# Sim Tool Kit (STK) based Drive Testing tool

*By*

NC Samrah Mirza

PC Huma Razzaq

NC Haris Mehmood NC

NC Ahmad Faraz

NC Dawood Ahmad

Project Supervisor
Lt. Col. Dr. Adnan Rashdi

Submitted to the Faculty of Electrical Engineering, Military College of Signals, National University of Sciences and Technology, Rawalpindi in partial attainment for the requirements of a B.E. Degree in Telecom Engineering

JUNE 2017

## CERTIFICATE OF CORRECTNESS AND APPROVAL

Certified that work contained in this thesis titled “SIM Toolkit based Drive Tesing Tool”, carried out by Samrah Mirza, Huma Razzaq, Haris Mehmood, Ahmad Faraz and Dawood Ahmad under the supervision of Dr. Adnan Rashdi for partial fulfillment of Degree of Bachelors of Telecommunication Engineering, is correct and approved.

Approved By

____________________

Lt.Col. Dr. Adnan Rashdi

EE Department

Military College of Signals, NUST

Dated: June 2017

## ABSTRACT

## Sim Toolkit (STK) based Drive Testing Tool

The SIM Toolkit (STK) is a set of commands or applications that define how a SIM card interacts with the outside world. The sim toolkit, is an application that is programmed into the SIM card and enables:

- Drive mobile equipment interface
- Build up interaction between the network application and end user
- Control the access to the network

Drive testing is common when a new network is created and the performance indicators are measured through moving  specific in the particular area. It is also done where the strength is less and for measuring the parameters the issues are solved. This is done by the moving in that particular area in a vehicle with the required equipment in it. The purpose of our project is despite of using man labor and such an expensive method of measuring the parameters these measurements can be taken via sim tool kit, in which a program as an applet is introduced in the sim. And the option for measuring the parameters is then displayed on the sim tool kit application. And by clicking on this application the message would be send to the operator with all the basic parameters performance for the particular area and improvements are made, if necessary.

## DECLARATION

No portion of work presented in this thesis “SimTool kit based drive testing tool” for the degree of B.E Electrical( Telecommunications) is a part of another award or qualification either from this institution or any other..

***In The Name Of Allah, the Most Benevolent, the Most Merciful.***

## DEDICATION

*Almighty Allah for HIS countless blessings*

*Instructors and friends for their help*

*And our beloved parents for their prayers and support*

# ACKNOWLEDGEMENT

All praises for Allah Almighty Who guided and enabled us to undertake this project. We would like to thank our project supervisor Lt. Col. Dr. Adnan Rashdi for his assistance, supervision and generous support throughout our Final Year Project.

We would like to thank our teachers and friends for their help in the completion of this project.

# Contents

Chapter 1: Introduction ........................................................................ 7

1.1. Project Overview ........................................................................ 7

1.2. Problem Statement ........................................................................ 8

1.3. Approach ........................................................................ 9

1.4. Objectives ........................................................................ 9

1.5. Limitations ........................................................................ 10

Chapter 2: Literature Review ........................................................................ 12

2.1. Project Domain ........................................................................ 12

2.2. Literature Review ........................................................................ 12

Chapter3: Technological Requirements ........................................................................ 15

3.1. Smart card Hardware platform ........................................................................ 15

3.2. Smart card software platform ........................................................................ 16

3.2.2. Open and Global platform ........................................................................ 17

3.2.3. The Java Card virtual machine (JCVM) ........................................................................ 17

3.3. Operating System Requirements ........................................................................ 18

3.3.1. Java Card virtual machine (JCVM) ........................................................................ 19

3.3.2 Java card API ........................................................................ 19

3.3.3. Java Card Runtime Environment (JCRE) ........................................................................ 19

3.3.4. Application layer (Applets) ........................................................................ 19

Chapter 4: Applet development Stages ........................................................................ 21

4.1 Application Deployment ........................................................................ 21

4.1.1 Service definition ........................................ 21

4.1.2 Service development ........................................ 22

4.1.3 Application validation ........................................ 22

4.1.4 Application deployment ........................................ 23

4.1.5 Maintenance ........................................ 23

4.2 Applet Development Stages ........................................ 24

4.2.1 Off-card process ........................................ 24

4.2.2 Java Card source code ........................................ 24

4.2.3 On-card process ........................................ 25

4.3 Sim card file system and security ........................................ 26

4.3.1 File Tree structure: ........................................ 27

4.3.2 File security: ........................................ 27

Chapter 5: Creating an Applet ........................................ 29

5.1 Main Code ........................................ 29

Chapter 6: Conclusion and Future work ........................................ 33

Bibliography ........................................ 34

Appendix: ........................................ 35

# List of Tables

Table 3-1 Name and description of sim card pins ........................................ 16

Table 4-1 Decription of file types.................................................................. 27

# List of Figures

Figure 1-1 Drive Testing  vehicle……………………………….…….…8

Figure 3-1 Sim card used in mobile phones  ………………….……….… 15

Figure 3-2 Sim card physical connections.…………………………..... 15

Figure 3-3 Layers of sim in Java card Platform……………..…..… .18

Figure 4-1 Applet Loading steps……………………………….……....25

Figure 4-2 Load File data block……………………………......... 26

Figure 4-3 Tree structure of file system within a sim…………...…..27

Figure 5-1 Graphical User Interface of development suite......…….29

Figure 5-2 Applet menus……………………........……………... ..30

Figure 5-3 RxLevel formula and measurement ……..……..……….. .30

Figure 5-4 RxQuality and Bit Error Rate………………………… .31

Figure 5-5 Signal to noise ratio formula and calculation………..... 31

Figure 5-6 IMEI number and local Area number……………..…. 32

# List of Abbreviations

| | |
|---|---|
| GSM | Global System for Mobile Communication |
| BTS | Base Transceiver Station |
| KPI | Key Performance Indicators |
| QoS | Quality of Service |
| SIM | Subscriber Identity Module |
| API | Application Programming Interface |
| APDU | Application Protocol Data Unit |
| ISO | International Standard Organization |
| OTA | Over The Air |
| BER | Bit Error Rate |
| SNR | Signal to Noise Ratio |
| IMEI | International Mobile Equipment Identity |
| LAN | Local Area Number |

# *Chapter 1:* Introduction

***1.1 Project Overview***

***1.2 Problem Statement***

***1.3 Approach***

***1.4 Objectives***

***1.5 Limitations***

# Chapter 1: Introduction

## 1.1. Project Overview

When a new network is established after the network planning, testing needs to be done. That is done through Drive testing, in which capacity, coverage and the Service quality is measured and assessed. The testing is done to check the coverage criteria of a particular area using testing tools. The data is collected in form of Log files through which Key Performance Indicators (KPIs) are measured around a particular area. The drive testing consists of a motor vehicle with mobile network air interface measurement equipment that can detect and save numerous parameters. By measuring what a wireless network subscriber would experience in any specific area, mobile operators can make directed changes to their networks to provide better coverage and service to their customers. The parameters that are collected during the drive testing field measurements includes various paramaters as follows:

- Signal intensity
- Signal quality
- Interference
- Dropped calls
- Blocked call
- Call statistics
- Service level statistics □ QoS information
- Handover information

- GPS location co-ordinates

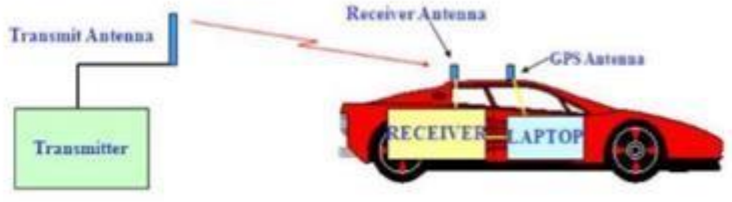


Fig 1.1: Drive testing machine

## 1.2. Problem Statement

Drive testing requires special tools/ equipment and a specially trained drive test engineer to drive and travel through the whole particular area to take the readings. Overall process is expensive including the equipment required.

## 1.3. Approach

- The Baseband processor inside a mobile phone gets the signals from all the nearby BTS stations of a cellular network and selects the best one for the transmission, on the basis of readings it gets from every BTS station.
- So, it is unintentionally getting most of the readings that are necessary for network evaluation.
- An applet will be installed on a programmable SIM using a SIM writer, that will get the readings from baseband processor as parameters and calculate the KPI's.
- The application will then write those calculations into a text message and automatically send it to the central server of particular Cellular network.

## 1.4. Objectives

Our aim is to deliver a system that will provide the cellular network`s KPI evaluation automatically by the users` mobile phones. So there will be no or lesser need of the traditional drive test, in order to reduce the overall cost needed for the process and save resources, while enhancing our knowledge of:

- Communication networks
- Smart cards and Sim architecture
- Sim Tool Kit API
- Concept of drive testing.

- Coding (Java and SIM card) An app will be installed on the SIM which will calculate the KPI`s, write them in a text message and send it to the main server of the operator.

## 1.5. Limitations

This project has its limitations. As the software does not support few of the commands due to which not all of the values can be extracted from the baseband processor.

# *Chapter 2:* Literature Review

***2.1 Project Domain***

***2.2 Literature Review***

# Chapter 2: Literature Review

## 2.1. Project Domain

The thesis is meant to give an introduction to a new method of extracting the key performance indicators via sim, which surely needs to be covered in more depth. Java Card/SIM Application development, The APDU command set, smart card file system are the important subjects deemed with more depth.

## 2.2. Literature Review

A conventional drive-test system consists of:

- A handset with a charger, headset and data cable and USB Hub.
- A Laptop with an investigation and an adapter.
- GPS with an EXT Antenna.
- Inverter and Terminal
- Scanner for GSM (Ext Antenna GPS and RF, Data Cable)

    - Battery and Charger etc

The goal of the measurements described above in the document is to assess the network under test for its quality parameters that is, the network quality for the respective transactions from the subscribers’ point of view

In carrying out a drive test, measurements are conducted in a way that user behavior is simulated, with a number of channel parameters/ characteristics under the control of the measurement equipment. Measurements are carried out simultaneously to ensure comparability of test results. The testing procedure is fully automated to eliminate the subjectivity that is inherent to human intervention.

The monitoring equipment is installed with a software tool which supports the storage and organization of information as well as the generation of statistics on the data obtained by the measuring unit. The software tool makes it possible to generate different reports on single or multiple monitoring sessions.  The equipment also incorporates a GPS receiver is a device that receives information from GPS satellites and calculate accurate geographical location and according to it, it does calculations with respect to the area.

# *Chapter 3*: Technological Requirements

***3.1 Hardware Platform***

***3.2 Software Platform***

***3.3 Operating Systems Requirements***

# Chapter3: Technological Requirements

In this section , the hardware requirement will be discussed along with the software specifications for the final deliverable.

## 3.1. Smart card Hardware platform

A Sim card has three different types of physical sizes, the sim card of interest for this project is the 25x15mm  version found in mobile phones.

Fig 3-1: SIM card used in mobile phones, version 25x15 mm.

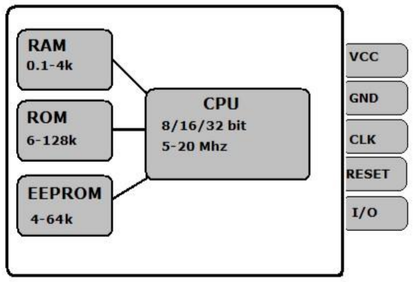
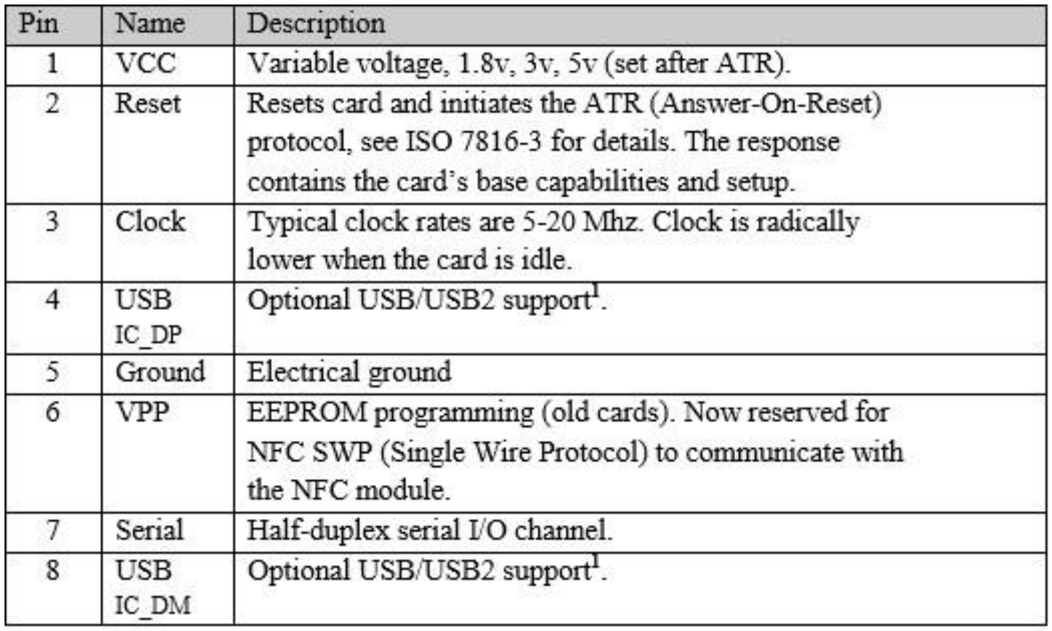


Fig3-2: Sim card physical connections

A sim card has eight connecting pins according to ISO standards 7816-3/4 and ISO 7816121.



| Pin | Name | Description |
|---|---|---|
| 1 | VCC | Variable voltage, 1.8v, 3v, 5v (set after ATR). |
| 2 | Reset | Resets card and initiates the ATR (Answer-On-Reset) protocol, see ISO 7816-3 for details. The response contains the card's base capabilities and setup. |
| 3 | Clock | Typical clock rates are 5-20 Mhz. Clock is radically lower when the card is idle. |
| 4 | USB IC_DP | Optional USB/USB2 support[1]. |
| 5 | Ground | Electrical ground |
| 6 | VPP | EEPROM programming (old cards). Now reserved for NFC SWP (Single Wire Protocol) to communicate with the NFC module. |
| 7 | Serial | Half-duplex serial I/O channel. |
| 8 | USB IC_DM | Optional USB/USB2 support[1]. |

Table 3-1: Name and Description of sim card pins

## 3.2. Smart card software platform

In the start the smart cards were having limited memory with only basic functions but with the passage of time the size has been reduced and the functionality has also been improved from basic to very much complex. Due to this reason it is now possible to run multiple applications directly onto the sim card.

The smart card is a closed environment such that nothing can be uploaded or gain access to any file from the sim until it is unlocked, which means that it is encrypted with complex techniques so to gain control of the sim it needs to be unlocked. All the applications and data can be controlled by the manufacturer.

Before 1990s, the sim cards were programmed in C/Assembly language and were highly dependent on the hardware. A solution was needed to improve the situation and then Java Card Specification was developed within this environment all the applications were controlled.

### 3.2.2. Open and Global platform

For implementing the new Java Card Specification security and control over the applications was an important factor and this was done by the Open Platform.

For all the technologies, the basic aim is to make the technologies is to make them generic even if the focus is to achieve secure environment. Therefore, all the Java Card enabled SIMs follow a Global Platform standard. It was started by the SIM Card R99 and onwards.

### 3.2.3. The Java Card virtual machine (JCVM)

Java Card virtual machine (JCVM) is present in the Java Card Specification and it secludes the hardware part from the software and its applications. The JCVM executes the applications in a virtual environment. The applet uses a library called the Java Card API to perform its functions. The JCVM combined with the Java Card API and the operating system together is known as the Java Card Runtime Environment (JCRE).

In the JCVM there is implementation of the functions which are provided by the Global Platform that is generic for all sim cards.

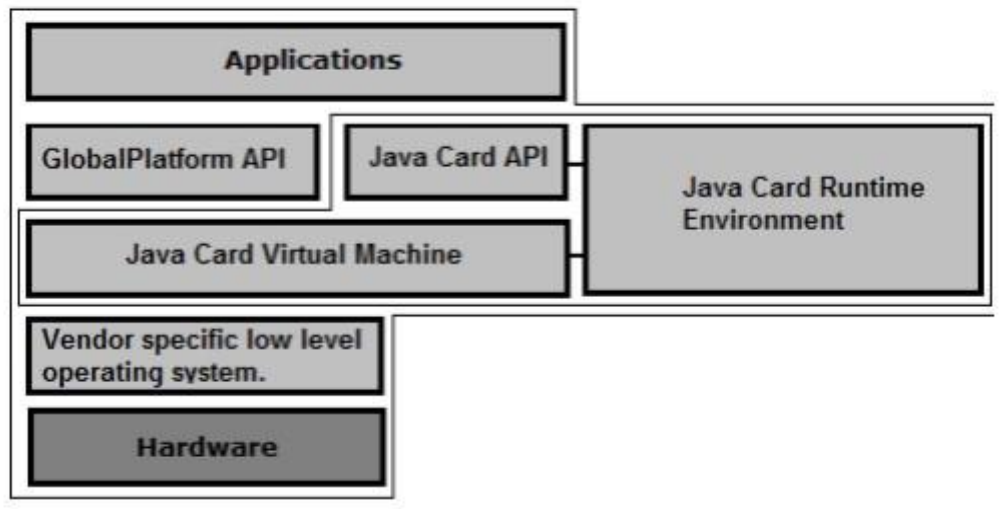

Applications
GlobalPlatform API
Java Card API
Java Card Runtime Environment
Java Card Virtual Machine
Vendor specific low level operating system.
Hardware

Fig 3-3: Layers of SIM in Java Card Platform

## 3.3. Operating System Requirements

The functions in this layer are usually developed by the Assembly or C language by the manufacturer for good performance, compact size and considering other requirements. It includes serial communication, application management and different encryption libraries.

### 3.3.1. Java Card virtual machine (JCVM)

The main goal of the Java Card technology was to make the applets which are portable as well as secure. This specific role is done by the Java Card Virtual Machine (JCVM) which is an integral part of Java Card Runtime Environment.

### 3.3.2 Java card API

The applets are made using some class libraries the collection of these class libraries lies in Java Card API. The class libraries are defined by the Java Card specifications.

### 3.3.3. Java Card Runtime Environment (JCRE)

It consists of the **operating system** and the functions and features defined by the Global Platform to make sim cards generic in nature. It consists of both, the Java Card Virtual Machine and Java Card API. The Applets are also considered as part of the JCRE.

### 3.3.4. Application layer (Applets)

This layer consists of the applications which are being run on the sim cards, the Applets. Applets are inside JCRE . The applet is executed within a Java Virtual Machine (JVM) using the class of libraries from Java Card API and moving onto the JCRE respectively.

# *Chapter 4*: Applet development Stages

***4.1 Application Deployment***

***4.2 Applet Development Stages***

***4.3 Sim card file system and security***

# Chapter 4: Applet development Stages

In this section we will learn how the application (Applet) is made and through what procedures it has to go through to become a working application to be a part of Sim Toolkit.

## 4.1 Application Deployment

The services which are required in our project are implemented on the card in the form of applets. Applets, as discussed above are the applications made by the developers/manufacturers using appropriate language and commands.

### 4.1.1 Service definition

The value added services depends on what type of communications needs to be implemented or depends on the operators. As we are using the Gemalto software for making the Applet. Therefore, the services which are being offered by Gemalto are considered while making this project. Those services depends upon:

- GSM features and functions:
- Sim Toolkit functions and services
- Functions specific to the operator so that only operator himself can control some of the features

- In case of providing services after the SIMs are provided or any update is required in the software OTA( Over the Air) services are of great importance.

For this project the Toolkit functions are used considering the requirement of the project, as only a toolkit application is required to be made.

### 4.1.2 Service development

A service provider is allowed realizing the service development the services are given through the service providers to the end users. Development of the services is done through following steps:

- Prototype the service, to test the service by observing the user's interaction with service.
- Program and later compile the application using development suite.
- Load the compiled application onto the card. Define any application specific files and configure the data according to the requirement.
- At the end load the application its files and its configuration data onto the card.

### 4.1.3 Application validation

Certifying the applications is an important factor which needs to be considered while deploying the applet on the sim card. Before loading the application on sim it is important to consider that the applet is bug free and all the security requirements are being fulfilled. Only the operator and Gemalto has these rights of getting certification for the Applications.

### 4.1.4 Application deployment

Applications can be deployed into the sim card. They can be loaded either :

- By Gemalto or
- In the mobile using OTA customization

### 4.1.5 Maintenance

Maintenance of the Application consists of three stages:

- Modify the application data, what the text is required for that particular application.
- Modification of application files (PIN code value)
- Adding a new menu in the applet.
  In this case the previous applet code has to be deleted and the updated code is then loaded.

## 4.2 Applet Development Stages

The application development stages are divided in to two phases depending on the physical environment. They are off-card and on-card phases.

### 4.2.1 Off-card process

The off-card process includes:

- Develop the source code java file
- Compiling the code.
- After compiling the code it needs to be converted in a format understandable y the sim. Therefore, it is converted in **.jar** or **.ijc** files respectively.

### 4.2.2 Java Card source code

When the applet is loaded on the sim card, it provides set of services to the user. They get activated in result of the requests being received from the end user. The client application at the terminal interacts with the applet as following:

- The client application firstly implements the functions
- The applet uses APDU commands to send parameters and the interaction is done between phone and sim via these commands.
- The applet sends a response that is received by the client application.

- The client application analyses the card’s response.

## 4.2.3 On-card process

The Card Manager is responsible for all the installation of the applications. It consists of a card registry and it also uses it, which consists of the information about the application life cycle states.

The on-card process includes:

- The package is Loaded.
- From the package the applet is installed.
- Making this instance selectable

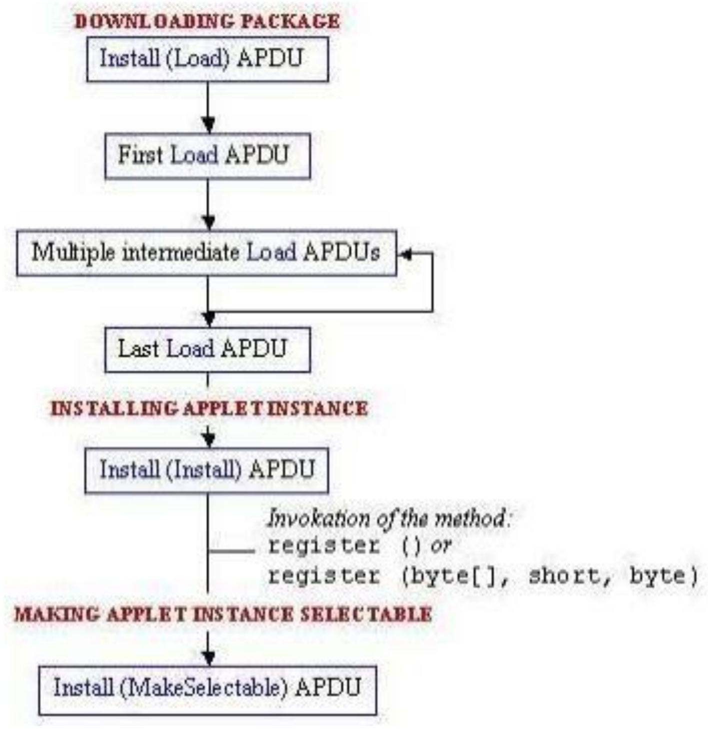

DOWNLOADING PACKAGE
Install (Load) APDU
First Load APDU
Multiple intermediate Load APDUs
Last Load APDU
INSTALLING APPLET INSTANCE
Install (Install) APDU
Invokation of the method:
register () or
register (byte[], short, byte)
MAKING APPLET INSTANCE SELECTABLE
Install (MakeSelectable) APDU

Fig 4-1: Applet loading steps

In the Load file, it consist of loading the package. The Load file has two blocks:

- A Data Authentication Pattern block, its purpose is to perform the verification procedure in Card Manager.
- A Load File Data Block, it consists of the content of .ijc file.

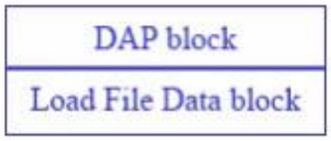


Fig 4-2: Load File data block

The Load File is loaded using two APDU commands:

- APDU command is installed on Load mode.
- After installing APDU command is loaded.

## 4.3 Sim card file system and security

### 4.3.1 File Tree structure:

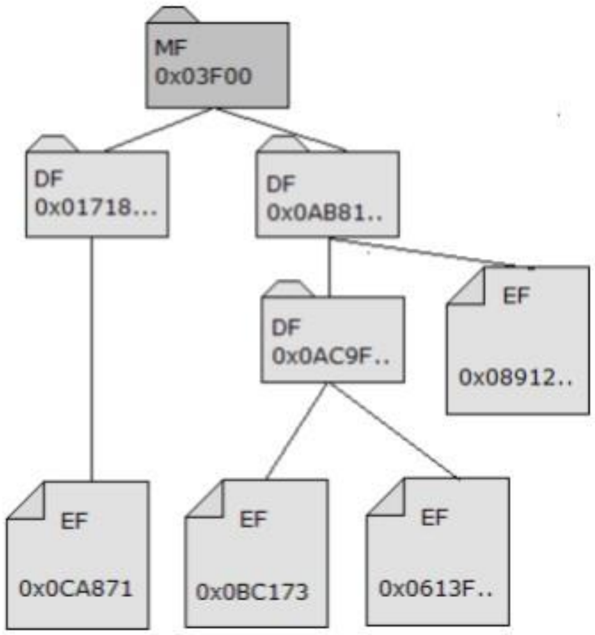
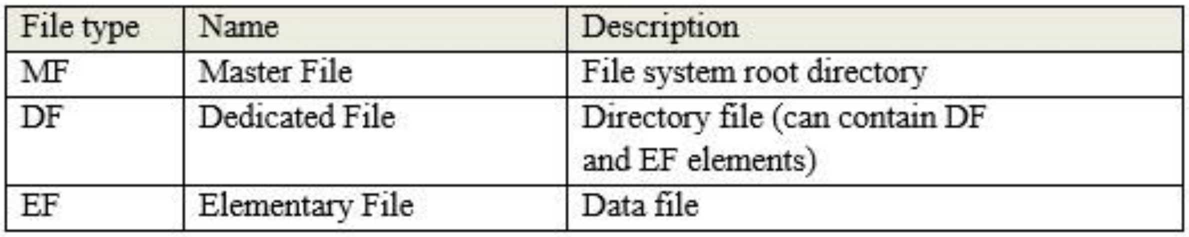


Figure 4-3: tree structure of file system within a sim



| File type | Name | Description |
|---|---|---|
| MF | Master File | File system root directory |
| DF | Dedicated File | Directory file (can contain DF and EF elements) |
| EF | Elementary File | Data file |

Table 4-2: Description of the file types

### 4.3.2 File security:

There are three main levels of security authentication on the sim card:

- After verification security status password is obtained.
- A key is used for the security
- Data authentication is done using checksums or digital signatures.
- Data enciphering is done using key management and data concealment with XOR.

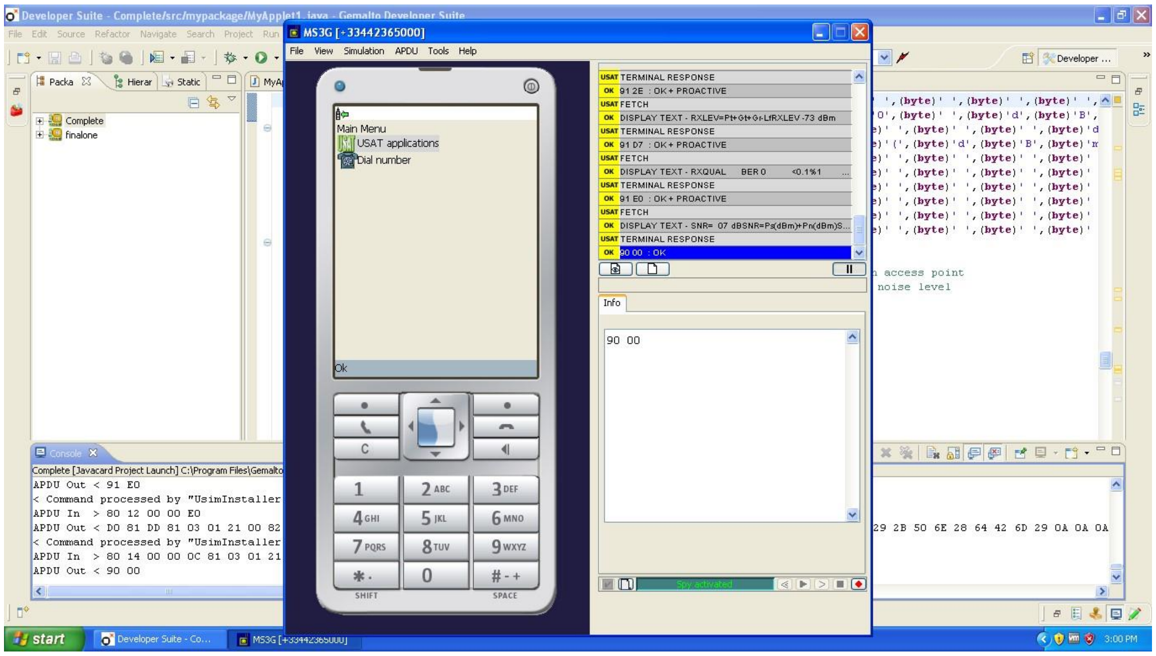
# *Chapter* 5: Creating an Applet

## ***5.1 Main Code***

# Chapter 5: Creating an Applet

## 5.1 Main Code

This code depicts the main network measurements results in KPIs which we have incorporated in our applet.


Developer Suite - Complete/src/mypackage/MyApplet1.java - Gemalto Developer Suite
MS3G [+33442365000]
File View Simulation APDU Tools Help
Main Menu
USAT applications
Dial number
Ok
Info
start
Developer Suite - Co...
3:00 PM

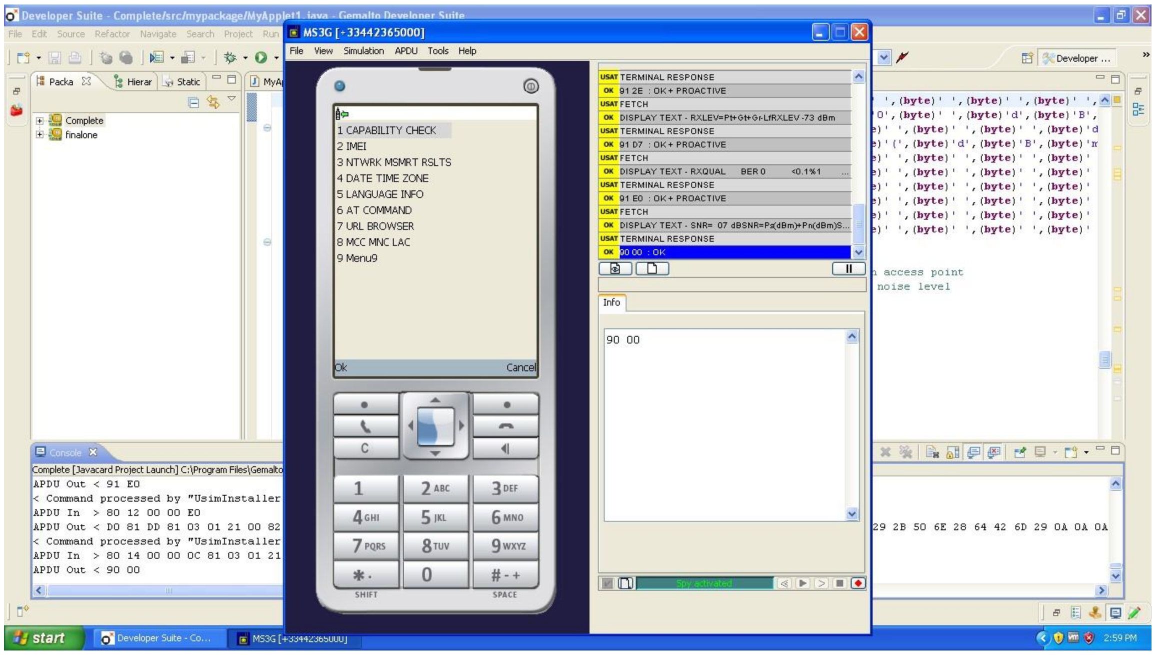
```
USAT TERMINAL RESPONSE
OK   91 2E  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXLEV=Pt+Gt+Gr-LfRXLEV -73 dBm
USAT TERMINAL RESPONSE
OK   91 D7  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXQUAL    BER 0        <0.1%1      ...
USAT TERMINAL RESPONSE
OK   91 E0  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - SNR=  07 dBSNR=Ps(dBm)+Pn(dBm)S...
USAT TERMINAL RESPONSE
OK   90 00  : OK
```

```
90 00
```

```
Complete [Javacard Project Launch] C:\Program Files\Gemalto
APDU Out < 91 E0
< Command processed by "UsimInstaller
APDU In  > 80 12 00 00 E0
APDU Out < D0 81 DD 81 03 01 21 00 82
< Command processed by "UsimInstaller
APDU In  > 80 14 00 00 0C 81 03 01 21
APDU Out < 90 00
```

```
29 2B 50 6E 28 64 42 6D 29 0A 0A 0A
```

Figure 5-1: Graphical user interface of development suite



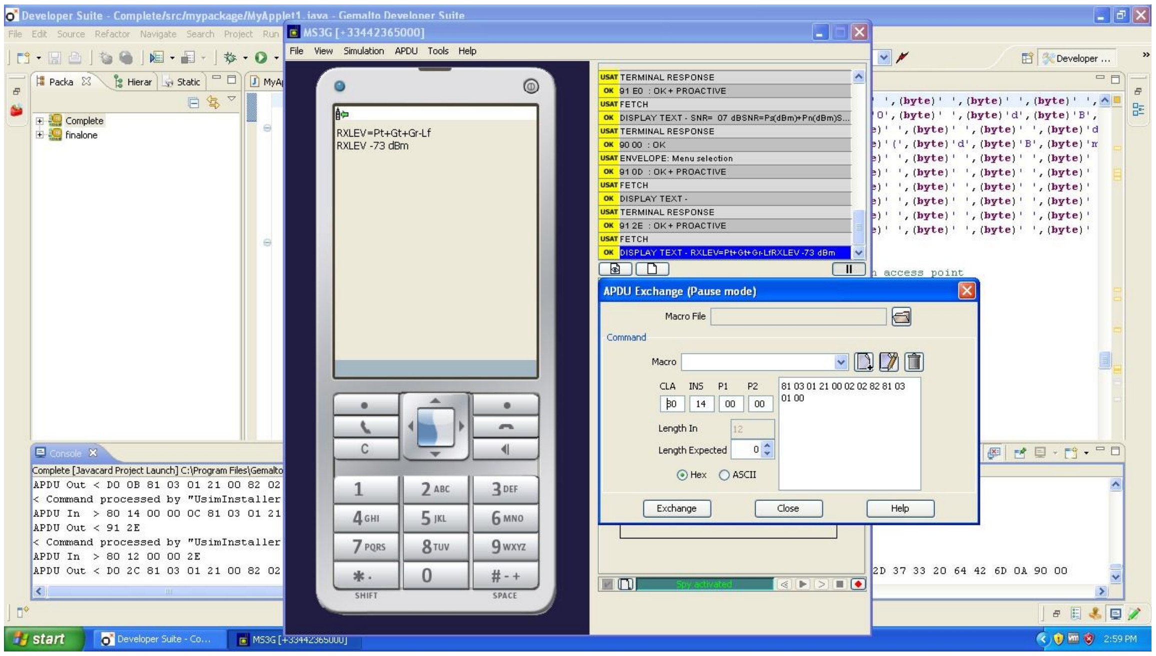
```
1 CAPABILITY CHECK
2 IMEI
3 NTWRK MSMRT RSLTS
4 DATE TIME ZONE
5 LANGUAGE INFO
6 AT COMMAND
7 URL BROWSER
8 MCC MNC LAC
9 Menu9
```

```
USAT TERMINAL RESPONSE
OK   91 2E  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXLEV=Pt+Gt+Gr-LfRXLEV -73 dBm
USAT TERMINAL RESPONSE
OK   91 D7  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXQUAL     BER 0        <0.1%1      ...
USAT TERMINAL RESPONSE
OK   91 E0  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - SNR=  07 dBSNR=Ps(dBm)+Pn(dBm)S...
USAT TERMINAL RESPONSE
OK   90 00  : OK
```

```
90  00
```

```
Complete [Javacard Project Launch] C:\Program Files\Gemalto
APDU Out < 91 E0
< Command processed by "UsimInstaller
APDU In  > 80 12 00 00 E0
APDU Out < D0 81 DD 81 03 01 21 00 82
< Command processed by "UsimInstaller
APDU In  > 80 14 00 00 0C 81 03 01 21
APDU Out < 90 00
```

Figure 5-2: Applet menus



```
RXLEV=Pt+Gt+Gr-Lf
RXLEV -73 dBm
```

```
USAT TERMINAL RESPONSE
OK   91 E0  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - SNR=  07 dBSNR=Ps(dBm)+Pn(dBm)S...
USAT TERMINAL RESPONSE
OK   90 00  : OK
USAT ENVELOPE: Menu selection
OK   91 0D  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT -
USAT TERMINAL RESPONSE
OK   91 2E  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXLEV=Pt+Gt+Gr-LfRXLEV -73 dBm
```

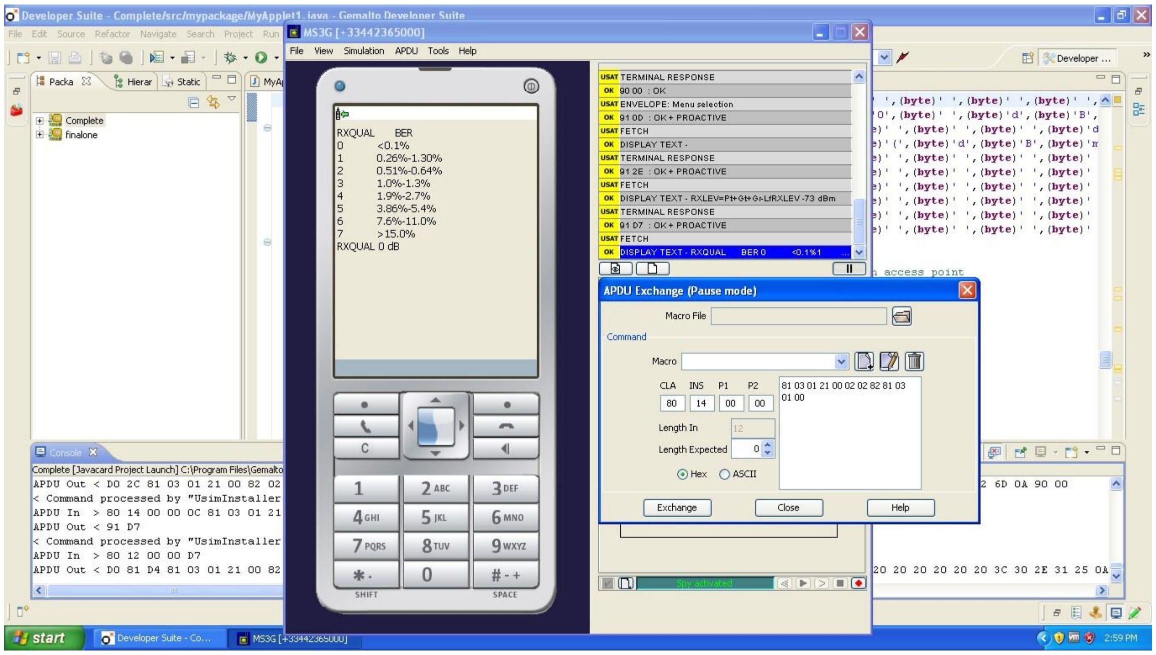
APDU Exchange (Pause mode)

| CLA | INS | P1 | P2 | |
|---|---|---|---|---|
| 80 | 14 | 00 | 00 | 81 03 01 21 00 02 02 82 81 03 01 00 |

Length In: 12
Length Expected: 0
(● Hex ○ ASCII)

```
Complete [Javacard Project Launch] C:\Program Files\Gemalto
APDU Out < D0 0B 81 03 01 21 00 82 02
< Command processed by "UsimInstaller
APDU In  > 80 14 00 00 0C 81 03 01 21
APDU Out < 91 2E
< Command processed by "UsimInstaller
APDU In  > 80 12 00 00 2E
APDU Out < D0 2C 81 03 01 21 00 82 02
```

Figure 5-3: Rxlevel formula and measurement


Developer Suite - Complete/src/mypackage/MyApplet1.java - Gemalto Developer Suite
MS3G [+33442365000]
File View Simulation APDU Tools Help
APDU Exchange (Pause mode)
Macro File
Command
Macro
Exchange
Close
Help

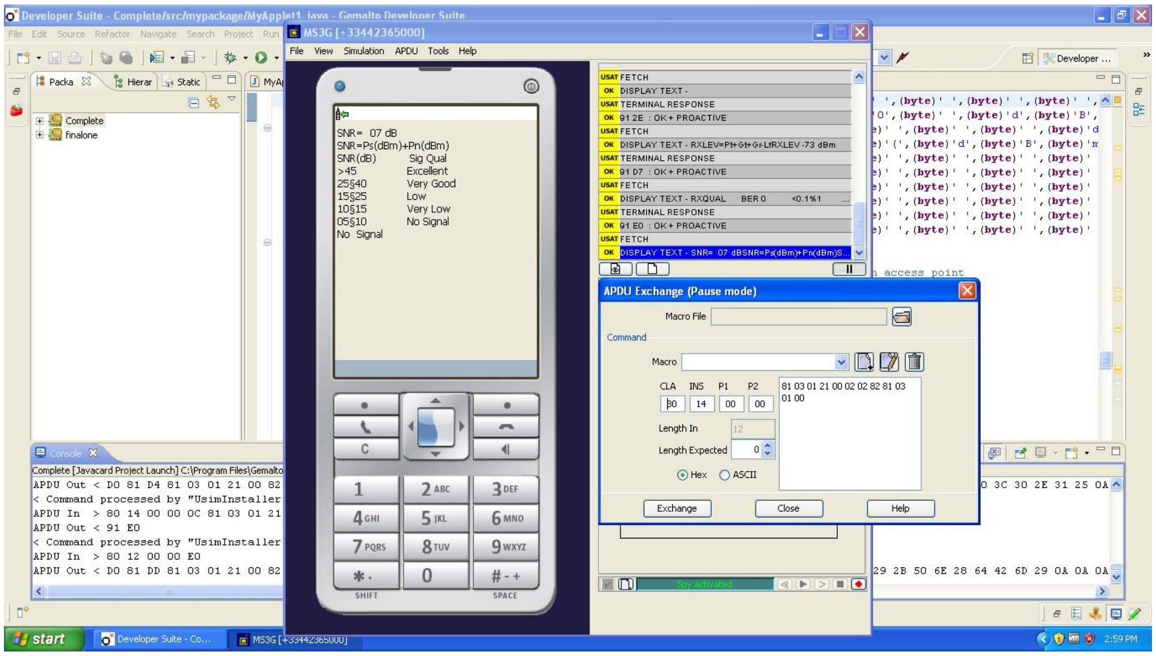
```
RXQUAL     BER
0          <0.1%
1          0.26%-1.30%
2          0.51%-0.64%
3          1.0%-1.3%
4          1.9%-2.7%
5          3.86%-5.4%
6          7.6%-11.0%
7          >15.0%
RXQUAL 0 dB
```

```
USAT TERMINAL RESPONSE
OK   90 00  : OK
USAT ENVELOPE: Menu selection
OK   91 0D  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT -
USAT TERMINAL RESPONSE
OK   91 2E  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXLEV=Pt+Gt+Gr-LfRXLEV -73 dBm
USAT TERMINAL RESPONSE
OK   91 D7  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXQUAL     BER 0      <0.1%1    ...
```

CLA 80, INS 14, P1 00, P2 00; data: 81 03 01 21 00 02 02 82 81 03 01 00; Length In 12; Length Expected 0; (● Hex) (○ ASCII)

```
Complete [Javacard Project Launch] C:\Program Files\Gemalto
APDU Out < D0 2C 81 03 01 21 00 82 02
< Command processed by "UsimInstaller
APDU In  > 80 14 00 00 0C 81 03 01 21
APDU Out < 91 D7
< Command processed by "UsimInstaller
APDU In  > 80 12 00 00 D7
APDU Out < D0 81 D4 81 03 01 21 00 82
```

Figure 5-4: RxQuality and Bit Error Rate


Developer Suite - Complete/src/mypackage/MyApplet1.java - Gemalto Developer Suite
MS3G [+33442365000]
File View Simulation APDU Tools Help
APDU Exchange (Pause mode)
Macro File
Command
Macro
Exchange
Close
Help

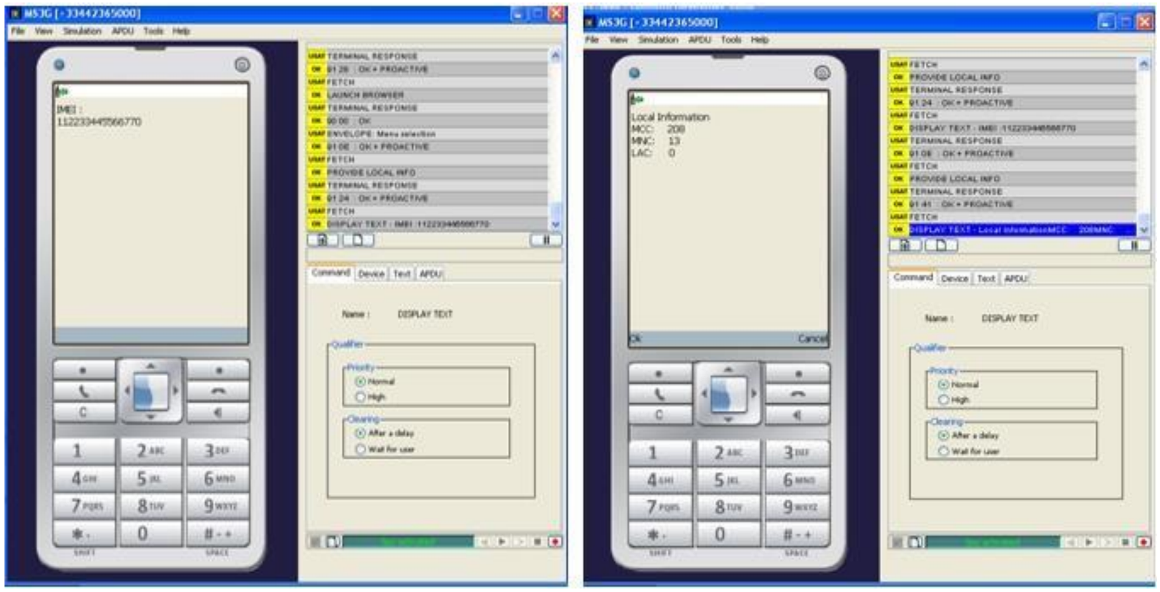
```
SNR=  07 dB
SNR=Ps(dBm)+Pn(dBm)
SNR(dB)        Sig Qual
>45            Excellent
25§40          Very Good
15§25          Low
10§15          Very Low
05§10          No Signal
No  Signal
```

```
USAT FETCH
OK   DISPLAY TEXT -
USAT TERMINAL RESPONSE
OK   91 2E  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXLEV=Pt+Gt+Gr-LfRXLEV -73 dBm
USAT TERMINAL RESPONSE
OK   91 D7  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - RXQUAL     BER 0      <0.1%1    ...
USAT TERMINAL RESPONSE
OK   91 E0  : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - SNR=  07 dBSNR=Ps(dBm)+Pn(dBm)S...
```

CLA 80, INS 14, P1 00, P2 00; data: 81 03 01 21 00 02 02 82 81 03 01 00; Length In 12; Length Expected 0; (● Hex) (○ ASCII)

```
Complete [Javacard Project Launch] C:\Program Files\Gemalto
APDU Out < D0 81 D4 81 03 01 21 00 82
< Command processed by "UsimInstaller
APDU In  > 80 14 00 00 0C 81 03 01 21
APDU Out < 91 E0
< Command processed by "UsimInstaller
APDU In  > 80 12 00 00 E0
APDU Out < D0 81 DD 81 03 01 21 00 82
```

Figure 5-5: Signal to noise ratio formula and calculation



MS3G [+33442365000]

IMEI :
112233445566770

```
USAT TERMINAL RESPONSE
OK   91 28 : OK + PROACTIVE
USAT FETCH
OK   LAUNCH BROWSER
USAT TERMINAL RESPONSE
OK   90 00 : OK
USAT ENVELOPE: Menu selection
OK   91 0E : OK + PROACTIVE
USAT FETCH
OK   PROVIDE LOCAL INFO
USAT TERMINAL RESPONSE
OK   91 24 : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - IMEI :112233445566770
```

Name : DISPLAY TEXT

Qualifier
Priority: [✓ Normal] High
Clearing: [✓ After a delay] Wait for user

MS3G [+33442365000]

Local Information
MCC: 208
MNC: 13
LAC: 0

```
USAT FETCH
OK   PROVIDE LOCAL INFO
USAT TERMINAL RESPONSE
OK   91 24 : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - IMEI :112233445566770
USAT TERMINAL RESPONSE
OK   91 0E : OK + PROACTIVE
USAT FETCH
OK   PROVIDE LOCAL INFO
USAT TERMINAL RESPONSE
OK   91 41 : OK + PROACTIVE
USAT FETCH
OK   DISPLAY TEXT - Local InformationMCC:   208MNC:
```

Name : DISPLAY TEXT

Qualifier
Priority: [✓ Normal] High
Clearing: [✓ After a delay] Wait for user

Figure 5-6: IMEI number and Local Area Number

# Chapter 6: Conclusion and Future work

- The software does not support much of the libraries and structure which are being used in the Gemalto development suite. And the literature regarding the SIM and its programming is very scarce. The hardware and software are difficult to receive. It is easily available to the network operators but are neither affordable nor available for a common individual.
- Requirement of our supervisor was that we had to calculate the Key performance Indicators using our telecommunications concept and then we had to implement and simulate it as an applet on the development suite. The parameters were the basic ones RxLevel, RxQuality, SNR, BER, Local Area information and IMEI number as other parameters can be taken out if the basic have been calculated.
- It is still a research project, no business model has been implemented yet so it needs many improvements. Most importantly such a software is required which supports maximum libraries so that the issue of calculating the remaining KPIs is resolved. And converting it all on Over The Air would be of much importance if this project is implemented and used by the network operators.

# Bibliography

[1] R. Budhiraja, J.S. Jadon, "Study and Implementation of Drive Test for Development of GSM Network", *International Journal of Engineering Trends and Technology (IJETT)*, vol. 4, no. 10, Oct 2013.

[2] Android based Drive Test platform for cellular system by Ibtihal Ahmed, Rufaida Mohamed and Ghassan Abdalla, *Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE),* 2015.

[3] "Sim Cards for Application development" by Peter Edsbäcker Mid Sweden University, Sweden (2010).

[4] "Drive Test Measurements between Maxis 2G and 3G Networks in Uitm Shah Alam Campus" presented in IEEE Student Conference on Research and Development in 2013.

[5] "A Novel Approach for Mobile Network QoS Evaluation" presented in 2014 published by IEEE.

# Appendix:

```
/**
          * Manage the Menu3 selection
         <b>RXLEV=Pt+Gt+Gr-Lf</b>
         Pt=TX O/P Power
         Gt=TX antenna Gain
         Gr=RX antenna Gain
         Lf=path Loss

          */
         private byte []
RXLEV={(byte)'R',(byte)'X',(byte)'L',(byte)'E',(byte)'V',(byte)'=',(byte)'P',(byte)'t',(byte)'
+',(byte)'G',(byte)'t',(byte)'+',(byte)'G',(byte)'r',(byte)'-
',(byte)'L',(byte)'f',(byte)'\n',(byte)'R',(byte)'X',(byte)'L',(byte)'E',(byte)'V',(byte)' ',(byte)'-
',(byte)'7',(byte)'3',(byte)' ',(byte)'d',(byte)'B',(byte)'m',(byte)'\n'};
       private byte [] RXQUAL={};
       private byte[] SNR={};          private
byte[] RXLEV_Formula={};
         /*
* In GPRS standard, RxQUAL stands for Receiver quality and BER stands          * for Bit

  Error Rate.

* RxQUAL parameter ranges between values 0-7 and it corresponds to

* range of BER values.



    These values are calculated on a block basis i.e. four bursts and       then

averaged. The RxQUAL corresponds to this averaged value of BER.

    This value is applicable to speech and not packet based data transfer.

* RxQUAL   BER
   RXQUAL_0        Less  than  0.1
                   %
```

| | |
|---|---|
| RXQUAL_1 | 0.26% to 0.30% |
| RXQUAL_2 | 0.51% to 0.64% |
| RXQUAL_3 | 1.0% to 1.3% |
| RXQUAL_4 | 1.9% to 2.7% |
| RXQUAL_5 | 3.8% to 5.4% |
| RXQUAL_6 | 7.6% to 11.0% |
| RXQUAL_7 | Greater than 15.0% |

As part of radio environment monitoring mobile station performs following measurements:

- Received signal level (RXLEV) measurements
- Quality (RXQUAL) measurements
- Interference measurements
- During packet-transfer mode the mobile estimates the quality of the received downlink blocks
- The RXQUAL is computed from the average BER before channel decoding. The RXQUAL can be used by the network for network-controlled cell reselection, dynamic coding scheme adaptation, and downlink power control.
- In packet idle mode, no quality measurements are performed.

```
    */
    private byte[]
BER={(byte)'R',(byte)'X',(byte)'Q',(byte)'U',(byte)'A',(byte)'L',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)'B',(byte)'E',(byte)'R',(byte)' ',(byte)'\n',
                    (byte)'0',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)'<',(byte)'0',(byte)'.',(byte)'1',(byte)'%',(byte)'\n',
                    (byte)'1',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)'
',(byte)'0',(byte)'.',(byte)'2',(byte)'6',(byte)'%',(byte)'',(byte)'0',(byte)'.',(byte)'3',
(byte)'0',(byte)'%',(byte)'\n',
                              (byte)'2',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)'
',(byte)'0',(byte)'.',(byte)'5',(byte)'1',(byte)'%',(byte)'',(byte)'0',(byte)'.',(byte)'6',
(byte)'4',(byte)'%',(byte)'\n',
                              (byte)'3',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)'1',(byte)'.',(byte)'0',(byte)'%',(byte)'-
',(byte)'1',(byte)'.',(byte)'3',(byte)'%',(byte)'\n',
                              (byte)'4',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)'1',(byte)'.',(byte)'9',(byte)'%',(byte)'-
',(byte)'2',(byte)'.',(byte)'7',(byte)'%',(byte)'\n',
                              (byte)'5',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)'3',(byte)'.',(byte)'8',(byte)'6',(byte)'%',(byte)'-
',(byte)'5',(byte)'.',(byte)'4',(byte)'%',(byte)'\n',
                              (byte)'6',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)'
',(byte)'7',(byte)'.',(byte)'6',(byte)'%',(byte)'',(byte)'1',(byte)'1',(byte)'.',
    (byte)'0',(byte)'%',(byte)'\n',
                              (byte)'7',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)'
',(byte)'>',(byte)'1',(byte)'5',(byte)'.',(byte)'0',(byte)'%',(byte)'\n',(byte)'\n',(byte)'\n',(byte)'\
n',
                              (byte)'R',(byte)'X',(byte)'Q',(byte)'U',(byte)'A',(byte)'L',(byte)'
',(byte)'0',(byte)' ',(byte)'d',(byte)'B',(byte)'\n'};
              private byte[] SNR1={(byte)'S',(byte)'N',(byte)'R',(byte)'=',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)'d',(byte)'B',(byte)'\n',

(byte)'S',(byte)'N',(byte)'R',(byte)'=',(byte)'P',(byte)'s',(byte)'(',(byte)'d',(byte)'B',(byte)'m',
(byte)')',(byte)'+',(byte)'P',(byte)'n',(byte)'(',(byte)'d',(byte)'B',(byte)'m',(byte)')',(byte)'\n',(
byte)'\n',(byte)'\n',(byte)'S',(byte)'N',(byte)'R',(byte)'(',(byte)'d',(byte)'B',(byte)')',
                                   (byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)'S',(byte)'i',(byte)'g',(byte)'
',(byte)'Q',(byte)'u',(byte)'a',(byte)'l',(byte)'\n',(byte)'\n',(byte)'\n',
                                   (byte)'>',(byte)'4',(byte)'5',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)'E',(byte)'x',(byte)'c',(byte)'e',(byte)'l',(byte)'l',(byte)'e',(byte)'n',(byte)'t',(byte)'\n',
                            (byte)'2',(byte)'5',(byte)'_',(byte)'4',(byte)'0',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)'V',(byte)'e',(byte)'r',(byte)'y',(byte)' ',(byte)'G',(byte)'o',(byte)'o',(byte)'d',(byte)'\n',
                                   (byte)'1',(byte)'5',(byte)'_',(byte)'2',(byte)'5',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
```

```
',(byte)'L',(byte)'o',(byte)'w',(byte)' ',(byte)'\n',
                                        (byte)'1',(byte)'0',(byte)'_',(byte)'1',(byte)'5',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)'V',(byte)'e',(byte)'r',(byte)'y',(byte)' ',(byte)'L',(byte)'o',(byte)'w',(byte)'\n',
                                        (byte)'0',(byte)'5',(byte)'_',(byte)'1',(byte)'0',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)'N',(byte)'o',(byte)'
',(byte)'S',(byte)'i',(byte)'g',(byte)'n',(byte)'a',(byte)'l',(byte)'\n',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)' ',(byte)'
',(byte)' ',(byte)' '};
         /*
* S/N = 20 log10(Vs/Vn)
* When performing a RF site survey, it's important to define the range boundary of an

  access point     based on signal-to-noise (SNR) ratio, which is the signal level (in dBm)

  minus the noise level

          (in dBm).

          SNR=Signal level(dBm)-Noise level(dBm)

* > 40dB SNR = Excellent signal (5 bars); always associated; lightning fast.

    25dB to 40dB SNR = Very good signal (3 - 4 bars); always associated; very fast.

    15dB to 25dB SNR = Low signal (2 bars); always associated; usually fast.

    10dB - 15dB SNR = Very low signal (1 bar); mostly associated; mostly slow.

    5dB to 10dB SNR = No signal; not associated; no go.

          */
      private byte snr=(byte)(48+6);
      private byte[] res=new byte[3];
      //private byte[] SNR_formula={};
      private byte ber=(byte)(48+1.3);
      private short l;          private
byte[] g;      private void menu3Action()
{
              /**@todo: Replace following sample code with your implementation*/
              // Get the received envelope
                     ProactiveHandler proHdlr = ProactiveHandler.getTheHandler();
```

```
                    ProactiveResponseHandler ProRespHdlr;
                    ProRespHdlr = ProactiveResponseHandler.getTheHandler();
                    //l=(short)BER.length;
                    //g[0]=(byte)'L';
                    //g[1]=(byte)'=';
          //g[2]=(byte)(48+l);
                    // Display the "Menu4" message text
                    // Initialize the display text command
                    //proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA, g, (short) 0,
                              //(short) (g.length));
                    //proHdlr.send();
                    //NETWORK MEASURMENT RESULTS

               //proHdlr.init(PRO_CMD_PROVIDE_LOCAL_INFORMATION, (byte)0x02,
DEV_ID_ME);

//proHdlr.appendTLV(TAG_NETWORK_MEASUREMENT_RESULTS,(byte)0x01);
             //proHdlr.send();

//ProRespHdlr.findAndCopyValue(TAG_NETWORK_MEASUREMENT_RESULTS
,network_measurement, (short)0x00);
            proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA,RXLEV_Formula,
(short) 0x00,
                              (short) (RXLEV_Formula.length));
            proHdlr.send();
            proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA,RXLEV, (short)
0x00,
                 (short) (RXLEV.length));
proHdlr.send();
              //proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA,BER, (short) 0x00,
            //                (short) (BER.length));
             //proHdlr.send();
             if(ber<(byte)(48+0.1)){
                    BER[51]=(byte)(48+0);
            }
             else if(ber>=(byte)(48+0.26)||ber<=(byte)(48+0.30)){
                    BER[51]=(byte)(48+1);
             }

             else if(ber>=(byte)(48+0.51)||ber<=(byte)(48+0.64)){
```

```
                        BER[51]=(byte)(48+2);
                  }
                  else if(ber>=(byte)(48+1.0)||ber<=(byte)(48+1.3)){
                        BER[51]=(byte)(48+3);
                  }
                  else if(ber>=(byte)(48+1.9)||ber<=(byte)(48+2.7)){
                        BER[51]=(byte)(48+4);
                  }
                  else if(ber>=(byte)(48+3.8)||ber<=(byte)(48+5.4)){
                        BER[51]=(byte)(48+5);
                  }
                  else if(ber>=(byte)(48+7.6)||ber<=(byte)(48+11)){
                      BER[51]=(byte)(48+6);
                  }
                  else if(ber>(byte)15){
                        BER[51]=(byte)(48+7);
                  }
                  proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA,BER, (short) 0x00,
                                (short) (BER.length));
                proHdlr.send();
      if(snr>(byte)(48+40)){
              SNR1[6]=(byte)(48+4);
                     SNR1[7]=(byte)(48+5);
                     SNR1[191]='\n';
                        SNR1[192]='E';
                        SNR1[193]='x';
                        SNR1[194]='c';
                        SNR1[195]='e';
                        SNR1[196]='l';
                        SNR1[197]='l';
                        SNR1[198]='e';
                        SNR1[199]='n';
                        SNR1[200]='t';
                        SNR1[201]=' ';
                        SNR1[202]=' ';
                        SNR1[203]='Q';
                        SNR1[204]='u';
                        SNR1[205]='a';
                        SNR1[206]='l';
                        SNR1[207]='i';
```

```
        SNR1[208]='t';
        SNR1[209]='y';
}

 else if(snr>=(byte)(48+25)&& snr<=(byte)(48+40)){
        SNR1[6]=(byte)(48+3);
        SNR1[7]=(byte)(48+3);
    SNR1[191]='\n';
        SNR1[192]='V';
        SNR1[193]='e';
        SNR1[194]='r';
        SNR1[195]='y';
        SNR1[196]=' ';
        SNR1[197]='G';
        SNR1[198]='o';
        SNR1[199]='o';
        SNR1[200]='d';
        SNR1[201]=' ';
        SNR1[202]=' ';
        SNR1[203]='Q';
        SNR1[204]='u';
        SNR1[205]='a';
        SNR1[206]='l';
        SNR1[207]='i';
        SNR1[208]='t';
        SNR1[209]='y';
 }
 else if(snr>=(byte)(48+15)&& snr<=(byte)(48+24)){
        SNR1[6]=(byte)(48+2);
        SNR1[7]=(byte)(48+1);
    SNR1[191]='\n';
        SNR1[192]='L';
        SNR1[193]='o';
        SNR1[194]='w';
        SNR1[195]=' ';
        SNR1[196]=' ';
        SNR1[197]='S';
        SNR1[198]='i';
        SNR1[199]='g';
```

```
            SNR1[200]='n';
SNR1[201]='a';
              SNR1[202]='l';
              SNR1[203]=' ';
              SNR1[204]=' ';
              SNR1[205]='Q';
              SNR1[206]='u';
              SNR1[207]='a';
            SNR1[208]='l';
              SNR1[209]='i';
              SNR1[210]='t';
              SNR1[211]='y';
        }
        else if(snr>=(byte)(48+10)&&snr<=(byte)(48+14)){
              SNR1[6]=(byte)(48+1);
              SNR1[7]=(byte)(48+7);
           SNR1[191]='\n';
              SNR1[192]='V';
              SNR1[193]='e';
              SNR1[194]='r';
              SNR1[195]='y';
              SNR1[196]=' ';
              SNR1[197]='L';
              SNR1[198]='o';
              SNR1[199]='w';
              SNR1[200]=' ';
              SNR1[201]='S';
              SNR1[202]='i';
              SNR1[203]='g';
              SNR1[204]='n';
              SNR1[205]='a';
              SNR1[206]='l';
              SNR1[207]=' ';
              SNR1[208]='Q';
              SNR1[209]='u';
              SNR1[210]='a';
              SNR1[211]='l';
              SNR1[212]='i';
              SNR1[213]='t';
              SNR1[214]='y';
```

```
                }
                else if(snr>=(byte)(48+5)&&snr<=(byte)(48+9)){
                    SNR1[6]=(byte)(48+0);
                    SNR1[7]=(byte)(48+7);
                      SNR1[191]='\n';
                      SNR1[192]='N';
                      SNR1[193]='o';
                      SNR1[194]=' ';
                      SNR1[195]=' ';

                      SNR1[196]='S';
                      SNR1[197]='i';
                      SNR1[198]='g';
                    SNR1[199]='n';
         SNR1[200]='a';
         SNR1[201]='l';}
                proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA,SNR1, (short) 0x00,
                            (short) (SNR1.length));
proHdlr.send();


                /*proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA,SNR, (short)
0x00,
                            (short) (SNR.length));
proHdlr.send();*/
                //proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA,res, (short) 0x00,
                      //        (short) (res.length));
                //proHdlr.send();

             /*else
                      {proHdlr.initDisplayText((byte) 0x00, DCS_8_BIT_DATA,failure,
(short) 0,
                                     (short) failure.length);
         proHdlr.send();
                      }*/

                      return;
             }
```