Android Communication

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Information Technology

By

Piyush Garg (121420)

Under the supervision of

Mr. Suman Saha

To

Department of Computer Science & Engineering and

Information Technology

Jaypee University of Information Technology Waknaghat,

Solan-173234, Himachal Pradesh

Certificate

I hereby declare that the work presented in this report entitled **"Android Communication"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Information Technology** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of our own work carried out over a period from August 2015 to May 2016 under the supervision of **Mr. Suman Saha** (Assistant Professor, Computer Science & Engineering).

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Date:

Prof. Suman Saha
Assistant Professor

:

Candidate's Declaration

I hereby declare that the work presented in this report entitled **"Android Communication"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Information Technology** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat is an authentic record of our own work carried out over a period from August 2015 to May 2016 under the supervision of **Mr. Suman Saha** (Assistant Professor, Computer Science & Engineering).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Piyush Garg (121420)

Acknowledgement

I place on record and warmly acknowledge the continuous encouragement, invaluable supervision, timely suggestions and inspired guidance offered by my guide Prof. Suman Saha, Assistant Professor, Dept. of Information Technology, Jaypee University of Information Technology in bringing this report to successful completion.

I would also like to express my gratitude to Brig. (Retd) S.P.Ghrera Head of Department, Computer Science and Information Technology, JUIT Waknaghat for guiding, encouraging and inspiring by giving us valuable thoughts to carry out the project. I will also thank my friends and colleague, who all helped me out during this project.

Finally, I extend my gratitude to one and all who are directly or indirectly involved in the successful completion of this project report.

Date: **Piyush Garg**

Table of Contents

List of Figures

List of Tables

Abstract

In this 21st century mobile phones are smart and capable enough to work as a small portable computer and perform the functionalities pretty efficiently. With the rapid development of mobile Internet technology and the widespread use of smart phones, the Smartphones are nowadays equipped with diverse collection of I/O devices like Camera, Sensors, Modems, Speakers, GPS and Microphone etc. There are various benefits of communicating the devices such as playing multi-user games, chatting, sharing files or multimedia content, seamless audio playing etc. Therefore, more and more attentions have paid on network access and interactive applications through mobile phones. Android becomes the most favorite smart phone by the people and programmers because of its open-source, easy to use and biggest global market share and has broad prospects for development. With the help of this project, I have used Bluetooth Technology to create a wireless touchpad to control the computer wirelessly and using Wi-Fi technology, I have tried to sync various connected devices via various equipped I/O devices and compare their performance by making an application to play music seamlessly.

There has been an increasing interest of a wide number of scholars and researchers in promoting the applications for Peer-to-Peer (P2P) communication in Android based smartphones as ubiquitous. Such increasing interest has been attributed to their increasing perception of these devices. There is a wide use of many P2P applications in mobile devices. These applications include IM, VoIP, file sharing, social networks, and video streaming. Thus, with this project I have tried to study the various methodologies to establish peer to peer (P2P) by using Wi-Fi, Bluetooth or Internet and finding the most efficient of them. In this project we have reviewed the performance of the local I/O to the network throughput limitations between various devices which are likely to palliate with emerging technology. Furthermore, the project moves on to providing the mobile devices limitations and the challenges encountered in the adoption of the P2P communication technology in the mobile environment.

Chapter 1: Introduction

In this chapter we will be looking into the basic architecture of Android and functioning of various possible Peer to Peer (P2P) connections such as Wi-Fi, Bluetooth, NFC, Internet and testing their real time performance in term of bandwidth, battery consumption and performance measures etc.

1.1 What is Android?

Android is the mobile operating system owned by Google; an American Company. It is one of today’s most popular terms in the world of mobile operation systems. Android phones and Android applications are so popular among people that they have established a secured position among users in the world of mobile phones. Android is the software stack for a mobile device that includes an operating system, middleware and key applications. It is a Linux based operating system. It offers the users access to Google’s own services like Search, YouTube, Maps, Gmail and more.

This means one can easily look for information on the web, watch videos, search for directions and write emails on your phone, just as you would on your computer, but there’s more to Android than these simple examples.

In the market many programming languages are available such as C, C++, PHP, Flash and Java. The reason for Android to choose Java over other programming languages is because Java is an object oriented programming language, simple, secure, dynamic, portable, has high performance, supports multithreading and has automatic garbage collection. Besides Java C/C++, Flash can also be used to create Android applications

1.2 History of Android

The code names of android ranges from A to L currently, such as Aestro, Blender, Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwitch, Jelly Bean, KitKat, Lollipop and Marshmallow. Let's understand the android history in a sequence

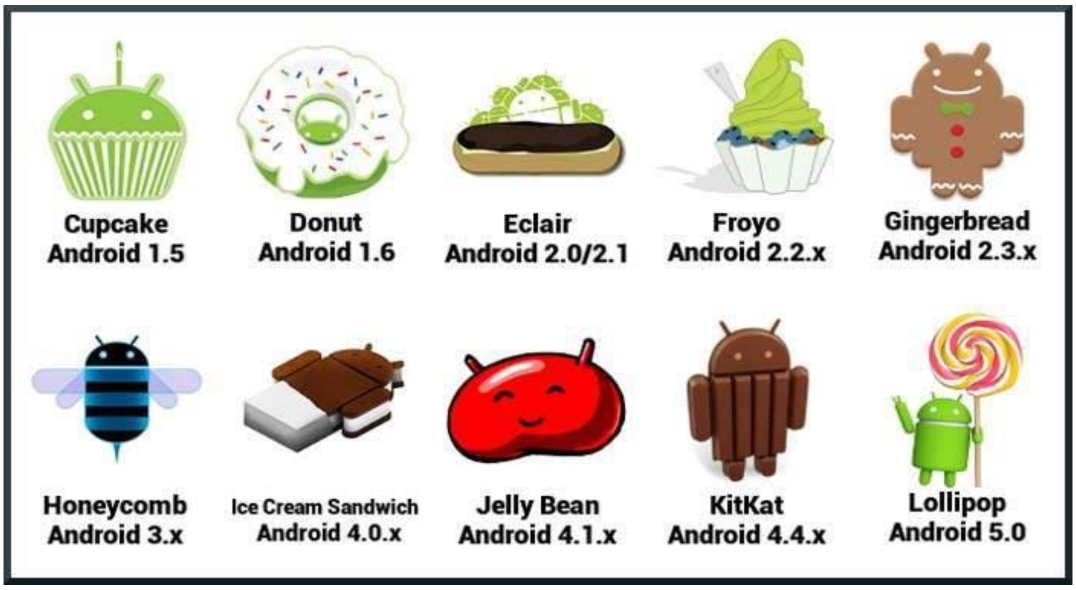


Figure 1. Android Updates from Cupcake to Lollipop

1.3 Android Architecture

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram. Android architecture or Android software stack is categorized into five parts:

1. **Linux kernel:** It is the heart of android architecture that exists at the root of android architecture. Linux kernel is responsible for device drivers, power management, memory management, device management and resource access.
2. **Native libraries (middleware):** On the top of Linux kernel, there are Native libraries such as WebKit, OpenGL, FreeType, SQLite, Media, C runtime library (libc) etc.
3. **Android Runtime:** In android runtime, there are core libraries and DVM (Dalvik Virtual Machine) which is responsible to run android application. DVM is like JVM but it is optimized for mobile devices. It consumes less memory and provides fast performance.
4. **Application Framework:** On the top of Native libraries and android runtime, there is android framework. Android framework includes Android API's such as UI (User Interface), telephony, resources, locations, Content Providers (data) and package managers. It provides a lot of classes and interfaces for android application development.

5. **Applications:** On the top of android framework, there are applications. All applications such as home, contact, settings, games, browsers are using android framework that uses android runtime and libraries. Android runtime and native libraries are using linux kernel.

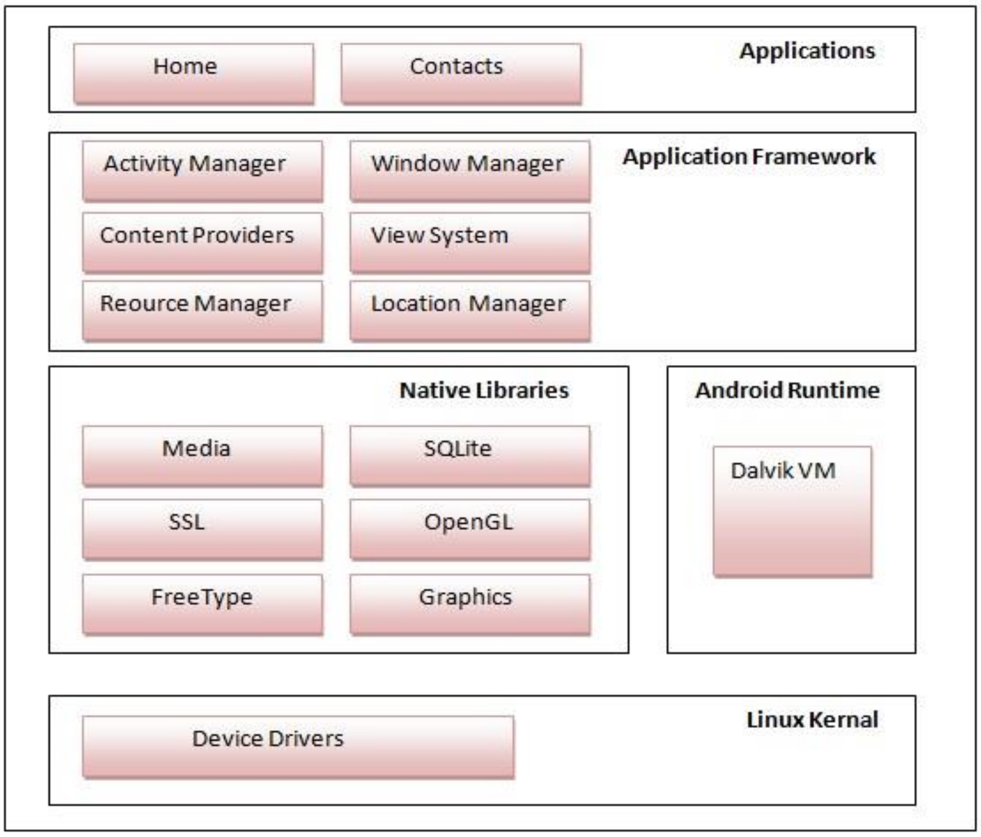


Figure 2. Android OS Architecture

1.4 Android Software Development Kit and Environment

The Android SDK is a collection of software development kits used to develop applications in the Android platform. The Android SDK is available for free to download and use from the official Android developer's website. The SDK is available for all operating systems. It can be used alone to develop an application with a command prompt or can be used with IDEs such as Eclipse and NetBeans. The IDE provides a graphical interface for easier and faster development of an application. The Android SDK consists of the following features:

- Required libraries
- Debugger
- An emulator
- Documentation for the Android application program interfaces (APIs)

- Sample source code
- Tutorials for the Android operating System

For each release of a new version of Android, a corresponding SDK is released. A separate SDK is available for a separate version of Android. In order to use the latest feature in Android, developers need to install the latest SDK depending on the phone's Android version. To develop an application Android provides a custom plug-in for 9 Eclipse called ADT. ADT provides an integrated environment for development of an application by extending the abilities of Eclipse. This helps developers to create an Android project, design an application user interface, debug an application and export and import an application package. The native language used for the Android application development is Java for which Java Development Kit (JDK) is needed. JDK can be downloaded for free from official Sun Microsystems website. All the above mentioned plug-ins and IDEs are available for free downloads, and complete sets of instructions on installation are available on the Android official website.

1.5 Android Communication

Besides enabling communication with the cloud, Android's wireless APIs provide wide range of APIs to let our app connect and interact with other devices over Bluetooth, NFC, Wi-Fi P2P, USB, and SIP enable communication with other devices on the same local network, and even devices which are not on a network, but are physically nearby. The addition of Network Service Discovery (NSD) takes this further by allowing an application to seek out a nearby device running services with which it can communicate. Integrating this functionality into our application helps to provide a wide range of features, such as playing games with users in the same room, pulling images from a networked NSD-enabled webcam, or remotely logging into other machines on the same network.

This class describes the key APIs for finding and connecting to other devices from your application. Specifically, it describes the NSD API for discovering available services and the Wi-Fi Peer-to-Peer (P2P) API for doing peer-to-peer wireless connections. This class also shows, how to use NSD and Wi-Fi P2P in combination to detect the services offered by a device and connect to the device when neither device is connected to a network.

1.6 Project Objective

The goal of this final year project was to work into the Android Communication space to compare the various technologies that are available. I have shortlisted Bluetooth and Wi-Fi for establishing a wireless connection due to the tremendous benefits, which will be discussed in detail afterwards.

In MuSync: Live Music Synching i.e. playing music streamlessly on all the connected devices, we have used the Wi-Fi connection because of the speed and Bandwidth it offers, which make it a perfect choice. Therefore, the user has the option to be a host or a client. The host is like a DJ, which selects and plays the song and all the connected peers will be playing the same music through their speakers. Thus, one can experience a loud sound without spending even a single penny on external devices like portable speakers.

A remote desktop application is built, allowing a user to have full control over the computer's mouse and the keyboard in an Android platform. The Android phone in the project was the client application acting as remote control for controlling the server desktop application. The Bluetooth 4.0 technology was used for the communication between the server and the client. The applications allow a user to control a computer without getting physically involved. The scope of the project can be extended by replacing the Bluetooth with the Internet as a communication mediator. This project then can be used to access the computer from any part of the world using the phone, provided the server and the client be in the same network.

1.7 Outline of the Report

This report is all about the Android Communication primarily the two most widely used technologies i.e. Wi-Fi and Bluetooth. Wi-Fi has an edge over other wireless technologies due to the bandwidth it offers because of which it is used for live music synching and streaming i.e. playing the same music on all the connected device. Therefore, one device acts like host and all the clients will be playing the same music. Bluetooth is used for the wireless touchpad which is used to control the Windows Desktop.

Report starts with the introduction to the project. In introduction we have covered basics of android, its architecture, communication and Project's Objective. Then we move on to

Literature survey. I have divided the literature into three: Wi-Fi based communication, Bluetooth based mouse and some general on Android Communication as a whole. In Literature Survey, I have also picked three most popular applications available on the plays stored and compare them on the basis of the key features and limitations.

After Literature survey we moved on to the MuSyc and Wireless Touchpad based dedicated chapters which deals with the basic architecture, Industrial Applications, Flow Chart and methodology behind its working. Then comes Project Design Phase which covers the various UML diagrams such as Activity, State, Use Case, DFD, Class, Database and Sequence Diagrams of both the applications. After this comes Chapter 5 which is Project Development, which deals with the Software Development Model used and the algorithm.

The Project Development phase also has the screen shorts of both the applications which is followed by the Performance Analysis of the Bluetooth Touchpad Application and MuSync Application which has been tested by distributing the applications to friend and colleagues. The Test Report and the results have been discussed in this Chapter.

Last Comes the Chapter 7, which is the Conclusion of the project which deals with the learnings outcomes, progress, efficient and future work of our study followed by the references.

Chapter 2: Literature Survey

2.1 Wi-Fi Based MuSync Application

2.1.1 Existing Systems:

We will be discussing about the various similar applications that are available at android Play Store and that could be downloaded and comparing them on the basis of performance, battery consumption and User Interface and finally tracing the bugs and areas of improvement

Group Play:

It is a pre-installed application of Samsung galaxy s4. It lets one share music play in real time with their nearby friends to play music on all devices and convert the connected devices into a single huge virtual stereo system. Group play supports a maximum of eight devices to share music play in real time but it is only supported on the Samsung galaxy smart phones (Particularly Galaxy S4) which run on android 4.2 or above. The initiative taken by Samsung is brilliant but there is a limitation that if one has Samsung galaxy s4 then also need to have at least one more Samsung galaxy s4 to act as a portable speaker. Therefore, it is rare to enjoy this functionality completely in practical life.

Key Features:

- Simplicity and practicality
- A much better product than its competitors
- Stable and very functional user interface design
- Music Synchronization was perfect (Hardly any lag)

Limitations:

- Limited to Samsung devices (particularly S4)
- Can connect to maximum of 8 devices only
- Excessive memory consumption
- Poor battery performance

Sound Seeder:

Sound seeder enables us to listen to our favorite music and synchronize our music play on the other devices by using a Wi-Fi connection on the go. The complete functionality of this app comes in the two distinct apps which user and his friends can download from the play store. Install the sound seeder music player from the play store on one device running on android 4.1 or above and sound seeder speaker on the other devices (android 2.2 froyo or above), on which one wants to share their music play and online radio streaming. It also lets one control play back and volume remotely. It has a google play music support. Premium version of sound seeder supports upto 16 portable speakers to synchronize and create a private disco while in the free version, one can connect up to two speakers for 15 minutes as often an individual wants to. In the free version one will also notice the banner ads. If a user wants to use full functionality, he can upgrade to premium by the in-app purchase.

Key Features:

- Trial Application support only works for 15 minutes
- Simple and practical design
- Supports upto 18 Connected Speakers
- Music Synchronization better than Chorus but not perfect
- Supports older Android versions also (android 2.2 froyo)
- Can select music from any Music Player Installed
- Can control sound from all he connected devices

Limitations:

- Paid Service
- Device Synchronization isn’t perfect
- Graphical User can be improved

Chorus:

Chorus brings the Samsung group play functionality to any android device running on android 2.3(Gingerbread) or above. The application is again based on the Wi-Fi connection but still faced a lot of issues and bugs in streaming and is still under the development phase. This app is available for free in play store. The complete functionality of this app comes in the two distinct apps which user and his friends can download from the play store. It has Simple and practical design. The app has a reasonably simple user interface but as it is currently in beta version so it may create some problems while synchronizing. It can support only up to six devices at the maximum.

Key Features:

- Free on Android Operating System
- Lots of bugs and Issues
- Simple design
- Can be used for any Android Devices
- Supports only 6 devices at a time

Limitations:

- Noticeable Delay while music Streaming
- User interface very poorly designed
- Takes a lot of time to pair the devices
- Lot of bugs that need to be resolved
- Can connect only 6 devices at a time.
- Under Development Phase

2.1.2 Overview of Research Papers/Journals on Wi-Fi Application

Title:	**Rio: A Solution for Sharing I/O between Mobile Systems [1]**
Author(s):	Ardalan Amiri Sani, Kevin Boos, Min Hong Yun, and Lin Zhong.
Date of Conference:	2nd June, 2014
Published in:	ACM New York, NY, USA ©2014 **MobiSys Mobile Systems, Applications, and Services**
Conference Location:	New York, NY, USA

Summary:

This research paper describes the modern and smart mobile systems, which are equipped with a diverse collection of I/O devices, including cameras, microphones, sensors, and modems. There exist many novel use cases for allowing an application on one mobile system to utilize I/O devices from another. This paper presents Rio, a I/O sharing solution that supports unmodified applications and exposes all the functionality of an I/O device for sharing. Rio's design is common to many classes of I/O devices, thus significantly reducing the engineering effort to support new I/O devices. There, implementation of Rio on Android consists of about 7100 total lines of code and supports four I/O classes with fewer than 500 class-specific lines of code. Rio also supports I/O sharing between mobile systems of different form factors, including smartphones and tablets. They applied this sharing module in various applications such as Multi-system gaming, One SIM card, many systems, Multi-sharing, Music Streaming and Multi-System video conferencing etc. and concluded that it achieves the performance close to that of I/O for audio devices, sensors, and modem, but suffers noticeable performance degradation for camera due to network throughput limitations between the two systems, which is likely to be alleviated by emerging wireless standards.

Title:	**Android and Wireless data-extraction using Wi-Fi [2]**
Author(s):	Busstra, Bert; Le-Khac, Nhien-An; Kechadi, Tahar
Date of Conference:	13-15 August 2014
Published in:	Institute of Electrical and Electronic Engineers (IEEE)
Conference Location:	University of Berdfordshire, United Kingdom

Summary:

By this research paper they have compared the traditional connection methodology like USB-cable with the latest and modern techniques like wireless networks such as Bluetooth, NFC and Wi-Fi. Today, mobile phones are very popular, fast growing technology. Mobile phones of the present day are more and more like small computers. The so-called "smartphones" contain a wealth of information each. This information has been proven to be very useful in crime investigations, because relevant evidence can be found in data retrieved from mobile phones used by criminals. In traditional methods, the data from mobile phones can be extracted using an USB-cable. The base idea is unifying voice and data onto a single network infrastructure by digitizing the voice signals. The have tried to implement this and listed the possibilities and difficulties that arise in Wi-Fi connections and the probable solutions to implement it in our project. However, if for some reason this USB-cable connection cannot be made, the data should be extracted in an alternative way. In this paper, we study the possibility of extracting data from mobile devices using a Wi-Fi connection. We describe our approach on mobile devices containing the Android Operating System. Through our experiments, we also give recommendation on which application and protocol can be used best to retrieve data. Moreover, they have compared the performance various connections like Wi-Fi, Bluetooth, NFC based on the performance parameters such as bandwidth, power consumption, reliability and feasibility with the USB connection.

Title:	**Android real-time audio communications over local Wireless [3]**
Author(s):	Román Belda, Pau Arce, Ismael de Fez, Francisco Fraile and Juan Carlos Guerri
Date of Conference:	June 2012
Published in:	Waves, Universitat Politècnica de València, Valencia (Spain)
Journal Details:	Waves - 2012 - year 4/ISSN 1889-8297

Summary:

This paper describes an Android mobile application that allows voice communications through short-range wireless networks, mainly Bluetooth and Wi-Fi. The application is able to replicate as close as possible the behavior of a two-way radio device. The application is designed to receive audio streams from multiple devices simultaneously and to send them. Therefore, they have looked into the basic architecture of Wi-Fi and Wi-Fi Direct to better understand and implement it in our project. There, proposed model would eliminate the use of service providers and would provide zero cost communication for short distance. The base idea is unifying voice and data onto a single network infrastructure by digitizing the voice signals. The main design considerations of the application, such as audio recording and playing, audio coding or data transmission, are explained through the paper. Application distribution platforms for mobile devices, such as the App Store or the Android Market, have promoted the appearance of an ever-increasing number of applications, drastically changing the landscape of the market. Although Apple pioneered this change, the convenience of the Android platform for both device manufacturers and application developers has established the Android Market as one of the most relevant application distribution platforms. Due to the fast growing number of Android terminals worldwide, application developers target this platform seeking for a massive number of potential users. The application hereby presented is bidirectional, that is, the different devices work as transmitters and as receivers.

Title:	**Wi-Fi Direct in Android Using Peer to Peer Communication [4]**
Author(s):	Neshat Karim Shaukat, Jamia Hamdard
Date of Publication:	January 2014
Published in:	International Journal for Research in Applied Science And Engineering Technology (ijraset)
Journal Details:	page: 30-37, Vol. 2 Issue I, ISSN: 2321-9653

Summary:

In this they have tried to implement intranet connection only via Wi-Fi due to its greater bandwidth instead of the previous internet connection as when we have to transfer large data or files, it isn't practically possible to use internet. Therefore, they have looked into the basic architecture of Wi-Fi and Wi-Fi Direct to better understand and implement it in our project. There, proposed model would eliminate the use of service providers and would provide zero cost communication for short distance. The base idea is unifying voice and data onto a single network infrastructure by digitizing the voice signals. The have tried to implement this and listed the possibilities and difficulties that arise in Wi-Fi connections and the probable solutions to implement it in our project. The Open Handset Alliance (OHA) a confederation of 50 Telecoms, mobile hardware, and software companies, headed by Google, were found on 5th of November, 2007 and unveiled a new platform that is called Android. At present become an important player in the mobile arena. Application distribution platforms for mobile devices, such as the App Store or the Android Market, have promoted the appearance of an ever-increasing number of applications, drastically changing the landscape of the market. This research paper evaluates and introduces the application "Wi-Fi Direct in Android using Peer to peer communication". The purpose of this research is to design and implement an android application that uses Wi-Fi direct protocol in P2P (Peer-to-Peer) as a means of communication between Smartphone's with Android at no internet cost.

Title:	**Wi-Fi AP Based Secure Data Sharing Among Smartphones and Computer System [5]**
Author(s):	Miss. Rachana N. Sawade, Prof. P. V. Dudhe
Date of Publication:	April, 2015
Published in:	International Journal on Recent and Innovation Trends in Computing and Communication
Journal Details:	ISSN: 2321-8169, Volume: 3, Issue: 4 2313 – 2316

Summary:

Smartphones operate independently of each other, using only local computing, sensing, networking, and storage capabilities and functions provided by remote Internet services. It is generally difficult or expensive for one smartphone to share data and computing resources with another. Coordinating smartphone data and computing would allow mobile applications to utilize the capabilities of an entire smartphone cloud while avoiding global network bottlenecks. In many cases, processing mobile data in-place and transferring it directly between smartphones would be more efficient and less susceptible to network limitations than offloading data and processing to remote servers. The main objective of this paper is to introduce a methodology to provide flexible media content sharing by exploiting collaborative amongst Wi-Fi devices via the temporarily-established links over the local server which is based on heterogeneous mobile which is having different mobile platform, users connected to the server like computer System via Wi-Fi. The realized prototype devices altogether show improved sharing performance by supporting two times more concurrent devices at target media quality when compared with conventional non-collaborative. In this the client and local server will upload or retrieve the data in authenticated and in confidential manner. The proposed method is based on sending/receiving data between client server via Wi-Fi connection without the need of taking any service from mobile service provider and without the use of internet connection.

2.2 Bluetooth Based Wireless Touchpad Application

2.2.1 Existing Systems:

In this we will be again looking into the similar applications that are available at android Play Store and that could be downloaded and comparing them on the basis of performance, battery consumption and User Interface and finally tracing the bugs and areas of improvement

Remote Mouse:

Remote Mouse turns the mobile phone or tablet into a wireless user-friendly remote control for the computer. It is equipped with fully simulated touchpad, keyboard and featured remote panels which make your remote experience simple and efficient. It is designed by keeping in mind the real time practical application areas such as giving a presentation or watching a movie. It has a fully simulated mouse, magic trackpad multi-touch gesture support, remote panel volume adjusting buttons with device volume side buttons, Spotify Remote Application Launcher and Switcher button to Shut Down/ Sleep/ Restart/ Log Off/ Remotely Numeric Keyboard. It can work with a Wi-Fi Connection or 3G network.
Just sit back and enjoy the day.

Key Features:

- Simplicity and practicality
- Stable and very functional user interface design
- Wi-Fi and 3G Support
- Compatible with Mac and Windows both

Limitations:

- Paid Application ($2)
- Frequent Connection Loss
- Constant Crashes
- Mouse Sensitivity can’t be controlled
- Basic features of touchpad are missing

Wi-Fi Mouse / Keyboard

It is a third party application available on Google Play Store for approximately $3. It can transform one’s Android enabled phone into a wireless mouse, keyboard, remote desktop and trackpad using Wi-Fi Mouse. It enables the user to control their PC, MAC or HTPC effortlessly through a local network connection.
Wi-Fi Mouse can perform the following functionalities:

- Wireless mouse: Support mouse left/right click and scroll
- Wireless keyboard: Support Android system keyboard and simulate computer keyboard
- Wireless joystick: Game controller for controlling your computer games
- Handwriting gestures: Control your windows, media player and presentation with cool gestures.
- Computer controllers: Control media player, internet explorer, Presentation, windows, even shutdown your computer.

Thus, one can relax on the sofa and control playing movie in the comfort of your own home.

Key Features:

- Simplicity and practicality
- Functional user interface design
- Convenient to use

Limitations:

- App isn’t smooth
- Double Click features and Scrolling mostly fails
- Connection failures
- Multiple Touch are not recognized
- Paid Application
- Payment issues

Remote Link (PC Remote)

It is again a third party application available on Google Play Store. The ASUS Remote Link turns your Android mobile phone or tablet into a WiFi or Bluetooth remote control for the PC. Including a touchpad remote, a keyboard remote, a presentation remote, a media remote and more. Touchpad remote turns your Android device into a wireless touchpad of PC which supports proper touch pad functions as well as ASUS Smart Gesture. In Presentation remote, it allows you to give a slick presentation. So easy! So professional!

In Media remote, control your Windows Media Player of your PC among your fingertips. When you connect your smartphone to your PC via WiFi or Bluetooth using ASUS Remote Link, and then connect ASUS ZenWatch to your smartphone, you can now remotely control your presentation using simple gestures on your ASUS ZenWatch. In addition it provides the functionality such as voice commands and swipe up and swipe down to launch ZenWatch to move forwards and backwards through the presentation slides.

Key Features:

- Most poplar application among wireless touchpads
- Works with Wi-Fi and Bluetooth both
- Hardly any latency and lag
- Dedicated keys to shut down the PC

Limitations:

- Difficulty in Wi-Fi Connection
- Connection failures
- Ads are annoying (flashes every 10sec)
- User Interface needs to be improved
- In the Recent update app disconnects with every ads

2.2.2 Overview of Research Papers/Journals on Bluetooth Application

Title:	**Android-based Control Interface Solution for Windows Applications [6]**
Author(s):	Iliyan Nachev, Stoyan Maleshkov
Date of Publication:	December, 2012
Published in:	International Virtual Conference
Journal Details:	Section 14. Information Technology, pp. 2073-2077

Summary:

In this research paper they have tried to implement the wireless touchpad mouse to control the windows based machine. The client application is compatible with the Android device and can monitor the desktop via wireless networks i.e. either Bluetooth or Wi-Fi. The client application can work as a wireless mouse as well as a portable keyboard i.e could be operated from a distance. All in all, the purpose of this paper is to present an approach for controlling remotely a desktop application under Windows operating system through a Wi-Fi or Bluetooth connection with a device running Android OS (smart phone or tablet). In comparison to similar techniques our solution has the advantage to be more user-friendly, provide a stable Bluetooth implementation and has minimal delay while performing the control actions. Although the Bluetooth based connection is more stable and more energy efficient yet have undesirable lags and performance issues that needs to be monitored and controlled. On the other end the Wi-Fi connection has more bandwidth and has better performance in comparison to the Bluetooth. Currently the Bluetooth connection is secured by default and cannot be easily sniffed or decoded. But this statement does not apply for the proposed Wi-Fi solution. The UDP packets are sent in clear text and could easily be sniffed, analyzed and compromised.

Title:	**Concept of Remote controlling PC with Smartphone Inputs from remote place with internet [7]**
Author(s):	Dr. Khanna SamratVivekanand Omprakash
Date of Publication:	January 2012
Published in:	International Journal of Advanced Research in Computer Science and Software Engineering
Journal Details:	Volume 2, Issue 1, ISSN: 2277 128X

Summary:

This paper represents how your PC can be controlled from remote place with your smartphone device with the help of Internet. It means the monitor of PC will be seen in mobile. It turns your phone into a wireless keyboard and mouse with touchpad, using your own wireless network. This application can be performed on android based mobile. It requires server application for your computer. It requires device running on the Android operating system with some sort of wireless connection between them. By getting IP address from the PC and directly browse it on mobile phone. The PC screen will be access on the mobile. A functioning wireless network to which your computer is connected or not in between mobile and PC. Another way to browse the PC on mobile by assigning real IP to the PC and make the PC web hosting server by DNS entry into ISP provider. A gateway of ISP and real IP can be easily available from ISP and purchasing domain name from market. In the control panel of domain, you can assign your IP address and put DNS entry into it. On mobile by accessing domain name or real IP on application server using browser which is installed on mobile. Access the PC directly into the mobile can be able to access its functions by mobile keys and using inbuilt keyboard. The beauty is that PC is connected with the internet and mobile is also connected with internet by its network facility due to sim card. Mobile user can access information from PC on remote place due to the internet connection. Operations can be controlled by mobile remote device of PC Operations will be processed on the CPU of PC and output will be send to the mobile for executing sms using sim card of network operator. A central database server is hosted on the real IP with particular domain name. A web hosting server is based on windows with IIS and apache.

Title:	**Controlling PC/Laptop via Android Phone (Android Remote Control) [8]**
Author(s):	Lushin Barde, Neha Dhole, Pragati Waghmare, Swati Suryawanshi
Date of Publication:	January 2014
Published in:	International Journal of Engineering Research & Technology (IJERT)
Journal Details:	Vol. 3 Issue 1, ISSN: 2278-0181, page-2806-2809

Summary:

By this research paper they represent how the PC and Laptop can be controlled from remote place with smartphone using internet. It basically turns your smartphone into wireless keyboard and mouse with touchpad. This application can be performed with some wireless connection between the PC or Laptop and the smartphone with Android operating system. By accessing the IP address of PC, we can establish a connection between them using Wi-Fi connection. This application not only turns your smartphone into wireless keyboard and mouse but it also provides various other features including voice to text conversion. The implemented application consists of two parts, the first one is an application for Android smartphone and the second one is a server application that executes the command selected by user’s application. Smartphone’s are common and commercially used device all over the world, user-friendly interface and lots of features such as Wi-Fi, Internet access, Bluetooth, Camera, Video recording etc. add-on to the Android smartphone to be popular all over the world with cheap cost. We propose application which is compatible and useful in both the areas, the aim is to utilize provided hardware features from smartphone devices along with various useful libraries from Android API. The outcome of this implementation is a handy, easy-to-use application.

Title:	**Remote Trackpad : Android Controlled - Computer Mouse Pointer using Bluetooth [9]**
Author(s):	P.Sanoop Kumar, K.Teja K.Bhavyaka, E. Sirisha
Date of Publication:	January 2015
Published in:	International Journal of Advanced Research in Computer Science
Journal Details:	Volume 6, No. 1, ISSN No. 0976-5697, page- 83-87

Summary:

This paper represents how your PC can be controlled from remote place with your smartphone device with the help of Internet. It means the monitor of PC will be seen in mobile. It turns your phone into a wireless keyboard and mouse with touchpad, using your own wireless network. This application can be performed on android based mobile. It requires server application for your computer. It requires device running on the Android operating system with some sort of wireless connection between them. By getting IP address from the PC and directly browse it on mobile phone. The PC screen will be access on the mobile. A functioning wireless network to which your computer is connected or not in between mobile and PC. Another way to browse the PC on mobile by assigning real IP to the PC and make the PC web hosting server by DNS entry into ISP provider. A gateway of ISP and real IP can be easily available from ISP and purchasing domain name from market. In the control panel of domain, you can assign your IP address and put DNS entry into it. On mobile by accessing domain name or real IP on application server using browser which is installed on mobile. Access the PC directly into the mobile can be able to access its functions by mobile keys and using inbuilt keyboard. The beauty is that PC is connected with the internet and mobile is also connected with internet by its network facility due to sim card. Mobile user can access information from PC on remote place due to the internet connection. Operations can be controlled by mobile remote device of PC Operations will be processed on the CPU of PC and output will be send to the mobile for executing sms using sim card of network operator. A central database server is hosted on the real IP with particular domain name. A web hosting server is based on windows with IIS and apache.

2.3 General Overview of Research Papers/Journals

Title:	**Instant Messaging Service on Android Smartphones and Personal Computers [10]**
Author(s):	Priya Mehrotra, Tanshi Pradhan and Payal Jain
Date of Publication:	April, 2014
Published in:	International Journal of Information and Computation Technology
Journal Details:	ISSN 0974-2239, Volume 4, Number 3, page. 265-272

Summary:

By this research paper they tried to implement a chat server for many platforms as Android is an operating system for smartphones, tablets and now will be used for Personal Computers also. It includes a touch screen user interface, widgets, camera, network data monitoring and all the other features that enable a cell phone to be called a smartphone. Basically, Multi-Purpose chat application allows users to send asynchronous messages, and enable sharing image files with other peers on the JXTA world using JXME. Instant messaging has become so ubiquitous; an entire generation of internet users is probably unaware there was ever life without it. The use of instant messaging nowadays is more than the calling function itself. The main objective of this paper is to introduce a methodology to provide instant Messaging Service over the intranet which is addressed to android based smartphone and tablet users connected over intranet via Wi-Fi. The proposed method is based on sending/receiving messages in intranet through intranet server via Wi-Fi connection without the need of taking any service from mobile service provider and without the use of internet connection. They have shown or discuss this by using Bluestacks software which will provide an efficient and fast way to perform instant messaging which will further increase the performance. With IM, one can keep a list of people he interacts with. User can IM with anyone on one's **buddy list** or **contact list** as long as that person is online. One types message to each other into a small window that shows up on both of the screens.

Title:	**Peer-to-peer communication on android-based mobile devices: Middleware and protocols [11]**
Author(s) :	Jabbar, W.A. , Ismail, M.Nordin
Date of Conference:	28-30 April 2013
Published in:	Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference
Conference Location:	Hammamet

Summary:

In this paper, they have showed the increasing interest of a wide number of scholars and researchers in promoting the applications for Peer-to-Peer (P2P) communication in Android based smartphones as ubiquitous. Such increasing interest has been attributed to their increasing perception of these devices. There is a wide use of many P2P applications in mobile devices. These applications include IM, VoIP, file sharing, social networks, and video streaming. This paper describes a proposed Android-based middleware acts as the interface point between mobile nodes and higher application layers for mobile ubiquitous computing. The middleware mainly aims at supporting and enhancing the protocols for direct P2P communication among users in the ensemble mobile environment. Moreover, the paper presents a discussion of the available P2P middleware for the mobile environment along with its applications. Furthermore, the paper moves on to providing the mobile devices limitations and the challenges encountered in the adoption of the P2P communication technology in the mobile environment. Finally, the paper is concluded by presenting the directions for future research as to develop a middleware with necessary APIs and implement an enhanced P2P protocol in the proposed middleware on Android-Based mobile devices.

Title:	**Intranet Based Messaging Service on Android Smartphones and Tablets [12]**
Author(s):	Bhoopesh Kumawat, Sudhendra Pal Singh, Chandra Prakash Verma
Date of Publication:	July, 2013
Published in:	International Journal of Advanced Research in Computer Science and Software Engineering
Journal Details:	ISSN: 2277 128X, Volume 3, Issue 7, *page. 1030-1032*

Summary:

In this Journal they have compared the internet and the intranet technologies. Instant messaging has been widely used with the power of internet; people can use an IM talk to family, friends and co-workers. In the company, colleagues can send and reply instant message in real time without face to face; meanwhile the work report can be shared during the instant chat session. People can speak to multiple people in the virtual conference, share ideas and get conclusions. People on a business trip can contact the co-works inside the company through IM as well. What's more, the staff can talk to customers or vendors online as well, in other word, now people can do business through the instant messenger direct rather than use the traditional method like make phone calls. The use of instant messaging nowadays is more than the calling function itself. The main objective of this paper is to introduce a methodology to provide instant Messaging Service over the intranet which is addressed to android based smartphone and tablet users connected over intranet via Wi-Fi. The proposed method is based on sending/receiving messages in intranet through intranet server via Wi-Fi connection without the need of taking any service from mobile service provider and without the use of internet connection.

Title:	**Implementation of Voice, Video and Text Data over Wi-Fi [13]**
Author(s):	Prof. Anil Hingmire, Ms. Mrunal Tipari , Mr. Rohit Gopalan, Mr. Sanman Chavan
Date of Publication:	March-April, 2015
Published in:	International Journal of Engineering Research and General Science
Journal Details:	Volume 3, Issue 2ISSN 2091-2730

Summary:

Communication through mobile phones comes at a certain expense. Therefore, if we cut down the expenses and make this communication a cost-free communication. This is the concept of Voice over Wi-Fi. This paper consists of three modes of mobile based communication over Wi-Fi i.e. Messaging, Voice calling and Video calling. Almost all of the recent launches of phones comes with Wi-Fi. The number of people using Wi-Fi devices is increasing day-by-day. There, proposed model would eliminate the use of service providers and would provide zero cost communication for short distance. The base idea is unifying voice and data onto a single network infrastructure by digitizing the voice signals, convert them into IP packets and send them through an IP network together with the data information. Internet-based instant messaging applications allow users to send/receive messages over the internet. There are many such existing apps. For example, Viber, Skype, BBM etc. But these applications require internet connection from mobile service provider which is not free of cost. In this paper, they proposed a communication system that allows android based smartphone users to send and receive messages over the Wi-Fi range which requires neither any internet connectivity nor any messaging service from the mobile service providers.

2.4 Integrated Summary of Literature Review

By looking into all the applications available on Play Store, it becomes absolutely clear that there's still a lot to be done in the intranet domain especially in the Audio streaming and sharing technology. There have been many attempts to overcome this communication barrier by syncing two devices to real time Music-Streaming but still the available applications such as Chorus, Sound Seeder and Samsung Group play are still not the perfect applications. The real time music sharing is yet to be achieved and the emerging hardware standards and the continuous improvement in technology can surely overcome these issue. In addition, the live Music Streaming has not been taken up by many and there's still a lot of scope of research and improvement needed in this space which can improves it to a great extent. This all motivated me to explore into this technology and make my contribution.

In case of the wireless mouse a lot of applications exist but most of them are the paid applications. Furthermore, they all suffer from the frequent connection loss and a lag i.e. aren't smooth enough to be worth using in day to day life. Most of the applications are developed using the Wi-Fi technology Bluetooth being more efficient in power management and the chances of frequent connection breaks are relatively less which motivated me to develop a wireless touchpad using the Bluetooth.

To sum up there have been a lot of connectivity options available in today's smartphones, yet internet has been used widely but it has an additional data cost and having the bandwidth limitation especially in the developing countries like India, it is not the most feasible connectivity platform for heavy files and music Streaming. Therefore, narrowing our study to the other's, we are left the Wi-Fi, Bluetooth, NFC, WLAN etc., which will provide much more transfer speed and being absolutely free as no additional cost being involved, are much better than Internet. Although, after studying these resources it become absolutely clear that Wi-Fi is still the most reliable and still desired connectivity parameter due to its speed, reliability and comparatively much better range than other connectivity parameters that are pre-equipped in the latest handsets yet the Bluetooth is still more reliable and much more power efficient. This motivated me to work on both the technologies i.e. Wi-Fi and Bluetooth in two different domains i.e. Live Music Syncing and a Wireless Touchpad respectively.

2.5 Motivation to Our Work

There has been a lot work done in intranet communication, the applications to share data or group messaging apps, but still the internet comes out to be the center of most of the over the top applications due to its availability.

Through the literature survey, I came to the conclusion that most of the work in local area network has been done on chat, data sharing and location based application. Still real time audio streaming applications to play the audio simultaneously on all the connected devices by inheriting the input/output devices of the smartphone and other portable devices. The applications like Chorus, Soundseeder and Group play aren’t successful enough as they face certain limitations and performance issues. Furthermore, Wireless Touchpads are mostly based on Wi-Fi Technology but I have tried to implement it using the Bluetooth because of the tremendous benefits such as more reliability, efficiency and less connection failures. Therefore, with the help of these project, I am trying to use the two most widely used connections i.e. Bluetooth and Wi-Fi and compare their performance in real time. So this motived me to design the applications that will overcome all the issues that are listed above in this report and that could prove to be useful in future.

2.6 My Contribution

By designing these application, we are trying to overcome the limitation faced by the users of similar existing applications that could prove to be really helpful for the users and we will be trying to minimize the synchronization gap which is common in all the existing applications. Moreover, we will be keeping our work as an open-source project on GitHub so that it could be used and could be improved the open-source community to meet the requirements of emerging users.

MuSync application could prove to be really helpful for travelers who don’t want to carry any extra luggage, small group parties and students who want to listen to loud music and don’t have the resources and while giving presentations or during lectures, watching movies etc. Bluetooth Wireless Mouse can be of great use.

Chapter 3 MuSync: Wi-Fi based Music Syncing

In this chapter we will be looking into the basic overview of the application MuSync i.e. Live Music Synching. With this project I am aiming to provide an application that will prove to be useful for all the age groups by providing a daily life solution, which is listening to loud music without spending upon an external accessory. By this application we are trying to inherit the speakers of all the connected devices and play same audio on the devices simultaneously.

This can prove to be extremely helpful for a group of travelers who can’t carry any extra weight and small group gatherings to experience a great music quality.

3.1 Why (Peer to Peer) P2P Connections

It’s very important and is a practical necessity to communicate other peers in a secure manner with off-the-shelf devices without any access point or internet because the direct communication between two mobile devices does not rely on any infrastructures and has very low cost and very fast preparation to begin this direct wireless communication. There are many scenarios which need this direct peer-to-peer wireless communication.

3.2 Industrial Applications

We are working on Live Audio Sharing as a part of our project and various connection methodologies have been depicted in this Report. This direct communication of Live Sharing allows two or more peers to share and play music via wireless channel by directly using two or more mobile devices.

In order to meet all those needs requiring Live Audio Streaming we designed and specified some goals or principles to design the wireless direct communication framework. First of all, this direct mobile communication does not rely on access point. Second of all, this direct mobile communication is wireless. Third of all, this direct mobile communication is low latency, meaning that the audio is responded very quickly. Fourth of all, this direct mobile communication is high bandwidth, meaning that there is no congestion and audio is very smooth. We summarized these design goals or principles to be ad-hoc, wireless and real-time. To implement these design principles, we choose to implement our music-sharing in android mobile devices on top of Wi-Fi-Direct.

3.3 Block Diagram Overview

Our main goal is to use Wi-Fi Direct and Audio real time playing to setup a communication between android devices which are in the same group frequency ID. Therefore, our Music-Streaming application's architecture has to handle Wi-Fi-Direct communication, Audio Player and Group Music Play. The MuSync architecture is divided into three layers: WiFi-Direct Layer, Music-Streaming Layer and Manager Layer.

The **Wi-Fi Direct Layer** is used to handle the Wi-Fi Direct communication including discovering peers, connecting peers and maintaining Wi-Fi Direct connection status. Inside this Wi-Fi-Direct Layer, there is a WifiBroadcastReceiver object; this object will help to monitor any event changing and inform our Music-Streaming Layer.

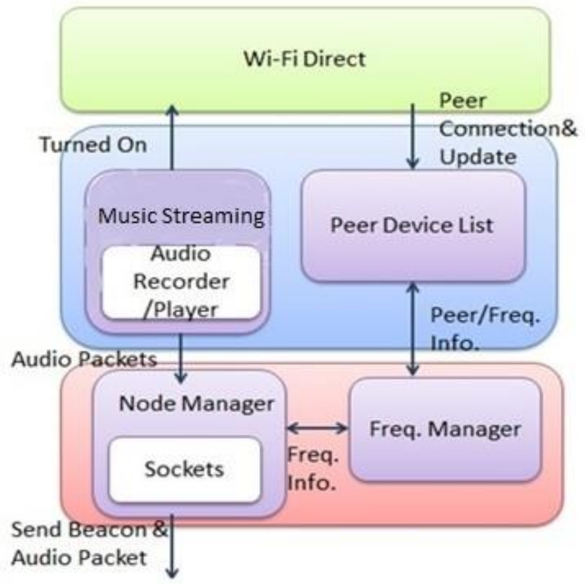


Figure 3. Music-Sharing Architecture

The **Music Sharing Function Layer** helps to handle the user interface and audio communication between peers. Once the application is turned on, it will generate a signal to inform Wi-Fi-Direct layer to discover and connect peers. Moreover, there is a peer-device-list object which is used to maintain available peers and store the peer information in the database. If Wi-Fi-Direct layer has discovered some peers, the Wi-Fi-Direct layer will generate peer list information and pass this information to the Music-Streaming layer.

The **Manager Layer** has two main functions: Node Manager and Frequency Manager. For the function of Node Manager, it helps to construct sockets that send/receive audio and beacon messages to/from peers. The Frequency Manager contains a frequency map which is used to decide whom it should send message to. In our Music-Streaming application on the android phone, user can only talk to another device if they have same virtual frequency. This behavior is like the real Music-Streaming device that uses the frequency to modulate the audio message. The receiver needs to know the frequency to hear the audio message. Therefore, in our MuSync application, we introduce this virtual frequency as a talk group ID that the audio message can only transmit and receive if they are in the same group.

3.4 Flow Chart

The flow chart shows how our Music-Streaming works when the user turns on the application. First of all, when the Music-Streaming on/off button is turned on, the WiFi-Direct will start to discover peers and connect to peers. After successfully connecting, the Music-Streaming will create three objects: **Frequency Manager, Node Manager and Audio Player**. Moreover, it will set the **talker button** valid to let user can start to talk to each other. The next figure shows more details about these three objects.

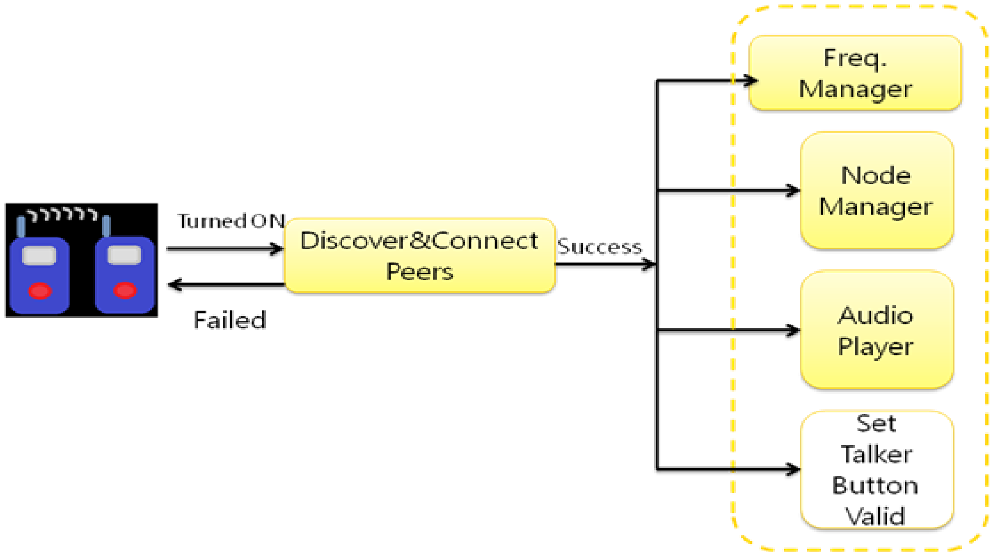


Figure 4. Audio-Streaming Flow-Chart 1

In our Music-Streaming, after connection is set up, we will have two kinds of flows: **Beacon Flow (control message)** and **Audio Flow (audio message)** to handle exchange information between devices. First of all, when a WiFi-Direct group is formed, due to the behavior of WiFi-Direct, each WiFi-Direct group has a group owner and rests of devices are group members. All devices in the group only know the group owner's IP address. This means that in the beginning they can only send data to the group owner. Therefore, we create two steps registration mechanism to handle this issue. If the device is not a group owner, it must register its' IP address to the group owner. After that the group owner can help to disseminate members' address to other devices.

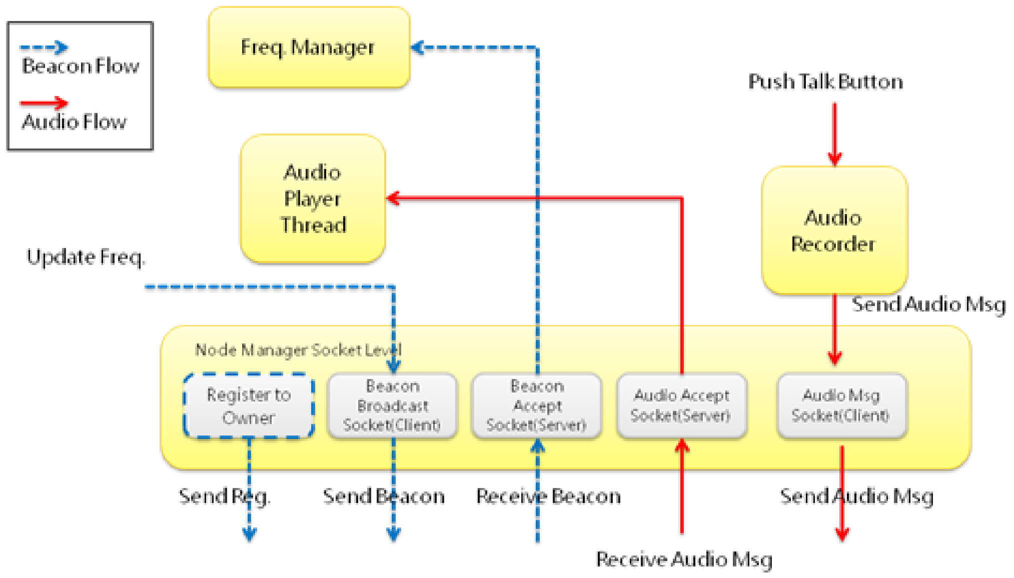


Figure 5. Audio-Streaming Flow-Chart 2

For the Beacon Flow, the **Beacon Broadcast Socket (Client)** will keep broadcast the control message which including the IP address and the virtual frequency. If the user updates the virtual frequency, this broadcast socket will help to send new beacon message to peers immediately.

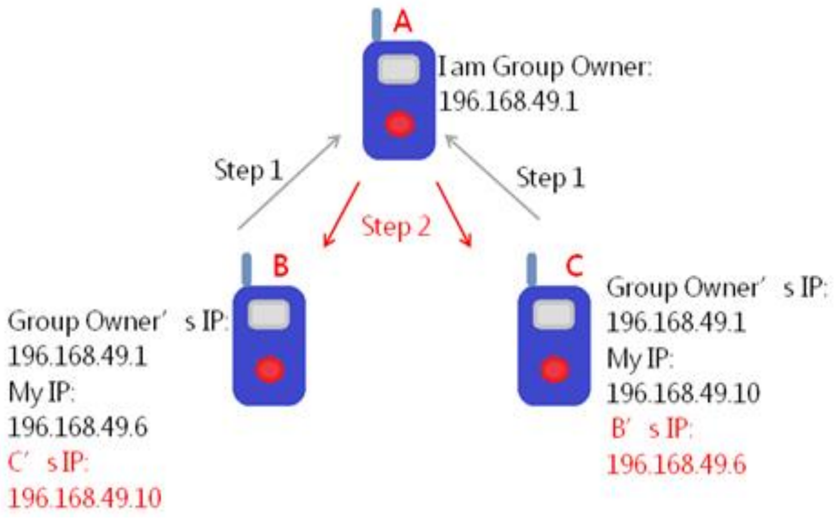


Figure 6. Smart Phones registration steps

3.5 Audio Queue Architecture

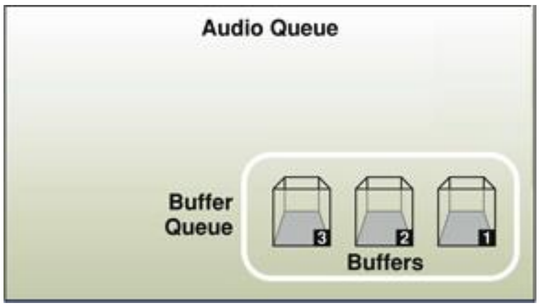


Figure 7. Audio Queue illustration

Every audio queue has virtually the same common data structure, containing three key elements:

- **Audio queue buffers:** a series of audio queue buffers, each of which is a temporary repository for some audio data
- **Buffer queue:** an ordered list for the audio queue buffers
- **Audio queue callback:** a callback function written by an application

The audio data structure in Figure 7 is employed both in audio file Streaming and audio playback.

3.6 Audio Queues for Forwarding the Clients

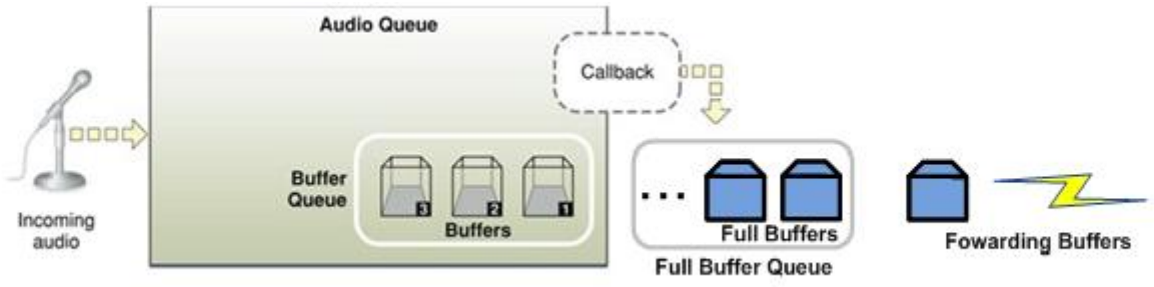


Figure 8. Illustration of Audio Queue for Recording

As shown in Figure 8, the incoming audio another connected device. And the incoming audio is stored and maintained by an audio queue. Figure 9 shows how the process audio recording works. Once the previous full buffer is read to the Music-Streaming and become empty, this empty buffer is returned to the Buffer Queue and pushed to the end of the audio buffer queue to be recharged or filled again.

In our Music-Streaming, we use our own audio queue to store the full buffers read from the audio Buffer queue of recorder. This queue is the reason for real time latency issues. In real time, the full buffer in our Music-Streaming queue is forwarded to the destination through our protocol on top of wifi-direct sockets.

3.7 Audio Queues for Playback

The second type of flow is Audio Flow which handles audio communication messages. When the MuSync is turned on, the Audio-Player Thread is created and is run in the background. After received audio message, the Audio Accept Socket will send the audio message to the Audio-Player Thread. Therefore, the user can hear the voice played by the Audio-Player Thread. The down layer Audio Message Socket will create sockets and send out the audio message to those devices in the same frequency group.

As shown in Figure 9, the outgoing audio is entering the loudspeaker. The incoming audio full buffers are being received from the sender by using protocol on top of Wi-Fi-direct. And the incoming audio is stored and maintained by an audio queue. However, the callback mechanism is on the left side of Audio Queue different from the right side of Audio Queue in Audio Recording. Figure 10 shows how the process audio playback works.

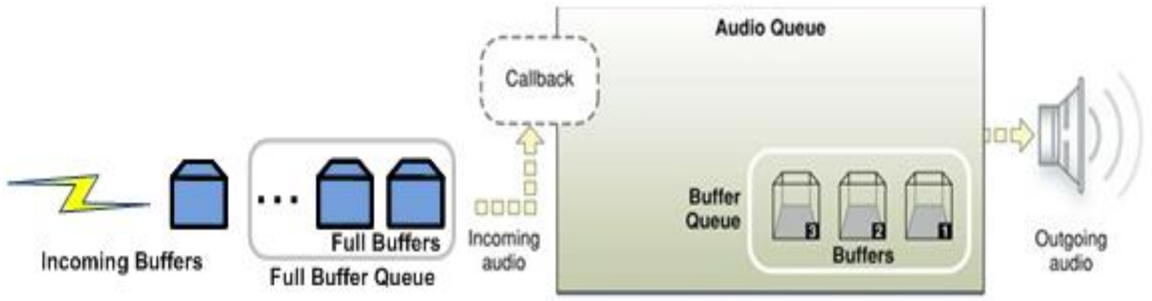


Figure 9. Illustration of Audio Queue for Playback

These incoming audio full buffers are stored in our own audio buffer queue in Music library. The reason to employ our own full buffer queue is to eliminate the non-synchronization of the incoming data and the outgoing data. The buffer in our own audio buffer queue is written into the buffer in the audio queue of the audio player. This writing process is controlled by a callback mechanism.

When the buffer is empty in the Audio Queue, it triggers callback to read data from our audio buffer queue. Once the buffer is fully filled or charged this full buffer is pushed into the buffer queue of the audio player to be lined up to be played. As soon as the full buffer finishes its playing, this empty buffer is popped out and triggers another callback to write audio data from our audio buffer queue. Simultaneously the full buffer in the second place is pushed to the head of the buffer queue to be playing.

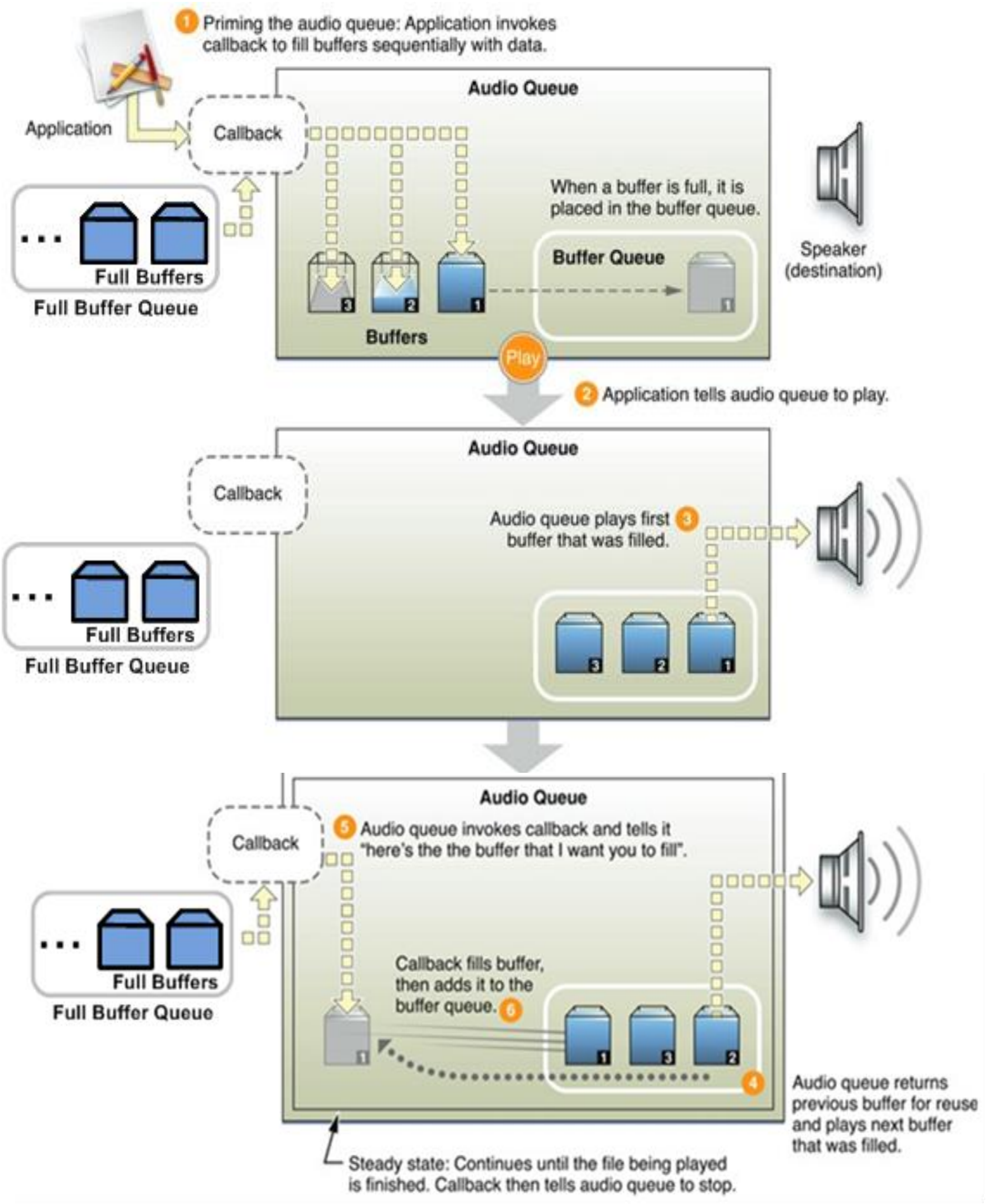

1 Priming the audio queue: Application invokes callback to fill buffers sequentially with data.
Application
Callback
Audio Queue
When a buffer is full, it is placed in the buffer queue.
Buffer Queue
Speaker (destination)
Full Buffers
Full Buffer Queue
Buffers
Play
2 Application tells audio queue to play.
Audio Queue
Callback
Audio queue plays first buffer that was filled. 3
Full Buffers
Full Buffer Queue
Audio Queue
Callback
5 Audio queue invokes callback and tells it "here's the the buffer that I want you to fill".
Full Buffers
Full Buffer Queue
Callback fills buffer, then adds it to the buffer queue. 6
4
Audio queue returns previous buffer for reuse and plays next buffer that was filled.
Steady state: Continues until the file being played is finished. Callback then tells audio queue to stop.

Figure 10. The process how the audio playback works

Chapter 4: Bluetooth Based Wireless Touchpad

In this chapter we will be looking into the basic of Wireless Touchpad Application. The purpose of this application to control the desktop based in Windows operating system through a Bluetooth connection with a portable device running on Android OS (smart phone or tablet). This will be extremely helpful while typing on the screen of a tablet PC, giving presentations during lectured or controlling the screen while watching movie etc.

In comparison to similar techniques my solution has an advantage of being more user-friendly, provide a stable Bluetooth implementation and has minimal delay while performing the control actions.

Remote control of a desktop computer is a convenient feature that could be performed by handheld devices. The solution provided in this paper relies on a smart device running Android OS (smart phone or tablet) and a Windows-based PC and offers a virtual touchpad and virtual keyboard that send commands to the desktop OS in real time. Such application bundle could ease the use of many application giving the user the ability to control them from any place in the room without having to bring a keyboard and a mouse with himself. It is suitable for virtual reality laboratories where a wireless mouse is not enough to implement advanced interaction techniques. Furthermore, the touch screen is comfortable to use as a mouse substitution when a person is standing in front of the audience giving a presentation. Java is the programming language used on both the desktop and the mobile devices, thus giving the opportunity for easier porting to other desktop operating systems.

4.1 Server based Application

The server software resides on the desktop PC (Windows). When started it awaits connection through Wi-Fi and/or Bluetooth. The remote control is provided by a Java SE API. The architecture follows the standard practices - main modules, used by a business logic layer, which is presented to the user with GUI.

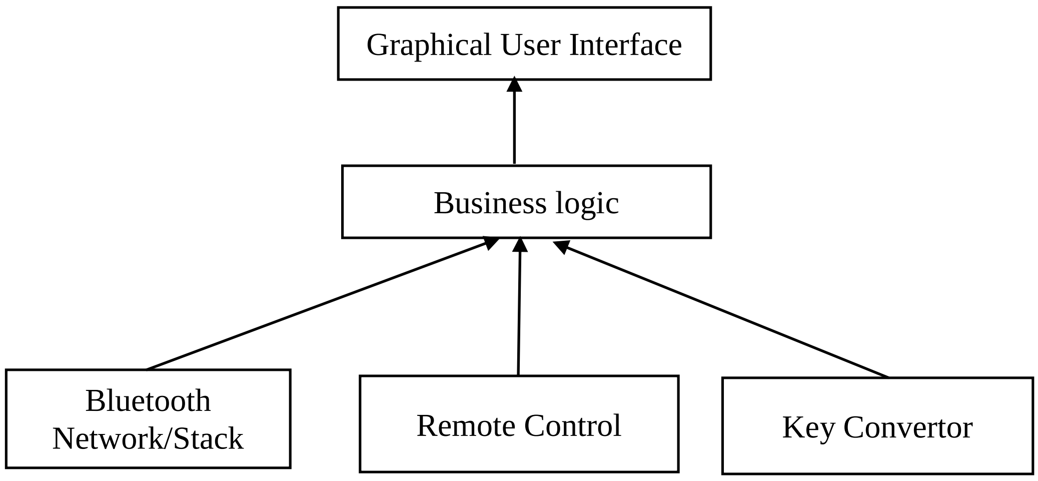


Figure 11. Bluetooth Server Application Architecture

The server application is distributed as executable JAR file and is intended for silently running in the background. Implementation details of the main modules are given below:

1) **Bluetooth stack:** Java SE doesn't provide an API for controlling a Bluetooth device, so in order to implement such kind of connection a third party open-source library (Bluecove) is used. It's an implementation of Java Specification Request 82 (JSR-82) and provides high-level control Microsoft Bluetooth stack. It has also an additional module for supporting Linux BlueZ.

   The implementation on top of the library is the following: first the Bluetooth device is made discoverable. Then a new connection notifier is opened for the unique application URL and finally the application starts waiting for a new connection. For Bluetooth connections the echo part of the protocol is obsolete, because the pairing of the devices takes care for this.

2) **Key convertor:** The key codes defined by the Android platform are unique for Android OS and do not correspond to the key codes found in the Java SE platform. There is also no algorithm available for converting between the two sets of key codes. That's why in order to operate correctly, the application needs a key translation module. The implementation uses a hash table with pre-populated pairs - the keys are the Android key codes and the values - the Java SE key code equivalent.

3) **Remote control:** The remote control in Java SE is straight-forward to implement using the Robot API [5] - it offers methods for simulating press and release of keyboard keys, full mouse control (keys, wheel, movement) and function to delay sending of events (which is needed for proper emulation of mouse click).

4) **Business logic:** The Message Parser is responsible for reading and understanding the protocol message received from the communication module. If the message has any faults (does not end with termination sign, does not contain a supported eventor the event has wrong parameters) an appropriate exception is thrown. If the message is correct, the appropriate method of the events listener is called.
Discoverable Thread is an abstract class that extends Thread and gets extended from the Bluetooth. The class holds a message parser (for handling of incoming messages) and a remote events listener (for the actual control over the PC). The child classes should rely on the parent implementation to handle the protocol message when receiving one and implement callbacks for "echo" request and for release of network-related resources (called when the thread should free up all its resources). The class where the server application starts is called Remote server. It holds the application screen and provides the binding between the discoverable threads and the UI. The main method creates the screen and starts the Bluetooth threads. The action listener is called by the GUI components (buttons and context menu) and performs the restart of the connectivity threads and implements the exit functionality. The Bluetooth listeners are called on successful initialization or when an error happens. That way the user could see the current status of both connections.

5) **User interface:** When started, the server begins to listen for Bluetooth connections automatically. As a consequence the user interface needs to provide only a user-friendly interaction for manually restarting the Bluetooth modules as well as information about the local IP address, Bluetooth status and error messages. The GUI consists of a single screen with buttons and labels and a system tray icon with a context menu.

4.2 Client based Application

The client software is launched by the user on the Android device. It searches for a server instance through Bluetooth and offers the ability to remotely control the desired PC. In terms of architecture it follows the same basic ideas present in the PC software - user interface that interacts with the business logic which covers a small set of different modules.

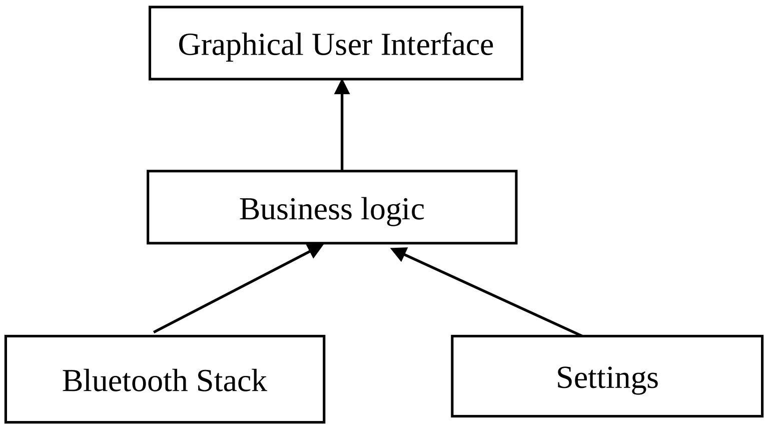


Figure 12. Bluetooth Client Application Architecture

The client application is distributed through a standard Android application package file (APK). The user interface provides a superb user experience, making the Android device (no matter if a smart phone with small screen or a large tablet) an excellent touchpad with keyboard input ability. The multi-touch capabilities of the devices are also utilized to full extent. The minimal Android version required is 2.2. (API level 8). Android devices which don't have a touch screen (e.g. smart TVs) are not supported.

1) **Connecting through Bluetooth:** Bluetooth as connection type is available since Android 2.0. The SDK offers a high-level API which gives information about the availability of the hardware, the list of paired devices, discovery-related methods and interface for radio frequency communication (RFCOMM). Important to note is that aperquisite for successful communication is the pairing of the devices prior to usage in the application. Because the server is found by the client through the Bluetooth discovery process and pairing, the echo/reply part of the protocol are obsolete. Therefore, the client application sends only events (messages) through Bluetooth. An error handling mechanism notifies the user interface when the connection is lost.

2) **Business objects:** The business logic of the application is implemented in different classes with well-defined functions as described shortly below. They are then used in the GUI (Activities) where the handling of the user input occurs. The Remote Device class is a business object which represents a remote network. Its purpose is to help handling both type of possible connections in a unified manner on the user interface layer. It contains information about the type of the connection, the device address and the device name. The Message Builder generates protocol messages, while an implementation of the Message Sender interface sends the messages to the remote device. The Bluetooth message senders implement methods for initialization and release of the network objects. The actual sending is realized through a Bluetooth socket. On valid reply message the UI layer shows the new server(s) to the user.

3) **User interface and corresponding business logic:** The application has three screens - login screen, a screen which offers the actual remote control and settings screen. The login screen is the initial screen of the application. Its purpose is to give the user the ability to connect to the desired PC (server). The user interface offers a brief introductory message, a list which contains the names of Bluetooth devices found at the moment that have replied to the echo request, two buttons for manual search (through Wi-Fi or Bluetooth) and an input field to manually enter the server IP address.

   The remote control screen exposes the essential functionality of the application. It's the place where the end user will spend most of hers/his time. In this activity reside the following user interface components: the virtual touchpad, trigger for the virtual keyboard (and the keyboard itself), three "special" buttons - Ctrl, Shift and Alt (shown/hidden from the action bar/menu) and four "presentation" buttons - shortcuts for presentation software products - F5 (full screen), left arrow (previous slide), right arrow (next slide) and escape (exit full screen). This activity runs in full screen and prevents the device from sleeping.

In order to consume all key events, the activity overrides the dispatchKeyEvent() method - that's the place where all key events from the hardware and virtual keyboard are handled. Some of the keys (e.g. back, camera, menu buttons) are meaningless for sending to a remote PC, so they are ignored and passed to be handled by the default implementation (the super method). Based on the button state (action up or down) a different message with the given key code is constructed and sent to the remote device. In the case of action up, all special keys are released, because that way the user experience is more intuitive. The most important and complex part of the activity logic is the one connected to the touch events (mouse/touchpad). The desired functionality from user's perspective looks the following way:

- **Left mouse click** is performed by making a tap.
- **Double left click** is made by tapping fast two times.
- **Right click** is done by tapping with two fingers.
- **The middle button** of the mouse is clicked when tapping with three fingers.
- **Moving a finger** around is interpreted as mouse move.
- **Moving two fingers up and down** is used for scrolling.
- **Tap and hold** results in down state of the left button

and ability to move afterwards - selector. From implementation perspective - first of all the onTouchEvent() method gets overridden. That way all touch events are dispatched from a custom logic inside the activity. Based on the pointer count (how many fingers are involved in the touch gesture) a different method is called. The last pointer count is saved for further use in the business logic. By holding information about the current and previous states (e.g. last position of the touch event, timestamp for start of one-, two and three-finger touch) the aforementioned user experience is implemented in a consistent manner. When the user executes faster move gestures, the mouse also moves faster (by doubling the values of the horizontal and vertical move). The application settings screen offers adjustment of the touchpad sensitivity and an invert scrolling option.

Chapter 5: Design of the Project

In this Chapter we are designing the UML (Unified Modeling Language) for our application i.e. MuSync which is designed for streaming the live music by playing it on all the connected devices by inheriting the speaker functionality of all the devices and for wireless touchpad applications via Bluetooth technology. We are now going to design the various UML diagrams such as Activity, State, Use Case, DFD, Class and Sequence to display its functionality and the working of project.

5.1 MuSync: Music Syncing Application

This application aims at developing a highly – efficient software which would cater to the basic need of the people who love to gather or who love to go on trips without unnecessary bag packs. It can be used for various purposes depending upon the situation to suit requirements of the users. It is developed in a manner so it can be used by anyone from a world famous IT Company to an ordinary man who barely knows how to operate the computer.

5.1.1 Use Case Modelling

It helps us to describe the functional requirements of the application developed in the overall system.
Use case diagrams gives a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions are interacted.

Use Case: Music Sharing

Primary Actor: User

Goal: To play the same music file in all connected devices at the same time.

Use Case Description:

Actor: User

Pre-Condition: The user must know how to operate and use a cellphone.

If the user clicks the application, then the application should open up. Then the user can choose to login or use the app without logging in. If the user logs in, they he will be able to use our cloud library or else he can only use his local storage library. After the particular library is loaded, user can either choose to become host or join an already created MuSync connection.

If the user chooses to host, then a MuSync network will get created and Media Player will open up, if the user chooses to join, then the app will search for available hosts and if there is any host available then user will get the host list and then the user can select the particular host from the list and the Media Player Screen will open up.
Post-Condition: The user must know how to switch off his WLAN network and must be able to access the app in future for further use.

5.1.2 Use Case Diagram:

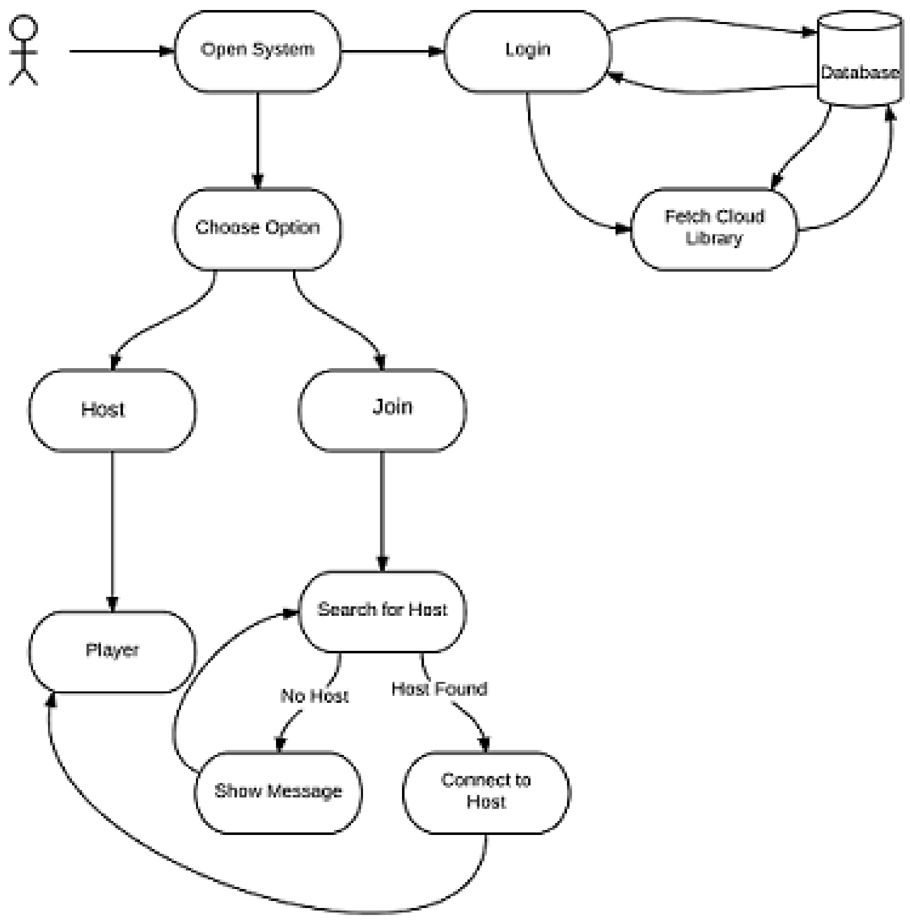


Figure 13. Use Case Diagram

5.1.3 Data Flow Diagram:

A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored. It mainly consists of three parts divided on the basis of its level:

- **a)** *Level '0' Data Flow Diagram (Context Diagram).*
- **b)** *Level '1' DFD.*
- **c)** *Level '2' DFD.*

Level '0' DFD or Context Diagram:

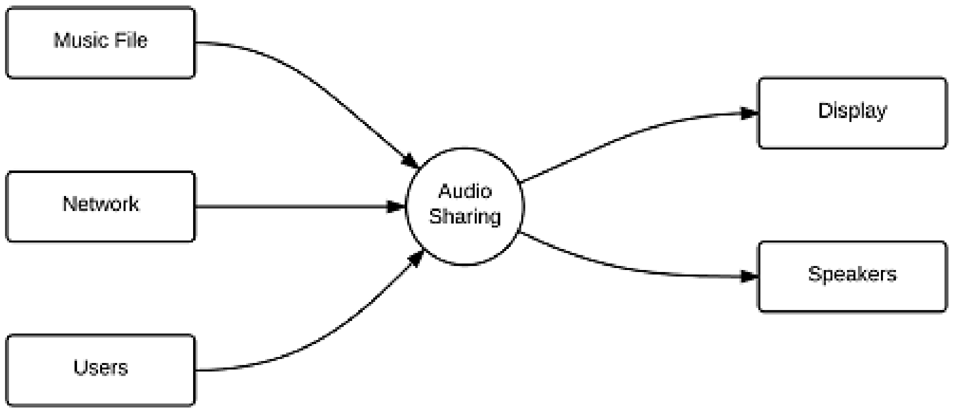


Figure 14. Level '0' DFD

Level '1' DFD:

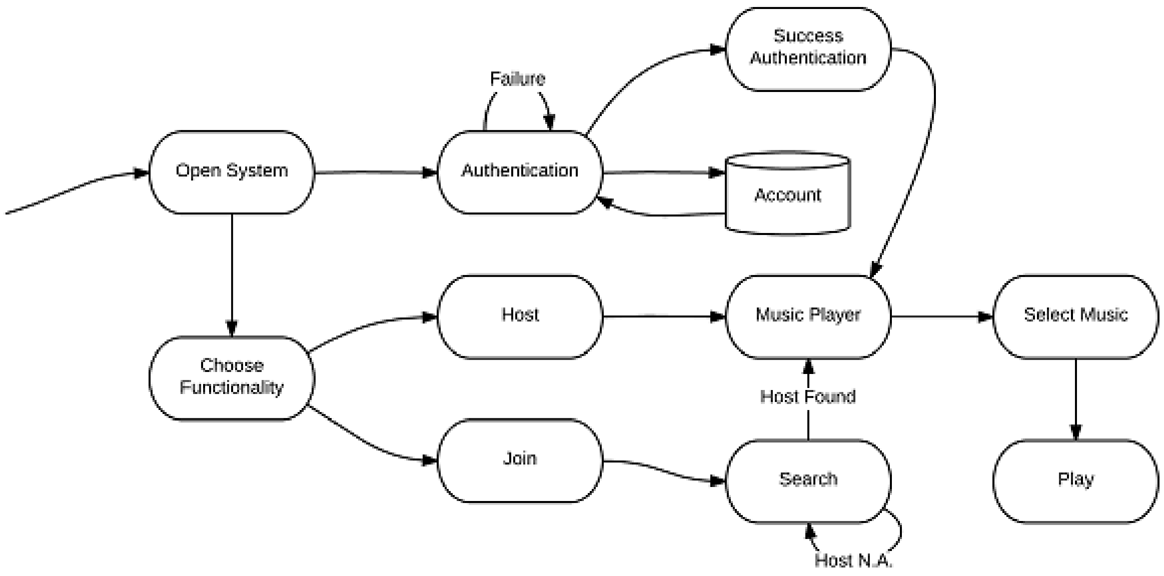


Figure 15. Level '1' DFD

Level ‘2’ DFD:

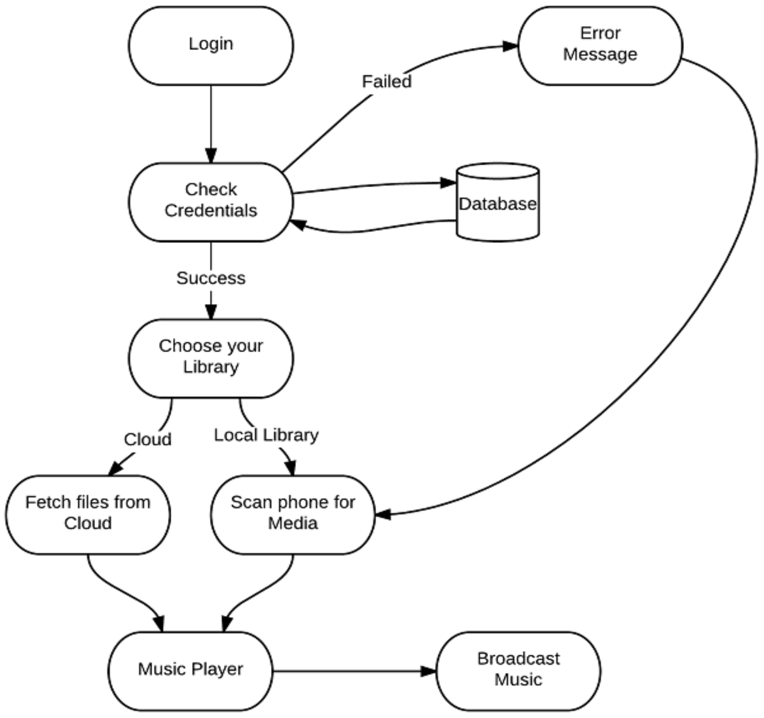


Figure 16. Level ‘2’ DFD

5.1.4 Sequence Diagrams:

Sequence diagrams in UML shows how object interact with each other and the order those interactions occur. It’s important to note that they show the interactions for a particular scenario. The processes are represented vertically and interactions are show as arrows.

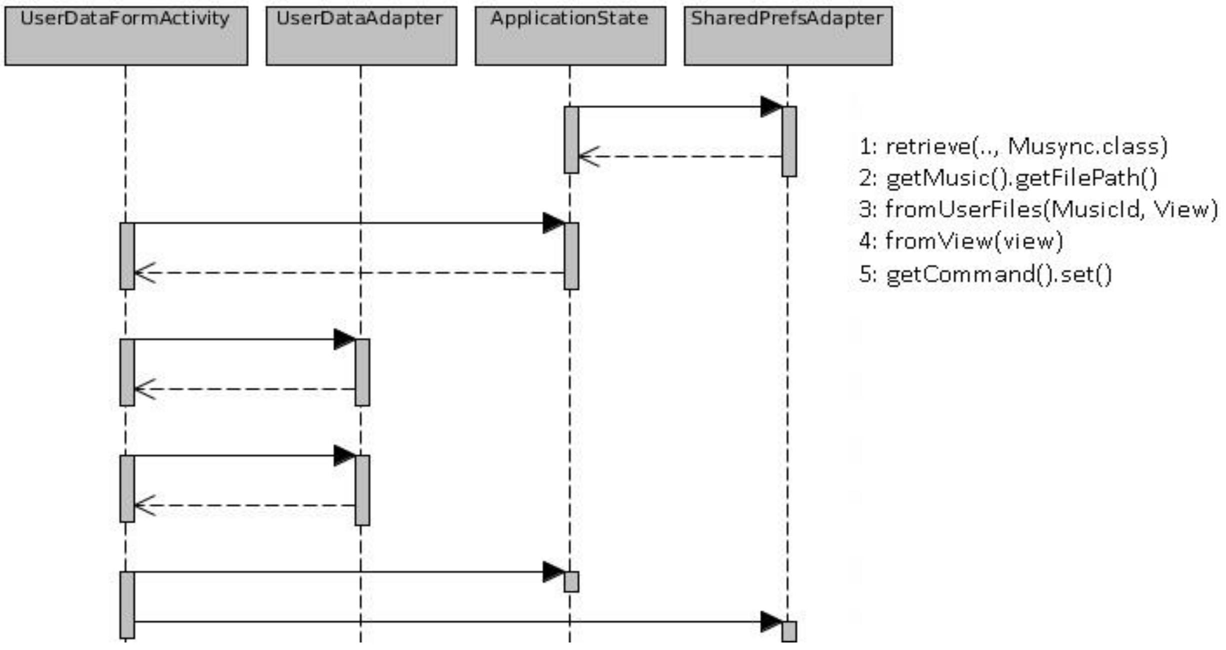


Figure 17. Sequence Diagram 1

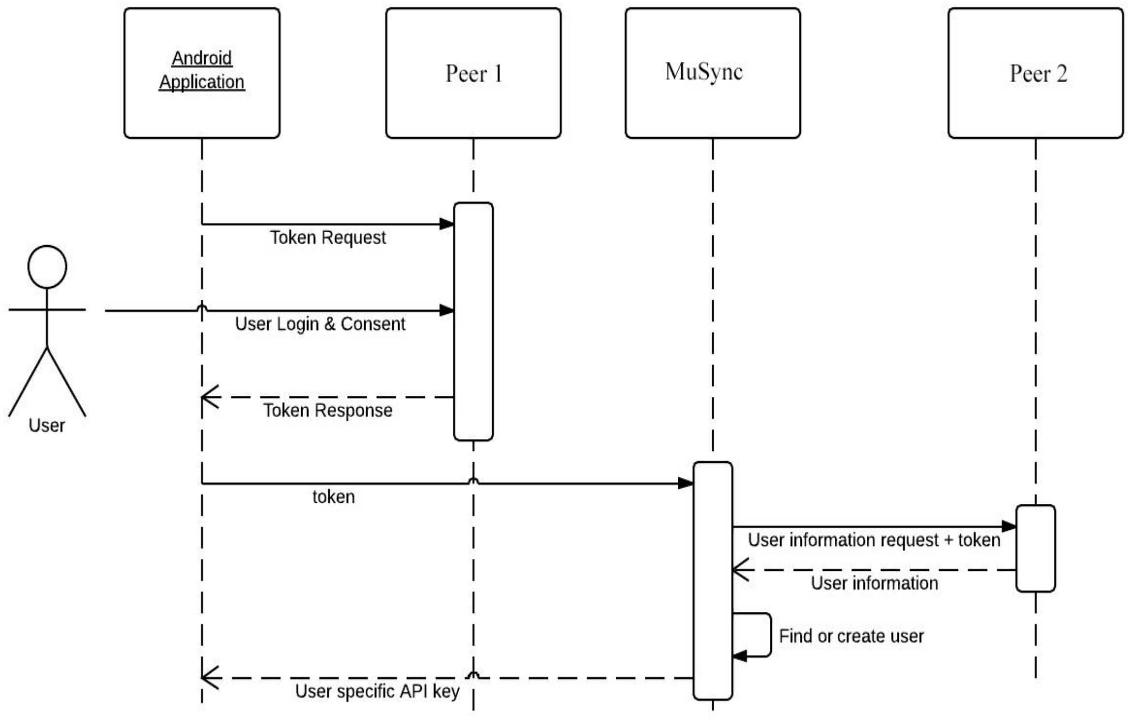


Figure 18. Sequence Diagram 2

5.1.5 Activity Diagram:

Activity diagrams represent workflows in a graphical way. They can be used to describe business workflow or the operational workflow of any component in a system. Sometimes activity diagrams are used as an alternative to State machine diagrams.

The initial step is user authentication that login by checking the credentials from the Database and after verification if valid then user can either access the data from the cloud or fetch it from the local library and then play it in the music player or else if the login attempt is unsuccessful then user will have to access only the local library and will be directed to music player to select a songs and host it.

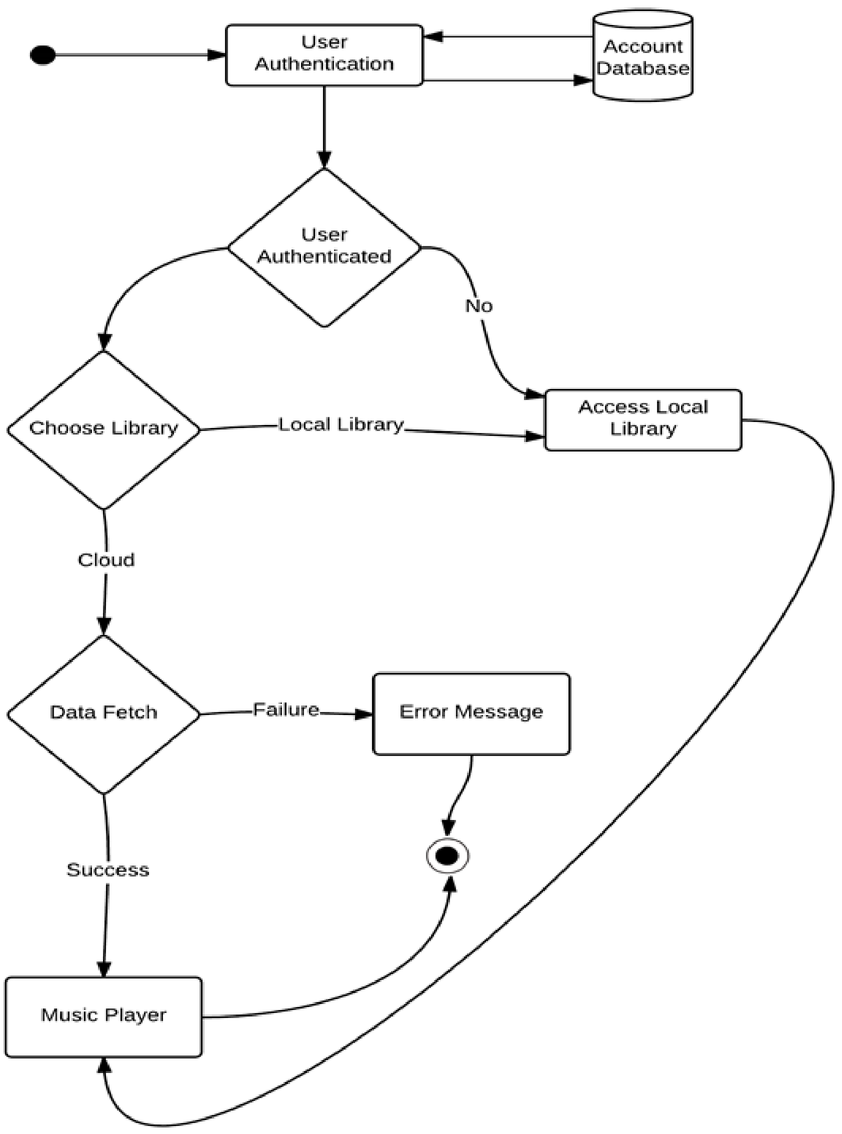


Figure 19. Activity Diagram

5.1.6 Class Diagram

Class diagrams are arguably the most used UML diagram type. It is the main building block of any object oriented solution. It shows the classes in a system, attributes and operations of each class and the relationship between each class.

In most modelling tools a class has three parts, name at the top, attributes in the middle and operations or methods at the bottom. In large systems with many related classes, classes are grouped together to create class diagrams. Different relationships between classes are shown by different types of arrows.

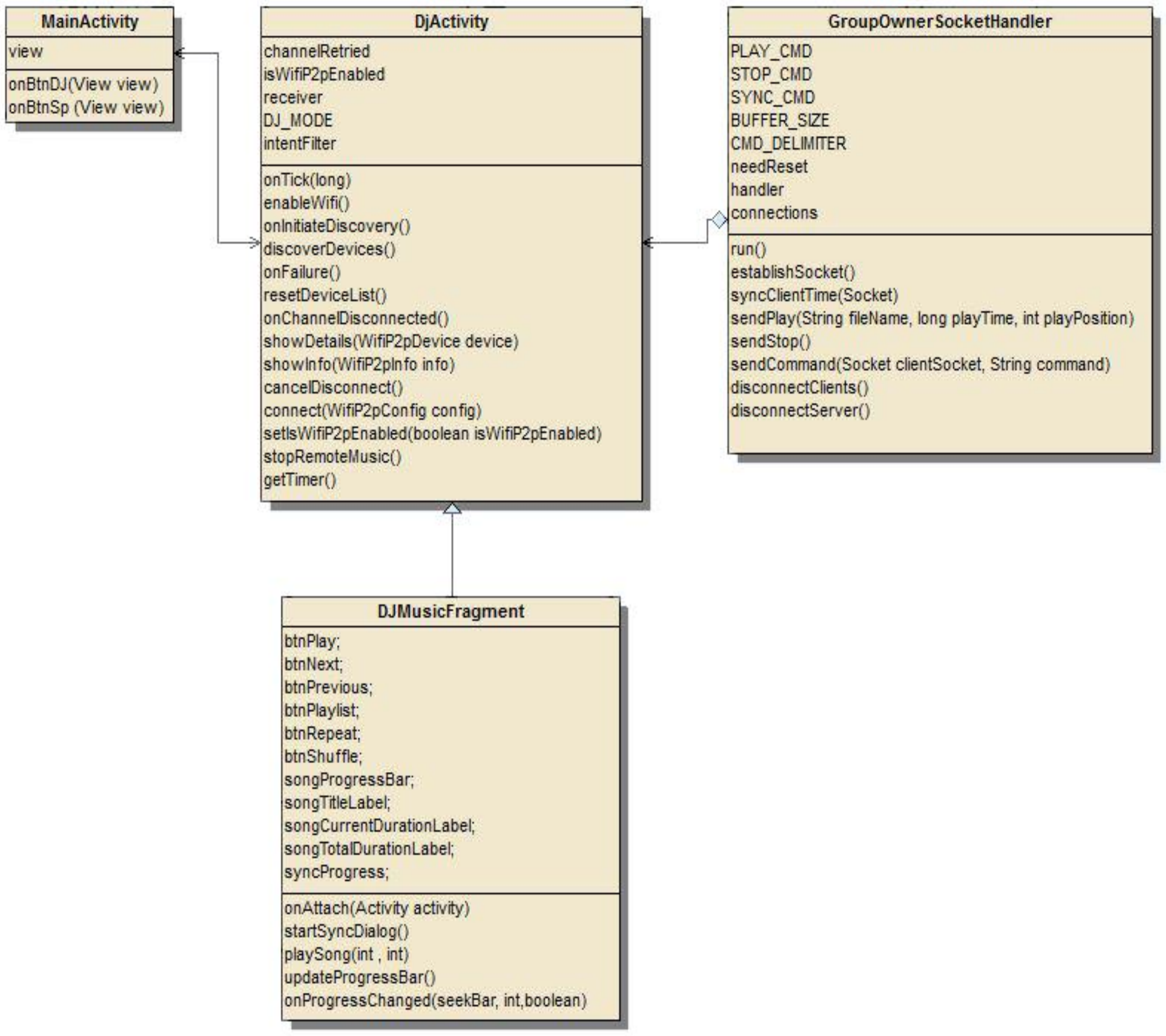


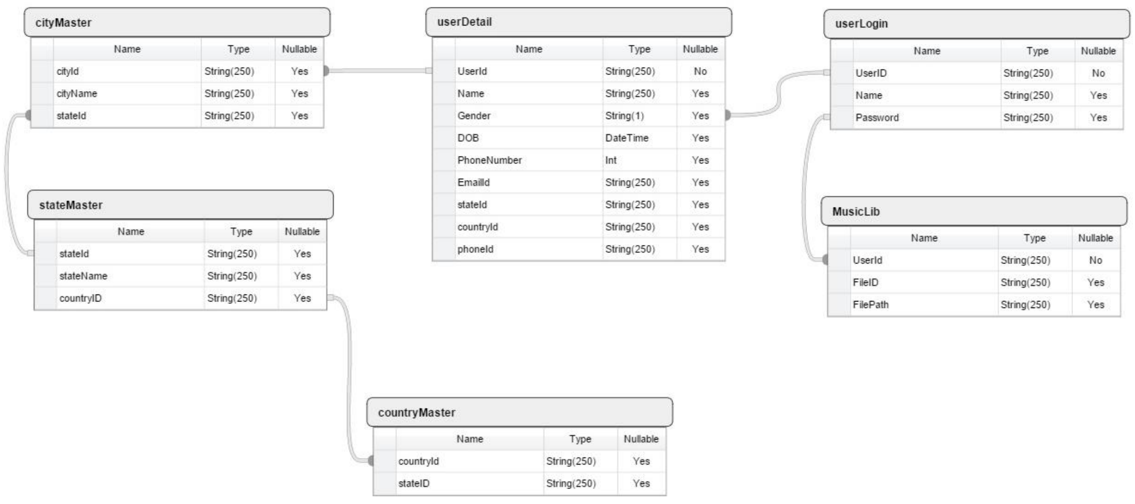
Figure 20. Class Diagram

5.1.7 Database Diagram

A database schema is a way to logically group objects such as tables, views, stored procedures etc. Think of a schema as a container of objects.

You can assign a user login permission to a single schema so that the user can only access the objects they are authorized to access.

Schemas can be created and altered in a database, and users can be granted access to a schema. A schema can be owned by any user, and schema ownership is transferable.



cityMaster

Name	Type	Nullable
cityId	String(250)	Yes
cityName	String(250)	Yes
stateId	String(250)	Yes

userDetail

Name	Type	Nullable
UserId	String(250)	No
Name	String(250)	Yes
Gender	String(1)	Yes
DOB	DateTime	Yes
PhoneNumber	Int	Yes
EmailId	String(250)	Yes
stateId	String(250)	Yes
countryId	String(250)	Yes
phoneId	String(250)	Yes

userLogin

Name	Type	Nullable
UserID	String(250)	No
Name	String(250)	Yes
Password	String(250)	Yes

stateMaster

Name	Type	Nullable
stateId	String(250)	Yes
stateName	String(250)	Yes
countryID	String(250)	Yes

MusicLib

Name	Type	Nullable
UserId	String(250)	No
FileID	String(250)	Yes
FilePath	String(250)	Yes

countryMaster

Name	Type	Nullable
countryId	String(250)	Yes
stateID	String(250)	Yes

Figure 21. Database Diagram

5.2 Bluetooth Based Wireless Touchpad Application

This application aims at developing a Wireless Touchpad that can be used in our day to day life. It is developed in a manner so it can be used by anyone from a world famous IT Company to an ordinary man who barely knows how to operate the computer. Therefore, to enhance the practicality and usability I have preferred it to be simple, efficient are reliable so that it doesn't face the lags in performance as the other competitors in the same space.

5.2.1 Use Case Modelling

Use Case: Wireless Touchpad

Primary Actor: User

Goal: To control all the functionalities of PC wirelessly by Bluetooth based Touchpad

Use Case Description:

Actor: User

Pre-Condition: The user smartphone must be connected to the Desktop Server.

The User can control all the functionalities of the Desktop by the wireless mouse which is via through the Bluetooth to the PC. The pre-requisite for the mouse to work, is a successful connection which could only be built once the Desktop Server is turned on. After the connection is successfully built one can take the full control of the Desktop by the wireless touchpad.

Post-Condition: The Desktop Server has to be turned off after usage. Thus, the devices will be disconnected.

5.2.2 Use Case Diagram:

a) For the Server Application (Desktop)

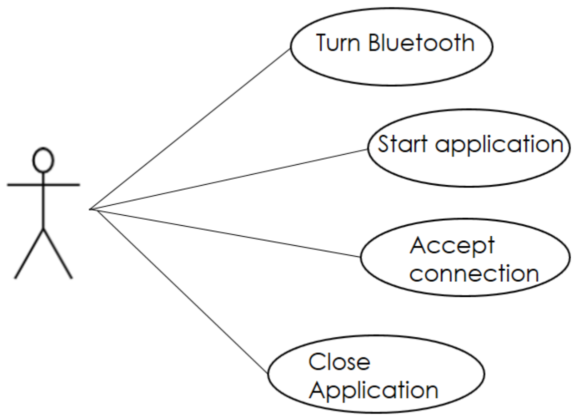


Figure 22. Use Case Diagram for the Server Application

b) For the Client Application (Smartphone)

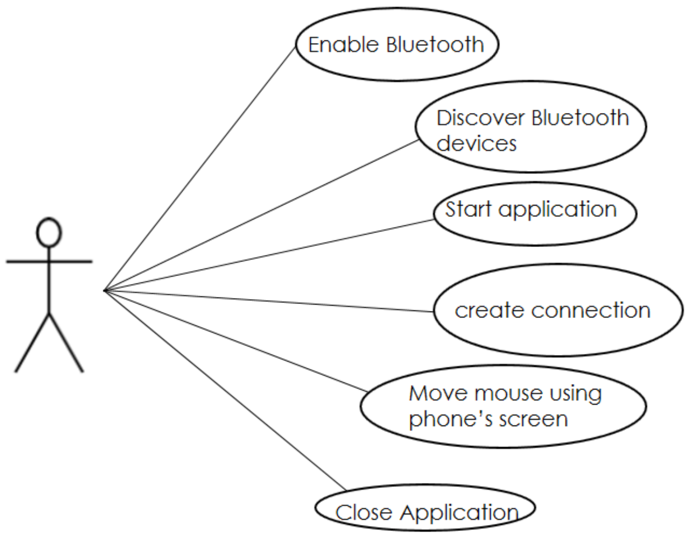


Figure 23. Use Case Diagram for the Client Application

5.2.3 Flow Chart:

When the server application is started, the state of the Bluetooth device is checked. When the request from the client application is accepted, then the RFCOMM communication takes place between the server and the client. The data is then used by the Robot library provide by Java to simulate the mouse movement and the keyboard press in the server. Finally, the Bluetooth socket connection is closed on termination of the application.

Below figure shows the steps that take place during the use of the application. When the application is started the availability of Bluetooth is checked. Before using this application, it should be kept in mind that not every Android device necessarily has Bluetooth in it. The application then scans the Bluetooth device and lists all the available devices as a list. The data received when a user touches the screen and types in the keyboard of the client are sent to the server. Finally, when a user closes the application Bluetooth is turned off closing the RFCOMM connection and the application terminates

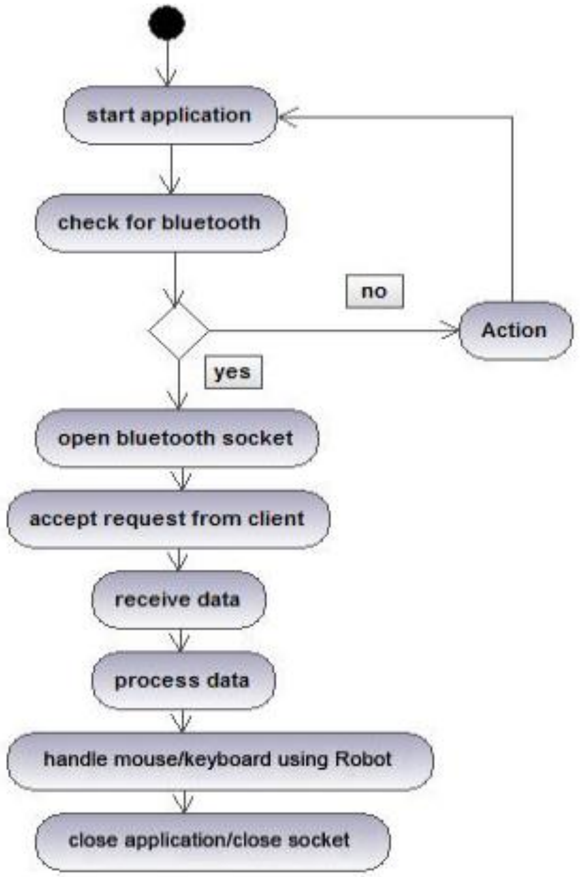


Figure 24. Flow Chart Wireless Touchpad Application

5.2.4 Sequence Diagram:

The sequence diagram is an interaction diagram that shows in what order and how processes are carried out. The classes contained in each application are somehow interconnected with each other. A class might call a function in another class to do a specific task or can send data to another class asking it to perform some tasks. Figure below shows the sequence diagram for the application.

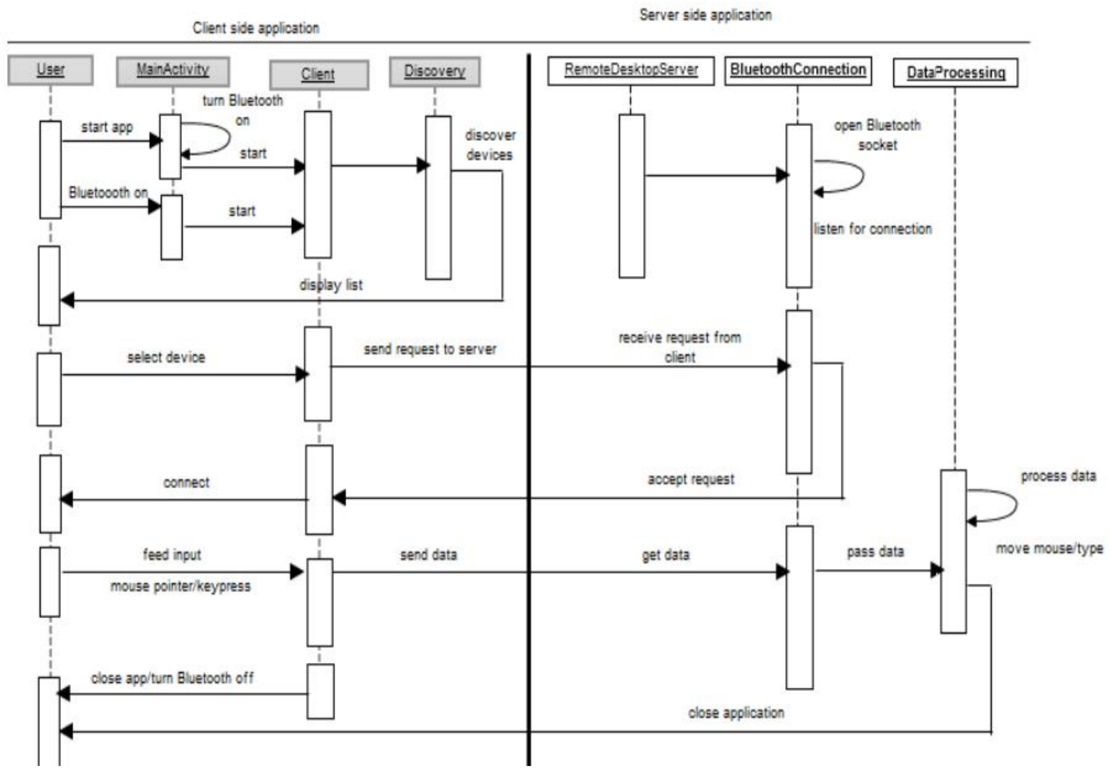


Figure 25. Sequence Diagram of Wireless Touchpad Application

5.2.5 Class Diagram:

The server application consists of three classes, RemoteBluetoothServer, BluetoothConnection and DataProcessing. RemoteBluetoothServer contains the main function and is the starting class of the application. This class calls the BluetoothConnection class where functions for opening the Bluetooth socket and connecting to the client are coded. The data is then passed to the DataProcessing class from the BluetoothConnection class. In the DataProcessing class all the data are processed and then handled by the Robot API. The movement of the mouse and the 38 keyboard pressed are all controlled by the Robot. Finally on terminating the application the Bluetooth socket is closed for the connection.

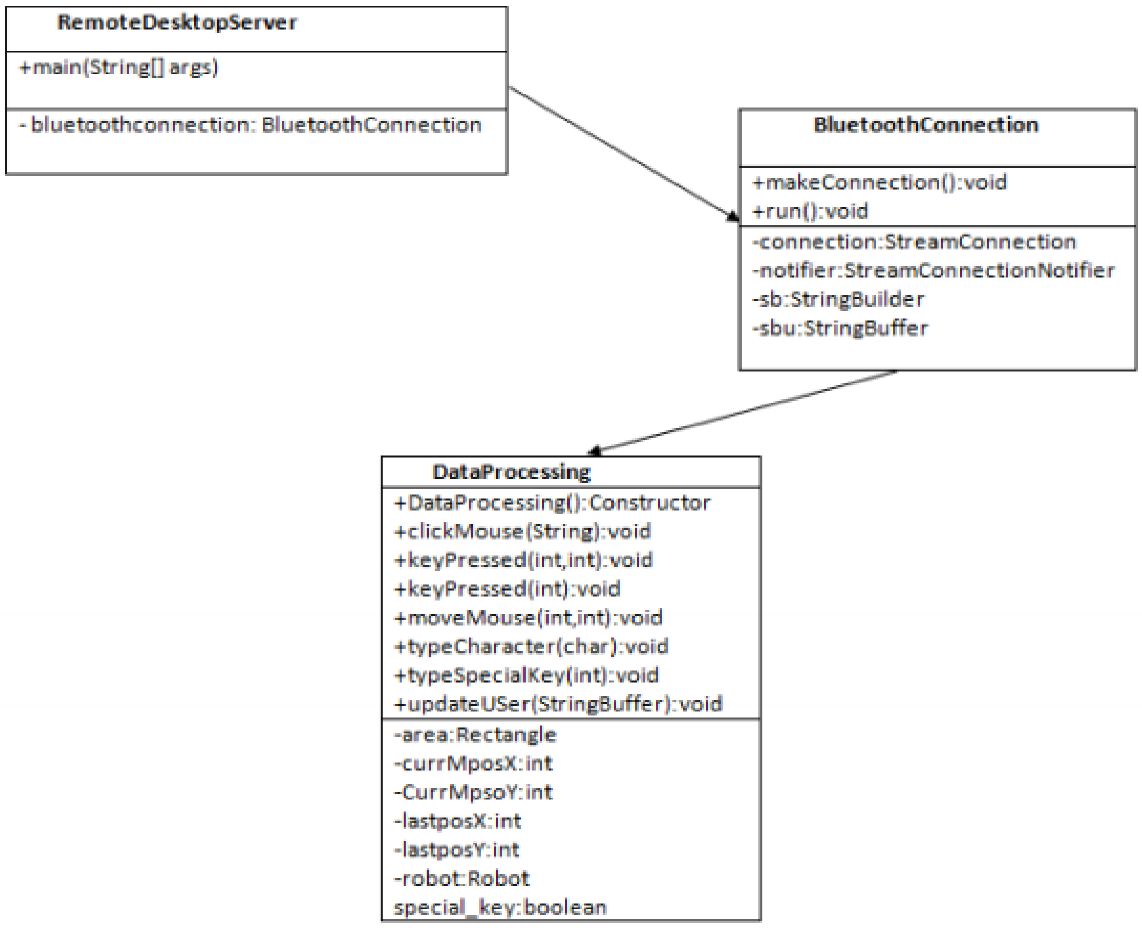


Figure 26. Class Diagram for Wireless Touchpad Server Application

The client application consists of three classes, MainActivity, Client and Discovery. MainActivity is the starting activity which starts the Bluetooth service in the application. This class then calls the Client class. Before doing anything the Client class calls the Discovery class which will scan the Bluetooth devices and list all the available devices. The Discovery class lies on the top of the view with the Client class still being 34 semi-visible. On selecting the Bluetooth device focus comes back to the Client class which will then make the Bluetooth connection and send the data to the server. On pressing the back button in an Android device the application is terminated. During the termination function like closing the Bluetooth socket connection, turning off the Bluetooth and closing the application are all handled by the Client class.

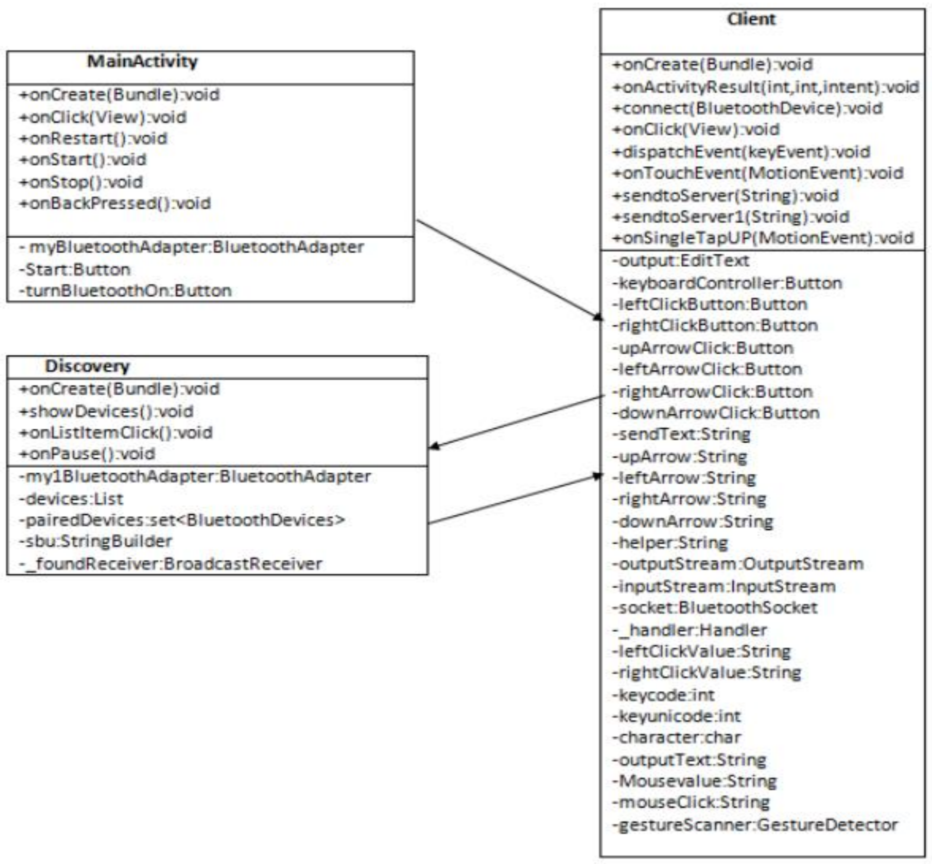


Figure 27. Class Diagram for Wireless Touchpad Client Application

5.3 Usefulness and Innovation

This software helps us to share our music without sharing the music files with our peers and enjoy loud music. After the particular library is loaded, user can either choose to become host or join an already created MuSync connection. If the user chooses to host, then a MuSync network will get created and Media Player will open up, if the user chooses to join, then the app will search for available hosts and if there is any host available then user will get the host list and then the user can select the particular host from the list and the Media Player Screen will open up. The WLAN connection upgrades the performance. It increases the bandwidth and improves the access latency.

In the another application i.e. Bluetooth Based touchpad application, I’m trying to reduce the latency and the performance barrier that the current application is facing. Further the application is Bluetooth based which will improve the performance and reliability.

Chapter 6: Project Development

In this Chapter, will be discussing about the implementation of our project, software development methodology, code and algorithm used during development of our project and the reason for choosing WLAN and Bluetooth for the corresponding applications.

6.1 Software Development Model

In the development of both the applications, I have used the incremental technique. This is the most appropriate because in incremental model the whole requirement is divided into various builds and thus, multiple development cycles take place here, making the life cycle a "multi-waterfall" cycle. Cycles are divided up into smaller, more easily managed modules. Each module passes through the requirements, design, implementation and testing phases. A working version of software is produced during the first module, so client have working software early on during the software life cycle. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved.

Figure 28: Overview of Incremental Model

The incremental development model generates the working software quickly and early during the software life cycle and this model being flexible is less costly to change scope and requirements which makes it easier to test and debug during a smaller iteration. In addition, it lowers initial delivery cost and is easier to manage risk because risky pieces are identified and handled during its iteration.

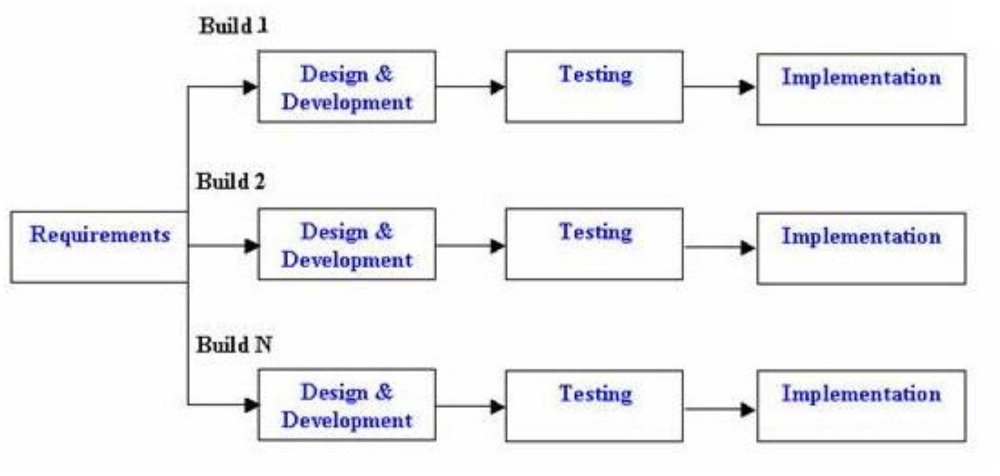


Figure 29. Incremental Development Model

6.2 Why Incremental Model?

Incremental model is used because the requirements of the complete system are clearly defined and understood and the major requirements are defined; however, some details can evolve with time. Moreover, a new technology is being used and the project requires a proper research and implementation of the technologies that are required to be learnt and acquired.

6.3 MuSync Application Development

The application is based on the Wi-Fi live streaming, in which a user has choose to be host or a client. The project has been divided into two categories, either a host which is termed as DJ Mode and a client i.e. Speaker Mode. I have designed the Music Player fragment, List of Connected Devices, P2P Settings etc. DJ mode has access to Music Player Fragment and List of Connected Devices fragment whereas Speaker Mode has access to Volume and Connected Peer Fragments. One can control Peer to Peer Connection from the application interface itself.

6.3.1 Wi-Fi Peer-to-Peer

Wi-Fi peer-to-peer (P2P) allows Android 4.0 (API level 14) or later devices with the appropriate hardware to connect directly to each other via Wi-Fi without an intermediate access. Using these APIs, one can discover and connect to other devices when each device supports Wi-Fi P2P, then communicate over a speedy connection across distances much longer than a Bluetooth connection. This is useful for applications that share data among users, such as a multiplayer game or a photo sharing application.

The Wi-Fi P2P APIs consist of the following main parts:

- Methods that allow user to discover, request, and connect to peers are defined in the WifiP2pManager class.
- Listeners that allow the user to be notified of the success or failure of WifiP2pManager method calls. When calling WifiP2pManager methods, each method can receive a specific listener passed in as a parameter.
- Intents that notify the user of specific events detected by the Wi-Fi P2P framework, such as a dropped connection or a newly discovered peer.

6.3.2 Pseudo Code for the Wi-Fi Connection

Connecting as a Server:

Once a connection is established, one can transfer data between the devices with sockets. The following code is to communicate via Wi-Fi.

```
public static class File ServerAsyncTask extends AsyncTask{

        private Context context;
              private TextView statusText;

 public FileServerAsyncTask(Context context, View statusText) {
    this.context = context;
    this.statusText = (TextView) statusText;
 }

 @Override
 protected String doInBackground(Void... params) {
    try {
            ServerSocket serverSocket = new ServerSocket(8888);
            Socket client = serverSocket.accept();

            final File f = new File(Environment.getExternalStorageDirectory() + "/"
                    + context.getPackageName() + "/wifip2pshared-" + System.currentTimeMillis()
                    + ".jpg");

            File dirs = new File(f.getParent());
            if (!dirs.exists())
                dirs.mkdirs();
            f.createNewFile();
            InputStream inputstream = client.getInputStream();
            copyFile(inputstream, new FileOutputStream(f));
            serverSocket.close();
            return f.getAbsolutePath();
        } catch (IOException e) {
            Log.e(WiFiDirectActivity.TAG, e.getMessage());
            return null;
        }
    }
```

Connecting as a Client

On the client, connect to the server socket with a client socket and transfer data. This code transfers a JPEG file on the client device's file system.

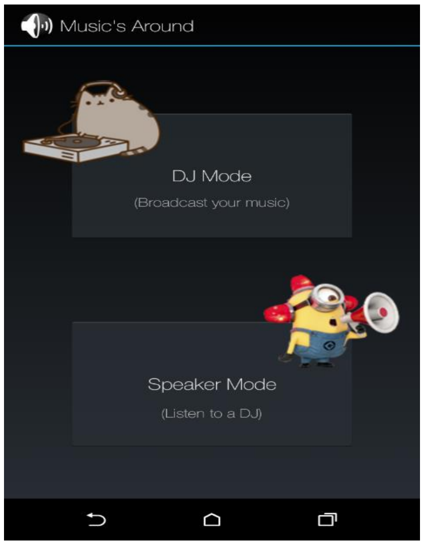
```
Context context = this.getApplicationContext();
String host;
int port;
int len;
Socket socket = new Socket();
byte buf[]  = new byte[1024];
...
try {
    socket.bind(null);
        socket.connect((new InetSocketAddress(host, port)), 500);

       OutputStream outputStream = socket.getOutputStream();
        ContentResolver cr = context.getContentResolver();
        InputStream inputStream = null;
        inputStream = cr.openInputStream(Uri.parse("path/to/picture.jpg"));
        while ((len = inputStream.read(buf)) != -1) {
            outputStream.write(buf, 0, len);
        }
        outputStream.close();
        inputStream.close();
    } catch (FileNotFoundException e) {
        }

finally {
    if (socket != null) {
        if (socket.isConnected()) {
            try {
                socket.close();
            } catch (IOException e) {
            }
        }
```

6.3.3 Screenshots

With the screen shorts which are attached below one can clearly examine the User Interface and the working of the application. Therefore, some of the screen shorts of the applications are attached below:

a) Main Activity

As we can see in the screen short below (Figure 29) the user can select from two options provided. Either he can choose to be a server/DJ or a client. All the connected clients will be playing the music played by the server.



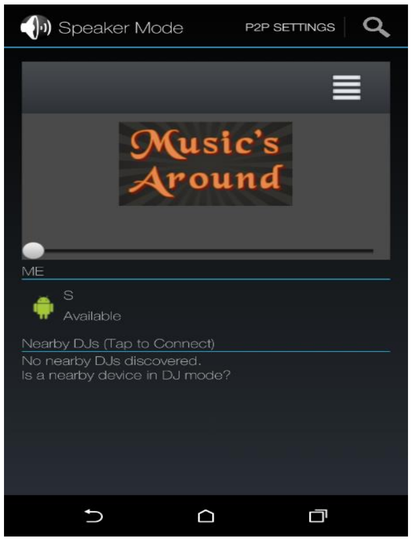

Speaker Mode
P2P SETTINGS
Music's Around
ME
S
Available
Nearby DJs (Tap to Connect)
No nearby DJs discovered.
Is a nearby device in DJ mode?

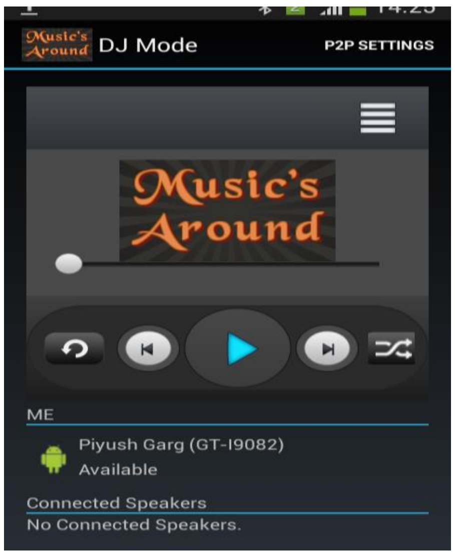
Figure 30. Main Activity Figure 31. Client Activity

b) Client Activity

Client is provided with two functionalities (Figure 30). Either he can control the volume or see the list of all the connected devices. These two fragments are same as to that of the host.

c) Music Player Fragment

This Music player fragment is to added into the Server Activity to play the music on all the connected clients, but Music Player has certain error due to which the activity stops.



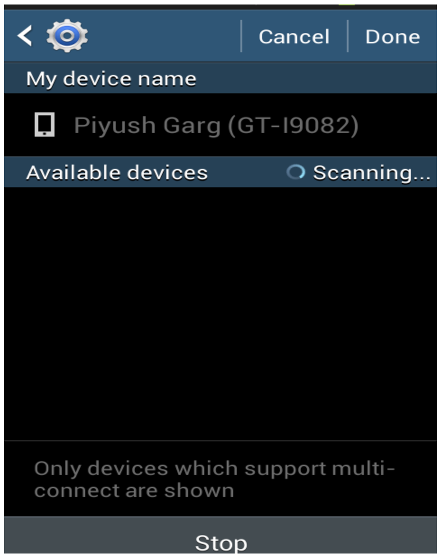


Figure 32. Music Player Figure 33. Creation of peer group

d) Creation of peer group

I have provided the functionality of creating a group so that each time user doesn't need to connect each and every device as shown in figure 32. One can add more peers anytime, if they wish.

e) Searching for the pees

The below screen short i.e. figure 33 shows the option of searching the nearby peer so that streaming could be made possible.

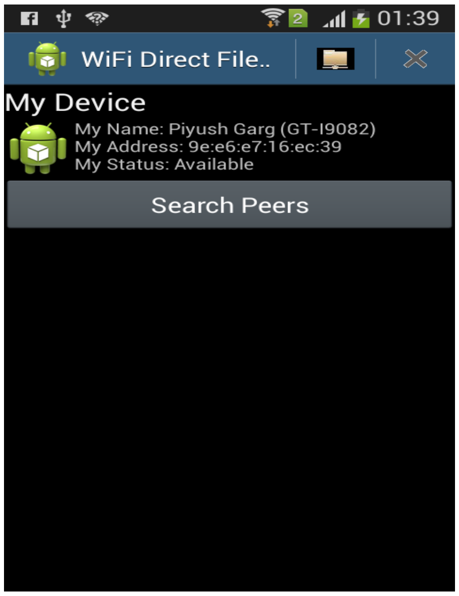

WiFi Direct File..
My Device
My Name: Piyush Garg (GT-I9082)
My Address: 9e:e6:e7:16:ec:39
My Status: Available
Search Peers

Figure 34. Search peers

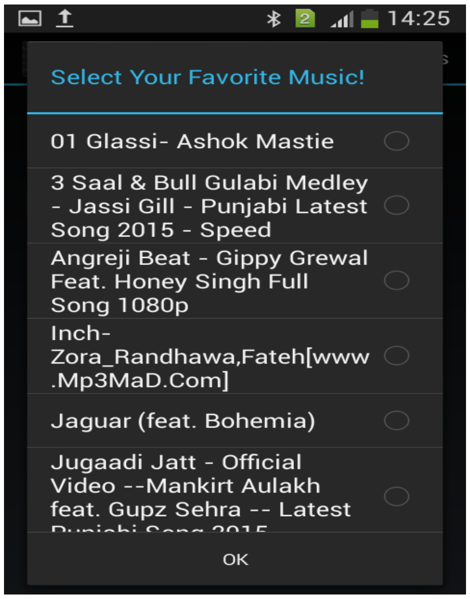

Select Your Favorite Music!
01 Glassi- Ashok Mastie
3 Saal & Bull Gulabi Medley - Jassi Gill - Punjabi Latest Song 2015 - Speed
Angreji Beat - Gippy Grewal Feat. Honey Singh Full Song 1080p
Inch-Zora_Randhawa,Fateh[www.Mp3MaD.Com]
Jaguar (feat. Bohemia)
Jugaadi Jatt - Official Video --Mankirt Aulakh feat. Gupz Sehra -- Latest
OK

Figure 35. Playlist to select music

e) Playlist to Select Music

A complete music Player is provided and one can select the desire music as shown in the above figure 34.

6.4 Bluetooth Touchpad Wireless Mouse

The main goal of the project was to develop a remote control system for computers in an Android platform using Bluetooth 4.0 and Java. The client and server applications would enable Bluetooth connection between the phone and computer, and once connected the applications would give the user full control of the computer’s mouse.

The Android platform includes support for the Bluetooth network stack, which allows a device to wirelessly exchange data with other Bluetooth devices. The application framework provides access to the Bluetooth functionality through the Android Bluetooth APIs. These APIs let applications wirelessly connect to other Bluetooth devices, enabling point-to-point and multipoint wireless features.

Using the Bluetooth APIs, an Android application can perform the following:

- Scan for other Bluetooth devices
- Query the local Bluetooth adapter for paired Bluetooth devices
- Establish RFCOMM channels
- Connect to other devices through service discovery
- Transfer data to and from other devices
- Manage multiple connections

6.4.1 Pseudo Code for the Bluetooth Communication

Connecting as a server

When one wants to connect two devices, 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 (and should) be discarded, unless the user want to accept more connections.

Here's a simplified thread for the server component that accepts incoming connections:

```
private class AcceptThread extends Thread {
    private final BluetoothServerSocket mmServerSocket;

    public AcceptThread() {
           BluetoothServerSocket tmp = null;
        try {
            tmp = mBluetoothAdapter.listenUsingRfcommWithServiceRecord(NAME, MY_UUID);
        } catch (IOException e) { }
        mmServerSocket = tmp;
    }
    public void run() {
        BluetoothSocket socket = null;
        while (true) {
            try {
                socket = mmServerSocket.accept();
            } catch (IOException e) {
                break;
            }
            if (socket != null) {
                manageConnectedSocket(socket);
                mmServerSocket.close();
                break;
            }
                mmServerSocket.close();
        } catch (IOException e) { }
    }
```

Connecting as a client

In order to initiate a connection with a remote device (a device holding an open server socket), one must first obtain a BluetoothDevice object that represents the remote device. (Getting a BluetoothDevice is covered in the above section about Finding Devices.) One must then use the BluetoothDevice to acquire a BluetoothSocket and initiate the connection.Here is a pseudo code of a thread that initiates a Bluetooth connection:

```
private class ConnectThread extends Thread {
```

```
private final BluetoothSocket mmSocket;
private final BluetoothDevice mmDevice;

public ConnectThread(BluetoothDevice device) {
   BluetoothSocket tmp = null;
   mmDevice = device;
    try {          tmp = device.createRfcommSocketToServiceRecord(MY_UUID);
   } catch (IOException e) { }
   mmSocket = tmp;
}

public void run() {
   mBluetoothAdapter.cancelDiscovery();

   try {          mmSocket.connect();
   } catch (IOException connectException) {

      try {             mmSocket.close();
      } catch (IOException closeException) { }
      return;
   }
   manageConnectedSocket(mmSocket);
}
public void cancel() {
   try {
      mmSocket.close();
   } catch (IOException e) { }
```

6.4.2 Screenshots of the Server Application

a) Main Java Server Application

The initial page of the Java Server Application is as shown below. The Java Server identifies the Desktop's Bluetooth device and if the devices are more than one, the user can select from the list. The Server can be initialized my tapping "Start Listening" button.

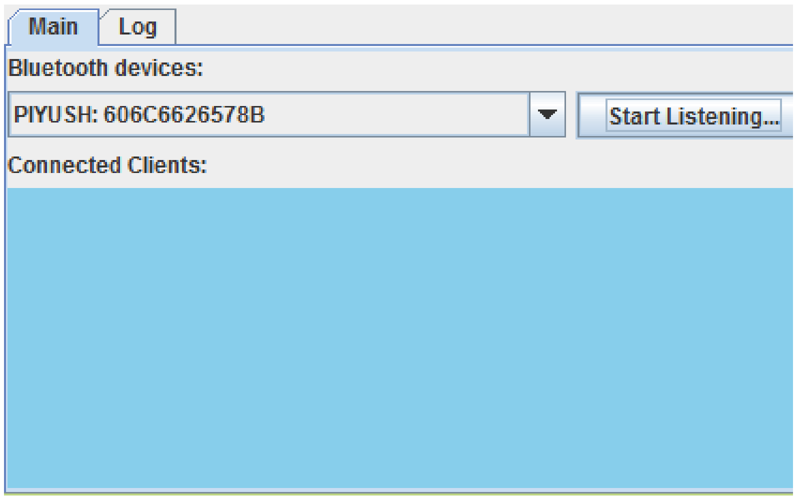


Figure 36. Before initiating Java Server

b) Java Server After Initialization

The Java Server is initialized after tapping the Start Button. Now the users can search the Server via the Android Smartphones and get connected to the server. Once the connection is built, the Android Phone will work as the wireless touchpad.

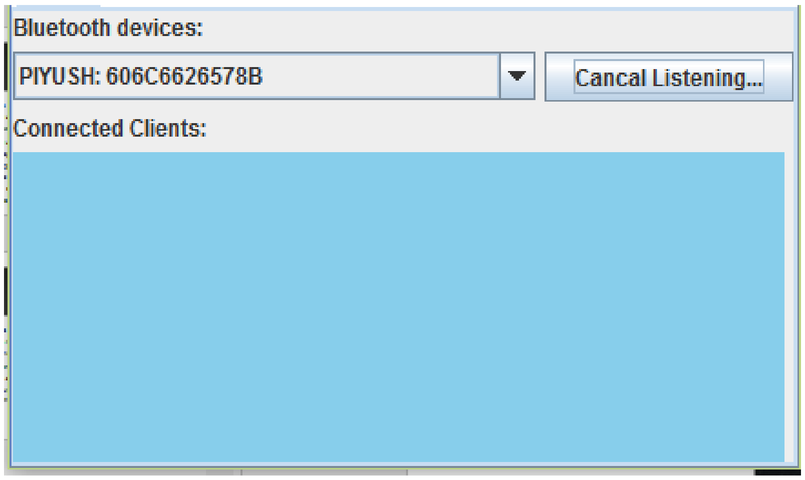
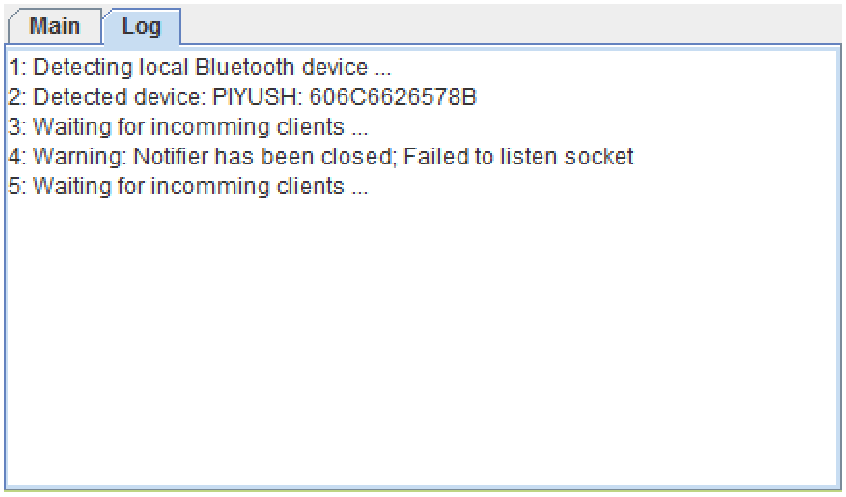


Figure 37. After java server has been initiated

c) Log to describe the current operations

The log window to display the current activities is attached below. It will show all the updated and error messages such as Successful connection, Connected device name, number of clients, connection failure etc.



```
1: Detecting local Bluetooth device ...
2: Detected device: PIYUSH: 606C6626578B
3: Waiting for incomming clients ...
4: Warning: Notifier has been closed; Failed to listen socket
5: Waiting for incomming clients ...
```

Figure 38. Log to describe the current operations

6.4.3 Screenshots of the Client Application

a) Main Activity

This is the initial page of the application, which gives the option to connect to the windows server via Bluetooth. If the user taps on the Bluetooth Symbol, then will be directed to the search activity as shown in the screen short towards right.

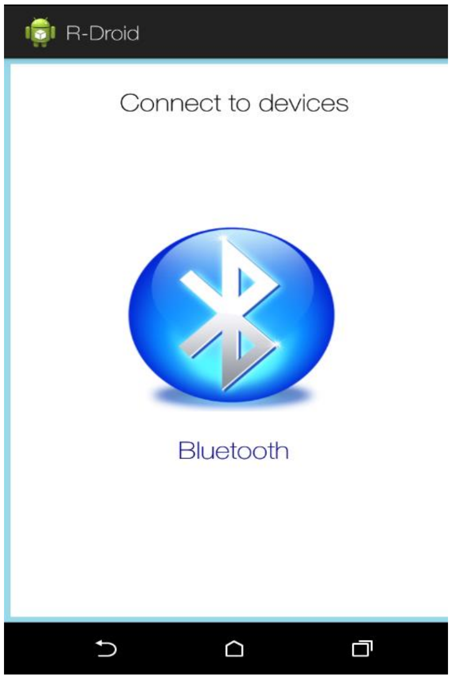


Figure 39. Main Activity

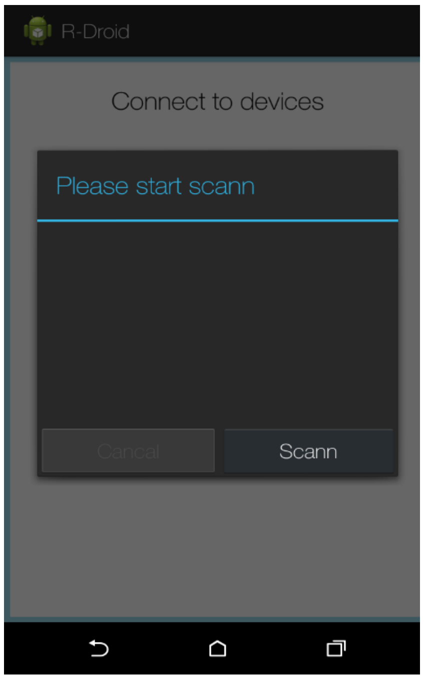


Figure 40. Scanning for available devices

b) Scanning the devices

This is the screen short of the second page where the user is scanning the available servers. The user can choose any of the server from the list and connect to it.

c) List of available devices

Here's the list of the all the Bluetooth devices available and we can connect to the adequate device which is to be monitored by the android smartphone.

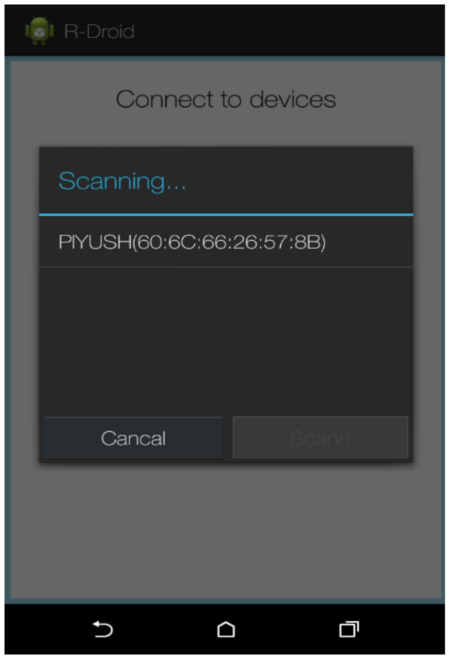


Figure 41. List of available devices

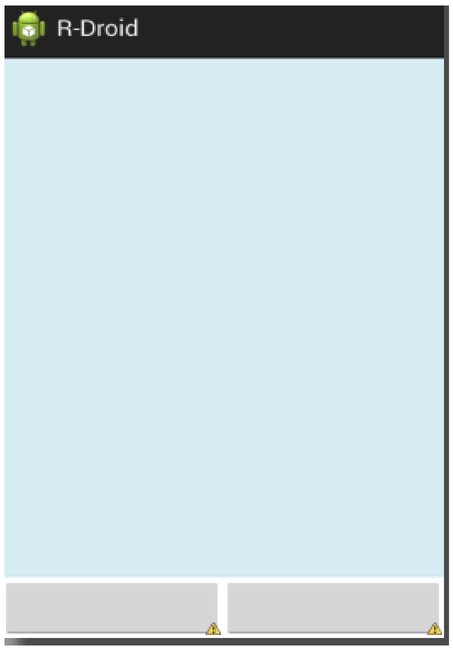


Figure 42. Virtual Touchpad mouse

d) Virtual Touchpad

Screen short of the virtual touchpad which is equipped with two button both left and right, scroll and double click functionalities etc. all are functional in this application.

Both the projects have been implemented successfully and are working as intended. The analysis is done in the next chapter.

Chapter 7: Performance Analysis

In this chapter we are testing the application on the different phones and laptops. Furthermore, the applications were given to my colleagues for testing and the feedback was received, on the basis of which the Test Report, Result and the Conclusion has been deduced.

Testing of the project was done by using the applications in different phones and computers. While testing on the different Laptops, Samsung Note 3 phone was kept the same, whereas the Laptops were changed.

7.1 Testing of the Wi-Fi based MuSync application

The application has been developed successfully using the API 17 i.e. Android 4.2. It is working fine for approximately 18 meters without any drop in quality or latency issues. After this range there was some degradation in performance and the application is best to be used within 20 meters of distance between two peers. The application was tested on the various smartphones at various distances from the host. The host is placed at a certain position and the distance of all the phones are measured taking the host as a reference.

The application can connect maximum of 18 peers yet we were only able to test it for maximum of 6 clients and 1 host, for which the application worked absolutely fine without any bugs or issues. The host smartphone used is Galaxy Grand, whereas the clients were One plus one, galaxy note 3, micromax canvas, Samsung S3, HTC 816g etc.

The host is working fine for API 18 or below after which the application stops working for the new updates. Therefore, the host preferred is grand as it supports 4.2.2 which is API 17.

The table below shows the performance of the application on the various smartphones and at certain distances. The observations are shown in Table 1.

Table 1: Test Cases for Wi-Fi Application

Host Device	**Peer 1**	**Peer 2**	**Peer 3**	**Peer 4**	**Peer 5**	**Peer 6**	**Using the Application**	**Rang e (1-15**	**Range (15-20 m)**	**Range (more than 20)**
Samsun g Grand (4.2.2)	Samsun g Note 3	-	-	-	-	-	Works Fine	Yes	Yes	Yes
Samsun g Grand (4.2.2)	Samsun g Note 3	One plus one	-	-	-	-	Works Fine	Yes	Yes	Yes
Samsun g Grand (4.2.2)	Samsun g Note 3	One plus one	Microm ax Canvas 2	-	-	-	Works Fine	Yes	Yes	fluctuation
Samsun g Grand (4.2.2)	Samsun g Note 3	One plus one	Microm ax Canvas 2	HTC 816g	-	-	Works Fine	Yes	Yes	Yes
Samsun g Grand (4.2.2)	Samsun g Note 3	One plus one	Microm ax Canvas	HTC 816g	Samsung S2	-	Works Fine	Yes	Yes	fluctuation
Samsun g Grand	Samsun g Note 3	One plus one	Microm ax	HTC 816g	Samsung S2	Note 4	Works Fine	Yes	Yes	Yes
One plus One (6.0)	Samsun g Note 3	Samsun g Galaxy	Microm ax Canvas	HTC 816g	Samsung S2	Note 4	Doesn’t Work	No	No	No

7.2 Testing of the Bluetooth Based Touchpad application

The Android application of this project cannot be used for Android phones with Android version earlier than 2.0. It should be kept in mind that both the server and the client must have properly functioning Bluetooth before testing the project. Also the server and the client must be within the range depending on the version of Bluetooth available in the phone. The maximum working range of recently designed Bluetooth devices are 100 meters. Keeping these facts in mind the testing was done. The test cases are as follows. Table 2 shows the different test cases done for the client application.

Table 2: Test cases for the client application

Devices	**Application's Compatibility**	**Bluetooth Connection Type**	**Using the App**	**Range (1m to 10m)**	**Range (10m to 15m)**	**Range (15m to 20m)**
Samsung Note 4	Yes	4.0	Works fine	Yes	Yes	fluctuation
HTC 816g	Yes	4.0	Works fine	Yes	Yes	fluctuation
Samsung S2	Yes	3.0	Works fine	Yes	fluctuation	No
One Plus Two	Yes	4.0	Works fine	Yes	Yes	fluctuation
Samsung Note 3 Neo	Yes	4.0	Works fine	Yes	Yes	fluctuation
Micromax Canvas 2	Yes	4.0	Works fine	Yes	Yes	fluctuation

Table 2 illustrates the different test cases conducted by using different Android phones. First of all, the server was run on the same computer and the client application was tested using different Android phones such as HTC 816g, Samsung Note 4, Samsung Galaxy Note II and Micromax Canvas 2 and One Plus Two. Regarding the compatibility, there were no problems encountered in running the application on different phones. Using the application in different phones was easy and almost the same as they all had almost the same screen sizes, so there was no alteration in the user interface of the client application. While testing the application by putting the server and the client within a certain range, 10-20 meters, there was an interruption encountered. The connection was weak when the distance was more than 17 meters. The phones were having difficulty connecting to the server. This project worked well within the range of 17-18 meters between the server and the client. Table 2 shows the different test cases conducted for the server application in different laptops.

Table 3. Different test cases for the server application in different laptops

Devices	**Application's Compatibility**	**Bluetooth Connection Type**	**Using the App**	**Range (1m to 10m)**	**Range (10m to 15m)**	**Range (15m to 20m)**
Dell (5521)	Yes	4.0	Works fine	Yes	Yes	fluctuation
Sony (VPCSE13FD)	Yes	4.0	Works fine	Yes	Yes	fluctuation
Dell (5510)	Yes	3.0	Works fine	Yes	fluctuation	fluctuation
HP Pavilion g6	Yes	4.0	Works fine	Yes	Yes	fluctuation

Table 3 illustrates different test cases conducted by using different laptops for the server application. The server application was run on different computers and the project was tested by using the Samsung Galaxy Note 3 as the client. This project worked well in all the computers and there was no problem regarding the issues mentioned in the test cases when the range was within 10 meters. In aggregate this project worked well in all devices within some range. There was no problem in running the server application in the computers. However, the client application crashed a few times during the use. Also the project was given to different users for testing purposes. The users found the applications to be simple and easy to use.

7.3 Results

The main goal of the project was to work into the Android domain, especially the communication as there’s still a lot of potential in this space. Both the project shave been implemented successfully and there have been certain bugs and limitations that I encountered during the testing of both the applications. I will try to overcome them in the next update.

The MuSync Application is designed in the API 17 i.e. Android version 4.2 and works well till API 18 i.e. 4.3. In the newer it is not possible to create a host, but the peers work well as clients. The application worked well up to 6 peers connected at a time and there was minimum lag till 20m of distance between peer and host. The application worked absolutely fine till 15m yet a minimal lag at around 18m. There were fluctuations in performance after 20m and stopped working at around 27-28m of distance between host and client. I tried to use one plus one as a host but it didn’t work as the application is not compatible with Marshmallow i.e. Android 6.0, thus need to be worked upon. Therefore, for this project I have used Samsung Galaxy Grand as host as it is empowered by Android 4.2.2, on which the application works fine.

remote control system for computers in an Android platform using Bluetooth 4.0 and Java. The client and server applications of this project would enable Bluetooth connection between the phone and computer, and once connected the applications would give the user full control of the computer’s mouse and keyboard. The mobile application was developed for Android version 4.2. However, it is compatible with the devices with Android version 2.0 and above 2.0. The server application was developed for the computer and is compatible with any computers integrated with Bluetooth. As the applications use Bluetooth to communicate between the client and server devices, the maximum distance between the client and server device should not be more than 100 meters.

The client application was developed using Samsung Galaxy III phone. However different brands of Android phones have different features that might cause the application to behave differently. The client and server applications have been tested using different Android phones and laptops simultaneously. The applications worked well in all the devices when the distance between the client and server was from 1-10 meters. However, when the distance between the client and server was from 10 -20 meters, there were a few interruptions encountered in Bluetooth communication between the client and server devices. This project can only be used for the computers that use English language layout keyboard. The application sometimes can crash as Bluetooth might not work properly. Also the controlling of mouse of the server device with phone is not as smooth as it is expected to be.

Thus, despite of several limitations, both the applications were successfully developed in confined time constraints. However further modification can be done to applications to fix the errors and to provide additional features.

Chapter 8: Conclusion

The project has come to a point where I have successfully implemented both the technologies Wi-Fi and Bluetooth. The Wi-Fi Technology has been used in the MuSync: Live Music Synching Application whereas the Bluetooth for the Wireless Touchpad Application. The reason for using the Wi-Fi for the MuSync was the requirement of high bandwidth, a much faster connection but the Bluetooth technology provided enough speed to control the wireless mouse.

The final MuSync version is an application where a user can be become the host and share his audio and all the connected peers will be able to join the host and the host will play the same audio across all the connected peers at the same time by making the use of its peers' speakers. In future, I will also include a feature a user who have logged in using his MuSync account can select the audio file from library available over the cloud using JSON technology. The future work of the application will make the application's connected users to control the volume and share the music files over the network.

The Wireless Touchpad mouse has been implemented successfully via Bluetooth and there's hardly any lag noticed in the application. The mouse pointer is smooth enough to carry out the day to day activities. On performance fronts it is much better and efficient in the various other applications that are exist in the google play store. The application can further be extended by adding a virtual keyboard option, well the newer versions of desktop operating systems do support the Virtual Mouse therefore, keyboard remains a secondary part. Despite being able in use, the application has certain limitations such as it cannot be used on the phones without Bluetooth and the phone with the Android operating system earlier than version 2.0.

I have kept both the applications as open source and have hosted them on GitHub for the coders to work on these applications and make it even better. In addition, both these projects have enough capabilities therefore, in near future I am planning to extend these to windows and iOs platform and incorporate video playing capabilities.

References

Research Papers And Journals:

[1] *Ardalan Amiri Sani, Kevin Boos, Min Hong Yun, and Lin Zhong.* **Rio: A System Solution for Sharing I/O between Mobile Systems,** *MobiSys '14 Proceedings of the 12th annual international conference on Mobile systems, applications, and services, Pages 259-272,* ACM New York, NY, USA, 02-06-2014.

[2] Bert Busstra, N-A. Le-Khac, M-Tahar Kechadi. **Android and Wireless data-extraction using Wi-Fi**, *Page: 170 - 175 Innovative Computing Technology (INTECH), 2014 Fourth International Conference* on 13-15 Aug. 2014, Luton

[3]*Román Belda, Pau Arce, Ismael de Fez, Francisco Fraile and Juan Carlos Guerri.* **Android real-time audio communications over local wireless,** *Waves - 2012 - year 4/ISSN 1889-8297, Pages 35-42.*

[4] Neshat Karim Shaukat, **Wi-Fi Direct in Android Using Peer to Peer Communication,** *Page: 30-35, IJRASET, Vol. 2 Issue I, January 2014,ISSN: 2321-9653.*

[5] *Miss. Rachana N. Sawade, Prof. P. V. Dudhe,* **Wifi AP Based Secure Data Sharing Among Smartphones And Computer System**, *International Journal on Recent and Innovation Trends in Computing and Communication, Volume: 3* Issue: 4 2313 – 2316.

[6] *Iliyan Nachev, Stoyan Maleshkov,* **Android-based Control Interface Solution for Windows Applications,** *International Virtual Conference, Section 14. Information Technology,* pp. 2073-2077, December, 2012

[7] *Dr. Khanna SamratVivekanand Omprakash,* **Concept of Remote controlling PC with Smartphone Inputs from remote place with internet,** *International Journal of Advanced*

Research in Computer Science and Software Engineering, Volume 2, Issue 1, January 2012 ISSN: 2277 128X

[8] *Lushin Barde, Neha Dhole, Pragati Waghmare, Swati Suryawanshi***, Controlling PC/Laptop via Android Phone (Android Remote Control),** *Vol. 3 Issue 1*, January - 2014, pp. 2806-2810, ISSN: 2278-0181

[9] *P.Sanoop Kumar, K.Teja K.Bhavyaka, E. Sirisha, Remote Trackpad,* **Android Controlled - Computer Mouse Pointer using Bluetooth,** *International Journal of Advanced Research in Computer Science***,** *Volume 6, No. 1, ISSN No. 0976-5697*, page- 83-87, January 2015

[10] *Priya Mehrotra, Tanshi Pradhan and Payal Jain, Priya Mehrotra, Tanshi Pradhan and Payal Jain***, Instant Messaging Service on Android Smartphones and Personal Computers,** *Volume 4, Number 3 (2014), pp. 265-272*, ISSN 0974-2239

[11] *Waheb A. Jabbar, M. Ismail and R. Nordin,* **Peer-to-Peer Communication on Android-Based Mobile Devices: Middleware and Protocols,** *Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference on***,** 28-30 April 2013, Hammamet.

[12] *Bhoopesh kumawat, Sudhendra Pal Singh, Chandra Prakash Verma***. Intranet Based Messaging Service on Android Smartphones and Tablets**, IJARCSSE, *Page 1030-1032, Volume 3, Issue 7,* July 2013**.**

[13] *Prof. Anil Hingmire, Ms. Mrunal Tipari , Mr. Rohit Gopalan, Mr. Sanman Chavan,* **Implementation of Voice, Video and Text Data over Wi-Fi,** *International Journal of Engineering Research and General Science Volume 3, Issue 2, March-April, 2015*, ISSN 2091-2730

Websites:

- http://ieeexplore.ieee.org/Xplore/home.jsp
- http://dl.acm.org/
- http://developer.android.com/index.html
- http://www.ijarcsse.com
- https://www.thenewboston.com/videos.php?cat=6
- http://anrg.usc.edu/
- http://bundlr.com/
- http://www.androidhive.info/
- http://github.com/

Appendix A

Technology and tools used:

1. **JSON** (JavaScript Object Notation) is a programming language. It is minimal, textual, and a subset of JavaScript. It is an alternative to XML.

   Android provides support to parse the JSON object and array.

   Advantage of JSON over XML

   1) JSON is faster and easier than xml for AJAX applications.

   2) Unlike XML, it is shorter and quicker to read and write.

   3) It uses array.

Android provides four different classes to manipulate JSON data. These classes are JSONArray, JSONObject, JSONStringer and JSONTokenizer.

The first step is to identify the fields in the JSON data in which you are interested in.

JSON – Parsing

For parsing a JSON object, we will create an object of class JSONObject and specify a string containing JSON data to it.

The last step is to parse the JSON. An JSON file consist of different object with different key/value pair e.t.c. So JSONObject has a separate function for parsing each of the component of JSON file.

The method **getJSONObject** returns the JSON object. The method **getString**returns the string value of the specified key.