**종합설계 프로젝트**

**PreFuture**

**Dynamic Presentation**

**Design Specification**

**담당교수 : 교수님**

**담당조교 : 조교**

**조 원 :**

Content

[1. Projects Overview 4](#_Toc324877847)

[1.1. Scope and Objectives 4](#_Toc324877848)

[1.1.1. Change about presentation tool - All-in-one 4](#_Toc324877849)

[1.1.2. Improvement about presentation environment - Anywhere 4](#_Toc324877850)

[1.1.3. Scope 4](#_Toc324877851)

[1.2. Supplementary Requirements 5](#_Toc324877852)

[2. Customer Requirements 8](#_Toc324877853)

[2.1 User requirements 8](#_Toc324877854)

[2.2 System requirements 9](#_Toc324877855)

[2.3 Domain requirements 10](#_Toc324877856)

[3. Requirements Analysis 11](#_Toc324877857)

[3.1Structural Analysis 11](#_Toc324877858)

[3.1.1. Boundary 11](#_Toc324877859)

[3.1.2. Control 11](#_Toc324877860)

[3.1.3. Entity 12](#_Toc324877861)

[3.2. Deployment Model 13](#_Toc324877862)

[3.3. Analysis Packages 14](#_Toc324877863)

[3.1.1 Application Specific Layer 15](#_Toc324877864)

[3.1.2. Application Generic Layer 15](#_Toc324877865)

[3.1.3. Middleware Layer 15](#_Toc324877866)

[3.1.4. Application System Software Layer 15](#_Toc324877867)

[4. Use Case Realization Design 16](#_Toc324877868)

[4.1. Use Case 1 - Process Connection 16](#_Toc324877869)

[4.1.1. Use Case 1 – Realization 16](#_Toc324877870)

[4.1.2. Use Case 1 – Relationship 17](#_Toc324877871)

[4.2. Use Case 2 – Move mouse using touchpad 18](#_Toc324877872)

[4.2.1. Use Case 2 – Realization 18](#_Toc324877873)

[4.2.2. Use Case 2 – Relationship 19](#_Toc324877874)

[4.3. Use Case 3 – Click mouse using touchpad 20](#_Toc324877875)

[4.3.1. Use Case 3 – Realization 20](#_Toc324877876)

[4.3.2. Use Case 3 – Relationship 21](#_Toc324877877)

[4.4. Use Case 4 – Move mouse using motion 22](#_Toc324877878)

[4.4.1. Use Case 4 – Realization 22](#_Toc324877879)

[4.4.2. Use Case 4 – Relationship 23](#_Toc324877880)

[4.5. Use Case 5 – Control keyboard 24](#_Toc324877881)

[4.5.1. Use Case 5 – Realization 24](#_Toc324877882)

[4.5.2. Use Case 5 – Relationship 25](#_Toc324877883)

[4.6. Use Case 6 – Process record sound 26](#_Toc324877884)

[4.6.1. Use Case 6 – Realization 26](#_Toc324877885)

[4.6.2. Use Case 6 – Relationship 27](#_Toc324877886)

[4.7. Use Case 7 – Process play sound 28](#_Toc324877887)

[4.7.1. Use Case 7 – Realization 28](#_Toc324877888)

[4.7.2. Use Case 7 – Relationship 29](#_Toc324877889)

[4.8. Use Case 8 – Transfer voice to PC speaker 30](#_Toc324877890)

[4.8.1. Use Case 8 – Realization 30](#_Toc324877891)

[4.8.2. Use Case 8 – Relationship 31](#_Toc324877892)

[4.9. Use Case 9 – Transfer document 32](#_Toc324877893)

[4.9.1. Use Case 9 – Realization 32](#_Toc324877894)

[4.9.2. Use Case 9 – Relationship 33](#_Toc324877895)

[5. Subsystem Design 34](#_Toc324877896)

[5.1. Action Manager Subsystem 34](#_Toc324877897)

[5.2. Sound Action Management 35](#_Toc324877898)

[5.3. Transfer Management 36](#_Toc324877899)

[5.4. Mouse Action Manager 37](#_Toc324877900)

[5.5. Keyboard Action Management 38](#_Toc324877901)

[5.6. PC Connection Management 39](#_Toc324877902)

[5.7. Device Connection Management 40](#_Toc324877903)

[6.Human Interfaces 41](#_Toc324877904)

[6.1. Client 41](#_Toc324877905)

[6.2. PC 43](#_Toc324877906)

[7. Use-Case Scenarios 43](#_Toc324877907)

[8. System/Data Dependencies & Requirements 45](#_Toc324877908)

[8.1. Software System 45](#_Toc324877909)

[8.2. Hardware System 45](#_Toc324877910)

[8.3. Data Dependencies & requirements 46](#_Toc324877911)

[9. Appendices 46](#_Toc324877912)

[9.1. Project Status 46](#_Toc324877913)

[9.2 Reference 47](#_Toc324877914)

# 1. Projects Overview

<Dynamic Presentation> aimed to provide easily an All-in-one technology to user for presentation when environment for presentation is not prepared in advance or a lack of equipment. A number of mike and remote-control application, based on Android platform, already developed. Because of some problems, for example delay, noise or UI without user experience, inconveniences often arise. Dynamic presentation can improves problems that existing system have and provides various and useful functions for user.

## 1.1. Scope and Objectives

### 1.1.1. Change about presentation tool - All-in-one

Dynamic Presentation provides just one system integrating mike, keyboard, pointing using Gyroscope sensor and virtual storage. This can replace existing mouse for presentation, control keyboard and mouse action and use documents with file management easily. The service containing mobility and convenience can be developed for presentation.

### 1.1.2. Improvement about presentation environment - Anywhere

Dynamic Presentation operates in Wireless network environment like 3G and Wi-Fi. Everywhere wireless network worked user can this anywhere.

### 1.1.3. Scope

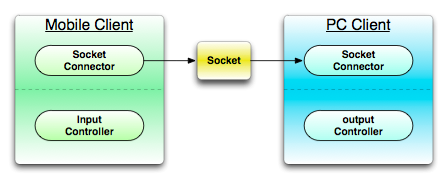


Figure 1 : Project scope

1.1.3.1. Mobile Client  
- Mobile Graphic User Interface

-Communication for connecting PC

- Mike, Recording

- Cloud service

- Keyboard action control

- Mouse action control

1.1.3.2. PC Client  
- Communication for connecting mobile device

- Output voice streaming

- Cloud service

- Keyboard action control

- Mouse action control

## 1.2. Supplementary Requirements

Introduction

This document includes all requirements that aren’t extracted when making Use case.

Functionality

• Logging and Error Handling

- Every errors have to be saved in storage.

• Pluggable Rule

- System’s functions can be specialized using optional regulation that will be invoked at the point generating various scenario of Use Case.

• Security

- All user must be authorized.

Usability

• document

- A manual that defined way to use is needed.

- A help mode that explained each functions is needed.

- A problem-solving document containing problems and solution is needed.

• Human Factors

- Customer can see Dynamic Presentation system using proper sized screen. So, the letters have to consist of suitable color, size and font.

- Customer wants to use Dynamic Presentation quickly. Then he/she has not a correct understanding. Thus, it is a critical factor to prevent error generating and use system easily.

Reliability

• Recoverability

- If system has problem with external problem, alternative is prepared in advance in order to handle that immediately.

• Responsibility

- The system sensor events (touch, acceleration and gravity) will respond to more than 95%.Because the value is to passed exactly.

- A value of sensor that user inputs have to respond within 0.1 seconds.

Performance

• Speed

- As be said in the Human Factors section, customer want to use Dynamic Presentation quickly. Therefore, delay has to take 0.5 second at most when system is used.

Supportability

• Internationality

- System provides an English service for foreigner who has no ability in a Korean.

Implementation Constraints

• System have to be implemented using Java technical solution. It is anticipated to give a guarantee of portability and scalability.

• System have to be implemented using Android SDK 2.3.

Interfaces

• Hardware Interfaces

- Smartphone(Android SDK 2.3.3, Android API Ver.10)

- Laptop or Desktop

- The system shall require network access.

• Software Interfaces

- System have to provide an intuitive UI for customer to use system easily.

Design

• The system will be client based and therefore platform dependent.

Implementation

• The system shall be written in Java programming language.

Security

• The system authenticates network information and secure code.

Accuracy

• Mouse point values shall be dependency to the display pixel.

• Transmission voice shall not be disconnected.

• Transmission voice delay shall be least.

Efficiency

• The system connection shall be reconnect when disconnected.

Interoperability

• The PC system shall interoperate with the following OS.

- Windows.

• The device system shall interoperate with the following OS.

- Android API 10 or above.

Response Time

• Voice transmission shall occur within 1 second.

• Controllers response shall occur immediately.

Safety

• When system is exit, the system shall lose connection information.

# 2. Customer Requirements

This section contains the architecture view of your use-case model.  
It may include natural language or (and) use-case diagram(s), actor descriptions and prioritized use-case descriptions.

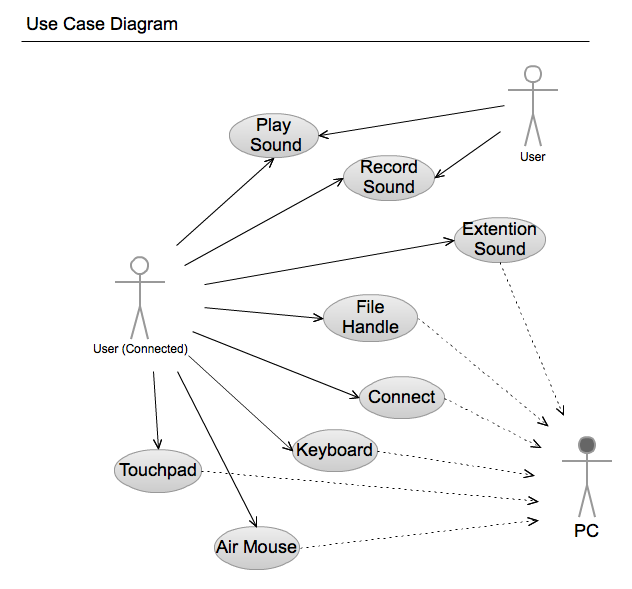


Figure 2 : Usecase Diagram

## 2.1 User requirements

- The system should provide users with connectivity features.  
- The system should be able to enter the required information to connect to user.

: The system should be telling wrong, if the user's connection information is wrong.

- The system will be allowed use some features, even though system is not connected.

- The system provides disconnect feature for the user.

- The system provides the feature to store connected information.

- The system provides the feature to record user's sound.

: The system should provide user's voice recording and allows user to enter information in the file.

: The system should allow storing user’s recorded sound.

: The system should allow managing files stored.

- The system provides the feature to transfer user’s voice to PC

: The system should be prevented delays when transfer sound.

- The system provides user's mouse pointer control.

- The system provides user's mouse click control.

- The system provides user's motion mouse pointer control.

- The system provides user's keyboard control.

- The system provides the feature to save files.

: The system should show the details of the file.

- The system provides the feature to transfer files.

: The system should show the list of stored files.

## 2.2 System requirements

- The system provides a connection between PC to device.

: The system might be able to input data needed by user in order to connect between PC to device

: When it failed to connect, a notice will be shown.

: User might be able to using system partially without connection.

: The system provides a result depends on connection.

- System provides the feature to connected user as follows.

: PC in the sound transmission.

:: The system activate the feature to transmit a sound to PC

: Mouse control

:: The system activate the feature to transmit a movement of mouse controlled by user to PC

: File transmission

:: The system provides the feature to transmit user selected file to PC

- The system provides a function to do not connect user as follows.

: Recording

:: The system saves a voice in the device.

: Showing file list

:: The system provides the feature to show a file in the device.

- The system provides the feature to user wanted connect to system as follows.

: Trying to connect to the system asks to enter the eigenvalues ​​of the PC.

: The user must enter the correct value that does not deviate from the type of eigenvalues at prompt.

- The system provides disconnect feature to the PC.

: The system provides that feature disconnection information to user that want to disconnect.

- The system provides feature to record sound..

: When user wants record the system provides enabled recording option feature to user.

- The system provides feature that transmit for user voice.

: When user want transmit the voice to PC the system provides feature after enabled option.

: The device must be requiring input the voice through the microphone.

- The system provides mouse control feature.

: The pointer that out of range should not move.

: It is prevented that multiple input pointing.

- The system activate feature to users who want to click.

: The mouse provides the ability to control points that user's movement.

:: It should handle user's move within the area shall be controlled in system.

- The system provides keyboard control.

: The user should not require process that consist multiple actions in time.

- The system provides file transmission.

: The system provides detail information of file for using to user.

: The user activate transfer files menu.

## 2.3 Domain requirements

- When the system error occurs, the system records error cause and the time information.

- When system creates a voice file in system storage should not be exceeded more than create a file.

- The system can handle only allows handling for the file type.

- The system file processing and file processing is handled separately about other details.

- The system is used Two-dimensional version of the labeling system (e.g. 1.0).

# 3. Requirements Analysis

## 3.1Structural Analysis

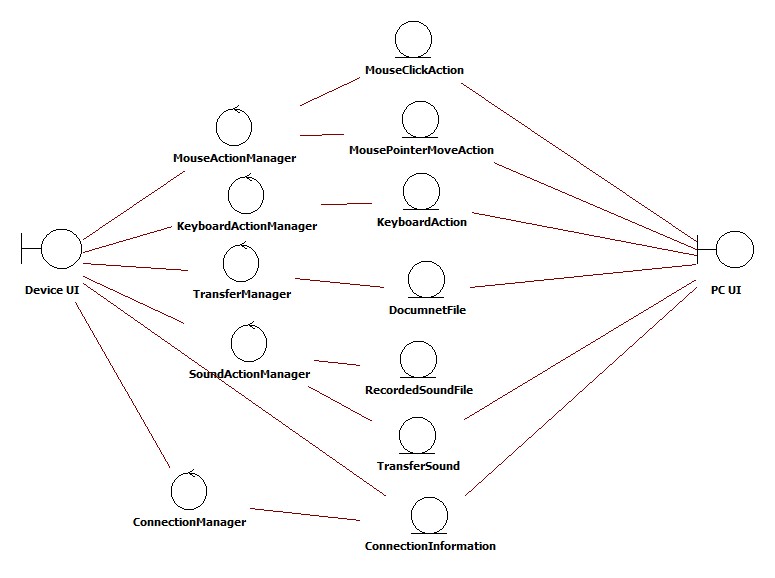


Figure 3 : Structure analysis Usecase model

### 3.1.1. Boundary

- Device UI provides the user interface, the main interaction between the user and the system. Used to initiate recording sounds, controlling mouse and keyboard actions, file transference to PC. Call the ConnectionManager to connect device and pc.

- PC UI shows the ConnectionInformation that use when user wants to start system.

### 3.1.2. Control

- MouseActionManager is the controller that user can move mouse pointer of computer and do click action remotely.

- KeyboardActionManager provides function that can use keyboard remotely.

- SoundActionManager is the controller that records sound and plays recorded sound file.

- ConnectionManager is the controller that used in the system-initiation step before another manager starts.

- TrasferManager is the controller that sends document and sound files from device to pc.

### 3.1.3. Entity

- MousePointerMoveAction

- MouseClickAction

- KeyboardAction

- DocumnetFile

- RecordedSoundFile

- ConnectionInformation

- TransferSound

## 3.2. Deployment Model

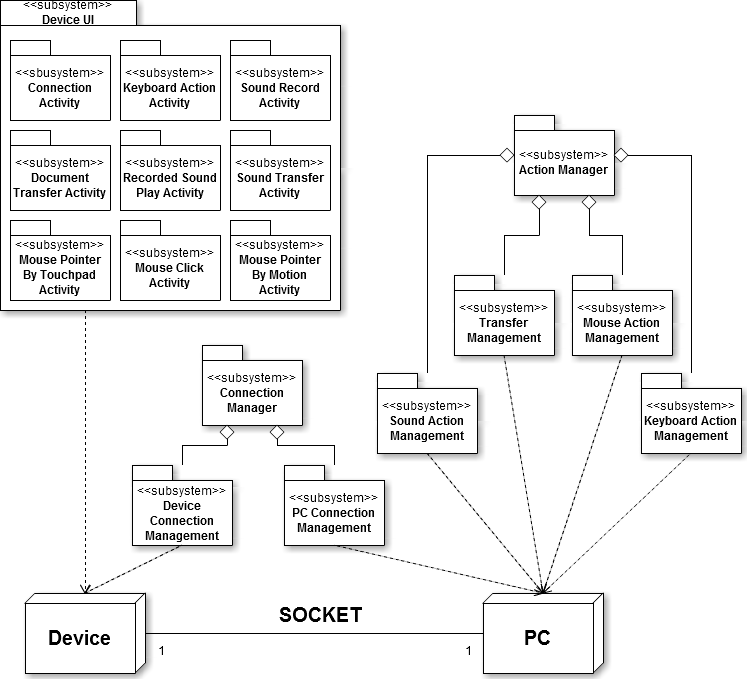


Figure 4 : Deployment Model

## 3.3. Analysis Packages

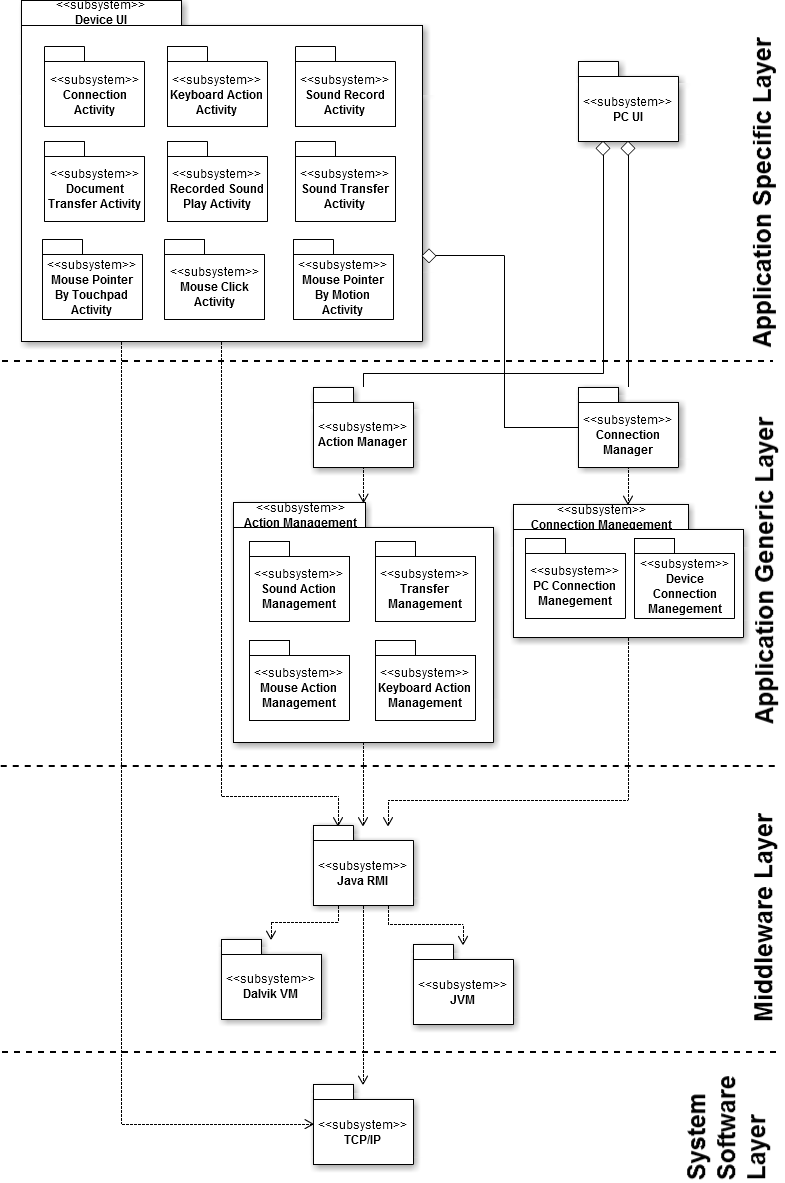


Figure 5 : Subsystem Model

### 3.1.1 Application Specific Layer

This layer consists of the Device UI subsystem and PC UI subsystem. The Device UI further consists of the ConnectionUI, KeyboardActionUI, SoundRecordUI, DocumnetTransferUI, RecordedSoundTransferUI, SoundTransferUI, MousePointerByTouchpadUI, MouseClickUI and MousePointerByMotionUI. The role of this subsystem is to create and manage the user interfaces and screens.

### 3.1.2. Application Generic Layer

This layer consists of the various sub-systems which are independent of the system and it consists of subsystem like Action Manager Facade, Sound Action Management, Transfer Management, Mouse Action Management, Keyboard Action Management and Connection Management. The Action Manager Facade handles request sent from Device UI and invokes the appropriate subsystem on the PC UI. The Connection Management interacts with Device UI and PC UI to initiate system.

### 3.1.3. Middleware Layer

This layer consists of the system specific subsystems like Java RMI, Dalvik VM and JVM. This layer provides the medium by which system can be developed in the manner desired. The upper layers realize this subsystem to communicate with other subsystems to perform appropriate functions. Since the middleware layer supports Java platform or Android platform, all this subsystem directly depends upon the JVM subsystem and Dalvik subsystem.

### 3.1.4. Application System Software Layer

This layer consists of TCP/IP subsystem. This layer provides the functionality to other subsystem to communicate over the network.

# 4. Use Case Realization Design

## 4.1. Use Case 1 - Process Connection

### 4.1.1. Use Case 1 – Realization



Figure 6 : Use Case 1 – Realization Diagram

Connection begins with the PC UI display the required connection information and connection credential on input on the system. And Connection Activity display input interface for the user. Then the user input connection information and connection credential on Connection Activity on Device. That will call inputInfo() method. In DeviceConnectionManager call connect() method that has consists of connection info and connection credential. PCConnectionmanager receive credential and call verifyCredential() method for that check verification. If credential is valid PCconnectionManager call displayStatus() for valid status. And return valid to DeviceConnectionManager. DeviceConnectionManager call displayStatus() method for display valid status to the user. If not PCconnectionManager call displayStatus() for invalid status. And return invalid to DeviceConnectionManager. DeviceConnectionManager call displayStatus() method for display invalid status to the user. Also invoke confirmRetry() method that is ask the user retry connect. If the user answer yes, that action call accept() method and go to sequence 2. If not that action call deny() method, consequence invoke cancelConnection() method in DeviceConnectionManager.

### 4.1.2. Use Case 1 – Relationship

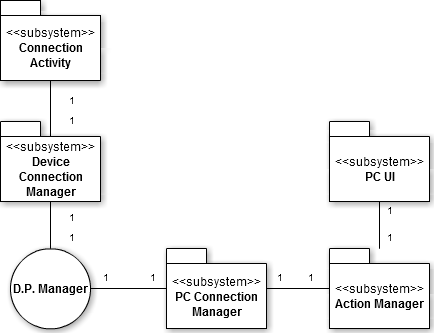


Figure 7 : Use Case 1 – Relationship Diagram

The Connection Activity uses exactly one Device Connection Manager. The Device Connection Manager is used by exactly one Connection Activity. They are mutually aware of each other. The PC Connection Manager uses exactly one D.P. Manager. The D.P. Manager is used by exactly one PC Connection Manager. The Action Manager uses PC Connection Manager. The PC Connection Manager is used by exactly one Action Manager. The PCUI uses exactly one Action Manager. The Action Manager is used by exactly one PC UI. They are mutually aware of each other.

## 4.2. Use Case 2 – Move mouse using touchpad

### 4.2.1. Use Case 2 – Realization



Figure 8 : Use Case 2 – Realization Diagram

This use case begins with MousePointerbyTouchpad Activity display touchpad interface for user. The user will invoke onTouch() method by touch on Device. and then MousePointerbyTouchpad Activity call sendMouse() method on DeviceConnectionManager that has consists differential x position, and differential y position. DeviceConnectionManager write on session, that received by PCConnetionManager. PCconnectionManager call moveMouse() method on MouseActionManager. MouseActionManager actually move by mouseMove() method. That's result displayed by PCUI. The user acknowledges the result.

### 4.2.2. Use Case 2 – Relationship

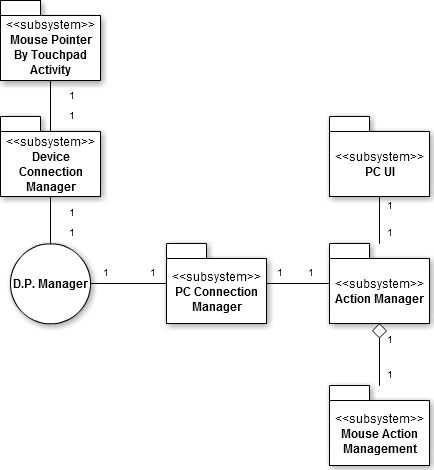


Figure 9 : Use Case 2 – Relationship Diagram

The Mouse Pointer by Touchpad Activity uses exactly one Device Connection Manager. The Device Connection Manager is used by exactly one Mouse Pointer by Touchpad Activity. They are mutually aware of each other. The PC Connection Manager uses exactly one D.P. Manager. The D.P. Manager is used by exactly one PC Connection Manager. The Action Manager uses PC Connection Manager. The PC Connection Manager is used by exactly one Action Manager. The PC UI uses exactly one Action Manager. The Action Manager is used by exactly one PC UI. They are mutually aware of each other. The Action Manager is composed of Mouse Action Management. Mouse Action Management composes Action Manager. Action Manager is aware of Mouse Action Management of which it is composed, but not vice versa.

## 4.3. Use Case 3 – Click mouse using touchpad

### 4.3.1. Use Case 3 – Realization



Figure 10 : Use Case 3 – Realization Diagram

This use case begins with MouseClick Activity displays touchpad interface for user. The user will invoke onTouch() method by touch on Device concurrently Mouse Click Activity call response() method for assure the user's action. And then MouseClick Activity call sendMouse() method on DeviceConnectionManager that has consists mouse factor -left or right. DeviceConnectionManager write on session, that received by PCConnectionManager. PCconnectionManager call moveMouse() method on MouseActionManager. MouseActionManager actually click by mouseClick() method. That's result displayed by PCUI. The user acknowledges the result.

### 4.3.2. Use Case 3 – Relationship

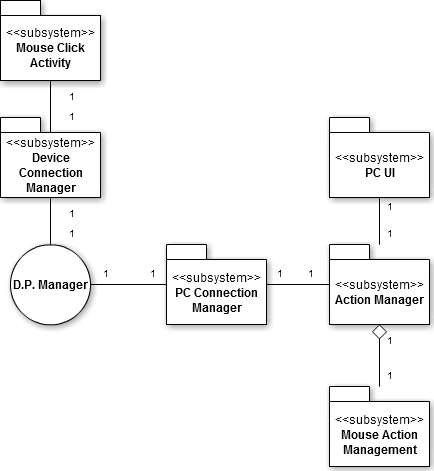


Figure 11 : Use Case 3 – Relationship Diagram

The Mouse Click Activity uses exactly one Device Connection Manager. The Device Connection Manager is used by exactly one Mouse Click Activity. They are mutually aware of each other. The PC Connection Manager uses exactly one D.P. Manager. The D.P. Manager is used by exactly one PC Connection Manager. The Action Manager uses PC Connection Manager. The PC Connection Manager is used by exactly one Action Manager. The PC UI uses exactly one Action Manager. The Action Manager is used by exactly one PC UI. They are mutually aware of each other. The Action Manager is composed of Mouse Action Management. Mouse Action Management composes Action Manager. Action Manager is aware of Mouse Action Management of which it is composed, but not vice versa.

## 4.4. Use Case 4 – Move mouse using motion

### 4.4.1. Use Case 4 – Realization



Figure 12 : Use Case 4 – Realization Diagram

This use case begins with MousePointer byMotion Activity display button interface for user. The user will invoke onTouch() method by touch on Device. And then the user starts move actually device. MousePointerbyMotionActviticy call sendMouse() method on DeviceConnectionManager that has consists differential x position, and differential y position by accelerometer. DeviceConnectionManager write on session, that received by PCConnectionManager. PCconnectionManager call moveMouse() method on MouseActionManager. MouseActionManager actually move by mouseMove() method. That's result displayed by PCUI. The user acknowledges the result.

### 4.4.2. Use Case 4 – Relationship

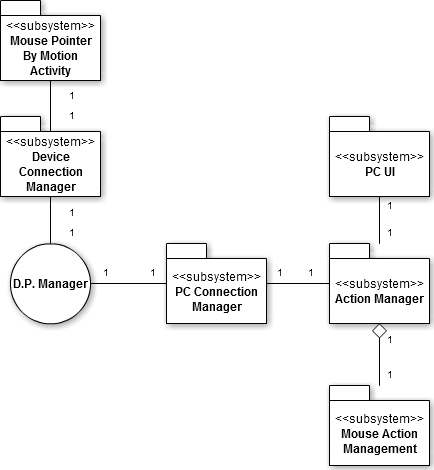


Figure 13 : Use Case 4 – Relationship Diagram

The Mouse Pointer by Motion Activity uses exactly one Device Connection Manager. The Device Connection Manager is used by exactly one Mouse Pointer by Motion Activity. They are mutually aware of each other. The PC Connection Manager uses exactly one D.P. Manager. The D.P. Manager is used by exactly one PC Connection Manager. The Action Manager uses PC Connection Manager. The PC Connection Manager is used by exactly one Action Manager. The PC UI uses exactly one Action Manager. The Action Manager is used by exactly one PC UI. They are mutually aware of each other. The Action Manager is composed of Mouse Action Management. Mouse Action Management composes Action Manager. Action Manager is aware of Mouse Action Management of which it is composed, but not vice versa.

## 4.5. Use Case 5 – Control keyboard

### 4.5.1. Use Case 5 – Realization



Figure 14 : Use Case 5 – Realization Diagram

This use case begins with KeyboardAction Activity displays keyboard interface for user. The user will invoke onTouch() method by touch on Device concurrently KeyboardAction Activity call response() method for assure the user's action. And then KeyboardAction Activity call sendKeyboard() method on DeviceConnectionManager that has consists keyboard factor. DeviceConnectionManager write on session, that received by PCConnectionManager. PCconnectionManager call keyPress() method on KeyboardActionManager. KeyboardActionManager actually press the keyboard by keyboardPress() method. That's result displayed by PCUI. The user acknowledges the result.

### 4.5.2. Use Case 5 – Relationship

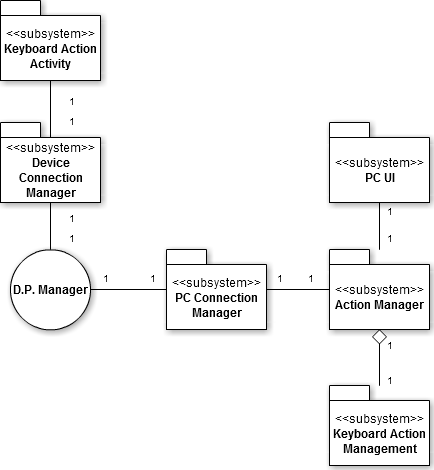


Figure 15 : Use Case 5 – Relationship Diagram

The Keyboard Action Activity uses exactly one Device Connection Manager. The Device Connection Manager is used by exactly one Keyboard Action Activity. They are mutually aware of each other. The PC Connection Manager uses exactly one D.P. Manager. The D.P. Manager is used by exactly one PC Connection Manager. The Action Manager uses PC Connection Manager. The PC Connection Manager is used by exactly one Action Manager. The PC UI uses exactly one Action Manager. The Action Manager is used by exactly one PC UI. They are mutually aware of each other. The Action Manager is composed of Keyboard Action Management. Keyboard Action Management composes Action Manager. Action Manager is aware of Keyboard Action Management of which it is composed, but not vice versa.

## 4.6. Use Case 6 – Process record sound

### 4.6.1. Use Case 6 – Realization



Figure 16 : Use Case 6 – Realization Diagram

Record sound process use case will starts with SoundRecord Activity display record interface to user. The user will invoke onTouch() method on Device concurrently SoundRecord Activity call response() method for assure the user's action. And SoundRecord Activity call startRecord() method on DeviceSoundActionManager. DeviceSoundActionManager will call itself initRecord() method for start recording. After the user speak and sound will be recorded. Finish speaking, the user invoke onTouch() method on Device for pause recording. SoundRecord Activity invoke pauseRecord() method on DeviceSoundActionManager consequence invoke askContinue() method for user's next selection. If the user answer yes by onTouch() method SoundRecordActivity call continueRecord() method. If the user answer no by onTouch() method SoundRecord Activity call stopRecord() method on DeviceSoundActonManager. After that DeviceSoundActionManager call itself saveRecord() method for saving recorded file.

### 4.6.2. Use Case 6 – Relationship

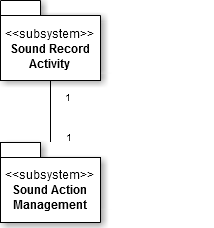


Figure 17 : Use Case 6 – Relationship Diagram

The Sound Record Play Activity uses exactly one Sound Action Management. The Sound Action Management is used by exactly one Sound Record Activity. They are mutually aware of each other.

## 4.7. Use Case 7 – Process play sound

### 4.7.1. Use Case 7 – Realization



Figure 18 : Use Case 7 – Realization Diagram

Paly sound process use case will starts with RecordedSoundPlay Activity display play interface to user. The user will invoke onTouch() method on Device concurrently Recorded Sound Play Activitycall response() method for assure the user's action. If device has not recorded file RecordedSoundPlay Activitycall notify() method for no file. If device has recorded file RecordedSoundPlay Activitycall displayRecordedList() method for user. And then the user select file on list that call onTouch() method.If that cannot play recorded sound, RecordedSoundPlay Activitycall notify() which consists can't play. If that can play the file RecordedSoundPlay Activity call playRecordedStart() method on DeviceSoundActionManager. DeviceSoundActionManager call itself playRecordedInit().

### 4.7.2. Use Case 7 – Relationship

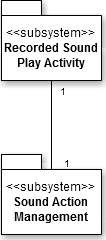


Figure 19 : Use Case 7 – Relationship Diagram

The Recorded Sound Play Activity uses exactly one Sound Action Management. The Sound Action Management is used by exactly one Recorded Sound Play Activity. They are mutually aware of each other.

## 4.8. Use Case 8 – Transfer voice to PC speaker

### 4.8.1. Use Case 8 – Realization



Figure 20 : Use Case 8 – Realization Diagram

Transfer voice to PC speaker use case will starts with SoundTransfer Activity display transfer voice interface to user. The user will invoke onTouch() method on Device concurrently SoundTransfer Activitycall response() method for assure the user's action. SoundTransfer Activitycall transferVoiceInit() method on DeviceSoundActionManager. Consequently DeviceSoundActionManager call transferVoiceInit() method on DeviceConnectionManager for making voice session. DeviceConnectionManager call itself sendVoice() method for PCConnectionManager preparing transfer voice. PCConnectionManager call transferVoiceInit() method on SoundActionManager. After preparing transfer voice the user speaking and SoundActionManager call palyOutput() method that actually use PC speaker. If the user wants to stop transferring voice.The user call onTouch() method on SoundTransfer Activity. SoundTransfer Activitycall transferVoiceTerm() method consequently DeviceSoundActionManager call transferVoiceTerm() for terminate session. After that DeviceConnectionManager call itself sessionClose() method.

### 4.8.2. Use Case 8 – Relationship

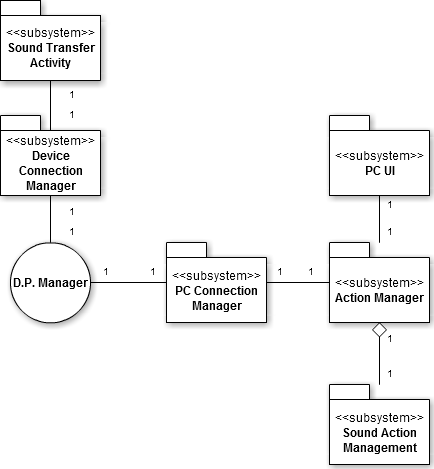


Figure 21 : Use Case 8 – Relationship Diagram

The Sound Transfer Activity uses exactly one Device Connection Manager. The Device Connection Manager is used by exactly one Sound Transfer Activity. They are mutually aware of each other. The PC Connection Manager uses exactly one D.P. Manager. The D.P. Manager is used by exactly one PC Connection Manager. The Action Manager uses PC Connection Manager. The PC Connection Manager is used by exactly one Action Manager. The PC UI uses exactly one Action Manager. The Action Manager is used by exactly one PC UI. They are mutually aware of each other. The Action Manager is composed of Sound Action Management. Sound Action Management composes Action Manager. Action Manager is aware of Sound Action Management of which it is composed, but not vice versa.

## 4.9. Use Case 9 – Transfer document

### 4.9.1. Use Case 9 – Realization



Figure 22 : Use Case 9 – Realization Diagram

Transfer Document use case will starts with DocumentTransfer Activity display transfer document interface to user. The user will invoke onTouch() method on Device concurrently DocumentTransfer Activity call response() method for assure the user's action. If Device has not any file to transfer DocumentTransfer Activity call notify() method to user for no file. If not DocumentTransfer Activity call displayFileList() for the user.

And then user select file by touch that call onTouch() method. DocumentTransfer Activity call confirmTransfer() method to user for that file is really wanted to transfer. If user deny by call deny() DocumentTransfer Activity call itself cancelTranfer() method. If not the user accept by call accept() method. If it ends, DocumentTransfer Activity call displayProgress() method. And call sendDocument() method on DeviceConnectionManager. DeviceConnectionManager uses sendDocument() on PCConnectionManager. PCConnectionManager call receiveDocument() method on TransferManager. In the end TransferManager call execDocumentFile() method.

### 4.9.2. Use Case 9 – Relationship

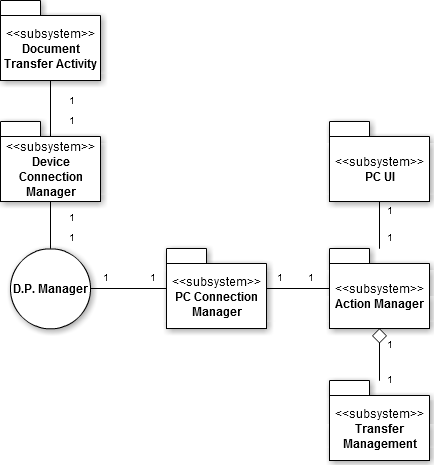


Figure 23 : Use Case 9 – Relationship Diagram

The Document Transfer Activity uses exactly one Device Connection Manager. The Device Connection Manager is used by exactly one Document Transfer Activity. They are mutually aware of each other. The PC Connection Manager uses exactly one D.P. Manager. The D.P. Manager is used by exactly one PC Connection Manager. The Action Manager uses PC Connection Manager. The PC Connection Manager is used by exactly one Action Manager. The PC UI uses exactly one Action Manager. The Action Manager is used by exactly one PC UI. They are mutually aware of each other. The Action Manager is composed of Transfer Management. Transfer Management composes Action Manager. Action Manager is aware of Transfer Management of which it is composed, but not vice versa.

# 5. Subsystem Design

## 5.1. Action Manager Subsystem



Figure 24 : Action Manager Subsystem Class Diagram

This subsystem is the main facade used in the system. It can invoke and call Sound Action Management, Transfer Management, Mouse Action Management, and Keyboard Action Management. It makes the decision to invoke a particular subsystem based on the request from the UI. For instance, if the user wishes to move a mouse pointer, the façade accepts this request and passes it on to a subsystem that can perform this operation. Hence, we can say that it acts as a simple interface to other subsystems.

## 5.2. Sound Action Management



Figure 25 : Sound Action Management Class Diagram

Sound Action Management subsystem manages the actions related sound. According to user’s choice, this subsystem can record user voice to sound file and transfer this file from device to pc.

## 5.3. Transfer Management



Figure 26 : Transfer Management Class Diagram

This subsystem is solely responsible for transfer control of the entire file. Because of this subsystem, it is possible to see the same document files –ppt, pdf, doc and so on – between Device and PC. A document file that user selected is transferred from device to PC5.4 Mouse Action Management

## 5.4. Mouse Action Manager



Figure 27 : Mouse Action Manager Class Diagram

This subsystem controls user requests for moving mouse pointer on the pc-side. There are two kinds of method that user can be selected for movement of pointer. One is to use touchpad of device; the other is to use a result of motion. It receives user request, parses and insights. Then It moves mouse pointer to direction that user wants to move.

## 5.5. Keyboard Action Management



Figure 28 : Keyboard Action Management Class Diagram

This subsystem management performs a requested operation that user wants. There is an action related keyboard in the requests. This subsystem analyzes these requests and invokes results of requests on the PC-side.

## 5.6. PC Connection Management



Figure 29 : PC Connection Management Class Diagram

This subsystem is system that controls connection between PC and device. It uses own address and randomly generated number. Comparing these with some information provided from Device Connection Management, it can validate whether correct user attempts to connection or not.

## 5.7. Device Connection Management



Figure 30 : Device Connection Management Class Diagram

This subsystem is responsible for connection management on the device. It interacts with PC Connection Management subsystem, gets initial connection information and some sequential random number from user and sends these data.

# 6.Human Interfaces

## 6.1. Client

6.1.1. Main

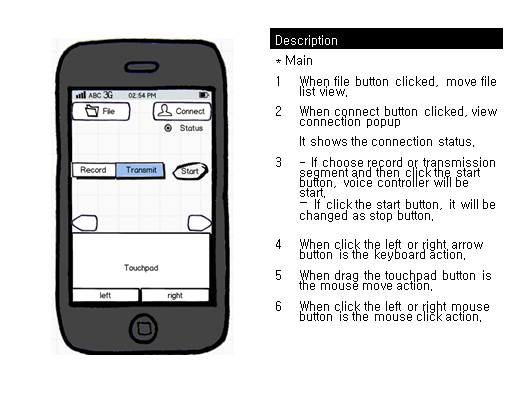


Figure 31 : Main Interface

6.1.2. Connection Popup

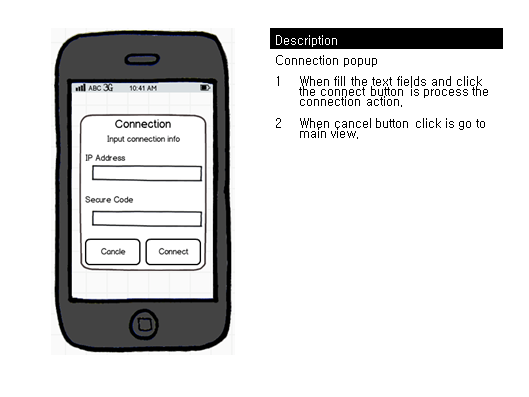


Figure 32 : Connection Popup Interface

6.1.3. File List

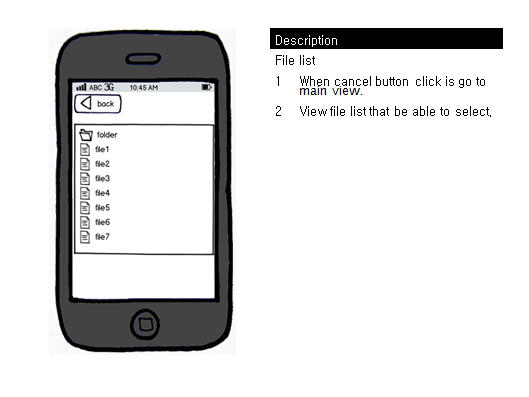


Figure 33 : File List Interface

6.1.4. File selection popup

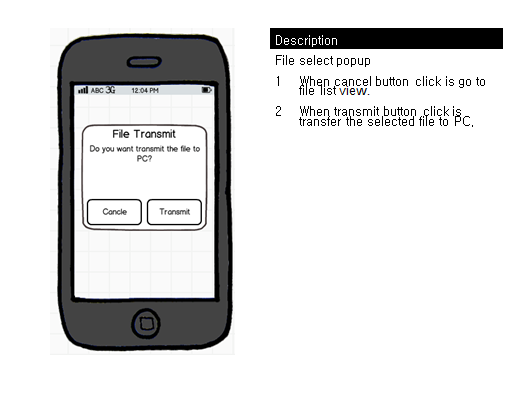


Figure 34 : File Selection Popup Interface

## 6.2. PC

6.2.1. PC Status

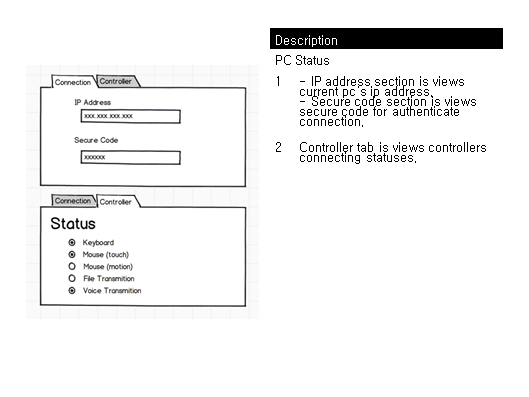


Figure 35 : PC Status Interface

# 7. Use-Case Scenarios

**7.1. Process Connecting**

User want to that PC connected to Device for Presentation. A User presses a button at pc, and User confirms IP information of pc. User input PC information confirmed to DEVICE, and User press "connecting "button. If connection failed, show message called "Fail" to users at device screens "retry?" User show message called. If a user wants a repeat performance, User input again information of pc, and User press "cancellation" button, and Device cancel a connection action if User do not want a repeat performance. The Pc on the information received from the user's information if the connection is then compared with the "success" shows the message tells User.

**7.2. Move mouse using touchpad**

User wishes that a mouse pointer of pc moves. You show a touch pad screen if user press a mouse button of Device. User want moving to courses a user draws a finger to a screen of device.

If connection failed, show message called "Fail" to users at device screens "retry?" User show message called. If a user wants a repeat performance, User input again information of pc, and User press "cancellation" button, and Device cancel a connection action if User do not want a repeat performance. Pc shows a movement of a mouse pointer to the courses that a user wants if software normally operate. Device work to a former step if user presses an end button.

**7.3. Click mouse using touchpad**

User wishes that a mouse click of pc occurs. Device shows a mouse click button screen to users. A user selects a mouse click to want at device. If a user presses a button on the left of mouse, pc generates a button click event on the left of mouse and pc generates a button click event right a mouse if a user press the right button of mouse. If connection failed, show message called "Fail" to users at device screens "retry?" User show message called. If a user wants a repeat performance, User input again information of pc, and User press "cancellation" button, and Device cancel a connection action if User do not want a repeat performance. Device uses sound and vibration for notify that user’s action is normally processed. Device work to a former step if user presses an end button.

**7.4. Move mouse pointer using motion**

User wants to operate by motion. User looks if you press a mouse motion button of Device so as to be able to move in a pointer by motion. User are moved in the directions that a situation of a mouse pointer of pc wants if User move in the directions where a user wants device. If connection failed, show message called "Fail" to users at device screens "retry?" User show message called. If a user wants a repeat performance, User input again information of pc, and User press "cancellation" button, and Device cancel a connection action if User do not want a repeat performance. Device work to a former step if user presses an end button.

**7.5. Control keyboard**

User wishes that an event of a keyboard of PC occurs Device shows buttons of a keyboard shape to user. A user presses a keyboard button to want at Device. . Device uses sound and vibration for notify that user’s action is normally processed. If connection failed, show message called "Fail" to users at device screens "retry?" User show message called. If a user wants a repeat performance, User input again information of pc, and User press "cancellation" button, and Device cancel a connection action if User do not want a repeat performance. Device work to a former step if user presses an end button.

**7.6. Process Record Sound**

A user wants to record the contents that one spoke. Recording begins since a user pressed a Record button in a Device screen if you press a Start button. A Start button is changed by pause button and stop buttons. User inputs a voice to Device by a voice a user. Recording becomes a stop, and you finish recording if you press a Pause button if you press a Stop button. A recording file finished is stored to Device.

**7.7. Process Play Sound**

Voice that is recorded in device is outputted through the speakers on the Device. If a user presses a voice playing button to a Device screen, User can read file lists. A user selects a voice file wanting a voice file list after confirmation. Device displays message called "empty" if there is not a voice file recorded. And you show message called "Problem" and Device returns to a relevant issue to telephone planes if you cannot display a voice file recorded. Device work to a former step if user presses an end button.

**7.8. Transfer voice to PC speaker**

A user wants to display own voice at the PC speakers which are not Device. If a user presses a voice transmission button to a Device screen, output works at pc speakers.

**7.9. Transfer document**

A user wants to use a file of Device at PC. If User presses a file transmission button of Device, User can see the file list that transmission is possible. A system shows message called "empty" to User if there is not a file list. If select the file which User wants "is OK?" User show message called. If the file which User selected meets, you press a button. If User wants other file, you press a cancellation button, and you cancel file transmission. Device looks if you pressed an OK button. If connection failed, show message called "Fail" to users at device screens "retry?" User show message called. If a user wants a repeat performance, User input again information of pc, and User press "cancellation" button, and Device cancel a connection action if User do not want a repeat performance. Device work to a former step if user presses an end button.

# 8. System/Data Dependencies & Requirements

## 8.1. Software System

Java Runtime

Java SDK 1.6

Android SDK 2.3

Libraries: JMF

## 8.2. Hardware System

Android Platform Smartphone(Android 2.3)

Java Runtime Environment

Network Card

## 8.3. Data Dependencies & requirements

In this system, data is not occurred (just values occurred). Occurred values in device client are transferred directly. And then these values are realized directly in PC client. So that data has not exists. Also data dependencies & requirements not exist.

# 9. Appendices

## 9.1. Project Status

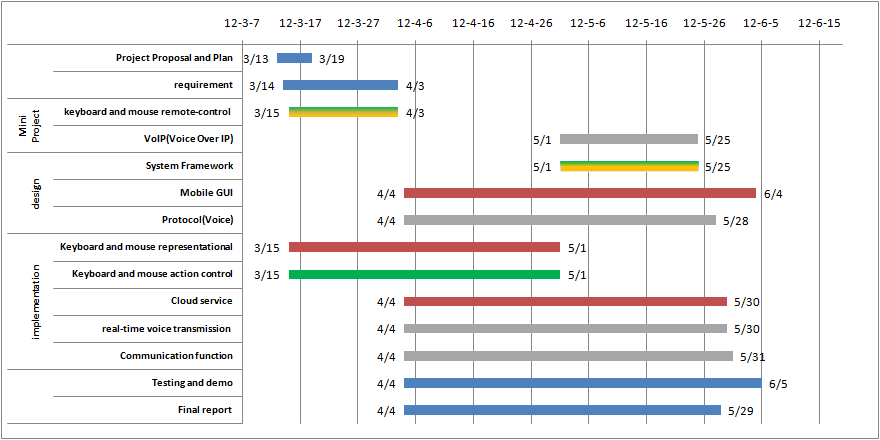


Figure 36 : Project Status

Project Proposal and Plan: Complete

Requirement: Complete

Mini Project: Keyboard and mouse remote-control – complete

VoIP – in progress.

Design:

- Protocol – in progress.

- Mobile GUI – finish structure only.

Implementation:

- Keyboard and mouse representational - Complete.

- Keyboard and mouse action control –Keyboard controller complete. Mouse controller progress

## 9.2 Reference

Applying UML and Patterns, 3rdEd.

Google Docs –Real Time Document **Collaboration**

Gliffy – diagramming tools

TeamViewer – Remote Control Collaboration

[www.cacoo.com](http://www.cacoo.com)- diagramming tools