Universidade de Aveiro
Ano 2017

Departamento de Eletrónica,

Telecomunicações e Informática

Tomás Marques Rodrigues

End-user quality of service and experience in mobile networks

Qualidade de serviço e experiência em redes móveis na ótica do utilizador final

Universidade de Aveiro
Ano 2017

Departamento de Eletrónica,

Telecomunicações e Informática

Tomás Marques Rodrigues

End-user quality of service and experience in mobile networks

Qualidade de serviço e experiência em redes móveis na ótica do utilizador final

"If your dreams do not scare you, they are not big enough"

– Ellen Johnson Sirleaf

Universidade de Aveiro
Ano 2017

Departamento de Eletrónica,

Telecomunicações e Informática

Tomás Marques Rodrigues

End-user quality of service and experience in mobile networks

Qualidade de serviço e de experiência em redes móveis na ótica do utilizador final

Tese apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor João Paulo Silva Barraca, Professor assistente convidado do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

agradecimentos

Ao meu orientador, João Paulo Barraca pela orientação e todo o apoio dado que levou a bom fim a escrita desta dissertação.

A toda a equipa da AlticeLabs associada ao projeto, e em especial ao meu co-orientador, Pedro Fernandes. Obrigado pela forma como me acolheram e me integraram no vosso grupo de trabalho. Foi uma ótima ajuda para que tudo corresse pelo melhor.

Aos meus pais e especialmente à minha irmã, o maior agradecimento. Para vos agradecer tudo o que já fizeram por mim eu precisava de uma tese inteira e não apenas de uma secção nos agradecimentos, vocês foram o meu pilar ao longo destes anos. Obrigado por terem sempre acreditado em mim e por me ajudarem a conseguir fazer sempre mais e melhor. Obrigado por me mostrarem que não interessam as notas ou os graus que eu consiga, que aquilo que interessa é eu seguir o meu sonho e ser feliz. Obrigado por todos os sacrifícios que já fizeram por mim. Obrigado por não me deixarem sentir mal mesmo quando as coisas correm menos bem e por toda a força no sentido de continuar. Obrigado por estarem sempre presentes. Muito, muito obrigado, sem vocês isto nunca teria sido possível.

À minha namorada, Joana e a todos os amigos, por toda a disponibilidade e por me apoiarem nas minhas derrotas e por festejarem comigo as minhas vitórias.

palavras-chave

Qualidade de serviço, qualidade de experiência, rede móvel, aplicação móvel, Android SDK

resumo

Os operadores de redes móveis recorrem a equipamentos dedicados (sondas) para obtenção de métricas relativas ao desempenho, QoS e QoE das suas redes e serviços. Pretende-se desenvolver uma app Android com funcionalidades de probing, não só complementando os equipamentos dedicados na recolha de informação relativa ao desempenho, QoS e QoE, como também detetando problemas ao nível da rede e dos seus serviços automaticamente no próprio terminal do cliente final, como também disponibilizando ferramentas de teste ao cliente e ao suporte para uma maior celeridade na resolução de problemas.

keywords

Quality of service, quality of experience, mobile networks, mobile app, Android SDK

abstract

Mobile network operators use dedicated equipment (probes) to obtain performance, QoS and QoE metrics for their networks and services. It is intended to develop an Android App with probing features, not only complementing the dedicated equipment in the collection of information regarding performance, QoS and QoE, but also detecting problems at the network and its services automatically in the final client terminal and providing customer test tools and support for faster troubleshooting.

Contents

List of figures

List of tables

List of snippets

Acronyms

2

2G – Second generation mobile networks

3

3G – Third generation mobile networks
3GPP – 3rd Generation Partnership Project

4

4G – Fourth generation of wireless mobile telecommunications technology

5

5G – Fifth generation mobile networks (not yet standardized)

A

ADB – Android Debug Bridge
API – Application Programming Interface
APN – Access Point Name
AuC – Authentication Center
AMPS – Advance Mobile Phone Service

B

BSC – Base Station Controller
BSS – Base Station Subsystem
BTS – Base Transceiver Station
BSSID – Basic Service Set Identifier

C

CO_2 – Carbon Dioxide
C-RAN – Cloud Radio Access Network
CDMA – Code Division Multiple Access
CEPT – European Conference of Postal and Telecommunications Administrations
CSFB – Circuit Switched Fallback
CAPEX – Capital Expenditure

D

DL – Downlink
DVB – Digital Video Broadcasting
DMTF – Dual Tone – Multi-Frequency

E

EIR –	Equipment Identity Register
EDGE –	Enhanced Data rate for GSM Evolution
ETSI –	European Telecommunications Standards Institute
EUTRAN –	Evolved Universal Terrestrial Radio Access Network

G

GSM –	Global System for Mobile Communications
GGSN –	Gateway GPRS Support Node
GSMA –	Global System for Mobile Communications Association
GPRS –	General Packet Radio Service

H

HLR –	Home Location Register
HTTP –	Hypertext Transfer Protocol
HSPA+ –	Evolved High-Speed Packet Access
HSCSD –	High-Speed Circuit Switched Data
HSDPA –	High-Speed Downlink Packet Access
HSUPA –	High-Speed Uplink Packet Access

I

IP –	Internet Protocol
IMS –	IP Multimedia Service
IoT –	Internet of Things
ISP –	Internet Service Provide
ICMP –	Internet Control Message Protocol
IEEE –	Institute of Electrical and Electronics Engineers
IMEI –	International Mobile Equipment Identity
IMSI –	International Mobile Subscriber Identity
IPv4 –	Internet Protocol version 4
IMAP4 –	Internet Message Access Protocol

J

JSON –	JavaScript Object Notation

K

KPI –	Key Performance Indicator
KQI –	Key Quality Indicator

L

LTE –	Long Term Evolution

M

ME – Mobile Equipment
MS – Mobile Station
MAC – Media Access Control
MCC – Mobile Country Code
MNC – Mobile Network Code
MMS – Multimedia Messaging Service
MOS – Mean Opinion Score
MSC – Mobile Switching Center
Mbps – Megabits per second
MIMO – Multiple Input Multiple Output

N

NFC – Near Field Communication
NMT – Nordic Mobile Telephone
NSS – Network Switching Subsystem

O

OS – Operating System
OEM – Original Equipment Manufacturers
OTA – Over the Air
OTT – Over the Top
OPEX – Operational Expenditure

P

P2P – Peer-to-Peer
PCU – Packet Controller Unit
PIN – Personal Identification Number
PESQ – Perceptual Evaluation of Speech Quality
POP3 – Post Office Protocol
PLMN – Public Land Mobile Network
PSTN – Public Switched Telephone Network
POLQA – Perceptual Objective Listening Quality Assessment

Q

QoE – Quality of Experience
QoS – Quality of Service

R

RF – Radio frequency
RNC – Radio Network Controller
RNS – Radio Network Subsystem

ROM – Read Only Memory
RUU – ROM Update Utility

S

SAE – System Architecture Evolution
SCP – Secure Copy
SS7 – Signaling System No. 7
SIM – Subscriber Identity Module
SFTP – SSH File Transfer Protocol
SGSN – Serving GPRS Support Node
SMTP – Simple Mail Transfer Protocol
SSID – Service Set Identifier

T

TACS – Total Access Communication System

U

UL – Uplink
URL – Uniform Resource Locator
UMTS – Universal Mobile Telecommunications System

V

VLR – Visitor Location Register
VoIP – Voice over IP (Internet Protocol)
VoLTE – Voice over LTE

W

WAP – Wireless Application Protocol
WCDMA – Wideband Code Division Multiple Access
Wi-Fi – Wireless Fidelity
WLAN – Wireless Local Area Network
WMAN – Wireless Metropolitan Area Network
WPAN – Wireless Personal Area Network

Chapter 1

Introduction

Human relationships are based on communication and the technology is changing to improve the way we interact with each other. Telecommunications evolved from analog services to the digital, including not only voice but also data services with better debits, capacity and lower latency. The requirement for higher data speed on smartphones is increasing rapidly, much due to the usage of social networks and other entertainment data in these small devices. Constant improvement in wireless data rate is already happening and different network technologies are integrated to provide seamless connectivity and transparency to user, making the network appear heterogeneous despite the complexity involved.

User's expectations are always growing with new services appearing constantly and the quality of service needs to be a constant improvement in order to follow this technological evolution. Although the internet was designed to provide services without quality assurances, in the case of operators they are contractually obliged to ensure certain quality in some services provided by them and working with clients that more and more want to be always connected and with mobility. Due to this a lot of work has still to be done to grant good quality, performance and experience to the user.

1.1 Motivation

Technology is, more and more, part of the daily life of the human being and according to GSMA, the number of mobile connections already surpassed the number of people on earth and it's growing five times faster [1]. Humanity communicates on a global scale thanks to the increasing development of mobile devices technology. Computing and communication had led to a notorious evolution of mobile devices, which have become not only a mean of communication but also a way of accessing extensive functionalities. The broadband speed let us view videos with high resolution or take photos everywhere, becoming this small device that fits in our pocket the communication and entertainment tool of today's election.

The rapid growth of wireless communications allowed the rising number of smartphones and tablets with battery improvements and connected in a real-time, faster network. Evaluate the network and what is happening is extremely important to the operator to assurance good quality of service to his clients and maintains them. Given the importance of this, operators have fixed and mobile probes to try to give the best user experience and grant network availability and performance. The current dissertation fits by adding a more flexible, transparent and dynamic solution to improve network service quality with the increasing functionalities and technology on these small devices.

1.2 Objectives

The key objective of this dissertation is to develop a solution that retrieves QoS metrics from the network and useful radio parameters dependent on the access technology being used in the moment with an Android smartphone to increase the QoE experienced by user. With this application, is still intended to run tests on the network (e.g. check the internet speed) to get more information and troubleshoot possible problems with it and observe the data gathered over time in a simple and attractive interface for the final user.

This solution is connected to a backend sending all data to a unified platform called ArQoS, a centralized and convergent product that evaluates the customer perceived quality in service usage (Voice, IPTV, SMS, MMS, e-mail, Internet, …), multi-technology and in multi-vendor environments in order to increase customer satisfaction and optimize resources in case of the operator [2].

Considering these points, there are a vast case of scenarios like in drive tests through the city since it's only needed a regular smartphone and can be used by operator's technicians to identify concerning locations that needs better coverage, to know how the network is working in real time with real metrics or used by a regular user to check internet connectivity or the downlink speed in that moment.

1.3 Project Structure

This dissertation is split into 6 chapters of which, chapter 1, Introduction, was already presented. The remaining chapters are:

- **Chapter 2:** presentation of the state of art. The core concepts of quality of service, experience and cellular networks are discussed in this chapter. Distinct mobile operations systems, as well as some new technologies like VoLTE and VoWi-Fi, are also presented in this chapter. Additionally, it is also given a detailed analysis of some solutions proposals relevant to the problem;
- **Chapter 3:** starts by mentioning the main objectives and requirements defined for the solution implementation. An explanation of what fixed and mobile probes already does and what the smartphone can and cannot do in comparison is presented referencing all stakeholders involved in the solution progression. Lastly, it IS detailed the solution architecture.
- **Chapter 4:** description of the implemented solution and project architecture along with the technologies used and a detailed explanation of the followed approach during the implementation;
- **Chapter 5:** presentation and evaluation of the results obtained, as well of insights about those. A description of the test methodology is described and the objectives and test results are then analyzed.
- **Chapter 6:** final conclusions about the chosen path and obtained solution, also addressing potential improvements for future work.

Chapter 2

State of Art

2.1 Mobile Network Evolution

Mobile networking is a technology that supports voice and data using radio transmission. In the past these communications used circuit switching to carry voice over a network, nowadays both data and voice are transmitted over circuit-switched and packet-switched networks. The use of wireless communications in mobile communications started in the 1970s with the zeroth generation. Today we have reached the fourth generation and several countries are working on the architecture and development of 5G. This section will give an overview of this evolution over all these years.

The pre-cellular system was the first mobile communication technology, it worked using a central antenna mounted per region and strong transmitters and receivers were used to send and receive the data. Preceding cellular mobile telephony technology these systems are also known as **0G** (zero generation) [3].

Based on analog transmission the first mobile systems emerged, later known as **1G**. Using multiple cell sites and having the ability to transfer calls as the user traveled, the first generation wireless signal established seamless mobile connectivity introducing mobile voice services.

Mobile phones at the time were not as we know them today. They were extremely large, heavy, not comfortable to carry, power inefficient and had high costs for the standards at that time. They used FDMA for spectrum sharing but was required a large gap of spectrum between users to avoid interference. Other major problems associated were low traffic density of one call per radio, limited services, calls susceptible to noise, low data rates, inadequate fraud protection due to that anyone with a radio scanner could eavesdrop your call and poor security with unencrypted transmission, perhaps do not be fooled, connections to a tower miles away was at that time really impressive [4].

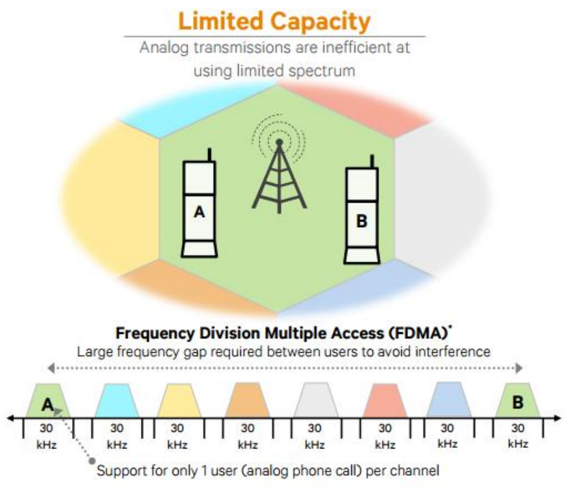

Limited Capacity
Analog transmissions are inefficient at using limited spectrum
A
B
Frequency Division Multiple Access (FDMA)*
Large frequency gap required between users to avoid interference
30 kHz
Support for only 1 user (analog phone call) per channel

Figure 2.1 - FDMA based on AMPS 1G technology [5]

In the 1990's, the **'second generation' 2G** mobile phone systems appeared, the first digital cellular network, bigger and better than 1G. The initial requirements for this network defined by CEPT/ETSI were: more efficiency with the radio frequency usage, low-cost cipher in terms of security, QoS, Numbering ITU-T and signaling protocol in the network.

This generation introduced the GSM architecture in 1991 enabling more users per channel with TDMA technique but still requiring a large gap of spectrum between users to avoid interference. This architecture is divided into three parts: Mobile Station (MS), Base Station Subsystem (BSS) and Switching Subsystem (NSS). The first one consists of Mobile Equipment (ME) and a Subscriber Identity Module (SIM) containing the subscriber identity, a password (PIN), subscription information such as last received/dialed numbers, last visited location area and more commonly used for authentication and other security procedures.

The BSS is composed by base station controller (BSC) and base transceiver station (BTS) placed in the center of a cell, a geographic area covered by a base station and maps transceivers and antennas to the cell, whose size is defined based on BTS's transmitting power. BSC manages radio resources of BTS's and handles BTS and MS power control, handovers, channel allocation and it also gathers the traffic towards the MSC [6].

At last the NSS it's composed by MSC containing four databases (HLR, VLR, AuC and EiR) and the GMSC. The MSC manages the communication between the mobiles and the fixed network, handles authentication and registration from connections with subsystem databases. Home Location Register (HLR) maintains permanent information about the subscribers of a GSM network and tracks the location and state of the mobile terminal within the network, Visitor

Location Register (VLR) maintains temporary information about the subscribers registered on a GSM network and keeps up-to-date information about the location of the user within the network, Authentication Center (AuC) is responsible for the authentication of the subscribers, maintains the encryption algorithms, the secret key for each subscriber and generates the session keys and finally Equipment Identity Register (EiR) provides security mechanisms and keeps a list of authorized/blocked mobile equipments.

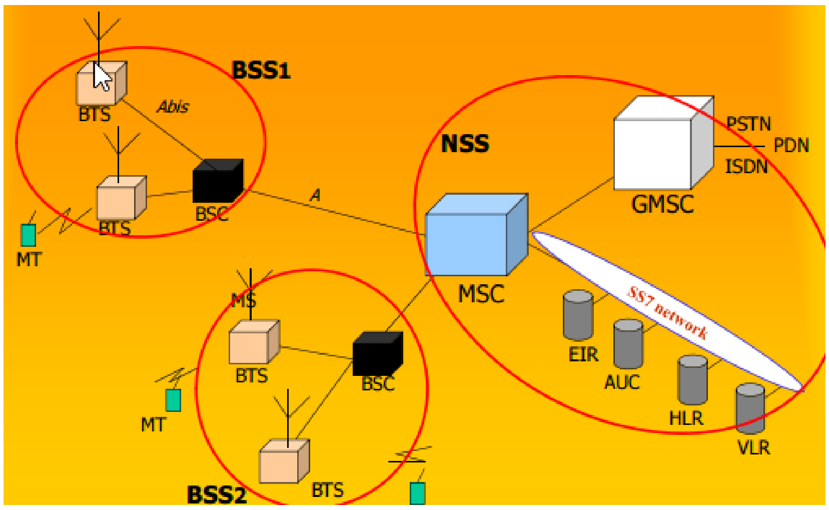


Figure 2.2 - GSM architecture [8]

The second generation introduced the SMS text, a new way to communicate and an alternative over voice calls nowadays preferred in many situations. In comparison with 1G digital signals consume less battery power making mobile batteries last longer. Voice clarity was improved, compressed into smaller "packages" and noise was reduced in the line. Furthermore, the cost/weight in digital components reduced a lot making possible have a pocket-sized device with more security and with encryption to the data on voice calls.

Mobile phone usage increased drastically and the first's prepaid mobile phones began to appear. However, although the network offers more capacity, the data was mostly just text messages and the delivery was rigid whether the user was or not actively talking. These constraints and 2G low air interface data rates were improved by High-Speed Circuit Switched Data (HSCSD) and General Packet Radio Services (GPRS).

HCSD uses multiple time slots per user (max 6) and increased data throughput at the time, however, since its circuit switched it allocates the time slots even when nothing is being transmitted increasing the blocking probability of the system.

On another front, GPRS was the new technology that made possible to make phone calls and transmit data at the same time. Its architecture provides both circuits switched and packet switched data for voice and data traffic respectively, although as the name indicates it's more a packet-oriented transport service. This period is seen as an extra period of mobile networking called **2.5G.** Lifetime network applications became to appear as well as new services such as Wireless Application Protocol (WAP) access, Multimedia Messaging Service (MMS) or e-mails. Bit rates transmission improvements from 56 Kbit/s up to 115 Kbit/s and new user-oriented billing mechanisms (by traffic e.g.), were an important step towards 3G and for the first evolution of GSM networks [4].

In comparison with GSM's architecture, in GPRS have been a BSS evolution with upgrades on BTS's, BSC's and network planning. A new element called Packet Controller Unit (PCU) that manages packets toward the radio interface was added. At NSS the core network was modified, adding new packet nodes dedicated to GPRS (SGSN, GGSN) and Internet equipment like DNS servers and firewalls.

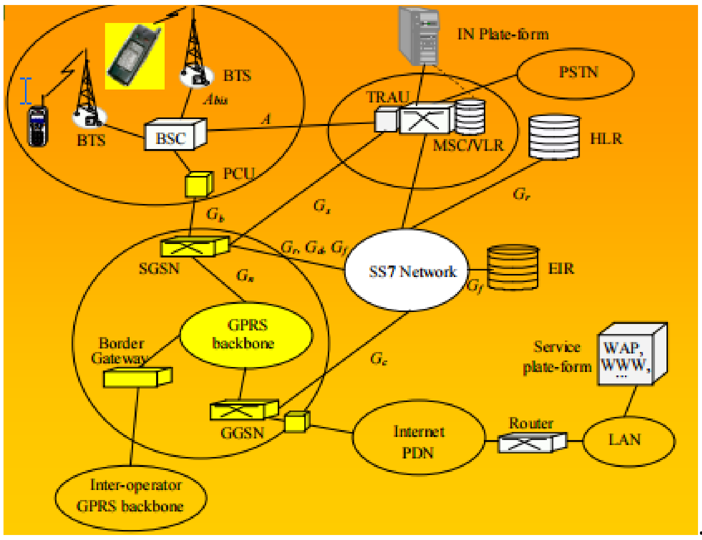


Figure 2.3 - GPRS architecture [8]

With no new technology required was found a way to double the transfer speed of 2.5G, but still was not fast enough to reach the 3G standards defined for the future. The idea was to change the modulation scheme to 8-PSK and with link adaptations, only software upgrades were required to increase the data rates. This technology was called EDGE or EGPRS and as explained it’s an extended only version of GSM.

Operators and providers' revenues were coming mainly from voice (>70%) and the mindset was migrating towards more applications, services and new systems. Technological progress and voice/data combination contributed as well to motivate a new system. Taking that in consideration European Telecommunications Standards Institute (ETSI) started the work, carried out later by Third Generation Partnership Project (3GPP) organization for the next generation mobile networks in a system called Universal Mobile Telecommunication System (UMTS) [10].

Combining aspects of 2G the 3GPP model idea was pass from a vertical model to an all uniformed service and transport layers to create a single set of services that apply network-wide, generic application servers, a common session control, a consistent user experience and improving operational efficiency to any device anywhere and in any access technology.

The core network of UMTS has base stations called now Node B and Radio Network Controllers (RNC). These components together form the Radio Network Subsystem (RNS) similarly to BSS that was seen before.

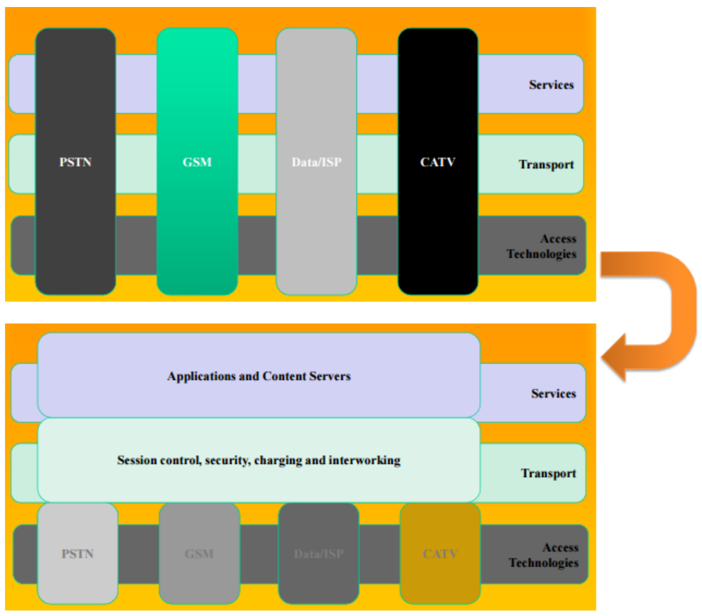


Figure 2.4 - Legacy Telecom Model and 3GPP Telecom Model respectively [8]

In release 5, IMS was introduced with the same core network and was the architectural foundation for next generation networks. It defined a common framework for delivering IP-based multimedia while maintaining QoS, billing and service integration. Release 6 extended to wideband fixed networks (xDSL, WLAN, cable, …) and supported services convergence on fixed and mobile networks (circuit-switched voice traffic in IP). For network providers, this brings a lot of advantages because a universal network reduces operational expenditures and service providers reach a new and broader marker with functions of authentication, charging and billing that could now, be outsourced [21].

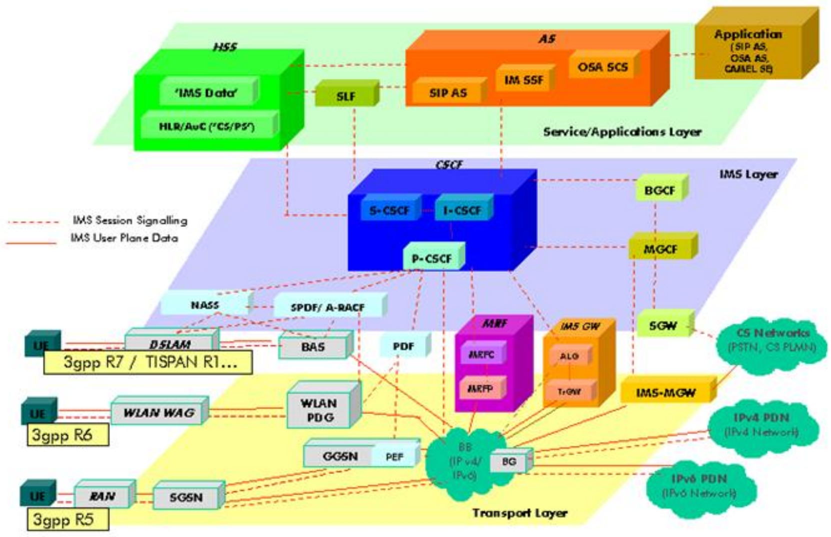


Figure 2.5 - 3GPP IMS architectural overview

Before the fourth generation, some sources consider an extra period, **3.5G** or **3G**+ where mobile telephony protocols like HSDPA and HSUPA provided a smooth evolution of data rates of 3G networks. Complimentary to each other, HSDPA is a packet-based data service in WCDMA downlink, already HSUPA is a UMTS / WCDMA uplink evolution technology that will enhance P2P data apps, e-mail, gaming, etc due to this boost in the link speed.

The successor of 3G UMTS was developed by 3GPP and is known as LTE. This **4G** network must deliver speeds of 100Mbps while moving, 1Gbps while stationary and smooth handover going from 4G to Wi-Fi and back with no data or call interruptions. It utilizes packet switching over the internet, LAN or WAN networks. The improvements on speed, reliability, capability, security and global mobility are also important upgrades from 3G technology.

Also known for the high-quality audio and video streaming, low cost in roaming networks and for providing services anytime, this network merges various numerous radio access interfaces into a single network that can be accessed by every subscriber.

The quality of service was one of many targets, in video chat, mobile TV, DVB and many other services that make use of bandwidth. Other targets include improved latency, spectrum efficiency, RF coverage, reduce CAPEX and OPEX and have compatibility with earlier releases, systems (including handover) and system wise providing a steady experience to the user [23].

LTE radio access network is called EUTRAN (Evolved Universal Terrestrial Radio Access Network), it also has an air interface called EUTRA (Evolved Universal Terrestrial Radio Access). The entire core network is known as EPC and it provides IP connectivity between the user equipment and an external packet data network using EUTRAN. Base stations are called eNB (EUTRAN Node B) and they take care of access control functions. Still, about the radio interface, it uses OFDMA, a multiple access technology on the downlink that transmits traffic across hundreds of parallel radio connections known as 'subcarriers'. An adapted version of this is SC-FDMA used on the uplink [7].

Mobile 4G LTE complements 3G providing more data capacity for richer content, more connections and along with speed and services, billing systems are also key aspects for the success of the 4G systems.

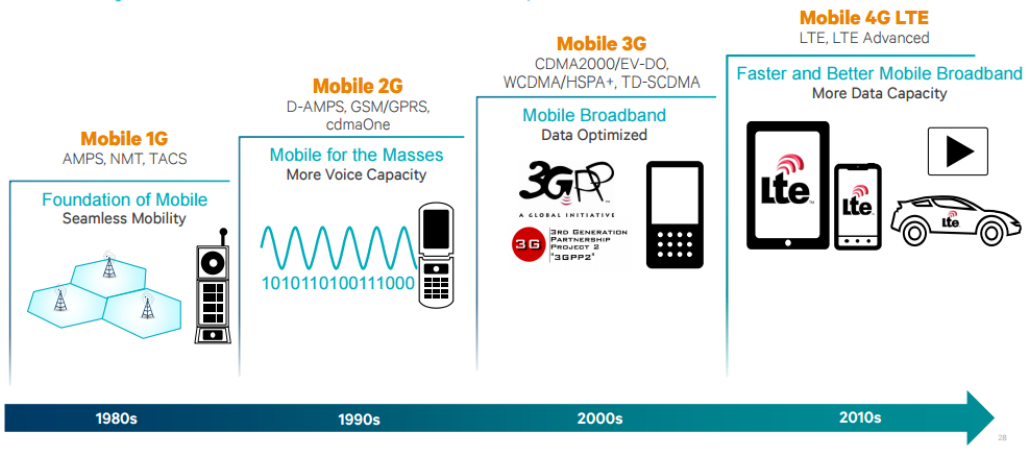


Figure 2.6 - Mobile technology evolution [5]

In 2020 it is expected that **5G** mobile technology will be fully operational in a connected society. Different research groups standards like Mobile and Wireless Communications Enablers for the Twenty–twenty Information Society (METIS), 5th Generation Non-Orthogonal Waveforms for Asynchronous Signaling (5GNOW), 5G Infrastructure Public Private Partnership (5GPPP) and many others organizations from America, China, Japan, Korea, Russia, etc around the world are actively working on different aspects and areas of 5G. Appendix A has a more detailed view of those projects and activities that are being developed all over the world.

For socio-technical requirements 5G has limits on energy dissipation, more context-related information (e.g. augmented reality), broadband Internet connectivity widely available, increased amount of remote virtual collaboration, more efficient, safer transportation means, personal data stored in the cloud and improvements on IoT (e.g. smart homes and cities) making projects to society, as shown in figure 2.7, available, more real and easy to achieve.

There are currently limitations in 4G networks already like latency, spectral efficiency and support for bursty data traffic. Several mobile applications require sending messages to servers with high data transfer rate for a very short time and nowadays this consumes a lot of battery lifetime in mobile equipments [16]. Co-channel interference due to separate channels (DL and UL) in a typical cellular network and base-station utilization efficiency are also points where improvements can be done on future networks [17].

With that said, with 5G networks the numbers of devices like smartphones, TV's, cameras, robots, video surveillance systems, high-resolution TVs, laptops and wearable devices (watches and glasses) connected is expected to continue to grow. The future networks will focus on scalability, high data rates, spectrum utilization, low latencies that will supposal increase the level of QoS and QoE with more satisfaction from users, which will be discussed in sections 2.3 and 2.4. Last but certainly not least, ubiquitous connectivity because operating bands are not identical over the globe and UE have to support a variety of radios and RATs [15, 18].

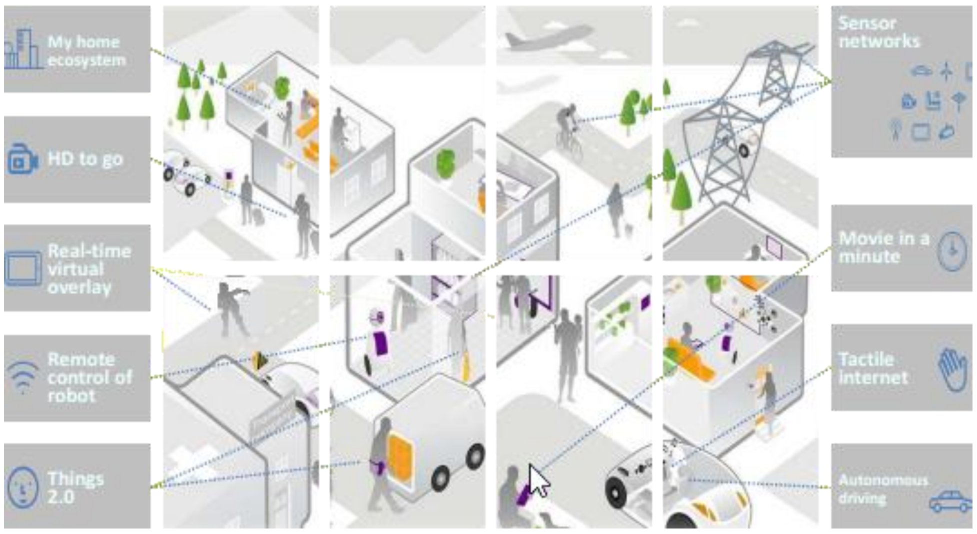


Figure 2.7 - 2020+ experience

In summary 5G networks will be more flexible with a lot of users requesting for different set of services simultaneously, having higher data rates, higher capacity, energy optimization, mobility, availability, reliability, connectivity, cost effectiveness, accuracy traffic, and less latency improving areas like IoT, gaming, personal clouds, and many areas of society with new applications for tourism, agriculture, e-government or e-health, however still facing many challenges such as security, low energy impositions breaking the uniform networking, structure management with self-healing to perform dynamic operations, handoff in very high speed vehicles and more devices connected brings more and new vulnerabilities.

Generations	Access Technology		Date Rate	Frequency Band	Bandwith	Forward Error Correction	Switching	Applications
IG	Advanced Mobile Phone Service (AMPS) (Frequency Division Multiple Access (FDMA0		2.4 kbps	800 MHz	30 KHz	NA	Circuit	Voice
2G	Global Systems tor Mobile communications (GSM) (Time Division Multiple Access (TDM At)		10 kbps	850/900/1800/1900 MHz	200 KHz	NA	Circuit	Voice + Data
	Code Division Multiple Access (CDMA)		10 kbps		1.25 MHz			
2.50	General Packet Radio Service (GPRS)		50 kbps		200 KHz		Circuit/ Packet	
	Enhanced Data Rate for OSM Evolution (EDGE)		200 kbps		200 KHz			
30	Wideband Code Division Multiple Access (WCDMA) *t* Universal Mobile Telecommunications Systems (UMTS)		3K4 kbps	800/850/900/ 1800/1900/ 2100 MHz	5 MHz	Turbo Codes	Circuit/ Packet	Voice + Data + Video calling
	Code Division Multiple Access (CDMA) 2000		384 kbps		1.25 MHz		Circuit/ Packet	
3.50	High Speed Uplink / Downlink Packet Access (HSUPA / HSDPA)		5-30 Mbps		5 Mllz		Packet	
	Evolution-Data Optimized (EVDO)		5-30 Mbps		1.25 MHz		Packet	
3.75G	Long Term Evolution (LTE) (Orthogonal / Single Carrier Frequency Division Multiple Access) (OFDM A / SC-FDMA)		100-200 Mbps	1.8GHz. 2.6GHz	1.4MHz to 20 MHz	Concatenated codes	Packet	Online gaming + High Definition television
	Worldwide Interoperability for Microwave Access (WIM AX X Scalable Orthogonal Frequency I»vision Multiple Access) SOFDM A »	Fixed WIM AX	100-200 Mbps	3.5GHz. and 5.8GHz initially	3.5MHz and 7MHz in 3.5GHz band; lOMHzin 5.8GHz bund			
4G	Long Term Evolution Advanced (LTE-A) (Orthogonal / Single Carrier Frequency Division Multiple Access) (OFDMA / SC- FDMA)		DL 3Gbjw UL l.5Gbps	1.8GHz, 2 6GHz	1.4MHz to 20 MHz	Turbo codes	Packet	Online gaming + High Definition
	Worldwide Interoperability for Microwave	Mobile WIM AX	100-200 Mbps	2.3GHz. 2.5GHz. and 3.5GHz	3.5MHz. 7MHz. 5MHz.			Television

	Access (WIM AX X Scalable Orthogonal Frequency-Division Multiple Access) SOFDMA))		initially	10MHz. and 8.75MHz initially			
5G	Beam Division Multiple Access (BDMA) and Non- and quasi-orthogonal or Filter Bank multi carrier (FBMC) multiple access	10-50 Gbps (expected)	1.8.2.6 GHz and expected 30-300 GHz	60 GHz	Low Density Parity Check Codes tl.DPC)	Packet	Ultra High definition video + Virtual Reality applications

Table 2.1 - Evolution of wireless technologies [25]

2.2 VoIP

Before explaining what VoIP is, it is important to give a brief overview of the Internet Protocol procedures. When a user establishes an Internet connection, most of the times the ISP (Internet Service Provider) assigns a public IP address to the computer, which uniquely identifies the network. IP is a protocol that allows the sending of the information, in the form of small packets, from one personal computer (PC) to another, through the Internet.

Voice over IP allows the user to establish telephone calls over a data network such as the Internet, converting an analog voice signal into a set of digital signals in the form of Packets with IP addressing.

The quality of service perceived by the user depends not only, but primarily on three aspects: Network topology, interconnection congestion and codecs used. Packets are not delivered instantaneously and this delay can affect the quality dependent of network topology and the number of nodes between the two points in communication. Interconnection congestion because other in curse calls already established on the network can affect the quality of the service and lastly dependency are the codecs used since voice signal compression can deteriorate or delay the communication, on the other hand, some codecs that usually require more computational processing have a corrective effect on transmission problems, minimizing the impact of packet loss.

2.2.1 VoLTE

LTE network architecture was already explained in section 2.1, Voice over Long-Term Evolution (VoLTE) is based on the IP Multimedia Subsystem (IMS) referred in the same section, with its own profiles for media and control planes well defined by GSMA. Using the IMS central network means that we are using LTE architecture with no dependency on legacy circuit-switched network to carry calls and being an IP-based data connection it makes VoLTE has three times more voice and data capacity than 3G UMTS.

Being a new technology VoLTE has a couple of challenges yet due to the multiplicity of protocols, technologies and implementation scenarios. In the situation of an international roaming, the call SIP signaling path and voice path are not essentially same because signaling could go through one or more Internet Exchanges (IPX). Also “the standards for voice facilities over LTE on 3GPP IMS are still growing" and carriers need to re-engineer their voice call network. Lastly, there is a user quality of experience challenge, in the sense that VoLTE needs to deal with the fact that a user can leave the LTE area coverage during a call and then make the call go via legacy network with CSFB technique [42].

It is needed to point out that VoLTE implementations are not yet interoperable, the technology is new and still needs development to reach an ideal level of quality. However, if we manage to overcome the challenges previously mentioned we will have calls with fast connections, HD Voice since the audio compression by VoLTE can reach the 13kbps and the 700MHz band contributes to more clean audio calls. Other benefits that can come are the possibility of operators offer to final user an extra service package containing, for example, video calls, file transfers, attachments or automatic translations. In addition, if we compare VoLTE with OTT apps like Skype the battery consumption level of the device will be lower on VoLTE service [40].

To finish this topic, there is a mobile network operator in India called Reliance Jio Infocomm Limited that lacks legacy network support of 2G/3G services and provides only wireless 4G LTE service network that offer VoLTE services to all those who have an Android 4.0 or higher compatible VoLTE device with their own SIM card, Jio SIM Card [41].

2.2.2 VoWiFi

To increase network capacity and extend their voice services, carriers are likely to use VoWiFi to extend coverage particularly indoors. VoWiFi is a solution that mobile services providers can deliver "the same set of mobile voice and messaging services they currently offer over their macro cellular network, over any Wi-Fi network" [43].

Such as VoLTE, this technology is defined by mobile industry standards organizations (3GPP and GSMA) and will allow a transparent transaction between the cellular network to any home, office or public Wi-Fi network. VoWiFi can really shine in helping with indoor mobile calls coverage. Providing good coverage in some scenarios is more complex and expensive than we think, especially in lower floors or in internal rooms.

Apart from extending reach, VoWifi can potential save millions of dollars to some operators as calls placed on a smartphone would be carried over the consumer's broadband network, enables traffic to be off-loaded to another network and freeing up some cellular capacity [44]. The control of subscribers is still completely retained by operator when a VoWiFi-enabled subscriber connects to Wi-Fi, they attempt to connect to operator's core network and register to receive the service, once authorized, traffic is routed over the Wi-Fi/Internet connection instead of the macro cellular network continuing the operator to handle all billing, charging and routing.

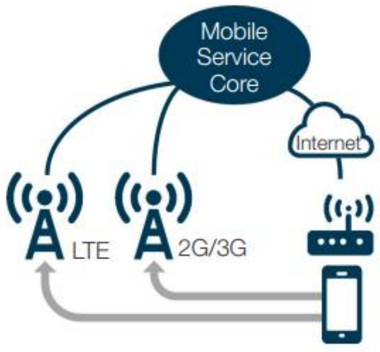


Figure 2.8 - Calls over Wi-Fi and cellular network

VoWiFi service quality is expected to be as good as voice quality over the cellular network, since the connections are over the internet and typically the Wi-Fi networks at home and office have good throughput, congestion and coverage overall, however in public Wi-Fi networks those same factors are variable and can affect the VoWiFi quality. Another issue to be considered is that carriers should also bear the cost implications for incorporating emergency service support to ensure that these calls go through reaching the destination [43].

Still, there is a cost and need to deploy an IP multimedia subsystem, but if operators already have VoLTE this already have been paid for as the same IMS core network supports both services and there is no doubt that VoWiFi will make OTT apps like Skype, Viber or WhatsUp be less relevant for user as long as they have Wi-Fi indoor coverage which is a big win for operators.

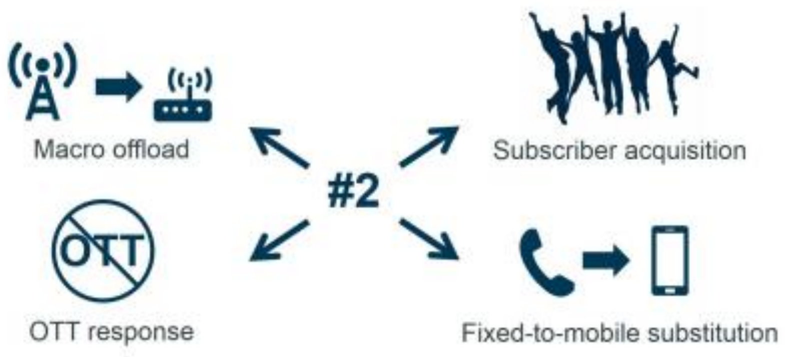


Figure 2.9 - VoWi deployment reasons

2.3 Quality of Service

The quality of Service (QoS) for networks is a set of mechanisms, standards and protocols for ensuring high-quality performance and granting end-to-end consistent treatment of data flow as they travel through the network. Taking this into account, the network can then be managed using the existing resources efficiently to ensure a QoS level and improve the final user global experience using the service.

Treating all the network traffic equally leads to no guarantees for reliability, delay and other performance parameters that will be mentioned after, with a more detailed approach. For some services or applications, the QoS is an important requirement and can even be critical if some traffic needs to have preferential treatment.

For the operator measure QoS is extremely important to grant levels of quality in services that they offer. As such, they have all the interests on measure quality parameters in similar scenarios experienced by user to improve, service availability and performance.

QoS provision in 4G networks is a challenge because the network "supports varying bit rates from multiple users and a variety of applications, hostile channel characteristics, bandwidth allocation, fault-tolerance levels, and frequent handoff among heterogeneous wireless networks" [19].

QoS support can occur at the transport, application, network, user and switching levels and to evaluate QoS performance there are some metrics can be used, such as, latency (delay in data transmission from source to destination), jitter (delay variation), bandwidth (the rate at which traffic is carried by the network), throughput and reliability or packet loss rate. Resource reservation and service prioritization can help improve these parameters, furthermore, we can apply QoS according to per flow (individual, unidirectional streams) or per aggregate (two or more flows having something in common) basis.

In section 2.1, it was already seen that improve QoS in 5G Networks is a must. Based on the projections the number of machine to machine (M2M) connections on the networks of mobile operators will surpass 20 billion [39], two times more than the present number and in 2022 mobile operators will have more than 26 billion machine to machine connections.

Some general trends related to 5G can be explained in terms of machine to machine traffic and number of machine to machine connections in mobile [36]. With these huge numbers at sight the

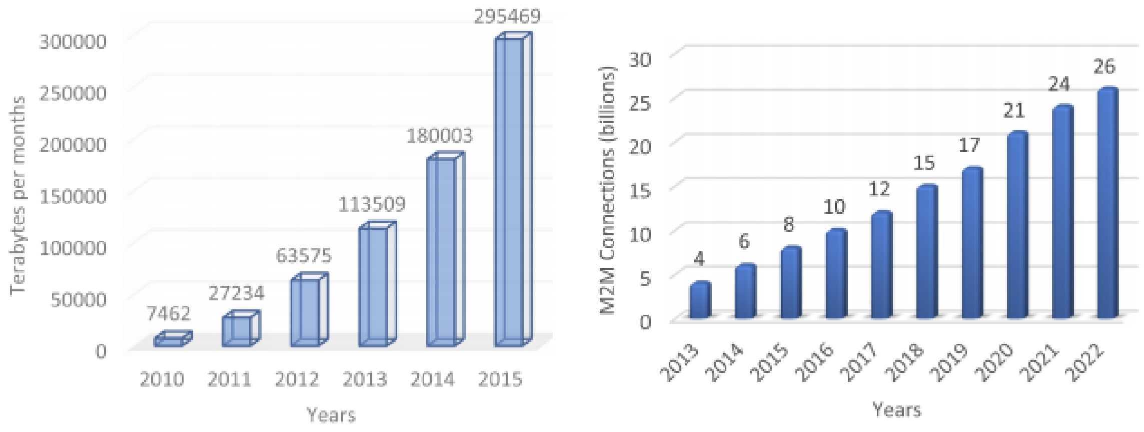


Figure 2.10 - Machine to machine traffic and connections number respectively [36]

network source data rate, delay bound, cell spectral efficiency and latency in 5G needs to be improved and that is supposed to be attained using non-orthogonal access methods in radio access networks [38].

The 5G networks are supposed to satisfy the highest level of QoS, a big issue about this topic is the tactile Internet as it requires the best QoS, especially latency at the order of 1 millisecond for senses such as touching, seeing, and hearing objects far away, as precise as human perception. However, the currently proposed architectures do not support efficient tactile Internet services. In future, it would be a promising area as to encode senses, exchange data satisfying the zero latency, and enable the user to receive the sensation [37]. Lastly, for cloud, it is proposed the deployment of a quality management element (QME) in the cloud for monitoring inter-UEs and inter-layer (the control and the data layers in C-RANs) QoS.

2.4 Quality of Experience

User's satisfaction is the ultimate target for a service provider achieves success. QoE describes the user's perception of a service and how well the service fulfills the user’s expectations. Defined by [49] is “a user-centric metric that captures the overall acceptability of the service and includes the end-to-end factors”.

Contrary to previously mentioned QoS this is a more “subjective” concept as it doesn’t have only specific metrics like latency or throughput and is dimensioned or evaluated many times with terms like "Good", "Fair", "Poor". Nevertheless, this perception is composed by subjective factors and overall experience, QoS differ from QoE because doesn’t really capture the actual experience of the user and that can dictate the use or not of an entire system or service [50].

With all of this in consideration some sources consider QoE as an extension of QoS as it try to capture a service performance with metrics that could be directly communicated by user, for example, the playback start time of a task, or in video streaming the number of interruptions on watching a video, the duration of those interruptions and how that affects the user or the user engagement with that specific video can contribute to a good or not quality of experience.

One way to try evaluating QoE in general with those more subjective factors is the use of Mean Opinion Score (MOS) classification. Users use the system/service and then rate it on a five-point discrete scale: 1-bad, 2-poor, 3-fair, 4-good, and 5-excellent. It is not easy to automate this process since many times the parameters to have in consideration are too many from user's age, genre to device screen size, device capabilities or internet connection [51]. Also, there are recent studies [52] that indicate that user’s psychological state, being this on pressured situation rather than relaxed situations can affect QoE perception of a service.

2.4.1 Audio Quality

The phone call quality is an essential feature to satisfy service subscribers, but are they satisfied with the quality of this voice service? Regardless of the more sophisticated and powerful networks and smartphones “none of the 100 plus cell phones in Consumer Reports' got an excellent or even a very good rating, for voice quality [45].

This can be explained by many reasons: smartphone size with cameras, radios, microprocessors and other hardware that allow those tiny devices do all the great stuff they do. Perhaps, sometimes the “speaker is wedged between the bezel and the front-facing camera” covered in plastic and microphone is placed on the back of the phone which is not optimal. Additionally, the reasons why voice calls aren’t perfect are not only due to devices design architecture, the closest base station can be loaded at the time of the day that the call is being done or can be dependent on the distance that the packets must travel till they reach its destination because signal has to jump from cell to cell and every building, mountain or even the weather it’s an obstacle making possible to have more than one path to reach the destination. Deciphering this multipath signals that reach to smartphones at different times is not easy having the signal sometimes to be retransmitted happening all of this in fractions of a second. Moreover, although a call can be done almost everywhere if we have coverage, some places are often very noisy or with a lot of background noise like traffic or nature sounds and neither noise-canceling technology that some phones have can make miracles on those situations [47, 48].

Due to all those experiences that we already had with voice calls, alternatives like texting, e-mail and instant messaging are constantly growing at the moment, but there are promises of improvements with HD voice transmitting at a range of frequencies from 50Hz to 7kHz that are more close to human voice frequencies (75Hz-14kHz) than the oldest telephone network that had this ranges between 300 and 3,400 Hz, perhaps we have to wait for the network backbone to be upgraded because the circuit-switched backbone still uses standard voice technology.

Thus, even though most of the new phones already support HD Voice, there is no notice yet on voice quality changes because carriers still have to activate AMR-WB codec that extends the frequency range of audio signals and deliver voice traffic over IP compressing and digitalizing voice signals and send it as data packets. VoLTE technology is the bet for fix this problem and boosts the cellular quality voice as it already supports AMR-WB, so it’s not very difficult for carriers like AT&T and Verizon to enable HD Voice once calls have moved over to its LTE network. Backbone networks, local broadband links, IMS infrastructure and other issues like ensure priority for different traffic over the LTE network are being studied and developed and it will not take much time to “let VoLTE packets flow seamlessly between mobile handsets and other IP phones” [46, 48].

2.4.2 Comparing Video Quality

Comparing quality of videos is not so simply because we cannot only compare video resolution, in fact, that could be really misleading because a 1080p movie rip at 700MB size might look worse than a 720p rip at 700MB, this is due to the fact that for the former the bitrate is too low and it affects the compression. The same principle can be applied for comparing bitrates at similar frame sizes because the encoders can deliver better quality at less or higher bitrate, for example, a 720p 700MB rip produced with XviD will look worse than a 700MB rip produced with x264, because the latter is much more efficient.

To evaluate video quality, exist a variety of quality metrics, but they can be divided into two big groups, no-reference metrics and full-reference metrics. The first ones simply have a video and output a quality score. Most of the times there is not access to the original video to make the comparison, but if that's the case full-reference metrics can be used to estimate the quality loss, obviously, this takes a longer time to compute.

In fact, to try defining a MOS composed of individual quality factors we had to consider at least video length, the viewing audience, original frame size, "quality" before the encoding and the type of video (cartoons, movies, news, etc.), not to mention that some people watch a movie sometimes for its content and they probably don't care about the quality so much, as long as the movie is funny or entertaining.

In Android case, there are video encoding recommendations that Android media framework supports for playback in the H.264 Baseline Profile and VP8 media codec.

	SD (Low quality)	SD (High quality)	HD 720p (N/A on all devices)
Video resolution	176 x 144 px	480 x 360 px	1280 x 720 px
Video frame rate	12 fps	30 fps	30 fps
Video bitrate	56 Kbps	500 Kbps	2 Mbps
Audio codec	AAC-LC	AAC-LC	AAC-LC
Audio channels	1 (mono)	2 (stereo)	2 (stereo)
Audio bitrate	24 Kbps	128 Kbps	192 Kbps

Table 2.2 - Examples of supported video encoding parameters for the H.264 codec.

	SD (Low quality)	SD (High quality)	HD 720p (N/A on all devices)	HD 1080p (N/A on all devices)
Video resolution	320 x 180 px	640 x 360 px	1280 x 720 px	1920 x 1080 px
Video frame rate	30 fps	30 fps	30 fps	30 fps
Video bitrate	800 Kbps	2 Mbps	4 Mbps	10 Mbps

Tabel 2.3 - Examples of supported video encoding parameters for the VP8 codec.

It is also possible to retrieve a variety of parameters from CamcorderProfile Android class, available since API level 8, such as video codec format, bit rate, frames per second, audio codec format, etc.

2.5 Mobile Operating Systems

A mobile operating system or mobile OS is the software that manages the resources and memory that other programs use, on top of it. Its development permits us now create and build specific applications with advanced functions and capabilities. The most popular mobile OS at the moment are Apple iOS, Google Android and Windows Phone OS.

Different operating systems bring different navigation controls, UI interfaces, features, productivity, security and privacy and could even have different support on peripheral support, for example. Said that, on buying a new mobile device is obviously very important to look at these distinct aspects and see what matters most.

Nowadays, others OS's besides those already mentioned have less than 1% of the market share worldwide, therefore they will not be focused on the next comparisons [27]. In terms of affordability it is well known that Apple doesn't make budget devices and with hardware manufacturers like Samsung, ZTE, LG, Lenovo and Huawei abandoning the Windows phones, Android wins this battle with the best choice for budget-conscious devices that have similar specifications in comparison to rivals at lower prices.

In terms of interface, every OS have its advantages, all different and unique with its owns animations and transitions, having experience with one of them could lead to a familiarity time when a transition to other OS happens. When talking about apps, Windows mobile is not even close of the iOS or Android market having less than 1milion apps on Windows Store. Android's market share continues to grow and has a higher percentage of free apps, however, iOS has been the most lucrative platform for developers with an excellent recommendation service and beats Android with more sales offer on apps and games.

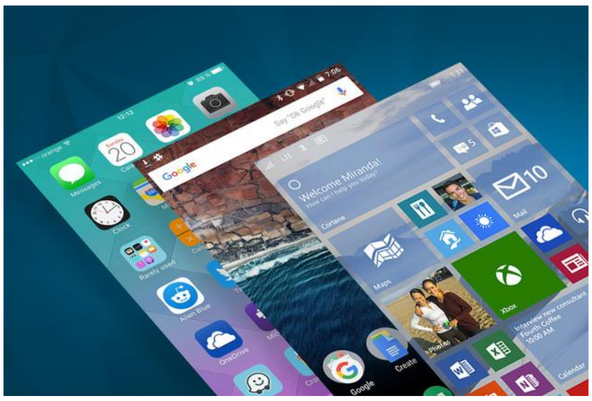


Figure 2.11 - User interface on different mobile OS's [28]

Other important and very relevant factor is the battery life and management that is difficult to compare in different OS because there is not common hardware, although can be said that "iOS optimized to squeeze the most out of the battery per mAh rating" but Apple's latest flagships are a bit disappointing and if we consider the time that a device needs to charge a "Galaxy S7 Edge can charge to 50 percent in 30minutes" which outlast almost every newer iPhone launched recently [28].

At OS updates iOS have a little advantage over Android correcting bug fixes more frequently because Microsoft and Apple have a bigger control over their software overall making easier for them to roll out updates. Even so, IoS offers the more consistent and timely software updates, for example, Widows Phone 7 devices can't be updated to 8, and the update from 8.1 to Windows 10 took a long time to actually happen. Despite the fragmentation with older devices the support of iOS software are 4-5 years which is almost the double that in Nexus phone usually left behind at 2-3 years after.

On the other hand, Android has a better space for user customization overall with alternative launchers, multiple home screens backgrounds, resizable widgets, shortcuts in comparison with iOS and Windows Mobile that has more limited offers.

Lastly and probably the main reasons this dissertation will focus from now on Android devices are the jailbreaking, bootloaders and the market share. This topic will be discussed in detail in the next subsection, but in comparison to Microsoft and mostly Apple, many Android OEMs offer a way to unlock their bootloaders. This type of control makes possible to do things that otherwise wouldn't be possible to do, upon which will be referred in the next chapters. Furthermore, Android dominates the market share hitting a record of 88% in the fourth quarter of 2016 [54].

Period	Android	iOS	Windows Phone	Others
2015Q4	79,6%	18,7%	1,2%	0,5%
2016Q1	83,5%	15,4%	0,8%	0,4%
2016Q2	87,6%	11,7%	0,4%	0,3%
2016Q3	86,8%	12,5%	0,3%	0,4%

Table 2.4 - Worldwide smartphone OS market share

2.5.1 Android

Android is a Linux-based Operating System for mobile devices, it was developed by Open Handset Alliance but is owned by Google nowadays. This system is designed for touchscreen devices so the primarily market target are smartphones, tablets, Android TV and some gadgets that run Android, for example, watches (Android Wear).

Everything started on Android Inc. a company founded in October 2003 to develop "Smarter mobile devices that are more aware of its owner's locations and preferences". The first intentions were to develop a system for digital cameras, but when they realized that the market was not big enough the attention and effort turned to produce a mobile operating system to rival Symbian and Windows Mobile at the time [26].

Google bought Android Inc. in 17th August 2005 developing a mobile operating system and making partnerships with mobile operators, software and hardware companies reaffirming that it would be open to mutual cooperation. Said that, the first commercially available smartphone running Android was the HTC Dream, released on October 22, 2008.

To show their latest updates and new features on Android versions Google usually launches a new Nexus device as a pilot, for example, the Nexus 5, made by LG or Nexus 7, made by Asus containing these upgrades and errors/bugs fixes. At each major update, the system's codename is changed, in alphabetical order, being always candy names.

With Wi-Fi and Bluetooth support Android usually comes with a bunch of Google apps like Maps with Street View, Contacts, Calendar, Google Sync, Google Search, Google, a YouTube video player, a Voice Dialer, instant messaging, MMS and the option of user customize almost everything in its UI. Regular updates and upgrades over the years made this OS become more powerful, redesigned and with an increased overall performance.

The most recent version of Android, launched at 2016 during Google I/O, **Android 7.0 Nougat**, that now it's in version 7.1.2 (April 4, 2017) has the incredible capacity in divide the screen in multiple, which is awesome for the multitasking user experience. It brings also native encryption, new emojis, change interface and font size, virtual reality support, the ability of send messages through the notification bar and more to explore [29].

Version	Codename	API	Distribution
2.2	Froyo	8	0,1%
2.3.3 - 2.3.7	Gingerbread	10	1,7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1,6%
4.1.x	Jelly Bean	16	6,0%
4.2.x		17	8,3%
4.3		18	2,4%
4.4	KitKat	19	29,2%
5.0	Lollipop	21	14,1%
5.1		22	21,4%
6.0	Marshmallow	23	15,2%

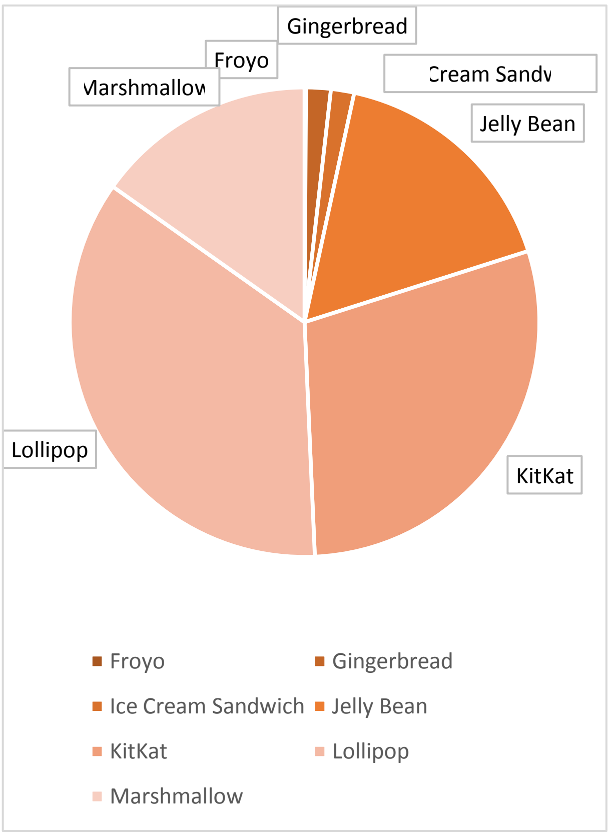


Figure 2.12 - Number of devices per Android version on 1 August 2016 [31]

2.5.2 Root/Jailbreak

So, Android phones use Linux permissions and the file system as was mentioned on the previously section. Therefore, different processes and users can have different privilege levels. These privileges determine what the process is allowed or prohibited from doing. All new apps that are installed have a type of user ID, and certain permissions to do certain things.

Root is nothing more than a user, but he has permissions “to do anything to any file any place in the system” [32], consequently root brings an incredible power to user since he can overcoming limitations that carriers and hardware manufacturers put on the device like uninstall pre-installed applications, speed-up the phone with overclocking or modify any file as he want, perhaps "with great power comes great responsibility" because changes you or apps that have root access do in the system file could lead to security risks if you aren’t sure of what you doing [34].

The process of allowing Android devices to have privileged control (root access) is called rooting and it gives similar access to administrative permissions to *superuser* on Linux. This process is legal if you are the owner of the device. However, If you use it to install spyware on other people's phones, that’s a crime in certain countries around the world. Other than this, many vendors such as HTC, Sony, Asus or Google provide the ability to unlock devices or even replace whole operating system, but keep in mind that on doing this you are “changing everything about the inherent security from Google and the people who built it” [32] and you will almost certainly void the warranty of your phone. Continuing with the topic if you need your warranty service for some hardware issue after is it possible to unroot a phone? Yes, you can often find instructions for unroot your phone, usually it involves flash an SBF(system binary file) or an RUU, however, some phones have a digital "switch" that flips when you unlock your device that is very difficult to revert, so remember to do some research before unlocking if you need your warranty [33].

The steps for rooting an Android device are in general unlocking or bypassing any bootloader protection to allow the system partition to be written and installing the relevant binaries to acquire root. Both of these steps vary from phone to phone, install a custom recovery, temporarily boot from an image over USB or flash a new bootloader are common procedures that can help advance the process. Other devices can be unlocked by accessing *fastboot* with a push-button combination. Many manufacturers now offer an official route to unlocking the bootloader.

If you root your phone without flashing a custom ROM, you will likely still get OTA updates from your carrier, perhaps, they will likely break your root. Fortunately, there are some free apps that can help you keep your root access. Otherwise, if you flash a custom ROM, you will not get OTA updates from your carrier [33].

2.5.3 System apps vs. Non-System app

Searching for an application on Google Play Store and click install or loading an APK to the device always install the app as data or user apps. However, there are some scenarios where there is a need to install an application as system app to be able to do certain actions such as programmatically change or add an APN to the device among other features that Android only permits to system app. This is due to certain permissions are protected under the "*signatureOrSystem*" protection level. Such permissions are not available to every application because they will grant risky privileges such as control over other applications, background installation and un-installation, among others. Such permissions can be utilized for malicious purposes, therefore Android will only grant them for system applications or ordinary applications signed by platform signatures [36].

A common mistake believed by many is that a system application is an application which is signed by the OS's platform signatures. A System application is merely an application which is placed under /system/app folder on an Android device. An application can only be installed in that folder with access to the OS's ROM (system.img). The application is placed under /app folder after the ROM has been extracted. A device which loads the custom ROM will have the new System application added. The benefit of a system application is that the application cannot be removed from the device (cannot be uninstalled by a non-root user) because /system/app is a read-only folder [35].

A non-system application is an ordinary application which will be installed under /data/app folder, and which is read-write. It is possible to check if an application is a system application or not, using "*ApplicationInfo.FLAG_SYSTEM"*. If the constant returns true, then the application in question is a System application.

2.6 Existing Solutions

Now that's been presented the motivations and objectives of the present dissertation it's time to present what similar solutions are available on the market and what they do. ArQoS Pocket solution and its features will be presented and described in detail in the next chapter, ArQoS Pocket solution, but very briefly we are talking about an Android app that measures and tests Wi-Fi and mobile network in the various technologies and communicates with a backend that gathers and process all the information received in order to present an overall or more detailed state of the operator's network.

There are hundreds of applications in the market testing the mobile network or the Wi-Fi network or retrieving multiple information about the device, but only a couple of solutions do all that in the same application and still have a strong test components for Voice or OTT apps, for example.

For possible radio information it is possible to retrieve the cell signal strength, cell id, operator's name, technology, if the roaming is active or not and many others parameters that can vary with the mobile network technology that is being used at that moment. For Wi-Fi network parameters, BSSID, SSID (network name), IP, MAC, primary and secondary DNS's addresses, network mask, etc are examples of possible retrieval parameters. Information about the device and its specifications such as location, IMEI, battery level or free memory available in the device will be relevant for some task that will be referred in chapter 3 and that's possible to retrieve as well from Android internal classes. Many of the apps that do some of these tasks have a free lite version available for users to test in market retrieving these and more parameters from both networks but, usually they have a more detailed and paid version as well, exploring other features and with a more complex set of network tests, as is the case, of the PRTG Network Monitor app, for example.

From the QoE optic, a highlight for *Xperience* app from Streambow, where user could input bad coverage zones, drop calls, bad quality audio or no data in order to give feedback to help the operator understand faster where are the points that need some improvement. Were also founded scenarios where solutions focused at driving tests and vehicle installation, passively testing and taking snapshots of the network through the city. Scheduling tests to a specific date and hour were also possible in some concurrent solutions and we also ended up implementing this feature in our solution, as well.

The tests implemented in our solution will be detailed in chapter 4. At market, the network tests founded vary from simple PING's, HTTP Download/Upload, send an SMS/MMS or access an e-mail server to, in some more expensive solutions encountered, voice tests using PESQ and POLQA, OTT applications like Skype or YouTube, video streaming quality and tests to other applications in vogue like Facebook, YouTube and Dropbox in the various technologies (EDGE, GSM, HSPA+, LTE, …). Still on these more complex solutions, already being developed for several years, the support for VoLTE mentioned in subsection 2.2.1 were introduced and for clients interested on this business, the deployment, training, support and stabilization were offered by a couple of years in solutions developed by big companies like Rohde-Schwarz or Focus infocom.

Were also found solutions like Xcal-Ranger from Accuver with a setup where the devices were all in a suitcase ready to be placed at critical points on the network or to be taken through the city analyzing the network in drive tests as was mentioned before. These types of solutions communicated with a backed sending all the information and results to the last and could be controlled remotely, for example, executing tests scheduled for a certain time in a backend interface solution.

Figure 2.13 - Existing concurrent solutions

The table 2.5 shows a comparison between the different solutions encountered. Reinforcing the idea and the mobile operating system's market share presented in subsection 2.5 only three of the sixteen solutions studied for this dissertation had a IoS version and the ones who had still had a Android version as well. The remaining solutions were developed exclusively for Android OS. The POLQA (Perceptual Objective Listening Quality Assessment) column it's related to voice tests and the goal is to predict speech quality by digital speech signal analysis.

Not all the solutions were tested because most of them were paid, some of them well paid. We want to give a special highlight the QualityProc android Swissqual from Rohde-Schwarz which seemed to be one of the most complete solutions with a suitcase for multiple devices that could be remotely controlled. Supporting multiple tests for voice, video, data and messaging, beside the many also performed by other solutions. Testing voice quality with POLQA, PESQ or Squad08 algorithms and already giving support for devices with VoLTE capacities. Additionally it has video streaming quality analysis and data information about OTT apps like Dropbox or Facebook. Lastly, it has more than 200 KPI's sent continuously to a backend application and full recording and decoding of protocol layers on the supported technologies: 3GPP, L2, L3, TCP/IP, IMS, and SIP.

Product's name	Android	IoS	POLQA	Device Information	VoLTE	Wifi Metrics	Video/ Apps	Test's Scheduler	Error Recuperation	Base Tests
PRTG	x			x		x			x	x
Xperience (Streambow)	x			x		x		x	x	x
Xcal-Ranger (accuver)	x		x		x		x	x	x	x
Nemo Handy (annite)			x							x
TEMS (Ascom)	x		x	x				x		x
Amanzitel – Customer IQ	x	x	x	x		x				
XGMA SP (FocusInfocom)	x		x	x	x		x			x
Keynote (SIGOS)	x	x	x		x		x			x
The TrueSite Solution (VIAVI)	x				x	x		x	x	x
Sigma -ML (Meritech)	x		x	x				x	x	x
FalconProbe	x			x		x	x		x	x
QualityProc android Swissqual	x		x	x	x	x	x	x	x	x
SQlive (Telchemy)	x					x	x			x
Gemalto	x	x				x	x	x	x	x
GL communications	x		x	x				x	x	x
Spirent Epitiro	x			x		x	x		x	x

* Base Tests = HTTP, FTP, SMS, PING.
**Columns not marked may only have not the information available in the product information.

Table 2.5 - Concurrent solutions: comparative table

Chapter 3

ArQoS Pocket solution

This chapter presents a solution to complement the ArQoS system referred in section 1.2. It is an Android app solution that works as a mobile probe collecting multiple data and indicators of the network.

Supporting multiple technologies on mobile networks (GSM, GPRS, UMTS, HSDPA, HSUPA, HSPA+, etc) and Wi-Fi this solution allows continuous tests to check the connectivity and availability of the network, as well as help in troubleshooting and monitor the quality of service with more intrusive tests.

The key features of the ArQoS Pocket solution are:

- **Integration** with ArQoS management system;
- **Scheduling** personalized tests;
- An alarm **failure notification;**
- Tests, anomalies and networks logs are **saved** and can be seen in a **history tab** with all the data information associated [20].

3.1 Objectives and Requirements

The presented solution is an ArQoS integrated solution that runs on Android terminals monitoring active and passively Wi-Fi and mobile networks. The objectives of this solution are testing the connectivity and the availability of operator’s networks by performing passive and intrusive tests to troubleshoot and monitor the network. Other main objectives of this software pass through collect KPIs and KQIs on a large scale to grant network and service performance and reliability. Those indicators are then sent to a backend where statistical graphics, colored and filtered maps can be created or observed in order to give an overall state of the operator’s network.

Simulating the user experience with scheduled tests plans, it is possible to provide continuous service performance monitoring and guarantee a healthy network by detecting behaviors that can jeopardize the performance and violate customer SLAs maintaining customer expectation and loyalty.

This is important not only to help operators know where the failure points are but also, to help on planning the network to cope with new business as well. Furthermore, this app will also help to improve operator's services that already exist and consequently raise the quality of experience percept by user.

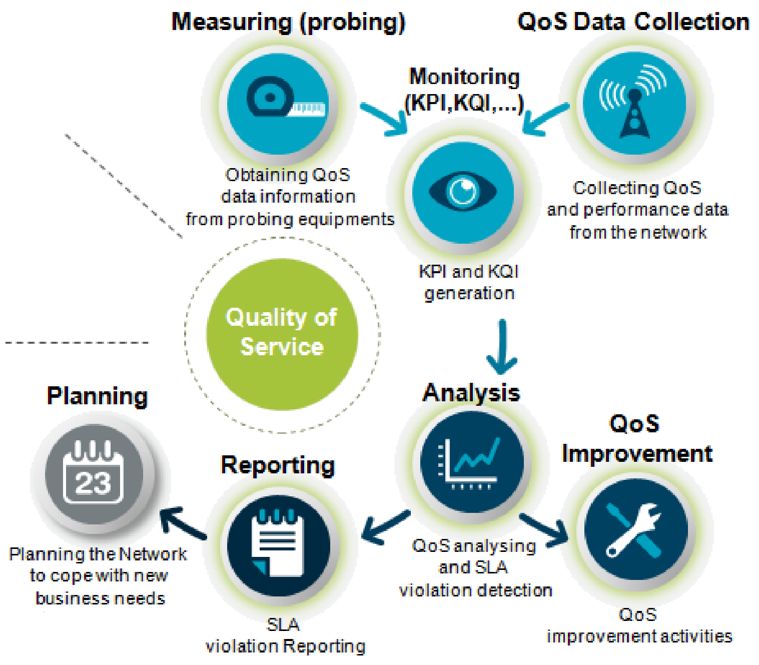

Measuring (probing)
QoS Data Collection
Monitoring (KPI,KQI,...)
Obtaining QoS data information from probing equipments
Collecting QoS and performance data from the network
KPI and KQI generation
Quality of Service
Analysis
Planning
QoS Improvement
Reporting
23
Planning the Network to cope with new business needs
QoS analysing and SLA violation detection
SLA violation Reporting
QoS improvement activities

Figure 3.1 - ArQoS Pocket objectives

ArQoS Pocket is a solution that is being created since 2011 that has passed and developed for different developers/programmers along these years, as such, the actual version already has some features implemented. Those features are: a dashboard with summary mobile network and Wi-Fi real-time information, an anomaly page that allows users to input failures detected on the network, being these associated with geographic information and stored locally and lastly a tests page where user can test the network. Tests already implemented are PING's to a specific website and HTTP tests (download/upload) that allow the app to do, for example, browsing or/and network speed tests.

Despite the passing of the years, this project has been stopped for a long time and there are still many functional requirements to be developed. The solutions still need to communicate with a backed and send anomalies, tests results and radiologs to the management system, as well as, receive configurations and scheduled tests that should run at a specific date from the same management system.

Still about the tests, improve and implement more tests, such as SMS/MMS tests, voice tests (establish a call, answer a call, hang up, reproduce and record audio) and much more are one of the main functional requirements to be developed. Those tests have must have a snapshot of the mobile and if possible Wi-Fi information attached, as well as the GPS location.

Other functional requirement is the production of radiologs in JSON format with device and mobile network information or Wi-Fi information, in the case of scanlogs. A variation of this is a snapshot, which is not more or less than a radiolog with a user commentary at the end with the description about why that snapshot was taken. In addition, there are special radiologs taken when certain events happen, these events can be a call establishment, a call drop, a handover, a roaming state change, a cell reselection, a call setup or a call release. Those radiologs must be activated by the user or remotely by the management system with a configurable interval and stored locally according to a configurable space limit.

Considering the last-mentioned requirement, there was a need to pass the local storage of those data files to a local database considering the amount of information that now needed to be stored. In addition, it is being stored Wi-Fi information equally to radiologs in a table called scanlogs. In the next chapter, will be explained how these requirements were implemented, how the database is structured, what parameters are being saved and what they mean with more detail. It should be possible for user to choose if he wants to store the results in the phone's internal memory or SD card and he should be able to consult those results in a map or list.

The results delivery to the management system is another big requirement that needs to be done from scratch. This connection has several issues associated with the probe discovery, the results delivery, the test's programming and schedule through the management system and the support for dynamic configurations.

The communication should be made based on HTTP requests with JSON content. The management system is ready to receive and handle HTTP POSTs and the application should do the same and answer through the port number 80.

The connection with the management network depends on the availability of the last, therefore there are two ways to initiate this connection: User request or automatically. In the first case, it is user's responsibility to turn on Wi-Fi and change the default APN on Android configurations, being able after to start delivering results. The “automatic” mode has an APN address configurable by the user and all the connection establishment happens automatically and with transparency for the user. In this last case, the app cleverly manages the connection delivering the results at free times where there are no tests to execute. In case of failure, the application must wait a “*backoff”* time and try the delivery again later.

A final requirement passes by having an option to automatically start the application when the device boots and boot the device automatically if it is charging, because there could be a scenario in the future where the device is analyzing the network at a specific location, with no user interaction.

With a deeper analysis of the code was considered that the project structure should be refactored, not only because the code has been developed for several years by different persons, but also to have a structured and thought Android project architecture closer that will allow to save time in the long-term for future developers. This is, therefore, part of the improve of some non-functional requirements since this refactor will make the application more scalable with the passage from local file storage to a database storage, more flexible and failure handling with the use of new frameworks that handle asynchronous tasks with separated threads running in parallel.

Moreover, all the solution developed within this dissertation tried to follow the KISS (Keep It Simple, Stupid) dividing, on the refactor, extensive classes in multiple shorter classes that only do what they should do and let other tasks to the respective entity. Beside this, user-friendly interfaces and fast responsiveness of the app were also two other focus points taking into account on the development, trying to give always feedback to user on every action performed, as well as parallelize or execute in background what could be executed with transparency for the user, not leaving him stuck waiting for the end of certain task.

3.2 Stakeholders

A project success depends a lot of the people involved on it and it’s crucial to identify the stakeholder’s expectations to satisfy them. ArQoS solution has essentially 3 parts involved, namely:

- Project team members;
    - ArQoS Pocket developers

    - ArQoS NG team
    - Management system team
    - Usability team
- Project managers;
- General customers/users.

Project team members can be applications users as well, but essentially they are the ones that have a direct influence on the project development and execution, consequentially, they have a big responsibility since are they that implement all the features and requirements of the project. In this solution, project team members can be divided into ArQoS Pocket developers, ArQoS NG team, the management system team and the usability team member.

As mentioned before ArQoS product already exists in the market and it’s working with fixed and mobile probes testing and retrieving QoS and QoE metrics from the network. These probes and the tasks they do are being improved all the time, therefore there are many features that ArQoS Pocket solution must do that is already implemented and executed on the fixed/mobile probes spread across the world. The experience and the knowledge of what is expected to happen provided by ArQoS NG team are a great help for the ArQoS Pocket developers making possible this last team progress more quickly and optimally. The management system team is composed by some developers and system administrators that work on the management and maintenance of the product backend.

The usability team contributes for the product on realizing usability tests, studying user’s interaction with the application UI and detecting possible problems that they have on using it. Focusing on behaviors and not in opinions, this team main goal is to understand how the people use the product and after an analysis, propose a solution to improve the product design and overall performance [53]. This team has contact with many frameworks and has many years of experience with users interface issues. Additionally, this team has great design skills too, making all the images and assets needed for the product, therefore also contributing to a quicker development since programmers can be focused on the code only, instead of implementing both functional features and the design at the same time.

ArQoS pocket developers are the ones that have a direct contact with the code, develop and implement all the requirements needed for the project. Led by the project managers whose function is to guide them, validate all the features implemented, create and order next features priority and dealing with the sale and communication with the final customer.

These teams have one or more project managers, need to communicate between them and are completely focused on the project success, aiming at satisfying the customer's needs and expectations. A project manager leads the project, the project team and is the one that makes the decisions when needed. He can also deal with the sale of the product, having the first contact and handling the communication with the final customers.

3.3 Use cases

To better understand the features offered by ArQoS Pocket solution and how interactions can be done with the users, in this section will be presented some use-cases examples demonstrating system reactions to contact with users.

The presented solution does some tasks in background such as receive programmed tests from the management system, collect events that happen like call drops or cell reselections and send tests results and radiologs (network information at a specific time and location) to the management system.

Further than tasks that happen in background without user perception, there are also on demand tests that user can run. He can also take network snapshots on demand too, see Wi-Fi and mobile networks details, report anomalies and change app settings: choose the time interval for radiologs registers, app language (English, Portuguese, French) or the periodicity used for sending information to the management system (daily or schedule).

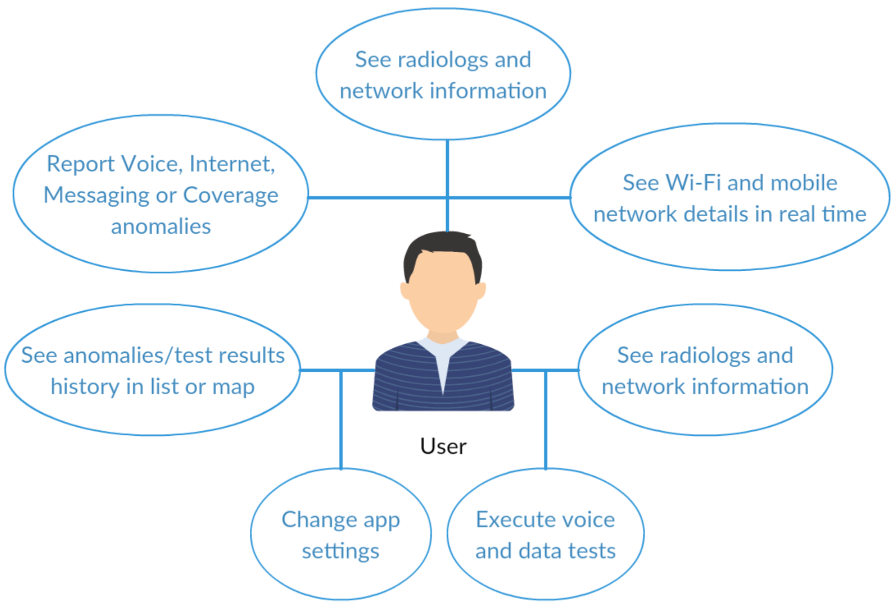

See radiologs and network information
Report Voice, Internet, Messaging or Coverage anomalies
See Wi-Fi and mobile network details in real time
See anomalies/test results history in list or map
See radiologs and network information
User
Change app settings
Execute voice and data tests

Figure 3.2 - Use case diagram: User-App interaction

All tests results, anomalies and radiologs information taken from the application running on multiple devices are gathered in a backend, making possible to system administrators have an overall overview of what is happening with its own network. This gives more knowledge to the operator and facilitates important decisions, such as prioritize the act points. Furthermore, with all these data, the operator can now filter all information by technology (2G, 3G, 4G), date or by events to see geographically where special events such as bad coverage zones or more dropped calls happen. At the management system is also possible to manage and change the configuration of all ArQoS probes (fixed, mobile and pocket) and configure scheduled tests to a certain date to troubleshoot network operations, for example.

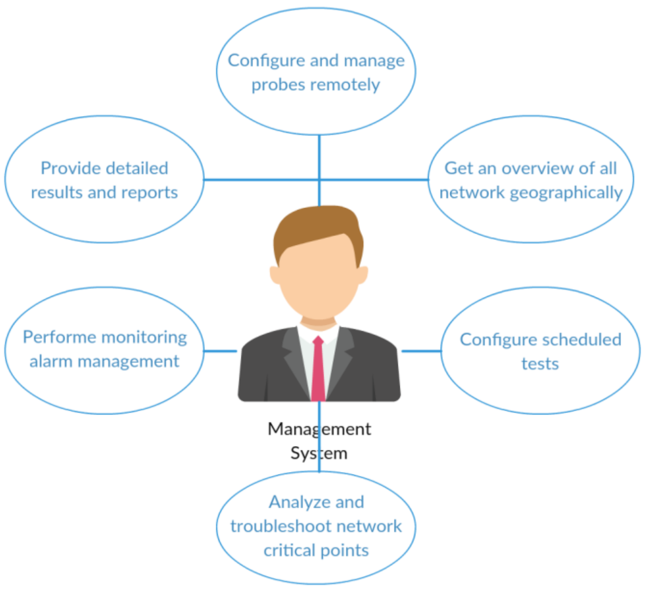

Configure and manage probes remotely
Provide detailed results and reports
Get an overview of all network geographically
Performe monitoring alarm management
Configure scheduled tests
Management System
Analyze and troubleshoot network critical points

Figure 3.3 - Use case diagram: System administrator

3.4 ArQoS Pocket Solution Architecture

ArQoS global product act all over the world measuring QoS data information from probing equipment's (mobile probes installed in mobile kits and fixed probes). After this dissertation and project, we hope that these data also come from ArQoS Pocket. All data is gathered, parsed and analyzed in management server to improve user quality of service and experience with operator's services.

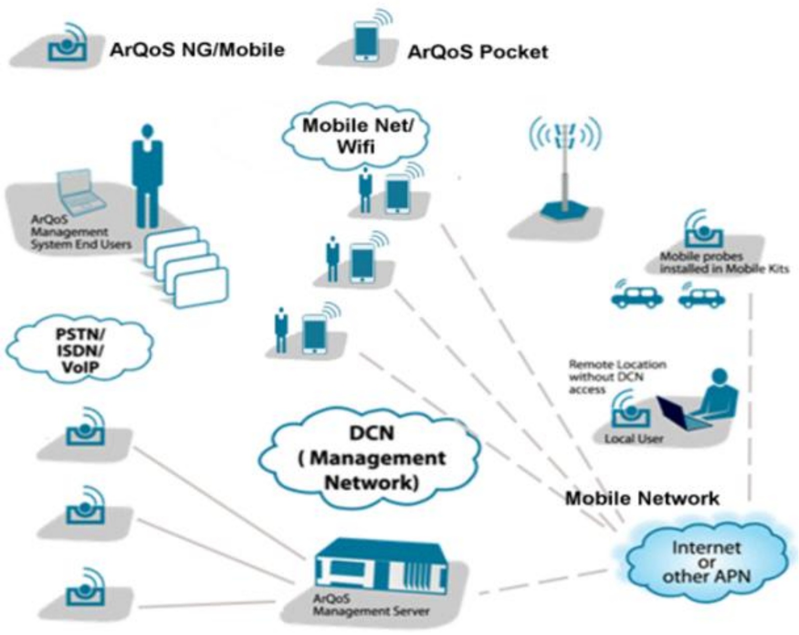

ArQoS NG/Mobile
ArQoS Pocket
Mobile Net/ Wifi
ArQoS Management System End Users
Mobile probes installed in Mobile Kits
PSTN/ ISDN/ VoIP
Remote Location without DCN access
Local User
DCN (Management Network)
Mobile Network
Internet or other APN
ArQoS Management Server

Figure 3.4 - ArQoS architecture diagram

Particularly mentioning now the ArQoS Pocket solution, the Android application is divided into 3 main packages: app, core and persistence. The first one contains all the code referent to Android activities, fragment's logic and manages all the animations and transitions in the user interface.

The core has all the code logic that is needed without affecting the UI directly, for example, retrieving mobile or Wi-Fi network parameters from Android internal classes on the various technologies and finally, the "persistence" package is related to the SQLite database that we can see in figure 3.6. This database contains all the anomalies detected on the network, radiologs (mobile network information), scanlogs (Wi-Fi network information) and test/tasks results that will be after sent to the management system via REST.

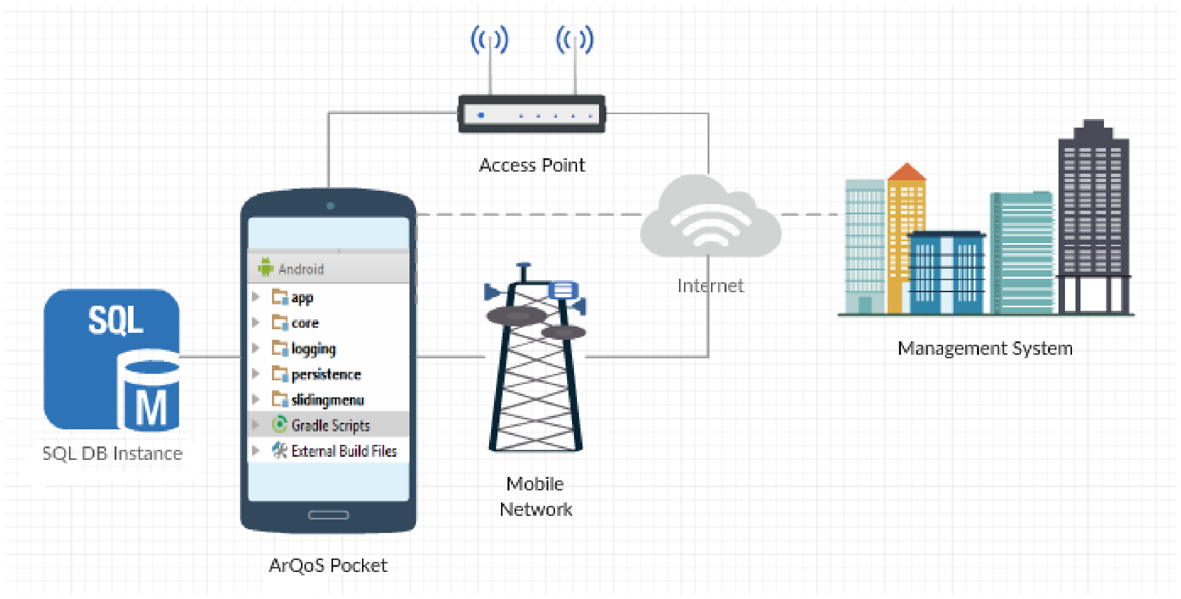


Figure 3.5 - ArQoS Pocket architecture diagram

The database is not yet fully implemented, but it has already support for tests and tasks results. In the next chapter, ArQoS Pocket implementation we will detail more this topic, but summarizing all the anomalies, tests and radiologs results were being saved in files at internal device memory and now the application is upgrading to a database allowing that the app run not only on-demand tests but scheduled tests as well with features which previously did not exist, like updating or cancelling a scheduled test.

This SQL database is currently in development, as well, but explaining what is implemented, for now, there are six test types: “Date”, “Boot”, “Iterations”, “Time interval”, “Week” and “User request”. The application supports both “Date” and “User request” tests, at the moment.

Date tests are scheduled tests to a certain date that must be in the future to be valid, otherwise, it will be ignored by the database and not executed. An example of this tests can be seen in snippet 1, along with the many parameters that can be configured and which will be explained in the next chapter. User request is a different test type where the initial and final dates of the test are ignored because it is a test to run on user demand by clicking the “play” button on the test details view.

One test can have multiple tasks associated that means that is always possible to create new tests with the same task multiple times, for example, a test sending multiple SMS’s. Beyond the parameters or columns on database seen in figure 3.6, tests can have more complexes fields such as “recursion” making him being executed repeatedly over an interval of seconds and executable if it is between the initial and final date configured a priori.

There are common parameters to all tasks and then specific parameters depending on the task being executed. After a task has executed, a result is obtained and saved into the database. Similarly to tasks, a test has a result too which may or may not be successful, but in the test case, if just one task fails for the test where it is associated, that test will be considered a failed test.

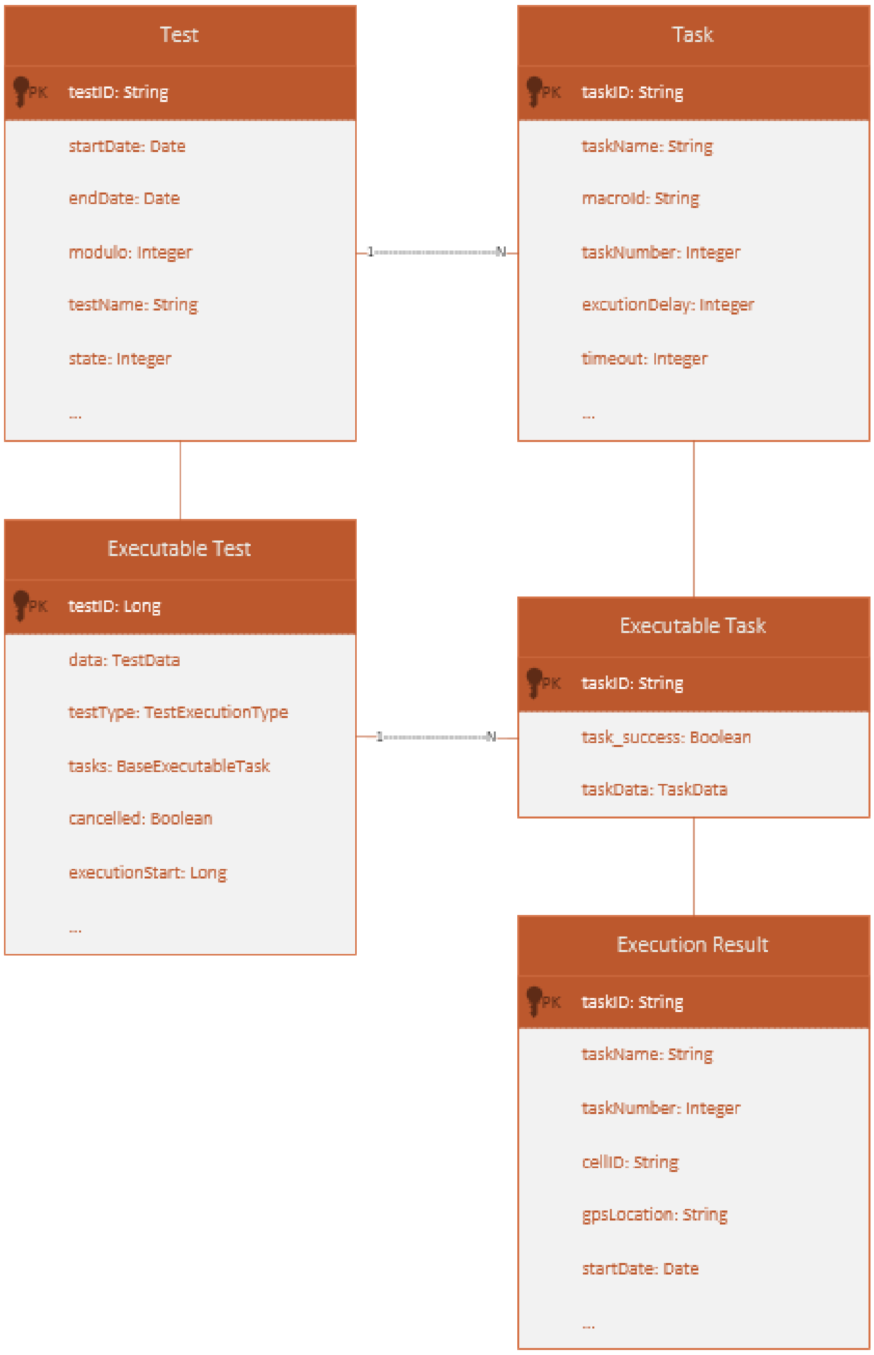


Figure 3.6 - ArQoS Pocket database schema

Chapter 4

ArQoS Pocket Implementation

In the previously chapter were described the requirements and the solution architecture in order to provide and improve the quality of service and experience to the end user. This chapter's objective is to describe all the implementation, explaining the steps taken and the reasoning behind each decision, giving a more powerful insight of the work done.

As referred before, this solution is intended to be applied in a real-world scenario used by either end users or operator technicians to troubleshot possible problems on the network. However, one more scenario was thought for this solution. Devices running the app can stay near the already existing mobile probes communicating with them using USB ports through ADB.

4.1 Application UI Overview

Independent of the scenario where the application is used the objectives continue the same, gather QoS metrics, performance data, network information and deliver that information to the management system, making possible to improve QoS and QoE to the final user.

The programming languages used were Java and C++. Java because is an Android application and C++ was used to encode/decode .wav audio files. This compressor is lossless, that means that no data quality is lost in the compression and when uncompressed, the data will be identical to the original. Working on multichannel 8, 16 and 24 bits, we will be able to compress files in a range of 30% - 70%.

It is possible to divide the application into six different main parts, Dashboard, Anomalies, Tests, Radiologs/Snapshots, Settings and the About section as we can see in figure 4.1.

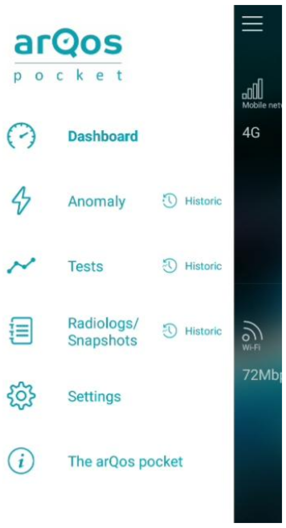


Figure 4.1 - ArQoS Pocket UI: Sliding Menu

4.1.1 Design options and techniques

In this subsection, will be presented some design techniques used to improve the user experience on using our solution, as well as patterns and templates used to provide a consistent user interaction.

App opening starts with a splash screen that can be seen below in figure 4.2, this allows a background service that initiates at the app start to load the app content and thus fill in all the information correctly before user reaches the application homepage. Detailed technical information about this and other services running at background will be explained in the next section, technical implementation.

Figure 4.2 - ArQoS Pocket UI: Splash screen

There is a lot of Android material design used, but there are a lot of customized buttons, textviews, checkboxes and dropdowns, as well. In the Settings page was changed the standard checkboxes to round ones due to the interaction do not be as an expected interaction with simple checkboxes. In the case of the homepage page displayed after the splash screen, only one homepage can be selected at the time. The settings page with this homepage's group can be seen in figure 4.10.

It was used a framework for the slide animations called "androidslidingmenu". Every time that is possible a user slide something, for example, to see more details about it, it is indicated in the app with a "plus" symbol or similar symbols placed on the right or left borders of the application UI. Examples of this interaction can be seen in the figures 4.4, 4.6, 4.8 and 5.9.

In order to help the user to search quickly what he intends, in history pages it is always possible to filter the information by categories. This feature is implemented either in the history list view and map view, filtering or cleaning the list or the map pins, respectively, of others than the selected category.

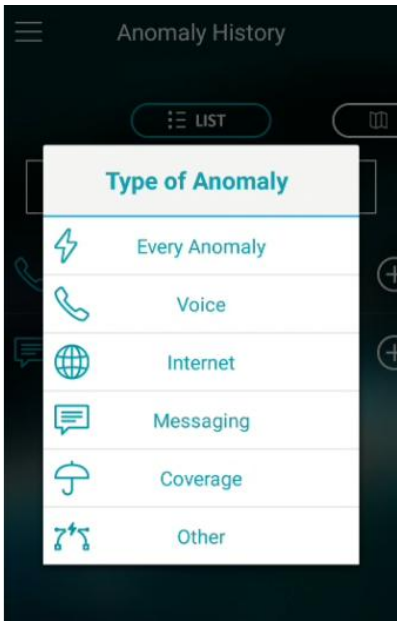
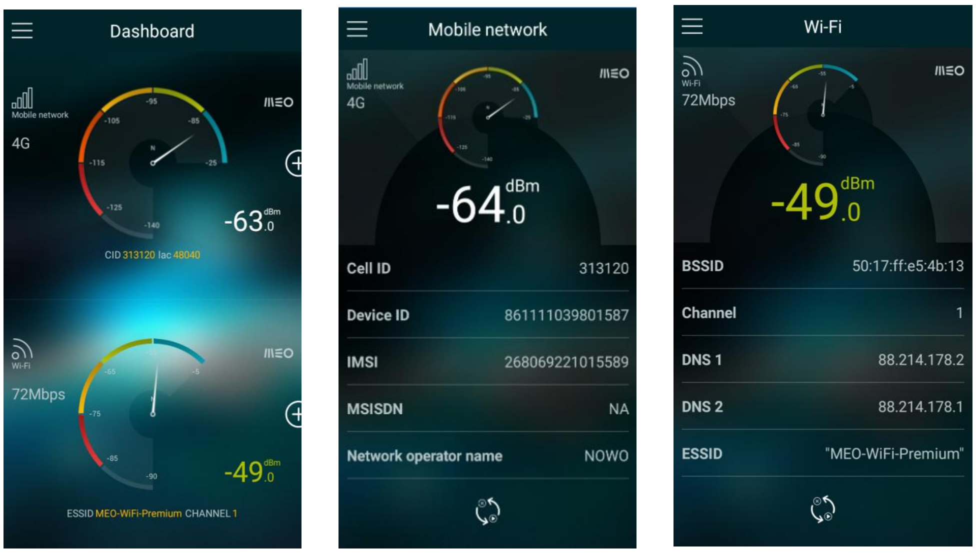


Figure 4.3 - ArQoS Pocket UI: Anomaly's filtering dropdown

4.1.2 Dashboard

On dashboard, it's possible to see mobile and Wi-Fi network information when available. In the main page, user also gets an idea of the signal strength observing both speedometers in the center. Bellow the mobile network icon it’s seen the technology of mobile network (2G, 3G, 4G) and bellow the Wi-Fi icon it’s the link speed connection in Mb/s.

Sliding in each half of the screen will lead us to a more detailed page where can be observed the Cell ID, Device ID (IMEI), IMSI, Operator name, MCC/MNC and if the roaming is or not active. On Wi-Fi side, it’s seen the AP’s BSSID, MAC address, channel number, primary and secondary DNS’s, the ESSID, Gateway, lease duration time in seconds, netmask and if the SSID is or not hidden.



CID 313120 lac 48040

ESSID MEO-WiFi-Premium CHANNEL 1

Cell ID	313120
Device ID	861111039801587
IMSI	268069221015589
MSISDN	NA
Network operator name	NOWO

BSSID	50:17:ff:e5:4b:13
Channel	1
DNS 1	88.214.178.2
DNS 2	88.214.178.1
ESSID	"MEO-WiFi-Premium"

Figure 4.4 - ArQoS Pocket UI: Dashboard

4.1.3 Anomalies

In this section, the user can report an anomaly detected on the network. As can be seen on the left of figure 4.5 anomalies can be distinguished by 5 types: voice, internet, coverage or other. Those categories have different anomalies associated, such as fall call or noise/poor audio quality call for Voice, no data access or slow access, for example, on the Internet category, bad indoor coverage for the Coverage category, etc. At the right side of figure 4.5, it’s seen a map that let the user choose the location where the anomaly was detected by dragging it. This is useful if a user detects an anomaly, but just reports it after a certain time. This map will open by default on the actual location of the user. The last step is optional and consists in attach a feedback message explaining why the anomaly is being reported.

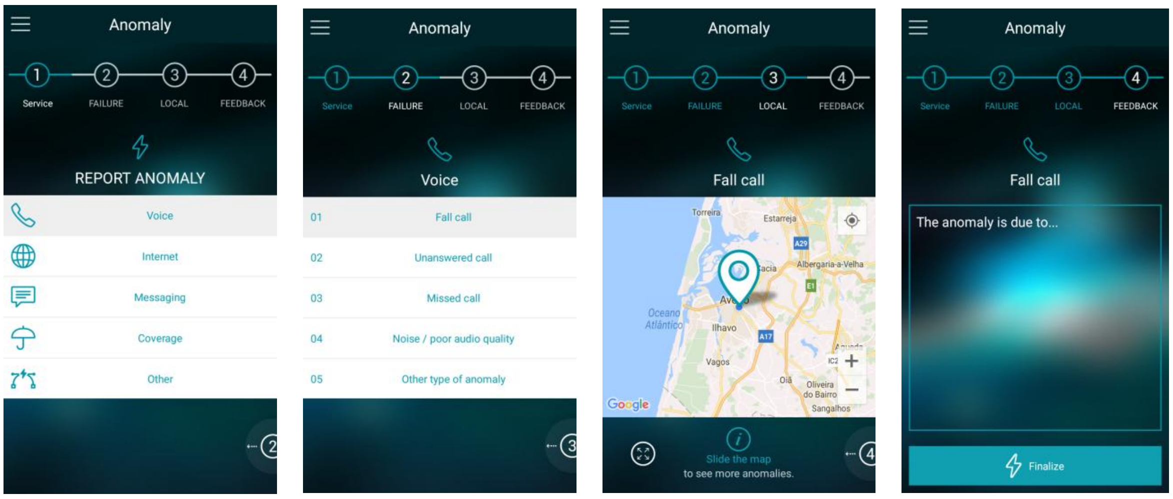


Figure 4.5 - ArQoS Pocket UI: Report an anomaly

For anomalies, tests and radiologs pages there is also a respective history page where the user can see all the inputs previously taken. This history can be observed in list or map format at left and middle of figure 4.6. It is also possible to filter information with a click on the dropdown seen at the same screens. This allows the user to find quickly, for example, an Internet related anomaly. With a tap on a specific anomaly, the app shows a detailed information about it with the feedback reported previously and information about the local with a map below containing only that specific anomaly on it. This can be seen at the right side of the figure below.

4.1.4 Tests

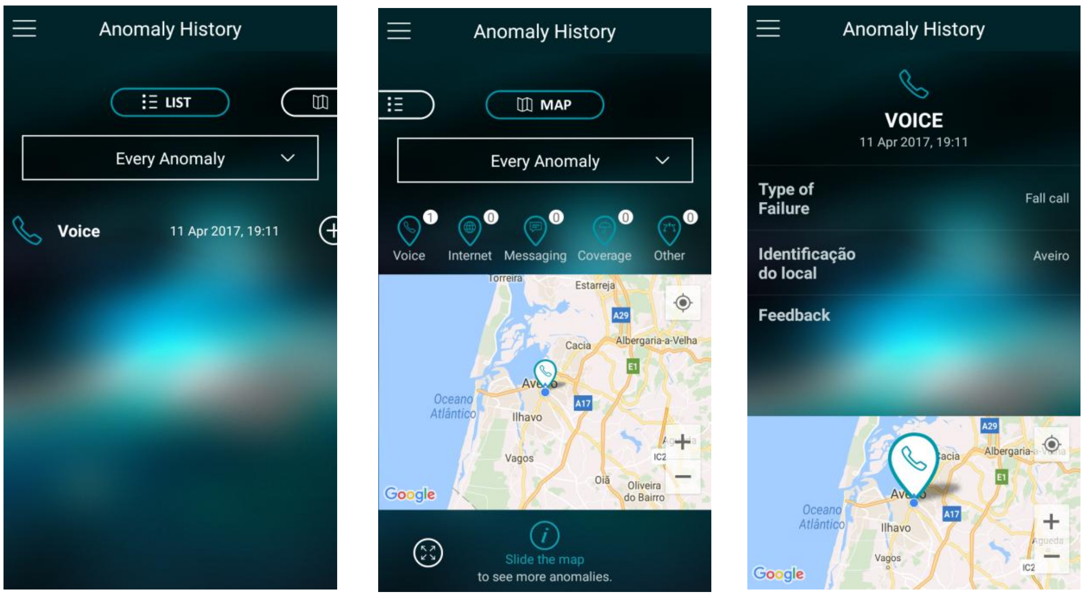


Figure 4.6 - ArQoS Pocket UI: Anomaly history

There was been developed several tests in the building of this application, not only to check quality of service metrics, but also to help on troubleshoot possible problems in the network. These tests can be divided into two categories: intrusive and passive tests. The intrusive ones introduce packets in the network in order to evaluate what is happening in the moment. Passive tests are tests that run in the background, not noticed by the user and don't need to introduce data in the network. Both tests retrieve important information about the environment and networks around and will be detailed in subsection 4.4.

The ArQoS Pocket app tests Voice and Data. In Voice, can be simulated a voice call answer, hang up or record the in-call audio to analyze its quality after. In the data part, web browsing, PING, speed test, portal logins and many more useful tests are done. In the next section, these tests will be used in different scenarios, explaining the QoS metrics that can be gather and what can be inferred analyzing them.

On test’s user interface, there are three lists, one with every test, the second one with the ongoing tests (tests that are currently running) and the third one with inactive tests. If a user clicks on a specific test a details page open. A test can have multiple associated tasks, as it’s seen in figure 4.7. Besides that, in this test’s detailed page the user has total feedback of what is happening with the application in background, being able to check which tasks were already executed with success and the current progression of the test. The orange color on the test title indicates that the test is currently running and when that is the case the “play” icon that it’s seen at the top right corner turns gray and not clickable.

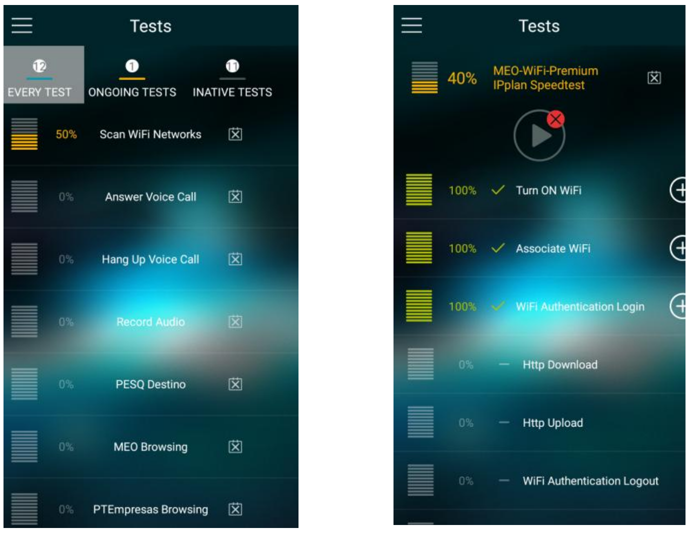
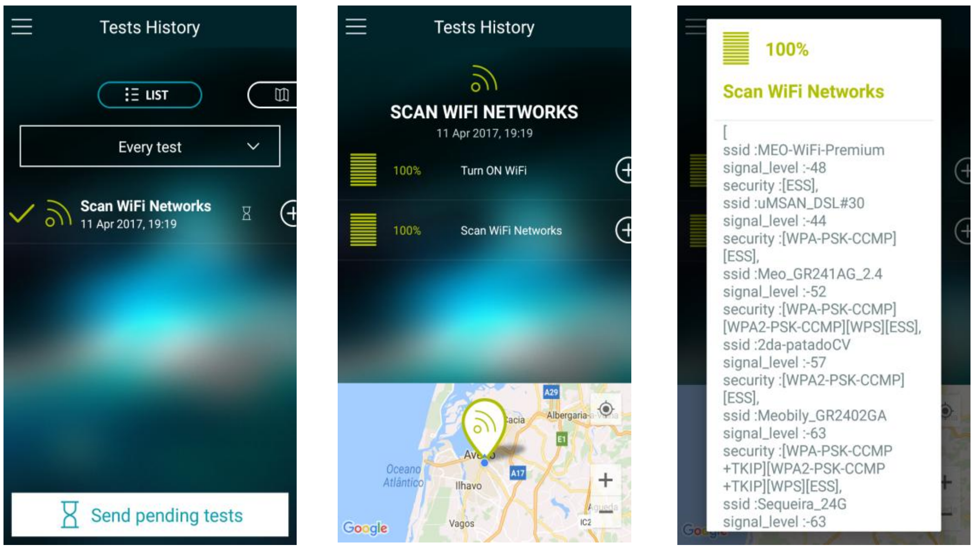


Figure 4.7 - ArQoS Pocket UI: List of tests and test details page

Similarly to anomalies history. It’s possible to see test’s results in a list or map, but in this page the available filters are mobile network, Wi-Fi or every test instead of anomaly’s categories. Tapping a specific test will open a “details page” showing the multiple tasks associated to it. Those tasks will appear with a success (green) or failure (red) symbol at left and after tapping a specific task a pop-up message will appear showing all the metrics and parameters retrieved from that specific task, as well as, a brief snapshot from the mobile and Wi-Fi network state, at the moment that the test initiated.



```
[
ssid :MEO-WiFi-Premium
signal_level :-48
security :[ESS],
ssid :uMSAN_DSL#30
signal_level :-44
security :[WPA-PSK-CCMP]
[ESS],
ssid :Meo_GR241AG_2.4
signal_level :-52
security :[WPA-PSK-CCMP]
[WPA2-PSK-CCMP][WPS][ESS],
ssid :2da-patadoCV
signal_level :-57
security :[WPA2-PSK-CCMP]
[ESS],
ssid :Meobily_GR2402GA
signal_level :-63
security :[WPA-PSK-CCMP
+TKIP][WPA2-PSK-CCMP
+TKIP][WPS][ESS],
ssid :Sequeira_24G
signal_level :-63
```

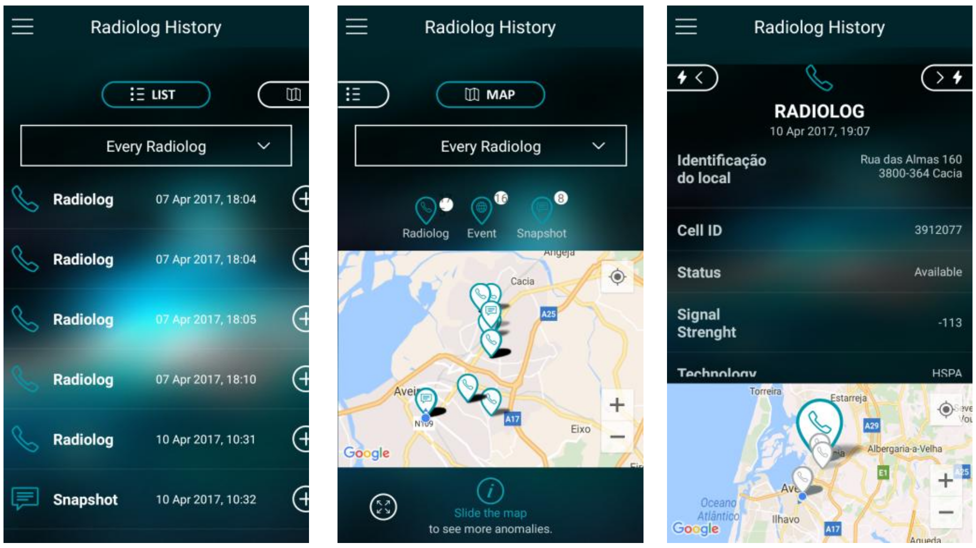
Figure 4.8 - ArQoS Pocket UI: Test history and task details pages

4.1.5 Radiologs/Snapshots

The Wi-Fi and mobile network signal strength and many other important parameters vary a lot either by internal or external conditions of the network. As explained in section 3.1, radiologs are snapshots of the mobile network state, but when it’s referred the term “snapshot” in this dissertation we are refereeing to instant radiologs taken at user’s request. Therefore, while radiologs are generated periodically without user interaction in a background task, a snapshot is taken once at a certain time. The interval time between each radiolog and the option for enabling or not radiolog captures is controlled by the user in the “Settings” page that will be mention right away.

There is a production of some “special” radiologs called “Events” when an event happens. The events that are currently being picked up by the app are roaming state changes, PLMN changes, call establishment, call end, call setup and handovers or cell reselections dependent if a call is currently active or not, respectively.

Radiolog's history page UI can be seen below in figure 4.9 with, one more time, the list or map view. On the right side, it's presented the full content of a radiolog with the local identification, cell id, network availability status, signal strength for that moment in that technology (HSPA), which indicates the device was in 3G, the operator name, if roaming is active, MCC, MNC, LAC or TAC and neighbor's cell information. TAC or LAC number depends if the device is in LTE technology or not, respectively. Signal strength and some other parameters may vary with network technology too, in 2G it's retrieved the RSSI, in 3G the RSCP and RSRP, and the RSRQ and CQI are only possible to retrieve in 4G or LTE.


Radiolog History

Radiolog	07 Apr 2017, 18:04
Radiolog	07 Apr 2017, 18:04
Radiolog	07 Apr 2017, 18:05
Radiolog	07 Apr 2017, 18:10
Radiolog	10 Apr 2017, 10:31
Snapshot	10 Apr 2017, 10:32

RADIOLOG
10 Apr 2017, 19:07

Identificação do local	Rua das Almas 160 3800-364 Cacia
Cell ID	3912077
Status	Available
Signal Strenght	-113
Technology	HSPA

Figure 4.9 - ArQoS Pocket UI: Radiologs history and entry details page

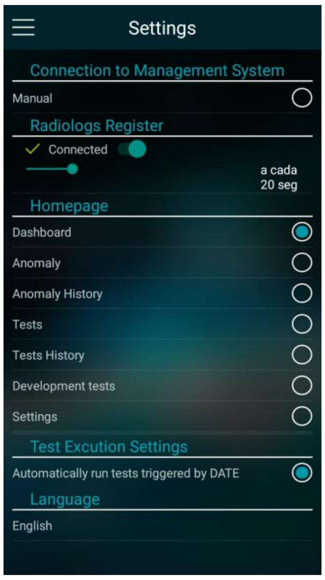
Wi-Fi network information is being retrieved in background too, but don't have yet a structure implemented ready to be delivered to the management system. Nevertheless, both mobile and Wi-Fi information are being attached to all tasks in test's history.

4.1.6 Settings

The settings page has relevant features that may change the application behavior. As was mentioned before, a user can enable radiolog's register and choose a time interval between those registers, has the possibility to set a homepage and last, but not least, the possibility of changing the app language. There are three languages supported: English, French and Portuguese. The

“Tests execution settings” are currently disable, since the app currently only has the possibility of automatically run tests triggered by date.

The app also has an “About” page explaining the objectives of the application and reserving all the copyrights associated to the last.



Settings

Connection to Management System

Manual ○

Radiologs Register

✓ Connected

a cada 20 seg

Homepage

[✓ Dashboard]
Anomaly ○
Anomaly History ○
Tests ○
Tests History ○
Development tests ○
Settings ○

Test Excution Settings

[✓ Automatically run tests triggered by DATE]

Language

English

Figure 4.10 - ArQoS Pocket UI: Settings page

4.2 Technical implementation

On the previously subsection 4.1.1 was referred “background service”, this services supports a large portion of the UI information that it’s seen on the application. Wi-Fi, mobile, and scheduled tests callbacks are done in this services, in other words, every time that it’s needed to update a scheduled test date, update the Wi-Fi or the mobile network information is those services that handle the updates and then sent the information again to update the respective UI’s pages.

Anomalies and radiologs results are yet being stored in files and it is a background service that manages this information calling the respective entities, procedures and loading the history pages using the respective adapters. Those adapters transform raw data in structured data that can then be used in the UI or delivered to the management system.

There is another type of adapters that handle the multiple fragments from an activity. These fragments represent a behavior or part of the UI and they can be reutilized in multiple activities. In figure 4.5 it’s presented a great example of this with multiple fragments all in the same activity and with the same purpose, report an anomaly.

The application can be adapted to different type of users and deployed in multiple scenarios, working as stand-alone or with the help of the already existing mobile probes on the network. Users have in the settings page a couple of options to explore, allowing the application to adapt itself according to user's preferences. The management system connection is being implemented at the moment, all the details and functional features are explained in section 4.3. Still about the settings page, a user can activate or not the radiolog register and choose the time interval between each register. He can choose his application homepage and the app language. All of these parameters/choices are saved locally on the S*haredPreferences* that is an internal Android interface for accessing and modifying preference data. That way the same application installed in different devices can run with a different configuration that is saved locally in each device.

A different service running in background, as well, is a based GPS location service that retrieves the latitude and longitude coordinates, altitude, speed from the accelerometer, among other device parameters. This service is used primarily to show pins/markers of the different anomalies, tests or radiologs on history's map view, but also to attach to every task details the GPS information that will after be sent to the backed system.

Lastly and associated to the test's schedule there is background service handling the Alarms. For now, there are only on-demand and scheduled tests from a JSON file or from the management system, respectively. An alarm is triggered every time that it's needed to insert, update or cancel a test. It is being used the *AlarmManager* Android class that already provides the tools to handle the alarm services and therefore, ensuring that the application is capable of running a test at a certain time in the future.

4.2.1 Frameworks used

Throughout this dissertation many tools were used to build a consistent, integrated and operational solution which helps to solve the identified problems discussed in previous chapters. Those frameworks/technologies used are the Android SDK, JSON, SQLite database, Google Maps API, ReactiveX for Java, Requery and image edition tools.

The Android SDK is the software that helps and enables the development or creation of applications for the Android platform. For this project it is being used Android studio with 2.3.1 version and the application is being tested with an emulator included in SDK, but primarily, in real devices whose specifications can be encountered in Appendix D.

Test's data, parameters and configurations are being retrieved from a JSON file and handled by an internal database. Every time the application boots it updates its database tables content if changes on this JSON file happened. The user interface of test's page is also updated with the available on-demand and scheduled tests in the same file. Still about the database implementation, it is being used a framework, named requery, that was built with Android in mind, providing classes for SQLite databases and useful UI adapters. Primarily to facilitating much of the database creation it also brings other advantages like lifecycle callbacks, JPA annotation support, caching and no dependencies are needed, however, RxJava is recommended.

ReactiveX is a cross-platform API available in multiple programming languages used in known companies like GitHub, Netflix and SoundCloud that manipulates UI events and API responses. Furthermore, it handles concurrency and asynchronous processes that almost every Android application needs to implement because of the requests on the internet that are not allowed to be run in the Android main thread. Compacting some of the code, this framework is equipped with proper mechanisms for handling errors and allowing the programmer to abstract away low-level threading, synchronization, and concurrency issues. In our solution, RxJava is being used mainly to handle asynchronous events and database transactions.

The map that it's seen in the distinct history pages, the personalized markers, the zoom and all the animations associated are done using the Google Maps API. For the customization and the design not only in the map, but in all the assets, images and icons were been used diverse image edition tools, such as Adobe Photoshop, GIMP among others.

4.3 Management System Connection

Reported anomalies, radiologs and tests results registered in every device running the application are sent to a management system that gathers all the data and analyze it, producing views, helping to troubleshoot operations and facilitating the production of network reports. Furthermore, it is also possible to manage all ArQoS probes and configure scheduled tests through this backend system.

The connection between the application and the management system is supposed to be automatic, but it is not possible without a rooted phone to change programmatically the default associated APN. Therefore, for this initial phase was created a slider in the "Connection to the Management System" group of *Settings* page that user must drag to initiate the connection with a management system simulator to test all the answers and requests made by both sides. With this setup, it’s possible to confirm that all JSON content is well structured and if it’s following the documentation previously defined.

In the ArQoS pocket solution, the probe discovery starts with an HTTP POST to the simulator endpoint with the equipment type, serial number, MAC (IMEI), IP address and timestamp. This way simulator knows to who he is talking to. After validating the IMEI and IP address it sends a response to the probe’s endpoint based on its IP. Then periodic keep-alive messages are exchanged between the probe and the management system. Furthermore, when the probe’s IP address change, for example, when 4G connection drop to 2G because of coverage issues there’s also an “IP update” message similar to the *auto discovery* message, but with the old and new probe’s IP address in addition.

From the management system is remotely possible to configure the probes, get their “hardware” information, get their occupation between two dates, check the configured tests and schedule tests. All of this is being done with *HTTP POST* messages, which goes a bit against the standard of HTTP protocol, due to the GET and DELETE methods that it has. However, in this dissertation was developed only the application side and those POST messages were already implemented and are being used on the fixed and mobile probes already. Therefore, to keep the consistency between all ArQoS probes was given the support for all these messages, already defined.

Test’s results delivery was implemented as well and following the initial requirements, they are being sent with a snapshot of the mobile, Wi-Fi network information and GPS location attached. After the results delivery to the management system, they are being deleted from the phone after 3 days. This time can be changed or configured through a different request.

The messages requests already implemented and supported are the “Load Test Request”, that loads a scheduled test into the app, the “Cancel Test Request”, which cancels a configured test, a request to change the probe configuration, for example, the endpoint of the simulator to the real management system in production, a request to get the probe’s hardware information and a request to get the real time probe’s state. The notifications implemented and that are being sent already are keep-alive’s, the auto-discovery process, the IP update process and the results process used when sending test’s results information.

An example of the request asking for the probe's state and the probe's respective response can be seen in the snippet 4.1.

```
{
  "msg_type": 2,
  "macaddress": "357810081051139",
  "modulo": 0
}
```

```
{
 "probenotification": {
  "associatedResponse": {
   "bat_level": 13,
   "bat_time": -1,
   "call": "48040-50976",
   "connection_type": 0,
   "country_code": "pt",
   "df": 85,
   "gbe_addr":"46.50.24.134",
   "gps": "4037.77631 N 838.75725 W
     0.000000 0.000000",
   "gsmroam": 0,
   "iccid": "8935106000061182108l",
   "mcc": 268,
   "mnc": 6,
   "modem": 10,
   "pwr_status": 2,
   "rede": "rede=gsm=1 gprs=0
     umts=0 lte=1 operadora=MEO",
   "rsrp": -84,
   "rsrq": -7,
   "sinal": -84,
   "sim_state": 0,
   "sim_type": 0,
   "sync": "GESTOR",
   "tarefa": 0,
   "temperatura": 31,
   "teste": 0,
   "wifi_addr":"46.50.24.134",
   "wifi_connection_state": 10
   },
 "equipmenttype": "13",
 "ipaddress": "46.50.24.134",
 "macaddress": "357810081051139",
 "msg_type": 254,
 "serialnumber": "357810081051139",
 "timestamp": "20170609180647"
 }
}
```

Snippet 4.1 - Get probe's state request and response, respectively

There is still much more to be implemented on the communication with the management system, but for now, that's what was considered a priority. Future requests like the delivery of radiologs or voice tests will be referred in chapter 6, Conclusions and Future work.

4.4 Network and Data tests

New different tests were implemented according to the established requirements and meeting what mobile probes already do. These tests can be more intrusive introducing packets in the network or non-intrusive, retrieving information from the Wi-Fi or Mobile network passively.

Many tests are already thought, such as FTP download and upload, MMS or e-mail tests some of them are already implemented and others only have their configuration set up, but aren't yet functional.

Tests can run on-demand in the app or be scheduled to a certain date by the management system. At the time this is being done through a JSON file edition placed on the */storage/emulated/0/arqospocket/tests/* of the device. Tests that are already implemented with all the requirements and ready to be delivered to the management system are SMS tests (send and receive), HTTP tests (download and upload) such as MEO GO browsing and MEO Wi-Fi Premium login or speedtest.

Every test has its specific configuration, in snippet 4.1 it is given an example of the complexity involved in an SMS test. All the tests have an initial and final date, despite these dates are only relevant to scheduled tests in the sense of the test validation that only happens if the actual date is between the referred dates. Furthermore, a test has a unique id, a test type number, for now we will focus only on on-demand tests (5) and scheduled tests (0), state, priority and can have more optional parameters, such as a recursion object with an event and an interval in seconds for the repetition of the task specified, in this case it would serve to send an SMS every "x" seconds.

In this specific test, there is a more complex parameter called "data". It has a lot of parameters that are specified in ArQoS internal documentation, such as the task timeout, the destination number, message text, etc.

The "Receive SMS" is a different task because the application must wait for a specific SMS, validating it by expected text or if metadata is included, which is the snippet's 1 case because the last pipe is empty, validate/accept the SMS based on the trailer text. This trailer, along a unique timestamp are combined allowing the receiver to know that SMS is or not the expected one, making the task have success. If the receiver does not receive an SMS that matches the requirements the task will end up finishing by timeout.

```
{
    "testeid": "T16",
    "dataini": "20170531164600",
    "datafim": "20200717133554",
    "modulo": 0,
    "testname": "Send SMS DATE COM METADATA",
    "macronr": 1,
    "testtype": 0,
    "state": 0,
    "priority": 1,
    "data": [
        "1|1||0|30|12|0|967200007|ArQoS Pocket SMS
Test1234|1234|0|0|||"
    ]
},
{
    "testeid": "T17",
    "dataini": "20170510105200",
    "datafim": "20200717133554",
    "modulo": 0,
    "testname": "Receive SMS DATE COM METADADOS",
    "macronr": 1,
    "testtype": 0,
    "state": 1,
    "priority": 1,
    "data": [
        "1|1||0|30|36|0|1234||"
    ]
}
```

Snippet 4.2 - Send/Receive SMS with metadata test configuration

In the next chapter "Evaluation and Results" will be approached many of these tests and analyze results varying both these parameters and the scenario's external conditions.

For tests related to the mobile network there are the following new tests implemented in the application:

- **Start call with a custom APN:** An APN is a gateway between a computer network, normally the public internet and the network used by the device technology (GSM, GPRS, 3G, 4G). In 3GPP data access networks, an APN is used to identify the packet data network that the user wants to communicate with or to define the type of service (e.g MMS, WAP). It also contains a mandatory *network identifier* that defines the external network to which the GGSN is connected and an optional *operator identifier* that defines the operator's packet domain (e.g. Internet-v4.mnc111.mcc222.gprs, mmsc.tmn.pt).

In the development of this test was found a list with 1370 APN's and its configurations, in further investigations was confirmed that Google has a *.xml* file on Android devices keeping this information, normally under the */etc* folder. The correct APN chosen is defined by the SIM card inserted in the device.

Changing the preferred APN requires the application to be a system application. As explained in section 2.5.3 these applications are located under the *system/app* folder and have some permissions that user apps don't have, allowing, for example, the change or insert of new APN's on the device's configurations.

- **Send DTMF tones:** On the first telephones the dialing was done through a "disk" that generated a sequence of pulses. Dual-tone multi-frequency signaling is a substitute for that on keypad phones. When a user presses a key is generated a high frequency and a low frequency based on the key pressed and the sum of the sinusoids of the two frequencies is then sent to the central that analyses the signal, knowing which key was pressed.

Hz	1209	1336	1477
697	1	2	3
770	4	5	6
852	7	8	9
941	*	0	#

Table 4.1 - DTMF table

In the app it's possible to send a sequence of DTMF tones, for example, interacting with voicemail service automatically (e.g. "200,3,,4,,1"). Unfortunately, in Android is not possible to send this tones on the voice's uplink during an active call as an app developer, but it's possible to do it when initiating the call. There are a few contributions, at the moment with new implementations of the CallManager Android class waiting for Google developers to review and accept it on "Android open source project".

Different tests are done passively in the app without user interaction, in other words, they are done only by listening the network itself gathering information or by analyzing other sources, as in the case of the Logcat reading:

- **Scan Wi-Fi Networks:** There are a few networks around us almost every time, in big cities more than a few. Knowing which network has the best signal level in a certain location, the network's name or which network have or not a security password allowing a free connectivity is important to the user. The Android does this scan natively periodically, but this scan is being performed, in the app because there can be retrieved valuable parameters that Android doesn't show us in his main interface.

This test is done passively and periodically in the app, but if the user wants he can run this test on-demand, as well. It is obviously required that the device's Wi-Fi is turned on, so it is turned on at the beginning of the test if it isn't already. After that is performed the scan and the parameters retrieved from this test are: ssid, bssid, signal level, management key protocol, security capabilities, frequency, timestamp, channel bandwidth, center freq, etc.

- **Write LogCat to a file:** Android has several files where he dumps information about what is happening with the system at that moment. The location of this files are not standardized (i.e. some can be ROM-specific):
  - *‌/data/anr:* Some trace files seem to get here (Dalvik writes stack traces here on ANR, i.e. "Application Not Responding" aka "Force-Close";)
  - */data/dontpanic* seems to be a standard location (AOSP), and contains some crash logs including traces.
  - */data/kernelpanics* is another location but not having any "kernel panic" on the Android device yet means no content there yet too.
  - */data/panic/panic_daemon.config* may point to other locations configured like */sdcard/panic_data/* or */data/panicreports* directory.
  - */data/tombstones* may hold several tombstone_nn files (with nn being a serial, increased with every new file). As tombstones are placed for the dead, it is done here for "processes died by accident" (i.e. crashed) and it is what is referred to as "core dumps" on Linux/Unix systems. However, not all apps create tombstones. This must be explicitly enabled by the developer.

Most of the logging is done on *tmpfs* but with a reboot, these data are lost. Most developers usually use these logs to help troubleshoot problems or crashes in applications, but there is a lot more information there divided into 5 levels – verbose, debug, error, info and warning. Furthermore, there have been concerns about user's privacy, because despite we can write messages to logfile to use as debug, it is also possible that SMS/MMS, contacts information or e-mails be written, as well [22]. In the snippet 4.2, it's seen a relevant example of private information that can be used by hackers:

```
I/ActivityManager(526): START u0 {act=android.intent.action.VIEW
        dat=google.navigation:///?ll=49.872600,8.636720&q=Alexanderplatz,Berlin&
token=FdjAIdMMmDACmJjSWej3C9RzGzM32OzZezHw&entry=w&opt=cmp=com.google.android.apps.maps/com.
google.android.maps.driveabout.app.NavigationActivity} from pid 22056
```

Snippet 4.3 - Logfile - information about user location [22]

Many more examples can be given, accessing a Wi-Fi network expose the SSID or the associated MAC address, after enabling Bluetooth was founded the Bluetooth address written in clear text or when a user opens a specific app, the log file contains the package name of the selected app. Although all these examples were founded in clear text, there is also information that is sanitized. In fact, there are some ways that explicitly prevent some information to be written in the logfile. If we call the toSafeString() method URI's are sanitized before they are written into the "log file". So, URI's started by "tel:", "smsto:" or "mailto:" are written as "xxxxxxx".

```
I/ActivityManager(522): START u0 {act=com.android.phone.SIP_SELECT_PHONE
        dat=tel:xxxxxxxxx flg=0x10000000
cmp=com.android.phone/.SipCallOptionHandler (has extras)} from pid 767
```

Snippet 4.4 - Logfile information: sanitized information

This is still a problem nowadays if you have a rooted phone. If you have not, since Android version 4.1, Google don't allow to read log entries from other applications anymore. So if someone dumps the "LogCat" in an unrooted device with a version higher than 4.1 he will only see log messages from its own app.

All this information has been tested and for reading logs on rooted devices it's needed to grant permissions, not only in the "Manifest.xml", but in real time via shell commands too. After that is only needed to write the LogCat in a background process, to a .txt file.

Chapter 5

Evaluation and Results

In this chapter, it is presented all the obtained results from the monitorial experiences, in order to validate the proposed solution, in chapter 4. All results will be discussed and compared with the already existing results from fixed probes deployed all over the country. Considering the requirements presented in chapter 3, will be analyzed the effectiveness, metrics, performance and behavior of the solution varying the conditions or environment on its usage.

5.1 Probe's deployment scenario

The ArQoS Pocket solution can be used as a stand-alone application, but it was thought deployment scenarios for this solution that passes by integrating the pocket probe with the already existing fixed and mobile probes. These two solutions communicate using the Android ADB through a USB connection and the synchronization needed for the tests results is guaranteed by an NTP server running on the probe by which pocket synchronize.

There are many scenarios thought for the pocket probe integration, this first test aims on studying the interference of the signals generated by the mobile probe on the signal strength captured on the phone from cell towers in the different technologies and conclude which deployment scenario is the best to be adopted.

The possible scenarios considered are: the pocket probe outside the suitcase and far away from the mobile probe, the pocket probe on top of the mobile probe with a cover dividing them and lastly, the pocket probe on top of the mobile probe without any cover dividing them.

Below it is presented graphics generated and displayed considering 20 measures with 8 seconds of interval between them and the devices used for the test were a Xiaomi Redmi 3S and a Samsung Galaxy S7, both using MEO's mobile network. Figure 5.1 takes into consideration the average value of all those measures, as well as the statistical parameters associated: variance and confidence interval.

For 4G LTE was additionally registered the interferences influence into RSRP and RSRQ parameters that only exist when the device is connected to this network. Taking all that into consideration is important to refer that only the graphics about the tests realized with the Samsung Galaxy S7, which is the one that will be used in the real scenario deployed in the future is shown and analyzed in this section.

Figure 5.1 shows the average, with a confidence interval, signal strength level captured by Samsung Galaxy S7 in the 3 scenarios previously described at 4G (LTE), 3G (HSPA), and 2G (EDGE) at left, middle and right respectively.

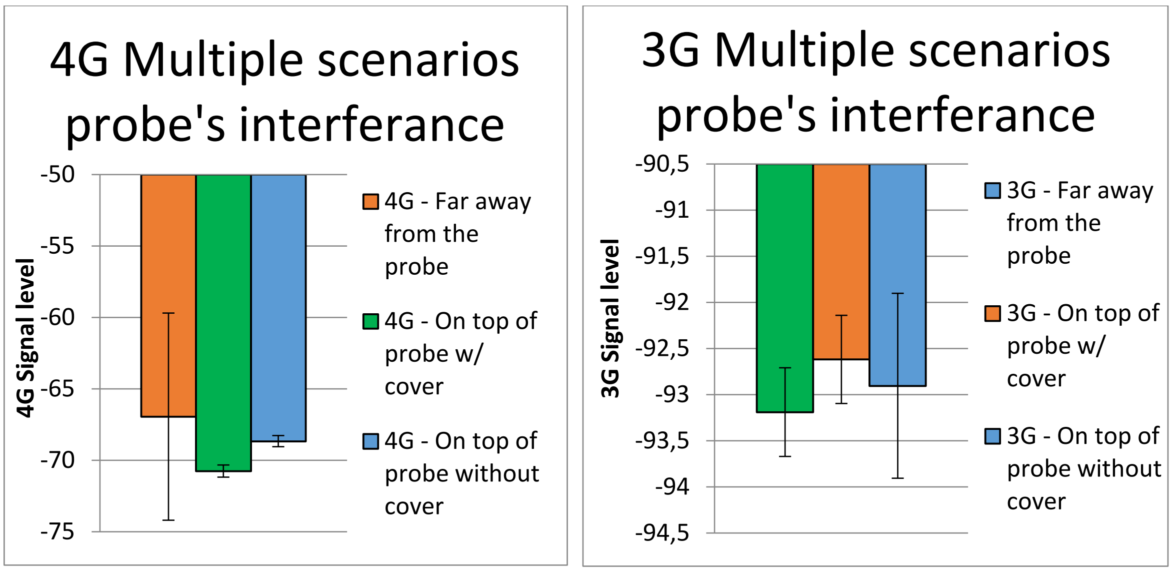


Figure 5.1 – Test A: Mobile probe interferences on pocket's radio signal strength in 4G and 3G

With an analysis of these results it's seen that in 4G the best-case scenario was the one with the pocket on top of the mobile probe with a cover dividing them, in 3G was the pocket far away of the mobile probe and lastly in 2G the best scenario was again the scenario where it was a metal cover dividing both probes. On a deeper analysis and plotting the confidence interval, was seen that the results between the case scenarios are similar although the differences of the signal strength values, but in other cases, there was a huge discrepancy between results. To try explaining this issue was performed a similar test, in different time of the day, whose results can be seen in figure 5.2 for 4G LTE technology.

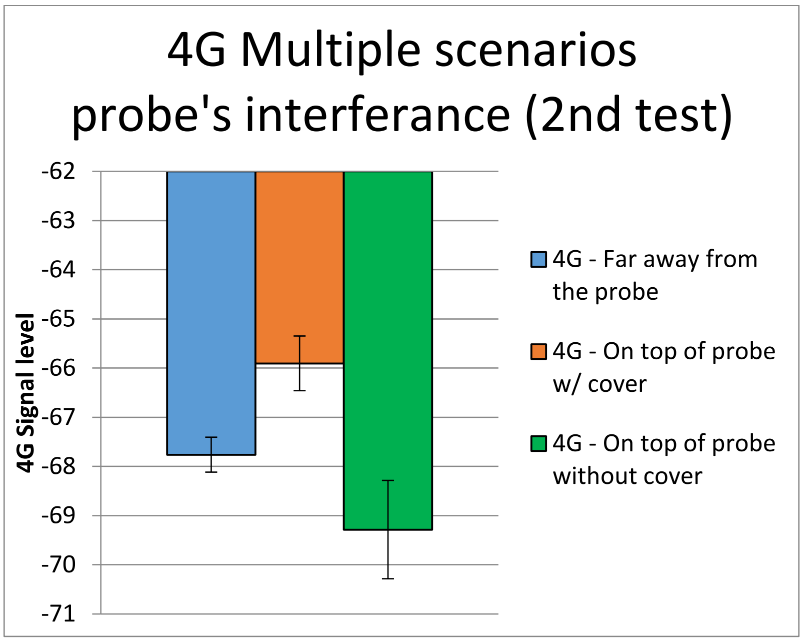


Figure 5.2 - Test B: Mobile probe interference with pocket 4G LTE radio signal

Comparing the overall values, as absolute, of signal level retrieved from the network with Test A it is seen that these values are a bit lower in all deployed scenarios. This can be explained with the time of the day the test was executed because at that test's hour many people were leaving the workplaces and going home, being the city with more movement and causing the cells to be busier with handovers.

However, comparing the individual deployed scenarios on both tests A and B was verified that the best-case scenarios results didn't match at all. Even over, the best-case scenario in Test A become the worst one because when analyzing radio signal strength, a more negative value is a better signal level value. The same inconsistency between tests A and B results happened in 3G and 2G. Therefore, can be concluded that the signal strength level is more affected by external factors like the number of people using the network at a specific moment or weather conditions than that the position of the pocket in relation to the mobile probe.

Taking that into consideration and following the best security procedures, the prototype scenario for the ArQoS Pocket solution integration can be seen in figure 5.3 with two devices placed on top of the mobile probe.

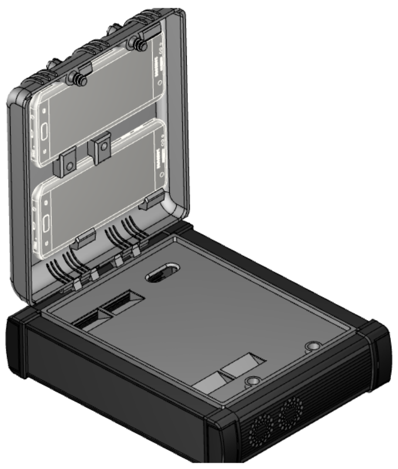


Figure 5.3 - 3D prototype of open probe with the Pocket solution integration without wiring

5.2 Performance tests

In this subsection, it would be tested the application overall performance. The first test was done by creating diverse Android virtual devices on the emulator and executes the app on different devices with different Android versions and sizes in order to detect possible anomalies in a specific device. Our solution targets the Android version 6.0.0 and above, but it was verified that multiple devices with a version below could still run the app and perform all its associated features. Taking that into consideration, it is presented below a detailed table with the different devices tested:

Model	Emulated (Yes or No)	Android version	Size (inch)	API level	Run the application? (Yes or no)
Nexus One	Yes	Android 2.3.3	3.7	10	No
Nexus One	Yes	Android 4.0.3	3.7	15	No
Nexus One	Yes	Android 4.3.0	3.7	18	No
Nexus One	Yes	Android 4.4.2	3.7	19	No
Nexus S	Yes	Android 5.1	4.0	22	Yes
Galaxy Nexus	Yes	Android 5.1	4.65	22	Yes
Nexus 4	Yes	Android 6.0.0	4.7	23	Yes
Nexus 5X	Yes	Android 7.1.0	5.2	25	Yes
Xiaomi Redmi 3S	No	Android 6.0.1	5.1	23	Yes
Samsung Galaxy S7	No	Android 7.0.0	5.0	24	Yes

Table 5.1 - Performance test: Which devices run the application

With this test, was verified that although the code targets the API 22 many devices with Android versions with lower API levels can still run the application with no issues, which is really satisfying. Was verified that Android devices with a version below that 5.1 can't run the application due to manifest permissions needed and in devices below the API 14 it is not even possible to install the application.

The second performance test made was measure loading times of different pages in the application, particularly the history pages varying the number of entries on it. For this test was used a Xiaomi Redmi 3S with Android version 6.0.1.

The transitions between distinct pages in the app are smooth because the operations that require more time are all being done in background with transparency to the user and never locking the UI. The first page can be seen when the initial splash screen disappears. This transition happens when a callback is sent saying that the UI is updated and the page is ready to be seen. The tests bellow measured this first page load time being the "Dashboard" page the application homepage selected. Furthermore, load times of anomalies and radiologs history page can also be seen with 2 anomalies entries and 180 radiolog entries, respectively.

Pages load time (first time)			**Pages load time (cached)**	
Initial page	Anomalies History	Radiologs History	Anomalies History	Radiologs History
2,6s	2,2s	2,7s	0,8s	1,5s

Table 5.2 - Performance test: Pages load times

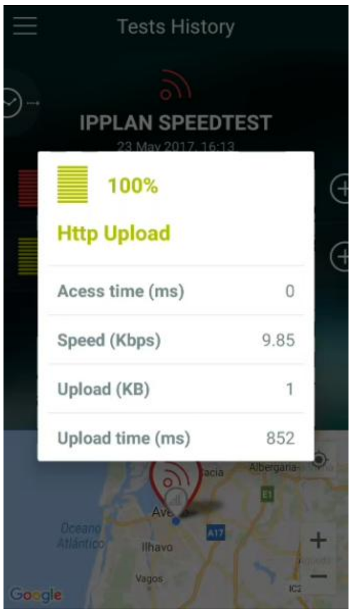
We think that wait for the callback in the splash screen to show the first page user interface with the correct details to the user is the best option, even so, we are quite happy with the load times presented above. If the pages have never been opened once, the times are a bit higher than 2seconds even with only a few entries on the history (anomalies case), but from the first consultation, the load times are significantly reduced (~54%), on further queries or history visits.

As already said, all the other transitions in the app are very smooth, this load times on history pages may be due to the Google maps call and the associated processing, for example, putting the markers in the correct geographically positions. Notwithstanding the times presented, especially after the first consultation are very satisfactory with the app performing all operations in a short time.

5.3 Internal tests evaluation

Continuing analyzing our solution the internal tests performed by the application are a very important requirement that needs to be functional and well implemented independently of the scenario where the application is deployed. To evaluate those tests fortunately, this project had already specific requirements specified for the executed tests from the ArQoS NG probes. There were already input and output parameters identified by other probes. This solution is getting the many parameters and values as it can and new parameters identified from ArQoS Pocket solution only, since this is a different mobile solution running in a real terminal, are being retrieved and documented to add support, in a close future, at the management system.

The internal tests that are mentioned are the ones already mentioned in section 4.1.3 since PING's to Browsing or speedtests. All of them have their specific metrics, whose examples can be seen in Appendix B. Obviously, as already mentioned and since this is a solution running on a real Android terminal not every parameter in the list is possible to get, however, the parameters that are possible to retrieve are being documented in internal documentation. In the app, this parameters/metrics can be seen at the task's details page, for example, for an HTTP Upload task can be retrieved the access time, upload speed link, size and time of the upload.


Tests History

IPPLAN SPEEDTEST

23 May 2017 16:13

100%

Http Upload

Acess time (ms)	0
Speed (Kbps)	9.85
Upload (KB)	1
Upload time (ms)	852

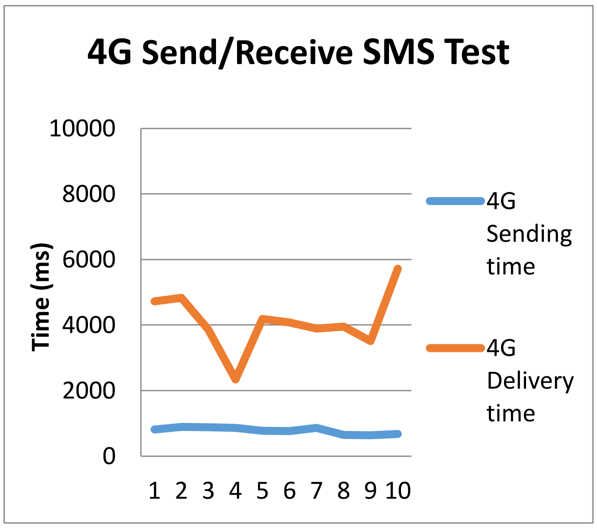
Figure 5.4 - Task detailed information

5.3.1 SMS tests

The SMS tests are an example of the help that is knowing a priori the expected values for test results. It is quicker to validate the experiment and the output values because the SMS's sent through pocket should have similar times compared with times measured on the mobile probes deployed already on the network.

To validate the SMS's times sent at the request of our application will be compared the times measured by the task results with the delivery and sending times measures on the ArQoS NG probes. In figure 5.5 can be seen, in the same operator's network but varying the mobile technology of the sending and receiver devices, the time of sending aN SMS to the SMS Center and the delivery time that is the time since SMS leaves the source till it reaches the destination number.

The values meet the expectations, as we go from 4G to 2G the average times of delivery and sending an SMS increase and those same values match the numbers measured in the mobile probes. Furthermore, sending an SMS to a different operator network or between different network technologies increases those times too which meet the expectations, as well because in the path that the SMS must travel in both cases is longer. Other factors that can influence these times are location issues, network traffic or mobile device issues, for example, if a device as low battery level, that may impact negatively the message delivery time.



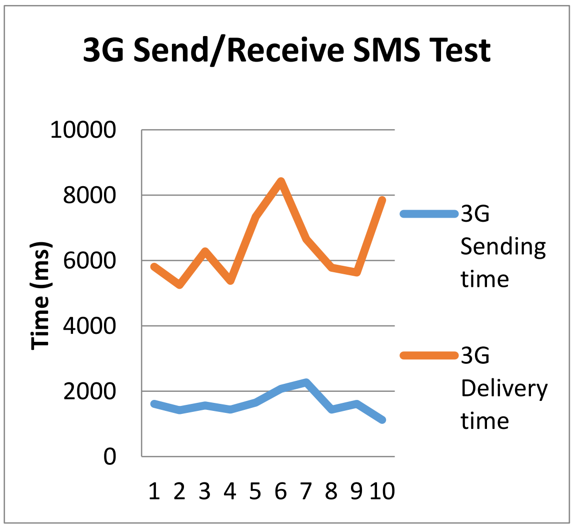


Figure 5.5 - SMS sending and delivery times on multiple technologies

5.3.2 Radiologs

Radiologs were tested in a different context and to validate this feature it was taken the device on a drive-test through the city with the application running in the background. Radiologs register were already been tested, but for catch especial events like active roaming, handovers or cell reselections it was needed to travel with the device.

In the figure 5.6 can be observed a map view from the "radiologs history page" after the test drive execution with many radiologs taken periodically and it is possible to see, as well 16 events captured at different locations.

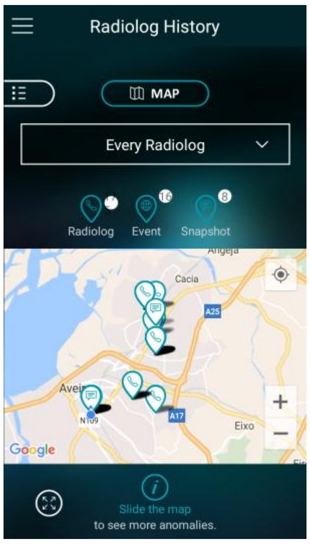
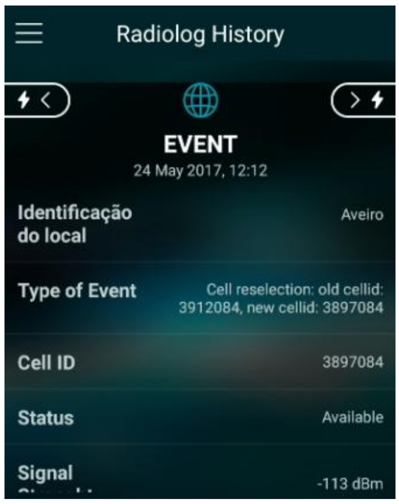


Figure 5.6 - Radiolog map view after drive-test

Opening those events, it's seen that they are special radiologs, taken apart from the periodical ones, capturing specific incidents on the network, for example, automatic cell reselections done by the device as it travels or moves to a different location with better coverage.


Radiolog History

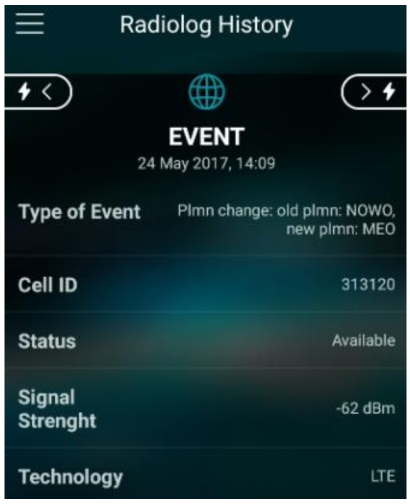
EVENT

24 May 2017, 12:12

Identificação do local	Aveiro
Type of Event	Cell reselection: old cellid: 3912084, new cellid: 3897084
Cell ID	3897084
Status	Available
Signal	-113 dBm

Figure 5.7.1 - Event: Cell reselection

Another test was made in this regard in order to try capture and validate a PLMN change. For this was withdrawn the SIM Card of the device and placed another from a different operator. The result of this procedure was as expected, an automatic event triggered with the details presented in figure 5.7.2. The full radiolog event can be seen in Appendix C.


Radiolog History

EVENT

24 May 2017, 14:09

Type of Event	Plmn change: old plmn: NOWO, new plmn: MEO
Cell ID	313120
Status	Available
Signal Strenght	-62 dBm
Technology	LTE

Figure 5.7.2 - Event: PLMN change

5.3.3 Log Analysis

With a view on discovering what logfile has to offer in terms of content, it was made specific tests to discover if is possible to get more information from the network than from internal Android classes and what information from OTT apps like Facebook or YouTube it's possible to retrieve from there as well. Was found interesting network information that may substitute the

QoE oriented anomalies report page in the app. Additionally, was invested time in seeing what is possible to retrieve from calls, Bluetooth, phone integrated apps and lastly features like the dialer or battery.

Mobile network information it's being extracted from internal Android classes like CellInfo, CellIdentity, TelephonyManager, etc. Was founded that outside the application the logfile contains calls to the SignalStrength API, similarly to what is done in the app, containing the signal level in dBm's, LTE Asu level and some other parameters.

Other periodic and frequently appearances are from the Battery Service giving the level, scale, status, health, voltage, temperature among others values. Interactions with the keyboard like pressing the Home button could be seen, as well with a search for the "*performOnHomePressed*" phrase. Having this in mind it was verified if the keyboard characters typed in the phone dialer were been saved in the logfile as well, which would be a major security problem worldwide. Fortunately and meeting the expectations only the press and release events could be seen. These actions messages can be seen below, in snippet 5.2.

```
I/InputReader: Touch event's action is 0x1 (deviceType=0) [pCnt=1, s=]
....
I/InputDispatcher: Delivering touch to current input target: action: 0x1
I/InputReader: Touch event's action is 0x0 (deviceType=0) [pCnt=1, s=0.1200 ]
```

Snippet 5.1 - Logfile information: Dialer touchscreen

Wi-Fi associations, disassociations and Bluetooth information were also tested, being encountered lot of information about the processes that the phone do in background such as, Bluetooth bond state changes, when discovery process starts, connection status or when Bluetooth is turned off and the internal broadcast states numbers for those transitions. The information extracted gives an overview of the internal protocol process, but none of the information seen is critical to explore potential vulnerabilities.

Other great point of focus, not only on the logcat reading, but also throughout the dissertation was the audio on the phone calls. Android has 3 phone states *IDLE, OFFHOOK* and *RINGING.* These states could be seen on logfile along with other state updates, notifications, requests and more interesting yet, call ends, drops or hang ups timestamps could be seen as well. In the snippet 5.2 is shown an example of a call establishment:

```
D/PhoneUtils: setAudioMode()...OFFHOOK
D/CallManager: setAudioMode useInCallMode = false, Baseband = csfb
D/PhoneUtils: setAudioMode() no change: MODE_IN_CALL
D/Ringer: stopRing()...
D/Ringer: - stopRing: null mRingHandler!
W/Vibrator: Vibrator.cancel()
D/CallNotifier: - posting UPDATE_IN_CALL_NOTIFICATION request...
D/PhoneUtils: - hangup(Call): regular hangup()...
D/PhoneUtils: ==> hungup = true
D/PhoneApp: pokeUserActivity()...
D/PhoneUtils: Not supported: BargeIn
D/PhoneUtils: ConnectionHandler: updating mute state for each connection
D/PhoneUtils: setMuteInternal: using setMicrophoneMute(true)...
```

Snippet 5.2 - Logfile information: Connected call with microphone mute

Similarly, it's also possible to check if a call has ended through the "*need to play CALL_ENDED tone*" phrase, if a call was not answered or if an in curse call dropped. This information can be useful and substitute partially the anomaly's page report since the information is automatically retrieved by parsing and analyzing the Android logfile. Although, this can only be used on rooted phones and many specific manufacturers' device messages may be mutable from phone to phone.

```
I/Telecom: Event: Call 4: SET_DISCONNECTED, disconnected set explicitly> DisconnectCause [ Code:
(OTHER) Label: () Description: () Reason: (Protocol error, unspecified,
PROTOCOL_ERROR_UNSPECIFIED) Tone: (-1)]
```

Snippet 5.3 - Logfile information: Dropped call

5.3.3.1 OTT Apps Analysis

In section 2.6, Existing Solutions, was founded several applications that supported OTT applications, like Facebook, Dropbox or Skype. It is not known what that support means exactly but, in order to try obtaining information about these Over The Top applications running almost in every device was tried to analyze the Android logfile giving special focus on finding the maximum possible information on it about these apps. The messages presented in this subsection are from a Xiaomi Redmi 3S and may vary from mobile to mobile.

On opening any Android app, it is possible to see the app's package name appearing in the logfile, for example, *com.google.android.youtube* or *com.facebook.katana.* The applications tested were YouTube, Facebook, Facebook messenger and WhatsApp.

It is well known that YouTube website has a reach analytics page for its users check easily their subscriptions, likes or views. When a user opens a video from YouTube's app it's seen, in the logfile, an *URL* call to a specific *API* for handling those statistics. Still about the video, it is possible to see a lot of information about its audio parameters such as AudioFocus, stateUpdate, tansportControlFlags, etc.

In the Facebook case, is incredible the amount of the information that can be extracted from the logfile. Despite that non-personal information is not seen, it's possible to see when a user clicks on the Facebook history button to check the chronology or when he enters on a person profile page. Clicking in the messenger button from Facebook app shows a message with "Start proc" followed by the messenger Android package name similarly to what happens with the other app's opening. Furthermore, due to these OTT messaging apps like, Facebook messenger or Google hangouts, nowadays, could be considered the device's default messaging app, was founded some errors exposing when a user clicks to start a conversation with someone without photo from internal device contacts.

```
W/msgr.RequestLoggingListener: time 271653383: onRequestFailure: {requestId: 44,
elapsedTime: -1 ms, throwable: java.io.IOException: Contact photo does not exist:
content://com.android.contacts/contacts/9739}
```

Snippet 5.4 - Logfile information: Start a conversation in Facebook messenger with a contact

Still about the subject, in WhatsApp application, it's possible to see exactly the same kind of information searching for the phrase "*Displayed com.whatsapp/.Conversation:*".

VoIP calls made through these applications have a lot more information, especially about the in-call audio in comparison with voice calls made from operator's network. An example of a VoIP call made through the WhatsApp app and its in-call audio information can be seen in Appendix F.

It is very interesting all the information that can be got from this source. This test scratched the surface since many of the messages presented may vary from phone to phone, but there is now a clear idea of the information that can be extracted from here and maybe for future work this information may be included in the application with automatically events on radiologs catching the percentage of dropped calls and support for well-known OTT apps, for example.

Chapter 6

Conclusions and Future Work

The present chapter serves as a conclusion of the work accomplished, analyzing all the requisites that were fulfilled and making a retrospective of the same. Some possibilities for future improvement are also addressed, notably by updating the application.

This dissertation aimed on the development of an application for Android smartphones to retrieve network's performance, QoE and QoS metrics complementing what is already done by operator's fixed and mobile probes scattered across the network.

The Android operating system was chosen to detriment of the others due to the high percentage of market share that it has, at the moment and equally important, because that market share tends to increase even more in the future.

We are very satisfied with the user interface and with the app-user interaction, having always been followed the requirements and templates given by the usability team, making this app's UI be more intuitive, simple and user-friendly.

The radiologs/snapshots feature is implemented, functional and automatically retrieving the same network events as the older ArQoS probes. The radiolog's JSON content is already structured and ready to be sent, however, it is still needed to add support in the database for saving radiologs and implement the process to send this information to the management system.

Contrariwise the test's results are already being sent to the management system with the GPS, Wi-FI and mobile network information attached. All SMS's tests were implemented and were given support in the app for test scheduling and recursion. Furthermore, following the requirements defined in section 3.1, the app handles test's execution by not allowing that two tests that need the same resources run at the same time and not allowing an on-demand test execution if a scheduled test to that time would interfere with it.

For future work, it is needed to add support for Voice tests, such as audio record and reproducing and give support to new VoLTE/VoWiFi technologies, registering by which network (mobile or Wi-Fi) an SMS or a call is sent or made, respectively.

The connection with the management system is being developed and working as supposed, for now. The automatically APN change for results delivering needs to be reviewed due to the *system app* permission. Moreover, due to time constraints, there are still requests and processes to be implemented, such as, reset the probe at management system request, define the radiologs/scanlogs time interval and configure if they are enable or not among other internal processes like alarms that ArQoS NG probes have.

Finally, other future features to be added that help with the integration of this solution in a non-user interaction scenario pass by boot the device automatically when it's charging and give the user the option of start the app automatically at the device boot.

REFERENCES

[1] **X. P. Kenechi Okeleke, M. Rogers**, The Mobile Economy 2017. 27 February 2017.

[2] “Improve Network Service Quality”, Optimize customer experience and network performance, http://www.alticelabs.com/content/products/BR_ArQoS_ALB_EN.pdf, Accessed at February 2017.

[3] **A. Jayanthiladevi, H. Premlatha, and G. Nawaz**, “Analysis study of Seamless Integration and Intelligent Solution in any situation by the Future Advanced Mobile Universal Systems 4G-(FAMOUS 4G)”, IEEE International Conference on Emerging Trends in VLSI, Embedded Systems, Nano Electronics and Telecommun.

[4] **Mora**, “2G, 3G, 4G or the evolution of mobile networks,” 2013. [Online]. Available: http://empireone.com.au/2g-3g-4g-mobile-network-evolution/. Accessed at March 2017.

[5] Qualcomm, (2014). "The Evolution of Mobile Technologies: 1G to 2G to 3G to 4G LTE" [Online], Available: https://www.qualcomm.com/documents/evolution-mobile-technologies-1g-2g-3g-4g-lte. Accessed at March 2017.

[6] GSM: Architecture, Tutorial Point, http://www.tutorialspoint.com/gsm/gsm_architecture.htm, Accessed at March 2017.

[7] 3GPP Technical Specification (2013) Numbering, Addressing and Identification, TS 23,003 v11.6.0 Section 19.6, www.3gpp.org, Accessed at March 2017.

[8] **R. Aguiar**, Apontamentos da cadeira de Redes Móveis, Universidade de Aveiro, 2016/1017

[9] Internet Resource, NEWCOM Deliverables 23.3: http://www.newcom-project.eu/images/Delivarables/D23.3-Secondreportontoolsandtheirintegrationontheexperimentalsetups.pdf

[10] **A. Dahiya**, "Evolution of Mobile Communication from 1(G) to 4G, 5G, 6G, 7G …" [Online], Available: https://www.linkedin.com/pulse/evolution-mobile-communication-from-1g-4g-5g-6g-7g-pmp-cfps. Accessed at March 2017.

[11] **G. Fettweis, M. Krondorf, S. Bittner**, GFDM—generalized frequency division multiplexing, in: 69th Vehicular Technology Conference, IEEE, 2009, pp. 1–4.

[12] **M. Mukherjee**, et al. Reduced out-of-band radiation-based filter optimization for UFMC systems in 5G, in: Wireless Communications and Mobile Computing Conference, IWCMC, 2015, pp. 1150–1155.

[13] **N. Van der Neut**, et al. PAPR reduction in FBMC systems using a smart gradient-project active constellation extension method, in: 21st International Conference on Telecommunications, ICT, 2014, pp. 134–139.

[14] **G. Wunder, S.A. Gorgani, S.S. Ahmed**, Waveform optimization using trapezoidal pulses for 5G random access with short message support, in: IEEE 16th International Workshop on Signal Proc.: Advances in Wireless Comm., 2015, pp. 76–80.

[15] Ericsson Mobility Report: On the Pulse of the Networked Society, Ericsson, 2016. Available at: https://www.ericsson.com/assets/local/mobility-report/documents/2016/ericsson-mobility-report-november-2016.pdf

[16] **F. Boccardi, R. W. Heath, A. E. Lozano, T. L. Marzetta, and P. Popovski**. Five disruptive technology directions for 5G. IEEE Communications Magazine, 52(2):74–80, 2014.

[17] **H. Elshaer, F. Boccardi, M. Dohler, and R. Irmer**. Downlink and uplink decoupling: A disruptive architectural design for 5G networks. In IEEE GLOBECOM, pages 1798–1803, 2014.

[18] Wireless World Research Forum, 2011. [Online] Available at: http://www.wwrf.ch/files/wwrf/content/files/ publications/outlook/Outlook7.pdf.

[19] **M. Mustaqim, K. Khan, M. Usman**, ”LTE advanced: Requirements and technical challenges for 4G cellular network”, Journal of Emerging Trends in *Computing* and Information Sciences, vol.3, Issue.5, pp. 665-671, May 2012.

[20] "Support system operations", http://www.alticelabs.com/pt/sistemas_suporte_operacoes.ht

ml#garantia_servico, Official AlticeLabs website, Accessed at February 2017.

[21] **H. Wang, S. Chen, H. Xu, M. Ai, and Y. Shi**, A Software Defined Decentralized Mobile Network Architecture toward 5G, IEEE Network March/April 2015.

[22] **S. Rasthofer**, "The Android Logging Service – A Dangerous Feature for User Privacy?" May 2013.

[23] **M. Rumney** (Agile Technologies), LTE and the Evolution to 4G Wireless, Design and Measurement Challenges: Wiley Publication, March 2013.

[24] *A One-Time Overview of Global 5G Initiatives as of the First Quarter of 2014*, Jun. 2014.

[25] **A. Gupta, R. Kumar Jha,** A Survey of 5G Network: Architecture and Emerging Technologies, IEEE August 7, 2015.

[26] **C. Welch**, “Before it took over smartphones, Android was originally destined for cameras" http://www.theverge.com/2013/4/16/4230468/android-originally-designed-for-cameras-before-smartphones” p. 1, 2013.

[27] IDC, “Smartphone OS Market Share, 2016 Q3.” [Online]. Available: http://www.idc.com/pro mo/smartphone-market-share/os. Accessed at May 2017.

[28] **S. Hill**, “Which smartphone OS wins 2016?” 2016. [Online]. Available: https://www.digitaltrends.com/mobile/best-smartphone-os/. Accessed at May 2017.

[29] "The Android History", https://www.android.com/versions/nougat-7-0/, Official Android website, Accessed at April 2017.

[30] **M. Mercato,** "A doce história do Android", AndroidPIT, 2015. Available at: http://www.androidpit.com.br/historia-do-android.

[31] Platform Versions, https://developer.android.com/about/dashboards/index.html Official Android website, Accessed at April 2017.

[32] **J. Hildenbrand**, "Everything you need to know about rooting your Android", androidcentral, 2016. [Online]. Available: http://www.androidcentral.com/root. Accessed at April 2017.

[33] **W. Gordon**, "Everything You Need to Know About Rooting Your Android Phone," *lifehacker*, 2013. [Online]. Available: http://lifehacker.com/5789397/the-always-up-to-date-guide-to-rooting-any-android-phone. Accessed at April 2017.

[34] **G. Sims**, "What is root," *androidauthority*, 2016. [Online]. Available: http://www.androidauth ority.com/what-is-root-680584/. Accessed at April 2017.

[35] **Sergio**, "Explaining the behavior of an Android application: System apps vs Non-System apps," *ricston*, 2013. [Online]. Available: https://www.ricston.com/blog/explaining-behavior-android-application-system-apps-nonsystem-apps/. Accessed at May 2017.

[36] **H. Q. Raja**, "How To Install Any App As A System App On Android," *addictivetips*, 2012. [Online]. Available: http://www.addictivetips.com/mobile/how-to-install-any-app-as-system-app-on-android/. Accessed at May 2017.

[36] **V. Tikhvinskiy and G. Bochechka**, ''Perspectives and quality of service requirements in 5G networks,'' J. Telecommun. Inf. Technol., no. 1, pp. 23–26, 2015.

[37] **Y. Park**, 5G Vision and Requirements, 5G Forum, Korea, Feb. 2014.

[38] 5GNOW, (5th Generation Non-Orthogonal Waveforms for Asynchronous Signaling) is a European collaborative Research Project Supported by the European Commission Within FP7 ICT Call 8, 2015.

[39] **M. Hatton**, The Global M2M Market in 2013. London, U.K.: Machina Research White Paper, Jan. 2013

[40] **E. Elkin**, "The Secret Value of VoLTE", TMCnet, April 2014.

[41] "4G VoLTE devices list in India – Updated (09-04-2017)," *newsmarkets*, 2017. [Online]. Available: http://newsmarkets.in/4g-volte-devices-in-india-updated/. Accessed at Frebuary 2017.

[42] **M. Wright**, "The Next Generation of Voice Calling Starts Today – Telstra VoLTE", September 2015.

[43] "Top 10 Questions About Kombucha & Sugar," *Kombuchakamp.Com*, 2015.

[44] "ACG Research: With VoWiFi, it's all about the economics", *Fierce WirelessTech,* Oct. 2015.

[45] "Consumer reports: cell phone voice quality put to the test", *abc7ny*, 2015. [Online]. Available: http://abc7ny.com/technology/consumer-reports-cell-phone-voice-quality-put-to-the-test/788284/. Accessed at May 2017.

[46] **E. Malykhina**, “Why Is Cell Phone Call Quality So Terrible?,” 2015. [Online]. Available: https://www.scientificamerican.com/article/why-is-cell-phone-call-quality-so-terrible. Accessed at May 2017.

[47] **M. Gikas**, “3 reasons voice quality on smart phones still sucks,” 2014. [Online]. Available: http://www.consumerreports.org/cro/news/2014/05/3-reasons-voice-quality-on-smart-phones-still-sucks/index.htm. Accessed at May 2017.

[48] **J. Hecht**, “Why Mobile Voice Quality Still Stinks—and How to Fix It,” 2014. [Online]. Available: http://spectrum.ieee.org/telecom/wireless/why-mobile-voice-quality-still-stinksand-how-to-fix-it. Accessed at May 2017.

[49] ITU-T Recommendation P.10/G.100-Amendment 2. Vocabulary for performance and quality of service. Recommendations of the ITU,Telecommunications Sector, 12 2012.

[50] **P. L. Callet, S. Möller, and A. Perkis,** “Qualinet white paper on defi- nitions of quality of experience,” in European Network on Quality of Experience in Multimedia Systems and Services (COST Action IC 1003). Wadern, Germany: Dagstuhl, 2013.

[51] **R. Schatz, T. Hoßfeld, L. Janowski, and S. Egger,** “From packets to people: Quality of experience as a new measurement challenge,” in Data Traffic Monitoring and Analysis, vol. 7754, **E. Biersack, C. Callegari, and M. Matijasevic**, Eds. Berlin, Germany: Springer-Verlag, 2013, ser. Lecture Notes in Computer Science, pp. 219–263. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-36784-7_10. Accessed at April 2017.

[52] **T. Yamazaki, M. Eguchi, T. Miyoshi and K. Yamori**, “Quality of Experience Modeling with Psychological Effect for Interactive Web Services”, 2014 IEEE Network Operations and Management Symposium (NOMS), pp.1-4, April 2014.

[53] **L. Munroe**, “Here’s how Mesosphere built a process for regular, efficient usability tests,” 2016. [Online]. Available: https://mesosphere.com/blog/2016/06/30/design-usability-tests. Accessed at May 2017.

[54] **A. Bhattacharya**, “Android just hit a record 88% market share of all smartphones,” 2016. [Online]. Available: https://qz.com/826672/android-goog-just-hit-a-record-88-market-share-of-all-smartphones/. Accessed at May 2017.

[55] GSMArena, “Xiaomi Redmi 3s.” [Online]. Available: http://www.gsmarena.com/xiaomi_redmi_3s-8150.php. Accessed at May 2017.

[56] GSMArena, “Samsung Galaxy S7.” [Online]. Available: http://www.gsmarena.com/samsung_galaxy_s7-7821.php. Accessed at May 2017.

Appendix A

5G Related Activities

Research Project / Institutions / Research Groups	Research area	HTTP location
5GNOW (5th Generation Non-Orthogonal Waveforms for asynchronous signaling)	Non-orthogonal waveforms	http://www.5gnow.eu/
5Ci PPP (5G Infrastructure Public Private Partnership)	Next generation of communication networks, ubiquitous super-fast connectivity	http://5g-ppp.cu
COMBO (Convergence of fixed and Mobile Broadband access aggregation networks)	Fixed Mobile Converged (FMC) broadband access/aggregation networks	http://www.ict-combo.eu/
iJOIN (Interworking and JOINT Design of an Open Access and Backhaul Network	RAN-as-a-Service, radio access based upon small cells, and a heterogeneous backhaul	http://www.ict-ijoin.cu/
MAMMOET (MAssive MiMO for Efficient Transmission)	Massive MIMO	http:// www.mammoct-projcct.cu
METIS (Mobile and wireless communications Enablers for Twenty-twenty (2020) Information Society)	Provide a holistic framework 5G system concept	https://www.mctis2020.com'
MCN (MobileCloud Networking)	Mobile Network. Decentralized Computing, Smart Storage	http://www.mobile-cloud-networking.eu/site/'
MOTO (Mobile Opportunistic Traffic Offloading)	Traffic offloading architecture	http://www.ict-ras.eu/index .php/ras-projects./moto
PHYLAWS (PHYsical Layer Wireless Security)	Security approaches for handsets and communications nodes	http://www.phylawvict.org/
TROPIC (Traffic Optimization by the Integration of Information and Control)	Femtocell networking and cloud computing	http://www.ict4ropic.cu/
5GrEEn	Environmentally friendly 5G mobile network	https://www.eiticlabs.cu/news-events/news/article/toward-green-5g-mobile-networks-5green-new- project- launched/#allView
University of Edinburgh	Indoor wireless communications capacity	www.ed.ac.uk
University of Surrey 5G Innovation Centre (5GIC)	Lowering network costs. Anticipating user data needs to pre-allocate resources. Dense small cells. Device-to- device communication. Spectrum sensing (for unlicensed spectrum)	http://www.surrey.ac.uk/5gic/

Table 0.1 - 5G related activities in Europe [24]

Research Project / Institutions / Research Groups	**Research area**	**HTTP location**
Berkeley SWARM Lab	Third layer of information acquisition, synchrony to the Cloud, pervasive wireless networking, novel ultra-low power technologies.	https://swarmlub.eecs.berkeley.edu/letter-executive-director#overlay context=node/5/panel_content
Berkeley Wireless Research Center (BWRC)	Radio Frequency (RF> and Millimeter Wave (mmWave) technology. Advanced Spectrum Utilization. Energy Efficient Systems and ocher Integrated Wireless Systems and Applications	http://bwrc.eecs.berkeley.edu/
Broadband Wireless Access & Applications Center (BWAC)	Opportunistic spectrum access and allocation technologies. Millimeter wave wireless. Wireless cyber security, Cognitive sensor networks of heterogeneous devices. Image and video compression technologies. 1C and low-power design for broadband access/applications	https://bwac.arizona.edu/
Center for Wireless Systems and Applications (CWSA) at Purdue University	Devices and materials. Low power electronics. Communications. Networking. Multimedia traffic. Security	http://cwsawcb.ecn.purduc.edu/
ChoiceNet Project	Architectural design for the Internet of the near future	https://code.renci.org/gf/projecl/choicen el/
Clean Slate Project at Stanford University for research on Software-Defined Networking (SDN)	Open Flow. Software Defined Networking, and the Programmable Open Mobile Internet	http://clcanslate.stanfiird.edu/
expressive Internet Architecture (XIA) Project	Internet Architecture	http://www.cs.cmu.cdu/-xia/
Intel Strategic Research Alliance (ISRA)	Enabling new spectrum, improving spectral efficiency and spectral reuse, intelligent use of multiple radio access technologies, and use of context awareness to improve quality of service and wireless device power efficiency.	https://www.intel-university-collaboration.net/focused-research
Joint University of Texas, Austin and Stanford Research on 5G Wireless	New architectures for dense access infrastructure	http://www.ccc.utcxas.edu/news/profs-dc-veciana-shakkottai-and-collaborators-received-nsf-grant-work-5g-wireles$-networks
Mobility First Project	Architecture centered on mobility	http://mobililyfirM.winlab.nilgers.edu/
Named Data Network (NDN) Project	Named Data Networking (NDN) architecture	http://namcd-data.net
NEBULA Project	Cloud computing data centers	http://nebula-fia.org
NSF Communications & Information Foundation (CIF)	secure and reliable communications	http://www.nsf.gov/funding/pgm_summ.jsp?pimsJd=503300&org=CISE
NSF Computer & Network Systems (CNS)	enterprise, core, and optical networks; peer-to-peer and application- level networks; wireless, mobile, and cellular networks; networks for physical infrastructures; and sensor networks	http://www.nsf.gov/funding/pgm_summ.jsp?pims_id=503307&org=CISE
NSF Extreme Densification of Wireless Networks	Network densification	http://www.nsf.gov/awardsearch/show Award?AWD_H>= 1343383&HistoricalAwarxls=false
NSF Future Internet Architectures (FIA) Program	Named Data Network (NDN). Mobility First. Nebula. Expressive Internet Architecture (XIA). ChoiceNet	http://www.nets-fia.net/
NSF Grant for Evaluation of 60 GHz Band Communications	Millimeter wave picocells	
Polytechnic Institute of New York University (NYU-Poly) Program	Smart and more cost effective wireless infrastructure	http://engi neeri ng. nyu .edu/
Qualcomm Institute	Robust wireless communication, multimedia communication systems, and devices for next-generation communication, wireless health	http://csro.calit2.net/proposals.html
UCSD Center for Wireless Communications	Low-power circuitry, smart antennas, communication theory, communication networks, and multimedia applications	http://cwc.ucsd.edu/rcscarch/focusarcas.php
Wireless OMIT Center	Spectrum and connectivity, mobile applications, security and privacy and low power systems	http://wireless.csail.mit.edu
Wireless Virginia Tech	Cognitive Radio Networks. Digital Signal Processing. Social Networks. Autonomous Sensor. Communication. Antennas. Very Large Scale Integration	http://wirclcss.vi.edu

Table 0.2 - 5G related activities in America [24]

Research Project / Research Groups	**Research area**	**HTTP location**
IMT-2020 PROMOTION GROUP	5G research and development	http://www.imt-2020.cn/en/introduction
MOST (MINISTRY OF SCIENCE & TECHNOLOGY) 5G PROJECT	Radio-access-network (RAN) architecture. Massive MIMO	http://www.mo8t.gov.cn/eng/programmes1/

Table 0.3 - 5G related activities in Asia [24]

Appendix B

ArQoS NG probe's task output parameters examples

Disassociate WiFi				
Parameter	**Description**	**Format Type**	**Param Type**	**Values**
1	Session duration.	Float	Seconds	Time interval
2	The total number of packets received by this interface.	Int	Packets	Packet count
3	The total number of bytes received by this interface.	Int	bytes	Traffic volume
4	The total number of packets transmitted by this interface.	Int	Packets	Packet count
5	The total number of bytes transmitted by this interface.	Int	Bytes	Traffic volume
6	Connection quality.	Int	%	0 to 100
7	Received signal strength indicator.	int	DBm	RSSI
8	Measured noise level.	Int	DBm	Noise level
9	Signal to noise ratio.	Int	dB	SNR

Table 0.4.1 - ArQoS NG probe: Disassociate Wi-Fi task output parameters

Portal login				
Parameter	**Description**	**Format Type**	**Param Type**	**Values**
1	Target URL of authentication process redirection	string	URL	URL
2	Authentication page load time	float	seconds	Time interval
3	Total portal authentication time	float	Seconds	Time interval
4	Authentication response code (form-based authentication only)	int	Numerlcal ID	http response code
5	Authentication response human readable message	string	text	text
6	Name of the file with the downloaded web portal content	string	file	Empty – no content saved File name – saved accessed web portal content
7	Name of the file with the authentication responce	string	file	Empty – no content saved File name – saved server response
8	Packet capture file name	string	file	Empty- no packet capture performed File name – packet capture was saved with the indicated name

Table 0.5.2 - ArQoS NG probe: Portal login task output parameters

Receive SMS message				
Parameter	**Description**	**Format Type**	**Param Type**	**Values**
1	End to end message delivery time.	int	seconds	time interval
2	Received SMS message text.	string	text	text
3	SMSC message timestamp.	string	text	human readable date and time
4	Sender number.	string	MSI SDN	MSI SDN
5	SMSC number.	string	MSI SDN	MSI SDN
6	End to end message delivery time.	int	milliseconds	time interval
7	Time spent waiting for the received message since the beginning of the task.	int	milliseconds	time interval
8	SMS encoding.	int	numerical ID	- default (GSM 7-bit) - 8-bit - USC2 (16-bit)

Table 0.5.3 - ArQoS NG probe: Receive SMS task output parameters

Ping				
Parameter	**Description**	**Format Type**	**Param Type**	**Values**
1	Minimum measured ICMP latency.	float	milliseconds	latency
2	Average measured ICMP latency.	float	milliseconds	latency
3	Maximum measured ICMP latency.	float	milliseconds	latency
4	Send ICMP echo request packet count.	int	packets	packet count
5	Send ICMP echo response packet count.	int	packets	packet count
6	ICMP echo packet loss.	int	packets	packet count
7	Packet capture file name.	string	file	empty - no packet capture performed file name - packet capture was saved with the indicated name
8	Destination IP address or name.	string	IP address	host name or address
9	IP address obtained from DNS lookup	string	IP address	server address
10	Operation log filename. File with additional information about the performed operation and any errors that might have occurred.	string	file	empty - no log file saved file name - log file name
11	Source IP address	string	IP address	address
12	Server name resolution time.	int	milliseconds	time interval

Table 0.5.4 - ArQoS NG probe: PING task output parameters

Appendix C

Radiolog with an event example

```
{
    "radiolog": [
        {
            "module": 0,
            "iccid": "89351060000611821081",
            "insi": "268069620647135",
            "timestamp": 1495631356,
            "mac": "861111039801587",
            "gps": "40.62965876,-8.64616829",
            "network": {
                "status": 0,
                "mode": 10,
                "cellid": "313120",
                "plmn": "MEO",
                "roaming": 0,
                "rxlevel": "-62",
                "mcc": 268,
                "mnc": 6,
                "pci": 4,
                "tac": "48040",
                "rsrp": -86,
                "rsrq": -6
            },
            "neighbours": [
                {
                    "pci": 11,
                    "rsrp": -188,
                    "rsrq": -28
                },
                {
                    "pci": 3,
                    "rsrp": -186,
                    "rsrq": -28
                }
            ],
            "event": {
                "type": 3,
                "origin": "old plan: NOWO, new plan: MEO"
            }
        }
    ]
}
```

Snippet 0.1 - Radiolog with an associated event

Appendix D

Used devices specifications

Xiaomi Redmi 3S		
NETWORK	Technology	GSM / HSPA / LTE
LAUNCH	Announced	2016, June
	Status	Available. Released 2016, June
BODY	Dimensions	139.3 x 69.6 x 8.5 mm (5.48 x 2.74 x 0.33 in)
	Weight	144 g (5.08 oz)
	SIM	Dual SIM (Micro-SIM/Nano-SIM, dual stand-by)
DISPLAY	Type	IPS LCD capacitive touchscreen, 16M colors
	Size	5.0 inches (~71.1% screen-to-body ratio)
	Resolution	720 x 1280 pixels (~294 ppi pixel density)
	Multitouch	Yes
		- MIUI 8
PLATFORM	OS	Android 6.0.1 (Marshmallow)
	Chipset	Qualcomm MSM8937 Snapdragon 430
	CPU	Octa-core 1.4 GHz Cortex-A53
	GPU	Adreno 505
MEMORY	Card slot	microSD, up to 256 GB (uses SIM 2 slot)
	Internal	16 GB, 2 GB RAM
CAMERA	Primary	13 MP, f/2.0, phase detection autofocus, LED flash, check quality
	Features	Geo-tagging, touch focus, face/smile detection, HDR, panorama
	Video	1080p@30fps, check quality
	Secondary	5 MP, f/2.2, 1080p
SOUND	Alert types	Vibration; MP3, WAV ringtones
	Loudspeaker	Yes
	3.5mm jack	Yes
		- Active noise cancellation with dedicated mic
COMMS	WLAN	Wi-Fi 802.11 b/g/n, Wi-Fi Direct, hotspot
	Bluetooth	4.1, A2DP
	GPS	Yes, with A-GPS, GLONASS, BDS
	Infrared port	Yes
	Radio	FM radio
	USB	microUSB 2.0
FEATURES	Sensors	Accelerometer, gyro, proximity, compass
	Messaging	SMS(threaded view), MMS, Email, Push Mail, IM
	Browser	HTML5
	Java	No
		- Fast battery charging

		- DivX/Xvid/MP4/H.264 player - MP3/WAV/eAAC+/FLAC player - Photo/video editor - Document viewer
BATTERY		Non-removable Li-Ion 4100 mAh battery
MISC	Colors	Gold, Dark Gray, Silver
	SAR EU	0.62 W/kg (head) 0.43 W/kg (body)
	Price	About 120 EUR
TESTS	Performance	Basemark OS II: 882 / Basemark OS II 2.0: 882
	Display	Contrast ratio: 1087:1 (nominal), 2.913 (sunlight)
	Camera	Photo / Video
	Loudspeaker	Voice 66dB / Noise 70dB / Ring 70dB
	Audio	Noise -94.3dB / Crosstalk -91.8dB
	Battery life	Endurance rating 104h

Table 0.5 - Xiomi Redmi 3S specifications [55]

Samsung Galaxy S7		
NETWORK	Technology	GSM / HSPA / LTE
LAUNCH	Announced	2016, February
	Status	Available. Released 2016, March
BODY	Dimensions	142.4 x 69.6 x 7.9 mm (5.61 x 2.74 x 0.31 in)
	Weight	152 g (5.36 oz)
	Build	Corning Gorilla Glass 4 back panel
	SIM	Single SIM (Nano-SIM) - G930F Dual SIM (Nano-SIM, dual stand-by) - G930FD
		- Samsung Pay (Visa, MasterCard certified) - IP68 certified - dust/water proof over 1.5 meteres and 30 minutes
DISPLAY	Type	Super AMOLED capacitive touchscreen, 16M colors
	Size	5.1 inches (~72.1% screen-to-body ratio)
	Resolution	1440 x 2560 pixels (~577 ppi pixel density)
	Multitouch	Yes
	Protection	Corning Gorilla Glass 4
		- Always-on display - TouchWiz UI
PLATFORM	OS	Android 6.0 (Marshmallow), upgradable to 7.0 (Nougat)
	Chipset	Exynos 8890 Octa
	CPU	Octa-core (4x2.3 GHz Mongoose & 4x1.6 GHz Cortex-A53)
	GPU	Mali-T880 MP12
MEMORY	Card slot	microSD, up to 256 GB (dedicated slot) - single-SIM model (G930F, G930W8) microSD, up to 256 GB (uses SIM 2 slot) - dual-SIM model (G930FD)
	Internal	32/64 GB, 4 GB RAM
CAMERA	Primary	12 MP, f/1.7, 26mm, phase detection autofocus, OIS, LED flash

	Features	1/2.5" sensor size, 1.4 µm pixel size, geo-tagging, simultaneous 4K video and 9MP image recording, touch focus, face/smile detection, Auto HDR, panorama
	Video	2160p@30fps, 1080p@60fps, 720p@240fps, HDR, dual-video rec
	Secondary	5 MP, 1/4.1" sensor size, 1.34 µm pixel size, f/1.7, 22mm, dual video call, Auto HDR
SOUND	Alert types	Vibration; MP3, WAV ringtones
	Loudspeaker	Yes
	3.5mm jack	Yes
		- 24-bit/192kHz audio - Active noise cancellation with dedicated mic
COMMS	WLAN	Wi-Fi 802.11 a/b/g/n/ac, dual-band, Wi-Fi Direct, hotspot
	Bluetooth	4.2, A2DP, LE, aptX
	GPS	Yes, with A-GPS, GLONASS, BDS
	NFC	Yes
	Radio	No
	USB	microUSB 2.0, USB Host
FEATURES	Sensors	Fingerprint (front-mounted), accelerometer, gyro, proximity, compass, barometer, heart rate, SpO2
	Messaging	SMS(threaded view), MMS, Email, Push Mail, IM
	Browser	HTML5
	Java	No
		- Fast battery charging (Quick Charge 2.0) - Qi/PMA wireless charging (market dependent) - S-Voice natural language commands and dictation - OneDrive (115 GB cloud storage) - MP4/DivX/XviD/WMV/H.264 player - MP3/WAV/WMA/eAAC+/FLAC player - Photo/video editor
BATTERY		Non-removable Li-Ion 3000 mAh battery
	Talk time	Up to 22 h (3G)
	Music play	Up to 62 h
MISC	Colors	Black, White, Gold, Silver, Pink Gold
	SAR	1.40 W/kg (head) 1.59 W/kg (body)
	SAR EU	0.41 W/kg (head) 0.62 W/kg (body)
	Price	About 500 EUR
TESTS	Performance	Basemark OS II: 2004 / Basemark OS II 2.0: 2128
	Display	Contrast ratio: Infinite (nominal), 4.376 (sunlight)
	Camera	Photo / Video
	Loudspeaker	Voice 69dB / Noise 69dB / Ring 71dB
	Audio	Noise -92.5dB / Crosstalk -92.7dB
	Battery life	Endurance rating 80h

Table 0.6.2 - Samsung Galaxy S7 specifications [56]

Appendix E

Logfile's VOIP call information

```
I/ActivityManager: Displayed
com.whatsapp/.VoipActivity: +254ms

Audio: D/audio_hw_primary: adev_open_input_stream:
enter: sample_rate(16000) channel_mask(0x10)
devices(0x80000004)
stream_handle(0xab5adf68) io_handle(772)
source(7)

D/compress_voip:
voice_extn_compress_voip_pcm_prop_check: VoIP PCM
property is enabled

D/compress_voip:
voice_extn_compress_voip_open_input_stream: enter

D/compress_voip: voip_set_mode: enter, format=1

D/compress_voip: voip_set_mode: Derived mode =
12

I/AudioFlinger: AudioFlinger's thread 0xef070008
ready to run

D/audio_hw_primary: in_standby: enter: stream
(0xab5adf68) usecase(27: compress-voip-call)

W/AudioFlinger::EffectModule: EffectModule
0xab54ad00 destructor called with unreleased
interface

W/AudioFlinger::EffectHandle: disconnect Effect handle
0xab523ae0 disconnected after thread destruction

D/audio_hw_primary: adev_close_input_stream:
enter:stream_handle(0xab5adf68)

D/compress_voip:
voice_extn_compress_voip_close_input_stream: enter

D/compress_voip: voip_stop_call: enter,
out_stream_count=0, in_stream_count=0

E/compress_voip: voip_stop_call: Could not find the
usecase (27) in the list

E/audio_hw_primary: adev_close_input_stream:
Compress voip input cannot be closed, error:-22

I/WhatsAppJni: EnsureThreadAttached: attached
current thread to JVM

D/audio_hw_extn: audio_extn_get_parameters: returns

I/str_params: key: 'audio_mode' value: ''

D/audio_hw_extn: audio_extn_get_parameters: returns

I/str_params: key: 'voip_out_stream_count' value: ''

D/audio_hw_extn: audio_extn_get_parameters: returns

I/str_params: key: 'voip_sample_rate' value: ''

D/AudioPolicyManagerCustom: Set VoIP and Direct
output flags for PCM format

I/AudioFlinger: openOutput(), module 1 Device 1,
SamplingRate 16000, Format 0x000001, Channels
1, flags 801

D/audio_hw_primary: adev_open_output_stream: enter:
sample_rate(16000) channel_mask(0x1)
devices(0x1) flags(0x801)
stream_handle(0xab4677a0)

D/compress_voip:
voice_extn_compress_voip_pcm_prop_check: VoIP PCM
property is enabled

D/compress_voip:
voice_extn_compress_voip_open_output_stream: enter

D/compress_voip: voip_set_mode: enter, format=1

D/compress_voip: voip_set_mode: Derived mode =
12

D/audio_hw_primary: adev_open_output_stream:
Stream (0xab4677a0) picks up usecase
(compress-voip-call) ...

I/AudioFlinger: AudioStreamOut::open(),
mHalFormatIsLinearPcm = 1

I/AudioFlinger: HAL output buffer size 320 frames,
normal sink buffer size 320 frames

I/AudioFlinger: AudioFlinger's thread 0xab539d50
ready to run

V/AudioPolicyManagerCustom: getOutput() returns
new direct output 776

V/SRS_Proc: ParamSet string: bluetooth_enabled=0

V/SRS_ProcWS: SRS_Processing - SourceOutAdd -
No Available Slot for 0xab539d50

D/audio_hw_primary: out_standby: enter: stream
(0xab4677a0) usecase(27: compress-voip-call)

D/audio_hw_primary: out_standby: Ignore Standby in
VOIP call

W/AudioRecord: AUDIO_INPUT_FLAG_FAST denied
by client; transfer 1, track 16000 Hz, primary
48000 Hz

D/audio_hw_primary: adev_open_input_stream: enter:
sample_rate(16000) channel_mask(0x10)
devices(0x80000004)
stream_handle(0xab55a8b8) io_handle(778)
source(7)

D/compress_voip:
voice_extn_compress_voip_pcm_prop_check: VoIP PCM
property is enabled

D/compress_voip:
voice_extn_compress_voip_open_input_stream: enter

D/compress_voip: voip_set_mode: enter, format=1

D/compress_voip: voip_set_mode: Derived mode =
12

I/AudioFlinger: AudioFlinger's thread 0xef070008
ready to run
```

Snippet 0.2 - Logfile information: WhatsApp VoIP call information