332:438 Software Engineering

Final Report

Team:

Shravanthi Muthuraman

Kartik Bhatnagar

Rashmi Loka

Madhumitha Harishankar

Project: RIS (Remote Input Solutions)

URL:https://sites.google.com/site/ruandromouse

05/03/2013

Acknowledgements

We would like to thank our project advisor Professor Manish Parashar for his guidance throughout the project.

Table of Contents

Introduction

In today's day and age cell phones have become more of a necessity for many people throughout the world. The majority of mobile phone users own a smartphone which emphasizes the increasing popularity of smart phones. This is because these phones combine the versatility of cell phones, gaming consoles, PDAs, and personal computers into one handheld device. Smartphone is built on a mobile operating system and has more advanced computing and connectivity capability compared to the feature phones. Among the various mobile platforms available in the market, Google's Andriod OS is among the most popular platforms. Android phones have tons of exciting features and easy access to thousands of applications via Google Play, the Android market.

Though the capabilities of smartphones are growing day by day, there are certain things for which we would still need to use a computer. One of the main drawbacks of using a computer is that the user has to stay at hand with the computer. Wireless mouse and keyboards tend to solve the problem but not to an appreciable extent. They might provide mobility but cannot be handheld and require you to carry additional devices.

Taking all this into account, our project (Remote Input Solutions) is geared towards building an Android application that further exploits the features of an Android phone and increases its multi functional capabilities. The application will enhance the user friendly nature of a computer by enabling distant control. RIS provides the functionality of the two key computer input devices, the mouse and the keyboard. By making the phone an input device RIS eliminates the need for additional physical devices.

Design

The application consists of two parts: the client, and the server. The client is the computer that is controlled remotely and the server is the phone that is used to control the client remotely. The user interacts with the server through the user interface on the phone. The phone then interprets the user's actions and sends commands to the client to execute the required action.

RIS is designed to provide the functionalities of the mouse and keyboard. When the user is using the phone as a mouse the phone screen acts like a touch pad. The different functionalities of the mouse have specific gestures assigned to them. The required action can be executed on the computer by performing the corresponding gesture on the phone screen. The application supports the following functionalities of a mouse:

- Move
- Left click
- Double click
- Right click
- Scroll
- Zoom

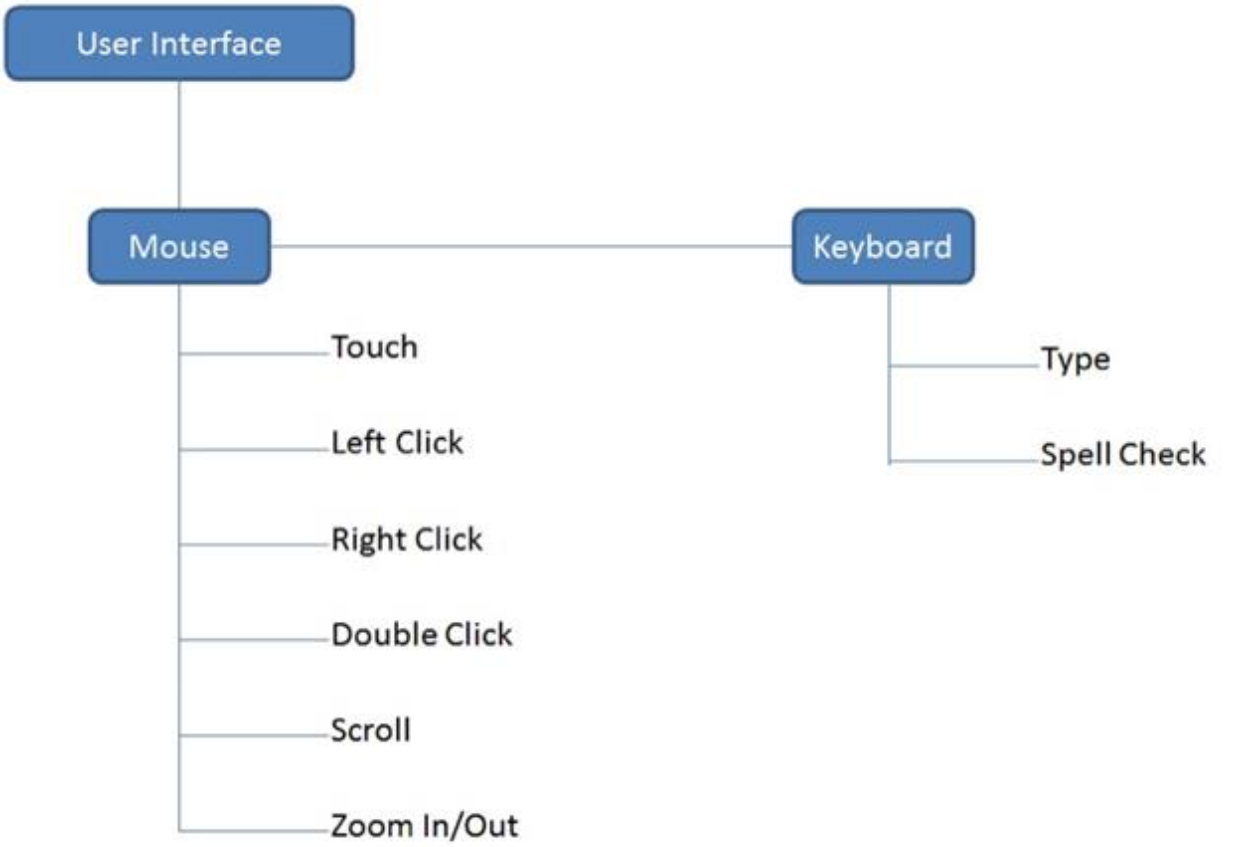
The keyboard functionality of the app lets the user to provide text input to the computer. The app also incorporates the swype functionality of the android keyboard thus enhancing the user friendly nature of the app.

Functionalities of the mouse and keyboard were handled as separate use cases. An additional use case called register was also considered. Register deals with establishing the connection between the client and the server.

Technical specifications:

1. The application is supported by Android OS 2.2 or greater
2. The client has to have a Windows (excluding Windows 8), Mac or Linux based OS.
3. The client and the server have to be connected to the same Wi-Fi network.

Conceptual Model



Implemented Use Cases:

UR-01: Register

This use case handles the work of establishing a connection between the server (the phone) and the client (the computer). When the corresponding components are installed on the computer and the phone and the application is run a listening socket is created in the phone and the phone is waiting for connections. The IP address of the phone and the port that it is listening to are displayed on the phone. In the start of the application, the computer prompts the user to enter the IP address and the port number that is displayed on the phone. Once the user enters this information the computer establishes a socket connection with the phone. This is the medium of communication between the phone and the computer. Once this step is complete the user can start using his phone as a mouse/keyboard.

UCT-02: Touch

When the user touches a specific point on the phone screen the cursor is made to appear in the corresponding point on the computer screen. Once the touch is recognized on the phone screen the coordinates of touch are retrieved. The scale factor is calculated by retrieving the dimensions of the client computer screen during the initial set up. During a mouse move, the previous position is stored and the distance the mouse moved is computed by subtracting the current and previous positions. This distance is then multiplied by the scaling factor and the command is sent to the client computer to move the cursor by the computed distance. Once you take your finger off the screen the current location is saved and when you put back your finger on the screen it starts from where you left off.

ULC-03: Left Click

The action associated with left click is single finger single tap. So when the user taps a single finger on the phone screen, a left click is executed at the current position of the cursor on the computer screen. To recognize the single finger tap we first check the number of fingers that are on the screen. If it is one finger and then we need to check the time between when the finger goes down (touches the screen) and when the finger goes up (lifts the finger from the screen). If the time interval is within the specified range we will also have to check the distance between the point where the finger went down and the point where the finger went up. If this distance is also within a specified range then we confirm that the action is a single finger single tap. Then the command is sent to the computer to perform a left click at the current position of the cursor.

URC-04: Right Click

The action associated with right click is a two finger single tap. When the user taps two fingers at once on the phone screen, a right click is executed at the current position of the cursor on the computer screen. To recognize the double finger tap we first check the number of fingers touching the phone screen. If two fingers are recognized the time difference between when the two fingers went down and then up is measured. If this time interval is within the range then we

need to check the distance between the two fingers. If the distance is also within the limits then a double finger single tap is confirmed and the command is sent to the computer to execute a right click in the current cursor position.

UDC-05: Double Click

The action associated with double click is a single finger double tap. Once the user clicks twice consecutively on the phone screen a double tap is executed on the computer screen. Once a single finger single tap is recognized the time difference between the current tap and the previous tap is calculated. If the difference is within a specific interval then the difference in distance between the current tap and previous tap is determined. If this is difference is also within a certain range then a single finger double tap is confirmed. The command is then sent to the computer screen to execute a double click in the current cursor position of the computer screen.

UCS-06: Scroll

The gesture associated with scroll is a two finger up or down motion. When the user swipes down with two fingers on the phone, the computer screen is scrolled down. When the user swipes upward with two fingers on the phone screen, the computer screen is scrolled up. When we recognize two fingers moving on the phone screen we need to determine if the two fingers are moving in the same direction. If they are moving in the same direction and the x coordinates of the points of touch remain the same while the y coordinates change then it is considered to be a scroll action. The distance between the points is measured in terms of y coordinates and the direction of move is determined. The commands corresponding to the direction of move are sent to the computer along with the scaled distance. The computer then executes the scroll for the required distance.

UCZ-07: Zoom

The gesture associated with zoom is pinch out or pinch in motion. When the user pinches out the computer screen zooms in. When the user pinches in the computer screen zooms out. When the user swipes two fingers on the phone screen we need to determine if the two fingers are moving in the opposite direction. If the fingers are moving in the opposite direction then we need to calculate the distance between the current position and the initial position when the action started. This needs to be performed for both the fingers. If the distance between them is within the specified range for zoom out then the command for zoom out is sent to the computer. If the distance between them is within the specified range for zoom in the command for zoom in is sent to the computer.

USK-08: Keyboard

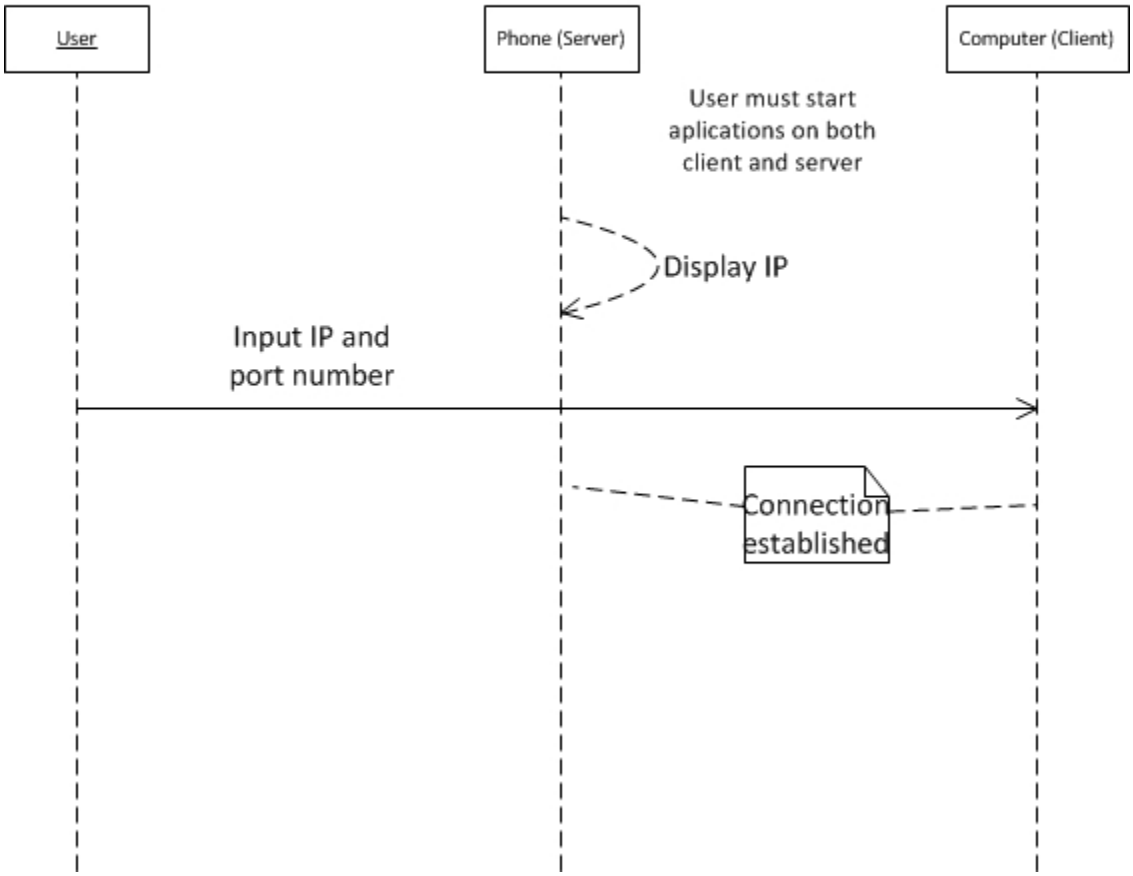
The user can bring up the keyboard by clicking on the keyboard icon provided in the main screen. When the user clicks on a key on the phone screen the letter is printed on the computer screen. When the user clicks on a key the value of the key is retrieved in terms of ASCII. The virtual key code corresponding to the ASCII value is determined using a hash map. The virtual key code is then used to instruct the computer to press the corresponding key. If it is a special key that requires a shift + operation on the computer then the key corresponding to the shift + operation on the computer is determined. The required key is then selected by executing a shift press and the corresponding key press on the computer. (For example to execute the '?' key we need to press the 'shift' and '/' keys.) When the user presses the keyboard icon again, the keyboard is minimized.

USI-10: Swype Input

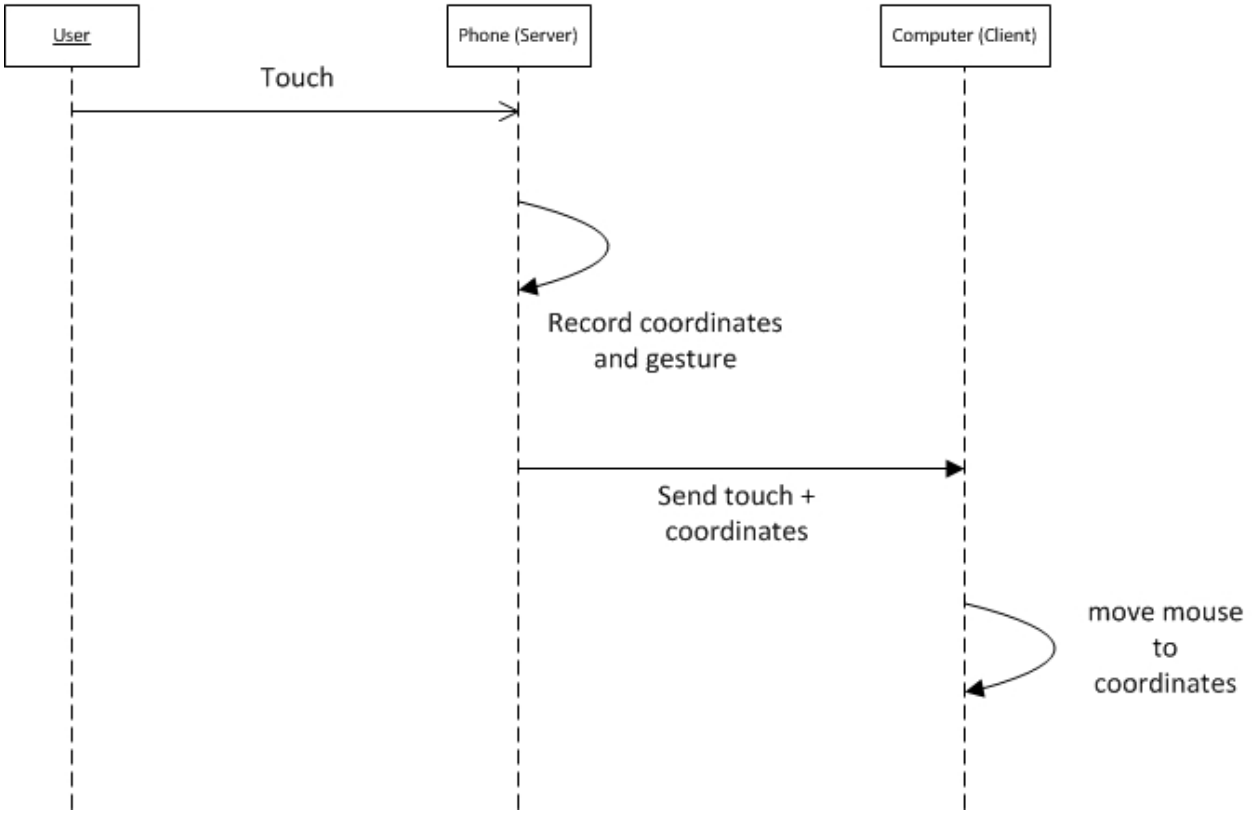
This functionality provides a convenient way for the users to provide input via keyboard. To use a Swype keyboard, a user can tap on the first letter of the word he/she wishes to type and then drag the touch on to the remaining letters of the word. The word will be typed on the screen. This allows for a convenient typing style compared to the conventional style of typing. The entire word is delivered from the swype keyboard after it is swiped by the user, and then is displayed on the text box on the phone screen. The textbox is then check to determine the change in it since the last time it was checked and the changes are sent to the computer. Thus the entire word gets reflected in the computer screen at once. When the user presses the keyboard icon again, the swype keyboard is minimized.

System Sequence Diagrams

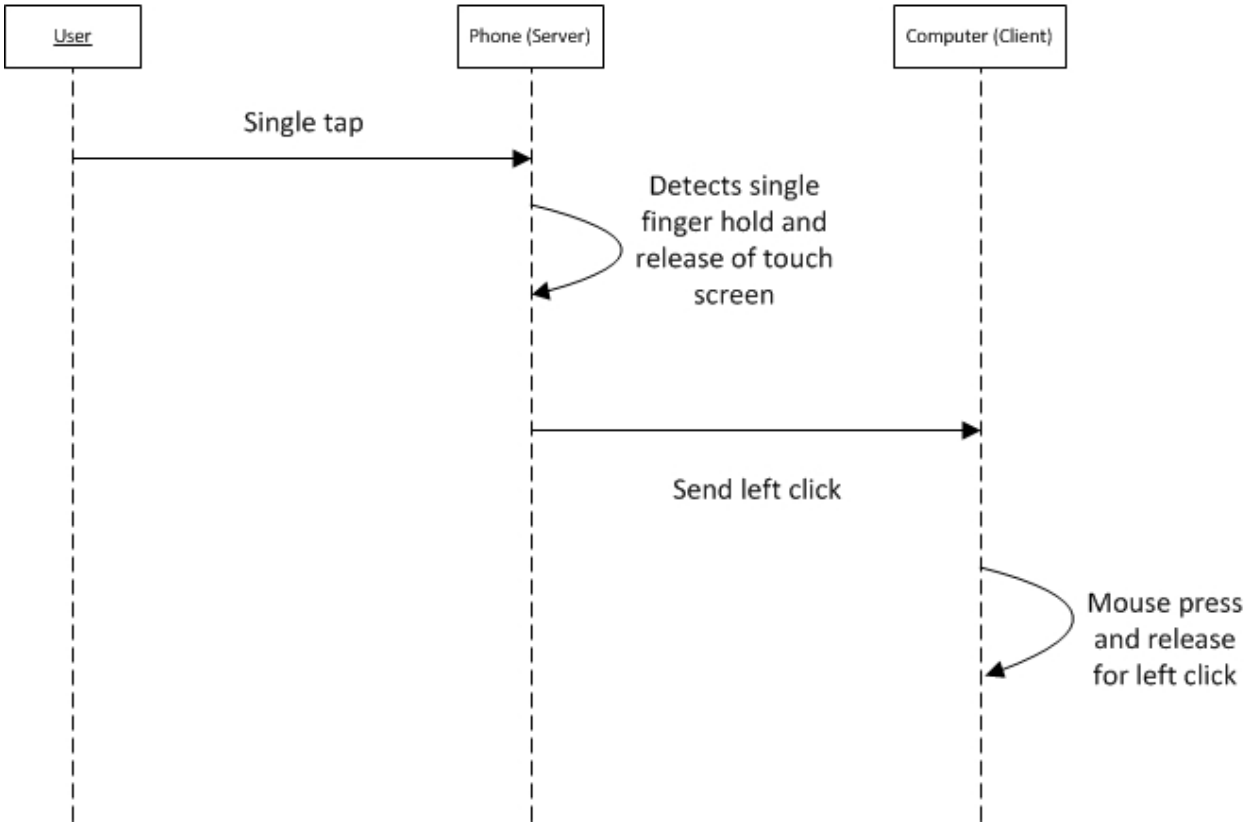
UC-01: Register



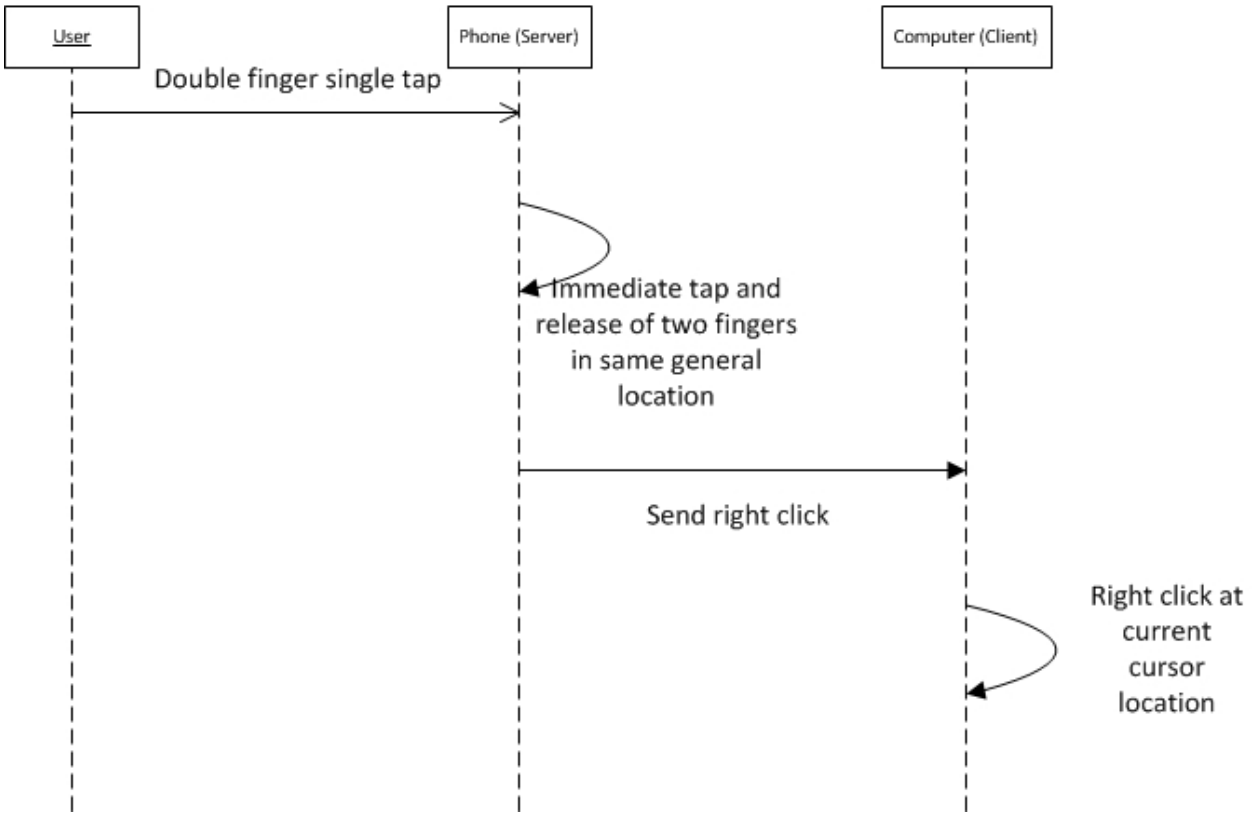
UC-02: Touch


User
Phone (Server)
Computer (Client)
Touch
Record coordinates and gesture
Send touch + coordinates
move mouse to coordinates

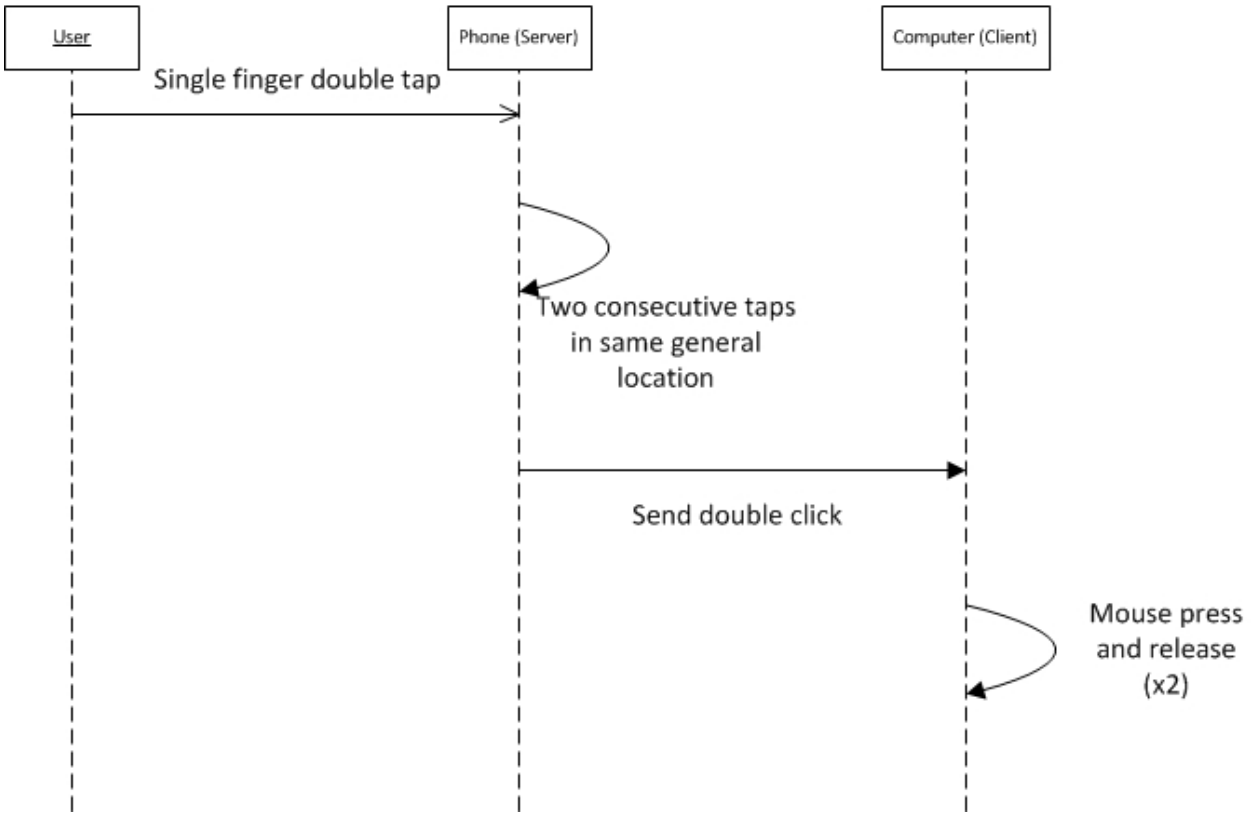
UC-03: Left Click



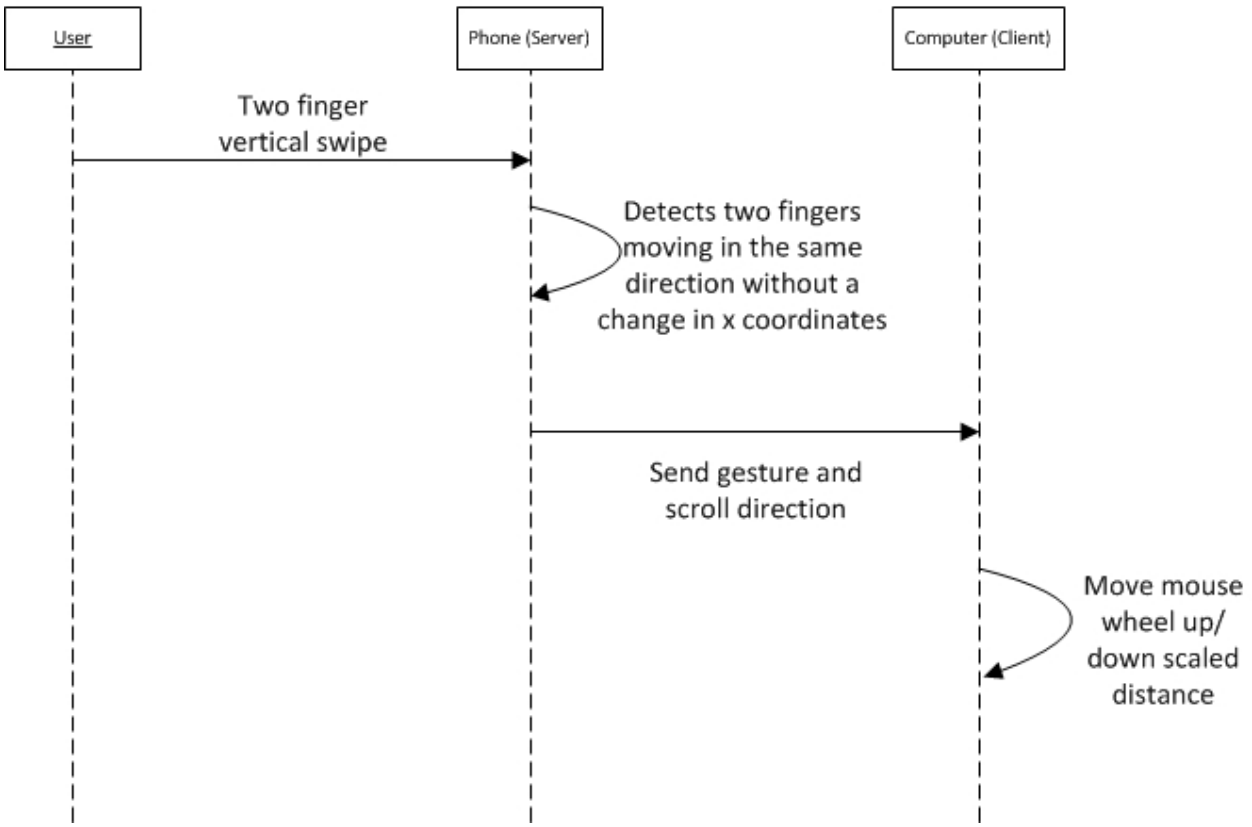
UC-04: Right-Click



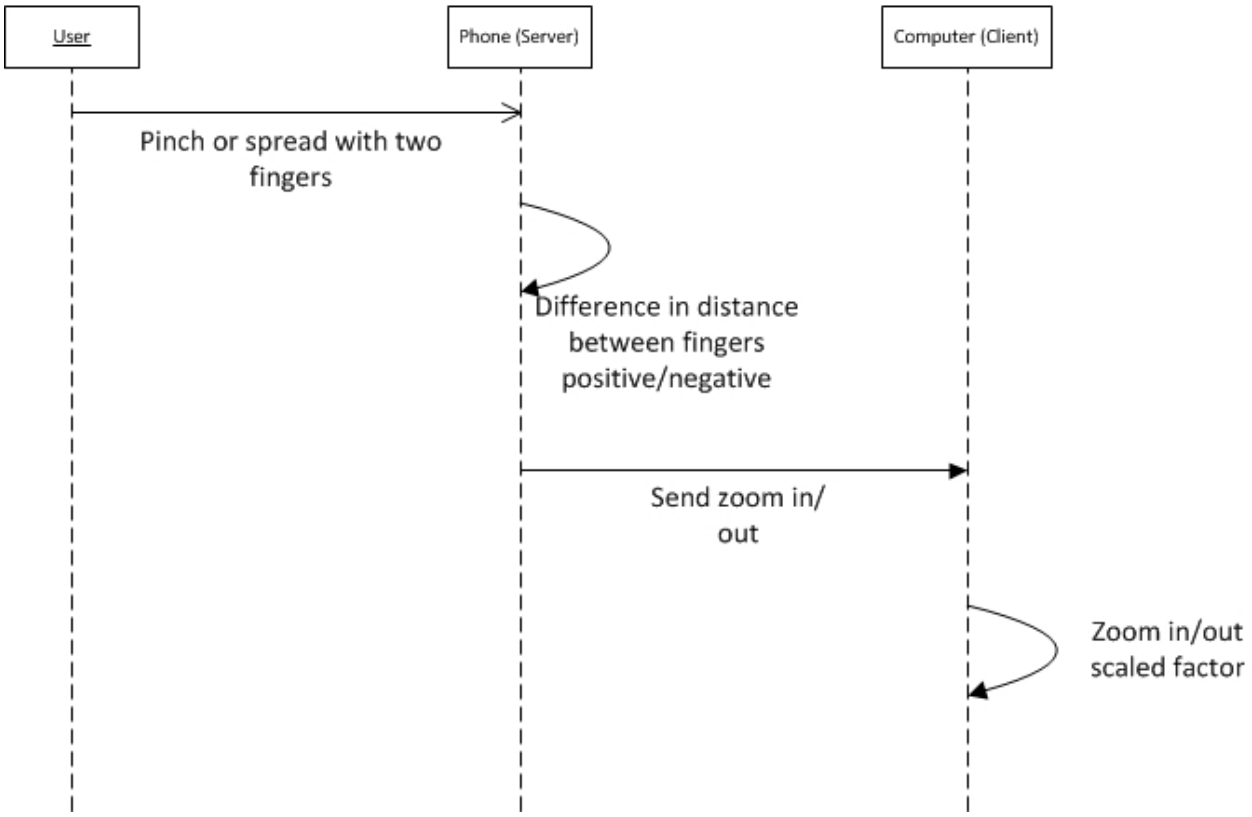
UC-05: Double Click

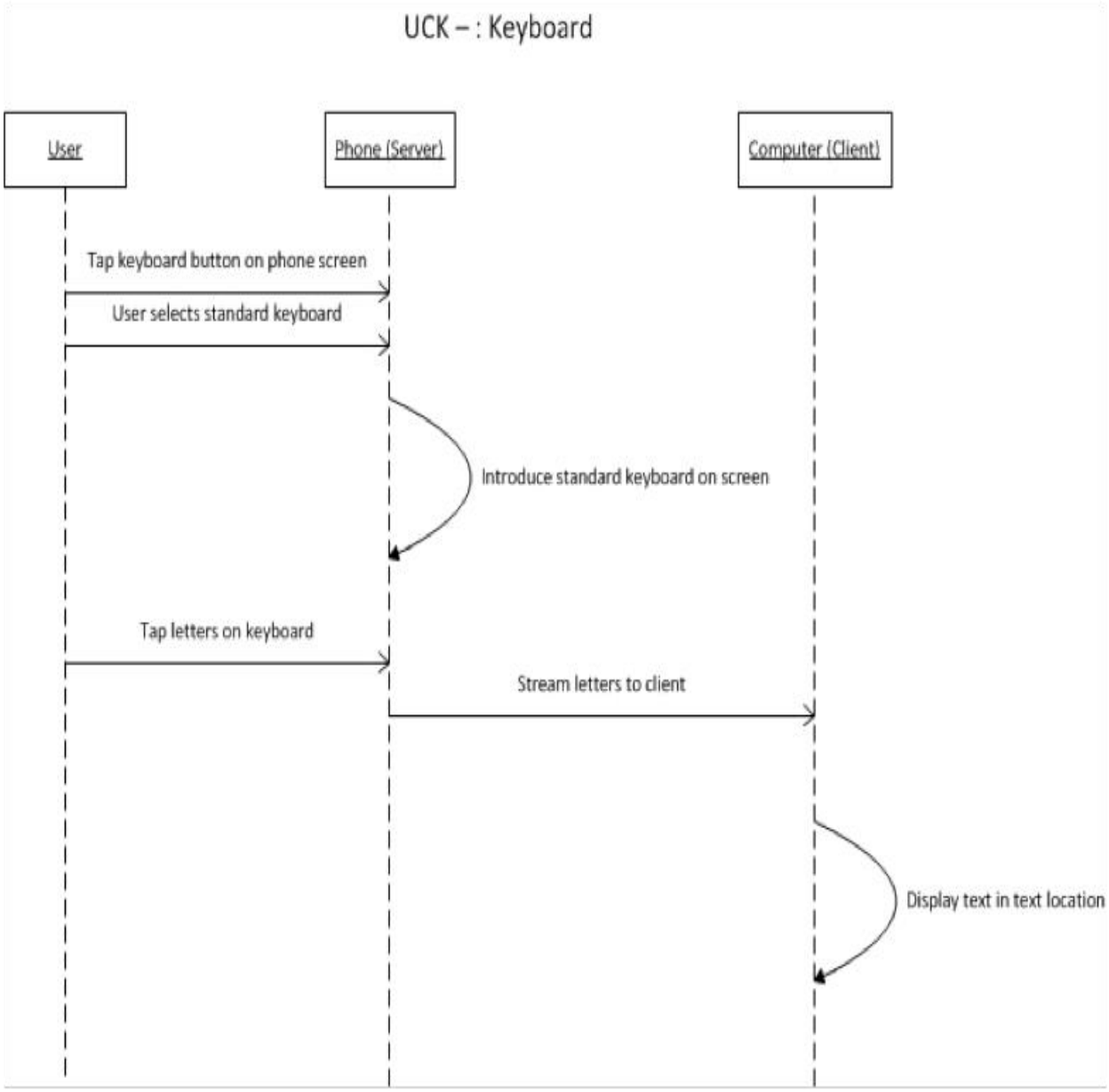


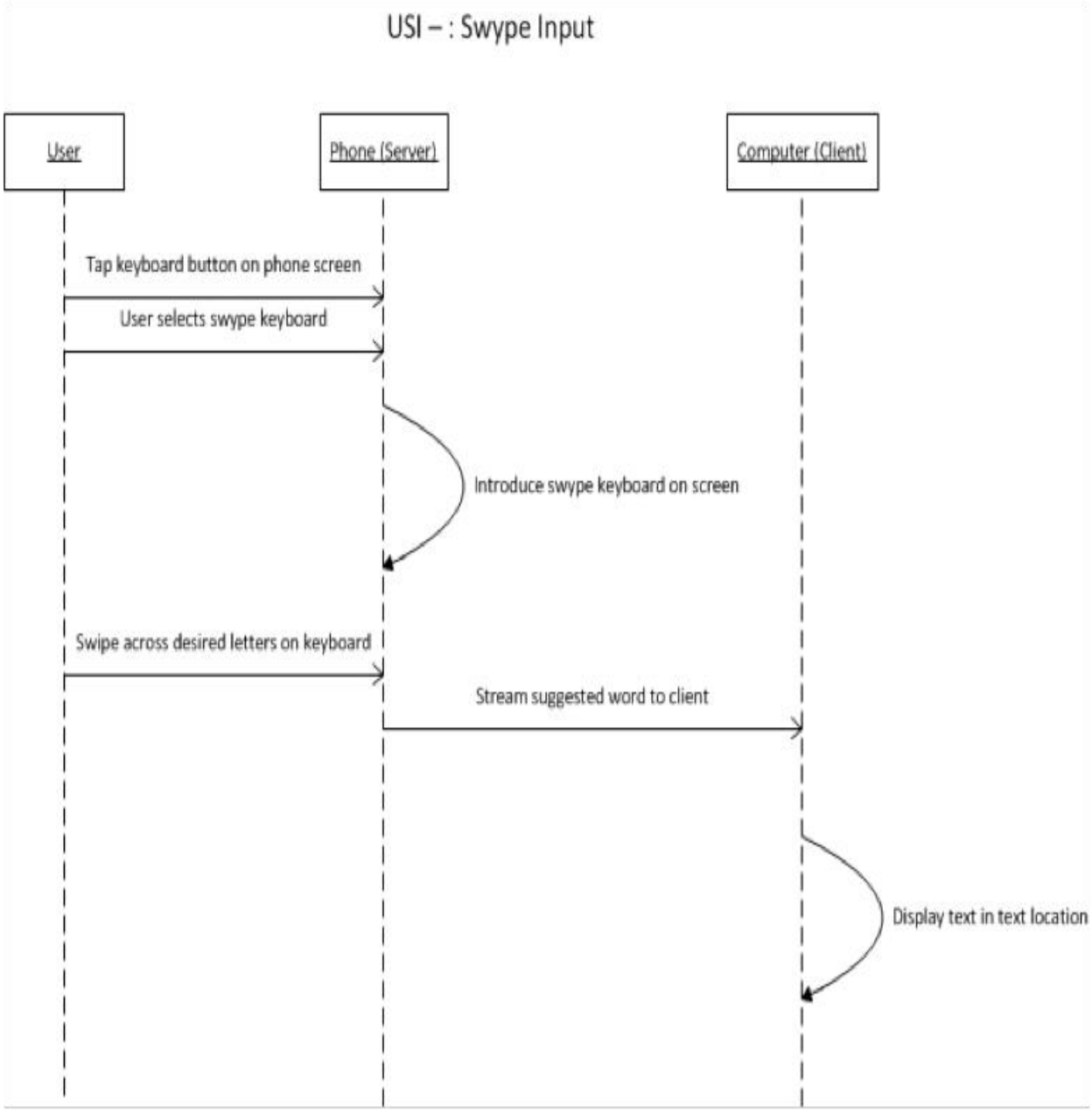
UC-06: Scroll


User
Phone (Server)
Computer (Client)
Two finger vertical swipe
Detects two fingers moving in the same direction without a change in x coordinates
Send gesture and scroll direction
Move mouse wheel up/ down scaled distance

UC-07: Zoom


User
Phone (Server)
Computer (Client)
Pinch or spread with two fingers
Difference in distance between fingers positive/negative
Send zoom in/ out
Zoom in/out scaled factor


UCK – : Keyboard
User
Phone (Server)
Computer (Client)
Tap keyboard button on phone screen
User selects standard keyboard
Introduce standard keyboard on screen
Tap letters on keyboard
Stream letters to client
Display text in text location


USI – : Swype Input
User
Phone (Server)
Computer (Client)
Tap keyboard button on phone screen
User selects swype keyboard
Introduce swype keyboard on screen
Swipe across desired letters on keyboard
Stream suggested word to client
Display text in text location

Implementation

Document Contracts

Operation:	Register
Cross Reference:	UR-01
Precondition:	The application is running on the computer and phone.
Post condition:	Connection established

Operation:	Touch
Cross Reference:	UCT-02
Precondition:	The application is running on the computer and phone. The computer and phone must be registered. The user is in the input screen on the phone.
Post condition:	Cursor is placed at the corresponding computer screen point.

Operation:	Left Click
Cross Reference:	ULC-03
Precondition:	The application is running on the computer and phone. The phone is connected to the computer using Wi-Fi. The user is in the input screen on the phone. The cursor is at some point on the computer screen.
Post condition:	The action of left click is performed on the cursor point on the computer. Current coordinates of the cursor is same as the previous cursor coordinates.

Operation:	Right Click
Cross Reference:	URC-04
Precondition:	The application is running on the computer and phone. The phone is connected to the computer using Wi-Fi. The user is in the input screen on the phone. The cursor is at some point on the computer screen.
Post condition:	The action of right click is performed on the

	cursor point on the computer. Current coordinates of the cursor is same as the previous cursor coordinates.

Operation:	Double Click
Cross Reference:	UDC-05
Precondition:	The application is running on the computer and phone. The phone is connected to the computer using Wi-Fi. The user is in the input screen on the phone. The cursor is at some point on the computer screen.
Post condition:	The action of double click is performed on the cursor point on the computer. Current coordinates of the cursor is same as the previous cursor coordinates.

Operation:	Scroll
Cross Reference:	UCS-06
Precondition:	The application is running on the computer and phone. The computer and phone must be registered. The user is in the input screen on the phone.
Post condition:	The scroll action is performed. The app running on the computer has moved corresponding swipe area.

Operation:	Zoom
Cross Reference:	UCZ-07
Precondition:	The application is running on the computer and phone. The computer and phone must be registered. Cursor must be at a point on the screen.
Post condition:	The window is zoomed in or out depending on the gesture.

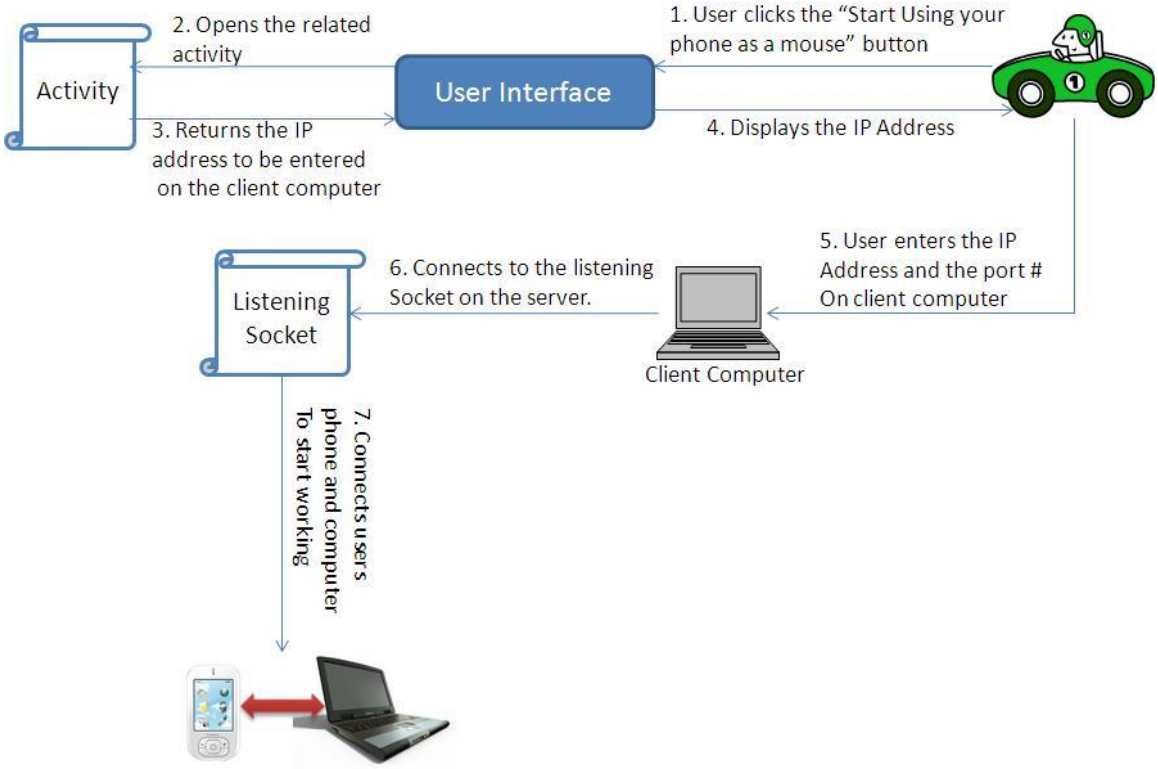
Operation:	Keyboard

Cross Reference:	USK-08
Precondition:	The user's phone needs to be connected to a Wi-Fi connection. The user's PC needs to be connected to the internet. The user's phone and PC need to be in close proximity. The user needs to have RIS application running on his/her phone and desktop The keyboard icon needs to be selected on the user's phone.
Post condition:	At the end of the use-case, the user has finished typing keyboard input. The system is running on both the phone and the computer. The phone and the computer are connected to the internet Keyboard is minimized.

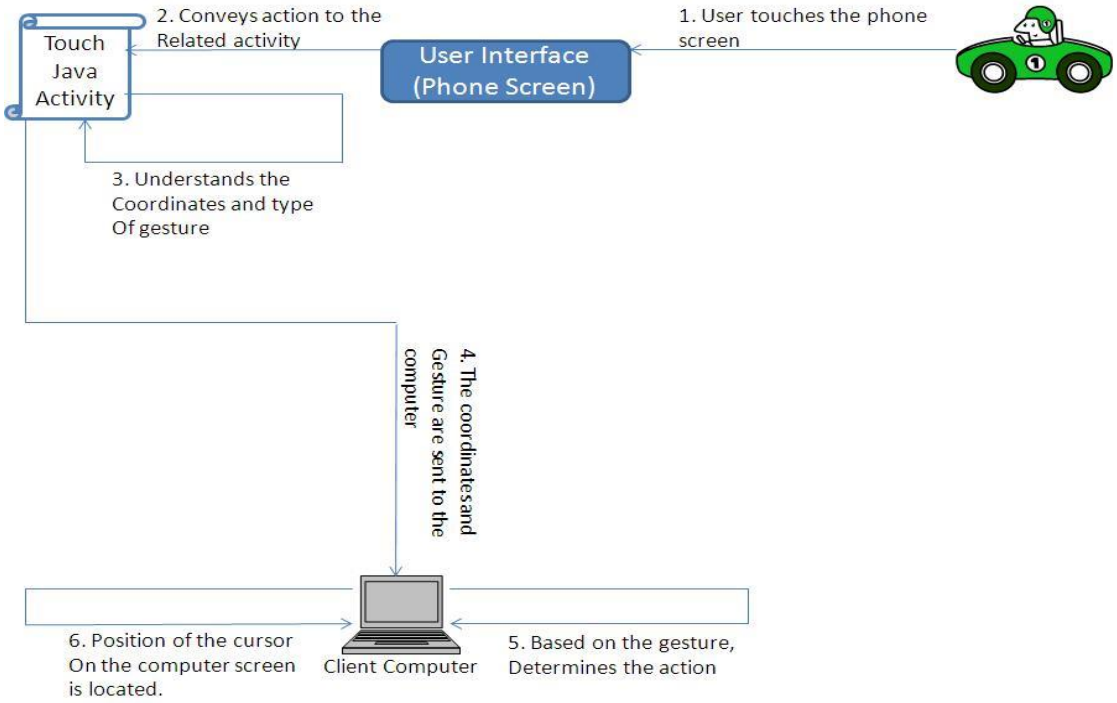
Operation:	Swype Input
Cross Reference:	USI-10
Precondition:	The user's phone needs to be connected to a Wi-Fi connection. The user's PC needs to be connected to the Internet. The user's phone and PC need to be in close proximity. The user needs to have RIS application running on the phone and the desktop. The keyboard icon needs to be selected on the user's phone.
Post condition:	At the end of the use case, the user has Swyped in all words he/she needed to type The system is running on both the phone and the computer. The phone and the computer are connected to the internet Keyboard is minimized

Collaboration Diagrams

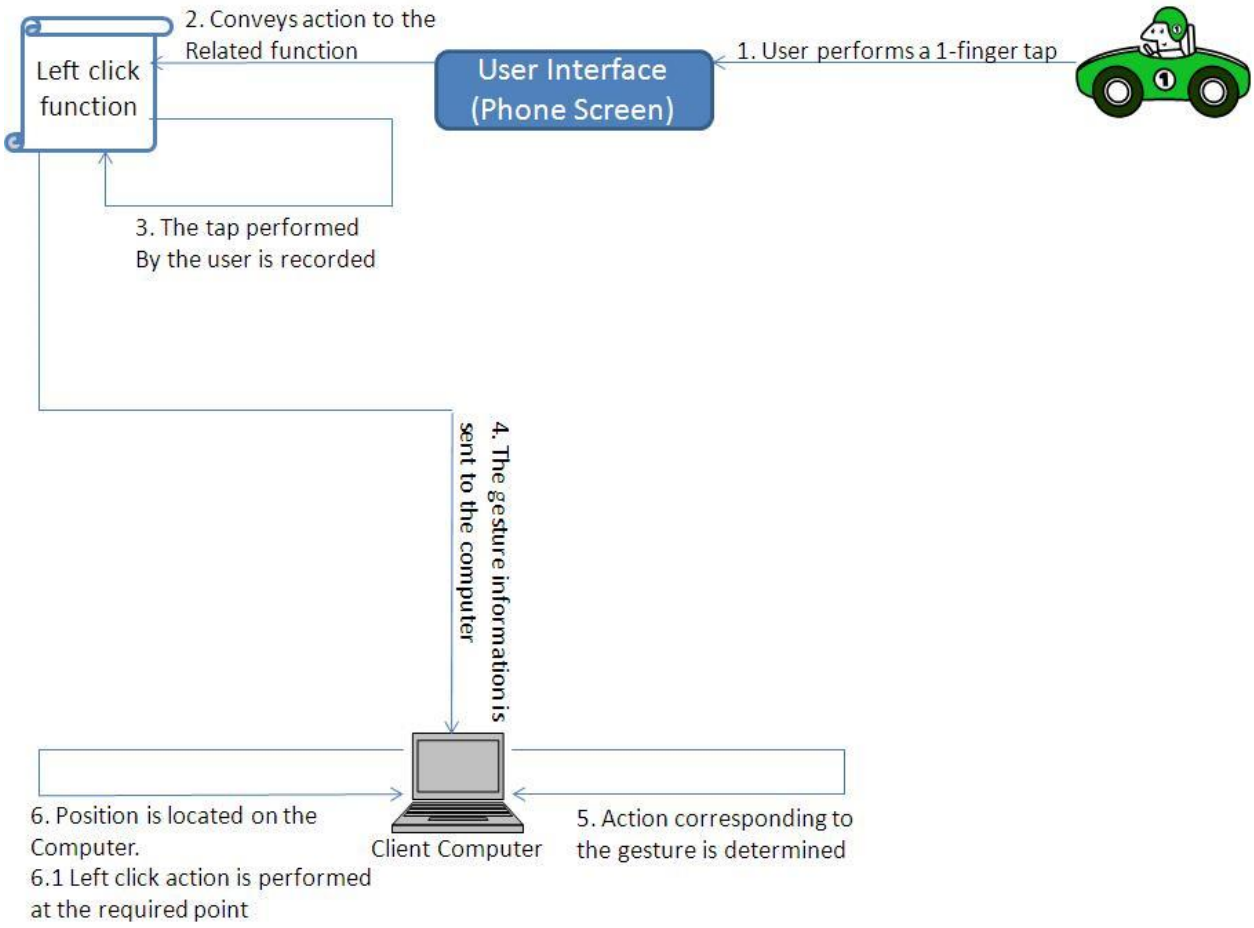
UR-01: Register



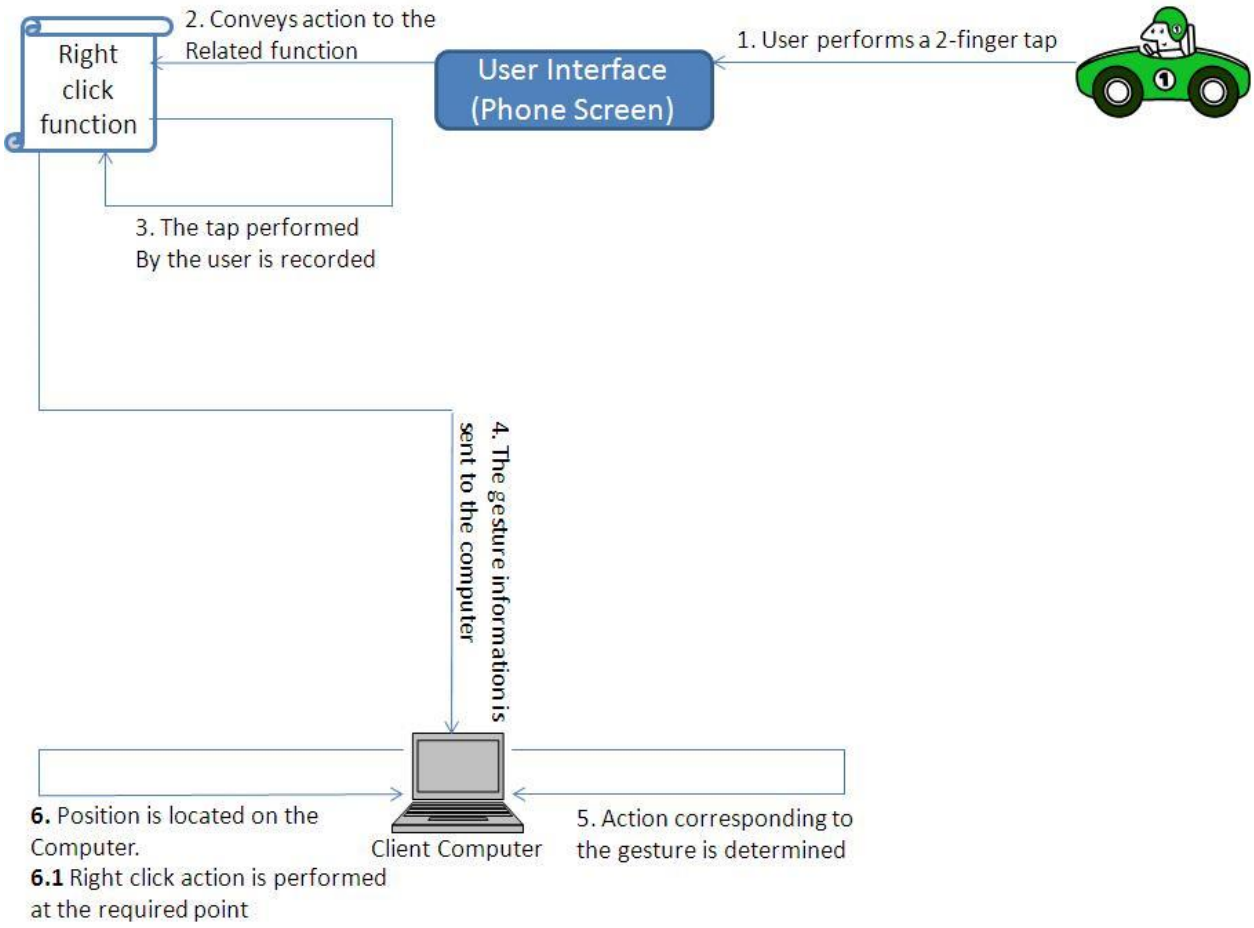
UCT-02: Touch



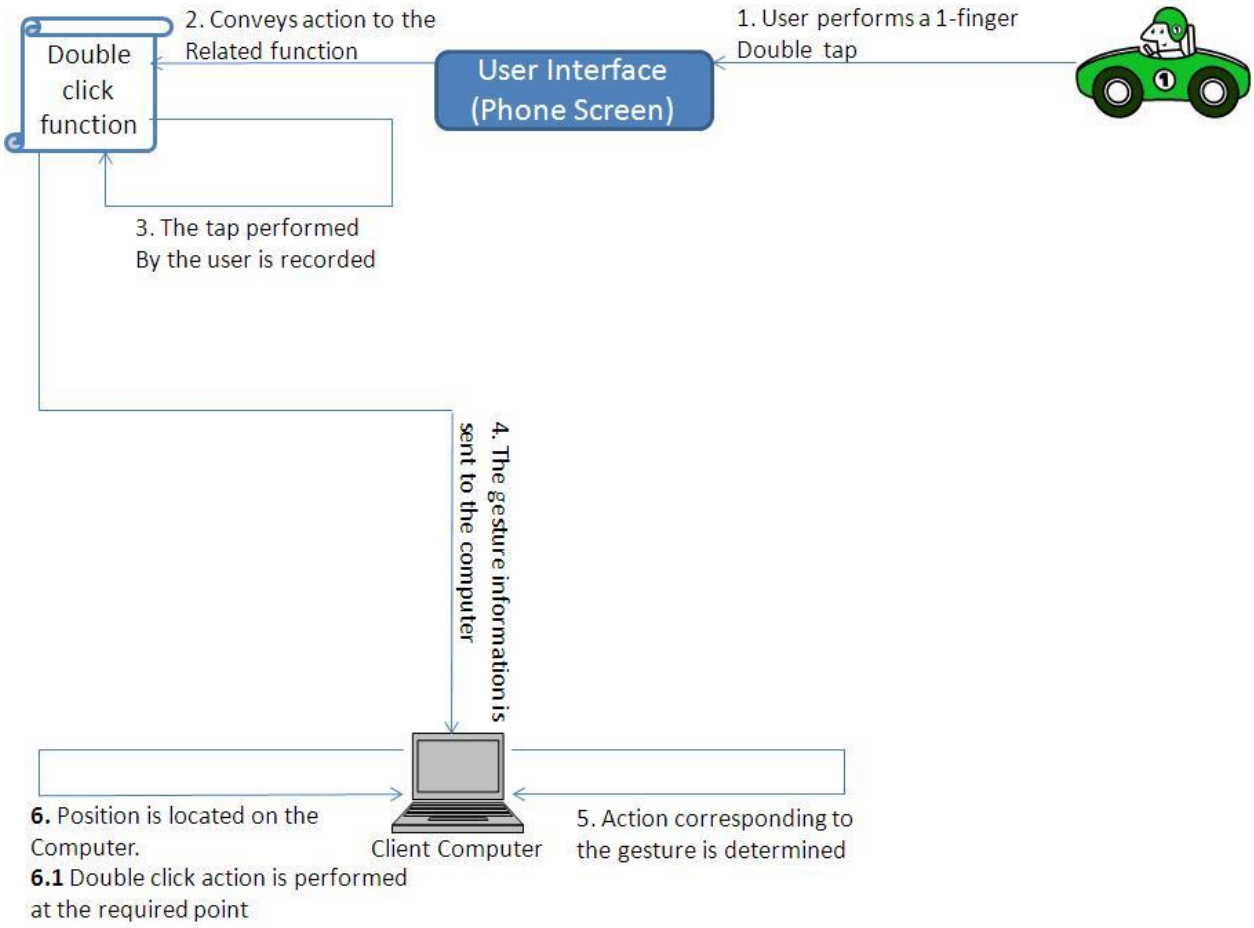
ULC-03: Left Click



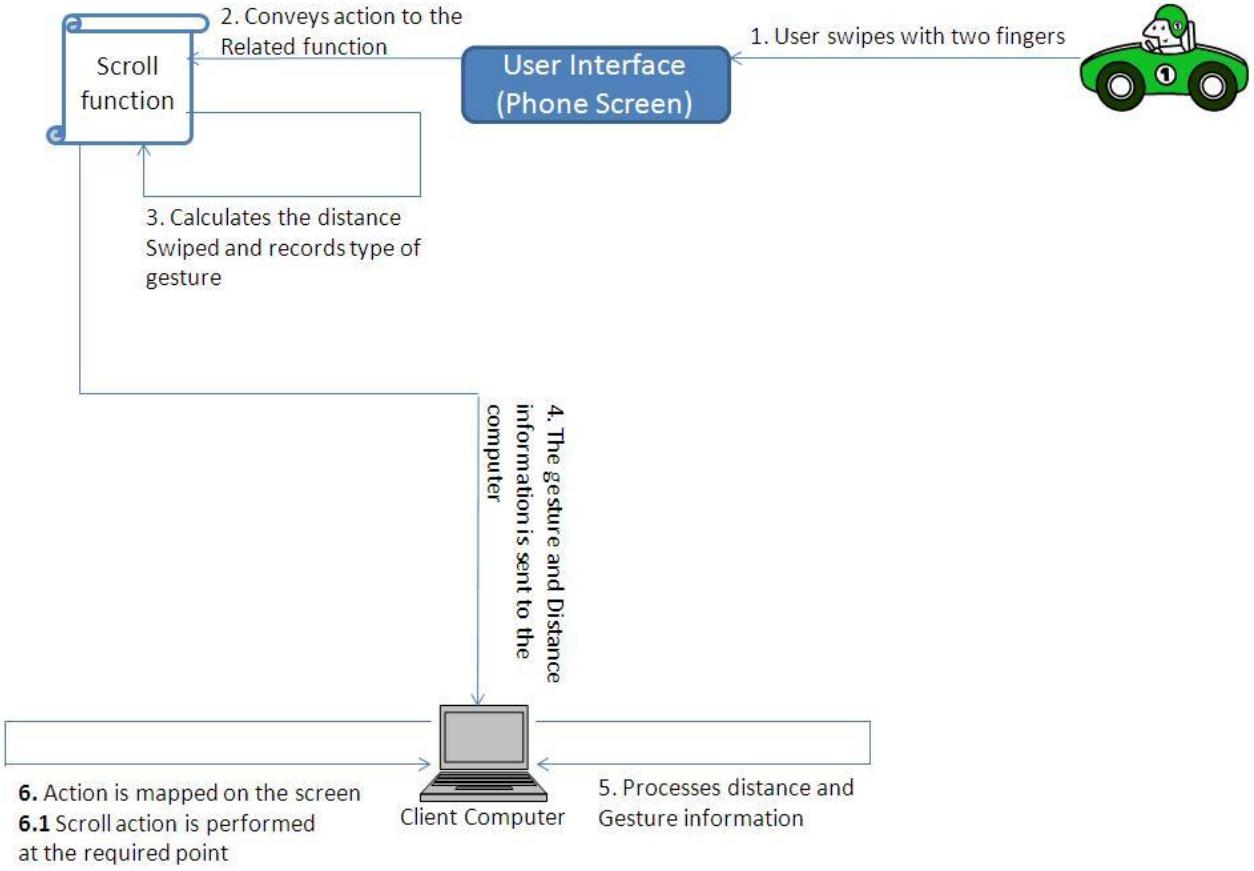
URC-04: Right Click



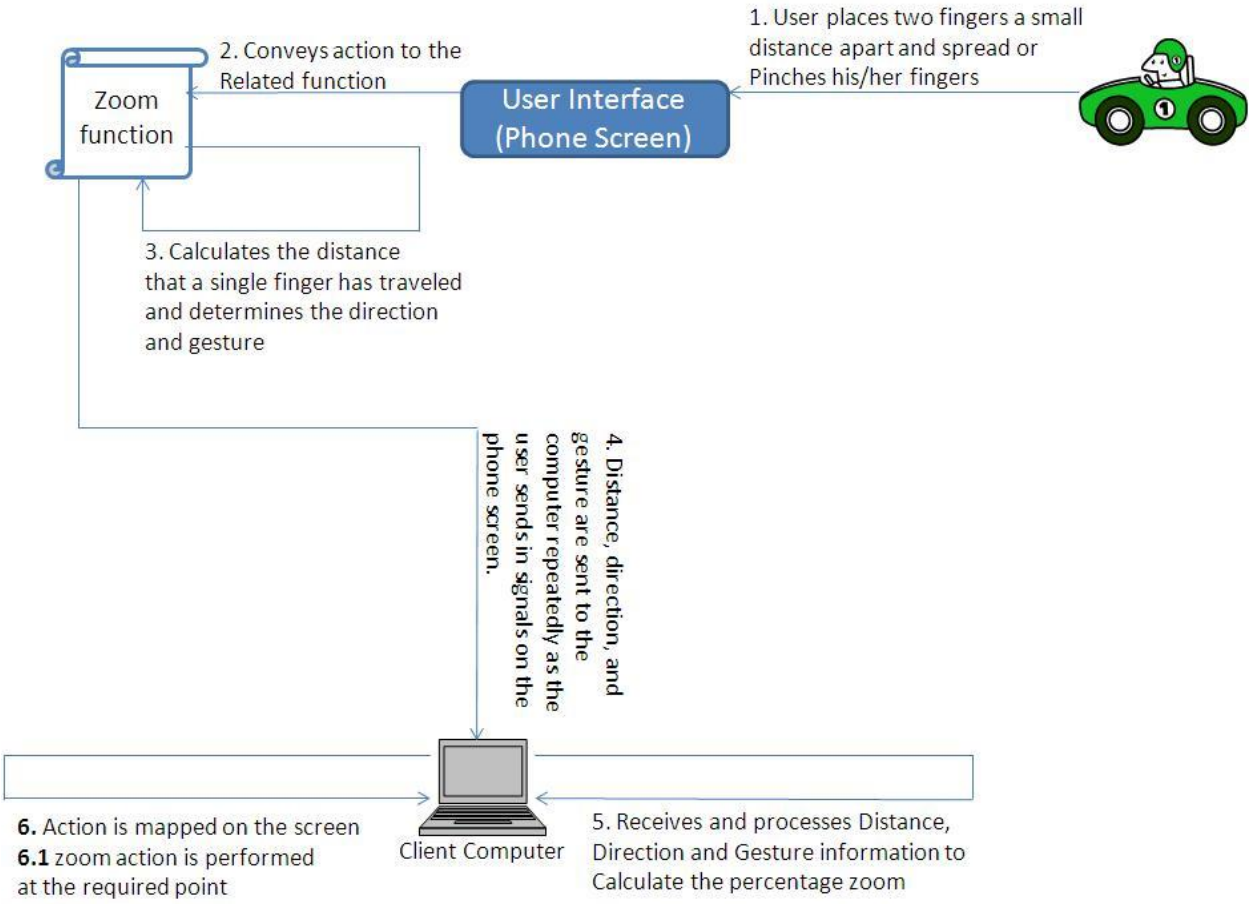
UDC-05: Double Click



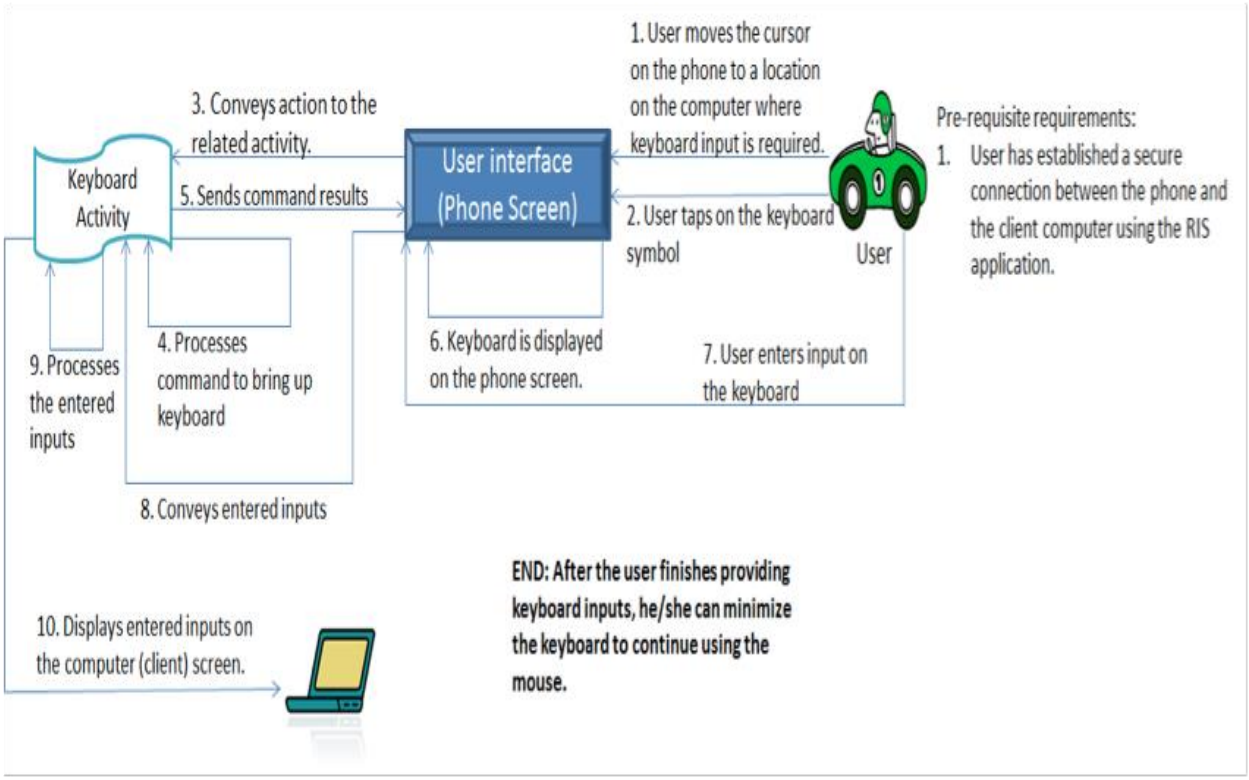
UCS-06: Scroll


Scroll function
2. Conveys action to the Related function
User Interface (Phone Screen)
1. User swipes with two fingers
3. Calculates the distance Swiped and records type of gesture
4. The gesture and Distance information is sent to the computer
Client Computer
5. Processes distance and Gesture information
6. Action is mapped on the screen
6.1 Scroll action is performed at the required point

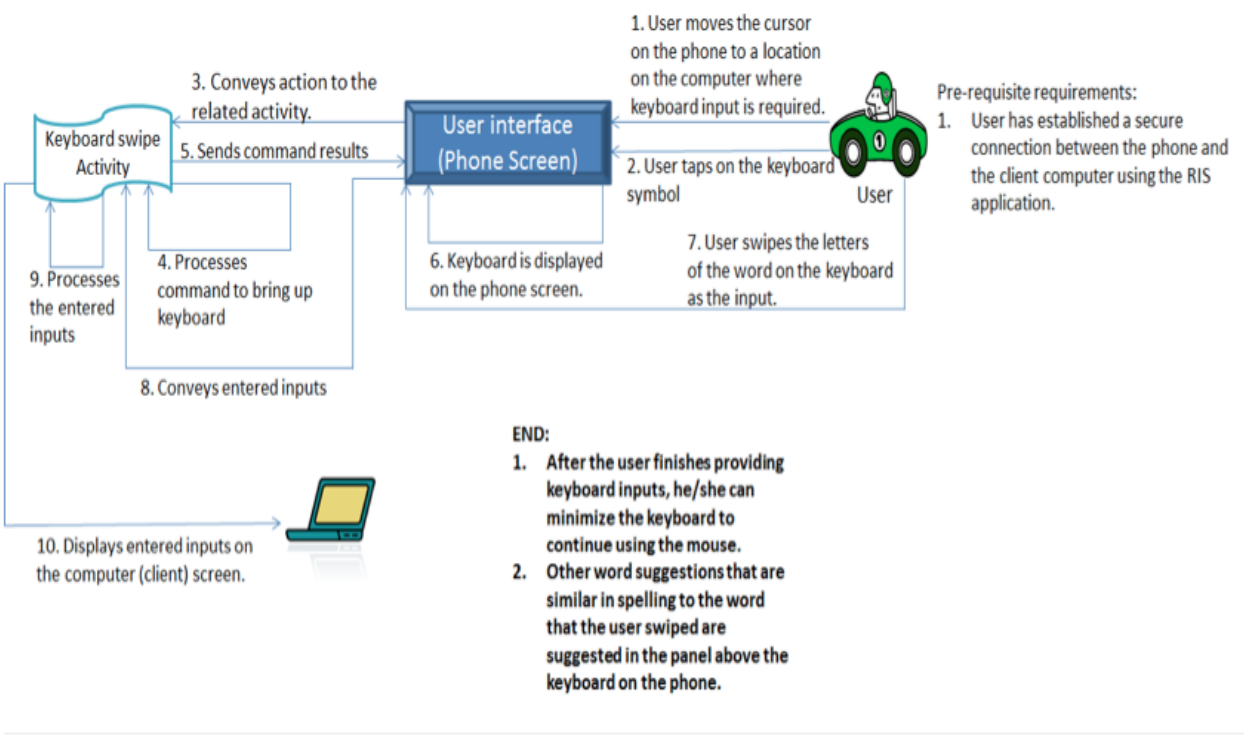
UCZ-07: Zoom



USK-08: Keyboard

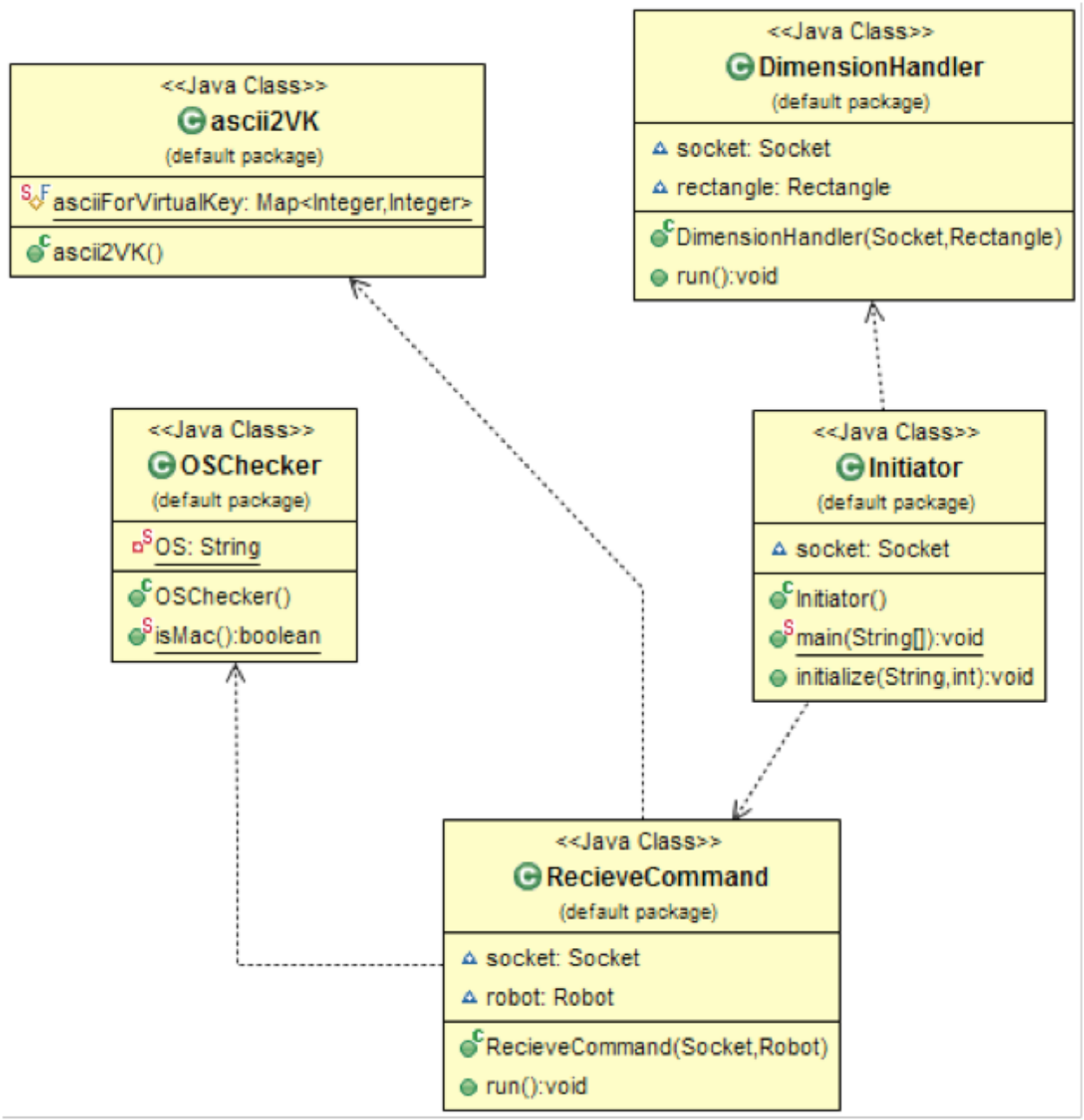


USI-10: Swype Input

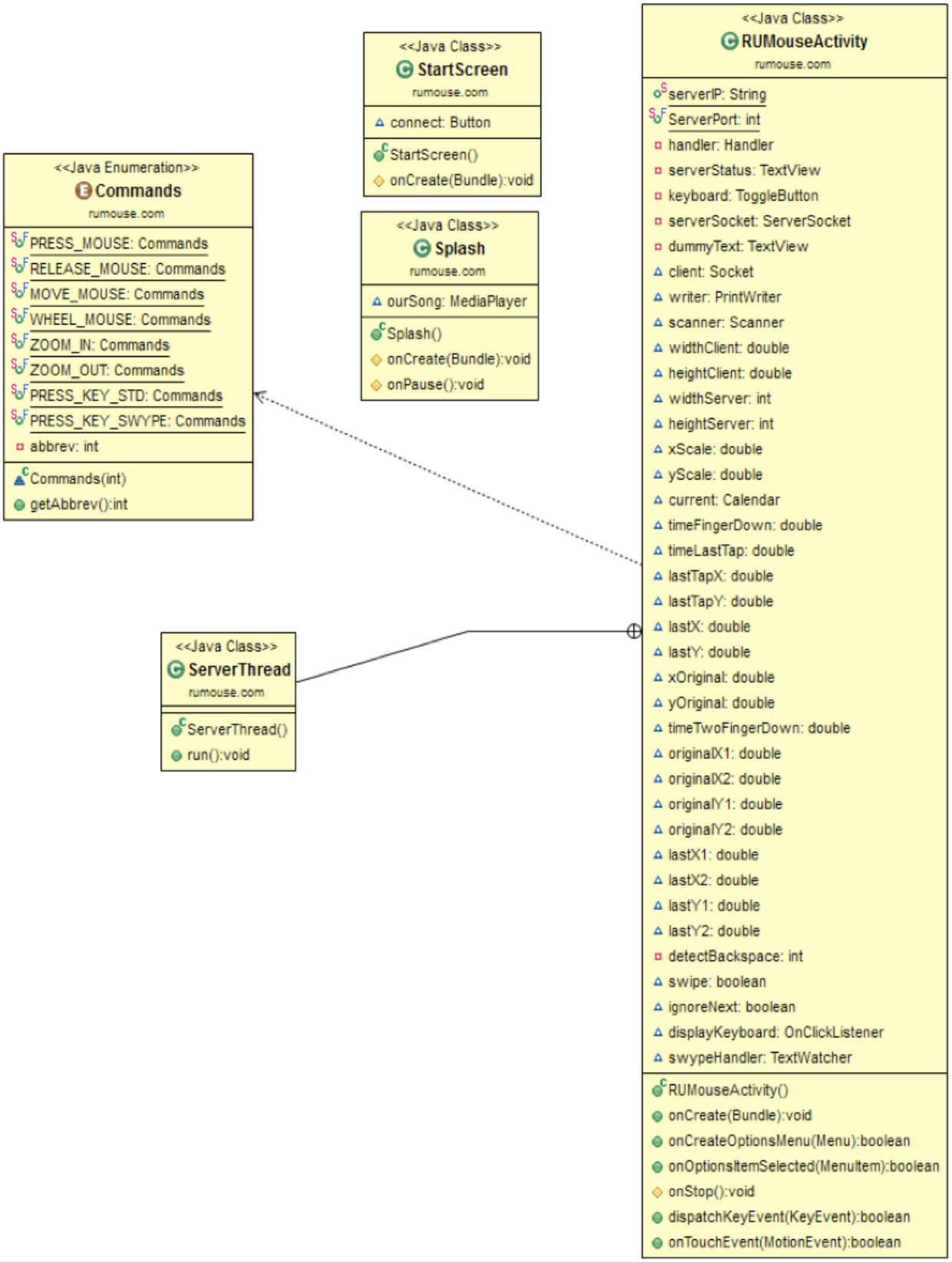


Class Diagrams

Client Class diagram:



Server Class Diagram:



System Evaluation

Our target audience would be current/future Android Smartphone/tablet users and desktop/laptop users. The application has some limitations in terms of the fact that the server and the client have to be in the same Wi-Fi network. The app doesn't die when we change orientation but the way the app works we would not obtain appreciable results in the horizontal view. This is because the app is built based on the vertical view's coordinate system. The swype functionality of the keyboard is supported only by phones that have the Android swype keyboard. All the other functionalities work on any phone that has Android OS 2.2 or higher.

Project Analysis

The Software Development Life Cycle of our project involved requirements analysis, design, implementation, and testing. Following is a brief description of the work that was done during each phase of the project:

Requirements Analysis: In this stage we analyzed and formulated the various functionalities that we aimed at supporting. Detailed research was done regarding the similar applications in the market to make sure we provide the best of currently inefficient functionalities. We made a thorough study of ways to establish connections between the phone and the computer. We initially concentrated on establishing connection using Wi-Fi direct but as we went into the further phases of development we realized that we couldn't do it without having specific hardware in the computer. Wi-Fi Direct would also be supported only by Android OS 4 or later. So we abandoned the idea of establishing connections through Wi-Fi Direct and concentrated on establishing connections by manually entering the IP address and port number. (Future work: We are currently working on establishing connections through Bluetooth and hope to have it done before the app can be released in the market.)

Design: The various gestures for the different mouse actions were determined during this stage. The various screens of the application were designed during this phase with the goal of making the app more user-friendly. The various algorithms and logic for mouse functionalities and keyboard operations were formulated and documented.

Implementation: All the plans from the design phase implemented in code. The UI for the app was built and code was written on both the client and server to establish socket connections. Once we were able to transfer data back and forth we started implementing the mouse functionalities. After all the mouse functionalities were completed we then implemented the keyboard operations and then the swype keyboard.

Testing: Continuous testing of the application was done after every additional functionality and new integrations to make sure the existing system wasn't broken by the new feature. We faced some bigger issues which would stop us from proceeding further and would require immediate attention. We also came across some inefficiency that didn't affect the basic functionality of the system. Such inefficiencies were handled as we moved forward with the project. Once the

application was complete a full system testing was carried out to make sure the system is completely functional.

Even though we strictly didn't follow the order of the software development life cycle most of the work followed the above order. We sometimes had to go back to a previous phase when we encountered problems. Detailed documentation was done during all the above phases.

Project Retrospect and Individual Contributions

This was a really exciting project to work on and provided a very good learning experience. Most of us in the team have never worked on Android projects before and this project gave us the opportunity to explore the various features that Android offers and use them effectively to suit our case. Since the project was designed and built from scratch we got to experience and handle the various stages of the software development life cycle.

Initially we had planned to use Wi-Fi direct to discover devices and establish connections between the computer and phone. Later we discovered that this wouldn't be possible without having specific hardware on the computer. So then we decided to use manual connections by inputting the IP address and the port number. We are currently working on establishing connections using Bluetooth. We expect to get this done before the app is released in the market.

We also aimed at implementing spell check while using the swype keyboard functionality but since the data is transferred instantly to the computer we might have to delete words from the computer to insert new words. This wouldn't be a problem if we are editing the recently added word. But this would be an issue if the user is attempting to edit a word in the middle of the stream because we don't have a direct access to words other than the last word on the computer screen. Thus we had to forgo the idea of implementing spell check.

Thus we learnt to completely research on an idea and try all the different possibilities before giving up on that idea.

Individual contribution:

Kartik Bhatnagar

He was responsible for building the User Interface for the application. He worked on building the various screens of the application like the home screen, welcome screen, main screen, preferences menu, and the help page. He was also involved in planning and designing the project. He also worked in documenting the various stages of the project.

Madhumitha Harishankar

She was responsible for setting up communication between the client and the server. She was involved in deriving logic for the mouse functionalities. She also worked on the code for mouse functionalities, keyboard, and swype keyboard. She also took part in planning and designing the project.

Rashmi Loka

She was responsible for receiving and interpreting data from the phone and performing the required action on the computer. She was widely involved in researching the various concepts and ideas that helped us formulate a solution to the problem at hand. She was also involved in

unit testing. She was also involved in planning and designing the project and worked on documentation.

Shravanthi Muthuraman

She was responsible for setting up communication between the client and the server. She was involved in formulating the algorithms or logic for mouse functionalities and keyboard code. She was also responsible for integrating and testing the code after various stages. She took part in planning the designing the project. She also worked on documentation during various stages.

Time Log

Time Frame (2013)	Tasks Completed
February 4th	Discussed project idea and started project proposal
February 9th	Completed official proposal for submission
February 13th	Submitted project proposal with plausible deliverables
February 23rd	Worked on the project statement
February 27th	Completed project statement with PowerPoint slides for presentation
March 2nd	Started researching Android Application development and functions that would be essential for development.
March 9th	Discussed & researched device networking – phones and computers (IP Address dependent)
March 13th	Android Application User Interface development began
March 14th	More multiple device networking research and development began
March 16th	Android UI Implementation completed. Move & Touch functionality implementation began. Client computer programming started (GUI and Socket implementation).
March 21st	Phone and Computer socket implementation completed. Continued working on touch and move functionality
March 23rd	Touch and move functionality completed. Socket implementation improved with functionality implementation.
March 24th to March 30th	Implemented/improved touch, registration, left click, right click, double click, zoom, and scroll functionalities.
March 31st to April 1st	Documentation was done with all the implemented use cases and its related diagrams. Presentation was prepared for the demonstration.
April 2nd to April 6th	Researched and implemented enabling keyboard at user's discretion.
April 8th to April 12th	Researched and implemented swype functionality.
April 13th	Tested both functionalities and some code optimization to build better functionality.
April 13th to April 15th	Second report was created with new use cases and its related diagrams. Second presentation was prepared for demonstration.
April 17th to April 20th	More research was done on how to implement Wi-Fi Direct.
April 21st to April 22nd	Further research was done on implementing spell check in the keyboard.
April 23rd	Started implementing Wi-Fi Direct and decided spell check is out of scope/impractical for this project.
April 24th to April 28th	Continued implementing Wi-Fi Direct, but decided this concept was too advanced for our project. Due to hardware limitations, it was not feasible for us to implement a feature that does not work on all the phones. Only limited amount of android phones incorporate Wi-Fi Direct API and its related functionalities.
April 29th to April 30th	Tested all the built functionalities. Fixed bugs in the code to improve the overall application and functionality performance. Edited system preferences and froze the code for demonstration on May 1st.
May1st to May 3rd	Final project report was created, project video was created, and finally the application user manual was developed to assist users in getting started with the application.

User Manual

- To use the application, please make sure that :
  - The application is downloaded on the phone
  - The client software is downloaded on the computer
  - The computer and the phone are connected on the same Wi-Fi network.

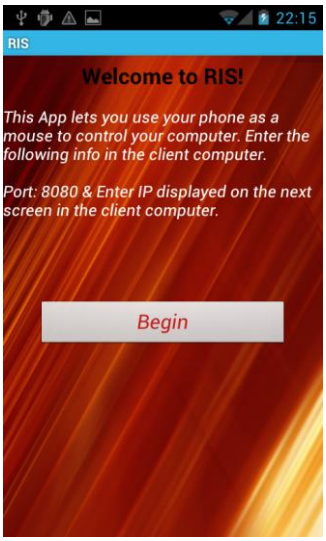
I. Open the application by click on the RIS icon **(Fig 1)**

(Fig 1)

II. Launching the application should lead to the logo screen **(Fig 2)**

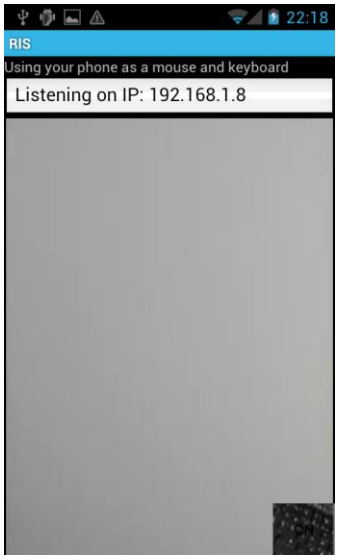
(Fig 2)

III. After 9 seconds, the application will automatically transition to the start screen **(Fig 3)**



(Fig 3)

IV. Click the **Begin** button, where an IP address will be displayed at the top of the screen **(Fig 4)**. This IP address is required to be entered in the client program (on the computer)



(Fig 4)

V. Run the client program on desktop/laptop and enter the IP address shown in **Fig 4** into the following pop-up on the computer **(Fig 5)**

(Fig 5)

VI. Click **OK** to continue.

VII. A pop-up will appear asking for the 'Server Port' number: enter **8080 (Fig 6)**

(Fig 6)

VIII. Click **OK** to continue.

IX. After couple seconds, the phone should show **Connected (Fig 7)**

(Fig 7)

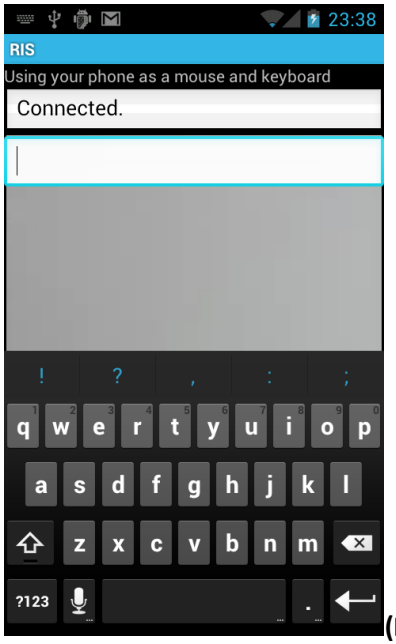
X. Once the connection has been established, the user can begin to use the phone screen as a mouse. Following operations are supported as part of the mouse functionality:

a. Left click – Single Finger tap
b. Right click – Two finger tap
c. Double click – Single finger double tap
d. Scroll – Two finger slide up/down on the phone screen
e. Zoom In – Two finger pinch out
f. Zoom Out – Two finger pinch in

XI. If typing is required, tap on the keyboard icon at the bottom right corner **(Fig 8)**

(Fig 8)

XII. The keyboard will pop-up and allow the user to type **(Fig 9)**



(Fig 9)

XIII. The application also supports Swype keyboard for Android, which can be easily activated in the phone's keyboard settings.

a. Go to settings on your phone and select a default keyboard.

XIV. The user also has access to the help and about screens which can be accessed by pressing the menu button during application use.

On menu button press:

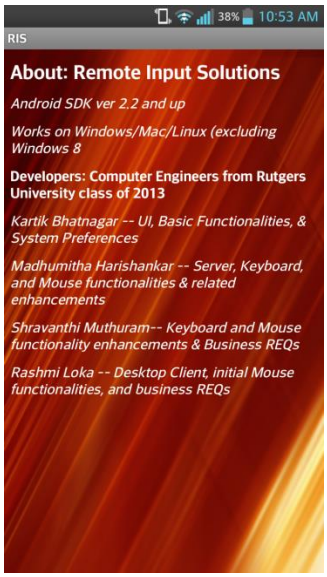
(Fig 9)

Help Screen:

(Fig 10)

About Screen:

(Fig 11)



Code

Client side:

Ascii2VK
Objects:
protected static final Map<Integer, Integer> asciiForVirtualKey

DimensionHandler
Objects:
Socket socket
Rectangle rectangle
PrintWriter writer

Methods:
public DimensionHandler(Socket socket,Rectangle rect)
public void run( )

Initiator
Objects:
Socket socket
Robot robot
Rectangle rectangle
String IP
String Port
GraphicsEnvironment gEnv
GraphicsDevice gDev
Dimension dim

Methods:
public static void main(String[] args)
public void initialize(String ip, int port)

OSChecker
Objects:
private static String *OS*
Methods:
public static boolean isMac()

ReceiveCommand
Objects:
Socket socket

Robot robot
Scanner scanner
Point current
Methods:
public RecieveCommand(Socket socket, Robot robot)
public void run()

Server Side:

Commands
Objects:
public enum Commands
Methods:
Commands (int abbrev)
public int getAbbrev()

RUMouseActivity
Objects:
public static String *serverIP*
public static final int *ServerPort*
private Handler handler
private TextView serverStatus
private ToggleButton keyboard
private ServerSocket serverSocket
private TextView dummyText
Socket client
PrintWriter writer
Scanner scanner
Calender current
Display display
WifiManager wim
Thread fst
MenuInflater inflater
InputMethodManager mgr
TextWatcher swypeHandler

Methods:
public void onCreate(Bundle savedInstanceState) - overridden
public boolean onCreateOptionsMenu(Menu menu) - overridden
public boolean onOptionsItemSelected(MenuItem item) - overridden

public void run( )
protected void onStop( ) - overridden
public boolean dispatchKeyEvent (KeyEvent event) - overridden
public void onClick(View v) - overridden
public boolean onTouchEvent(MotionEvent event) - overridden
TextWatcher ( ) - overridden
public void afterTextChanged (Editable s) - overridden
public void beforeTextChanged ( )-overridden
public void onTextChanged ( ) - overridden

Nested class:
public class ServerThread implements Runnable

Splash
Objects:
MediaPlayer ourSong
Thread timer
Intent openStart

Methods:
protected void onCreate(Bundle splash_bundle) - overridden
protected void onPause( ) - overridden
start ( )
public void run ( )

StartScreen
Objects:
Button connect

Methods:
protected void onCreate(Bundle savedInstanceState) - overridden
onClick ( )-overridden

Server side android manifest code:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="rumouse.com"
        android:versionCode="1"
        android:versionName="1.0" >

        <uses-sdk android:minSdkVersion="8" />
        <uses-permission android:name="android.permission.INTERNET" />
```

```
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />

    <application
  android:icon="@drawable/ic_launcher"
    android:label="@string/app_name" >
    <activity
        android:name=".Splash"
    android:label="@string/app_name" >
        <intent-filter>
      <action android:name="android.intent.action.MAIN" />

      <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    </activity>
    <activity
        android:name=".StartScreen"
    android:label="@string/app_name" >
        <intent-filter>
      <action android:name="rumouse.com.STARTSCREEN" />

      <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
    </activity>

    <activity
        android:name=".RUMouseActivity"
    android:label="@string/app_name" >
        <intent-filter>
      <action android:name="rumouse.com.RUMOUSE" />

      <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
    </activity>
    </application>

</manifest>
```