Real Time Multiple Cross Platform Communication through Bluetooth

Ms Prerna Hingle, Ms Shubhangi Giripunje

Abstract— Bluetooth provides low power and low cost connections between two Bluetooth devices. Bluetooth has high level of compatibility because of the standardized Bluetooth protocol stack that is available. To establish a wireless connection between two or more devices, the platform on which the application is working opens a socket. This paper introduces a concept to establish multiple connections for real time communications through Bluetooth, independent of the platform.

Index Terms— Bluetooth; RFCOMM; TCP; DOTNET; JAVA; server; client.

1 Introduction

Wireless Technology in every sense has improved our life to a great extent. The most commonly used wireless technology is Bluetooth Technology. Bluetooth uses short wavelength Ultra High Frequency (UHF) radio waves to transfer data form one point to another. Bluetooth operates in unrestricted and unlicensed Industrial, Science and Medical (ISM) band at 2.4GHz. Bluetooth Enabled devices use frequency hopping technique that randomly hops between frequencies unto 1600 hops/sec. Range of Bluetooth depends on the classes of radios used in the implementation. Class 1 radios have range of 100 meters or 300 feet, Class 2 radios have range of 10 meters or 33 feet and class 3 radios have range of 1 meter or 3 feet.

Bluetooth technology is used for number of purposes for example file transferring form one Bluetooth enabled device to another. Recently Bluetooth chatting between two androids has been introduced. This paper introduces a concept for cross domain communication using Bluetooth where different devices that work on different domains will be able to communicate with each other in real time. The two domains implemented here are JAVA and DOTNET. The main intention of this paper is to create path for multiple connections through Bluetooth, where many devices that support Bluetooth will be able to communicate with each other in real time.

2 Review of Literature

In Android real time communication [6] Server and a client are formed. The server keeps a socket open to listen to incoming connections. A thread is formed between a server and a client and through this thread communication takes place. The first step is to find and discover available Bluetooth devices. This is done by opening the socket. After that the devices in range are paired. After proper pairing is performed the connections are established for the communication to take place between them. Here only two androids can communicate with each other that is for this communication to be possible the devices must be working on the same platform.

In a technique for cross platform communication, to support cross platform communication a set of classes in JAVA and J2ME have been implemented [3]. Here the framework provides the implementation of the Bluetooth classes for communication between the two platforms. That is communication between Android to Android, J2ME to J2ME and Android to J2ME. The framework supports the Radio Frequency Protocol (RFCOMM) communication protocol for the Bluetooth communication.

To overcome the limitations of the RFCOMM layer An Energy-aware Multipath TCP Scheme is introduced [4]. Here the problem of using multipath TCP in battery powered mobile devices is discussed. The architecture involves the two components deployed at the upper transport layer: The first is Sub-flow Interface State Detector (SISD) which continuously observes the channel state of the WiFi and LTE interfaces and the second is Offload Controller which prepares for offloading some traffic from the LTE sub-flow to the WiFi sub-flow. The offload amount is decided according to the current congestion window size of the LTE sub-flow and the feedback of interface channel state change from SISD. The simulation done proves how energy saving is achieved and what performance eMTCP obtains in comparison with single-path solutions and MPTCPrecreated, but rather converted into the final published version.

3 Research Design

The goal of the research is to make possible Real time offline communication of multiple devices of different platforms through Bluetooth. For this the main area will be Transport Layer and Multiclient Asynchronous TCP server will be formed.

The objectives will be achieved in the following manner through Bluetooth.

1) Cross domain Bluetooth communication will be made possible.
2) A server will be able broadcast real time messages to multiple clients irrespective of the domain.
3) A server will be able to communicate with individual client in real time.

4 Bluetooth Architecture In Different Platforms

4.1 Bluetooth Architecture in Microsoft

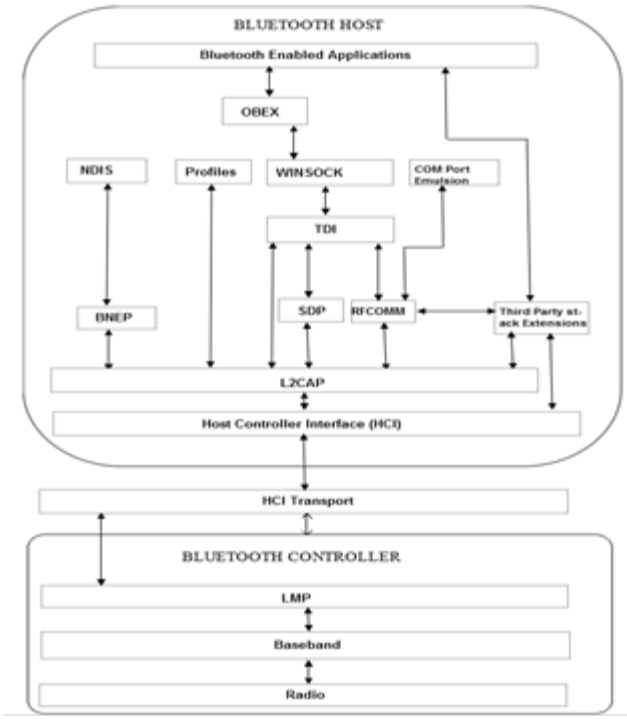

Fig 1: Microsoft Bluetooth Protocol Stack

The Bluetooth architecture in Windows Microsoft is divided into three layers, Bluetooth Host, Host Controller Interface (HCI) Transport layer and Bluetooth Controller. Bluetooth Host includes implementation of core Bluetooth protocols such as Bluetooth stack and high-level Layers of Bluetooth Architecture. HCI Transport layer delivers data between Bluetooth Host and Bluetooth Controller.
Bluetooth Controller is a device that implements the lowest levels of Bluetooth Architecture.
Fig 1 shows Microsoft Bluetooth Architecture.

In Microsoft Bluetooth Protocol Stack the primary way in which an Application interacts through Bluetooth is by using winsock interface that exposes the RFCOMM layer to the application.

A Bluetooth-enabled application must then interface with certain profiles to support specific types of Bluetooth activity. Each serial port connections can connect two Bluetooth enabled devices. Winsock that is windows socket is an Application Programming Interface API that defines how software accesses networking services such as TCP/IP. It facilitates communication between networking applications, the TCP/IP, and Bluetooth protocol stacks.

4.2 Bluetooth Architecture in Android

A default Bluetooth Stack is provided by the android.

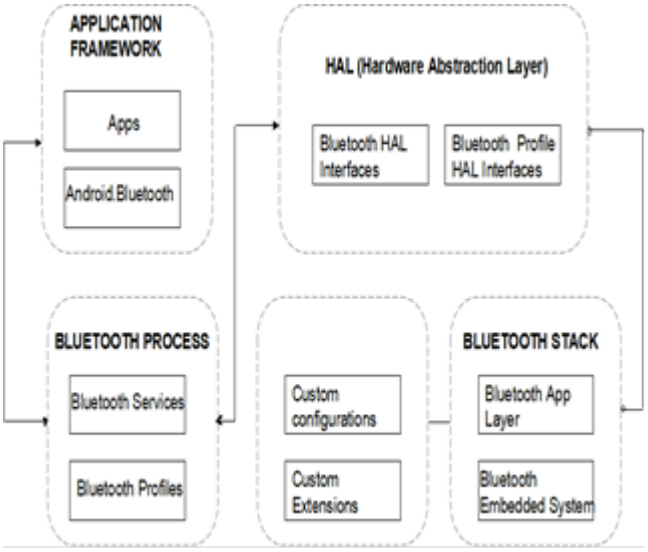

Fig 2: Bluetooth Architecture in Android

This Bluetooth stack in android is divided into two layers.

The first is the Bluetooth Embedded System (BTE) that implements the core Bluetooth functionality and the second is Bluetooth Application Layer (BTA) that communicates with the Android Framework. Figure 2 shows the Bluetooth Architecture in Android.

The application framework is the layer where the applications are and contains the application's code. This is connected to the Bluetooth Process. This application code calls the Bluetooth process through the binder mechanism. A Binder is an inter process communication mechanism in Android. This interacts with the Bluetooth Hardware. The JNI code is called in between the Bluetooth services and the Hardware abstraction layer (HAL). Java Native interface (JNI) is a programming framework that allows Java code running in Java Virtual machine to call and be called by native applications. The hardware abstraction layer is the standard interface that the Bluetooth process calls. This HAL layer is connected to the main Bluetooth Stack.

4.3 Communication in Bluetooth

Piconet is the basic unit of networking in Bluetooth. It consists of one master and upto seven slaves. The determination of channel and phase that will be used by all the devices in a piconet is determined by the radio designated as the master. All communication in a piconet passes through the master node in a piconet. As an example in a piconet a computer may be the master node and Bluetooth USB dongle, Bluetooth keyboard, Bluetooth Mouse and Bluetooth enabled cellphones may be the slaves in that piconet. The master that is the computer will be able to communicate will all the slaves but the slaves will not be able to communicate with each other. [8], [9]

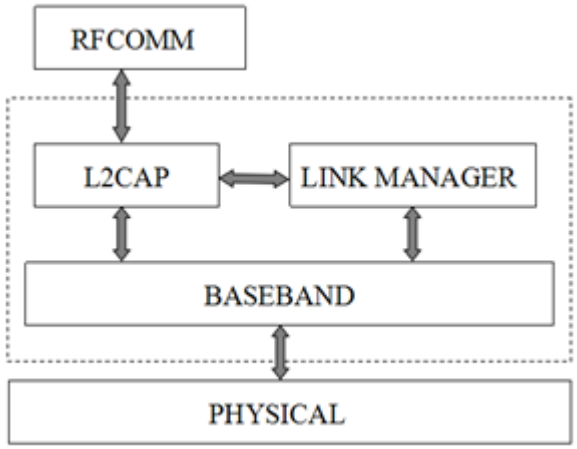

Fig 3: Protocol stack with RFCOMM

Scatternet facilitates communication between piconets. A scatternet is formed when a device in one piconet also exists as a part of another piconet and causes overlapping.

In this way a number of devices are able to share the same physical area and bandwidth is used efficiently. This is at the Controller layer.

The RFCOMM that is Radio Frequency Communication in Bluetooth Host and replaces RS-232 serial ports over the L2CAP layer. That is when applications on two Bluetooth devices wish to communicate, the two radios form a link and then data can be transferred over the connection using the L2CAP protocol. The primary way for an application to use Bluetooth technology is through the Winsock interface, which exposes the RFCOMM layer to the application. The functionalities of RFCOMM ports include, it implements multiplexing, flow control and it acts as a carrier for attention (AT) command.

Fig 3 tell how the RFCOMM works in Bluetooth

The physical layer is in the Bluetooth Controller and its basic functionality is to receive bit stream from the Medium Access Control (MAC) sub layer. The baseband is concerned with the connection establishment. The link manager is responsible for link setup between Bluetooth devices. The L2CAP layer provides protocol multiplexing and adapts upper layer protocols to the baseband layer. And this is connected to the RFCOMM layer.

Another transport layer protocol that is used is the TCP (Transmission Control Protocol). TCP is advantageous because it is a double way communication protocol, synchronous and connection oriented. TCP is reliable transport protocol. The advantage of TCP is that it is platform independent. A socket is an interface that is used to send and receive data from TCP/IP stack provided by the Operating System. As soon as communication is established with the client a dedicated socket object is created with the client.

The choice of port numbers is the biggest difference between the RFCOMM and the TCP. RFCOMM only allows 30 ports whereas TCP supports up to 65535 open ports on a single machine.

4.4 Core Architectural Blocks

1. Channel Manager: L2CAP layer contains channel manager. It is responsible for creating, managing and closing L2CAP channels for transport of protocols. L2CAP protocol is used by channel manager for peer to peer communication between two devices.
2. Device manager: The functional block in the baseband is the device manager that controls the behavior of Bluetooth device. The device manager requests access to the transport medium to carry out its functions.
3. Link manager: The link manager achieves communication between remote Bluetooth devices using Link Manager Protocol. It is responsible for creating and modification of logical links and physical links between the devices.
4. Baseband resource manager: It's main function is to access the radio medium. It acts as a scheduler and provides expected performances.
5. Link controller: It encodes and decodes Bluetooth packets.

5 Interaction Bluetooth Bluetooth Devices

Bluetooth Communication is based on unique MAC. For the security before any Bluetooth communication, the devices must be paired. In order to create a connection between the two devices, both the server-side and client-side mechanisms must be implemented, because one device must open a server socket and the other one must initiate the connection (using the server device's MAC address to initiate a connection). The server and client are considered connected to each other when they each have a connected BluetoothSocket on the same channel. At this point, each device can obtain input and output streams and data transfer can begin.

5.1 Server

Server is a base class that can adapt to accept network connections over Bluetooth and TCP. A server socket that is formed is also known as listening socket. To get a useful data connection out of a server socket there are three steps an application must take.

1) It must bind the socket to local Bluetooth resources, specifying which Bluetooth adapter and which port number to use.

2) The socket must be placed into listening mode. This indicates to the operating system that it should listen for connection requests on the adapter and port number chosen during the bind step.

3) The application uses the bound and listening socket to accept incoming connections.

When two devices are to be connected, one must act as a server by holding an open *BluetoothServerSocket*. The purpose of the server socket is to listen for incoming connection

requests and when one is accepted, provide a connected *BluetoothSocket*. When the *BluetoothSocket* is acquired from the *BluetoothServerSocket*, the *BluetoothServerSocket* can be discarded, unless more connections are to be accepted.

5.2 Client

In order to initiate a connection with a remote device (a device holding an open server socket), *BluetoothDevice* object that represents the remote device must be obtained. Then use the *BluetoothDevice* to acquire a *BluetoothSocket* and initiate the connection.

Once the socket has been created by the server, the client program only needs to issue the connect command, specifying which device to connect to, and on which port. The operating system then takes care of all the lower level details, reserving resources on the local Bluetooth adapter, searching for the remote device, forming a piconet, and establishing a connection. Once the socket is connected, it can be used for data transfer.

6 Interaction Between The Two Platfors

6.1 Initializing Bluetooth

First the Bluetooth radio is initialized and it is checked whether the Bluetooth is supported or not. Then the Bluetooth radio's Name, Manufacturer is checked.

A basic Windows application runs on a single thread usually referred to as UI thread. This UI thread is responsible for creating all the controls and upon which the code execution takes place. So when you are running a long-running task, the UI thread locks up and your application is no longer responsive. To make the UI thread free, .NET has made the Background Worker object available to us to simplify threading. This object is designed to simply run a function on a different thread and then call an event on your UI thread when it's complete.

6.2 Link Establishment

1) The link is established through Link Manager Protocol. To support different application data transport requirements a variety of logical links are available.
Logical links that can be are,

i. Asynchronous connection oriented (ACL) logical transport that is reliable, bi-directional and point to point.
ii. Synchronous connection oriented (SCO) that is bi-directional, symmetric and point to point.
iii. Extended Synchronous connection oriented (eSCO) that is bi-directional, symmetric or asymmetric and point to point.

2) The logical link control and adaption protocol (L2CAP) provides multiplexing, allowing different applications to share logical link. L2CAP provides per channel flow control with

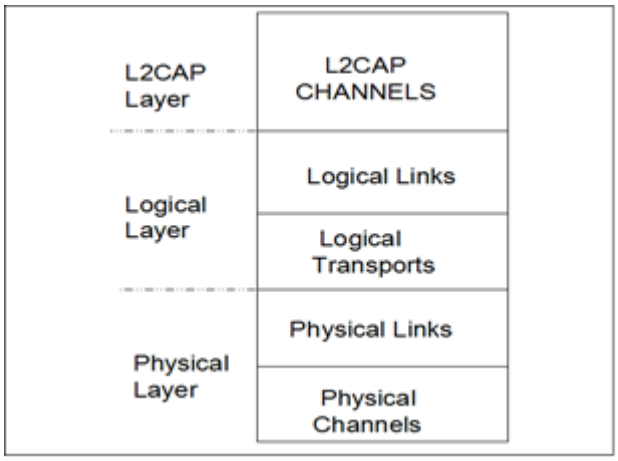

Fig 4: Bluetooth data transport architecture

Peer L2CAP layer.
Figure 4 shows the hierarchy of Logical layer and L2CAP layer.

3) This then directly communicates with the RFCOMM layer in the Bluetooth layer of the server which then communicates with the TCP layer.

The problem with using only RFCOMM is that the ports available here are less as compared to the TCP and RFCOMM takes into account only one port number at a time whereas TCP takes into account two port numbers at a time, that is port number of server as well as that of client. So the TCP is used in both the platforms along with RFCOMM. Another reason for using TCP is that TCP is platform independent.

In every operating system there is a TCP/IP stack. To send data and to receive data from this stack socket interface is used. So the main challenge is to use this TCP in Bluetooth.

4) Then at the client side this again this data is received through the TCP layer.

6.3 Pairing And Establising Connections

For establishing connections a server and a client are formed. "Server" devices are usually characterized by having data that is needed by some other device. A "client" may be a device that needs information from a server. Server is a base class that can adapt to accept network connections over Bluetooth and TCP.

Bluetooth applications/services are identified and registered by UUID (Universally Unique Id), a 128-bit value that is represented by the System.Guid class in .NET. If one is creating a new service then a new UUID should be created at design time and entered into the two applications' source code.

Bluetooth in Android works on JAVA platform by using JAVA Native Interface (JNI) and Bluetooth in Laptops use Windows protocol stack which works on Dot Net.

In C Sharp there are some classes that makes the socket open for incoming connections. Here these classes are used in windows Bluetooth Protocol Stack. Hence RFCOMM layer will open sockets for all the incoming connections and

will form a thread.

At the android side the Bluetooth Stack which consists of Bluetooth Embedded System (BTE) and the Bluetooth Application Layer (BTA), communicates with Android Framework Application.

This Bluetooth Stack is connected to the Hardware Abstraction Layer (HAL). This is connected to the Application Framework which utilizes Application Interfaces (APIs) to interact with Bluetooth hardware. The Application framework calls the Bluetooth process.

This Application Framework communicates with the C sharp application of the windows Bluetooth Protocol Stack. That is this will connect to the socket that was kept open for incoming connections. In windows Bluetooth Protocol stack there is a layer called WINSOCK layer that is Windows Socket. This application Framework will communicate directly with this layer. Windows Socket is an Application Interface which defines how networking services such as TCP/IP are accessed by software. It basically facilitates communication between Bluetooth protocol stack and there networking services.

Once the connection is established between the Application Framework and the WINSOCK layer then the cross domain communication will be possible and an Android device and a Windows OS Laptop will be able to communicate with each other in Real Time.

7 Result

The real time communication between two platforms was successfully performed through Bluetooth. A server and a client are formed where the server is in Dot-Net platform and the clients are in the JAVA platform and Dot Net Platform.

References

[1] Changlin Jiang, Dan Li, Member, IEEE, and Mingwei Xu, Member, IEEE " LTTP: An LT-Code Based Transport Protocol for Many-to-One Communication in Data Centers " IEEE Journal on Selected areas in communication VOL. 32, NO. 1, JANUARY 2014.

[2] Jason Cloud, Flavio du Pin Calmon, Weifei Zeng, Giovanni Pau, Linda M. Zeger, and Muriel Médard , "Multi-Path TCP with Network Coding for Mobile Devices in Heterogeneous Networks", 2013 IEEE.

[3] Sundeepsingh Neerunjun, Chervine Bhiwoo, and Leckraj Nagowah, " A Framework for Android and J2ME Bluetooth Communication ", International Journal of Computer Applications (0975 – 8887) Volume 57– No.22, November 2012.

[4] Shengyang Chen, Zhenhui Yuan and Gabriel-Miro Muntean "An Energy-aware Multipath-TCP-based Content Delivery Scheme in Heterogeneous Wireless Networks" , 2013

[5] Anesh R, Sreekumari & Jiju, "Design and Implementation of Bluetooth MAC core with RFCOMM on FPGA ", 2012 IEEE.

[6] Weihua Pan, Fucai Luo, Lei Xu, " Research and Design of Chatting Room System based on Android Bluetooth", 2012 IEEE.

[7] Bo Fu, Yang Xiao, Hongmei (Julia) Deng, and Hui Zeng " A Survey of Cross-Layer Designs in Wireless Networks ", IEEE Communications Surveys and Tutorials , Vol. 16,No. 1, First Quarter 2014.

[8] Yang Wang, Jianhua Lu and Ning Ge, " Priority-based Adaptive Frequency Hopping for Blueooth in Multi-piconet Environments" , 2013 IEEE.

[9] Wang Feng, A. Nallanathan and H. K. Garg, " Performance of PHY and MAC Layers of a Bluetooth Piconet in Multi-Bluetooth Interference Environment" 2004 IEEE.

[10] Roy Friedman, Alex Kogan, IEEE, and Yevgeny Krivolapov, " On Power and Throughput Tradeoffs of WiFi and Bluetooth in Smartphones", IEEE Transactions on mobile computing. Vol. 12, No. 7, July 2013.