilt Bluetooth and 802.11 interfaces, it's only natural that an interoperable framework is inevitable. IEEE 802.15.2 task group [10] looks into existence issues of Bluetooth WPAN's and 802.11 WLAN's.

In [44] Buddhikot et al. present integration of 802.11 and GPRS on seamless connectivity. Min-hua et al. [43] introduce a Mobile IP handoff scheme between 802.11 and GPRS. In [45] Sharma et al. bring forward a vertical handoff system between GPRS and WLAN by using extended Mobile IP. Albrecht et al. [42] present protocol concepts for an extension of IP for mobility issues in Bluetooth networks. Baatz et al. [17] concentrates on handoff support for mobility with IP over Bluetooth. Perkins et al. [16] present a handoff scheme for Mobile IP's. None of the previous works on Mobile IP have focused on using Mobile IP to achieve handoff of Bluetooth with 802.11 or GPRS.

In this thesis we focus on Bluetooth Scatternet formation and interoperability with other prevalent wireless technologies- 802.11 and GPRS. In the first few chapters we concentrate on Scatternet formation issues and formation of a remote control and query enabled personal area network. We investigate load balancing issues and present a load balanced Scatternet formation algorithm.
In the later part of the thesis, we focus on interoperability of Bluetooth with 802.11 and GPRS. We try to solve this problem using two approaches - one using a software daemon that acts as a software layer conceptually between the protocol stack of Bluetooth and 802.11 by trapping packets between the interface and protocol stack. In the other approach, we use Mobile IP to achieve handoff between a Mobile IP enabled 802.11 and GPRS network.

1.2 Thesis contributions

This thesis addresses two major deficiencies in the area of Bluetooth technology: aspects of Scatternet formation and interoperability with 802.11 and GPRS networks.

To address the problem of Scatternet formation, we have proposed RCBTEE and EquiBlue. RCBTEE presents a system for Remote controlling and remote querying Bluetooth devices organized in a Bluetooth hotspot according to a Scatternet formation algorithm. A remote web interface allows to control the Bluetooth devices in the hotspot as well as view the status of devices in the hotspot. EquiBlue presents a novel load balanced Scatternet formation algorithm. EquiBlue assigns a weight to a device based on processing power and tries to reduce the piconet size for low weight devices and maximize the piconet size for higher weight devices. At the same time, EquiBlue tries to keep the difference in size between neighboring piconets small, in case the neighbouring master devices are of similar weight. This would be very useful in applications where the Master device needs more resources like memory, processing power, since in our system the higher processing power device is assigned the role of Master.

To address the problem of interoperability we have proposed BlueFi and BlueMobile. BlueFi presents a convergence architecture for Bluetooth and 802.11 by introducing a software layer in the protocol stack of Bluetooth and 802.11. We
have analyzed various delay elements and optimized the convergence model. Blue-Mobile presents a novel solution for interoperability of Bluetooth, 802.11 and GPRS networks using Mobile IP. By introduction of Mobile IP elements like Home Agent (HA) and Foreign Agent (FA) in the network architectures, we analyze various delay elements in the network and optimize the Handoff model.

The following are the research contributions of this thesis:

- We propose a system for remote Control and remote querying of devices present in a Bluetooth hotspot arranged according to a simple Scatternet formation algorithm. This would be very useful in applications where remotely querying the devices in the hotspot is controlling devices is required. We have done a thorough estimation of the order of complexity of messaging between the Bluetooth devices.

- We present a load balanced Scatternet formation algorithm based on assigning weights to devices. We have simulated with different number of nodes and evaluated it and shown that our algorithm is effective in achieving a load balanced network. Our algorithm would be effective in applications where the Master device needs more processing power, or where a load balance would increase the throughput in the network.

- We propose a system for interoperability between Bluetooth and 802.11 devices by introduction of a conceptual software layer in the protocol stack of Bluetooth and 802.11 which would actually be implemented as a operating system daemon which would trap the packets from the interface, modify or analyze them and send them to the protocol stack. We have analyzed the various delay elements in the handoff system. This system would be able to support interactive audio, video and data applications over the network.
• We present a novel Handoff system between Bluetooth, 802.11 and GPRS links based on Mobile IP. We introduce Mobile IP elements in the networks of Bluetooth, 802.11 and GPRS and analyze all the cases of handoff between the different technologies. We have also done an analysis of the delay elements in Mobile IP handoff system. This system would be able to maintain a TCP/IP connection over the Access point since typically the maximum retransmission timeout of TCP/IP is 1 minute. Audio or Video applications would suffer a slight jitter during handoff.

1.3 Thesis Outline

This thesis is composed of seven chapters. In Chapter 2, background information and theory related to Bluetooth Scatternet formation and 802.11 and GPRS technologies is discussed. Chapter 3 introduces a novel idea for remote control and querying Bluetooth device state in hotspots - called “A Remotely controlled Bluetooth Enabled Environment (RCBTEE)”. Chapter 4 presents EquiBLUE - a load balanced Scatternet formation algorithm. Blue-Fi: An architecture for convergence of Bluetooth and 802.11 networks, is presented in Chapter 5. Chapter 6 presents BlueMobile - A Mobile IP based Handoff system for Bluetooth, 802.11 and GPRS links. Finally the thesis conclusion and future research work are discussed in Chapter 7.
Chapter 2

Background

2.1 Bluetooth Technology

Bluetooth is short range 2.4 GHz ISM band technology. It requires connection oriented links before data can be sent or received [21]. Bluetooth radio system employs the Frequency Hopped Spread Spectrum (FH-SS) to gain access to the wireless medium. Bluetooth devices share 79 channels of 1MHz bandwidth within the 2.45 GHz band. Frequency hopping improves security and FH-SS gives interference immunity.

A Bluetooth Device (BD) can either take the role of a Master or Slave device. One Master BD can connect to at most 7 Slave BD’s to form a piconet. A Master BD in INQUIRY state hops 3200 times per second according to a 32 channel hopping sequence. A Slave BD in INQUIRY SCAN state alters its listening frequency every 1.28 seconds along the same sequence. Once a Master BD discovers a Slave BD in radio range and the Master BD learns the address and clock of the Slave BD, it switches to PAGE state and the slave switches to PAGE SCAN state. The pseudo-random frequency hopping sequence of both BD’s are tightly synchronized prior to their entering the CONNECTED state. Connection setup delay may be as long as 10 seconds during device discovery, and overcrowding the medium may increase the
delay more [25]. The radio range of BD equipped with a class-B transmitter is only 10 metres [21]. With class-A transmitter, the Bluetooth device range is 100 metres.

A piconet consists of 1 master and 7 slaves (as in Bluetooth 1.1b). Slaves can only communicate with masters and Masters can only communicate with slaves. A slave may be shared between two piconets as a “bridge-slave” allowing communication between the two piconets. After master and slave are connected they communicate with a hopping sequence over all 79 channels at the rate of 1600 hops per second. Several Piconets join together through bridge slaves to form a Scatternet. A typical tree structured scatternet is shown is Fig. 2.1.

![Figure 2.1: A Tree Structured Scatternet](image)

2.2 Bluetooth Protocol Stack

The various layers of the Bluetooth protocol stack (Fig. 2.2) are briefly described below:
2.2.1 Radio Layer

Bluetooth operates in the unlicensed ISM band in the frequency range 2400 MHz to 2483.5 MHz, in most countries. Frequency Hopping Spread Spectrum (FHSS) is used with hops at 2402 + k MHz, where k = 0,1,2,3,...,78. The Frequency hopping is used to combat interference and fading. The nominal hop rate is 1600 hops/second. Gaussian Binary Shift Keying is used to modulate the radio waves. The Gaussian prefilter has a bandwidth delay product \( BT = 0.5 \). The transmitter power is 0dBm for 10 metre range and 20 dBm for 100 metre range. The packet format of Bluetooth data packets is shown in Fig. 2.3. Fig. 2.4 shows the Bluetooth Packet Access Code format and Fig. 2.5 shows the Bluetooth packet header format.
2.2.2 Baseband

The baseband layer controls the radio layer. The frequency hop sequences are provided by the baseband layer. Baseband layer also takes care of the encryption in the lower layers, as well as packet handling. Two types of links can be established between a master and a slave:

1. Synchronous Connection Oriented (SCO) Link

The SCO link is symmetrical and point to point between the master and a specific slave. The SCO links reserve slots and can therefore be considered as a circuit-switched connection between the master and slave. The SCO links support time critical information like voice traffic. Upto three SCO links are supported by a master for the same slaves or to different slaves. A slave can support upto three SCO links from the same master or two SCO links if the links originate from different masters. SCO packets are never retransmitted. SCO links are optional and are not necessary to be implemented for Bluetooth compliance.

2. Asynchronous Connection-Less (ACL) Link

The ACL links provide a packet-switched connection between the master and all active slaves participating in the piconet. ACL links support both synchronous and asynchronous data. Between a master and a slave, only a single ACL link can exist. In majority of ACL packets, packet retransmission is done
to ensure data integrity.

2.2.3 Link Manager Protocol (LMP)

The link manager Protocol is used in link setup and control. The signals are interpreted and filtered out by the link manager on the receiving side and are not propagated to the higher layers. The LMP manages the piconet, and configures the links for service parameters like encryption. It provides functionality to attach or detach slaves, switch roles between a master and a slave and to establish ACL or SCO links. The LMP is also used to handle the low power modes - hold, sniff and park, which are designed to save power when the device is not sending data. Fig. 2.7 shows the state diagram of the Bluetooth Link Controller.
2.2.4 Logical Link Control and Adaptation Protocol (L2CAP)

The L2CAP protocol supports packet segmentation and reassembly, higher level protocol multiplexing, and conveying of quality of service information. The protocol multiplexing allows multiple applications to use the Bluetooth link. The segmentation and reassembly function allows the protocol to reduce the packet size provided to the size of the packets accepted by Baseband layer. The L2CAP accepts packet sizes upto 64Kb but the baseband layer can accept packets upto 2745 bits, so the L2CAP packets need to be segmented before sending to the lower layers. For received packets, the reverse procedure has to be followed where the packets are combined in the proper order. L2CAP also allows applications to impose QoS on parameters like latency, delay variation and peak bandwidth. In general, L2CAP provides network layer functions to applications and higher layer protocols.
2.2.5 Host Controller Interface

The Host Controller Interface provides a command interface to the baseband controller and the link manager, and to access to the hardware status and control registers. The HCI provides an uniform method of accessing Bluetooth baseband capabilities. The HCI link commands provide the host with the ability to control the link layer connections to other Bluetooth devices. These commands involve the Link Manager to exchange LMP commands with remote Bluetooth devices. The HCI driver and HCI firmware is shown in the lower layers in Fig. 2.6.

2.3 Bluetooth in Operation

For A to talk to B:

Step 1: Discovering a Bluetooth device:

1. Device A transmits one or more inquiry packets
2. Device B replies with FHS packet

Step 2: Connecting to service discovery database:

1. ACL baseband connection is established.
2. L2CAP connection is set up over ACL channel.
3. SDP connection over L2CAP channel.
4. Device A receives DUN info from B's service discovery database.

Step 3: Connecting to Bluetooth service:

1. ACL link is set up.
2. Device A utilizes LMP to configure link.
3. L2CAP connection using the RFCOMM protocol
4. DUN connection is set up using RFCOMM connection.
2.4 Bluetooth Power Modes

Table 2.1: Bluetooth Power Modes

<table>
<thead>
<tr>
<th>Class</th>
<th>Max output power</th>
<th>Range</th>
<th>Power control</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100 mW (20 dbm)</td>
<td>100 m</td>
<td>Mandatory</td>
</tr>
<tr>
<td>2</td>
<td>2.5 mW (4 dbm)</td>
<td>10 m</td>
<td>Optional</td>
</tr>
<tr>
<td>3</td>
<td>1mW (0 dbm)</td>
<td>1 m</td>
<td>Optional</td>
</tr>
</tbody>
</table>

The various Bluetooth Power modes are given below. Table 2.1 shows the power requirements and the range of various Bluetooth devices.

Sniff Mode:

1. Slave agrees with its master to periodically listen for the master's transmissions.

Hold Mode:

1. Device agrees to remain silent (in that particular piconet) for a given amount of time.
2. Device keeps its temporary address, AM_ADDR.

Park Mode:

1. Slave device agrees with its master to park until further notice.
2. Device relinquishes its active member address, AM_ADDR.
3. Slave periodically listens to beacon transmissions from the master.
4. Slave can either be invited back (by the master) or can get itself unparked.

2.5 802.11 Technology

IEEE Standard 802.11 specifies a single Medium Access Control (MAC) sub layer and three Physical Layer Specifications.
The MAC provides the following services:

1. Authentication (station service)
2. Deauthentication (station service)
3. Privacy (station service)
4. MSDU delivery (station service)
5. Association (distribution system service)
6. Disassociation (distribution system service)
7. Distribution (distribution system service)
8. Integration (distribution system service)
9. Reassociation (distribution system service)

Stations can operate in two configurations:

1. **Independent configuration**: the stations communicate directly to each other, so there is no infra-structure need to be installed. That is why we called this “ad-hoc” networks. It is easy to operate, but the disadvantage is that the coverage area is limited. Stations in such configuration are in a Basic Service Set (BSS). Without the ESS the stations operate in an Independent BSS (IBSS).

2. **Infra-structure configuration**: the stations communicate to Access points which are part of a Distribution System. An Access point serves the stations in a BSS. The set of BSSs are called Extended Service Set (ESS). Note that 802.11 only specifies the air-interface, that is the interface between stations
and between stations and Access points. With a Distribution system, the coverage area can be extended to whatever the internals of the distribution system for instance with bridged wired LANs.

The standard provides the above mentioned services with the following functionality: roaming within a ESS, multiple data rates in BSSs and Power Management (stations can switch off their transceivers to conserve power). The MAC protocol is Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). The standard includes a formal description of the MAC protocol using the SDL method standardized by the International Telecommunications Union, Section Telecommunication (ITU-T, formerly CCITT) The standard provides 2 Physical layer specifications for radio, operating in the 2.400 - 2.483.5 MHz band (depends on local regulations) and one for infrared.

**Frequency Hopping Spread Spectrum Radio PHY.** This PHY provides for 1 Mbit/s (with 2 Mbit/s optional) operation. The 1 Mbit/s version uses 2 level Gaussian Frequency Shift Keying (GFSK) modulation and the 2 Mbit/s version uses 4 level GFSK. Direct Sequence Spread Spectrum Radio PHY. This PHY provides both 1 and 2 Mbit/s operation. The 1 Mbit/s version uses Differential Binary Phase Shift Keying (DBPSK) and the 2 Mbit/s version uses Differential Quadrature Phase Shift Keying (DQPSK).

**Infrared PHY.** This PHY provides 1 Mbit/s with optional 2 Mbit/s. The 1 Mbit/s version uses Pulse Position Modulation with 16 positions (16-PPM) and the 2 Mbit/s version uses 4-PPM. Each PHY specification includes state diagrams to formally describe the protocols.
2.6 GPRS Technology

GPRS (General Packet Radio Service) uses the bursty nature of voice traffic to make use of the physical channels of GSM for its packet traffic. Although it uses the same physical channels, GPRS uses new logical radio channels for its packet data traffic. It uses GSM network for operation. Fig. 2.8 shows the GPRS architecture in the GSM network.

![GPRS Network Architecture](image)

Figure 2.8: GPRS Network Architecture

The Mobile Host (MH) accesses the GPRS network via the Base Station. The Packet Control Unit (PCU) is a hardware upgrade for the GPRS to be used in the GSM. Two service nodes are defined in GPRS - serving GPRS support node (SGSN) and gateway GPRS support node (GGSN). The GGSN acts as an interface to public networks like the Internet and contains the routing information to be used to tunnel packets from the Mobile Host through the SGSN. The SGSN are in charge of one or more Base station and they do location management through the HLR and VLR and are responsible for the delivery of packets. The GGSN determines which Mobile Host the packet belongs to and the packet is forwarded to the SGSN to be delivered to the Mobile Host (MH). Fig. 2.8 shows the generalized architecture of GPRS.
Chapter 3

A Remotely Controlled Bluetooth Enabled Environment

3.1 Introduction

Bluetooth [1],[2],[3],[4] technology's popularity is due to its low-cost, low power radio technology and operation in the ISM band. Bluetooth devices form an ad-hoc network called piconet, provided the devices are within the communication range (10m - 100m). A Piconet has one master and a maximum of seven slaves and the master-slave communication is based on time-division duplex mechanism: The master allocates the transmission slots and these slots are then used alternatively by master and slave to transmit Bluetooth frames.

Bluetooh technology employs frequency hopping, to allow for concurrent Bluetooth communications within radio range of each other without perceptible interference. Multiple piconets can, thereby, coexist which can then be inter-networked. Such a system of networked piconets is called a Scatternet. A device participating in two piconets is said to form a "bridge" between them. These bridge nodes are used for communication between different piconets in a Scatternet.
In this chapter, we present a simple Scatternet formation algorithm as compared to a distributed scheduling algorithm described in [26].

![Figure 3.1: The architecture of the Bluetooth Remote control environment](image)

### 3.2 Related Work

Earlier works in remote control of applications include controlling a video conferencing application [27] and providing a framework for controlling lights and stereo components [28]. In CMU's Pebbles project, a PDA is used to control a single PC's screen and various PC applications, such as PowerPoint, and Web browser [29]. However, we are focusing on controlling multiple device outputs. Also, user interaction with multiple devices and sensors in a room is being investigated in Stanford's Interactive Workspaces project [30]. They have proposed multi-device user interfaces in which a user can “pick” an object from a PDA and “drop” it on a PC screen or digital whiteboard [31]. In our work, we are focusing on the Web, and enabling a web page to simultaneously control multiple Bluetooth enabled devices.
3.3 Architecture of Remotely Controlled Bluetooth Enabled Environment (RCBTEE)

![Diagram](image)

Figure 3.2: Piconet formation

The Bluetooth specifications specify OSI layer 1 (baseband radio) and OSI layer 2 (connection set-up, link management). So we implement the RCBTEE in OSI layer 3 as a middleware. Fig 3.1 shows the architecture of the Bluetooth Remote control environment. The Bluetooth scatternet is formed on top of the IP Backbone. Fig 3.2 shows Piconet formation in the RCBTEE Architecture. The Master device connected to the Internet and communicating with a webserver. Fig 3.3 shows Scatternet formation with multiple piconets. A Master connected to the Internet and piconets are interconnected via slave bridges. Fig 3.4 shows A tree structured Scatternet like the RCBTEE formation topology.

3.4 Scatternet Formation Algorithm

We devised a simple scatternet formation algorithm so that the range of the control of Bluetooth devices can be maximized. Our main focus was to have at least one Bluetooth enabled device to act as a bridge to overcome the 10-metre radius barrier.
The proposed algorithm employs only local information and can adapt to changes such as arrival and departure of nodes. On activation, the devices can issue a page/inquiry to the master or go into inquiry scan state and wait to be discovered by the master by matching with the frequency hop of the master device. The Inquiry Procedure systematically scans all 32 Inquiry channels in order to discover other devices. A target detects an Inquiry while performing an Inquiry and its Scan executes an Inquiry Response. In this way, one by one Bluetooth device connects to the master and forms the piconet. The scatternet formation algorithm is devised and implemented as follows:

Fig 3.5 shows MI (Access point) is initialized. Fig 3.6 shows MI performs INQUIRY procedure to discover devices in the neighborhood. Fig 3.7 shows the case when MI discovers devices and forms the piconet, other Bluetooth devices in the neighborhood are alternating between Master and Slave mode. Fig. 3.8 shows some devices in the neighborhood assume the role of Master and begin INQUIRY, other devices perform INQUIRY SCAN. Fig.3.9 shows the case when Piconets join with the piconet formed by MI through slave bridge. Fig. 3.10 shows all the devices in
Figure 3.4: A Tree Structured Scatternet

the environment within radio range join in a tree like manner through slave bridges.
Figure 3.5: RCBTEE Step 1

Figure 3.6: RCBTEE Step 2
Figure 3.7: RCBTEE Step 3

Figure 3.8: RCBTEE Step 4
Let MI be the Master connected to the internet, undis is a device still not discovered by the scatternet, ML and SL are the master list and slave list respectively, which are maintained at the webserver.
Do While MI is connected

1. Initialize (MI); MI performs INQUIRY procedure.

2. Bluetooth device ∈ ML ∪ SL performs INQUIRY SCAN periodically waiting to be discovered by new device nd (every 1.28s over a window of 11.25 ms).

3. For any new device nd ∈ undis
   nd performs INQUIRY
   if nd < D of ML (where D = 10m, range of Bluetooth devices [1])
     then connect to mi ∈ ML
   else if nd < D of si ∈ SL
     then connect to si as new Master mi where si is Slave Bridge
     ML_{i+1} ← ML_i ∪ m_i
   else
     nd goes to UNCONNECTED

4. mi ∈ ML performs MAINTAINENCE and UPDATE
UPDATE:
∀ m ∈ ML sends ∀ s ∈ piconet (m) messages: Master list, number of masters connected, request for parameter update.

UNCONNECTED:
nd goes to power save mode and performs INQUIRY after certain interval. As recommended in specification [1], some randomization is introduced to the intervals between any two inquiries.

MAINTAINENCE In case of any node arbitrarily leaving the network (i.e. maybe switched off):
If node was a slave, master sends message to MI.
If node was a master, all slaves connected to it does step 3 of the algorithm.
If node was a bridge slave, master performs INQUIRY to find other slaves to use as a bridge.

DATA tuple:
<message-type, message-id, source, destination, data>, message type can be data, control or broadcast.

If two or more Bluetooth devices come into a piconet simultaneously, they will connect to the master one by one based on success of INQUIRY. In case the master fulfils its quota of 7 slaves after connected on device, the second device will form a master, connected to the piconet master by a slave bridge. In this case the bridge is chosen when the device acts as a master and performs INQUIRY and discovers the first slave device.

An important issue is how slave devices know the direction and the nodes on the way to the primary master and how to avoid loops;

A slave node connects to a piconet master, and sends messages to the pi-
The piconet master in turn sends and forwards messages (with a TTL = M+1, where M is number of master devices in the scatternet) only to the slaves which are bridge slaves. The bridge slaves in turn forward messages to their masters and in this way the message reaches the primary master connected to the internet. Flooding is avoided by the TTL assigned to messages at their origin.

A scatternet formation algorithm has two important performance measures: time complexity and communication complexity.

Time complexity is defined as the amount of time to form a scatternet. Our algorithm is designed to work in such an environment that nodes arrive and leave in an incremental fashion. Instead of presenting the total amount of time used for the whole scatternet formation, we analyze the delay for a new node to connect to the scatternet successfully, which implies the waiting time for a node to join a scatternet.

Firstly we consider the link formation delay $T_{conn}$. The link formation delay components include the inquiry procedure delay $T_q$ and the paging procedure delay $T_p$: $T_{conn} = T_q + T_p$ \( (1) \)

Inquiry procedure delay $T_q$ is given by [32]:

$T_q = 2T_{sync} + T_{rb}$ \( (2) \)

where $T_{sync}$ refers to Frequency Synchronization delay and $T_{rb}$ refers to Random Backoff delay.

Now consider the connection delay for a new node to join the scatternet. Assume that within the node’s radio range there exits a master or slave and thus the new node could receive at least one response from the neighbor nodes at its first round Inquiry. Ignoring the paging delay $T_p$ because it is typically much less than Inquiry delay $T_q$, we can approximate the connection delay $T_c$ using the following
equation:

\[ T_c = T_w + 2T_{sync} + T_{rb} \] (3)

where \( T_w \) denote the amount of time consumed by the new node to discover a neighbor node that it decides to connect to. Given Inquiry timer set to 10.24s in our algorithm, \( T_w \) is a variable that ranges from 0 to 10.24s. According to [32], \( T_{sync} \) and \( T_{rb} \) are uniform random variables in \([0, T_{coverage}]\) and \([0, r_{max}]\) respectively, where \( T_{coverage} \) is 10ms (20ms) for the 16 (32) hop system, and \( r_{max} \) is 639.375ms. Thus the maximum connection delay for a new node could be 10.24s + 659.375ms = 10.899375s for the 32-hop system. Such a delay can be accepted by non time-critical applications.

If a new node could not discover any neighbor nodes present in the scatternet during the 10.24s Inquiry procedure, it will give up and try later.

Communication complexity is the number of messages sent between the devices. Since Bluetooth devices are usually small and low powered, minimizing the number of messages sent would be significant for Bluetooth devices. Messages used in scatternet formation process can be categorized into two groups:

1. Messages used by link establishment between two devices, i.e. packets sent during Inquiry and Page procedures.

2. Algorithmic messages - messages used by the scatternet formation algorithm.

In our algorithm, we employ asymmetric method for link formation, that is, inquiry or scan role of the node is pre-assigned. Since a node who performs Inquiry will become a master later, a newcomer would always become a master node even if it were trying to connect with a master. Actually this is undesirable. To solve
this problem, our algorithm employs Master/Slave role switching in this case, which inevitably introduces overhead message exchange between two nodes. Moreover algorithm messages also include the message sent when the algorithm is managing to adapt to the topology changes caused by the movement of nodes. Compared with some complex algorithms [33] which involve operations for optimization of scatternet topology, our simple scatternet formation has a very low communication complexity.

3.5 Simulation and Analysis of the Bluetooth Environment (RCBTEE)

Our algorithm implemented in the RCBTEE is designed to work in such a dynamic environment that nodes arrive and leave in incremental fashion. To make it efficient as well as simple, the inquiry or scan role of the node is pre-assigned, that is, each new node coming to the scene would perform Inquiry, and all the member nodes of the scatternet perform Inquiry Scan periodically.

Java simulations were performed to demonstrate the feasibility of the proposed scatternet formation algorithm under the desired scenarios.

Once a piconet or a scatternet around the primary master connected to the web server is formed, the Bluetooth participants of the formed piconet/scatternet can rely on it to communicate with the primary master. For these participants of the scatternet, the primary master performs like a gateway to the Internet, through which a remote control of local Bluetooth devices from the other side of the Internet could be realized. Another desired function for the primary master is locally controlling of the Bluetooth devices. To offer such services, primary master should have a exact connectivity list of the whole scatternet and maintain some state information in its memory for each local Bluetooth device.
Fig. 3.11 shows Interactive simulation of Scatternet formation. Circular radius shows the range of each master, labeled as "M"; slaves are labeled as "S". Unlabelled dots are devices not connected into any piconets or scatternets.

Since a master node has information about all slave nodes in its piconets, it would be responsible for sending the required information, such as the list of its connected slaves and its slaves' address, to the primary master connected to the Internet. When the primary master receives information packets from a master in the scatternet, it will update corresponding data in its memory at once. In case of any arrival or departure of nodes, the scatternet will reform again according to our algorithm. The masters that have new nodes add to or delete from their piconets should update its connectivity list and send the update to the primary master. And the new selected masters during scatternet reforming process should also report their lists to the primary master. Thus the primary master will have a global view of the scatternet and always hold up-to-date information of the Bluetooth devices present in the scatternet.
Another issue should be addressed for our local scatternet and primary master architecture is data routing between primary master and local Bluetooth devices. We employ a method similar to source-initiated on demand routing. Whenever a node in the scatternet desires to communicate with the primary master connected to the Internet, or vice versa, the source node creates a route to the destination on a demand basis. Once a route has been created, it is maintained until the destination becomes inaccessible along the path from the source or until the route is no longer needed. The primary advantage of this method is that nodes that are not on the selected path do not need to maintain routing information or participate in routing message exchanges. It is particularly significant for Bluetooth devices with power constraint and limited memory.

The master and each slave devices exist as threads. The state and parameters of each device are stored in a buffer (implemented as a ASCII text file) in the web site. A server is kept running in the web site, the home device connected to the internet and the web page updates are reflected on the buffer files.

A real-time update function updates any changes made to the home device or on the web page directly to the buffer files. The Home devices and the web page are then instantly updated from the buffer files. The update function only works on the webserver and the Master node connected to the Internet only when some change occurs onsite or in the remote control page(Fig. 3.12). Hence the overhead due to updates is negligible since it does not congest the Bluetooth scatternet. An encrypted password protection security feature is added to the web site to make it less vulnerable to misuse. Bluetooth's own protection mechanism protects the scatternet from any intruding Bluetooth enabled device from entering the scatternet. On entering the password, the web page is instantly updated and shows the current state of the devices. Since the file updates are immediate, the probability of deadlock
due to multiple writes is negligibly small. Moreover, in case of contention between onsite change of device state, and remote change request, priority is given to onsite change to avoid any deadlock or system crash.

Figure 3.12: RCBTEE Remote Control Centre

Figure 3.13: RCBTEE Scenario
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Max. No. of nodes</th>
<th>All nodes within communication range of each other</th>
<th>Support of Dynamic Join</th>
<th>Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCBTEE</td>
<td>No Limit</td>
<td>Yes</td>
<td>Yes</td>
<td>Tree</td>
</tr>
<tr>
<td>BTCP</td>
<td>36</td>
<td>Yes</td>
<td>No</td>
<td>Mesh</td>
</tr>
<tr>
<td>Bluetree</td>
<td>No Limit</td>
<td>No</td>
<td>No</td>
<td>Tree</td>
</tr>
<tr>
<td>Bluenet</td>
<td>No Limit</td>
<td>No</td>
<td>No</td>
<td>Mesh</td>
</tr>
<tr>
<td>LS01</td>
<td>No Limit</td>
<td>Yes</td>
<td>Yes</td>
<td>Mesh</td>
</tr>
<tr>
<td>Bluerings</td>
<td>Less than 40</td>
<td>Yes</td>
<td>No</td>
<td>Ring</td>
</tr>
</tbody>
</table>

### 3.6 Applications for RCBTEE

1. To remotely control home equipments, like set the microwave ON when returning home from office. Fig. 3.12 shows the Remote Bluetooth control centre web page. The web page is password protected. The web page allows users to remotely control the devices.

2. To remotely control office equipments. Fig. 3.13 shows A typical scenario of Bluetooth enabled home/office environment. Devices are connected to the master Laptop, connected to the Internet.

3. To know the status of equipments from the values of the webpage form.

4. Limitless applications on safety control.

5. Applications on process monitoring and control.

6. To give processing power to any electronic devices which are Bluetooth enabled. For example a lighting system connected to the system mentioned in RCBTEE can be given processing power, with all processing done at the Internet based on the data input from the device parameters. The lighting system can be switched on and off depending on the time of the day.
7. Remote Control of BT devices is an interesting BT and Internet application for pervasive computing and thin computing.

3.7 Discussions

A new prototype system is simulated for home/office networking with Bluetooth enabled devices. The primary contribution of this system is to economically have a remote control system for home/office devices arranged in a wireless network. The Scatternet algorithm presented compares well with current Scatternet algorithms (Table 3.1) and shows some improvement over many algorithms. The status of the devices in the network can be monitored by a remote web page, which can be viewed by any computer connected to the Internet. Also, the devices in the network can be controlled e.g. switched on/off, given certain inputs remotely via the web page. The possibilities and applications of such a system are tremendous.

Future work will be to port the web page enable it to be opened in any Mobile phone and thus control the Bluetooth devices even from a WAP enabled mobile phone which runs J2ME. Also, future work remains to add more functionality to the devices being controlled like adding some amount of processing power based on the input parameters. Another future direction will be to mix wireless cell phone activity (i.e. wirelessly tethered until disconnect) with Bluetooth Scatternets. We also want to look at Mobility handoff as a device leaves one piconet and enters another.
Chapter 4

EquiBlue: A Load Balanced Scatternet formation algorithm.

4.1 Introduction

Mobile Ad Hoc Networks (MANET) using Bluetooth technology has gained immense popularity with more and more devices fitted with Bluetooth. The MANET formed by Bluetooth is also known as Scatternet. It consists of connection of a Master device with a few (upto 7) slaves known as a piconet. Some Scatternet formation rules laid down by earlier research work [33],[9],[19] are:

1. Node type constraint: Each node is either a Master or a Slave, a subset of slave nodes may act as shared bridge nodes.
2. Connectivity constraints: Two slaves cannot be connected directly [9], and two Masters should not be connected directly [19].
3. Bridge node constraint: A bridge node must connect only two piconets [9].
4. A bridge node must not be a Master in any piconet in which it is active.
5. The resulting Scatternet should be fully connected [33],[9],[19].
In this Chapter we present a novel load balanced Access point aware Bluetooth Scatternet formation protocol. Fig. 4.1 shows a Mesh structured Scatternet, as formed by the EquiBlue Algorithm.

Figure 4.1: Mesh Structured Scatternet
4.2 Related Work

In [46] we proposed a Bluetooth algorithm for a simple Scatternet formation with the ability to be controlled remotely.

BTCP [19] has a two phase Scatternet formation protocol. In the first phase a coordinator is elected among the nodes and it has all the information about the other participating nodes. In the second phase the leader makes a decision about the role of each device and conveys the information to the devices. Masters PAGE the slaves which are in the connectivity list provided by the leader and the Scatternet is formed.

Law et. al. [33], [25] propose a one phase Scatternet formation algorithm which first partitions the network into independent piconets and then elects a super master which knows about the nodes. Devices alternate probabilistically between Seek and Scan states and get the role of a Seeker or a Scanner. Devices try to merge(), migrate() or move() depending on the how many devices it has under the scanning device and how many slaves are made collectively (both seek and scan). When a leader u running SEEK connects to a slave v running SCAN, procedure ConnectedQ is called. If v's other master is w, the piconets of u and w will try to merge if possible. The possibility depends on whether the piconets of u and w can be accommodated in a single piconet, then w and its devices become slaves in the piconet of u. This procedure is performed by Merge() function. Otherwise, some slaves are moved from u to w piconet by the procedure Migrate(). All the communication between u and w in Connected(), Merge(), Migrate() and Move() are via their common slave v.

Bluetrees [36] form a hierarchical tree structure. An inquiry procedure is run by each device which knows about the identifier to its one-hop neighbors, before
the protocol for connecting devices begins. At this stage, one node is assumed as a root. After designating a node as root, the protocol starts with the root sending page messages to all its neighbors trying to connect. The connected nodes are now the slaves of the root (master) piconet. The loop continues with the slaves, thus formed, connecting to their neighbors which form the slaves for the paging devices. The paging continues until the leaves of the tree are formed. The intermediate devices in the final tree excluding the root and leaves act as master-cum-slaves in two different piconets. Distributed Blue Trees is proposed as an extension to the Bluetrees [36]. In the second step where a node is chosen as a root, distributed Bluetrees chooses many nodes as roots. Each roots forming its own tree which in the final step gets connected as scatternet.

Bluenet [35] algorithm employs three-phases to construct a Scatternet in a distributed way, there is no designated root node and it can be carried out at each node, based only on the local knowledge of the node's neighbor. Bluenet's network is a flatter structure compared to the one formed by BlueTrees algorithm. Bluenet tries to spread the network resources as evenly as possible through out the scatternet so as to prevent communication bottlenecks. The concept of visibility graph is introduced in this paper. In phase 1 of the Bluenet algorithm, a visibility graph is formed by every device collecting the information about neighboring devices in the inquiry state. Later each device pages its neighbors randomly. By the end of phase 1, individual piconets are formed. In phase 2, each node now does INQUIRY procedure again trying to connect to another device. If any device gets connected to more than one node, it then acts as bridge node. Phase 3 is used to form a connected Scatternet. And at the end of phase 3, a connected network is formed with master informing slaves to open outgoing links. The disadvantages of this algorithm are that a device has to inquire twice in order to connect to the entire set of devices, and addition of nodes into the existing structure is difficult.
4.3 EquiBlue Algorithm

The EquiBlue Algorithm is schematically shown in the next few pages. Fig. 4.2 shows the Access point calling SEEK and starting INQUIRY procedure. Fig. 4.3 shows other nodes perform the biased probability function to determine whether they enter INQUIRY or INQUIRY SCAN. The biased probability function calculates probability based on a random function and the weight of the device (weight symbolizes the processing power, hence higher processing power devices have a higher weight value). Connection Window of device is set at this stage. Connection window of a Master device is the number of active slaves it currently has in its piconet. Fig. 4.4 shows devices which are determined to perform INQUIRY, start the procedure to determine devices in its range to form a piconet. Fig. 4.5 shows Master's form piconets, they connect with bridge slaves to other piconets. Maximum No. of bridge slaves is one for two neighboring piconets. Fig. 4.6 shows that in the first phase, Scatternet is formed. Fig. 4.7 shows that in the Scatternet, Masters send slavelist to neighboring Masters, in case of disparity in the number of slaves between the two neighboring piconets, Masters with more slaves will disconnect slaves if it is in radio range of neighboring Master. Fig. 4.8 shows Neighboring Master connects to the disconnected slave by PAGE message, since it already got the frequency and address information from the neighboring Master, thus avoiding the time expensive INQUIRY procedure.
Figure 4.2: EquiBlue Step 1

Figure 4.3: EquiBlue Step 2
Figure 4.4: EquiBlue Step 3

Figure 4.5: EquiBlue Step 4
Figure 4.6: EquiBlue Step 5

Figure 4.7: EquiBlue Step 6
In this section we present our load balanced Scatternet formation algorithm. Then we evaluate its performance at a later section.

Load Balanced Scatternet formation Algorithm:

1. Nodes form master and slave based on a biased probability and weight. The biased probability function calculates probability based on a random function and the weight of the device (weight symbolizes the processing power, hence higher processing power devices have a higher weight value). Connection window (CONWIN) of a Master device is the number of active slaves it currently has in its piconet.

   Device calls Seek when
   - Probability $> 0.4$ & weight $> 50$ or
   - Probability $< 0.4$ & weight $< 50$

   Else device calls Scan.

   Access point always calls Seek.

Set CONNECTION_WINDOW of Master proportional to weight of the device.
Heavyweight device: CONWIN = 5
Lightweight device: CONWIN = 3

2. After connection: If Master reaches CONWIN,
   It sends REFUSE message to Scanning slave
   Until p retries
   And sends address of slave and CONWIN to neighbor Masters through bridge slaves.

3. If Master has finished p retries
   or CONWIN of neighbor <> CONWIN +/- 1
   then CONWIN = CONWIN +1

   If CONWIN of neighbor Master <> +/- 2
   then CONWIN = CONWIN -1
   Disconnect an unshared slave

4. If Master receives an address of Slave that neighbor-Master has refused or send
   DISCONNECT to
   then, Master will send PAGE message to Slave

5. If Undiscovered Slave or Undiscovered Master
   then perform step 1 with changed Master/Slave state.

   Our Load balanced scatternet formation protocol can also work with dynamic environments when devices join and leave the scatternet.

   When a new device wants to join the scatternet, it performs the biased master-slave selection of step 1 and does step 2-4 to discover other neighboring devices and join to them.
When a unshared slave leaves the scatternet, the Master under whose piconet the slave was connected, knows about the disconnected slave during a polling attempt and updates its slavelist and routing tables.

When a shared slave leaves the piconet, both Master's send PAGE message to their slaves with the address and clock information of the other Master. The slaves perform PAGE SCAN and the first slave to connect to the other Master forms the shared slave. The other slaves trying to connect are send a REFUSE message.

When a Master device leaves the scatternet, all its connected slaves know about the disconnected status when their polling attempt fails. The slaves perform a PAGE SCAN (since the retired Master had given its neighborMaster address and clock information to its slaves. The Neighbour Master tries to connect to the disconnected slaves by PAGE (skipping the time expensive INQUIRY). In case the slavelist of the Master is > 7, then the Master ‘parks’ some of its slaves and connects the slaves. It then sends the extra (more than 7) slave addresses to its neighbor Masters. The neighbor Masters try to connect to the slaves by PAGE. As soon as the neighbor Master connects to the slave, the previous Master disconnects the slave and ‘unparks’ some of its parked slaves.

We do our time complexity and message complexity analysis similar to [9] and find that our algorithm has O(log n) time complexity and O(n) message complexity.

4.4 Simulation Results and Analysis

In this section, we present the results of our simulation using Simjava [34], a discrete event based simulation package for java. This allowed the use of object oriented programming making each node its own entity, running on a separate thread. Simjava[34] was used to simulate the sending and receiving of messages between nodes.
during discrete time intervals. Fig. 4.9 shows a snapshot of the Load balanced Access point aware Scatternet formation algorithm simulated in Simjava.

Simjava is quite a versatile tool. It is used to simulate an environment in which many objects, nodes in this case, frequently interact by sending messages to one another. Nodes are linked together by ports. These ports allow messages to be sent between connected nodes. The nodes in the environment derive from an entity class specified by Simjava. Deriving from the entity class makes each node have its own thread. A central system class is responsible for controlling the threads. This centralized system class also controls simulation time and delivers messages as appropriate to each node. It is aware of each node and can signal termination of the simulation.

The behavior of each node is implemented within the class that derives from the entity class. Simjava provides many useful actions that allow ease in implementation. Simply executing through a loop each time interval, each node performs
actions and changes state as necessary. Nodes can wait for a specified amount of simulation time, check for messages waiting, and send messages instantaneously or with a delay. This gives an excellent framework for communication between nodes. In combination with the implemented node behavior, this framework allowed the simulation of the Scatternet formation algorithm.

A nodeGenerator class generates randomly device co-ordinates, device ID, weight (representing processing power) and device clock. A Node class handles the functionality of Bluetooth nodes, it contains device states, logic for alternating between master and slave, message queues, slavelist, masterlist and neighbormasterlist, timing information and most of the logic needed for the simulation of the node as a Bluetooth node and implementing the EquiBlue algorithm. A Matcher class simulates lower level link control protocols. A Message class acts as a wrapper between information send between the nodes and between the node and the Matcher class. An InitSimMsg class is used to send the initial message from the Matcher to the nodes. Classes BlueFrame and BlueDraw are used to visually draw the workings of the Scatternet simulated as the algorithm proceeds.

![Graph](image)

**Figure 4.10: Total Number of Piconets Vs Number of Nodes**

Our results are the averages of 10 trials for up to 120 devices. We plotted the
results for an area of 30 X 30 metres, 40 X 40 metres, 50 X 50 metres.

Fig. 4.9 shows a snapshot of the load-balanced access point aware scatternet formation algorithm simulation. The M denotes Master, S denotes slave and AP denotes access points. The device ID is given after the (M/S/AP) label.

Fig. 4.10 show the number of piconets are rising linearly with linear increase of number of nodes. The number of piconets increase as the area increases for the same number of nodes, this ensures that the piconet size is kept even at the cost of
Fig. 4.13: Variation in the mean number of piconets as the number of piconets increases.

Fig. 4.14: The total Inquiry messages send in the algorithm the number of piconets to achieve a load balance.

Fig. 4.11 shows that the number of hops from end to end (scatternet diameter) increases linearly. The Scatternet diameter increases for a larger area with the same number of nodes. The Scatternet diameter is below 20 even for 120 devices.

Fig. 4.12 shows that the number of devices per piconet is between 4 and 5. This verifies our load balanced scatternet algorithm since it works effectively to balance out the load between the piconets so that the difference in slave devices is not very large between piconets.
Fig 4.13 shows a flat trend in the mean number of piconets as the number of piconets increase. This implies that as the number of piconets increase the network load is balanced evenly among the piconets between 4 to 5 devices per piconet.

Fig 4.14 shows the total number of Inquiry messages send as the number of nodes increases. The number of Inquiry messages increases as the area increases, since it means more piconets are formed and hence some of the Inquiry messages are rejected as the slave responds to one or two masters depending whether it is an unshared slave or a shared slave.

Fig 4.15 shows the total number of PAGE messages send increase linearly as the number of nodes increases.
Fig 4.16 shows the total number of algorithmic messages send increase linearly as the number of nodes increases. More algorithmic messages are send when the area of operation increases, since more piconets are formed and hence more inter-piconet messages have to be send to achieve the load balance mechanism.

### 4.5 Discussions

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Max. No. of nodes</th>
<th>All nodes within communication range of each other</th>
<th>Support of Dynamic Join</th>
<th>Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>EquiBlue</td>
<td>No Limit</td>
<td>Yes</td>
<td>Yes</td>
<td>Mesh</td>
</tr>
<tr>
<td>BTCP</td>
<td>36</td>
<td>Yes</td>
<td>No</td>
<td>Mesh</td>
</tr>
<tr>
<td>Bluetree</td>
<td>No Limit</td>
<td>No</td>
<td>No</td>
<td>Tree</td>
</tr>
<tr>
<td>Bluenet</td>
<td>No Limit</td>
<td>No</td>
<td>No</td>
<td>Mesh</td>
</tr>
<tr>
<td>LS01</td>
<td>No Limit</td>
<td>Yes</td>
<td>Yes</td>
<td>Mesh</td>
</tr>
<tr>
<td>Bluerings</td>
<td>Less than 40</td>
<td>Yes</td>
<td>No</td>
<td>Ring</td>
</tr>
</tbody>
</table>

Table 4.1: Comparison of EquiBlue with other Scatternet Algorithms

In this Chapter we introduced a load balanced Bluetooth Scatternet formation algorithm named EquiBlue. We described the algorithm in detail and did a simulation in a Java based discrete event simulator named Simjava[34]. We took numerous performance metrics using our simulation and showed that the EquiBlue algorithm achieves a good performance in equalizing the number of nodes per piconet. We compared our algorithm with numerous other Scatternet formation algorithms shown in Table 4.1. Our results are comparable to the results given in [25] while still achieving a good load balance factor. Two contributions of EquiBlue are achieving a load balanced Scatternet (where each piconet would not differ with other piconets by more than one or two devices), and biasing the Master/Slave role selection so that higher processing power devices get the role of Master. This would be very efficient in applications where a repository is kept on the Master or the Master offers a service and hence needs higher processing power.
Chapter 5

Blue-Fi: A Convergence Architecture for Bluetooth and 802.11 wireless networks

5.1 Introduction

Bluetooth [1], [3] 1.1 specification grew to include the formation of Personal Area Networks (PANs) which is narrower in scope of operation than WLAN. The standardization of PANs is carried is being carried out by the 802.15 working group [21]. The IEEE 802.11 standard [23], [22] for WLANs is the most widely used WLAN standard today. Co-existence between Bluetooth and 802.11 is a hot topic for research since both technologies have advantages and disadvantages. Co-existence of both seems to be the trend of the near future.

In this chapter we present a novel architecture for a hybrid Bluetooth-802.11 access point and algorithms for interoperability of Bluetooth and 802.11 on a software switching level. In section 5.2 we discuss some relevant research on interoperability in hybrid networks. In section 5.3 we propose algorithms for a hybrid network.
consisting of Bluetooth and 802.11 devices. We analyse the handoff latency in our proposed protocol in section 5.4. We investigate Co-existence issues of Bluetooth with 802.11 in section 5.5. In section 5.6 we conclude with some discussions and directions for future work on this subject.

5.2 Related Work

Kansal et al. [14] introduce a Handoff scheme for Bluetooth devices to allow mobility of devices in Bluetooth public access (BluePAC) environments. Baatz et al. [17] concentrates on handoff support for mobility with IP over Bluetooth. Perkins et al. [16] present a handoff scheme for mobile IP’s.


5.3 BlueFi Handoff Algorithm

In our hybrid Bluetooth/802.11 network we have access points that have both the Bluetooth and 802.11 antennae and physical interfacing devices/network cards. The devices can access the backbone 802.11 network resources either through the “802.11 AP- 802.11 Client” interface or through “Bluetooth(BT) AP- Bluetooth Client” interface. When the client uses its Bluetooth interface, the Access Point (AP) should be able to forward the Bluetooth packets to the backbone 802.11 network and the incoming packets from the 802.11 to the Bluetooth interface so that they can be passed on to the client. However, when the client uses the 802.11 interface, the packets are directly forwarded to the corresponding 802.11 interface in the AP and from there to the backbone network.
We propose to introduce a handoff algorithm through which the client can switch from one network to the other, either voluntarily (due to his power or bandwidth requirements) or because his mobility takes him out of the Bluetooth radio range. When this switching of interfaces occurs, the application layer must remain oblivious to this change and a lower software layer must be able to activate the appropriate interface that the client will use. For this purpose, we propose to introduce a software engine or daemon which we call layer of software control (LSC) in between the TCP/IP and the next lower layer in the hierarchy (LLC for 802.11 and PPP or L2CAP layer for Bluetooth) that would take care of the switching between the Bluetooth device and the 802.11 device both in the client as well as at the hybrid Bluetooth/802.11 AP. Conceptually this software engine is common to both the Bluetooth and 802.11 physical interfaces, as shown in Fig 5.1. This layer traps all the outgoing and incoming TCP packets, buffers them appropriately and then releases them to the lower layers of the stack (after deciding which interface to activate) so as to facilitate smooth handoff.

The application layer can directly send commands to LSC daemon. For example, when the user wants to voluntarily switch operation from one network to the other, he will simply send a specific command to LSC, which are implemented as APIs' and this will, in turn, activate that physical interface and divert all outgoing TCP/IP packets to it & listen for incoming packets from that interface. In other words, after receiving the commands from the application, it becomes the responsibility of this layer to initiate a manual handoff. In order to facilitate this, LSC must send some direct HCI commands to the Bluetooth physical chip. These direct commands are Get_Address (for getting PHY address and CLK information), Create_Connection (connect device and set scan mode), Write_Page_Timeout (set time spent on paging), Read_Scan_Enable (read device configuration regard-

![BlueFi Software Layer Concept](image)

**Figure 5.1: BlueFi Software Layer Concept**

Let us take two specific cases of voluntary handoff and a single case of natural forced handoff due to disconnection.

**A. User is in the radio range of both 802.11 & BT and wants to voluntarily switch from his currently running BT communication to 802.11, possibly because he needs a higher bandwidth:**

1. The LSC layer in client receives request command from the application layer to switch to 802.11.

2. LSC then sends a control packet destined for the LSC layer in the Bluetooth AP that this current client is initiating a manual handoff from BT to 802.11. The control packet also contains the physical address of the client machine.

3. Client LSC creates a buffer for the unsent TCP/IP packets during handoff.
4. The LSC layer at Bluetooth AP, after receiving the control packet creates a buffer for all unsent packets destined for the client address specified in the control packet and acknowledges the handoff request. The acknowledgement packet contains the channel information about the 802.11 AP where the client will listen for beacon frames (passive scanning) or send its probe request (active scanning), once it switches its interface to 802.11. Further, at AP, the address information for all such clients who wishes to switch to 802.11 network is kept in an address queue. For each item in the address queue, there exists a buffer of all unsent data packets from AP for that client.

5. When the acknowledgement packet reaches the LSC layer of the client, the client LSC deactivates the Bluetooth interface and activates the 802.11 interface. The 802.11 interface searches for available 802.11 AP by the scan method. Once the connection is established, all the unsent packets destined for the client physical address is sent by the AP. The unsent packets at the client are also sent. Reauthentication takes less time as the address of the client seeking handoff had already been sent to the AP previously by the LSC control packet.

B. User is in the radio range of both 802.11 and BT and he wants to voluntarily switch from his currently running 802.11-802.11 communication to Bluetooth-Bluetooth communication, possibly because he needs to save power:

Similar steps are followed as in Case A in steps 1 - 4, as during switchover from BT to 802.11. In step 5, BT interface directly goes to the R0 page scan mode (skips the time-expensive enquiry mode). Since the BT AP knows about the client address and clock information, it pages the client with these information until the client responds. If the client is not in radio range of BT, the BT AP will make four page attempts to connect to the BT interface of the client before giving up. This is explained below in the case when there is a natural handoff from BT to 802.11.
After connection is established, all the unsent packets destined for the client physical address is sent by the AP. The unsent packets by the client are also sent.

C. A client running BT connection, moving away from the BT AP so that it no longer lies in its radio range. Our design allows to seamlessly switch to the 802.11 network so the client does not feel any break in connection.

1. BT AP constantly polls the client when the client is accessing the backbone 802.11 network through BT interface. The clients respond to these poll packets.

2. The poll packets reach the LSC layer in both the AP and the client. Polling scheme is round robin, in which the AP (master) polls each connected BT (slave) device. Data requirements may be different for different clients, and the packet duration may be adjusted for this, using single slot packets for slaves with low data rate requirements and multi slot packets of length 3 or 5 for higher data rates.

3. To detect connection loss, the AP keeps a timer and if no reply occurs for the timeout period, \( T_{\text{polltimeout}} \), connection is assumed to be broken. At the mobile too, a similar procedure is followed, to detect loss of connection. At both the access point and the mobile clients, the timeout value, \( T_{\text{polltimeout}} \), is specified to be equal to the maximum number of slots that may pass between two successive turns. One poll round time takes \( s \times 2 \times 1 \) where \( s \) is the number of clients (slaves) attached to the AP (master). Assuming \( s = 7 \) (can vary from 1 to 7) and \( l=5 \) (maximum), \( T_{\text{polltimeout}} = 70 \) slots which is the worst case value. If the AP needs to communicate with more than 7 slaves, it can do so by instructing active slave devices to switch to low power park mode and inviting other parked slaves to become active in the piconet. This can be repeated to
allow a master to serve a large number of slaves.

4. Each Bluetooth AP finishes its poll round and checks if the address queue has any pending addresses of clients trying to connect to BT from 802.11. If there is such an element in the queue, the AP sends a HOLD message to all its connected clients to suspend their connection loss detection timers for a period of $T_{AP\_Page}$.

5. BT AP then pages the mobile client using the address and clock information received. The clients resume their connection loss detection poll timers either on the expiry of the $T_{AP\_Page}$, or if the AP sends a regular poll packet before that. $T_{AP\_Page}$ is the maximum time an access point spends on paging for a slave who switches over from 802.11 to BT.

All incoming data packets for all the BT clients must also be buffered so as to be delivered once the HOLD stage is over which is equal to $T_{AP\_Page}$. Since we configure the clients to use the R0 page scan mode, each page train needs to be attempted only once by the AP, which means that both the trains can be tried out in 32 slots. Four page attempts are made for robustness, leading to $T_{AP\_Page}$ equal to 128 slots, which is 80 milliseconds. Thus, the worst case time required to discover a break in BT connectivity is $T_{poll\_timeout} + T_{AP\_Page} = 70 + 128$ slots = 198 slot time (about 124 milliseconds). Since it is possible that the mobile clients move out of the BT range during the HOLD period, we do the following: as the clients will discover that they are out of range only when the HOLD period is over, they will start scanning for 802.11 AP's beacon after the HOLD period. At the same time, the Bluetooth AP will also discover that the particular client is out of the BT range only after the HOLD stage, i.e. both client and AP discover the switchover at the same time. The AP keeps track of the addresses of those Bluetooth clients which are detected as lost & then looks forward for a connection request from the client.
When a client moves away from the BT radio range, the loss of connection is detected at the first unsuccessful poll attempt because only one poll attempt can occur within $T_{\text{polltimeout}}$ period after a successful poll. Since the round robin poll always occurs within this time, live BT-BT connections will be able to refresh the timeout counter before timeout.

Once a poll timeout occurs, the LSC at AP immediately forwards the address of this client to the 802.11 interface AP so that when this client connects to 802.11 network, the authentication will take less time. Meanwhile, loss of connection will also be detected by the client due to lack of arrival of the poll packets from the Bluetooth AP after an interval of $T_{\text{polltimeout}}$. The client will then activate the 802.11 interface and go to the scan mode to discover the closest available 802.11 AP. When the 802.11-802.11 connection gets established, the unsent data packets from either end are sent.

One important consideration is the length of the address queue that we need to keep at both the 802.11 and Bluetooth side. The number of elements in this queue depends on the number of clients that switch between the networks in a specified time interval. In order to accommodate the fact that the queue can be very large in a region where the clients move very fast or they voluntarily switch network frequently, we follow the following procedure. The AP completes one round of polling all the attached clients and then checks to see if the address queue has any elements. If an element exists, it pages that device. The AP processes a single element of the queue at a time between each round of polling. This ensures that the existing clients are not kept waiting for indefinite time while the AP pages for new devices. On the 802.11 side, the AP’s are discovered by the mobile clients themselves from the beacons and so there does not arise any problem if the address queue is large.
Since we cannot modify the existing accepted TCP/IP headers or introduce new header for LSC because that would lead to serious backward incompatibility, we need to device some technique so as to distinguish LSC control packet from ordinary data packets. For this reason, we use the 6 unused bits in the TCP header between “TCP header length” field and “URG” flag. When all six of these bits are set to 1 (i.e., 111111), this signifies to the remote LSC engine that the packet is a dummy TCP packet sent by the transmitter and contains control information in the data area of the packet for handoff. A dummy TCP packet looks like an ordinary TCP/IP packet, but it is not sent by any client for data transfer. It is generated by the LSC alone & actually contains the following information bits in the data area that facilitates handoff :-

1. A single bit RQ/RES denoting whether it’s a client request packet or server response packet. When set, this signifies client request. Otherwise its server response.

2. Two bits HREQ signify whether a switchover from 802.11 to Bluetooth or vice versa is requested.
   00 => Client switching off both its interfaces
   01 => 802.11 to Bluetooth switchover is requested.
   10 => Bluetooth to 802.11 switchover is requested.
   11 => Not used.

3. Bits (variable) denoting address and clock information of the client. Relevant only when RQ/RES bit is set. It is used when the client switches over from 802.11 to Bluetooth, i.e., HREQ bit is 01.

4. Bits (variable length, depending on the physical medium used) CH denoting the channel information of the 802.11 AP that will be used by the clients when
switching over from Bluetooth to 802.11. It is relevant only when RQ/RES bit is reset.

When the six unused bits of the TCP header is not all set, it indicates that the packet is a normal TCP datagram sent by any mobile host for data communication. Table 5.1 shows the API's that are available with LSC which any application programmer may use to interface his program with LSC for seamless handoff.

5.4 Analysis of the Algorithm

We take up all the three cases discussed above one after the other. In each of these cases, we split the total handoff process into three phases, detection, search and execution.

<table>
<thead>
<tr>
<th>API</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activate_Bluetooth()</td>
<td>Activates Bluetooth interface</td>
</tr>
<tr>
<td>Activate_Wlan()</td>
<td>Activates 802 interface</td>
</tr>
<tr>
<td>Read_current_interface()</td>
<td>Returns the current interface</td>
</tr>
<tr>
<td>Send_control()</td>
<td>Sends control signal to interface</td>
</tr>
<tr>
<td>Read_control()</td>
<td>Reads control signal to interface</td>
</tr>
<tr>
<td>Send_control()</td>
<td>Sends control signal to interface</td>
</tr>
<tr>
<td>Send_Hold()</td>
<td>Sends Hold signal to interface</td>
</tr>
<tr>
<td>Create_buffer()</td>
<td>Creates local buffer</td>
</tr>
<tr>
<td>Acknowledge_request()</td>
<td>Acks request to interface</td>
</tr>
</tbody>
</table>

Table 5.1: API's available for the LSC for use

Case 1: When the client is operating using the Bluetooth-Bluetooth interface and voluntarily switches to 802.11.

a. Detection of Handoff:

Here, the AP detects that a handoff is requested by using the LSC control frame. Once it receives the LSC control frame, it sends its response in the next slot. Assuming the length of the packet to be of 5 slots, the total delay is 10-slot
time, which is about 6 milliseconds. Once the Bluetooth AP receives the response, it switches its interface from Bluetooth to 802.11 and goes to the scanning mode.

b. Search:

Searching by 802.11 clients for possible nearest AP takes place ONLY in a single channel as was sent by the acknowledgement packet when the client requested a handoff. Thus, even in the active scanning mode, the scanning time is reduced. According to the analysis done in [37], we note that

\[ \text{MinChannelTime} = \text{DIFS} + (aCW_{\text{min}} + \delta a\text{SlotTime}) \]

According to 802.11b standard,

\[ aCW_{\text{min}} = 31 \text{ slots}, \quad a\text{SlotTime} = 20 \text{ sec} \quad \text{and} \quad \text{DIFS} = 50 \text{ sec}. \]

Thus,

\[ \text{MinChannelTime} = 670 \text{ sec}. \]

Using the analysis in [37] \[ \text{MaxChannelTime} = 10.24 \text{ ms}. \]

Now, in our case, the client scans only one single channel. Assuming that there is equal probability for this channel to be unused as well as to be free,

Total Search Time, \( s = (T_u + T_e) / 2 \) where \( T_u \) = Time needed to scan a used channel and \( T_e \) = Time needed to scan an empty channel.

Now, \( T_u = 2T_d + \text{MaxChannelTime} \) & \( T_e = 2T_d + \text{MinChannelTime} \)

Using \( T_d = 65 \text{ ms} \) (for 20 stations), \( T_u = 140.24 \text{ ms} \) & \( T_e = 130.67 \text{ ms} \);

So, \( s = 135.5 \text{ ms} \)

C. Execution of Handoff:

Now worst-case handoff execution time is 3 ms using a Spectrum24 card.

Thus, total handoff latency: \( 6 + 135.5 + 3 \text{ ms} = 144.5 \text{ ms} \)

Case 2: When the client is operating using the 802 interface and voluntarily switches to Bluetooth.

a. Detection of Handoff:
Here, a control packet is sent by the LSC layer of the client to inform the 802.11 AP that it needs to initiate a handoff to Bluetooth. When the 802.11 AP acknowledges, the client switches over. If we ignore the time taken for the packets to travel, the overall delay in this case would be very low and hence we neglect the delay in this packet transmission.

b. Time taken to resume Bluetooth connection:
The time taken to resume connection depends on the number of elements in the address queue (i.e., clients who are willing to switchover from 802.11 to Bluetooth and have sent their requests). Since maximum number of active slaves in a piconet is 7, we assume that already 6 addresses are present in the queue when the 7th one arrives. Time taken to process each of the preceding 6 addresses as well one round polling through all the attached slaves is: $T_{AP,Page} + (1\times2\times5) + T_{AP,Page} + (2\times2\times5) + T_{AP,Page} + (3\times2\times5) + T_{AP,Page} + (6\times2\times5)$

Since one poll round = $s\times2\times1$ slots where $s$ = number of slaves, $1$ = slot length of the packet and $s$ increases as each slaves get attached to the AP (master).
The above expression equals $6 \times T_{AP,Page} + 210$ slots = $128 + 210 = 338$ slot-time = 212 milliseconds.

But, for all practical purposes, this value would be much less as probability for simultaneous 7 handoff requests coming from the clients for 802.11 to Bluetooth switchover is very less.

Now, for the current address (7th), paging takes approximately 16 slots in R0 scan mode. This is approximately 10 milliseconds. Thus, total delay in the worst case is 222 milliseconds.

Case 3: When the client is operating using the Bluetooth-Bluetooth interface and moving away from the Bluetooth radio range.
a. Detection of Handoff:
The time taken to detect loss of connection is $T_{\text{polltimeout}}$ when no paging attempt takes place. If however, paging attempt is taking place then worst case duration within which a break in connection will be detected is

$$T_{\text{polltimeout}} + T_{\text{AP\_Page}} = 70 + 128 \text{ slots} = 198 \text{ slot time (about 124 milliseconds)}$$

as was discussed before.

Thus, the time to detect break in connection is much less than that when a normal handoff between two 802.11 AP takes place using any 802.11b physical cards [37].

b. Search and Execution of Handoff:

Search for possible nearest 802.11 AP and then finally execution of the handoff procedure takes place by normal 802.11-802.11 handoff mechanism (active scanning by 802.11 clients). Worst case time required for search and execution using D-Link 520 802.11 interfaces is 290 millisecond, as is analyzed in [37].

So total handoff latency is $124 + 290 \text{ ms} = 414 \text{ ms}$

<table>
<thead>
<tr>
<th>Handoff Scenario</th>
<th>Average Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voluntary Bluetooth to 802.11</td>
<td>144.5 milliseconds</td>
</tr>
<tr>
<td>Voluntary 802.11 to Bluetooth</td>
<td>222 milliseconds</td>
</tr>
<tr>
<td>Disconnection: Bluetooth to 802.11</td>
<td>414 milliseconds</td>
</tr>
</tbody>
</table>

Table 5.2: Summary of Results of BlueFi

5.5 Bluetooth and 802.11 Coexistence

Bluetooth and 802.11 both operate in the same spectrum in the ISM band. Co-located operation of Bluetooth and 802.11 causes mutual interference and packet loss or introduction of bit errors. The comparison of Bluetooth and 802.11 is given in Table 5.3. Table 5.4 gives the comparison of Bluetooth and various 802.11 stan-
Various works have looked into coexistence problems and proposed some solutions to co-existence between Bluetooth and 802.11. IEEE 802.15 [40] has created task group 2 (TG2) to look into ways to mitigate the co-existence interference between Bluetooth and 802.11. Two class of co-existence mechanisms have been defined in [40]- collaborative and non-collaborative co-existence. Mobilan Corporation in [39] have proposed co-location without co-existence of Bluetooth and 802.11 to avoid interference. The work in [39] also proposes driver level switching and adaptive hopping that changes the hopping sequence based on interference. Chiasserini et. al. [41] have proposed two schemes V-OLA, to be applied to WLAN stations to avoid overlap between 802.11 traffic and Bluetooth voice packets, and D-OLA, to be executed at the Bluetooth devices to avoid overlap in frequency between 802.11 traffic and Bluetooth data packets. For BlueFi we conclude that driver level switching is the best method to avoid Bluetooth and 802.11 interference, since while Bluetooth is operating, 802.11 is in power save mode, and while 802.11 is operating, Bluetooth is in park mode.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Bluetooth Access</th>
<th>802.11 Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbol Rate</td>
<td>1 Msps</td>
<td>11 Msps</td>
</tr>
<tr>
<td>Spread Spectrum</td>
<td>FSSS</td>
<td>DSSS</td>
</tr>
<tr>
<td>Profiles</td>
<td>Almost Unlimited</td>
<td>LAN station or Access Pt</td>
</tr>
<tr>
<td>Data Distribution</td>
<td>No Restriction</td>
<td>Access point only</td>
</tr>
<tr>
<td>Current Consumption</td>
<td>60mA</td>
<td>300mA</td>
</tr>
<tr>
<td>Audio</td>
<td>PCM Channels</td>
<td>Voice over 802.3</td>
</tr>
<tr>
<td>Cable Replacement</td>
<td>Serial, USB, UART, Audio</td>
<td>802.3</td>
</tr>
<tr>
<td>Mobility Management</td>
<td>Master</td>
<td>Mobile Station</td>
</tr>
</tbody>
</table>

Table 5.3: Comparison of Bluetooth and 802.11b

We used the NIST simulator [38] to measure performance of Bluetooth with 802.11 interferer and performance of 802.11 with a Bluetooth interferer. The NIST simulator models the network in simple manner. A transmitter takes input bits and outputs a complex-valued signal. The channel model combines the desired
### Table 5.4: Comparison of Wireless standards.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Bluetooth</th>
<th>802.11</th>
<th>802.11b</th>
<th>802.11a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spectrum</td>
<td>2.4 GHz</td>
<td>2.4 GHz</td>
<td>2.4 GHz</td>
<td>5 GHz</td>
</tr>
<tr>
<td>Max Data Rate</td>
<td>723 kbps</td>
<td>1.2 Mbps</td>
<td>5 Mbps</td>
<td>32 Mbps</td>
</tr>
<tr>
<td>Connections</td>
<td>PTMP</td>
<td>PTP</td>
<td>PTP</td>
<td>PTP</td>
</tr>
<tr>
<td>Frequency Selection</td>
<td>FHSS</td>
<td>FHSS/DSSS</td>
<td>DSSS</td>
<td>OFDM</td>
</tr>
<tr>
<td>Authentication</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Fixed Networks</td>
<td>Any</td>
<td>Ethernet</td>
<td>Ethernet</td>
<td>Ethernet</td>
</tr>
<tr>
<td>CQDDR</td>
<td>Option</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Encryption</td>
<td>40 bit RC4</td>
<td>40 bit RC4</td>
<td>40 bit RC4</td>
<td>40 bit RC4</td>
</tr>
</tbody>
</table>

signal with that produced by an interference transmitter and adds distortion to the combined signal. The channel outputs this noisy signal, which is then input to the desired receiver, which uses the same technology (Bluetooth, IEEE 802.11b, etc.) as the desired transmitter. The receiver outputs a string of bits, which is compared to the input bits, accounting for the delay characteristics of the transmitter and receiver.

![Figure 5.2: BER Vs CIR in a Bluetooth Transceiver](image)

We did detailed simulations by varying the Carrier Interference Ratio (CIR) power and the frequency difference (df) measured in MHz, and measured the Bit Error Rate (BER).

Fig. 5.2 shows the Bit Error rate (BER) measurements while varying the Carrier Interference Ratio (CIR) and df in a Bluetooth Transceiver with a 1 Mbps signal with that produced by an interference transmitter and adds distortion to the combined signal. The channel outputs this noisy signal, which is then input to the desired receiver, which uses the same technology (Bluetooth, IEEE 802.11b, etc.) as the desired transmitter. The receiver outputs a string of bits, which is compared to the input bits, accounting for the delay characteristics of the transmitter and receiver.
802.11 interferer. The Carrier to Noise (CNR) ratio is kept at 30dB. The BER decreases when the CIR decreases for the same frequency offset. The BER decreases when the frequency offset increases ($df$ increases) for the same CIR. This implies that as the carrier power signal becomes stronger with respect to the interferer signal, the error rate decreases for the carrier signal. Also, the figure shows that with increase in frequency difference between the carrier signal and the interferer, the error rate is decreased.

Fig. 5.3 shows the Bit Error rate (BER) measurements while varying the Carrier Interference Ratio (CIR) and $df$ in a Bluetooth Transceiver with a 11 Mbps 802.11 interferer. The Carrier to Noise (CNR) ratio is kept at 30dB. The BER decreases when the CIR decreases for the same frequency offset. The BER remains the same for different frequency offset for the same value of CIR when the interferer
802.11 (11Mbps) Transceiver with Bluetooth Interference

Fig. 5.4 shows the Bit Error rate (BER) measurements while varying the Carrier Interference Ratio (CIR) and df in a 802.11 Transceiver at 1 Mbps with a Bluetooth interferer. The Carrier to Noise (CNR) ratio is kept at 35dB.

802.11 is at 11 Mbps.

Fig. 5.5 shows the Bit Error rate (BER) measurements while varying the Carrier Interference Ratio (CIR) and df in a 802.11 Transceiver at 11 Mbps with a Bluetooth interferer. The Carrier to Noise (CNR) ratio is kept at 35dB.

Fig. 5.6 shows the Bit Error rate (BER) measurements while varying the Carrier Interference Ratio (CIR) and df in a Bluetooth Transceiver with a 802.11 interferer at 1Mbps and at 11 Mbps. The Carrier to Noise (CNR) ratio is kept at

69
Figure 5.7: 802.11 Transceiver at 1Mbps and 11Mbps with Bluetooth Interferer

30dB. The comparisons show that a higher Bit Error Rate is introduced when the interferer bandwidth is increased from 1Mbps to 11 Mbps with the same CIR and frequency offset.

Fig. 5.7 shows the Bit Error rate (BER) measurements while varying the Carrier Interference Ratio (CIR) and df in a 802.11 Transceiver at 1Mbps and at 11 Mbps with a Bluetooth interferer. The Carrier to Noise (CNR) ratio is kept at 35dB. The comparisons show that a higher Bit Error Rate is introduced when the 802.11 Transceiver bandwidth is increased from 1Mbps to 11 Mbps with the same CIR and frequency offset.
5.6 Discussions

In this Chapter we introduced Handoff algorithms' for different scenario's of switching from a Bluetooth AP- Bluetooth Client connection to 802.11 AP - 802.11 Client and vice versa. We have modified the current protocol stack of Bluetooth and 802.11 to introduce a new layer LSC for interoperability. We have designed the header for messages in the LSC and shown the delay analysis on various handoff's (Table 5.2). We propose to implement the LSC daemon as an external kernel module that can be attached using 'insmod' command in UNIX. The exact implementation and evaluation of this daemon as kernel module is left as a future extension of this work in another project. We investigated the Co-existence mechanisms for Bluetooth and 802.11 and through simulations found the different error rates while varying the Carrier Interference ratio and the frequency offset.
Chapter 6

BlueMobile - A Mobile IP based Handoff system for Bluetooth, 802.11 and GPRS links

6.1 Introduction

While Bluetooth technology [1], [3], [4] has become popular as a market leader for short range wireless networks, the IEEE 802.11 standard [24], [21], [22], [23] for WLANs is the most widely used WLAN standard today. The standard uses the carrier sense multiple access (CSMA), medium access control (MAC) protocol with collision avoidance (CA). Both Bluetooth and 802.11 physical coverage is limited because of the engineering constraints of the underlying radio technology. On the other hand, GPRS (General Packet Radio System) [53], [54] is the packet mode extension of GSM and is a prevalent cellular technology that has a wide area of coverage.

Bluetooth, IEEE 802.11 and cellular network technology like GPRS have properties complementing one another in different amplitudes of power consum-
tion, area of operation and data rate. While Bluetooth comes with low power consumption, reliable connectivity, low bandwidth (in the order of 721 kbps) and a small area of operation (about 10 meters) with only a maximum of 7 slaves per master device, 802.11 provides wider area of operation, bandwidth in the order of 11 Mbps, coupled with higher power consumption. On the other hand, while 802.11 supports data rate from 1 to 54 Mbps and can cover up to a few thousand square meters, GPRS technology offers limited data rates from 64 kbps to 2 Mbps [44] but a much wider area of coverage with all-over connectivity. Thus these three technologies if converged, can actualize a system which will enable roaming users to smoothly switch between technologies and thus to use all the advantages bestowed by these three types of network. With this idea in mind, we designed a handover system called BlueMobile. BlueMobile integrates Bluetooth, 802.11 and GPRS technologies, by introducing a simple extension to the already existing Mobile IP implementation [49], [51]. Mobile IP is an extension to the TCP/IP protocol suite and takes gives the wireless access networks the provision to provide nodes the ability to roam across multiple IP subnets while assuming the same network address and maintaining network layer connectivity.

The task of designing the network architecture for a handoff system [14], [15], [50], [52] switching between 3 different technologies is challenging since the aim is smooth interaction both from the end user and network operator’s viewpoint.

In this chapter we describe an approach to the design of an integrated Bluetooth/802.11/GPRS network architecture. We introduce Mobile IP elements to various networks and using a simple extension to the Mobile IP, we integrate the three networks. We also present algorithms for handoff between the networks in various situations. Finally we analyze the performance and delay of our Handoff algorithms.
6.2 Related Work

In [46] we proposed a Bluetooth algorithm for fast scatternet formation with the ability to be controlled remotely. In [47] we proposed an architecture for co-existence of Bluetooth and other wireless networks. In [55] we presented a novel convergence protocol between Bluetooth and 802.11 through the introduction of a Software Layer (LSC) in the protocol stack of Bluetooth and 802.11. The present work is in continuation of the previous works.


6.3 Mobile IP

6.3.1 Mobile IP Architecture

Traditional IP technology can't support mobility in the IP layer. IETF defines Mobile IP. Mobile IP introduces two network entities: home agent (HA) and foreign agent (FA) to manage mobility. When MH is in its home subnet (its initial subnet), it uses normal IP protocol to communicate. While it enters the foreign subnet (other subnets except home subnet), it acquires an IP address, called care of address (COA). It then sends registration message to HA to inform HA its current location, COA. The data packets sent by correspondent host (CH) to MH arrive at MH's home subnet by normal IP routing. HA captures these packets on behalf of MH, and encapsulates them with new IP header, whose destination address is COA, source address is HA (it is tunneling). Then the encapsulated packets are forwarded
to MH's COA. FA or MH restores the original IP packets. Data packets from MH to CH are routed normally. The flow of data transmission is illustrated as Fig. 6.1.

Figure 6.1: IP datagram flow to and from a Mobile Host using Mobile IP

When MH enters new subnet, it needs handoff. It acquires a new COA and registers it to HA again, so that HA can correctly forward IP packets to it. During the time between MH leaving its old foreign subnet and HA receiving MH's new registration request message, because HA doesn't know MH's current COA, it still forwards those packets whose destination address is MH to the old FA, and these packets will be dropped by the old FA. It is possible that the connection will be disrupted. If the distance between the MH and HA is a bit long, the disruption time will be large. In this case, decreasing handoff delay and packet loss is the crucial issue for Mobile IP handoff. Mobile IP is proposed to support mobility in computer network. But because of its characteristic of easy realization, Mobile IP can be used in many wireless networks to support mobility.

Fig. 6.1 shows the IP datagram flow between a mobile host connected to a foreign network and its communicating element in the internet. Datagrams from the mobile host to the communicating element are routed via path 1 through the Host.
to its home network. But Datagrams from the Internet are tunneled by the Home Agent to the Care-of-Address of the Mobile Host through Path 2 via the Foreign Agent.

6.3.2 Mobile IP Adaptation for Bluetooth, 802.11 AND GPRS

In order to support mobility between the two networks, we use a peer network structure. That is, GPRS and Bluetooth access Internet as peer networks, and implement the function of Mobile IP respectively. In the Bluetooth network, we assume that the existing architecture is enhanced by network elements using the concepts from the BLUEPAC IP [42], which is based on ideas from Mobile IP. Also, the GPRS network has Mobile IP components for supporting handoffs. In GPRS network, we propose to implement the HA function at GGSN. When MH whose home network is GPRS moves to a foreign network (it is possibly not GPRS, such as Bluetooth), it registers to HA (GGSN) its current COA through the FA at the foreign network. GGSN(Fig. 2.8) checks all the IP packets that came from outside Internet. Once there are some packets whose destination is MH, it acts as HA, that is, it re-encapsulates these IP packets and forwards them to MH by tunnel.

We can also implement the FA function at GGSN(gateway GPRS support node), but we propose to implement it at SGSN. Then the FA function can be distributed to the SGSNs(serving GPRS support node), but not centralized at GGSN, which can alleviate the burden of GGSN.

When MH moves to GPRS network, which is a foreign network to it, GGSN will assign an IP address to it (assuming IPG). IPG can be a private IP address, but also can be a public one. At this time, SGSN acts as the FA of MH, so it broadcasts the Agent Advertisement messages to MH [17]. MH registers the IP address of SGSN (assuming as IPS) as its COA to the HA. SGSN relays this registration
message, and records a mapping in its database: <MH, IPG>.

When HA receives the registration message, it forwards the data packets belonging to MH to SGSN. When SGSN receives these packets, it looks up in the database and finds the mapping of MH. It de-encapsulates these packets, and re-encapsulates them to new IP packets, whose destination address is IPG and source address is IPS. SGSN forwards the new packets to MH using GPRS routing mechanisms. At last, MH de-encapsulates the IP packets and restores the original IP packets. The packets from MH to CH are sent to SGSN firstly, and then are tunneled to GGSN. They are forwarded to Internet by the GGSN at last.

When MH moves in the service area of a SGSN, it only considers handoff between different BSSs (This is a problem of link layer handoff.); when it moves between different SGSNs, it should take the Mobile IP handoff. It should register the new SGSN to its HA.

The mobility support in Bluetooth is comparatively simple. Bluetooth itself defines OSI layer 1 (baseband radio) and OSI layer 2 (connection setup, link management), so it only need add the layer 3-Mobile IP function to Bluetooth network: adding the HA and FA module using hardware or software in the fixed network it connects. The HA and FA function can be implemented in a router or a host or an access point. Similarly, in a 802.11 network, the HA and the FA module are implemented in an Access point, router or a Host. In our BlueMobile architecture, we formulate 802.11 as the home network with the Home Agent (HA).

The Foreign Agent (FA) in the GPRS and Bluetooth network have the functionality of a DHCP-server [48] to assign Care-Of-Addresses (COA) from a pool of locally available IP addresses. The COA's are assigned for a period only, after that
they are reclaimed by the FA for reuse. The Mobile Host has to renew subscription to a particular COA if it wants to keep it for a longer time.

Figure 6.2: The BlueMobile network architecture

6.4 Mobile IP Handoff Algorithm between Bluetooth, 802.11 & GPRS

Let us take two specific cases of voluntary handoff & a case of forced handoff due to disconnection.
A. User is in the radio range of both GPRS & BT and wants to voluntarily switch from his currently running BT communication to GPRS, possibly because he needs a wider area of connection (Fig. 6.2): 

We introduce a polling scheme whereby the Bluetooth Access Point (AP) constantly polls the client when the client is using the Bluetooth network. The clients also must respond to these poll packets, even when they have no data to transfer. Though the Link Supervision timer is present on the link layer in Bluetooth protocol stack that detects broken links, we do not depend on this timer. This is because, the default timeout value specified in the Link Supervision Timer is 20 seconds, which introduces a very large handoff latency when considering a handoff between GPRS and Bluetooth. When the MH wants to switch over to GPRS, it must inform its old Bluetooth FA that it intends to switch over. It does this by sending some special control bits along with these poll acknowledgements. Immediately, the FA in Bluetooth sends a control packet to the HA in WLAN giving the current address of the client that wishes to switch over to GPRS and acknowledges the switchover to the mobile node using another control packet. This sending of acknowledgement and transmission of control packets to HA takes place simultaneously or within negligible time interval so that for all practical purposes, we may safely neglect this latency.

When the Host Agent receives this message, it begins to put all the unsent packets intended for this MH in a FIFO buffer. The size of the buffer depends on the maximum latency of the GPRS-Bluetooth handoff and also on the transmission datarate.

Once MH decides to enter the GPRS network, it changes interface to GPRS and sends Host Agent in the WLAN network the formal registration message through the GPRS Foreign Agent to confirm the Handoff. After receiving the final registration message, the HA tunnels the IP packets to the GPRS Foreign Agent from the FIFO buffer to the Mobile Host. The FA of GPRS then tunnels these packets back
to MH. Similarly, all packets that are sent by the mobile node are first received by the FA of the GPRS network and is then tunneled over to the HA which then sends these packets to its true destination.

![Diagram](image)

**Figure 6.3:** MH moves from GPRS to Bluetooth

**B. User is in the radio range of both GPRS and BT and he wants to voluntarily switch from his currently running GPRS communication to Bluetooth-Bluetooth communication, possibly because he needs to save power or GSM airtime (Fig. 6.3):**

When the MH wants to switch over to GPRS, it must inform its old GPRS FA that it intends to switch over before activating its Bluetooth interface. The GPRS FA immediately informs this to the node's HA along with its address information. The HA immediately stops sending packets to the old GPRS FA and creates a buffer
for all unsent packets destined for this Bluetooth mobile node. The size of the buffer as usual depends on the maximum latency of the GPRS-Bluetooth handoff and also on the transmission datarate. The 802 HA informs all Bluetooth FA in range that a device with that address wants to register itself with the Bluetooth network. The Bluetooth FA sends control packets to the Bluetooth AP's in the network which in turn starts paging for a device using that address. This saves precious time on INQUIRY procedure. The paging attempt is tried four times and if the device is not found, it is assumed that some other AP must have discovered the Bluetooth device.

C. A client running BT connection, moving away from the BT AP so that it no longer lies in its radio range. Our design allows seamlessly switching to the GPRS network so the client does not feel any break in connection.

This will be similar to case A. The only difference here will be that the MH will discover that it is out of the Bluetooth range only when it no longer receives the poll packets from the Bluetooth AP. The Bluetooth AP will also discover that MH has moved away from its range when it does not receive the acknowledgement packets in response to its poll packets. Immediately, the FA in Bluetooth sends a control packet to the HA in WLAN giving the current address of the client that is missing. When the Host Agent receives this message, it begins to put all the unsent packets intended for this MH in a FIFO buffer until HA receives the registration message from MH through FA of GPRS that it has successfully registered itself with GPRS network.
6.5 Analysis of the Algorithm

In our design of BlueMobile, we have considered only standard technology features, like INQUIRY and PAGE procedures in Bluetooth. In the following, we analyze the handoff duration and give an estimate of the handoff delay in two cases: of the voluntary switch from Bluetooth to GPRS or 802.11 and vice versa, and in another case of the disconnection from Bluetooth hotspot due to the Mobile Host going away from the region of Bluetooth coverage.

Case 1: Voluntary Bluetooth to GPRS handoff

Here, the FA in the Bluetooth Access Point detects that a handoff is requested by using the control frame of the message sent by the Mobile Host Decision module. Once it receives the control frame, it sends its response in the next slot. Assuming
the length of the packet to be of 5 slots, the total delay is 10-slot time, which is about 6 milliseconds. Once the Bluetooth AP receives the response, \textit{FA-BLUETOOTH} forwards the Request message along with the Mobile Host address to the HA\textsubscript{802.11} in the Home network. The HA\textsubscript{802.11} responds to the request and sends an advertisement to the FA\textsubscript{GPRS} within 2 ms of notification time [45]. Mobile-IP responds to this advertisement by invalidating previous agent advertisements and sending a registration request within 2 milliseconds. The GPRS foreign agent responds with registration reply after approximately 800 to 1100 milliseconds. The length of the duration corresponds to the round trip time on the GPRS link. The registration reply completes the handoff and further packets are sent over the GPRS network. Thus total handoff latency is \(= (6 + 2 + 2 + 800/1100)\) ms = 810 - 1110 milliseconds.

\textbf{Case 2: Voluntary GPRS to Bluetooth handoff}

Here, the FA in the GPRS station detects that a handoff is requested by using the control frame of the message sent by the Mobile Host Decision module. Once it receives the control frame, it forwards the request to the HA\textsubscript{802.11}. The HA\textsubscript{802.11} responds to the notification with 2 milliseconds and sends the address of the Mobile Host to the all available Bluetooth Foreign Agents in range (since the 802.11 HA would not know which Bluetooth FA the current MH is close to and there can be more than one Bluetooth hotspots. The Mobile IP responds to this notification within 2 milliseconds. The \textit{FA-BLUETOOTH} in turn instructs the \textit{AP-BLUETOOTH} to PAGE (T\textsubscript{AP-Page}) the Mobile Host using that address; thus saving the time expensive INQUIRY procedure as it already has the Bluetooth Device Address. The round trip time for WLAN is approximately 250 milliseconds. T\textsubscript{AP-Page} = 128 slot time = 2.5 milliseconds (min) for mobile in Scan Repition(SR) Mode R0; 1.28 seconds (average) for mobile in SR mode R1 and 2.56 seconds (maximum) for mobile in SR mode R2. The paging procedure completes the handoff and further packets
are sent over the Bluetooth Network.

So delay in this case is:
Minimum delay = 2 + 2 + (250 or 2.5) = 254 milliseconds;
But latency in the GPRS links is 400 - 700 milliseconds. So the mobile node will keep receiving some out of order packets upto 400 - 700 milliseconds on its GPRS interface. If the notification request is send well in advance then there is no packet loss during the handoff.
Average delay = 2 + 2 + (250 or 1280) = 1284 milliseconds
Maximum delay = 2 + 2 + (250 or 2560) = 2564 milliseconds.

Case 3: Voluntary Bluetooth to 802.11 handoff

This situation is shown when the Bluetooth and 802.11 coverage areas overlap (Fig. 6.4). This case is similar to the handoff from Bluetooth to GPRS, the FA in the Bluetooth Access Point detects that a handoff is requested by using the control frame of the message sent by the Mobile Host Decision module. Once it receives the control frame, it sends its response in the next slot. Assuming the length of the packet to be of 5 slots, the total delay is 10-slot time, which is about 6 milliseconds. Once the Bluetooth AP receives the response, FA_{\text{BLUEETOOTH}} forwards the Request message along with the Mobile Host address to the HA_{\text{802.11}} in the Home network. The HA_{\text{802.11}} responds to the request within 2 ms of notification time [45].

Searching by the mobile 802.11 clients (which switched over from Bluetooth) for possible nearest AP takes place ONLY in a single channel as was sent by the acknowledgement packet when the client requested a handoff. Thus, even in the active scanning mode, the scanning time is reduced.
According to the analysis done in [37], we note that

$$\text{MinChannelTime} = \text{DIFS} + (aC\text{Wmin} + a\text{SlotTime}).$$
According to 802.11b standard, 
\( aCW_{\text{min}} = 31 \) slots, \( aSlotTime = 20 \) sec and \( DIFS = 50 \) sec.

Thus \( \text{MinChannelTime} = 670 \) sec.

Using the analysis in [37] \( \text{MaxChannelTime} = 10.24 \) ms.

Now, in our case, the client scans only one single channel. Assuming that there is equal probability for this channel to be unused as well as to be free,

Total Search Time, \( s = \left( \frac{T_u + T_e}{2} \right) \) where \( T_u = \) Time needed to scan a used channel and \( T_e = \) Time needed to scan an empty channel.

Now, \( T_u = 2T_d + \text{MaxChannelTime} \) & \( T_e = 2T_d + \text{MinChannelTime} \)

Using \( T_d = 65 \) ms (for 20 stations), \( T_u = 140.24 \) ms & \( T_e = 130.67 \) ms;

So, \( s = 135.5 \) ms

Now worst-case handoff execution time is 3 ms using a Spectrum 24 card.

Thus, total handoff latency is: \( 6 + 2 + 135.5 + 3 = 146.5 \) ms

**Case 4: Bluetooth to GPRS handoff due to disconnection**

This case is similar to Case 2: voluntary handoff to GPRS from Bluetooth. The only addition is the minimum time taken to detect loss of connection is \( T_{\text{polltimeout}} \) when no paging attempt takes place. If however, paging attempt is taking place then worst case duration within which a break in connection will be detected is \( T_{\text{polltimeout}} + T_{\text{AP Page}} = 70 + 128 \) slots = 198 slot time (about 124 milliseconds), since we assume that the Bluetooth AP does not send poll packets when it is paging for new devices to connect to the piconet.

Thus total delay is \( 124 + (6 + 2 + 2 + 800/1100) = 934 \) milliseconds to 1234 milliseconds.
Case 5: Voluntary 802.11 to Bluetooth handoff

This occurs when 802.11 and Bluetooth hotspot areas overlap as shown in Fig. 6.4.

This case is similar to Case 1 in which GPRS connection handoffs to Bluetooth.
Minimum delay = 2 + 2 + (250 or 2.5) = 254 milliseconds
Average delay = 2 + 2 + (250 or 1280) = 1284 milliseconds
Maximum delay = 2 + 2 + (250 or 2560) = 2564 milliseconds.

Case 6: Bluetooth to 802.11 handoff due to disconnection

This case is similar to Case 3: voluntary handoff to 802.11 from Bluetooth. The only addition is the time taken to detect loss of connection is $T_{polltimeout}$ when no paging attempt takes place. If however, paging attempt is taking place then worst case duration within which a break in connection will be detected is $T_{polltimeout} + T_{AP\_Page} = 70 + 128\text{ slots} = 198\text{ slot time (about 124 milliseconds)}$ as was discussed earlier.
Thus total delay is $124 + (6 + 2 + 135.5 + 3) = 270.5\text{ milliseconds}$.

<table>
<thead>
<tr>
<th>Handoff Scenario</th>
<th>Average Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voluntary Bluetooth to GPRS</td>
<td>810 - 1110 milliseconds</td>
</tr>
<tr>
<td>Voluntary GPRS to Bluetooth</td>
<td>1284 milliseconds</td>
</tr>
<tr>
<td>Voluntary Bluetooth to 802.11</td>
<td>146.5 milliseconds</td>
</tr>
<tr>
<td>Disconnection: Bluetooth to GPRS</td>
<td>934 - 1234 milliseconds</td>
</tr>
<tr>
<td>Voluntary 802.11 to Bluetooth</td>
<td>1284 milliseconds</td>
</tr>
<tr>
<td>Disconnection: Bluetooth to 802.11</td>
<td>270.5 milliseconds</td>
</tr>
</tbody>
</table>
6.6 Discussions

In this chapter we introduced Mobile-IP based Handoff algorithms' for different scenario's of switching from a Bluetooth connection to 802.11 connection or GPRS connection and vice versa. We have introduced certain elements of Mobile IP to the GPRS network. We have shown the delay analysis on various handoff's(Table 6.1 provides a summary of our results). The delay parameters suggest that BlueMobile system can handle audio and video connectivity with a jitter during handoff. Also, TCP/IP connection can be supported over BlueMobile since the typical maximum retransmission timeout is 1 second. Thus we see that given proper modifications to the already existing Bluetooth/GPRS/WLAN architectures, we may allow any mobile device with hardware interfaces for all the three types of network to seamlessly operate in any network domain. Thus a single mobile device can be operated in a large area depending upon the network available without loosing connectivity. Further work on the hardware and interference aspects of Bluetooth, GPRS and 802.11 would be needed for hardware interoperability.
Chapter 7

Conclusions

7.1 Summary

Bluetooth technology has become very popular nowadays. More and more devices like cell-phones, laptops, pda's are equipped with inbuilt Bluetooth chips. To support 802.11, already 802.11 hotspots with numerous access points are deployed extensively. For GPRS networks, GSM cellular networks have already established ubiquitous coverage. We foresee that in the near future Bluetooth hotspots will gain immense popularity and widespread deployment.

In this thesis we investigated current Bluetooth Scatternet algorithms. We found some deficiencies in the treatment of Scatternet algorithms, as the current ones were not customized in regard to issues with Bluetooth hotspots. Also, Load balancing in Scatternet formation has hitherto not been studied in detail. Hence we presented a load balanced Scatternet formation algorithm with respect to usage in Bluetooth hotspots. In the load balanced algorithm, each Bluetooth device is assigned a weight depending on their processing power. A laptop would be assigned a higher weight than a pda, which in turn would be assigned a higher weight than a cell phone. A higher weight device would be able to function as a master and accommodate more number of devices as slaves, than a relatively lower weight device.
For devices with similar weights, the neighboring piconets would try to minimize the differences in piconet size. We have evaluated our system with detailed simulation and found that our algorithm performs well with respect to the current Scatternet algorithms. Also, we proposed a novel remote control and querying scheme for Bluetooth devices arranged in a Scatternet with one or more devices having access point capability. Such a remote control system has hitherto not been investigated in detail.

Then we investigated the issues with interoperability of Bluetooth with 802.11 and GPRS technologies. The interoperability of Bluetooth with GPRS has hitherto not been studied in detail. We propose BlueFi - a convergence system for Bluetooth and 802.11. BlueFi conceptualizes a software layer in the protocol stack of Bluetooth and 802.11. However, to keep backward compatibility by not modifying the protocol stacks of Bluetooth and 802.11, we propose implementation of BlueFi using a system daemon that traps packets from the Bluetooth and 802.11 interfaces and sends it to the protocol stack after processing it. We did a detailed analysis of my system with various delays involved. Then we proposed BlueMobile, a Mobile IP based solution for interoperability of Bluetooth with 802.11 and GPRS. By introduction of Mobile IP elements to the architecture of Bluetooth, 802.11 and GPRS, these technologies can become Mobile IP compatible. We analyzed the various situations for handover and did a detailed delay analysis involved in various situations. The analysis demonstrates that our interoperability system helps achieve good performance and can be deployed in current devices with interactive audio, streaming video and TCP/IP data requirements.
7.2 Future Work

Our Bluetooth Scatternet algorithms and internetworking architecture provides many issues for further investigation and research.

First, issues with interoperability of Bluetooth with CDMA networks should be studied for possible practical convergent systems.

Second, in EquiBlue, the effect of clustering should be studied with respect to Scatternet formation. An overlay clustering on top of the existing Scatternet formation would open research issues for optimization of resource and routing strategies.

Third, access point placement in a hotspot supporting Scatternet formation is an open research issue in Bluetooth technology. The placement of access points can include static placement and mobile access points to address the coverage area. Also, nodes within one hop of an access point can be made a virtual access point and neighboring nodes of the virtual access point can get resources and a route to the access points.

Fourth, the use of Agents in the use of Scatternet formation is an emerging research topic. Agents like Grasshopper or Wave can be used in a Scatternet for efficiently solving many problems related to routing, resource allocation and discovery, conflict resolution in allocation of piconet frequency.

Fifth, during Bluetooth internetworking with 802.11 and GPRS technologies, the issue of security needs to be looked into. Since this thesis focuses on performance issues, security analysis has been left out. For example, during handover's from one network to another, security authentication mechanisms should be formulated.
Sixth, in BlueFi, implementation issues for a system daemon supporting internetworking of Bluetooth and 802.11 should be investigated.

Seventh, in BlueMobile, a prototype implementation might reveal some extra delays, unaccounted for, in a theoretical analysis as done in our thesis.

Finally, a hardware design for Mobile IP elements for Bluetooth should be done, or a software module can also emulate the function of Home Agent and Foreign Agent in a Bluetooth network.
Bibliography


## Appendix A

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACL</td>
<td>Asynchronous Connection-Less</td>
</tr>
<tr>
<td>AP</td>
<td>Access Point</td>
</tr>
<tr>
<td>BD</td>
<td>Bluetooth Device</td>
</tr>
<tr>
<td>BER</td>
<td>Bit Error Rate</td>
</tr>
<tr>
<td>BluePAC</td>
<td>Bluetooth Public Access</td>
</tr>
<tr>
<td>CIR</td>
<td>Carrier Interference Ratio</td>
</tr>
<tr>
<td>COA</td>
<td>Care Of Address</td>
</tr>
<tr>
<td>CSMA</td>
<td>Carrier Sense Multiple Access</td>
</tr>
<tr>
<td>CA</td>
<td>Collision Avoidance</td>
</tr>
<tr>
<td>DUN</td>
<td>Dial Up Networking</td>
</tr>
<tr>
<td>DAC</td>
<td>Device Access Code</td>
</tr>
<tr>
<td>FIFO</td>
<td>First In First Out</td>
</tr>
<tr>
<td>FA</td>
<td>Foreign Agent</td>
</tr>
<tr>
<td>FH-SS</td>
<td>Frequency Hopped Spread Spectrum</td>
</tr>
<tr>
<td>GFSK</td>
<td>Gaussian Frequency Shift Keying</td>
</tr>
<tr>
<td>GGSN</td>
<td>gateway GPRS support node</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Service</td>
</tr>
<tr>
<td>HCI</td>
<td>Host Controller Interface</td>
</tr>
<tr>
<td>HA</td>
<td>Home Agent</td>
</tr>
<tr>
<td>HLR</td>
<td>Home Location Register</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>LMP</td>
<td>Link Manager Protocol</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>L2CAP</td>
<td>Logical Link Control and Adaptation Protocol</td>
</tr>
<tr>
<td>LSC</td>
<td>Layer of Software Control</td>
</tr>
<tr>
<td>MANET</td>
<td>Mobile Ad-Hoc Network</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium Access Control</td>
</tr>
<tr>
<td>MH</td>
<td>Mobile Host</td>
</tr>
<tr>
<td>MI</td>
<td>Master Connected to Internet</td>
</tr>
<tr>
<td>ML</td>
<td>Master Device List</td>
</tr>
<tr>
<td>PAN</td>
<td>Personal Area Network</td>
</tr>
<tr>
<td>PCU</td>
<td>Packet Control Unit</td>
</tr>
<tr>
<td>QOS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RCBTEE</td>
<td>A Remotely Controlled Bluetooth Enabled Environemnt</td>
</tr>
<tr>
<td>SDP</td>
<td>Service Discovery Protocol</td>
</tr>
<tr>
<td>SCO</td>
<td>Synchronous Connection Oriented</td>
</tr>
<tr>
<td>SGSN</td>
<td>serving GPRS support node</td>
</tr>
<tr>
<td>SL</td>
<td>Slave Device List</td>
</tr>
<tr>
<td>TTL</td>
<td>Time to Live</td>
</tr>
<tr>
<td>VLR</td>
<td>Visitor Location Register</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
</tbody>
</table>