Bachelor's Thesis (UAS)

Information Technology

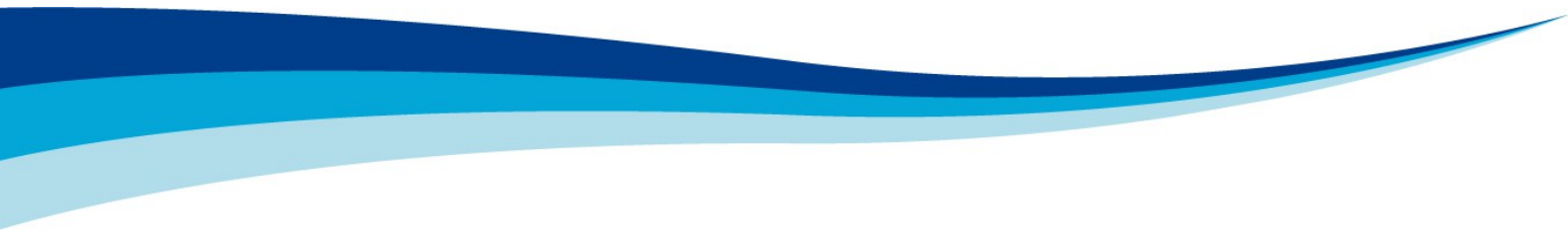
Internet Technology

2014

Anthony Nwannabuogwu

FINMOBILE CALL ANALYZER

– An Android application



BACHELOR'S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information Technology | Internet Technology

2014 | 51 Pages

Instructor: Patric Granholm

Anthony Nwannabuogwu

FINMOBILE CALL ANALYZER

Currently in Finland, there are about five major mobile network operators and due to the competitive nature of these networks it is becoming increasingly difficult for users to select which of these networks would be most suitable for them. When users finally decide on a network, picking the right call plan that would best suit their needs taking cost and their call history into consideration becomes another nightmare.

The main objective of this thesis has been to use Java programming to develop an Android application that will use a rule-based system to determine as well as recommended to users which Finnish mobile phone network will be most suitable for them.

The Android application which we named "FinMobile Call Analyzer" was successfully developed and tested on an Android device.

With the successful development and testing of FinMobile Call Analyzer, users will be confident choosing a particular Finnish mobile network knowing that their call log history was taken into account.

KEYWORDS:

Android

FinMobile Call Analyzer

Java

Rule-based system

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

Acknowledgement

During the development of this final year project, I encountered several setbacks which I overcame with the help of some people and companies both here in Finland and abroad.

I would like to thank first the companies including DNA, Sonera, Elisa and Saunalahti for the information they gave to me which enabled me to complete this project. Most importantly, I would like to thank my family and friends for all their support and encouragement during the time I spent developing this project and also finding workarounds to the various drawbacks.

CONTENTS

FIGURES

TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

API	Application programming interface
ARM	ARM microprocessor architecture
GUI	Graphical user interface
HTTP	Hypertext transfer protocol
IDE	Integrated development environment
MIPS	Microprocessor without interlocked pipeline stages
OS	Operating system
PHP	Hypertext pre-processor
SDK	Standard development kit
x86	32 bit Intel processor architecture
xml	Extensible mark-up language

1 INTRODUCTION

Android operating system devices are gaining huge popularity as the age of information technology grows making many IT developers to give mobile applications a priority when developing and implementing an idea.

A few reasons why the Android operating system is growing fast is because it is an open source Linux-based operating system with a powerful development framework where developers can create a single application for several targeted devices (such as mobile phones, tablets, television etc.) and an open marketplace for distributing applications.

There are several numbers of mobile call plans being offered by mobile operators in Finland but none of these call plans take into account the subscribers call history. All these operators do is try to get the better of one another by using pricing to lure customers.
FinMobile Call Analyzer guides subscribers to making the right choice. In other to achieve this, FinMobile Call Analyzer first determines if the SIM card in the Android device is of Finnish locale and if it is, it then collects all the available call logs data from the Android device and filters them to only outgoing calls made to Finnish mobile numbers.

The outgoing calls to Finnish mobile numbers are sorted by an external library which checks if a specific phone number is “MOBILE” and if the number is a valid Finnish mobile number. If both of these conditions are true, the valid Finnish mobile numbers are then sorted into the three major Finnish mobile network operators (DNA, Elisa and Sonera) by checking the first three to five digits.
As the application (FinMobile Call Analyzer) does this, it adds up each call duration (s) taking into account the specific date the call was placed. As it collates the dates each call was placed, it counts the total number of days calls

were made and this is used together with the total call duration to calculate the monthly average call duration to each mobile network.

Using the monthly average total call duration and the monthly average total call duration to each mobile network operator, the application then recommends which of the mobile network operator is best suitable for the user of that particular Android device.

1.1 ANDROID OS

An Android operating system is a Linux kernel-based system developed and maintained by the Open Handset Alliance/Google. It is an open source mobile operating system with a market share of 81.3%, making it the largest mobile platform in the world.

Devices running Android OS have ARM, MIPS and X86 supported CPU architecture. Android OS is programmed in C, C++ and Java, while applications that run on it are developed using Java programming (Wikipedia 2013).

1.2 VERSIONS OF ANDROID

Together with the high market share, there is also an increase in the various versions of the Android operating system. The first commercial version was the 1.0 otherwise now known as “Alpha” which was released in November 2008. Immediately after this, the Beta version was released (version 1.1).
As an operating system, it is still undergoing developments by both the Open Handset Alliance and Google and newer versions are now being developed under a codename and released in an alphabetical order.

Table . Android versions with their codenames

Codename	Version	API Level
Cup Cake	1.5	3
Donut	1.6	4
Éclair	2.0-2.1	7
Froyo	2.2-2.2.3	8
Ginger Bread	2.3-2.3.7	9,10
Honey Comb	3.0-3.2.6	11-13
IceCream Sandwich	4.0-4.0.4	14,15
JellyBean	4.1-4.3	16-18
Kit kat	4.4	19

1.3 ANDROID IDE AND VIRTUAL MACHINE

An IDE basically consists of a source code editor with a debugger and together with a compiler and interpreter helps a programmer to develop and run applications.

In the case of Android, the most used IDE is Eclipse. In other to develop an Android application using Eclipse, a programmer needs to download the software development kit (SDK) from Google and integrate it to Eclipse.

Other Android IDEs are Netbeans, Android Developer Tool and Android Studio.

An Android Virtual machine helps a programmer to run, test, and debug a developed application without having the actual Android device. The Android virtual machine is created from the SDK by specifying what version of Android

the programmer wants the application to target.

1.4 THE “LIBPHONENUMBER” EXTERNAL LIBRARY

This is a Java library developed by a team of developers to help programmers manage and handle phone number data. The importance of this library has driven many developers to develop and make it readily available to other programming languages including PHP, JavaScript, C#, C++ to mention but a few.

This library helps to parse, format, and validate phone number data by using several variable data types. For the purpose of this application (FinMobile Call Analyzer), the data types used which were of great significance include:

- PhoneNumberUtil- This variable helps to instantiate all the functions and keywords associated with library to be used. Without it, certain keywords would produce a runtime error during the execution of this application.

- PhoneNumber- This variable parses two strings in the form of a number and an international standardization organisation (iso) code to form a phone number. This data type is necessary for this application (FinMobile Call Analyzer) because the contents of the *”numbercolumn”* index (see page 42 line 51) is a string type and this value cannot be easily validated as a phone number, so parsing the string converts it to a phone number.

- PhoneNumberType- This variable classifies the type of phone number, i.e., whether it is a mobile, landline or a toll-free number. Since this application (FinMobile Call Analyzer) analyses only mobile phone numbers, this data type is important to check the string the string parsed to a*”PhoneNumber”* is a mobile number.

2. APPLICATION DESIGN AND DEVELOPMENT

FinMobile Call Analyzer is an application meant only for Finnish locale with the sole purpose of guiding, assisting and recommending which mobile phone network operator a user should choose. The design and development process was divided into two phases.

The first phase of the application was the design phase which in this case is the Graphical user interface while the second phase is the coding and development phase.

2.1 THE GRAPHICAL USER INTERFACE (GUI)

This application targets a minimum SDK/API level of 11 which is the Android version 3.0 otherwise known as "*HoneyComb*". This application has two GUIs which were developed using *.xml* format and the graphical layout. Each of the GUIs has a main layout which acts like a container. In our case, we used both a "*RelativeLayout*" and a"*LinearLayout*".

The first GUI *.xml* file is the "*main_call_analyzer.xml"* (see page 30 line 3) which displays to the user information about the call log duration on the user's Android device. This information includes the total durations, the time period that the application has analysed, the monthly average total duration for that period, the total call durations to Elisa, DNA and Saunalahti networks and the application's recommendation (s) for the user.

This GUI contains several text views and a button all with different Android attributes.

Table . Description of Android ids

Android id	Description/Use
"@+id/textview_description"	Displays the application's description
"@+id/textView_timeperiod"	Displays the time period of the Analysis
"@+id/textView_totalduration"	Displays the total call durations
"@+id/textView_AvgtotalDuration"	Displays the monthly average total call Duration
"@+id/textView_durationDna"	Displays the total call duration to DNA mobile network
"@+id/textView_durationElisa"	Displays the total call duration to Elisa mobile network
"@+id/textView_durationSonera"	Displays total call duration to Sonera mobile networks
"@+id/textView_recommendation"	Displays application's recommendation
"@+id/button1_showanalysis"	Starts the analysis activity

The second GUI *.xml* file is the "*activity_analysis.xml"* (see page 32 line 16). This GUI displays to the user detailed information about the call logs to the different Finnish mobile network providers in a tabular form. The user can check each network provider's details by touching each tab on the device's screen. The call logs provide detailed information for each of the Finnish mobile network providers, such as their total duration, average monthly duration, total number of days calls were made during the specified period and a list containing the phone numbers dialled, the date the call was placed and the duration of the call.

Table . Description of Android ids

Android id	Description
"@+id/tabhost"	The container for the tab widget

"@android:id/tabs"	Displays the list of tabs
"@+id/tab1"	Displays DNA tab
"@+id/ttextViewDnaDuration"	Displays DNA duration
"@+id/ttextViewAvgDnaCallDuration"	Displays average DNA call duration
"@+id/ttextViewDnaNumberofDays	Displays the total number of days for DNA calls
"@+id/tlistViewDnaLogs"	Displays a list of DNA call logs
"@+id/tab2"	Displays Elisa tab
"@+id/ttextViewElisaDuration"	Displays Elisa duration
"@+id/ttextViewAvgElisaCallDuration"	Displays average Elisa call duration
"@+id/ttextViewElisaNumberofDays"	Displays the total number of days for Elisa calls
"@+id/tlistViewElisaCalllogs"	Displays a list of Elisa call logs
"@+id/tab3"	Displays Sonera tab
"@+id/ttextViewSoneraDuration"	Displays Sonera call duration
"@+id/ttextViewAvgSoneraCallDuration"	Displays average sonera call duration
"@+id/ttextViewSoneraNumberofDays"	Displays the total number of days for Sonera calls
"@+id/tlistViewSoneraCalllogs"	Displays a list of Sonera call logs

2.2 CODING AND DEVELOPMENT

Before writing the code for this application a flow process was implemented to accomplish the application's goal and make the coding process smoother and easier.

The code part of this application contains two *.java* files which represent two activities. The first activity, "*CallAnalyzer.java" (see page 38 line 22*) references the "*main_call_analyzer.xml"* and all its components while the second activity

"*Analysis.java" (see page 49 line 16*) references the "*activity_analysis.xml"* and all its components.

2.2.1 FLOWCHART

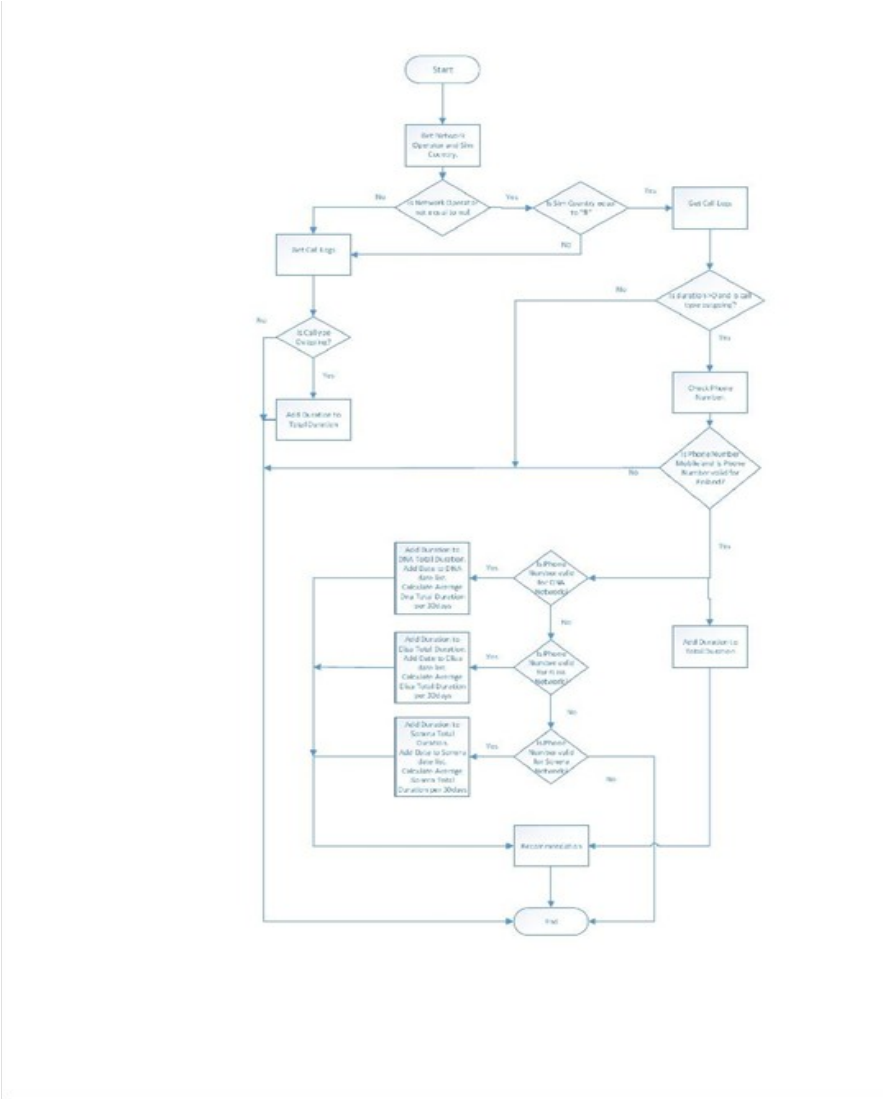


Figure . Flowchart used to develop the application

2.2.2 FLOWCHART DESCRIPTION

Figure 1 describes the process used in implementing the application.

When the application starts, it undergoes a process that checks the network operator and the SIM country of the SIM card in the device. After checking this,

it uses the information from the initial process to determine if the SIM country is Finland or not.

If the SIM country is Finland, it calls a process that gathers all the call logs available in the device for sorting. This sorting process includes checking if a particular call in the call logs is an outgoing call with duration greater than zero.

Call logs that pass this check are then processed further to verify if they are a valid Finnish mobile phone number and if they are, the call durations for each of these calls are summed up and the average is calculated. A final decision process then uses these values to give a recommendation to the user.

2.2.3 JAVA CODE

I. CallAnalyzer.java Activity

Since this application is only meant to analyse call logs for Finnish mobile networks, the method "*SimCountryandSimState ()"* (see page 40 line 24) would first check if the SIM Card in the Android device is from Finland. To do this, the method first confirms if there is a SIM card in the Android device and if there is, it then checks if the SIM card is of Finnish locale. This confirmation is needed to help the application in sorting out the call logs for only calls made to Finnish mobile numbers.

After the *"SimCountryandSimState ()"* method has confirmed that there is a SIM card in the Android device and it is of a Finnish mobile network, the method *"getCallLogs()"* (see page 42 line 45) is called. This method is the "brain box" behind this application. This method accesses the data logs contents of the Android device.

In an Android device, each call log data are stored in the device in a database form with columns. For the purpose of this application, only the phone number, call type, duration and date columns are referenced because they are all that is needed to achieve its objective.

The first action this method does after it has referenced the necessary columns is to check for call logs labelled as “outgoing” and have a duration that is greater than zero. If both these conditions are met, the next step for the method is to confirm if the phone number dialled is a valid Finnish phone number and if it is a “MOBILE” number. This is achieved by the method *“numberSorter()”* (see page 46 line 14). This method uses an external library the “libphonenumber” to help make the validation.

If both these conditions are true, the sum of the duration is calculated and the date each call was placed is stored in an arraylist.

After this step, the method (getcalllogs()) then classifies each of the mobile numbers that met all the above criteria into the three major mobile networks providers in Finland (DNA, Elisa, and Sonera). To do this, each phone number is first formatted into the Finnish National phone number digit system and then checked if they begin with the network digit identifier for each of the three network providers listed above.

As the “getcalllogs()” method does this, it sums the duration for each network provider and also saves the date of each network provider to its own arraylist and counts the total number of days in their arraylist.

After this step is completed, the “getCallLogs()” method then calculates the monthly average for both the total call duration and the duration for each of the Finnish mobile networks.
On the completion of all these steps, the application then uses both the average total call duration and the average call duration to each of the

mobile networks to recommend which network is best suited for the user together with the average monthly amount of minutes.

II. Analysis.java Activity

The function of this activity is to display detailed information of the call logs analysis in the *"CallAnalyzer activity"* (see page 38 line 27). After the *"CallAnalyzer activity"* has completed all its steps, it passes some data that are needed by this activity. These includes the total duration for each of the mobile networks together with their averages, total number of days and a list of the call logs containing the phone number dialled, the duration spent and the date the call was placed.

These data are then divided to its respective tab created in this activity and displayed to the user when the user clicks the *"Show Analysis"* (see page 40 line 51) button in the *"CallAnalyzer activity"*.

2.2.4 ANDROID MANIFEST

Every Android application has an "*AndroidManifest.xml"* file (see page 37 line 1). This file is required for every Android application to function. Every Android application has a unique package name and the Java package name for an application is named here.

Amongst other things, this file describes and determines the components of an application and which process the components in the application needs. Most importantly, any permission needed by an application is declared here along with a list of the different activities in the application. For this application, the Android Manifest file contains two activities and it uses two permissions.

- <uses-permission android:name="android.permission.READ_PHONE_STATE" />
- <uses-permission android:name="android.permission.READ_LOGS" />

The first permission helps the application to check the if the Android device has a SIM card while the second permission helps the application to read the call logs in the Android device.

3. PROBLEMS ENCOUNTERED AND LIMITATIONS

3.1 PROBLEMS ENCOUNTERED

During the development process of this application (FinMobile Call Analyzer), there were some problems and setbacks.

- The first problem was forcing the application to verify that the SIM card in the device is of Finnish SIM card. Initially this problem was approached by using the Android telephony service to check the SIM state (if the SIM card is active or not) in the device. This approach, however, does not confirm if the SIM card is a Finnish one thus defeating the main purpose of the application. It is necessary to verify the SIM card country because it helps the application in sorting the phone numbers for only Finnish mobile numbers.

  To solve this problem, the telephony services "*getNetworkCountryIso()"* and the *"getSimOperatorName()"* (see page 40 line 25 & 26 respectively) *were used* to replace the SIM state. The "*getSimOperatoNamer()"* first verifies if there is a SIM card in the device and if there is, the *"getNetworkCountryIso()"* then checks if the SIM card is a Finnish SIM card.

- Verifying the phone numbers if they were a valid mobile phone number for Finland was the toughest problem faced during the development of this application. This is an integral part of the application as it is what guides the application to making its recommendation(s) to the user.

  The initial approach here was to manually sort the numbers into two types; those that begins with a "plus" sign and those that do not. After sorting them, a decision process was initiated that checks first if the

number begins with the Finnish dialling code and then if the total length of the number conforms with the Finnish phone number system (Wikipedia 2013). The result of this approach was inaccurate as most of the phone numbers that were Finnish mobile numbers in the call logs were not processed.

The "*libphonenumber"* library was the solution to this problem as it made sure that all phone numbers in the call logs were processed and checked accurately to only collect valid Finnish mobile phone numbers.

- Identifying the different Finnish network operators was another obstacle. The "carrier" library (another library associated with "libphonenumber") was used to try and solve this problem but it was only able to identify two Finnish mobile network operators, Elisa and DNA.

  This problem was solved by implementing a decision process using the Finnish mobile network codes found in the Finnish Communications and Regulatory Authority website (Finnish Communications Regulatory Authority 2013) to sort the phone numbers into the three major network operators in Finland.

- The last problem was that of when the application resumes after being demoted by another application when it was running. The method "onResume()" (see page 48 line 46) was used to solve this problem; however, each time the application resumes, the call durations were doubled each time. So another method called the "ResumeDataCheck()" (see page 47 line 43) was created to initialize all the durations back to zero when the "onResume()" method is called during the application's process.

3.2 LIMITATIONS

Before the implementation of mobile number portability in Finland, the first three to five digits in a mobile number served as the network operator identifier. This application uses this digit system to identify which network a particular dialled number in the call logs belongs to thus limiting its accuracy.

The solution to this problem which we plan to implement on later versions of this application is to create a method that will use “HTTP” and “PHP” to pass each of the dialled numbers to website that maintains the Finnish mobile Number system (Finnish information service of ported numbers 2013). This website will identify correctly which particular network the dialled number belongs thus increasing the accuracy of the application.

Being unable to contact the host of this website together with the inability to find another way to accurately identify which mobile number is ported; this solution was not implemented for the application.

4. FINMOBILE CALL ANALYZER VS. OTHER CALL LOGS APPS

FinMobile Call Analyzer is unique in its function as in the Google play store, there were many applications whose function was based on call logs but there was none that calculates and recommends the amount of duration that is best suitable for a user of a particular Android device.

Some of the applications are:

a. Call Filter- All this application does is filter incoming calls. It blocks unwanted calls and prevents both incoming and outgoing calls to certain numbers specified by the user.

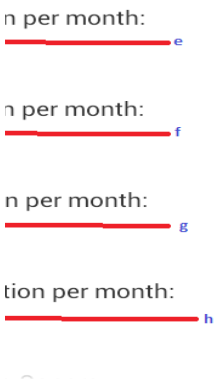
b. Call Meter- This was the closet application on the Google play store with almost similar functions to *"FinMobile Call Analyzer"*. The idea of this application is based on monitoring a user's mobile plan. Users inputs and sets the limit for calls, SMS and data according to their mobile call plan and the application monitors how the plan is used.

These were the only two applications published on the Google play store that bore similarities with *"FinMobile Call Analyzer"* as all the other application that uses call logs focused on displaying a device's call logs in different ways.

5. TESTING AND CONCLUSION

As the first version with a minimum SDK of 3.0, FinMobile Call Analyzer was tested on an Android device running the “Icecream Sandwich” Android operating system version (version 4.0.2). The figures below show the results obtained.

5.1 EXPLANATIONS FOR THE ITEMS IN FIGURES 2, 3 & 4



- Item a: This is the general view of the call analysis activity. This view contains several elements that are used to display to the user the result of the application’s analysis of the call logs in the Android device.

  The elements in this view are eight different text views and a button. Each of these elements has its own function in the view with respect to the analysed result of the call logs.

- Item b: This is a text view that displays the function of the application to the user. The content of this text view is a constant string which displays all the time the user starts the application.

- Item c: This is a text view that displays the period from which the call logs were analysed that gave the displayed result. The value of this text view changes taking into account the data available on the Android device.

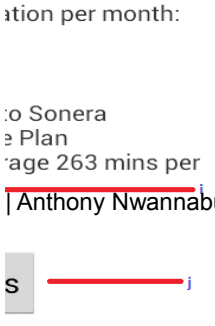
- Item d: This is a text view that displays the total call duration of call logs to Finnish mobile phone numbers for the specified period in item c.

- Item e: This is a text view that displays the monthly average total call duration of call logs to Finnish mobile numbers for the specified period in item c.

- Item f: This is a text view that displays the monthly average total call duration of call logs to DNA network for the specified period in item c.

- Item g: This is a text view that displays the monthly average total call duration of call logs to Elisa network for the specified period in item c.

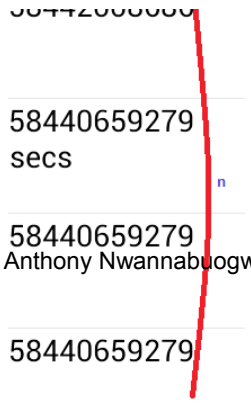
- Item h: This is a text view that displays the monthly average total call duration of call logs to the Sonera network for the specified period in item c.



- Item i: This is a text view of high importance in the application. It displays the recommendation of the application based on the result of the analysis of the call logs. It is in this text view the user receives the application's suggestion on which mobile network and how much call minutes per month is best suitable for the user of the Android device.

- Item j: This is a button that executes the Analysis activity when clicked. This activity displays a detailed result of the analysed call logs in tabs.

  The Analysis activity divides the analysed call logs into three different tabs that each contains the three mobile networks being considered and their analysed results.



- Tab a: This tab holds the view for the DNA network analysed call logs. This tab contains three text views and a list view.

- Item k: This is a text view contained in “Tab a”. It displays the total call duration to DNA networks for the specified period in item c.

- Item l: This text view in “Tab a” displays the monthly average call duration to DNA network for the specified period in item c.

- Item m: This text view in “Tab a” displays the total number of days within the specified period in item c that phone calls were made to the DNA network.

- Item n: This is a list view in “Tab a” that displays each call logs details (the mobile number, the duration of the call and the date the call was made) to DNA network.

- Tab b and c: These tabs have the same contents as “Tab a”, the only difference being that “Tab b” displays analysed call logs details to Elisa network while “Tab c” displays analysed call logs details to Sonera network.

From the result of this test, the application (FinMobile Call Analyzer) recommended that the user should consider a mobile plan for Sonera network because its average monthly call duration was greater than that of DNA and Elisa.

Since this is the first version of the application, this recommendation is on average accurate because of the method used to analyse the dialled mobile numbers (the network digit system). However, in later versions this recommendation will give a high accuracy as the call logs will be analysed using a more advanced method which will involve sending each dialled mobile number through the internet to the Finnish mobile ported numbers webpage where the network for each dialled number will be correctly identified.

REFERENCES

World Wide Web page

Android Developer [Online], http://developer.android.com/ [November 2013]

Android OS, Wikipedia [Online], Available: http://en.wikipedia.org/wiki/Android_(operating_system) [November 2013]

Android Versions, Wikipedia [Online], Available: http://en.wikipedia.org/wiki/Android_version_histor [November 2013]

Finnish Communication Regulatory Authority [Online], Available: https://www.viestintavirasto.fi/internetpuhelin/puhelinnumerot/matkaviestinverkot/matkaviestinverkkojensuuntanumerot.html [November 2013]

Finnish Information Service of Ported Numbers [Online], http://www.numpac.fi [November 2013]

Finnish Mobile Network Operators, Wikipedia [Online], Available: http://en.wikipedia.org/wiki/Telephone_numbers_in_Finland [November 2013]

LibPhoneNumber [Online], https://code.google.com/p/libphonenumber/ [November 2013]

Mobile OS, Wikipedia [Online], Available: http://en.wikipedia.org/wiki/Comparison_of_mobile_operating_systems [November 2013]

Open Handset Alliance [Online], http://www.openhandsetalliance.com/ [November 2013]

APPENDIX

Main_call_activity.xml

```
//Container for elements in the main_call_activity

<ScrollViewxmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout_height="fill_parent"

android:layout_width="fill_parent" tools:context=".CallAnalyzerActivity">

<RelativeLayout android:layout_width="match_parent" android:layout_height="match_parent"

android:paddingBottom="@dimen/activity_vertical_margin"  android:paddingLeft="@dimen/activity_horizontal_margin"

android:paddingRight="@dimen/activity_horizontal_margin" android:paddingTop="@dimen/activity_vertical_margin"

>

//Textview to display the function of the application

<TextView  android:layout_width="wrap_content"  android:layout_height="wrap_content"

android:layout_centerHorizontal="true" android:paddingTop="10dp" android:id="@+id/textview_description"

android:text="Analysis for outgoing calls to\nFinnish Mobile Networks "/>

//Textview to display the period of the analysis

<TextView

android:id="@+id/textView_timeperiod"  android:layout_width="wrap_content"  android:layout_height="wrap_content"

android:layout_alignParentLeft="true" android:layout_below="@+id/textview_description"

android:layout_marginLeft="20dp" android:layout_marginTop="30dp" android:text=""/>

//Textview that displays the total call duration

<TextView android:id="@+id/textView_totalduration" android:layout_width="wrap_content"

android:layout_height="wrap_content" android:layout_alignLeft="@+id/textView_timeperiod"

android:layout_below="@+id/textView_timeperiod" android:layout_marginTop="30dp"

android:text=""/>
```

```
//Textview to display the call duration for DNA network

<TextView

android:id="@+id/textView_durationDna"  android:layout_width="wrap_content"  android:layout_height="wrap_content"

android:layout_alignLeft="@+id/textView_totalduration"  android:layout_below="@+id/textView_totalduration"

android:layout_marginTop="30dp"

android:text=""/>


//Textview to display the call duration for Elisa network

<TextView

android:id="@+id/textView_durationElisa"  android:layout_width="wrap_content"  android:layout_height="wrap_content"

android:layout_alignLeft="@+id/textView_durationDna"  android:layout_below="@+id/textView_durationDna"

android:layout_marginTop="30dp"

android:text=""/>


//Textview to display the call duration to Sonera network

<TextView

android:id="@+id/textView_durationSonera" android:layout_width="wrap_content

android:layout_height="wrap_content" android:layout_alignLeft="@+id/textView_durationElisa"

android:layout_below="@+id/textView_durationElisa" android:layout_marginTop="30dp"

android:text=""/>

//Textview to display the application's recommendation

 <TextView

android:id="@+id/textView_recommendation" android:layout_width="wrap_content"

android:layout_height="wrap_content" android:layout_alignLeft="@+id/textView_durationSonera"

android:layout_below="@+id/textView_durationSonera" android:layout_marginTop="30dp"

android:text=""/>


//Button to execute the Analysis activity

<Button

android:id="@+id/button1_showanalysis" android:layout_width="wrap_content" android:layout_height="wrap_content"
```

```
android:layout_alignLeft="@+id/textview_description" android:layout_below="@+id/textView_recommendation"
android:layout_marginTop="30dp" android:visibility="invisible"
android:text="Show Analysis"/>
</RelativeLayout> </ScrollView>
```

activity_analysis.xml

```
//Container for items in the Analysis activity
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android" android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical">
//Tab holder for the different mobile network
<TabHost
android:id="@+id/tabhost" android:layout_width="match_parent" android:layout_height="match_parent"> <LinearLayout
android:layout_width="match_parent" android:layout_height="match_parent" android:orientation="vertical"> <TabWidget
android:id="@android:id/tabs" android:layout_width="match_parent" android:layout_height="wrap_content">
</TabWidget>
<FrameLayout android:id="@android:id/tabcontent" android:layout_width="match_parent"
android:layout_height="match_parent"> <RelativeLayout android:orientation="vertical" android:id="@+id/tab1"
android:layout_width="match_parent" android:layout_height="match_parent">
//Textview to display the total call duration to DNA network in the DNA Tab
<TextView android:id="@+id/ttextViewDnaDuration" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignParentLeft="true" android:layout_alignParentTop="true"
android:layout_marginLeft="20dp" android:layout_marginTop="20dp" android:text="TextView"/>
//Textview to display the monthly average call duration to DNA network in the DNA Tab
<TextView android:id="@+id/ttextViewAvgDnaCallDuration" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/ttextViewDnaDuration"
android:layout_below="@+id/ttextViewDnaDuration" android:layout_marginTop="20dp" android:text="TextView"/>
```

```
//Textview to display the total number of days calls were made to DNA network in the DNA Tab

<TextView android:id="@+id/ttextViewDnaNumberofDays" android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/ttextViewAvgDnaCallDuration"
android:layout_below="@+id/ttextViewAvgDnaCallDuration" android:layout_marginTop="20dp"
android:text="TextView"/>

//Listview to display the call log information for calls made to DNA network in the DNA Tab

<ListView android:id="@+id/tlistViewDnaLogs"
android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_alignLeft="@+id/ttextViewDnaNumberofDays"
android:layout_below="@+id/ttextViewDnaNumberofDays" android:layout_marginTop="30dp">
</ListView> </RelativeLayout>

//Layout for Elisa Tab

<RelativeLayout
android:orientation="vertical" android:id="@+id/tab2" android:layout_width="match_parent"
android:layout_height="match_parent">
//Textview to display the total call duration to Elisa network in the Elisa Tab

<TextView android:id="@+id/ttextViewElisaDuration" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignParentLeft="true" android:layout_alignParentTop="true"
android:layout_marginLeft="20dp" android:layout_marginTop="20dp" android:text="TextView"/>

//Textview to display the monthly average call duration to Elisa network in the Elisa Tab

<TextView android:id="@+id/ttextViewAvgElisaCallDuration" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/ttextViewElisaDuration"
android:layout_below="@+id/ttextViewElisaDuration" android:layout_marginTop="20dp" android:text="TextView"/>
```

```
//Textview to display the number of days calls were made to Elisa network in the Elisa Tab

<TextView android:id="@+id/ttextViewElisaNumberofDays" android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/ttextViewAvgElisaCallDuration"
android:layout_below="@+id/ttextViewAvgElisaCallDuration" android:layout_marginTop="20dp"
android:text="TextView"/>

//Listview to display the call log information for calls made to Elisa network in the Elisa Tab

 <ListView
android:id="@+id/tlistViewElisaCalllogs" android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_alignLeft="@+id/ttextViewElisaNumberofDays"
android:layout_below="@+id/ttextViewElisaNumberofDays" android:layout_marginTop="30dp">
</ListView>
</RelativeLayout>


//Layout for Sonera Tab

 <RelativeLayout
android:orientation="vertical" android:id="@+id/tab3" android:layout_width="match_parent"
android:layout_height="match_parent">


//Textview to display the total call duration to Sonera network in the Sonera Tab

<TextView android:id="@+id/ttextViewSoneraDuration" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignParentLeft="true" android:layout_alignParentTop="true"
android:layout_marginLeft="20dp" android:layout_marginTop="20dp" android:text="TextView"/>

//Textview to display the monthly average call duration to Sonera network in the Sonera Tab

<TextView android:id="@+id/ttextViewAvgSoneraCallDuration" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/ttextViewSoneraDuration"
android:layout_below="@+id/ttextViewSoneraDuration" android:layout_marginTop="20dp" android:text="TextView"/>
```

```
//Textview to display the number of days calls were made to Sonera network in the Sonera Tab

<TextView android:id="@+id/ttextViewSoneraNumberofDays" android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/ttextViewAvgSoneraCallDuration"
android:layout_below="@+id/ttextViewAvgSoneraCallDuration" android:layout_marginTop="20dp"
android:text="TextView"/>


//Listview to display the call log information for calls made to Sonera network in the Sonera Tab

<ListView
android:id="@+id/tlistViewSoneraCalllogs" android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_alignLeft="@+id/ttextViewSoneraNumberofDays"
android:layout_below="@+id/ttextViewSoneraNumberofDays" android:layout_marginTop="30dp">
</ListView></RelativeLayout>
</FrameLayout> </LinearLayout>
 </TabHost> </LinearLayout>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.callanalyzer"
    android:versionCode="1" android:versionName="1.0" >
        //Minimum standard development kit that the application will function on
    <uses-sdk
        android:minSdkVersion="11" android:targetSdkVersion="11" />

        //Permissions used by the application
    <uses-permission          android:name="android.permission.READ_PHONE_STATE" />
     <uses-permission android:name="android.permission.READ_LOGS" />
     //Attributes of the application
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >

     //Attributes of the activities in the application
        <activity
            android:name="com.callanalyzer.CallAnalyzer"
            android:label="@string/app_name"
            android:screenOrientation="portrait" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category              android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
```

```
        </activity>
        <activity
            android:name="com.callanalyzer.Analysis"
            android:label="@string/title_activity_analysis"
            android:screenOrientation="portrait">
        </activity>
    </application>
</manifest>
```

CallAnalyzer.java

```
//                  The callAnalyzer main class

public class CallAnalyzer extends Activity{
// Declared variables
        TextView TimePeriod,TotalCallDuration,SoneraCallDuration,DnaCallDuration,ElisaCallDuration,Recommenda
        tion,Description;
        Button Analyze;
        String DisplayTimePeriod,DisplayRecommendation;
        int DisplayTotalCallDuration=0,DisplaySoneraCallDuration=0,DisplayDnaCallDuration=0,
        DisplayElisaCallDuration=0;
        int TotalCallDurationInMins,TotalDurationSecs;
        int DnaCallDurationInMins,DnaDurationSecs,ElisaCallDurationInMins,ElisaDurationSecs,
        SoneraCallDurationInMins,SoneraDurationSecs;
        int AvgTotalCallDuration,AvgDnaCallDuration,AvgElisaCallDuration,AvgSoneraCallDuration;
        int AvgDnaSecs,AvgElisaSecs,AvgSoneraSecs;
        int returnMins,returnSecs,returnAvgMins,returnAvgSecs;

        int ElisaTotalCallDays,DnaTotalCallDays,SoneraTotalCallDays;
        String ElisaCallList,DnaCallList,SoneraCallList;
        String Startdate,EndDate;
        int arraydatesize,SoneraDateSize,DnaDateSize,ElisaDateSize;
```

```
TelephonyManager telly;

String SimCountry,NetworkName;

int SimState;


ArrayList<String>ArrayDateList,SoneraCallLogs,SoneraDateList,DnaCallLogs,DnaDateList,ElisaCa
llLogs,ElisaDateList;


PhoneNumberUtil PhoneUtil;

PhoneNumber DialledNumber;

PhoneNumber NetworkNumber;


PhoneNumberType type;

boolean isValidforSimCountry=true;

String nationalNumber;

int howmanydays;


@Override

protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.main_call_analyzer);


        Initializers();

        SImCountryandSimState();

}
//Class that instantiates all the variables declared at the heading of the main class

private void Initializers(){


TimePeriod=(TextView) findViewById(R.id.textView_timeperiod);

 TotalCallDuration=(TextView) findViewById(R.id.textView_totalduration);

SoneraCallDuration=(TextView) findViewById(R.id.textView_durationSonera);

DnaCallDuration=(TextView) findViewById(R.id.textView_durationDna);

ElisaCallDuration=(TextView) findViewById(R.id.textView_durationElisa);

Recommendation=(TextView) findViewById(R.id.textView_recommendation); Description=(TextView)

findViewById(R.id.textview_description); Analyze=(Button) findViewById(R.id.button1_showanalysis);
```

```
        ArrayDateList=new ArrayList<String>();
        SoneraCallLogs=new ArrayList<String>();
        SoneraDateList=new ArrayList<String>();
        DnaCallLogs=new ArrayList<String>();
        DnaDateList=new ArrayList<String>();
        ElisaCallLogs=new ArrayList<String>();
        ElisaDateList=new ArrayList<String>();

        telly=(TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);
        PhoneUtil=PhoneNumberUtil.getInstance();

}
// Class that checks the country of the inserted Sim card

private void SImCountryandSimState(){
        SimCountry=telly.getNetworkCountryIso();
        NetworkName=telly.getSimOperatorName();
        SimState=telly.getSimState();
        if((NetworkName!=null) && (SimCountry.equalsIgnoreCase("fi"))){
            Analyze.setVisibility(Button.VISIBLE); getCallLogs();
            if((AvgDnaCallDuration>AvgElisaCallDuration)&&(AvgDnaCallDuration>AvgSonera CallDuration)){
            Recommendation.setText("You average more calls to DNA Network. DNA Mobile Plan
            Recommended.");
            }elseif((AvgElisaCallDuration>AvgDnaCallDuration)&&
            (AvgElisaCallDuration>AvgSoneraCallDuration)){
            Recommendation.setText("You average more calls to Elisa Networks. Elisa
            Mobile Plan Recommended.");

        }elseif((AvgSoneraCallDuration>AvgElisaCallDuration)&&
        (AvgSoneraCallDuration>AvgDnaCallDuration)){
        Recommendation.setText("You average more calls to Sonera Networks. Sonera
        Mobile Plan Recommended.");
                }else Recommendation.setText("Recommendation Undecisive.");
                Analyze.setOnClickListener( new OnClickListener(){
                    @Override
                public void onClick(View v) {
```

```
                    // TODO Auto-generated method stub

//                                  Passing the datas to the TabsActivity for display

                                Intent TabIntent=new Intent(CallAnalyzer.this,Analysis.class);

                                TabIntent.putExtra("DNATotalDurationInMins",
                                DnaCallDurationInMins);

                                TabIntent.putExtra("DnaDurationInSecs", DnaDurationSecs);

                                TabIntent.putExtra("AvgDnaCallDurationInMins",
                                AvgDnaCallDuration);

                                TabIntent.putExtra("AvgDnaCallDurationInSecs", AvgDnaSecs);

                                TabIntent.putExtra("DnaCallDays", DnaDateSize);

                                TabIntent.putStringArrayListExtra("DnaCallLogsList", DnaCallLogs);


//                              Do same for Elisa and sonera


                                TabIntent.putExtra("ElisaTotalDurationInMins",
                                ElisaCallDurationInMins);

                                TabIntent.putExtra("ElisaDurationInSecs", ElisaDurationSecs);

                                TabIntent.putExtra("AvgElisaCallDurationInMins",
                                AvgElisaCallDuration);

                                TabIntent.putExtra("AvgElisaCallDurationInSecs", AvgElisaSecs);

                                TabIntent.putExtra("ElisaCallDays", ElisaDateSize);

                                TabIntent.putStringArrayListExtra("ElisaCallLogsList",
                                ElisaCallLogs);

                                TabIntent.putExtra("SoneraTotalDurationInMins",
                                SoneraCallDurationInMins);

                                TabIntent.putExtra("SoneraDurationInSecs", SoneraDurationSecs);

                                TabIntent.putExtra("AvgSoneraCallDurationInMins",
                                AvgSoneraCallDuration);

                                TabIntent.putExtra("AvgSoneraCallDurationInSecs", AvgSoneraSecs);

                                TabIntent.putExtra("SoneraCallDays", SoneraDateSize);
                                TabIntent.putStringArrayListExtra("SoneraCallLogsList",
                                SoneraCallLogs);

                                startActivity(TabIntent);

                    }

                });

                }else{


                    AlertDialog.Builder SimCountryAlert=new AlertDialog.Builder(this);

                    SimCountryAlert.setTitle("Attention!!!");
```

```
                    SimCountryAlert.setMessage( "Your Sim Country is not supported if you wish to continue,
                    only the total outgoing call durations will be displayed.");


                    SimCountryAlert.setPositiveButton("Ok",new
                    DialogInterface.OnClickListener() {


                            @Override

                            public void onClick(DialogInterface dialog, int which) {


                                    // TODO Auto-generated method stub

                                    getCallLogs();

                                    TimePeriod.setText(null);

                                    SoneraCallDuration.setText(null);

                                    ElisaCallDuration.setText(null);

                                    DnaCallDuration.setText(null);

                                    Recommendation.setText(null);

                                    Description.setText("Total Outgoing Call
                                    Durations");


                            }

                    }); SimCountryAlert.create().show();

            }


    }

private void getCallLogs(){

Cursor Logs_Query=getContentResolver().query
        (android.provider.CallLog.Calls.CONTENT_URI,null,null,null,null);

         if(Logs_Query!=null){


        int numberColumn=Logs_Query.getColumnIndex(CallLog.Calls.NUMBER);

        int typeColumn=Logs_Query.getColumnIndex(CallLog.Calls.TYPE);

        int durationColumn=Logs_Query.getColumnIndex(CallLog.Calls.DURATION);

        int dateColumn=Logs_Query.getColumnIndex(CallLog.Calls.DATE);


        while (Logs_Query.moveToNext()){
```

```
String logPhoneNumber=Logs_Query.getString(numberColumn);

String logCallType=Logs_Query.getString(typeColumn);

String logCallDuration=Logs_Query.getString(durationColumn);

String logCallDate=Logs_Query.getString(dateColumn);


int INTlogCallType=Integer.parseInt(logCallType);

int INTlogCallDuration=Integer.parseInt(logCallDuration);

Date CallDate=new Date(Long.valueOf(logCallDate));

String formattedDate=DateFormat.getDateInstance().format(CallDate);


if (INTlogCallType==CallLog.Calls.OUTGOING_TYPE && INTlogCallDuration>0){

//                  Filter number through libphonenumber library.

                    if(SimCountry.equalsIgnoreCase("FI")){

                    NumberSorter(logPhoneNumber,SimCountry);

//                          sort numbers that are only mobile using type

                if(isValidforSimCountry&&type.toString()=="MOBILE"){

                        ArrayDateList.add(formattedDate);

                        nationalNumber=PhoneUtil.format(DialledNumber,
                        PhoneNumberFormat.NATIONAL).replace(" ", "");
                        DisplayTotalCallDuration=INTlogCallDuration+DisplayTotalCallDuration;

//                  Sort Elisa Numbers

                if(nationalNumber.startsWith("0451")||nationalNumber.startsWith("0452")||
                nationalNumber.startsWith("0453")||nationalNumber.startsWith("0456")||
                nationalNumber.startsWith("0458")||nationalNumber.startsWith("046")||
                nationalNumber.startsWith("050")){

                        DisplayElisaCallDuration=INTlogCallDuration+DisplayElisaCa llDuration;
                        ElisaDateList.add(formattedDate);

                        if(INTlogCallDuration>=60){

                        ConvertSecsToMins(INTlogCallDuration);
                ElisaCallList=("Mobile Number: "+logPhoneNumber+"\nDuration: "+returnMins+"mins
                "+returnSecs+"secs"+"\nDate: "+formattedDate);


                        }else{

                        ElisaCallList=("Mobile Number: "+logPhoneNumber+"\nDuration:
                        "+INTlogCallDuration+"secs"+"\nDate: "+formattedDate);

                        }

                        ElisaCallLogs.add(ElisaCallList);
//                                              Sort Dna Numbers

                }elseif(nationalNumber.startsWith("041")||nationalNumber.startsWith("044")||
                nationalNumber.startsWith("04574")||nati                onalNumber.startsWith("04576")||
                nationalNumber.startsWith("04577")||nationalNumber.start                sWith("04578")||
```

```
nationalNumber.startsWith("04579")||nationalNumber.startsWith("04944")){

                    DisplayDnaCallDuration=INTlogCallDuration+DisplayDnaCallDu ration;

                    DnaDateList.add(formattedDate);

if(INTlogCallDuration>=60){

                    ConvertSecsToMins(INTlogCallDuration);

                    DnaCallList=("Mobile Number: "+logPhoneNumber+"\nDuration:
                    "+returnMins+"mins "+returnSecs+"secs"+"\nDate: "+formattedDate);

}else{
                    DnaCallList=("Mobile Number: "+logPhoneNumber+"\nDuration:
                    "+INTlogCallDuration+"secs"+"\nDate: "+formattedDate);


}



                    DnaCallLogs.add(DnaCallList);


//                                                  Sort Sonera Numbers

}elseif  (nationalNumber.startsWith("040")||nationalNumber
.startsWith("042")||nationalNumber.startsWith("0450"))

{
                    DisplaySoneraCallDuration=INTlogCallDuration+DisplaySoneraCallDuration;

                    SoneraDateList.add(formattedDate);

if(INTlogCallDuration>=60){

                    ConvertSecsToMins(INTlogCallDuration);

                    SoneraCallList=("Mobile Number: "+logPhoneNumber+"\nDuration:
                    "+returnMins+"mins "+returnSecs+"secs"+"\nDate: "+formattedDate);

}else{

                    SoneraCallList=("Mobile Number: "+logPhoneNumber+"\nDuration:
                    "+INTlogCallDuration+"secs"+"\nDate: "+formattedDate);


}

//

                    SoneraCallLogs.add(SoneraCallList);

}else ;


}


}else if(SimCountry!="FI"){

                        DisplayTotalCallDuration=INTlogCallDuration+DisplayTotalCallDurati on;

                }
```

```
}
}
Logs_Query.close();
        TotalCallDurationInMins=ConvertSecsToMins(DisplayTotalCallDuration);

        TotalCallDuration.setText("Total Duration:"+TotalCallDurationInMins+"mins "+returnSecs+"secs");

        if(SimCountry.equalsIgnoreCase("FI")){
        arraydatesize=DateCalculator(ArrayDateList);
        AvgTotalCallDuration=AvgCallDuration(DisplayTotalCallDuration,arraydatesize);
        Startdate=ArrayDateList.get(0);
        EndDate=ArrayDateList.get(arraydatesize-1);
        TimePeriod.setText("Period from "+EndDate+" - "+Startdate);
        ElisaCallDurationInMins=ConvertSecsToMins(DisplayElisaCallDuration);
        ElisaDurationSecs=returnSecs;
        ElisaDateSize=DateCalculator(ElisaDateList);
        AvgElisaCallDuration=AvgCallDuration(DisplayElisaCallDuration,ElisaDateSize);
        AvgElisaSecs=returnAvgSecs;
        ElisaCallDuration.setText("Elisa call duration: "+ElisaCallDurationInMins+"mins
        "+ElisaDurationSecs+"secs");

        DnaCallDurationInMins=ConvertSecsToMins(DisplayDnaCallDuration);
        DnaDurationSecs=returnSecs;
        DnaDateSize=DateCalculator(DnaDateList);
        AvgDnaCallDuration=AvgCallDuration(DisplayDnaCallDuration,DnaDateSize);
        AvgDnaSecs=returnAvgSecs;
        DnaCallDuration.setText("DNA call duration: "+DnaCallDurationInMins+"mins
        "+DnaDurationSecs+"secs");

        SoneraCallDurationInMins=ConvertSecsToMins(DisplaySoneraCallDuration);
        SoneraDurationSecs=returnSecs;
        SoneraDateSize=DateCalculator(SoneraDateList);
        AvgSoneraCallDuration=AvgCallDuration(DisplaySoneraCallDuration,SoneraDateSize);
        AvgSoneraSecs=returnAvgSecs;
```

```
                SoneraCallDuration.setText("Sonera call duration: "+SoneraCallDurationInMins+"mins
                "+SoneraDurationSecs+"secs");
        }else;


        }



}
private boolean NumberSorter(String number, String simcountry){
        try {
                DialledNumber=PhoneUtil.parse(number, simcountry);
                type=PhoneUtil.getNumberType(DialledNumber);
        } catch (NumberParseException e) {
                / TODO Auto-generated catch block
                e.printStackTrace();
        }
        isValidforSimCountry=PhoneUtil.isValidNumberForRegion(DialledNumber, simcountry);
        return isValidforSimCountry;
}
private int DateCalculator(ArrayList<String> calldate){

                ArrayList<String> Newcalldate=new ArrayList<String>();
                Iterator<String>IteratedList=calldate.iterator();

        while(IteratedList.hasNext()){

                String verify=IteratedList.next();
                if(Newcalldate.contains(verify)){ IteratedList.remove();
                }else{
                  Newcalldate.add(verify);
                }
        }
                howmanydays=Newcalldate.size();
                return howmanydays;
}
```

```
private int ConvertSecsToMins(int TimeinSecs){

        int ExtraMins = 0;

        int AvgSecs=TimeinSecs%60;
    while(AvgSecs>=60){
        ExtraMins+=ExtraMins;
        ExtraMins=AvgSecs/60;
        AvgSecs=AvgSecs%60;

    }
        returnSecs=AvgSecs; returnMins=(TimeinSecs/60)+ExtraMins;

    return returnMins;

}

private int AvgCallDuration(int TimeinSecs,int TotalDays){
        int ExtraMins = 0;

        int AvgSecs=((30/TotalDays)*(TimeinSecs%60));
        while(AvgSecs>=60){ ExtraMins+=ExtraMins; ExtraMins=AvgSecs/60; AvgSecs=AvgSecs%60;

    }
        returnAvgSecs=AvgSecs; returnAvgMins=((30*TimeinSecs)/(60*TotalDays))+ExtraMins;
    return returnAvgMins;
}
private void ResumeDataCheck(){
        DisplayTotalCallDuration=0;DisplaySoneraCallDuration=0;DisplayDnaCallDuration=0;Dis
        playElisaCallDuration=0;
        TotalCallDurationInMins=0;TotalDurationSecs=0;
        DnaCallDurationInMins=0;DnaDurationSecs=0;ElisaCallDurationInMins=0;ElisaDurationSe
        cs=0;SoneraCallDurationInMins=0;SoneraDurationSecs=0;
        AvgTotalCallDuration=0;AvgDnaCallDuration=0;AvgElisaCallDuration=0;AvgSoneraCallDur
        ation=0;
```

```
AvgDnaSecs=0;AvgElisaSecs=0;AvgSoneraSecs=0;

returnMins=0;returnSecs=0;returnAvgMins=0;returnAvgSecs=0;


ElisaTotalCallDays=0;DnaTotalCallDays=0;SoneraTotalCallDays=0;


ElisaCallList=null;DnaCallList=null;SoneraCallList=null; Startdate=null;EndDate=null;

arraydatesize=0;SoneraDateSize=0;DnaDateSize=0;ElisaDateSize=0;

ArrayDateList=null;

SoneraCallLogs=null;

SoneraDateList=null;

DnaCallLogs=null;

DnaDateList=null;

ElisaCallLogs=null;

ElisaDateList=null;

SimCountry=null;NetworkName=null;

TimePeriod.setText("");

TotalCallDuration.setText("");

TAvgTotalCallDuration.setText("");

SoneraCallDuration.setText("");

DnaCallDuration.setText("");

ElisaCallDuration.setText("");

Recommendation.setText("");

Analyze.setVisibility(Button.INVISIBLE);

}

@Override

protected void onResume() {

// TODO Auto-generated method stub

super.onResume();
```

```
                ResumeDataCheck();


                Initializers();

                SImCountryandSimState();

        }
}
```

Analysis.java

```
// Main class for the Analysis activity

public class Analysis extends Activity {

//Declared variables

TextView DnaTotalDuration,DnaTotalDays,AvgDnaCallDuration,ElisaTotalDuration,

ElisaTotalDays,AvgElisaCallDuration,SoneraTotalDuration,SoneraTotalDays,AvgSoner aCallDuration;

ListView DnaCallLogsList,ElisaCallLogsList,SoneraCallLogsList;

ArrayAdapter<String>DnaCallLogsAdapter,ElisaCallLogsAdapter,

SoneraCallLogsAdapter;

ArrayList<String>DnaList,ElisaList,SoneraList;


@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_analysis);

        initializer();

        //              Obtain the values sent from the Main Activity

        Bundle getTransferedData=getIntent().getExtras();

        if (getTransferedData!=null){

        /               Values for Dna

        DnaTotalDuration.setText(" Total Duration: "+getTransferedData.getInt("DNATotalDurationInMins")+"mins
        "+getTransferedData.getInt("DnaDurationInSecs")+"secs");

        AvgDnaCallDuration.setText("Avg. call duration /30days:
        "+getTransferedData.getInt("AvgDnaCallDurationInMins")+"mins
        "+getTransferedData.getInt("AvgDnaCallDurationInSecs")+"secs");

        DnaTotalDays.setText("Total number of Days: "+getTransferedData.getInt("DnaCallDays"));
```

```
DnaList=getTransferedData.getStringArrayList("DnaCallLogsList");

DnaCallLogsAdapter=new
ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,DnaList);

DnaCallLogsList.setAdapter(DnaCallLogsAdapter);

//                      Values for Elisa
ElisaTotalDuration.setText("TotalDuration: "+getTransferedData.getInt("ElisaTotalDurationInMins")+"mins
"+getTransferedData.getInt("ElisaDurationInSecs")+"secs");

AvgElisaCallDuration.setText("Avg. call duration /30days:
"+getTransferedData.getInt("AvgElisaCallDurationInMins")+"mins
"+getTransferedData.getInt("AvgElisaCallDurationInSecs")+"secs");

ElisaTotalDays.setText("Total number of Days: "+getTransferedData.getInt("ElisaCallDays"));

ElisaList=getTransferedData.getStringArrayList("ElisaCallLogsList");

ElisaCallLogsAdapter=new ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,ElisaList);

ElisaCallLogsList.setAdapter(ElisaCallLogsAdapter);


/       Values for Sonera

        SoneraTotalDuration.setText("TotalDuration:"+getTransferedData.getInt("SoneraTotalDurationInMins")
        +"mins "+getTransferedData.getInt("SoneraDurationInSecs")+"secs");

        AvgSoneraCallDuration.setText("Avg. call duration /30days:
        "+getTransferedData.getInt("AvgSoneraCallDurationInMins")+"mins
        "+getTransferedData.getInt("AvgSoneraCallDurationInSecs")+"secs");

        SoneraTotalDays.setText("Total number of Days: "+getTransferedData.getInt("SoneraCallDays"));

        SoneraList=getTransferedData.getStringArrayList("SoneraCallLogsList");

        SoneraCallLogsAdapter=new
        ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,SoneraList);

        SoneraCallLogsList.setAdapter(SoneraCallLogsAdapter);

}


/       Create Tabs for the Analysis TabHost th=(TabHost)findViewById(R.id.tabhost);


/       Tab for Dna

th.setup();

TabSpec specs=th.newTabSpec("tag1");

specs.setContent(R.id.tab1);

specs.setIndicator("DNA");

th.addTab(specs);


//                      Tab for Elisa
specs=th.newTabSpec("tag2"); specs.setContent(R.id.tab2); specs.setIndicator("Elisa"); th.addTab(specs);
```

```
                //                      Tab for Sonera
specs=th.newTabSpec("tag3"); specs.setContent(R.id.tab3); specs.setIndicator("Sonera"); th.addTab(specs);


th.setOnTabChangedListener(new TabHost.OnTabChangeListener() {


@Override
public void onTabChanged(String tabId) { // TODO Auto-generated method stub


}
});
}


private void initializer(){
                        DnaTotalDuration=(TextView) findViewById(R.id.ttextViewDnaDuration);
                        DnaTotalDays=(TextView) findViewById(R.id.ttextViewDnaNumberofDays);
                        DnaCallLogsList=(ListView) findViewById(R.id.tlistViewDnaLogs);
                         AvgDnaCallDuration=(TextView) findViewById(R.id.ttextViewAvgDnaCallDuration);
                        DnaList=new ArrayList<String>();
                        ElisaTotalDuration=(TextView) findViewById(R.id.ttextViewElisaDuration);
                        ElisaTotalDays=(TextView)findViewById(R.id.ttextViewElisaNumberofDays);
                        ElisaCallLogsList=(ListView) findViewById(R.id.tlistViewElisaCalllogs);
                        AvgElisaCallDuration=(TextView) findViewById(R.id.ttextViewAvgElisaCallDuration);
                        ElisaList=new ArrayList<String>();
                        SoneraTotalDuration=(TextView)                          findViewById(R.id.ttextViewSoneraDuration);
                        SoneraTotalDays=(TextView) findViewById(R.id.ttextViewSoneraNumberofDays);
                        SoneraCallLogsList=(ListView) findViewById(R.id.tlistViewSoneraCalllogs);
                        AvgSoneraCallDuration=(TextView) findViewById(R.id.ttextViewAvgSoneraCallDuration);
            SoneraList=new ArrayList<String>();
}
}
```