NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Electrical & Electronic Engineering

BLUETOOTH EFFECTS AND RADIATION

Graduation Project
EE – 400

Student : Ömer Anıl TOK (20020431)

Supervisor : Jamal Abu HASNA

Nicosia - 2006

CONTENTS

ACKNOWLEDGEMENT

I am very thankful to my supervisor Dr. Jamal Abu HASNA who was generous with his help at every stage in preparation of this project.

Special thanks to Near East University education staff, especially to Electrical and Electronic Engineering staff for concern of me.

Also my heartily gratitude goes to my friends Gülsima TURHAN, Ertuğ GÜNAY for their help and suggestion during preparation of my project.

Finally, I want to thank to my family for their sacrificed, supported and encouraged me during my education. Without their endless support and love for me, I would have never achieved my current position.

INTRODUCTION

In this paper, important characteristics and applications of Bluetooth wireless communications and effects and radiations of bluetooth devices are examined.Also effects of bluetooth devices in our life is studied.

The document begins with an technology overwiev part where the Wireless communication history, marketing aspects and technology basics are described. In chapter 2 also includes the story of how this technology came to be named Bluetooth. In the technology basics subtopic, the basic of wireless communications and some necessary a priori knowledge about Bluetooth as master an slave roles, communication topologies, are explained.In chapter 3 bluetooth software and hardware development is explained. Also in this chapter bluetooth technology Fundamentals and Radio characteristics of 802.11 and Bluetooth is written. In chapter 4 application stack and design selection is introduced on light of the some examples.In chapter 5 you can see an interference about a software study.In this chapter you can see lots of experimental works.In chapter 6 bluetooth usage models and products are explained.In chapter 7 advantages of bluetooth is written. In this chaper you will learn lots of advantages of bluetooth devices in our real life.

Finally, The effects of Bluetooth systems on human health is presented in chapter 8.

ABSTRACT

Bluetooth is an enabling technology. As such, it is created to change our world in ways we can not imagine. New usage patterns will emerge as a result of this new technology.

In this project, , I provide you with a look at the Bluetooth technology, its basic design, structure, and applications, as well as the processes involved to develop and launch new products. I keep the very technical jargon to a minimum and give you a detailed, thorough, yet understandable look at bluetooth and its world. In this project, I Show you the following objectives, to help you get started on the road to implementing Bluetooth technology

This project also presents an evaluation of Bluetooth positioning in a general positioning platform. Proceeding the evaluation a Bluetooth based positioning system was implemented in order to complement the theoretical evaluation with empirical tests. Three different ways of positioning with Bluetooth have been developed. With a registered positioning service a Bluetooth device has an active role in the positioning task as it sends a position on request. A Bluetooth device can also take a more passive role in a positioning task, where the unique address of the device is used by a connected device to look up respective position in a database. It is also possible to forward a position gained from the positioning platform using the peer to peer characteristics in Bluetooth. This project does also contain a discussion on the theoretical time requirements for a positioning system based on Bluetooth. Empirical tests show that these requirements hold.

1. WIRELESS TECHNOLOGIES

1.1 Overview of Wireless Technology

Wireless technologies, in the simplest sense, enable one or more devices to communicate without physical connections without requiring network or peripheral cabling. Wireless technologies use radio frequency transmissions as the means for transmitting data, whereas wired technologies use cables. Wireless technologies range from complex systems, such as Wireless Local Area Networks (WLAN) and cell phones to simple devices such as wireless headphones, microphones, and other devices that do not process or store information. They also include infrared (IR) devices such as remote controls, some cordless computer keyboards and mice, and wireless hi-fi stereo headsets, all of which require a direct line of sight between the transmitter and the receiver to close the link. A brief overview of wireless networks, devices, standards are presented in this section.

1.2 Wireless Networks

Wireless networks serve as the transport mechanism between devices and among devices and the traditional wired networks (enterprise networks and the Internet). Wireless networks are many and diverse but are frequently categorized into three groups based on their coverage range: Wireless Wide Area Networks (WWAN), WLANs, and Wireless Personal Area Networks (WPAN). WWAN includes wide coverage area technologies such as 2G cellular, Cellular Digital Packet Data (CDPD), Global System for Mobile Communications (GSM), and Mobitex. WLAN, representing wireless local area networks, includes 802.11, HiperLAN, and several others. WPAN, represents wireless personal area network technologies such as Bluetooth and IR. All of these technologies are "tether less"—they receive and transmit information using electromagnetic (EM) waves. Wireless technologies use wavelengths ranging from the radio frequency (RF) band up to and above the IR band. The frequencies in the RF band cover a significant portion of the EM radiation spectrum, extending from 9 kilohertz (kHz), the lowest allocated wireless communications frequency, to thousands of gigahertz (GHz). As the frequency is increased beyond the RF spectrum, EM energy moves into the IR and then the visible spectrum. This document focuses on WLAN and WPAN technologies.

1.3 History of WLAN

Motorola developed one of the first commercial WLAN systems with its Altair product. However, early WLAN technologies had several problems that prohibited its pervasive use. These LANs were expensive, provided low data rates, were prone to radio interference, and were designed mostly to proprietary RF technologies. The IEEE initiated the 802.11 projects in 1990 with a scope "to develop a Medium Access Control (MAC) and Physical Layer (PHY) specification for wireless connectivity for fixed, portable, and moving stations within an area." In 1997, IEEE first approved the 802.11 international interoperability standard. Then, in 1999, the IEEE ratified the 802.11a and the 802.11b wireless networking communication standards. The goal was to create a standards-based technology that could span multiple physical encoding types, frequencies, and applications. The 802.11a standard uses orthogonal frequency division multiplexing (OFDM) to reduce interference. This technology uses the 5 GHz frequency spectrums and can process data at up to 54 Mbps.

1.3.1 Frequency and Data Rates

Ethernet that has been available for many years. The IEEE 802.11a standard is the most widely adopted IEEE developed the 802.11 standards to provide wireless networking technology like the wired member of the 802.11 WLAN families. It operates in the licensed 5 GHz band using OFDM technology. The popular 802.11b standard operates in the unlicensed 2.4 GHz–2.5 GHz Industrial, Scientific, and Medical (ISM) frequency band using a direct sequence spread-spectrum technology. The ISM band has become popular for wireless communications because it is available worldwide. The 802.11b WLAN technology permits transmission speeds of up to 11 Mbits per second. This makes it considerably faster than the original IEEE 802.11 standard (that sends data at up to 2 Mbps) and slightly faster than standard Ethernet.

Table 1.1 Key Characteristics of 802.11 Wireless LANs

Characteristic	Description
Physical Layer	Direct Sequence Spread Spectrum (DSSS), Frequency Hopping Spread Spectrum (FHSS), Orthogonal Frequency Division Multiplexing (OFDM), infrared (IR).
Frequency Band	2.4 GHz (ISM band) and 5 GHz.
Data and Network Security	RC4-based stream encryption algorithm for confidentiality, authentication, and integrity. Limited key management. (AES is being considered for 802.11i.)
Operating Range	Up to 150 feet indoor and 1500 feet outdoors.
Positive Aspects	Ethernet speeds without wires; many different products from many different companies. Wireless client cards and access point costs are decreasing.
Negative Aspects	Poor security in native mode; throughput decrease with distance and load.

1.3.2 Wireless LAN Components

A WLAN comprises two types of equipment: a wireless station and an access point. A station, or client, is typically a laptop or notebook personal computer (PC) with a wireless NIC. A WLAN client may also be a desktop or handheld device (e.g., PDA, or custom device such as a barcode scanner) or equipment within a kiosk on a manufacturing floor or other publicly accessed area. Wireless laptops and notebooks "wireless enabled" are identical to laptops and notebooks except that they use wireless NICs to connect to access points in the network. The wireless NIC is commonly inserted in the client's Personal Computer Memory Card International Association (PCMCIA) slot or Universal Serial Bus (USB) port. The NICs use radio signals to establish connections to the WLAN. The AP, which acts as a bridge between the wireless and wired networks, typically comprises a radio, a wired network interface such as 802.3, and bridging software. The AP functions as a base station for the wireless network, aggregating multiple wireless stations onto the wired network.

1.3.3 Range

The reliable coverage range for 802.11 WLAN's depends on several factors, including data rate required and capacity, sources of RF interference, physical area and characteristics, power, connectivity, and antenna usage. Theoretical ranges are from 29 meters (for 11 Mbps) in a closed office area to 485 meters (for 1 Mbps) in an open area. However, through empirical analysis, the typical range for connectivity of 802.11 equipment is approximately 50 meters (about 163 ft.) indoors. A range of 400 meters, nearly ¼ mile, makes WLAN the ideal technology for many campus applications. It is important to recognize that special high-gain antennas can increase the range to several miles.

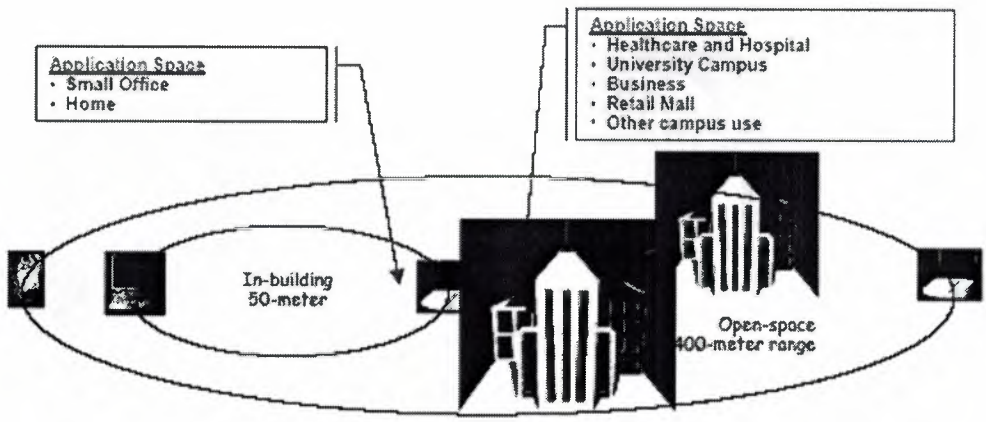

Figure 1.1 Typical Range of 802.11 WLAN

AP's may also provide a "bridging" function. Bridging connects two or more networks together and allows them to communicate to exchange network traffic. Bridging involves either a point-to-point or a multipoint configuration. In a point-to-point architecture, two LANs are connected to each other via the LAN's respective AP's. In multipoint bridging, one subnet on a LAN is connected to several other subnets on another LAN via each subnet AP. For example, if a computer on Subnet A needed to connect to computers on Subnets B, C, and D, Subnet A's AP would connect to B's, C's, and D's respective AP's. Enterprises

may use bridging to connect LANs between different buildings on corporate campuses. Bridging AP devices are typically placed on top of buildings to achieve greater antenna reception. The typical distance over which one AP can be connected wirelessly to another by means of bridging is approximately 2 miles. This distance may vary depending on several factors including the specific receiver or transceiver being used. Figure 1.2 illustrates point-to-point bridging between two LANs. In the example, wireless data is being transmitted from Laptop A to Laptop B, from one building to the next, using each building's appropriately positioned AP. Laptop A connects to the closest AP within the building A. The receiving AP in building A then transmits the data (over the wired LAN) to the AP bridge located on the building's roof. That AP bridge then transmits the data to the bridge on nearby building B. The building's AP bridge then sends the data over its wired LAN to Laptop B.

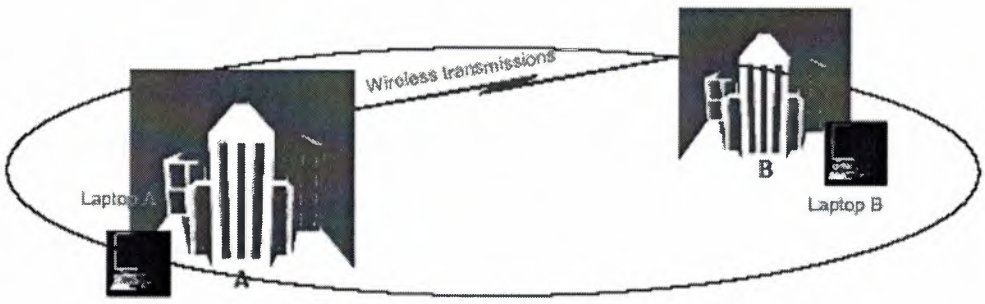

Figure 1.2 Access Point Bridging

1.3.4 Benefits

WLANs offer four primary benefits:

- **User Mobility**—Users can access files, network resources, and the Internet without having to physically connect to the network with wires. Users can be mobile yet retain high-speed, real-time access to the enterprise LAN.
- **Rapid Installation**—The time required for installation is reduced because network connections can be made without moving or adding wires, or pulling them through walls or ceilings, or making modifications to the infrastructure cable plant. For

example, WLANs are often cited as making LAN installations possible in buildings that are subject to historic preservation rules.

- **Flexibility**—Enterprises can also enjoy the flexibility of installing and taking down WLANs in locations as necessary. Users can quickly install a small WLAN for temporary needs such as a conference, trade show, or standards meeting.
- **Scalability**—WLAN network topologies can easily be configured to meet specific application and installation needs and to scale from small peer-to-peer networks to very large enterprise networks that enable roaming over a broad area.

Because of these fundamental benefits, the WLAN market has been increasing steadily over the past several years, and WLANs are still gaining in popularity. WLANs are now becoming a viable alternative to traditional wired solutions. For example, hospitals, universities, airports, hotels, and retail shops are already using wireless technologies to conduct their daily business operations.

1.4 Ad Hoc Networks

Ad hoc networks such as Bluetooth are networks designed to dynamically connect remote devices such as cell phones, laptops, and PDAs. These networks are termed "ad hoc" because of their shifting network topologies. Whereas WLANs use a fixed network infrastructure, ad hoc networks maintain random network configurations, relying on a master-slave system connected by wireless links to enable devices to communicate. In a Bluetooth network, the master of the piconet controls the changing network topologies of these networks. It also controls the flow of data between devices that are capable of supporting direct links to each other. As devices move about in an unpredictable fashion, these networks must be reconfigured on the fly to handle the dynamic topology. The routing that protocol Bluetooth employs allows the master to establish and maintain these shifting networks.

Figure 1.5 illustrates an example of a Bluetooth-enabled mobile phone connecting to a mobile phone network, synchronizing with a PDA address book, and downloading e-mail on an IEEE 802.11 WLAN.

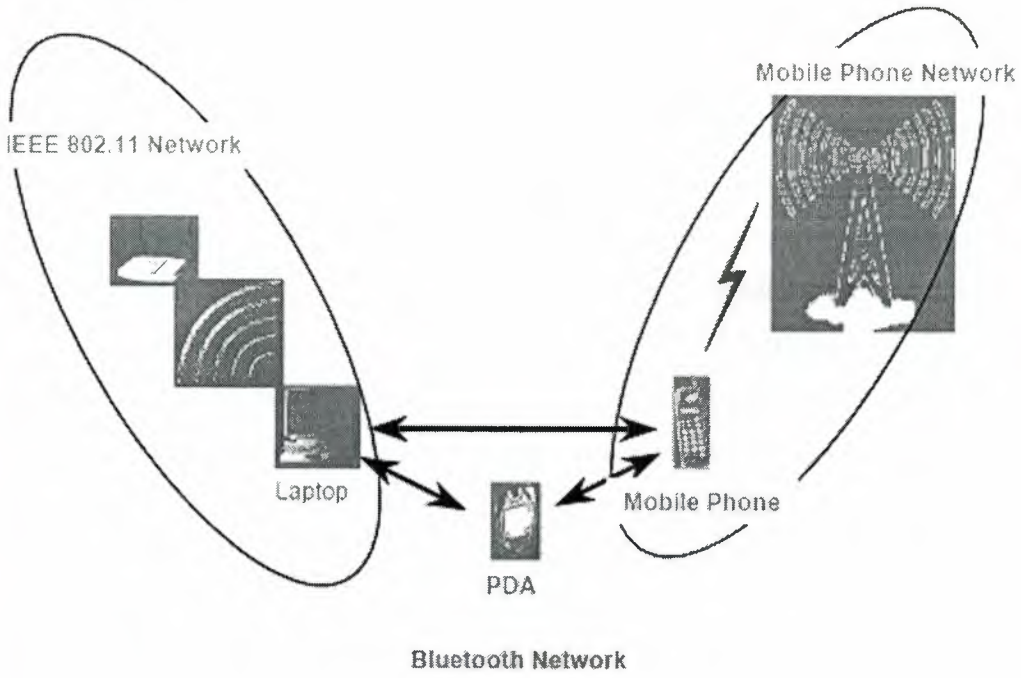

Figure 1.5 Notional Ad Hoc Network

1.5 Wireless Devices

A wide range of devices use wireless technologies, with handheld devices being the most prevalent form today. This document discusses the most commonly used wireless handheld devices such as text messaging devices, PDAs, and smart phones.

1.5.1 Personal Digital Assistants

PDAs are data organizers that are small enough to fit into a shirt pocket or a purse. PDAs offer applications such as office productivity, database applications, address books, schedulers, and to-do lists, and they allow users to synchronize data between two PDAs and between a PDA and a personal computer. Newer versions allow users to download their e-mail and to connect to the Internet. Security administrators may also encounter one-way and two-way text-messaging devices. These devices operate on a proprietary networking standard that disseminates e-mail to remote devices by accessing the corporate network.

Text-messaging technology is designed to monitor a user's inbox for new e-mail and relay the mail to the user's wireless handheld device via the Internet and wireless network.

1.5.2 Smart Phones

Mobile wireless telephones, or cell phones, are telephones that have short wave analog or digital transmission capabilities that allow users to establish wireless connections to nearby transmitters. As with WLANs, the transmitter's span of coverage is called a "cell." As the cell phone user moves from one cell to the next, the telephone connection is effectively passed from one local cell transmitter to the next. Today's cell phone is rapidly evolving to integration with PDAs, thus providing users with increased wireless e-mail and Internet access. Mobile phones with information processing and data networking capabilities are called "smart phones." This document addresses the risks introduced by the information processing and networking capabilities of smart phones.

1.6 Wireless Standards

Wireless technologies conform to a variety of standards and offer varying levels of security features. The principal advantages of standards are to encourage mass production and to allow products from multiple vendors to interoperate. For this document, the discussion of wireless standards is limited to the IEEE 802.11 and the Bluetooth standard. WLANs follow the IEEE 802.11 standards. Ad hoc networks follow proprietary techniques or are based on the Bluetooth standard, which was developed by a consortium of commercial companies making up the Bluetooth Special Interest Group (SIG). These standards are described below.

1.6.1 IEEE 802.11

WLANs are based on the IEEE 802.11 standard, which the IEEE first developed in 1997. The IEEE designed 802.11 to support medium-range, higher data rate applications, such as Ethernet networks, and to address mobile and portable stations. 802.11 are the original WLAN standard, designed for 1 Mbps to 2 Mbps wireless transmissions. It was followed in 1999 by 802.11a, which established a high-speed WLAN standard for the 5 GHz band and

supported 54 Mbps. Also completed in 1999 was the 802.11b standard, which operates in the 2.4 - 2.48 GHz band and supports 11 Mbps. The 802.11b standard is currently the dominant standard for WLANs, providing sufficient speeds for most of today's applications. Because the 802.11b standard has been so widely adopted, the security weaknesses in the standard have been exposed. Another standard, 802.11g, still in draft, operates in the 2.4 GHz waveband, where current WLAN products based on the 802.11b standard operate. Two other important and related standards for WLANs are 802.1X and 802.11i. The 802.1X, a port-level access control protocol, provides a security framework for IEEE networks, including Ethernet and wireless networks. The 802.11i standard, also still in draft, was created for wireless-specific security functions that operate with IEEE 802.1X.

1.6.2 IEEE 802.11 Architecture

The IEEE 802.11 standard permits devices to establish either peer-to-peer (P2P) networks or networks based on fixed access points (AP) with which mobile nodes can communicate. Hence, the standard defines two basic network topologies: the infrastructure network and the ad hoc network. The infrastructure network is meant to extend the range of the wired LAN to wireless cells. A laptop or other mobile device may move from cell to cell (from AP to AP) while maintaining access to the resources of the LAN. A cell is the area covered by an AP and is called a "basic service set" (BSS). The collection of all cells of an infrastructure network is called an extended service set (ESS). This first topology is useful for providing wireless coverage of building or campus areas. By deploying multiple APs with overlapping coverage areas, organizations can achieve broad network coverage. WLAN technology can be used to replace wired LANs totally and to extend LAN infrastructure. A WLAN environment has wireless client stations that use radio modems to communicate to an AP. The client stations are generally equipped with a wireless network interface card (NIC) that consists of the radio transceiver and the logic to interact with the client machine and software. An AP comprises essentially a radio transceiver on one side and a bridge to the wired backbone on the other. The AP, a stationary device that is part of the wired infrastructure, is analogous to a cell-site (base station) in cellular communications. All communications between the client stations and between clients and

the wired network go through the AP. The basic topology of a WLAN is depicted in Figure 1.3

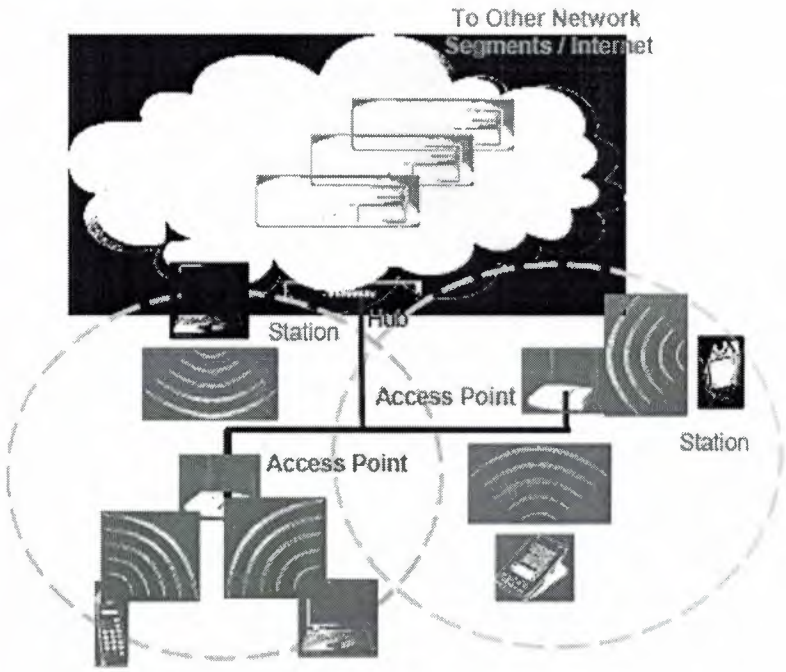

Figure 1.3 Fundamental 802.11 Wireless LAN Topology

Although most WLANs operate in the "infrastructure" mode and architecture described above, another topology is also possible. This second topology, the ad hoc network, is meant to easily interconnect mobile devices that are in the same area (e.g., in the same room). In this architecture, client stations are grouped into a single geographic area and can be Internet-worked without access to the wired LAN (infrastructure network). The interconnected devices in the ad hoc mode are referred to as an independent basic service set (IBSS). The ad hoc topology is depicted in Figure 1.4 below.

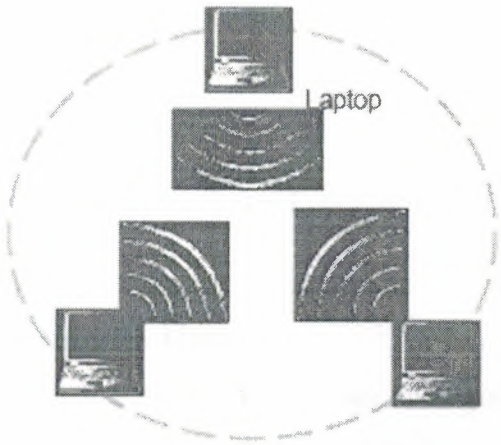

Figure 1.4 802.11 Wireless LAN Ad Hoc Topology

The ad hoc configuration is similar to a peer-to-peer office network in which no node is required to function as a server. As an ad hoc WLAN, laptops, desktops and other 802.11 devices can share files without the use of an AP.

2. INTRODUCTION TO BLUETOOTH

2.1 Bluetooth History

Bluetooth is a notable technology among the other high technologies in several respects, but its name garners much attention. Most new industry enterprises are known by a name that describes their associated technology or its applications and often they quickly become known by an acronym describing the full name. So why the name of the technology is "Bluetooth"? And why an acronym has not been considered for Bluetooth? The answer lies in the heritage of the original inventors. There are numerous histories and accounts of the Bluetooth namesake and how that name came to be chosen. Harald Bluetooth was a Viking and King of Denmark between 940 and 981. In fact, his name was Harald Blåtand, but by the time "Blåtand" became "Bluetooth" and it has probably taken from two Old Danish words, 'ble' (blue) meaning dark skinned and 'tan' meaning great man. One of King Harald's skills was getting people to talk to each other, and during his rule Denmark and Norway were Christianized and united. Today Bluetooth wireless technology enables devices to talk to each other, but this time by means of a low-cost short-range radio link. In the Danish town of Jelling, Harald Bluetooth raised an enormous rune stone that still stands in its original position. It has the following runic inscription, adorned with an image of Christ: King Harald raised this monument to the memory of Gorm his father and They're his mother, that Harald which won all Denmark and Norway and made the Danes Christian. Originally, the stone was painted. In September 1999, a new stone was raised outside of Ericsson Mobile Communications in Lund, this time to the memory of Harald Bluetooth. In 1998, IBM, Intel, Nokia, and Toshiba formed the Bluetooth SIG, which serves as the governing body of the specification. The SIG began as a means to monitor the development of the radio technology and the creation of a global and open standard. Today more than 2,000 organizations are part of the Bluetooth SIG, comprising leaders in the telecommunications and computing industries that are driving development and promotion

of Bluetooth technology. Bluetooth was originally designed primarily as a cable replacement protocol for wireless communications. However, SIG members plan to develop a broad range of Bluetooth-enabled consumer devices to enhance wireless connectivity. Among the array of devices that are anticipated are cellular phones, PDAs, notebook computers, modems, cordless phones, pagers, laptop computers, cameras, PC cards, fax machines, and printers. Bluetooth is now standardized within the IEEE 802.15 Personal Area Network (PAN) Working Group that formed in early 1999. The Bluetooth SIG Web site provides numerous links to other Web sites with additional information. The IEEE Web site provides updates on the IEEE 802.15 Working Group. This is the Working Group that develops Personal Area Networking consensus standards for short distance wireless networks, or WPANs.

2.2 Bluetooth Overview

Ad hoc networks today are based primarily on Bluetooth technology. Bluetooth is an open standard for short-range digital radio. It is touted as a low-cost, low power, and low profile technology that provides a mechanism for creating small wireless networks on an ad hoc basis. Bluetooth is considered a wireless PAN technology that offers fast and reliable transmission for both voice and data. Untethered Bluetooth devices will eliminate the need for cables and provide a bridge to existing networks. Bluetooth can be used to connect almost any device to any other device. An example is the connection between a PDA and a mobile phone. The goal of Bluetooth is to connect disparate devices (PDAs, cell phones, printers, faxes, etc.) together wirelessly in a small environment such as an office or home. According to the leading proponents of the technology, Bluetooth is a standard that will ultimately:

- Eliminate wires and cables between both stationary and mobile devices
- Facilitate both data and voice communications
- Offer the possibility of ad hoc networks and deliver synchronicity between personal devices.

Bluetooth is designed to operate in the unlicensed ISM (industrial, scientific, medical applications) band that is available in most parts of the world, with variation in some

locations. The characteristics of Bluetooth are summarized in Table 2.1 Bluetooth-enabled devices will automatically locate each other, but making connections with other devices and forming networks requires user action. As with all ad hoc networks, Bluetooth network topologies are established on a temporary and random basis. A distinguishing feature of Bluetooth networks is the master-slave relationship maintained between the network devices. Up to eight Bluetooth devices may be networked together in a master-slave relationship, called a "piconet." In a piconet, one device is designated as the master of the network with up to seven slaves connected directly to that network. The master device controls and sets up the network (including defining the network's hopping scheme). Devices in a Bluetooth piconet operate on the same channel and follow the same frequency hopping sequence. Although only one device may perform as the master for each network, a slave in one network can act as the master for other networks, thus creating a chain of networks. This series of piconets, often referred to as scatter-nets, allows several devices to be Internet worked over an extended distance. This relationship also allows for a dynamic topology that may change during any given session: as a device moves toward or away from the master device in the network, the topology and therefore the relationships of the devices in the immediate network change.

Table 2.1 Key Characteristics of Bluetooth Technology

Characteristic	Description
Physical Layer	Frequency Hopping Spread Spectrum (FHSS).
Frequency Band	2.4 – 2.4835 GHz (ISM band).
Hop Frequency	1,600 hops/sec.
Data Rate	1 Mbps (raw). Higher bit rates are anticipated.

Data and Network Security	Three modes of security (none, link-level, and service level), two levels of device trust, and three levels of service security. Stream encryption for confidentiality, challenge-response for authentication. PIN-derived keys and limited management.
Operating Range	About 10 meters (30 feet); can be extended to 100 meters.
Throughput	Up to approximately 720 kbps.
Positive Aspects	No wires and cables for many interfaces. Ability to penetrate walls and other obstacles. Costs are decreasing with a $5 cost projected. Low power and minimal hardware.
Negative Aspects	Possibility for interference with other ISM band technologies. Relatively low data rates. Signals leak outside desired boundaries.

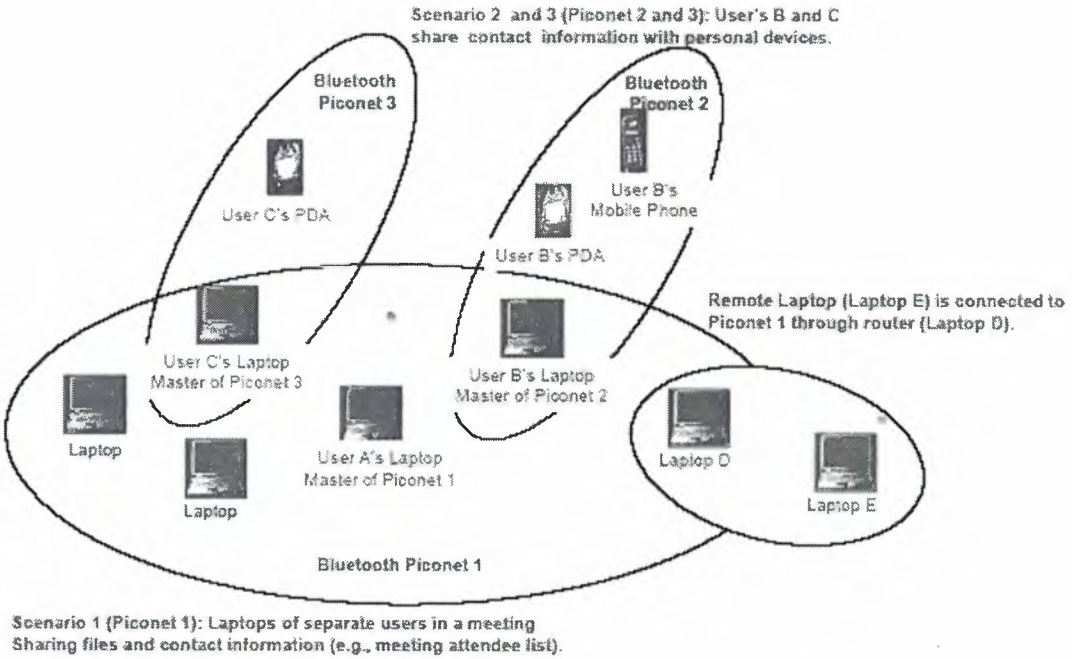

Figure 2.2 Typical Bluetooth Network – A

control the flow of data between devices that are capable of supporting a direct link to each other. As devices move about in a random fashion, these networks must be reconfigured on the fly to handle the dynamic topology. The routing protocols it employs allow Bluetooth to establish and maintain these shifting networks. Bluetooth transceivers operate in the 2.4 GHz, ISM band, which is similar to the band WLAN devices and other IEEE 802.11 compliant devices occupy. Bluetooth transceivers, which use Gaussian Frequency Shift Keying (GFSK) modulation, employ a frequency hopping (FH) spread spectrum system with a hopping pattern of 1,600 times per second over 79 frequencies in a quasi-random fashion. The theoretical maximum bandwidth of a Bluetooth network is 1 Mbps. However, in reality the networks cannot support such data rates because of communication overhead. The second generation of Bluetooth technology is expected to provide a maximum bandwidth of 2 Mbps. Bluetooth networks can support either one asynchronous data channel with up to three simultaneous synchronous speech channels or one channel that transfers asynchronous data and synchronous speech simultaneously. Bluetooth uses a combination of packet-switching technology and circuit-switching technology. The advantage of using packet switching in Bluetooth is that it allows devices to route multiple packets of information by the same data path. Since this method does not consume all the resources on a data path, it becomes easier for remote devices to maintain data flow throughout a scatter-net.

2.2.1 Frequency and Data Rates

The designers of Bluetooth like those of the 802.11 WLAN standard designed Bluetooth to operate in theunlicensed 2.4 GHz–2.4835 GHz ISM frequency band. Because numerous other technologies also operate in this band, Bluetooth uses a frequency-hopping spread-spectrum (FHSS) technology to solve interference problems. The FHSS scheme uses 79 different radio channels by changing frequency about 1,600 times per second. One channel is used in 625 microseconds followed by a hop in a pseudo-random order to another channel for another 625-microsecond transmission; this process is repeated continuously. As stated previously, the ISM band has become popular for wireless communications because it is available worldwide and does not require a license. In the ISM band, Bluetooth technology permits transmission speeds of up to 1 Mbps and achieves a

throughput of approximately 720 kbps. Although the data rates are low compared to those of 802.11 wireless LANs, it is still three to eight times the average speed of parallel and serial ports, respectively. This rate is adequately fast for many of the applications for which Bluetooth was conceived. Moreover, it is anticipated that even faster data rates will be available in the future.

2.2.2 Frequency Hopping Spread Spectrum

In the RF communications, spread spectrum refers to dividing the available spectrum based upon frequency, time, a coding scheme or some other method. Messages to be sent are then divided into various packets that are transmitted across the divided spectrum (or frequency hopping). The method is employed with Bluetooth wireless communication, divides the spectrum into different frequencies, or channels. A single message packet is transmitted on a selected channel, then the radio selects a new channel (this process is called hopping to a new frequency) to transmit the next packet, and the process repeats, by that means spreading the message across the available frequency spectrum. Obviously the receiver of the message must know the hopping pattern to tune to the correct channels successfully to receive each packet and assemble the complete message. This process is called *frequency hopping spread spectrum*, or FHSS. The devices that communicate with each other must transmit and receive on the same frequency at the same time. The frequency-selection module (FSM) contains the procedure for selecting the next frequency to be used under various operating conditions.

2.2.3 Bluetooth Architecture and Components

As with the IEEE 802.11 standard, Bluetooth permits devices to establish either P2P networks or networks based on fixed access points with which mobile nodes can communicate. In this document, however, we only discuss the ad hoc network topology. This topology is meant to easily interconnect mobile devices that are in the same area (e.g., in the same room). In this architecture, client stations are grouped into a single geographic area and can be inter-networked without access to the wired LAN (infrastructure network). The basic Bluetooth topology is depicted in Figure 2.2. As shown in this piconet, one of the devices would be a master, and the other two devices would be slaves.

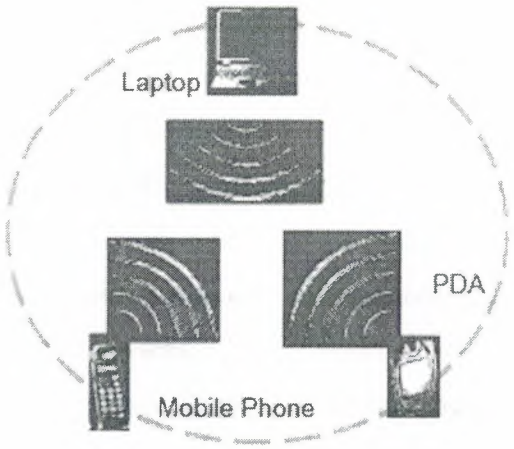

Figure 2.3 Bluetooth Ad Hoc Topology

Unlike a WLAN that comprises both a wireless station and an access point, with Bluetooth, there are only wireless stations or clients. A Bluetooth client is simply a device with a Bluetooth radio and Bluetooth software module incorporating the Bluetooth protocol stack and interfaces.

2.3 Benefits

Bluetooth offers five primary benefits to users. This ad hoc method of untethered communication makes Bluetooth very attractive today and can result in increased efficiency and reduced costs. The efficiencies and cost savings are attractive for the home user and the enterprise business user.

Benefits of Bluetooth include:

- **Cable replacement:** Bluetooth technology replaces cables for a variety of interconnections. These include those of peripheral devices (i.e., mouse and keyboard computer connections), USB at 12 Mbps (USB 1.1) up to 480 Mbps (USB 2.0); printers and modems, usually at 4 Mbps; and wireless headsets and microphones that interface with PCs or mobile phones.
- **Ease of file sharing:** Bluetooth enables file sharing between Bluetooth-enabled devices. For example, participants of a meeting with Bluetooth-compatible laptops

can share files with each other. In another example, a Bluetooth-compatible mobile phone acts as a wireless modem for laptops. Using Bluetooth, the laptop interfaces with the cell phone, which in turn connects to a network, thus giving the laptop a full range of networking capabilities without the need of an electrical interface for the laptop–to–mobile phone connection.

- **Wireless synchronization:** Bluetooth provides automatic wireless synchronization with other Bluetooth-enabled devices. For example, personal information contained in address books and date books can be synchronized between PDAs, laptops, mobile phones, and other devices.
- **Automated wireless applications:** Bluetooth supports automatic wireless application functions. Unlike synchronization, which typically occurs locally, automatic wireless applications interface with the LAN and Internet. For example, an individual working offline on e-mails might be outside of their regular service area on a flight, for instance. To e-mail the files queued in the inbox of the laptop, the individual, once back in a service area (i.e., having landed), would activate a mobile phone or any other device capable of connecting to a network. The laptop would then automatically initiate a network join by using the phone as a modem and automatically send the e-mails after the individual logs on.
- **Internet connectivity:** Bluetooth is supported by a variety of devices and applications. Some of these devices include mobile phones, PDAs, laptops, desktops, and fixed telephones. Internet connectivity is possible when these devices and technologies join together to use each other's capabilities. For example, a laptop, using a Bluetooth connection, can request a mobile phone to establish a dial-up connection; the laptop can then access the Internet through that connection. Bluetooth is expected to be built into office appliances (e.g., PCs, faxes, printers, and laptops), communication appliances (e.g., cell phones, handsets, pagers, and headsets), and home appliances (e.g., DVD players, cameras, refrigerators, and microwave ovens). Applications for Bluetooth also include vending machines, banking, and other electronic payment systems; wireless office and conference rooms; smart homes; and in-vehicle communications and parking.

2.4 Bluetooth and Security

Bluetooth has been alternately touted as a taste of things to come and the answer to all our wireless connectivity prayers. It promises everything from the ability to program our microwaves from work, to pushing ads from pop machines to your pocket device. The Gartner Group seems to agree that it will catch on in a big way it predicts a market of $700 million for Bluetooth chips by 2006. Put simply, Bluetooth is a wireless standard that facilitates communications between devices. A Bluetooth capable device sends out a signal in a 30-foot radius, allowing any Bluetooth enabled device to speak to another. Therein lie the biggest advantages and people's worst fears of Bluetooth. The Gartner Group predicts that by 2004, 70 percent of new cell phones and 40 percent of personal digital assistants (PDA) will use some sort of wireless technology to communicate with other devices, and a great deal of that technology will include Bluetooth. Millions of other devices will be shipped with Bluetooth capability as well, including computers, stereos, even refrigerators. In short, Bluetooth will be everywhere. But Bluetooth's promise of seamless, pervasive wireless connectivity begs an important question is it secure? Researchers from Lucent technologies recently discovered security holes in the Bluetooth specification, making this question even more important and pressing. If Microsoft's own servers can be hacked, why not your Bluetooth capable laptop or the Bluetooth equipped security system in your home? Because Bluetooth will be so widespread, security will be of paramount importance. IrDA, a wireless data transfer method based on infrared signals, provided a measure of security by requiring a line of sight to devices. Bluetooth provides no such requirement. It is not hard to envision a scenario where a shadowy figure could sit on the other side of a wall from an executive's Bluetooth-enabled PC and hack his way into it via the wireless connection, mining whatever data he can from the information stored on the PC, or even the network the computer is connected to. Even more frightening to many people, but a much less likely scenario, is that someone could sit in a coffee shop and search for Bluetooth devices within range, pulling personal information, even credit card numbers, off the devices. Weaknesses in the encryption scheme could allow a hacker to listen and determine the authentication/pairing key thus be able decipher even encrypted data being sent between authenticated bluetooth devices. Another possible issue is a type of "denial of service attack" that drains batteries by forcing constant intensive utilization of a device's processor.

The Bluetooth specification provides little to no protection against that sort of attack. These scenarios are highly unlikely, but plausible without serious attention given to the security of Bluetooth. The Bluetooth protocol already has several security measures built in at the hardware level, but they are only truly effective if device manufacturers work to understand and take advantage of them. Security issues associated with scatternets within Bluetooth are still being ironed out as well. Bluetooth security starts at the hardware level. The Bluetooth chips themselves have built-in security considerations. The Bluetooth hardware specifications include encryption, random number generation, encryption key management, authentication (unidirectional and bi-directional), and authorization. These are based on a secret link key that is shared by a pair of devices. The key is generated by a technique referred to as "pairing/bonding". Authentication is the process of verifying 'who' is at the other end of the link. It is performed for devices and is not done on a per user or service level. Authorization is the process of deciding if device X is allowed to have access to service Y. This is where the concept of "trusted" comes in. Authorization always includes authentication. Bluetooth allows selective security in that it allows device X access to service Y and not service Z, while allowing device M to have unrestricted access to all services (once paired) and provide no access to device N. In addition, the frequency-hopping and power-adaptation features within Bluetooth set a limited range on the signal, making the system difficult to eavesdrop on. However, these measures only go so far. Bluetooth currently provides adequate security for smaller applications, but for larger ad-hoc applications there still are quite a few unanswered questions because the Bluetooth Special Interest Group (SIG) initially left many aspects of Bluetooth security implementation specific. Device manufacturers have to take the next step and add their own security measures to make Bluetooth truly secure, especially given the recently discovered security holes. The first issue is for manufacturers to simply take advantage of the built-in security features Bluetooth offers. It all starts wi th the link, where Bluetooth devices initially establish communications with one another. Other built-in security features of Bluetooth also play heavily into creating a secure networking environment. F requency hopping, where the device r apidly cycles through preset frequencies o n the Bluetooth wavelength, occurs at 1600 hops per second. This may seem like a minor feature, but it makes it much more difficult to intercept Bluetooth signals. Without having a device in

sync with the frequency hop, bits of data can be intercepted, but the full stream cannot. Adaptive power capabilities make it difficult to eavesdrop on Bluetooth transmissions. Bluetooth devices have variable ranges, potentially reaching 30 feet away. However, that sort of range isn't necessary in devices like PDAs and cell phones. The hardware allows device developers, and even consumers if the developers code the necessary interfaces and options in, to modify the coverage area to reduce the chances of someone hacking their way in to a Bluetooth-enabled device from 30 feet away. Should someone manage to intercept a data stream, the Bluetoo th specification includes hardware e ncryption, which makes it difficult to make any sense of the data, but unfortunately far from impossible.

2.4.1 Bluetooth Security Modes

The General Access profile in the Bluetooth Profiles specification specifies three security modes within a device:

- **No security (mode 1):** A device will not initiate any security procedure.
- **Service level (mode 2):** A device does not initiate security procedures before channel establishment at the L2CAP level; security is only enforced after channel establishment. This mode allows different and flexible access policies for applications, especially running applications with different security requirements in parallel.
- **Link level (mode 3):** A device initiates security procedures before the link set-up at the LMP level is completed.

2.4.2 Security Levels

There are two levels to Bluetooth security: the device level and the service level.

2.4.2.1 Device Trust Level

At the device level, Bluetooth devices fit in one of two categories when making a link:

- **Trusted:** The device has a fixed relationship with the other device and has unrestricted access to all services on the host device. A trusted device is allowed total access to the host and provides an authenticated encrypted key to the device it is paired with upon login.

- **Untrusted:** The device has no permanent relationship and is not paired with the host device so has restricted access to services. Without the encrypted key, access to services on the host device is restricted according to whatever security protocols are in place on the device.

2.4.2.2 Security Levels of Services

At the service level, security may be again set at 3 levels:

- Services open to all devices: Neither authentication nor authorization is needed.
- Services that require authentication only: Authorization is not needed.
- Services that require authorization and authentication: Automatic access is only granted to trusted devices. Other devices need a "manual" authorization.

In addition, some services may require encryption once authorization and/or authentication are complete. Legacy applications are provided a default security level that would be used unless a different policy is defined in the security database.

2.5 Communication

A Bluetooth transceiver is a frequencyhopping spread-spectrum (FHSS) device that uses the unlicensed (worldwide) 2.4 GHz ISM (Industrial, Scientific, Medical) frequency band. In most countries, there are 79 channels available; however some countries allow the use of only 23 channels. The nominal bandwidth for each channel is 1MHz. FCC part 15.247 regulations restrict the maximum allowed peak power output to 1 watt and require that at least 75 of the 79 channels be used in a pseudorandom manner. A device cannot operate on a given channel for longer than0. 4 seconds within any 30-second period These limits (or restrictions) were put into place to minimize the amount of interference in the ISM band, which is also used by 802.11 b/g devices, HomeRF devices, portable phones and microwave ovens.

When connected to other Bluetooth devices, a Bluetooth device hops (changes frequencies) at the rate of 1600 times per second for typical use, with a residence time of 625 sec . When in inquiry or page mode, it hops at 3200 hops per second with a residence time of 312.5 sec. A Bluetooth transceiver uses all 79 channels, and hops pseudo-randomly across all channels at a rate of 1600 hops per second for standard transmissions. It has a range of approximately10 meters, although ranges up to 100 meters can be achieved with amplifiers. Because the transceiver has an extremely small footprint, it is easily embedded into physical devices, making it a truly ubiquitous radio link. The Bluetooth specification uses time division duplexing (TDD) and time division multiple access (TDMA) for device communication. A single time slot is 625 sec in length, representing the length of a single-slot packet. At the Baseband layer, a packet consists of an access code, a header, and the payload, as shown in Fig. 2.4 .The access code contains the piconet address (to filter out messages from other piconets) and is usually 72 bits in length. The header contains link control data, encoded with a forward error-correcting code (FEC) with a 1/3 rate for high reliability. Such code is a repetition code and thus every bit in the header is transmitted three times. The header is usually 18 bits in length, and includes the active member address for a currently active slave. The payload can contain from 0 to 2745 bits of data, and may be protected by a 1/3 rate FEC (simple bit repetition, for SCO packets only), a 2/3 rate FEC (which is a (15,10) shortened Hamming code capable of correcting all one-bit errors and detecting all two-bit errors), or a 3/3/ rate (no FEC). For SCO connections, packets must be exactly one time-slot in length. For ACL links, packets may be 1, 3, or 5 time slots in length.

Bluetooth uses polling-based packet transmission. All communication between devices takes place between a master and a slave, using time-division duplex (TDD), with no direct slave-toslave communication. The master will poll each active slave to determine if it has data to transmit. The slave may only transmit data when it has been polled. Also, it must send its data in the time slot immediately following the one in which it was polled.

The master transmits only in evennumbered time slots, while the slaves transmit only in odd-numbered time slots. In each time slot, a different frequency channel f is used (a hop in the hopping sequence).

2.5.1 The piconet

The Bluetooth specification defines a piconet as an ad-hoc, spontaneous clustering of Bluetooth devices. In it, one device holds the role of master, while the rest of the devices are slaves. While there is no limit to the total number of slaves in a piconet, a maximum of seven slaves can be active in a piconet at any given point in time. If there are more than seven slaves, the rest of the slaves must be "parked." The maximum number of "parked" slaves is 255 per piconet with direct addressing via a parked slave address as defined by the SIG; however, indirect addressing of parked slaves by their specific.

Bluetooth device address is also permitted, effectively allowing any number of parked slaves. To reactivate a parked slave, the master must first place a currently active slave into a parked state.

When two Bluetooth devices enter into communication range, they will attempt to communicate with each other. If no piconet is available at that time, a negotiation process will ensue. One device will become the master (usually the device which initiated the communication) and the other will become a slave. Any Bluetooth device can function within a piconet as a master, a slave or a bridge. These roles are temporary and exist only as long as the piconet itself exists. The master device selects the frequency, the frequency-hopping sequence, the timing (when the hops will actually occur) and the polling order of the slaves. The master is also responsible for instructing the slave devices to switch to different device states for periods of inactivity.

A master and slave must exchange address and clock information in order for the slave to join the master's piconet. Bluetooth devices each have a unique Global ID used to create a hopping pattern. The master radio shares its Global ID and clock offset with each slave in

its piconet, providing the offset into the hopping pattern. A slave must be able to recreate the frequency-hopping sequence of the piconet it has joined, must know which frequency to use at which time, and must synchronize itself with the master's clock. The slave device does not actually adjust its own clock. Rather it tracks the amount of clock drift between its clock and the master's, and adjusts its transmission schedule accordingly.

A Bluetooth bridge device (or gateway) interconnects two or more piconets for multi-hop communication. The bridge communicates with all the piconets connected to it by aligning itself with the clocking of each piconet when it is ready to communicate. However, it can only communicate with one piconet at a time. Because the bridge incurs additional overhead shifting from one clocking to another to communicate with each connected piconet, it has the potential to become a bottleneck.

A bridge device may be a slave in all of the piconets to which it is connected, or it may be a master in one piconet and a slave in the others. The interconnection of two or more piconets via bridge devices results in the formation of a Bluetooth scatternet. A Bluetooth device can be in one of the following states: standby, inquiry, page, connected, transmit, hold, park or sniff, as shown in Fig. 2.4

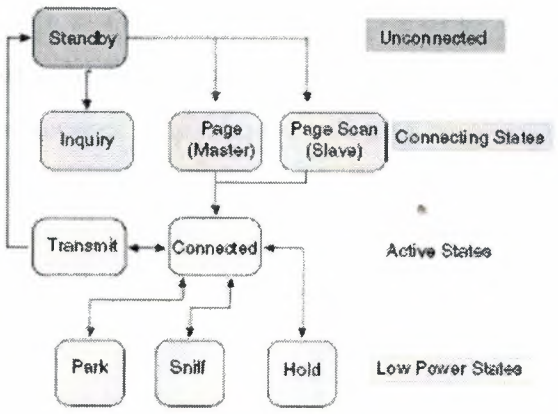

Figure 2.4 States of bluetooth device

A device is in Standby mode when it is powered on but has not yet joined a piconet. It enters the Inquiry state when it sends out requests to find other devices to which it might connect.

A master of an existing piconet may also be in a Page state, sending out messages looking for devices that it can invite to join its piconet.

When successful communication is made between the master and the new device, the new device assumes the slave role, enters the Connected state, and receives an active address. While connected, the slave can transmit data when the master polls it. During the transmission of its data, the slave is in a Transmit state. At the end of its transmission, it returns to the Connected state.

The Sniff state is a low-power consumption state in which the slave "sleeps" for a pre-determined number of time slots. It wakes up at its appointed time slot for data transmission. It then returns to the inactive state until its next designated Sniff time slot arrives. The Hold state is another low-power state in which the slave is not active for a predetermined amount of time. However, there is no data transfer within the Hold state.

When a slave device has no data to send or receive, the master may instruct the slave to enter the Park state. When it enters a Park state, the slave relinquishes its active address in the piconet. The address will then be given to another slave that the master is reactivating from Park state.

3. BLUETOOTH SOFTWARE AND HARDWARE DEVELOPMENT

3.1 Bluetooth Technology

3.1.1 Fundamentals

The following statement describes the application and target markets for Bluetooch technology:

The term Bluetooth refers to an open specification for a technology to enable short-range wireless voice and data communications anywhere in the world. The statement provides a useful starting point for understanding Bluetooth, and in particular two aspects merit further explanation. [2] Short range wireless: Bluetooth communication uses radio waves to transmit and receive data similar to other technologies such as Wireless LAN (WLAN or Wi-Fi) or HomeRF. Unlike such computer networking technologies,

Bluetooth is specifically targeted at small portable devices. The requirements for such devices necessitate low power consumption and small physical dimensions — the result is tiny 1 mW Bluetooth chips that provides a theoretical range of approximately 10 meters.

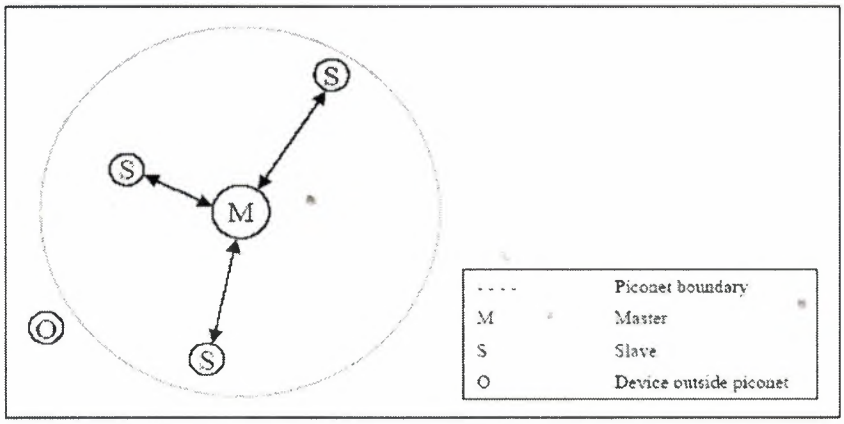

Figure 3.1: A Bluetooth piconet with 4 devices

* Anywhere in the world: Bluetooth operates in the unlicensed 2.4 GHz band. This band is a portion of the radio spectrum that is freely available for use in most countries, with the current exception of France and Spain. Using this unlicensed portion of the spectrum has the advantage of nearglobal availability. The disadvantage is that other wireless technologies also operate in the same band, which may cause interference when the different technologies coexist in time and space.

Interaction between Bluetooth-enabled devices occurs in ad-hoc networks termed piconets. A piconet consists of one master and up to seven active slaves that share the available bandwidth on a time-division basis for details of the Bluetooth radio). Although the technical specification permits up to seven active slaves, only the most recent hardware devices support point-to-multipoint functionality. Without such capabilities piconets are generally limited to two members, one master and one slave. Both voice and data packet types are supported in a piconet. This project is only concerned with data packets, which are transmitted across asynchronous connectionless (ACL) links, as opposed to voice packets which use synchronous connection-oriented (SCO) links. ACL packets themselves differ in terms of error correction used and size in order to offer flexible bandwidth allocation. Table 3.1 lists the available ACL packet types and their characteristics. DH packets are high speed packets occupying one, three or five transmission slots with no forward error correction (FEC). DM packets are medium speed packets also occupying one, three or five transmission slots, but with 2/3 FEC applied to the payload to allow for correction of errors that occur during transmission.

Symmetric transmissions signify that both the master and the slave are using the same type of packet, as opposed to asymmetric transmissions where different packet types are used by each party. Refer to the specification for a detailed explanation of the different packet types available.

Table 3.1: ACL packet type characteristics

Type	User payload (bytes)	FEC	Symmetric max rate (Kbps)	Asymmetric max rate (Kbps)	
				forward	reverse
DM1	0–17	2/3	108.8	108.8	108.8
DH1	0–27	no	172.8	172.8	172.8
DM3	0–121	2/3	258.1	387.2	54.4
DH3	0–183	no	390.4	585.6	86.4
DM5	0–224	2/3	286.7	477.8	36.3
DH5	0–339	no	433.9	723.2	57.6

Source: Specification of the Bluetooth System, Volume 1

3.1.2 Stack layers

The Bluetooth protocol consists of several layers forming a protocol stack.At the bottom of the stack are the firmware layers that are usually implemented as part of the Bluetooth device itself. The Radio and the Link Controller perform low-level functions such as timing and error correction and cannot be accessed directly by the programmer. The link manager implements control functions such as link setup between devices and power modes. These are features that an application will want access to in order to control the operation of the device. The upper layers of the stack are usually implemented in software. This means that they run as an executable on the host system and applications requiring Bluetooth functionality must communicate with the executable. The actual Bluetooth hardware can be connected to a host using a variety of transport media such as UART, USB and PCMCIA. To simplify development an abstraction layer exists. It is called the Host Controller Interface (HCI) and provides a standardised interface between the HCI Driver (implemented in software on the host) and the Host Controller layer (residing in firmware on the Bluetooth device). The Host Controller firmware layer must be capable of handling all commands defined in the specification — however, the capabilities of a particular HCI Driver implementation defines the set of functions that are accessible to applications running on the host. This makes the software stack a critical component in a Bluetooth system.

The L2CAP layer is an abstraction layer that hides the complexities of the underlying transport protocol. L2CAP is short for Logical Link Control and Adaption Protocol and the layer performs tasks such as fragmentation, reassembly and higher layer multiplexing, allowing existing protocols such as TCP/IP to run unmodified over Bluetooth. The RFCOMM layer is a serial port emulation protocol that is intended primarily for adapting existing applications so that they can use Bluetooth. A Bluetooth link similar to the use of modems for dial-up networking. This is achieved using an additional protocol layer on top of the Bluetooth protocol stack, called the point-to-point protocol (PPP). In the early days of computer communications, modems were connected externally to the serial port of a computer. PPP was developed to handle modem connections established between two computers and provide a route for IP packets. The combination of PPP's ability to support links over serial ports and the RFCOMM layer's serial port emulation provides a method for carrying IP traffic between hosts using Bluetooth technology.

The boundary between the firmware and software layers is flexible. In this section the structure shown is that of a typical development kit implementation where the developer cannot change the firmware layers.

For further information on the basics of Bluetooth technology consult the core Bluetooth specification or one of the several books on the subject .

3.2 Interference

3.2.1 The 2.4 GHz ISM band

Wireless communication systems use one or more carrier frequencies, collectively termed as a frequency band, to communicate. Bluetooth, 802.11 wireless LAN (also known as Wi-Fi) and HomeRF wireless home networking all share the 2.4 GHz portion of the license-free Industrial, Scientific and Medical (ISM) band, which extends from 2.4 to 2.4835 GHz. In order to ensure fair access to the medium all products must use one of two transmission methods: frequencyhopping spread spectrum (FHSS) or direct-sequence spread spectrum (DSSS).

3.2.2 Radio characteristics of 802.11 and Bluetooth

Bluetooth is based on FHSS, where data is transmitted on a frequency that is regularly changed to another frequency in a pseudo-random pattern known to both transmitter and receiver. Both the transmitter and the receiver 'hop' between frequencies based on the same pattern, transferring a piece of data during each hop. In the case of Bluetooth technology the hops occur across 79 frequencies, each 1 MHz wide, at a rate of 1600 hops per second. The time spent at a single frequency is referred to as a slot, and packets occupying one, three or five slots may be used. Properly synchronised, the net effect is to maintain a single logical channel. The use of multiple frequencies also means that if a packet is corrupted due to interference it will be resent on the next frequency in the sequence, which will hopefully be clear. 802.11 is a group name for a number of wireless data communication standards issued by the Institute of Electrical and Electronic Engineers (IEEE). The most common such standard in use today is 802.11b, more commonly known as Wi-Fi, which provides a specification for high-rate wireless local area networks(WLANs). The remainder of this document will use the terms Wi-Fi and WLAN interchangeably. Although it has a greater range and through put than Bluetooth it also consumes much more power, making it a poor choice for small battery-operated devices. Wi-Fi uses DSSS, a technique where transmitters spread the signal over a frequency band that is wider than required to accommodate the information signal by mapping each bit of data into a redundant bit pattern. Wi-Fi uses a transmitted signal width of 17 MHz that can be located in different areas of the ISM band [11]. A total of 11 channels are allowed in the standard and each channel is defined by its centre frequency. Three channels are non-overlapping and thus allow up to three Wi-Fi networks to coexist in time and space without interfering with each other.

Despite sharing the same frequency spectrum Bluetooth and Wi-Fi should be considered complementary technologies. There are many devices such as notebook computers that might use Bluetooth for wireless peripheral connections and Wi-Fi for wireless network access. Ideally it should be possible to collocate the two wireless subsystems (within 20 cm or less) and have them continue to operate simultaneously. Presently the simultaneous operation of Bluetooth and 802.11b in close proximity leads to degradation in the performance of both systems. To determine the extent of the interference problem there has

been a substantial amount of research into the coexistence of different wireless technologies in the 2.4 GHz band .However, there are several scenarios that have not been studied.

3.3 Reconfigurable computing

3.3.1 Basic concepts

Reconfigurable computing refers to computing hardware that can alter its design to suit a particular task at hand [24]. Such adaptability is possible through the use of Field Programmable Gate Arrays (FPGAs) which are a type of logic chip that can be reprogrammed at a very high rate1. FPGAs do not rely on the use of a central processing unit to perform a task, as a normal computer does. Instead the logical structure of the chip itself determines what functions are performed on the input data. As a result, a well-designed program implemented in an FPGA can perform several tasks in parallel rather than processing data sequentially like a normal computer. The performance gains can be very high in demanding applications where parallelism can be exploited, such as real-time video image processing [12], encryption [8] and data compression [22].

Although FPGAs can provide impressive performance gains they belonged solely to the domain of hardware engineers until recently. Early FPGAs took several seconds to reprogram and were mainly used for prototyping integrated circuit designs. A hardware engineer would take a set of software functions, describe them in a hardware description language (HDL) and reprogram the FPGA from time to time to test the design. In the last few years such tools have become accessible to software engineers. Not only can modern FPGAs reconfigure themselves thousands of times per second, but it is also possible to use special programming languages and compilers to implement software functions directly in hardware, without the need for using a HDL [21]. This combines the flexibility of software development with the increased performance of hardware.

An example of such a software tool is Celoxica's Handel-C programming language, which is introduced in the next section.

3.3.2 Handel-C

Handel-C is a specially developed programming language marketed by Celoxica Ltd. in the UK. It is based around the syntax of ANSI C but contains extensions necessary for hardware development. It compiles directly to a hardware specification, and the extensions allow the developer to take advantage of hardware level features such as parallel execution, flexible data path widths and interfaces to external hardware. By using the Handel-C compiler and simulator a software engineer can write and debug applications intended for hardware before configuring the FPGA.

3.4 Available Bluetooth technologies

Development of solutions for Bluetooth requires both hardware modules and a software protocol stack. Hardware modules are available in a variety of configurations ranging in complexity, performance and flexibility. Software stacks are abundant but differ widely in features, cost and ease of use. This section is intended to provide a basic overview of the different options available when choosing Bluetooth hardware and a protocol stack. The following chapter looks at how the infrastructure for this project was developed and highlights the importance of the stack.

3.4.1 Hardware

A Bluetooth interface consists of three functional components at the lowest level corresponding to the firmware stack layers in section 2.2.2: Radio, Link Controller and Link Manager2. Most implementation strategies rely on the use of existing Application Specific Standard Products (ASSPs) to provide these functional components or combinations thereof in a massproduced, integrated module. Bluetooth interfaces can be organised into three different categories: Original ASSPs, development kits and dongles.

Original ASSPs. ASSPs from different manufacturers implement different combinations of the three stack layers. For the purpose of this section a summary of the possible configurations is shown in Table 3.2.

Table 3.2: Different functional configurations in Bluetooth ASSPs

Radio	Link Controller	Link Manager
√	√	√
	√	√
	√	

Source: Xilinx White Paper [27]

The combination of stack layers included in an ASSP determine the flexibility and workload involved in integrating such a unit into a system level design. If an ASSP provides all three layers integration can often be achieved by connection to the host system's USB or UART port. Such solutions provide ease-of-use at the expense of flexibility, restricting the choice of features to that found on the ASSP. ASSPs providing only link controller functions require a larger amount of design work but may result in closer system integration [27]. Development kits. Development kits from a variety of vendors3 are available on the market and offer a fast route to getting started with Bluetooth application development. Simple kits consist of an ASSP providing radio, link controller and link manager functionality, together with a software protocol stack and the necessary connectors and cables. A UART connection allows kits to connect to a large variety of host systems at the expense of speed, while a USB connection enables Bluetooth to run at maximum speed, provided the host supports USB4. The kits usually support most of the functionality contained in the Bluetooth specification, including the optional features. Some sample programs are often provided. More advanced kits offer features such as audio in/out connections, choice of radio transmit power and more. The cost of development kits varies between suppliers but at the time of writing a simple kit from Teleca Comtec (formerly Sigma Comtec) costs USD $500 for academic institutions and USD $1500 for commercial organisations. Advanced kits cost in excess of USD $3000.

Dongles. A Bluetooth dongle can be loosely defined as a small device that is externally attached to a system in order to provide Bluetooth functionality. Dongles consist of the same type of ASSP as a development kit although they do not necessarily implement any of

the optional parts of the Bluetooth specification. Usually they provide only one HCI transport layer option and come bundled with a software stack and some utility programs. Bluetooth dongles are not yet common in the marketplace.

3.4.2 Software

In addition to selecting hardware it is necessary to decide on what stack to use(recall that a host system communicates with the Bluetooth interface using a software protocol stack). Stacks fall into three categories: 1) those that are proprietary, 2) those that are freely available but only in compiled form and 3) those that are freely available and open source5. The remainder of this section explains the difference between these categories.

Proprietary. Proprietary stacks are developed by a company either for use with its own Bluetooth products or for licensing the technology to others. Development kits and dongles often come with a proprietary stack from the hardware manufacturer. Their purpose is to allow the particular product to communicate with the host system and may not be re-distributed with any other hardware unless licences for commercial use are purchased. An example of such a stack is the Ericsson Bluetooth Software6 distributed by Teleca Comtec. At the time of writing the licence fee is USD $40000 per project plus a royalty fee of USD$0.60 per unit if more than 10000 units are produced. Free, compiled. This type of stack is free for evaluation and non-commercial use but is only available in executable form. This makes the user reliant on the stack provider for updates and improvements. At the time of writing there exists at least one such stack for Linux and one for Windows — IBM's BlueDrekar and CStack, respectively.

Free, open source. Open source stacks have been developed and released by a few companies. Initially they were used internally to support a specific Bluetooth product developed by the company, such as Axis' Bluetooth access points. The source code was then released to the public in the hope that development would accelerate. Currently such stacks are only available for Linux although a Java stack is reportedly under development. Because open source stacks are free, they are often works-in-progress and poorly documented. Getting started with such software can require a considerable amount of time.

3.5 Summary

This chapter has presented an overview of Bluetooth technology, the topic of interference and an introduction to reconfigurable computing. The types of hardware and software tools available for developing solutions using Bluetooth have been described and found to be highly diverse. Preparing the infrastructure for a development project can require a considerable amount of time and the following chapter discusses how to facilitate the process.

4.APPLICATION DESIGN AND STACK SELECTION

4.1 Introduction

Deciding what resources to use for a particular Bluetooth project is a complex procedure. The optimal combination of hardware and software depends on individual product requirements, experience and a variety of other factors. The specifications of Bluetooth hardware are often quite clearly stated and can be examined to determine if a particular device meets a particular set of requirements. Choosing an appropriate stack, however, is more difficult yet one of the most important decisions made at the beginning of a project. Any application developed is stack-specific — it is not possible to run an application written for stack A on a host running stack B1. At presents Bluetooth literature consists mostly of books that summarize the specification or technical reports that study a particular aspect of the technology such as .To date there are no publications covering application design and stack selection for Bluetooth solutions. This chapter attempts to fill the void by presenting a classification of a stack features and explaining how the stack forms part of the overall application design.

An example program will be used to illustrate how stack interactions function. The program is designed to measure the throughput between two hosts, and details of the implementation are provided in chapter 4. After describing the requirements of the application, an introduction to the hardware equipment used in this project is provided in order to set the context. Next a classification of stack features is introduced and applied to a variety of common stacks. Several potential stacks are examined more closely and one is finally chosen as suitable for this project. Lastly, the example program is re-examined to see how a stack allows it to operate using Bluetooth.

4.2 Sample application without stack

In order to measure the throughput between two hosts it is necessary to have two applications, each implemented on a separate host. The server application runs on one host

and waits for connections. The client application is then started with certain parameters on the other host. It will attempt to connect to the server and send a known amount of data. As the data is received, the server records the time at the start of the transfer and counts the amount of data. When the client has sent all the data it disconnects. The server then records the time at the end of the transfer, and can calculate the overall throughput by dividing the total time spent receiving the data with the amount of data received. Figure 4.1 contains a block diagram showing the interactions and flow of data between the client and the server applications. To use the applications described it is necessary to both configure the transport medium and send the data across the transport medium. Parameters such as connection speed and data quantity need to be set on the client side before the connection is established. The remainder of this chapter describes the resources required to implement such an application over Bluetooth. Finally, the sample application is re-examined to illustrate how Bluetooth fits into the picture.

4.3 Hardware

At the outset of this project the Department of Computing had no Bluetooth hardware available. In order to meet the conditions of the client-server model, two Bluetooth devices capable of communicating with each other out of the box were required. Standard ASSPs did not fit this criteria as they do not come bundled with a software stack. Dongles also did not fit this criteria as there

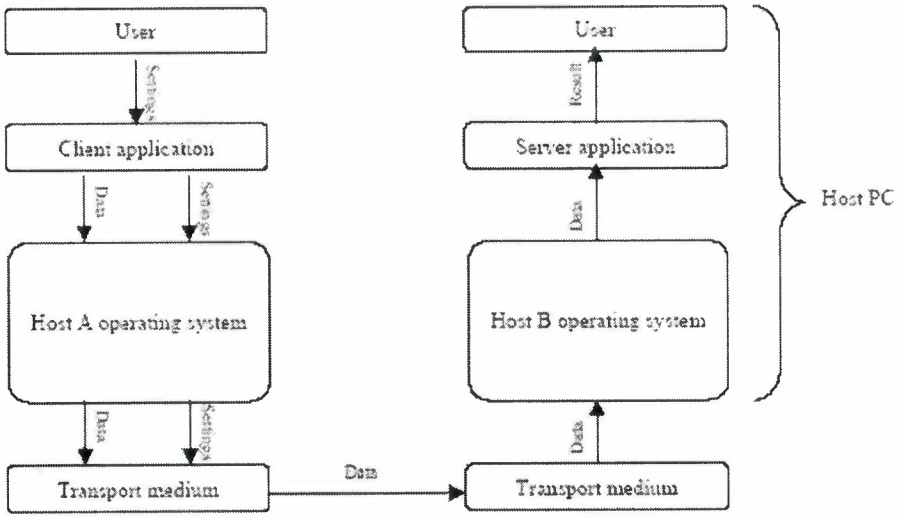

Figure 4.1: A generic throughput measurement application

Table 4.1: Features of the Ericsson Application and Training Toolkit

Feature	Value
External connectors	UART & USB
Range	10 metres
Point-to-Multipoint support	No
Stack	Ericsson Host Reference Stack
Firmware	ROK 101 008

were none easily accessible on the market. Although advanced development kits offer a wide variety of functions, they were costly and contained superfluous features. After reviewing the specifications of several simpler development kits, the choice was made to acquire two Ericsson Application and Training Toolkits from Teleca Comtec in Sweden. The details of these simple development kits are listed in Table 4.1.(It is above)

The development kits came bundled with a chat program and a connection test program, both with source code. These client/server based programs enabled two PCs running the Windows operating system to connect to each other and execute the applications. Once suitable hardware has been acquired the developer must decide what software protocol

stack to use. This is the decision where the developer has the most flexibility, and the following sections focus on the requirements and features of stacks.

4.4 Analysis of stack requirements

The first step in choosing a stack is to define what functionality the project demands from a stack. In order to do this it is necessary to have an understanding of the features that differentiate one stack from the next. Some are suitable for rapid prototyping while others are commercial-grade stacks portable across several operating systems. This section introduces for the first time the notions of primary and secondary stack features as a means to distinguish between stacks at a high level.

4.4.1 Primary features

Primary features are those that define the functionality of a stack. They are mostly static, determined by the developer and often publicly known. A list of such features and their significance is given below.

*Operating System (OS): Currently there are stacks available for Linux and Windows. The choice of OS should be determined by the objective of the project, as well as how comfortable the developer is with working with the particular OS. A developer familiar with Microsoft Visual Studio may find the transition to Linux time consuming and frustrating, and vice versa.

*Cost: Depending on the type of stack one may or may not have to pay to acquire it. Proprietary stacks tend to provide a more complete API than others, but free open source stacks are unrestricted in terms of use and redistribution2.

* Source availability: The availability of the complete source code for a stack makes it possible to optimise it for a particular task or port it to a certain environment. This is necessary if one wishes to develop and distribute embedded solutions. For evaluation and

training purposes stacks distributed in compiled form may still be a viable solution. However, if the stack is distributed in compiled form one cannot modify its functionality.

* Transport layers: The transport layers supported by a stack affect the type of connectors that can be used to attach a Bluetooth device. This in turn determines the maximum speed that a Bluetooth link can support. The most common transport layers are USB and UART (see section 2.5.1 for more information on transport layers). Certain stacks also support the PC Card layer for use with PCMCIA Bluetooth devices.

4.4.2 Secondary features

Secondary features are those that have no impact on a stack's functionality but facilitate the development process. They can make the difference between a few hours or a few days spent solving a problem. Unfortunately the quality of these features is not often known at the time when a stack is selected.

*Documentation: As with any complex software, a stack with up-to-date, comprehensive documentation can save considerable amounts of time. However, such is rarely the case for the open source stacks as they are works-in-progress.

* Sample code: Some stacks are based on virtual serial ports, others use socket communications while yet others use virtual operating systems. Understandable sample code helps the developer grasp the syntax of stack interactions and accelerates development.

* Support: Support for stacks mainly comes in the form of mailing lists or newsgroups. These vary in popularity and helpfulness.

4.4.3 Project stack requirements

Before comparing different stacks, the requirements for this project can be placed within the framework introduced in section 3.4.

* OS: Not important, the developer is familiar with the operation of Windows and Linux.

* Cost: Important, as use of the stack must not incur any additional costs.

* Source availability: Not important, due to the exploratory nature of the project.

* Transport layers: Important, because the application measures throughput and the full range of Bluetooth speeds is desired.

* Secondary features: Important, as there is no access to local expertise.

4.5 Stack comparison

After determining the project requirements a review of available stacks is necessary. This section examines six stacks in common use and their features.

Ericsson Host Reference Stack: This stack is developed by Ericsson, one of the founding members of the Bluetooth Special Interest Group (SIG). It comes bundled with the Ericsson toolkits available from Teleca Comtec and can also be licensed independently.

Digianswer Bluetooth Software Suite: This stack comes bundled with Digianswer's PCMCIA Bluetooth toolkits and products.

CStack: The origin of this stack is unclear. It is available for download3 but there seems to be no company backing it.

Axis OpenBT: The Axis stack was originally developed for use in the company's own access point products. It is the stack that has been open source the longest and is continuously evolving.

Qualcomm BlueZ: This stack has only recently been released under an open source licence but is growing in popularity. It is included in the Linux kernel distribution and can thus be considered the 'official' Bluetooth stack for Linux.

IBM BlueDrekar: IBM's stack has been available for over a year and is available for download. It does not appear to be very popular based on the low number of posts in the mailing list archives.

Table 4.2 presents a chart containing the primary features of each stack and the requirements for this project. By presenting the available options in this format a developer can quickly determine which stacks should be suitable for his or her purposes. Typically this is all the information that a developer has access

Table 4.2: Stack feature comparison

	OS	Cost	Source	Transport layers
Ericsson	Windows	Free w/ kit	No	UART/USB
Digianswer	Windows	New kit	No	PC Card
CStack	Windows	0	No	UART only
OpenBT	Linux	0	Yes	UART/USB
BlueZ	Linux	0	Yes	UART/USB
BlueDrekar	Linux	0	No	UART only
Requirements	—	0	—	USB

to when selecting a stack, and it should be emphasised that such information gives no indication as to the ease of use associated with a particular stack. Rather than selecting a stack on the basis of primary features alone, a shortlist of potential stacks should be prepared and investigated. For this project it is clear from Table 4.2 that three stacks may be suitable: Ericsson, OpenBT and BlueZ. What is not clear, however, is which stack to choose. The detailed technical and secondary features for each of the shortlisted stacks have been reviewed and are presented in the following section.

4.6 Stack evaluation and selection

This section presents the three shortlisted stacks in the order that they were evaluated. For each stack the technical features and quality of the secondary features are discussed.

4.6.1 Ericsson Host Reference stack

The Ericsson stack was the first stack evaluated in this project. It is a sample implementation of the Bluetooth Host Stack adapted for the Windows OS. The stack runs as a process in the background and uses a virtual operating system to interface with user applications based on Microsoft's Component Object Model (COM). All the upper layers in the Bluetooth protocol stack are supported.

According to the sales literature the stack can be ported to real-time operating systems for use in embedded systems. Although the stack was delivered with documentation covering all the API commands available there were no examples of how to use them in practice. Two sample applications were included, written in a mix of C and C++ and using Microsoft Foundation Classes. They worked in Windows 2000 most of the time but the source code was advanced and specific to the Microsoft Visual Studio development environment. There is no public mailing list dedicated to use of the stack. In light of the above it was decided early on to evaluate alternative stacks. The stack was simply not suitable for introductory-level applications. In addition, the fact that the stack requires a licence for redistribution was considered restrictive, limiting the potential uses of any software produced.

4.6.2 OpenBT stack

Next the OpenBT stack for the Linux operating system was evaluated. OpenBT is written in C and provides support for all stack layers up to and including RFCOMM, which is also the main layer used for communications. The stack has reportedly been ported to an embedded system using the ETRAX processor from Axis.

OpenBT can run in either kernel mode or user mode. This means that the code is either executed within the kernel space or user space. The ability to choose the mode allows for safe prototyping of applications in user mode, where the worst that can happen is a core

dump. When the application is complete the kernel mode of the stack can be used to improve efficiency and, if required, port the complete solution to an embedded system.

When developing solutions using OpenBT, applications communicate via emulated serial ports at the RFCOMM layer. This allows data to be sent and received by writing to or reading from the emulated ports as if they were standard serial ports. The device /dev/ttyBTC represents the control channel used to issue HCI control commands for an explanation of the HCI layer) while devices /dev/ttyBT0 through /dev/ttyBT7 represent the emulated serial ports used for data communications.

Although certain HCI functions from the Bluetooth specification are not included in the stack they can be added by studying the specification and modifying the stack source code. It may require some time to understand how to add the desired functionality but will also improve one's understanding of the stack. This project contributed the function that is now included in the official OpenBT distribution. It creates an HCI packet requesting the value of the country code parameter and sends it to the HCI driver. A part of the code contributed is shown below:

```
// btcommon.h
#define HCIREADCOUNTRYCODE IOR(BT_IOC_MAGIC, 0X43, S32)
// hci.c
s32 hci_read_country_code(void) {
s32 tmp;
D_CMD(__FUNCTION__"\n");
c_pkt.type = CMD_PKT;
c_pkt.opcode = hci_put_opcode(READ_COUNTRY_CODE, HCI_IP);
c_pkt.len = 0;
tmp=send_cmd((u8*)&c_pkt,c_pkt.len+CMD_HDR_LEN+HCI_HDR_LEN);
if (tmp < 0) {
return tmp;
} else {
```

```
return result_param;
}
}
```

Because OpenBT is an open source stack it is downloaded as source code and compiled locally. Unfortunately the compilation process failed on all four Linux distributions tested in this project unless parts of the Makefiles were modified The documentation was outdated and did not mention several important changes that affected the operation of the stack as a whole. There were a few sample command-line applications available, and for support there was an active mailing list covering questions or suggestions regarding the stack. OpenBT was at one stage intended as the main stack used for this project. USB did not work at first, which restricted the speed of the Bluetooth devices, and development proceeded in parallel with searching for a solution to the USB problem. After a great deal of effort it became apparent that the combination of OpenBT, USB and Ericsson toolkits did not work. In addition stack instability was apparent when connecting and disconnecting the two devices multiple times.

4.6.3 BlueZ stack

In an attempt to get USB working the BlueZ stack was tested. It supports the same layers as OpenBT although it does not provide a complete API for using the RFCOMM layer. Because it is a relatively recent addition to the list of available stacks there have been no reports of anyone using it in an embedded system. Unlike OpenBT, BlueZ only runs in kernel mode. The individual stack layers are compiled into object files and inserted into the kernel using the modprobe command. This means that the stack is highly responsive but any errors can cause instability in the host OS. During the implementation phase this was observed several times.

Although BlueZ does not offer complete support for the RFCOMM layer it does provide a socket-based approach to Bluetooth communications at the L2CAP layer. This makes the stack easily accessible to developers familiar network applications using TCP/IP sockets.

The following code fragment shows how the sample server application might wait for an incoming connection:

```
// Server accept connection example
// Create socket, bind and listen
int s = socket(PF_BLUETOOTH, SOCK_SEQPACKET, BTPROTO_L2CAP);
bind(s, (struct sockaddr *) &loc_addr, sizeof(loc_addr));
listen(s, 10);
// Accept incoming connections
while(1) {
if((s1=accept(s,(struct sockaddr *)&rem_addr,&opt))< 0)
// ... handle the connection
}
```

After establishing a connection between two applications one side waits for data to arrive on the socket (server) while the other side writes data to the socket (client).

```
// Client send data example, after connection
for (i = 0; i < limit; i++){
send(s, buf, i, 0);
}
// Server receive data example, after connection
while(1) {
r = recv(s, buf, data_size,0);
}
```

With release 1.2 the documentation was more comprehensive than that of OpenBT. Several sample applications were provided to configure the devices and test connection capabilities. USB is supported and worked with Ericsson toolkits, which was essential for this project. Although BlueZ had not been available for long it displayed advantages over OpenBT for

this project. The latter was originally developed to support the needs of a Bluetooth access point, and some of the its features still reflect this target application. While this may be useful to some, it was found to be distracting during the development of this project. The following section revisits the sample application from the beginning of this chapter to illustrate how BlueZ was used to enable interaction between the client and server application.

4.7 Sample application with stack

It is now possible to show how Bluetooth hardware and a software stack can be used in conjunction with the sample application from section 3.2. The following pseudo-code clarifies the operations required for server and client operation:

```
// Server
Open L2CAP socket
Listen on socket for incoming connections
Take timestamp when connection established
While connected
Read socket to receive data
Count data received
End while
Take timestamp when connection dropped
Calculate throughput
Exit and return result
// Client
Enter test parameters
Open HCI socket
Write device configuration parameters
Close HCI socket
Open L2CAP socket
Connect L2CAP socket to remote device
```

While data left to send
Write data to socket
End while
Close connection
Exit

First the modules for the HCI Driver and L2CAP layers are inserted into the Linux kernel. Next the server is started and waits on a socket for an incoming L2CAP connection. The user then starts the client application and enters the settings to use for the test. The settings are sent to the Bluetooth device via the HCI Driver kernel module and an L2CAP socket is opened to send the data via the L2CAP kernel module. Notice that all interaction between the host PC and the Bluetooth device occurs via the HCI Driver. This is because it is the abstraction layer responsible for communicating with a Bluetooth device across a variety of transport layers such as USB, UART and PCMCIA. Figure 4.2 shows how the stack is used to support interaction between the application and the Bluetooth devices. It is important to understand that a particular application must be used in conjunction with the stack that it was developed for. At the same time, however, Bluetooth communication between two different host applications running different stacks is possible. For instance, the client application above could be written using the OpenBT stack and the server application using BlueZ. In order to run the client on a host, the OpenBT stack would have to be installed on the host and similarly for BlueZ. The stack used determines how data connections are processed but not what processing needs to be performed at each layer — this is defined in the Bluetooth specification.

The application used as an example in this chapter will be specified more fully in the following chapter, which implements the client-server application to measure the effect of interference on a Bluetooth link.

4.8 Summary

This chapter has investigated how an infrastructure for Bluetooth development is built. It has presented a classification method for simplifying the software stack selection that is the first of its kind. Various popular stacks have been examined and the results show that there are a large variety of stacks available and that they differ widely in many aspects. Developers would benefit from enhanced documentation or literature describing the features and use of these stacks in a manner similar to that provided here. A sample application has been used throughout this chapter to clarify how a stack forms part of the overall design for a Bluetooth solution.

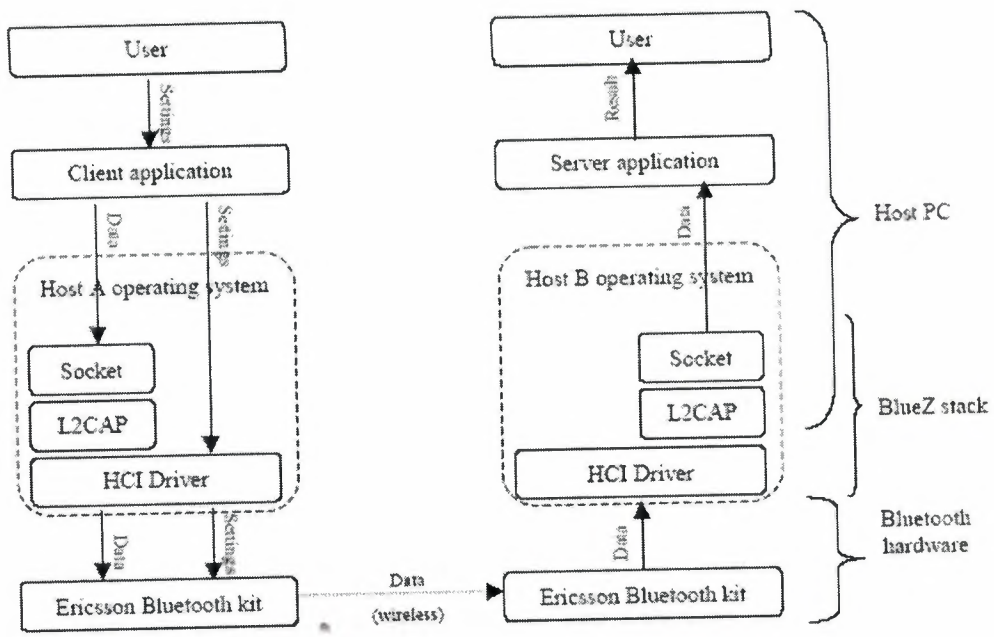

Figure 4.2: A Bluetooth throughput measurement application using BlueZ

5.INTERFERENCE: A SOFTWARE STUDY

5.1 Introduction

The topic of harmful interference between Bluetooth and WLAN was introduced in chapter 2. This chapter describes the development of an application used to measure the effect of interference on Bluetooth data rates. The application is based on the BlueZ stack introduced in the previous chapter and offers the following contributions:

1. It provides a tool that measures the effect of interference on a Bluetooth link at the lowest layer in the stack, L2CAP.

2. The results obtained using the application reveal new information that complements existing research.

3. The application can be modified to measure throughput between devices that use other wireless technologies to communicate.

5.2 Modelling interference

The topic of interference between Bluetooth and other wireless technologies such as Wi-Fi has been studied by several commercial companies. To date the majority of research has been one-sided, primarily how Bluetooth interferes with a Wi-Fi network. A possible cause for the lack of studies considering the effect of interference on a Bluetooth piconet might be the lack of Bluetooth devices available. One report from Ericsson thus presents a theoretical model and uses it to speculate on the quality of a Bluetooth link in a Wi-Fi environment . The model is based on DH1 packets only, however, which offer low speed connections and are unlikely to be used in Bluetooth scenarios such as LAN-access. A second report, written by Texas Instruments, briefly presents a summary of some results obtained when using class 1 Bluetooth devices1 . Such Bluetooth devices are not common, however, and not likely to be used in embedded devices due to their high power consumption. There is thus an urgent need for further tests to fill the gaps left by these reports — in particular, interference tests covering high-speed Bluetooth links between standard devices. The

between standard devices. The model is based on the probability of Bluetooth and Wi-Fi devices transmitting simultaneously on the same frequency, as shown in equation 5.1.

$$P(Frequency\ overlap) = \frac{Number\ of\ overlapping\ channels}{Number\ of\ Bluetooth\ channels} \quad (5.1)$$

Recall that a Wi-Fi channel is typically 17 MHz wide and is located within the 79 frequencies that Bluetooth uses. This means that 22% of the 79 frequencies used by Bluetooth will overlap with the Wi-Fi channel. If the Wi-Fi station is transmitting continuously in the vicinity of a Bluetooth piconet then all Bluetooth packets sent on the overlapping frequencies are subject to interference. Whether or not the interference will corrupt the Bluetooth packet depends on the signal to noise ratio at the receiving device. The signal strength depends on the distance between the transmitter and the receiver, the transmit power used and the geography of the environment. The noise level depends on the interferer's transmit power and distance from the affected receiver. By analysing the signal strengths of Bluetooth and Wi-Fi devices it is possible to determine at what range a Wi-Fi station will produce sufficient noise to drown out the Bluetooth signal. Microwave signals such as those used in the ISM tend to attenuate rapidly, and even when two Bluetooth devices are only a metre away from each other a Wi-Fi station anywhere within 12 metres from the receiver will cause interference.

Although the model permits the variation of input parameters, such as signal strengths, the purpose of the interference study is not to verify the range at which interference becomes a problem. Instead it attempts to determine the extent of the effect when interference is occurring. Assuming that the receiving Bluetooth device is within the radius of interference, the theoretical maximum packet loss is 22%. This represents the worst case scenario, when a WLAN channel filled with traffic, causing 22% of packets to become corrupted and resent in the following hop. Confirming or disproving that percentage is the reason for developing the application described in the remainder of this chapter.

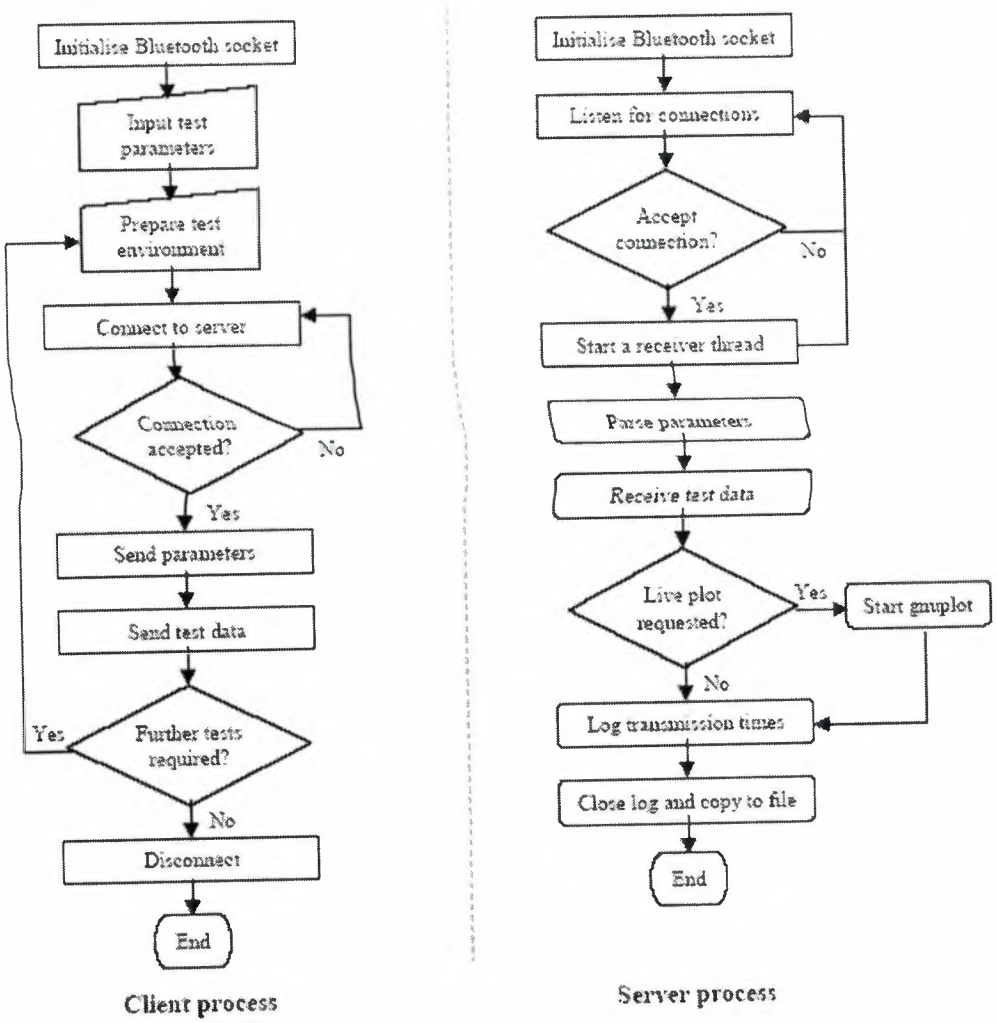

Figure 5.1: Interference application flowchart

5.3 Architecture

In order to test the theoretical results from the model the sample application from chapter 3 was implemented and extended to provide additional capability. The server waits for incoming Bluetooth connections and creates a new thread to handle each connection request. The client connects to the server, negotiates the session parameters and transmits data across the link. The server times the arrival of the data and writes the results to a log file for analysis. The log can be imported into Excel at a later stage as well as being plotted

in real time for visual feedback. Figure 5.1 shows the operational flow of the client and server applications.

The development environment consisted of two host PCs running Mandrake Linux 8.0. Each PC had version 1.1 of the BlueZ protocol stack installed and an Ericsson Application and Training Toolkit attached to the USB port. The structure of the development environment is given in Figure 5.2.

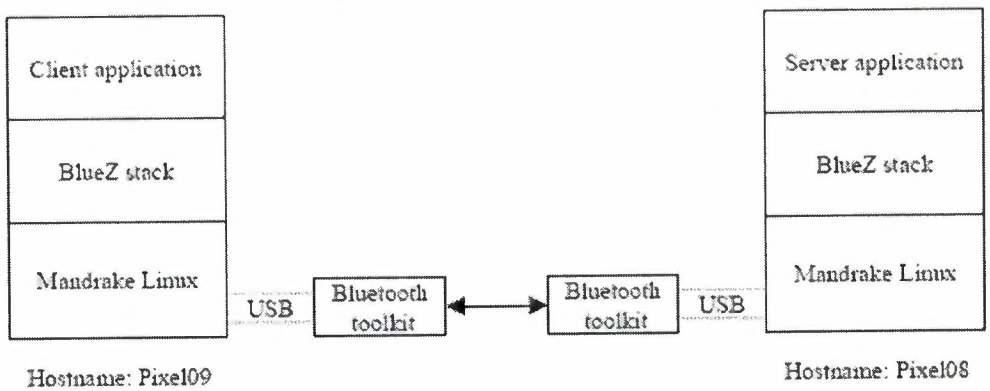

Figure 5.2: Structure of the development environment

5.4 Method

The design of the client and server application was based on the standard software development waterfall model. The requirements for the applications themselves were:

1) Measure the throughput on a Bluetooth link,

2) log detailed data for traffic analysis,

3) provide a menu-driven setup for ease of use and

4) plot the results for graphical analysis. By meeting these requirements the application should provide a useful tool for examining the effect of interference on a Bluetooth link.

Due to the nature of the advanced technology involved, documentation and other sources of information were scarce. As a result, although the structure of the application is not complicated, the method for accomplishing it over a Bluetooth link was not obvious. An iterative implementation approach was adopted where the results of each iteration were evaluated in order to determine how to proceed. This evolutionary method resulted in the following stages of development:

1. Establish a connection, send data and measure practical link throughput.

2. Understand how to control throughput using different packet types and buffer sizes.

3. Investigate possibility of using existing IP-based software to collect data.

4. Develop an application to log data and examine results.

The implementation of these four stages will be explained in the next section.

5.5 Implementation

The implementation of the interference application will be described in a chronological order to illustrate the challenges faced. This approach should prove particularly informative for anyone wishing to develop their own application based on Bluetooth technology.

5.5.1 Stage one

The aim of stage one was to establish a Bluetooth piconet between the two host PCs, send data from one to the other and measure the throughput. By using the socket interface at the L2CAP layer (as described in chapter 3) data buffers were sent from the client to the server over a Bluetooth link. The server timed the data transfer by taking a timestamp before and after it occurred, using the gettimeofday() function which provides microsecond accuracy. By dividing the number of bytes received by the time taken to complete the transfer a throughput measure was produced.

An important discovery emerged once data transfer was achieved and timed. The BlueZ stack does not automatically select the packet type required to maximise throughput. The throughput reported was around 12.5 kbps which suggested that DM1 packets were being used (see Table 3.1 for a listing of packet types and their associate link speeds). In order to perform the interference tests it would be necessary to have full control over the packet types used.

5.5.2 Stage two

The aim of stage two was to understand how to use different packet types in order to achieve variable transfer speeds. A Bluetooth device can transmit packets of different lengths as described in chapter 2. To maximise throughput on a Bluetooth link one device must transmit DH5 packets while the other transmits DH1 packets. Such an asymmetric arrangement results in the overall piconet bandwidth being divided into 723.2 kbps in one direction and 57.6 kbps the other (see Table 3.1 for other packet combinations).

Setting the packet type in BlueZ was possible using the hciconfig utility. The chosen packet types apply to outgoing packets under the condition that multislot packets, such as DH3 and DH5, will only be used if the amount of data to be sent exceeds the maximum payload of a smaller packet. This meant that it was not possible to create a symmetric DH5 link simply by configuring the stacks to use DH5 packets, as smaller packets would be used unless enough data was waiting to be transmitted.

Initial tests using DH5 packets yielded disappointing results. The throughput in practice was about 480 kbps in contrast to the theoretical speed of 723.2 kbps. Tests using other packet types yielded similar results. Another variable was found to affect throughput, namely the maximum transmission unit (mtu) used by the L2CAP layer. The optimal setting was determined at 2000 bytes, yielding a throughput of 693 kbps. This figured compared favourably with other reported results [15]. Figure 5.3 displays a graph comparing the theoretical throughput rates with those observed for each packet type. The differences

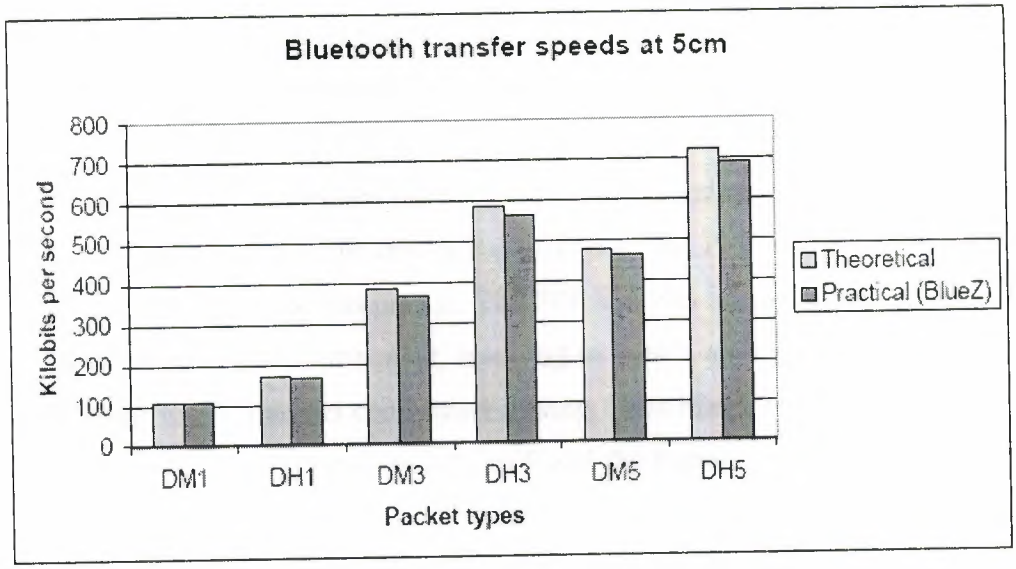

Figure 5.3: Comparison of theoretical and observed throughput

seen can be attributed to delays incurred as the result of using an external Bluetooth device.

Gaining full control over the transmission speeds through the use of different packet types meant that realistic throughput rates could be used in the measurement of interference effects. In order to collect the required data there were two options available:

1) Write an application specifically for Bluetooth or

2) use existing network monitoring software and carry IP traffic across the Bluetooth link.

Although the aim of the interference study was to produce an application for Bluetooth it was decided that the feasibility of carrying IP traffic should be investigated.

5.5.3 Stage three

The aim of stage three was to investigate the option of carrying IP traffic over a Bluetooth link. In the Bluetooth specification there are two usage scenarios that make use of such abilities: The dial-up networking profile and the LANaccess profile. They rely on the use of the RFCOMM layer of the Bluetooth protocol stack.

When two Bluetooth devices connect to each other at the RFCOMM layer it is as if an invisible serial cable connected them to each other. By correctly configuring and starting the PPP daemon pppd on each host a new route for IP traffic is set up. This can be used to implement Internet connection sharing facilities over Bluetooth. The BlueZ stack includes an application named rfcommd that is used to connect two Bluetooth devices at the RFCOMM layer. Once the connection has been established pppd can be launched to place a new route for IP traffic between the two hosts, assuming that PPP is correctly configured for the configuration defined for this project. The PPP link was initially tested using FTP to download files from one host to the other, resulting in speeds around 400 kbps. A second test was performed to test Internet connection sharing. Host Pixel09 was configured to act as a gateway routing traffic between host Pixel08 and the Internet over the PPP link. This allowed Pixel08 to be unplugged from the Ethernet LAN while remaining connected to the Internet via a Bluetooth piconet. The speeds reported were around 100 kbps.

Although the PPP trial showed that existing software could be used to perform interference tests there were two main reasons that argued against it:

1. The use of additional protocols above the L2CAP layer adds overhead to the Bluetooth link in the form of headers and processing time. This reduces the reported throughput and complicates the analysis. Results from using PPP links have been reported in and range from 550 kbps to 653 kbps. This is significantly less than the 692 kbps measured at the L2CAP layer in stage two.

2. Not all Bluetooth devices will require the use a TCP/IP stack. Many scenarios include Bluetooth components within embedded devices such as luggage tags , car keys and headsets . By measuring the effect of interference at the L2CAP layer no assumptions are made regarding the capabilities of the devices involved.

5.5.4 Stage four

The aim of stage four was to refine the client-server application from stage one in order to provide a detailed picture of the Bluetooth link characteristics at the L2CAP layer. In order

to do so timestamps were recorded after the arrival of each 2000-byte buffer and written to a log file for later analysis in Excel. Trial runs proved that the data provided an adequate level of detail, although occasionally the connection would not be established on the first attempt. To compensate for this the client was modified to perform repeated connection attempts:

```
static in MAXTRIES = 10;
for (i=1; i<=MAXTRIES; i++) {
if ( (s = do_connect(svr)) < 0 ){
syslog(LOG_ERR, "Can't connect. \%s", strerror(errno));
if (i == MAXTRIES)
exit(1);
} else {
syslog(LOG_INFO,"Connected after \%d attempts.", i);
break;
}
}
```

Before performing the full interference measurement study the application was refined to automate tasks such as switching packet types and organising the storage of logs. It was also found that integration with the package gnuplot made it possible to continuously plot the server logs during capture, thus providing a graphical view throughput as the data arrived at the server. The functionality of the final product is summarised in the following section. The interference study itself and the analysis of the results obtained is covered in section 5.7.

5.6 Functionality of final product

The final product consists of two applications written for the BlueZ open source Bluetooth stack. Together they operate in a client-server relationship to measure the throughput between two Bluetooth devices. Several features are provided to enable the user to efficiently collect and log measurement data for analysis.

The server application is intended to be used with a static Bluetooth device and provides the following functionality:

* Multithreaded for independent operation and multi-point support

* Writes logs to files in .csv format or gnuplot format

* gnuplot integration for optional real-time plotting The client application is intended for portable use, either on a laptop or on a desktop with a Bluetooth device connected to a long USB cable. It provides the following functionality:

* Menu-based command-line operation

* Choice of automated or manual testing

* Remote control of server parameters

The final product facilitates the collection of detailed throughput data on a Bluetooth link at the L2CAP layer. The following section describes how the application was used to study the effects of interference from a Wi-Fi station on a Bluetooth link.

5.7 Measuring interference effects

By using the application developed it was possible to collect a series of data sets measuring the effect of interference. This section discusses the design of the experiments, the results obtained and interesting observations made.

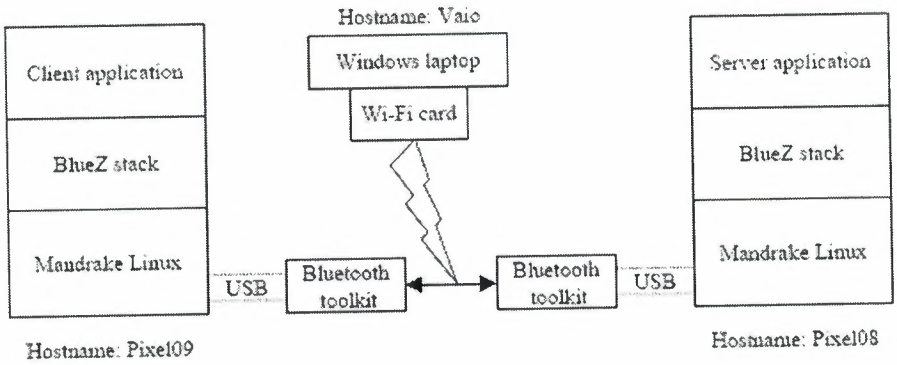

Figure 5.4: Structure of the interference test environment

5.7.1 Experiment design

Recall that the level of interference in a wireless system depends on the signal to noise ratio at the receiver. In the experiments described here the only variable affecting the received signal strength was the distance between the Bluetooth devices. The transmit power was constant at 1mW (0 dB/m) for a maximum range of 10 metres. The testing area was kept clear of obstacles to provide line-of-sight between the two devices. Lastly, the noise level was modified by varying the distance between the Wi-Fi interferer and the Bluetooth receiver. To perform the tests the Bluetooth devices were attached to the host PCs Pixel09 and Pixel08 using 5 metre USB cables. A Wi-Fi network card was installed on a laptop in order to provide a means for generating interference by constantly uploading files to an FTP server. Figure 5.4 provides an overview of the setup used.

5.7.2 Tests and results

A series of tests were performed based on the transmission of 2000000 bytes from the client to the server using an asymmetric DH5 link. A selection of the tests performed and the results obtained are described in this subsection.

Test one: The first test measured the throughput as the distance between the Bluetooth devices varied. It was performed using DH5 and DM5 packets without interference in order to determine the base case. The results are shown in Figure 5.5 and verify that the throughput remains reasonably stable across the operative range of Bluetooth. In the case of DH5 packets the throughput varies between 617–693 kbps.

Test two: The second test measured the throughput under interference conditions. The distance between the Bluetooth devices as well as the distance from the receiver to the source of the interference were varied. The results

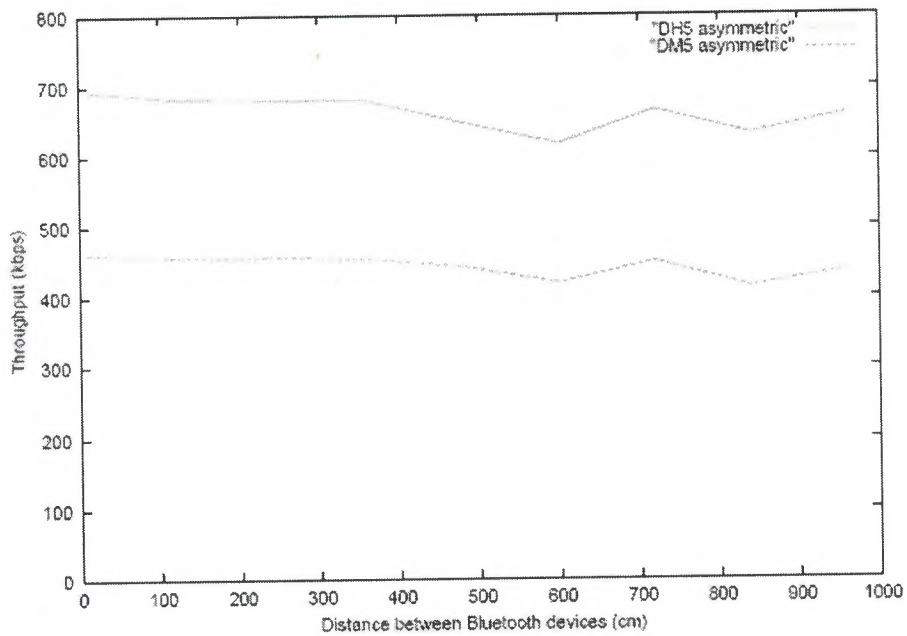

Figure 5.5: Throughput as function of distance in absence of interference

when using DH5 packets at 1 metre and 60 cm from the source of the interference are shown in Figure 5.6. The effect of the distance to the interferer does not appear significant except at around 7 metres. When the source of interference is one metre from the receiver the throughput varies between 381–500 kbps.

Figure 5.7 compares the results from test one and two to display the throughput with and significantly reduced. Across most of the range it represents a reduction of over 35%. The raw data collected also provides information on how long it took to transmit each 2000-byte buffer. This is useful for detailed traffic analysis. Figure5.8 displays the time in milliseconds that it took to send 2000 bytes. At the bottom of the chart the solid black line represents the data collected in the absence of interference. The highly fluctuating dotted black line above it represents the data collected in the presence of interference. The chart illustrates how the effect of interference leads to large fluctuations in transmission times at a low level.

Interesting non-quantitative observations were also made during the tests. If the source of interference came within half a metre from the receiving Bluetooth device it caused severe disruptions. Either the sender would fail entirely to connect or, if the interference was generated once a connection had been established, the operating system on the receiving host would crash.

Figure 5.6: Throughput as function of distance in presence of interference

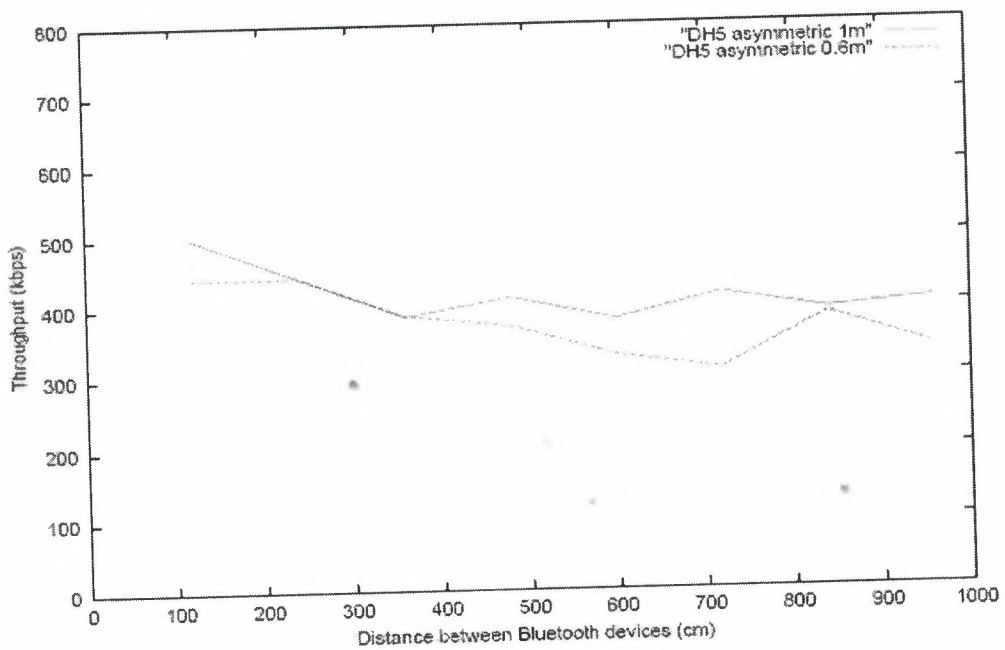

Figure 5.6: Throughput as function of distance in presence of interference

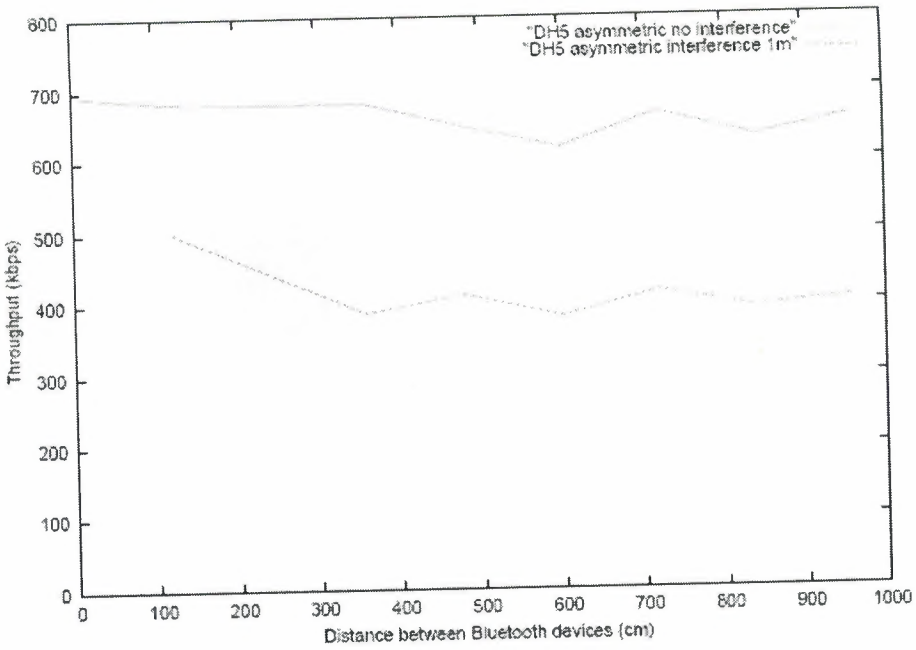

Figure 5.7: Raw buffer transmission times

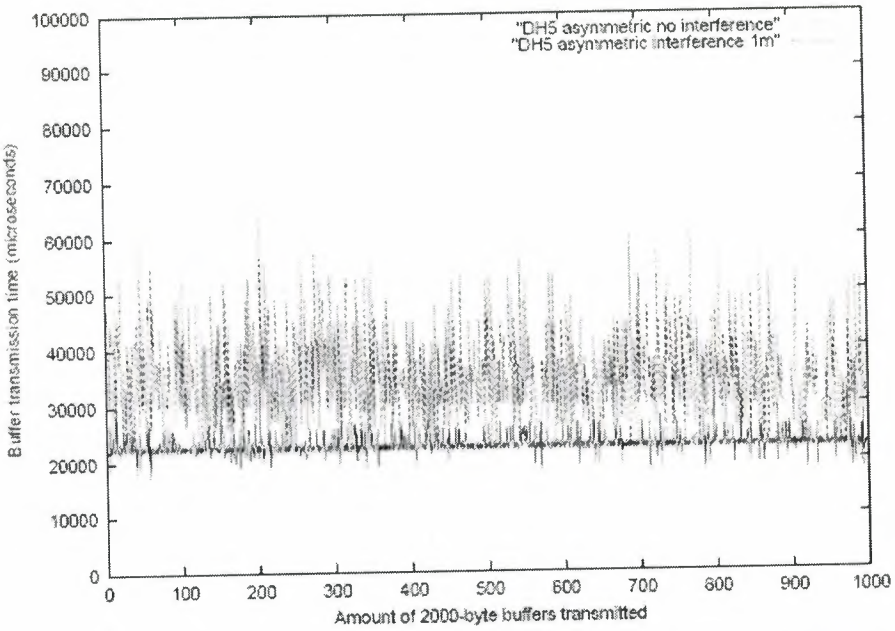

Figure 5.8: Comparison of transmission times in presence and absence of interference

5.8 Evaluation

5.8.1 Application

The requirements for the application were defined as:

1) Measure the throughput on a Bluetooth link,

2) log detailed data for traffic analysis,

3) provide a menu-driven setup for ease of use and

4) plot the results for graphical analysis.

The application developed meets the requirements above and serves its purpose of measuring the effect of interference. It is written in C and operates at the L2CAP layer to

minimise protocol overhead, thus generating the most accurate throughput figures. It is capable of supporting multi-point Bluetooth connections once such hardware becomes available. Furthermore, the applications can be adapted to analyse traffic across a Wi-Fi or HomeRF link for studying the effect of Bluetooth interference on a wireless LAN.

There are several possible extensions to the application that could be explored:

* Provide a component for triggering the Wi-Fi interferer remotely from the host where the client application is run. This would require a simple application running on the Wi-Fi host, waiting for a command on a TCP port and starting to send data when the command is received. Such an application could be written to simulate real-world WLAN traffic rather using constant uploading.

* Provide a server component for Windows to improve mobility through the use of a PocketPC. Digianswer [6] offer a PCMCIA card for Windows CE that could be used for such a purpose.

* Provide a server component for the OpenBT stack for cross-stack functionality. Such work would highlight the differences between the way the stacks operate and provide insight into potential compatibility issues.

One of the limitations encountered with the application was the strain caused to the host PC by the real-time visual analysis feature. When running tests based on DH5 packets the throughput could reach up to 693 kbps, and every incoming buffer caused the server to record a timestamp and write it to a log file. Each buffer consisted of 16000 bits (2000 bytes) which resulted in up to 43 timestamps recorded per second. In order to graphically display the results as they occurred gnuplot read the log file every second and updated the graph. This meant that towards the end of an interference test, typically based on 2000000 bytes, gnuplot was plotting over 900 points per second. The resulting strain on the processor prohibits detailed real-time analysis of the data in software applications. The

following chapter examines the potential of field programmable gate arrays to accelerate Bluetooth solutions through hardware. The possibility of improving the real-time analysis of the interference measurements is discussed there.

5.8.2 Results

Overall the effects of Wi-Fi interference on Bluetooth were worse than expected. The throughput reduction on an asymmetric DH5 connection was over 35% compared to the baseline throughput, as shown in Figure 5.7. Not only is this higher than the 22% suggested by the model, but when comparing with the theoretical speeds of 723.2 kbps it represents a loss in throughput of over 40%. Table 5.1 shows the loss in throughput relative to both the observed speeds and the theoretical speeds.

Table 5.1: Observed throughput loss as percentages of practical and theoretical maxima

	Throughput loss (%)	
Distance (cm)	Practical	Theoretical
120	27	31
240	35	39
360	43	46
480	36	43
600	38	47
720	37	42
840	38	46
960	38	44

Given that the results were much worse than predicted by the model might suggest that the model was flawed. It is more likely, however, that certain input parameters were incorrect. Recall equation 5.1 that defines the probability of frequency overlap between Bluetooth and a Wi-Fi channel. It states that the probability is a function of the number of Bluetooth frequencies used and the frequency width of a Wi-Fi channel. This probability varies positively with the width of a Wi-Fi channel. Hence one explanation for the results obtained is that the interference generated was covering more than 17 out of the 79 Bluetooth frequencies. The model was used to simulate the effect of increasing the Wi-Fi passband by 50% to 25 MHz, and using that input parameter the predicted reduction in

throughput matched the observations made. It is therefore possible that not all Wi-Fi devices correspond to the formal specification, and the generation of out-of-band noise should be considered when modelling interference.

The problem of out-of-band noise is complicated further in office environments with a high Wi-Fi density. Many companies are making extensive use of wireless LANs and to maximise network performance they might use several channels in the same area. This would increase the number of frequencies that are shared with Bluetooth and should cause considerable interference. If one busy Wi-Fi channel reduces the throughput in practice by up to 35% rather than 22%, it is possible that simultaneous use of multiple channels will prevent Bluetooth from functioning at all.

Perhaps the most disturbing results came from the adverse effects observed when the Wi-Fi interferer came within half a metre of the receiving Bluetooth device. Having the operating system crash during transmission might be due to the stack used and should not be considered cause for immediate concern. The fact that the sender on such occasions could not connect at all raises the question of whether certain critical functions in everyday life should rely on Bluetooth for their operation. Volvo's safety concept car [26] is based around a Bluetooth-enabled personal communicator used to remotely interact with the car. If interference can render such a device unreachable it then begs the question of whether Bluetooth denial of service attacks will be seen in the future.

In order to address these concerns the following tests should be pursued:

* Perform frequency analysis tests on common Wi-Fi equipment to determine their actual frequency use.

* Perform tests in a saturated WLAN environment using all three non overlapping channels to see if Bluetooth remains functional.

* Perform tests in using other stacks to investigate the abnormal behaviour observed. These test would tell whether the results were affected by the stack used. Further tests should also be performed using alternative Bluetooth devices to determine if other hardware behaves similarly.

5.9 Summary

This chapter presented a study of how interference from a Wi-Fi station affects a fast Bluetooth link. It first described a theoretical model developed to determine throughput loss. Next it described the architecture and implementation of a C client-server application that measures throughput on a Bluetooth link at the L2CAP layer. Lastly it presented the results obtained when measuring throughput while using a Wi-Fi station to generate harmful interference. The results were found to be substantially worse than predicted by theory and might be attributed to Wi-Fi equipment generating substantial out-of-band noise. The practical implication is that Bluetooth may not function very well or at all near Wi-Fi equipment, and several further tests were proposed to pursue the topic.

6. BLUETOOTH USAGE MODELS AND PRODUCTS

6.1 Bluetooth Usage Models

In this section a number of Bluetooth usage models are described. For each usage model there is one or more corresponding profiles defining protocol layers and functions to be used. The profiles are not described in detail in this document, for more information refer to the Bluetooth standardization documents.

6.1.1 File Transfer

The File Transfer usage model offers the capability to transfer data objects from one Bluetooth device to another. Files, entire folders, directories and streaming media formats are supported in this usage model. The model also offers the possibility of browsing the contents of the folders on a remote device. Furthermore, push and exchange operations are covered in this usage model, e.g. business card exchange using the vCard (Electronic Business Card) format. The File Transfer model is based on GOEP.

6.1.2 Internet Bridge

The Internet Bridge usage model describes how a mobile phone or cordless modem provides a PC with dial-up networking capabilities without the need for physical connection to the PC. This networking scenario requires a two-piece protocol stack, one for AT-commands to control the mobile phone and another stack to transfer payload data.

6.1.3 LAN Access

The LAN Access usage model is similar to the Internet Bridge user model. The difference is that the LAN Access usage model does not use the protocols for AT commands. The usage model describes how data terminals use a LAN access point as a wireless connection to a Local Area Network. When connected, the data terminals operate as if it they were connected to the LAN via dial-up networking.

6.1.4 Synchronization

The synchronizations usage model provides the means for automatic synchronization between for instance a desktop PC, a portable PC, a mobile phone and a notebook. The synchronization requires business card, calendar and task information to be transferred and processed by computers, cellular phones and PDAs utilizing a common protocol and format.

6.1.5 Three-in-One Phone

The Three-in-One Phone usage model describes how a telephone handset may connect to three different service providers. The telephone may act as a cordless telephone connecting to the public switched telephone network at home, charged at a fixed line charge. This scenario includes making calls via a voice base station, and making direct calls between two terminals via the base station. The telephone can also connect directly to other telephones acting as a "walkie-talkie" or handset extension i.e. no charging needed. Finally, the telephone may act as a cellular telephone connecting to the cellular infrastructure. The cordless and intercom scenarios use the same protocol stack.

6.1.6 Ultimate Headset

The Ultimate Headset usage model defines how a Bluetooth equipped wireless headset can be connected, to act as a remote unit's audio input and output interface. The unit is probably a mobile phone or a PC for audio input and output. As for the Internet Bridge user model, this model requires a two-piece protocol stack; one for AT-commands to control the mobile phone and another stack to transfer payload data, i.e. speech. The AT-commands control the telephone regarding for instance answering and terminating calls.

6.2 Early Products and Prototypes

6.2.1 Plug-in modules

Initial products consist of plug-in modules to allow users to Bluetooth enable existing devices. These are basic cable replacement devices that interface through existing ports, and include the following:

Figure 6.1 Plug-in modules

- PCMCIA card
- USB dongle
- Memory stick
- Serial port dongle
- Parallel port dongle
- Springboard for Handspring Visor
- LAN access points
- Cellular phone dongle

As chip prices drop and manufacturers begin to integrate Bluetooth chips into motherboards and other devices, integrated solutions will become more widely available.

6.2.2 Digital image messaging

Pictured are a Bluetooth-enabled Nokia 9110 and a Fuji Film digital camera that can communicate with one another using Multimedia Messaging Services (MMS). In this example, Bluetooth is enabling digital image messaging. A user takes a digital picture, transfers the image via Bluetooth to the Nokia 9110, adds a few lines of text, and then mails it to another 9110, a PC, or to FugiFilm.net for prints and saving to CD-R.

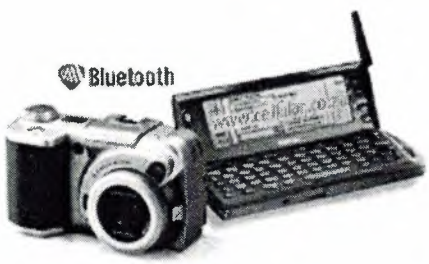

Figure 6.2 Digital image messagers

6.2.3 Bluetooth Infowear

In what the Bluetooth community calls "unconscious" or "hidden" computing, Bluetooth-enabled products will automatically seek each other out and configure themselves into networks – most often, with just two nodes. Though small, such networks can be quite useful. In this example, a prototype wristwatch that acts as an organizer and synchronizes information wirelessly with a PC is shown. At the Bluetooth Developers Conference in December 2000, IBM demonstrated a working prototype of a Linux-based wristwatch, complete with VGA touch-screen, speaker, microphone, and Bluetooth radio. During a keynote presentation, the presenter used this watch to control his PowerPoint presentation, while 3000 people looked on in amazement.

Figure 6.3 Bluetooth Info wear

6.2.4 Bluetooth Pen

With the Anoto Bluetooth pen, e-mails, faxes and e-commerce orders can be sent electronically by simply putting pen to paper. The technology was developed by Ericsson, Anoto and Time Manager, and is scheduled for introduction in the second half of 2001. The device looks, feels, handles and writes like an ordinary ink pen, albeit a bulky one with a little LED indicator on the side. In addition to the usual ink cartridge, the Anoto pen contains image processing and Bluetooth radio circuitry designed to automatically transmit what is written to a Bluetooth enabled cellular phone, handheld computer or network base station. A pressure sensor at the back end of the ink cartridge senses when the pen is actually writing, and a small imaging sensor under the ink cartridge tracks the motion of the pen on the paper. The system requires special paper with a pattern printed on it, too small to be seen with the naked eye, to allow the pen's image processor to track the movements of the pen.

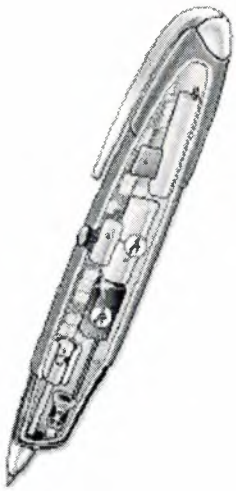

Figure 6.4 Bluetooth Pen

6.2.5 Xyloc

Ensure Technologies patented Xyloc technology allows a user to wear a key in the form of an ID badge-sized KeyCard or a small, pager-sized KeyFob. A Lock attaches to the user's PC through the keyboard, USB or serial port. The Lock and Key use an encrypted two-way Bluetooth radio link to identify the user to the Xyloc software on the computer. When a user approaches a Xyloc secured computer, the Key transmits a unique encrypted code to the Lock, which relays the information to a security database for validation. If the user is authorized, the system unlocks the keyboard and screen; if unauthorized the system remains secure. When the user steps away from the computer Xyloc immediately and automatically secures the computer. At CeBIT 2001, Seiko Instruments Inc. and Ensure Technologies demonstrated Xyloc technology incorporated into an interactive Bluetooth wristwatch [47]. This technology is one to follow, as it could have wide application for CSC and its clients, both commercial and government.

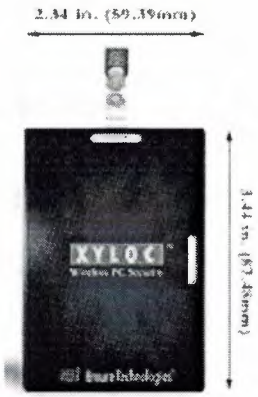

Figure 6.5 KeyCard

Figure 6.6 Key Fob

6.2.6 Convergence Products

The Ericsson Communicator Platform is an example of some of the capabilities that the next generation of products will offer, combining features of the hottest technologies into one device. This prototype device combines mobile Internet browsing, messaging, imaging, location based applications and services, mobile telephony and personal information management. This kind of product convergence will truly make life more pleasant by eliminating the need to carry multiple devices. And, of course, Bluetooth will mean that proprietary cables will no longer be needed in order to connect to other devices.

Figure 6.7 Bluetooth based prototype device

7. ADVANTAGES OF BLUETOOTH

7.1 Bluetooth

7.1.1 Introduction

Bluetooth, initiated by Ericsson in 1994 and V1.0 published by Special Interest Group (SIG) in 1999, is a RF-based communication technology for connections of devices in near proximity. Originally, the purpose of Bluetooth is to serve low-power wireless handheld devices (e.g. mobile phones, PDAs, laptops), but can also be used to serve any wired devices (e.g. desktops, printers)
Bluetooth has the followings features:

- Low-cost. Bluetooth uses 2.4GHz-ISM-Band (Industrial, Scientific, Medical) for communication. The ISM radio bands are reserved globally for the non-commercial and unlicensed use of RF for industrial, scientific and medical purpose [30]. Thus, there is no fee for channel rent as well as for cable consumption in Bluetooth communication.

- Low-power consumption. Bluetooth applies a power control scheme to reduce the power consumption of devices. The mechanism of the power control scheme is to adjust the RF output power to the minimum level which is mandatory to maintain the communication link. Furthermore, in case of receiving a strong signal, the power control scheme also enables the receiver request a less RF output power from the transmitter.

- Short-range. Normally, the Bluetooth range for two-device communication is 10 meters without caring about obstacle (e.g. wall, desk, people) in between. Potentially, this distance could be extended to more than100 meters by increasing transmitter power.

- No-cable connection. Bluetooth is a RF-based communication technology, which significant reduces the complexity of what cable setting up brings. Engineers won't be headache for consideration of cable length any longer.

- Little configuration. A business man with a laptop, for example, comes into the conference room of his customer company and can be immediately connected to the Bluetooth-enabled printer without further configure such as device driver installation. Except some security information is required at the beginning of connection establishment.

- Piconet. Two or up-to-8 active Bluetooth units form a piconet. There are some rules need to obey to create a piconet:

1) Each piconet only has a single "Master", and the rest (up-to-7) units are active "Slaves".

2) More than 7 slaves are allowed but in "Park" mode. The Master can unpark slaves to make them active, and park some as well.

3) There is no dedicated Master. Any Bluetooth unit could a Master.

4) The roles of Master and Slave can be interchanged.

5) Each Slave can participate in different piconets.

6) A Master in one piconet can be a Slave in other piconet at the same time.

7) A Bluetooth unit can NOT be the Master of two piconets.

- Scatternet. Two or more piconets with overlapping coverage areas form a scatternet.
- Security Connection. Bluetooth defines three levels of security:

1) Secure Mode 1 – Non-secure. No security procedure is applied. Canlendar (vCanlendar), for example, could be exchanged by this mode.

2) Secure Mode 2 – Service-level Security. Security scheme (e.g. encrypting transmitted data) is initiated AFTER establishing a connection.

3) Secure Mode 3 – Link-level Security. Security scheme (e.g. authenticating the device) is initiated BEFORE establishing a connection.

7.1.2 Protocol Stack

The complete Bluetooth protocol family covers the whole OSI 7-layer excluding the presentation layer (while UPnP do NOT cover the lowest two layers). The Bluetooth protocol family can be divided into four parts:

- Core Protocols including:
SDP / L2CAP / LMP / BaseBand & Bluetooth Radio

- Cable Replacement Protocol including:
RFCOMM

- Telephony Control Protocols including:
AT-Commands and TCS BIN

- Adopted Protocols including:
vCard or vCal / OBEX
WAE / WAP / UDP or TCP / IP / PPP

But depending on capabilities of Bluetooth application, only necessary protocols, instead of the whole protocol stack, are required to be involved. For instance, the file transfer application only needs to comprise the following protocols:

7.1.3 Networking

Networking in Bluetooth is simple follows three steps:
- Master sends paging requests to detect potential Slaves nearby;
- Bluetooth units around respond the paging request if they want to participate in the piconet;
- Master opens connections for those units who respond.

8. BLUETOOTH EFFECTS ON HUMAN HEALTH

It is a matter of concern for some people that the carrier waves used by Bluetooth´s transmitters use the same frequency range as microwave owens (Bluetooth uses 2.402 GHz to 2.480 GHz). Someone may wonder about how it feels like to get in the path of such waves.

Actually, the transmitting power is far too weak to be noticeable for humans. Moreover, the radiation is not concentrated in a beam, but dispersed more or less in all directions. When using a wireless phone or a Bluetooth device, the body absorbs some of the emitted RF energy. The penetration depth is about 1.5 cm at 2450 MHz (about 2.5 cm at 900 MHz), which means that the absorption is very superficial. The main absorption mechanism is field-induced rotation of polar molecules (for example H_2O), which generates heat through molecular "friction".

Heating by means of radio frequencies is possible over a broad frequency range. This is taken advantage of in microwave ovens at 2450 MHz using very high power levels (up to 1,000,000 times the power used by Bluetooth devices). However, 2450 MHz is not a resonance frequency of water. But whether the Bluetooth RF exposures to emission heat the human body? No, it does not. The output power of a Bluetooth-enabled device is far too low to cause any detectable temperature increase. Again, in comparison, the maximum increase from handheld cellular phones is less than 0.1°C.

There is, however, another side to this; some people are demonstrably over-sensitive to electromagnetic radiations. Long exposure to strong fields makes some individuals so sensitive, after a few years that they can no longer be near such fields without considerable discomfort. Bluetooth fits into a general development pattern where antennas for GSM-transmission and other sources of electromagnetic radiations become more and more prevalent in our cities. The future will show whether this is a healthy development.

CONCLUSION

this project I present a positioning system for Bluetooth enabled devices. Bluetooth is an expensive and small size solution and will probably be integrated in a range of different evices.

nyway i try to show you the effects of bluetooth in human life and how to work bluetooth nd why we use bluetooth in our life.

'e know that bluetooth has an importance in technology now.

luetooth wireless technology encompasses several key points that facilitate its widespread doption:

) it is an open specification that is publicly available and royalty free;

) its short-range wireless capability allows peripheral devices to communicate over a single ir-interface, replacing cables that use connectors with a multitude of shapes, sizes and umbers of pins;

) Bluetooth supports both voice and data, making it an ideal technology to enable many ypes of devices to communicate;

) Bluetooth uses an unregulated frequency band available anywhere in the world.

'o fully realize the Bluetooth vision, full networking of multiple Bluetooth devices is equired. This leads to the investigation of Bluetooth scatternets, which must address catternet formation and reconfiguration, scheduling, and routing issues.

With a positioning system based on Bluetooth we are able to make these devices location ware. This will open up for new applications and services that can be developed on these levices. We have showed that the current implementation is fast enough for the task at least or one device. We have also showed that future changes will probably make it even faster nd by that make it suitable for more devices and in that higher accuracy. With the use of our ocation server, even the devices that cannot run any third party software can be used as osition sources.

REFERENCES

1. Bluetooth Special Interest Group. *Specification of the Bluetooth System v1.1*, volume 1, available from http://www.bluetooth.com, December 2003.

2. Jaap Haarsten, Mahmoud Naghshineh, Jon Inouye, Olaf Joeressen and Warren Allen, Bluetooth: Vision,Goals and Architecture, Mobile Computing and Communications Review. ACM SIGMOBILE, Vol.2, No. 4, Oct. 1998

3. Miller, Brent A. Bisdikian, Chatschik. *Bluetooth Revealed*, Printice Hall, Inc. 2001

4. J. Proakis, and M. Salehi, Communication System Engineering, Prentice Hall, 2002.

5. C. Bisdikian, S. Bouet, J. Inouye, R. Mettälä, B. Miller, K. Morley, et al. (1999, August 25). "Bluetooth Protocol Architecture - Version 1.0." Bluetooth White Paper - Bluetooth Special Interest Group. [Online}. Available: https://www. bluetooth.org/ foundry/sitecontent/document/ Protocol_Architecture

6. Ling-J. Chen, Rohit Kapoor, M. Y. Sanadidi, Mario Gerla, .Enhancing Bluetooth TCP Throughput via Link Layer Packet Adaptation,. The 2004 IEEE International Conference on Communications (ICC 2004), Paris, France, 2004.

7. Palo wireless Bluetooth Resource Center, tutorials, available from, http://www.palowireless.com/infotooth/tutorial.asp, 2002

8. Bluetooth Special Interest Group. Specifications of the Bluetooth Systems, Profiles volume 2, available from, http://www.bluetooth.com, 22 February 2001