Developing a Geo-positioning C# Framework for Radio Network Optimization based on 3G Network Recordings

Rúben Borralho
ruben.borralho@ist.utl.pt

Instituto Superior Técnico, Lisboa, Portugal

May 2017

Abstract

With constant traffic increase in cellular networks, it became demanding and complex for operators to manage and monitor its networks in order to provide desirable Quality of Service (QoS) to its users. Due to such situation, this project appears as a solution to costs reduction in terms of network monitoring and management. The objective is to produce and implement a framework that collects protocol events exchanged within network elements, stored at the Radio Network Controller (RNC), called traces, gathering useful information from them and present results relative to network performance. To achieve this goal, a C# application was developed encompassing all procedures since traces collection to network performance metrics visualization. It also makes use of a Geolocation algorithm in order to attribute latitude and longitude coordinates to traces and so, to be possible associate network performance metrics to specific locations. The program was developed under certain high quality parameters, following Extreme Programming concepts as well as S.O.L.I.D. Principle. The first one takes to code development with bugs occurrence minimization and early errors detection. Unit Tests usage and Code Review by different developers contributes to such scenario. S.O.L.I.D. Principle complements the benefits of Extreme Programming and also encompasses a set of practices which takes to the production of an organized code, well structured, modular, allowing new features addition easily, and comprehensive to future usage by other developers. A deep study of traces was done in order to know which is the relevant information contained on them. A set of use cases was initially defined and respective results were obtained to radio frequency metrics in terms of coverage, with RSCP or Ec/N0 levels, and also dropped and blocked calls location. Comparisons of such results were then performed, using drive tests as reference. In one of these comparisons, specifically in Tapada das Necessidades, it was verified a difference between average RSCP value received of just 1.3 dB between traces and drive tests.

Keywords: QoS, SON, Extreme Programming, S.O.L.I.D. Principles.

1. Introduction

1.1. Motivation

Quality of Service (QoS) providing is one of the major concerns from telecommunications operators nowadays. Users increasing demand, growing traffic or usage of more powerful applications are aspects to be fulfilled. In order to know if their network performance is within reference parameters, measures of its performing must be done. Methods allowing these measures to be done in an effective and cost reduced way are valuable. Typical practices, like drive tests, require large operation expenditure (OPEX). Moreover, with the increasing number of network parameters to monitor and set, the network extension with the augmented number of base stations and the parallel operation of technologies (Second Generation (2G), Third Generation (3G) and Long Term Evolution (LTE)) take this expenditure to unsustainable values. This factor motivates the search of alternative methods that allow to reduce network monitoring costs and increasing the range of monitored areas. It is here where Self-organizing Networks (SONs) concept came up. It is a collection of functions for automatic configuration, optimization, diagnosis and healing of the network in a remote manner. One way of implement solutions based on SON concept is the collection of protocol messages exchanged within terminal and network elements which contain valuable information about network conditions experienced by users. This type of solutions increase effectiveness in network management, widen monitored areas and reducing operational costs.

1.2. Objectives

There are two main objectives regarding this work. Firstly, to analyze Universal Mobile Telecommunications System (UMTS) layer 3 protocol messages to understand which relevant information can be taken from them, in order to acquire network performance metrics. Then, it is suppose to create a C# application, capable of collect identified data from protocols, producing use cases visualization in terms of network performance. This enters directly with SON concept, in a way that allows the remote analysis of the network performance, in real-time, in a wide area and...
2. Background

This section aims to supply information about the environment where this project is inserted.

2.1. Network Performance Data Collection

There are several methods used to collect data from networks in order to understand not only if the network is providing the required and established quality to the users, but also to help in its planning and optimization.

2.1.1 Performance Management

The Performance Management (PM) involves evaluation and reporting of network elements behavior and effectiveness, by gathering statistical information, maintaining and examining historical logs, determining system performance and altering the system modes of operation [2]. It includes measurements of network and applications traffic in order to provide a consistent and predictable level of service at a given instance and across a defined period of time. Accordingly, PM involves network monitoring, applications and service activity and design and configuration adjustments.

KPIs

Key performance indicators (KPIs), in telecommunication networks, are measures of the network performance. They result from statistical calculations based on counters installed along network elements (PM data), that can register, among many other indicators, the number of voice calls performed as well as data calls, blocked calls, dropped calls, handover types or failed handovers.

KPIs represent a crucial management factor in telecommunication networks, allowing to get an overview of the entire network performance and quality, identifying performance gaps between current and desired performance and providing indications concerning the progress of closing those gaps [3].

There are three types of KPIs, RATIO KPI reflecting the percentage of a specific case occurrence to all cases, MEAN KPI reflecting a mean measurement value based on a number of sample results and the CUM KPI which represents a cumulative measurement which is always increasing [4].

2.1.2 Configuration Management

The Configuration Management (CM) provides the operator with the ability to assure correct and effective operation of the 3G network as it evolves. It encompasses control and monitoring mechanisms to verify the actual configuration of network elements and resources. These actions may be initiated by the operator or by functions in the Operations System and can be performed as part of implementation programmes, optimisation programmes and to general QoS maintenance [5]. The CM service components are the System Modification Service component and System Monitoring Service component. The first one is an action performed to introduce new or modified data into the system due to optimisation or configuration. The second referenced component, provides the operator with the ability to receive reports on the configuration of the entire network, or parts of it, from managed Network Elements. In terms of CM functions, they encompasses operator assistance in making the most timely and accurate changes, ensure that CM actions will not result on secondary effects, traffic should be protected from effects of CM actions and there must exist mechanisms to overcome data inconsistencies [6].

2.1.3 Drive Tests

Drive test is a test performed in cellular networks and consists on network data collection on a moving vehicle. It provides an accurate real-world capture of the Radio Frequency (RF) environment under a particular set of network and environmental conditions. The collected data depends on the needs, and it could be relative to voice or data communications as well as coverage analysis and RF metrics, checking for example the interference level. The most common reasons to do drive tests are the network performance analysis, integration testing of new sites, changes on sites parameters verification, marketing and benchmarking. The hardware required for a drive test is a notebook with specific installed software, at least one mobile phone and a Global Positioning System (GPS) device. Although being of great value due to its reliability, drive tests brings huge costs to the mobile operators in terms of both equipment and manpower.

2.1.4 Traces

When connected to a Node B from a certain RNC, a User Equipment (UE) is constantly exchanging data with these network elements, either to establish any type of communication or to inform about its communication conditions. All this data is collected and logged at the RNC from that area and represents a powerful feature to analyze and monitor network performance. This data is called traces, and they are...
layer 3 protocol events resulting from communication between RNC and network elements attached to it, UE (via Node B), Node B, other RNCs and Core Network (CN).

Traces represent a huge amount of information, providing really useful data of the communications quality. As an example, they contain measurements made by UEs relative to the quality of signal that they are receiving with the storage of RSCP or Ec/N0s values, or information relative to cells to which they are connected, with Scrambling Codes (SCs) and Active Set Cell IDs information. It includes also data concerned to UE identification (IMISIs), Propagation Delays and Time Spans, among others. A dropped or blocked call, as well as its causes, is reported and stored at RNC in the form of traces. Additionally, by processing and crossing traces information it is possible to reach the location of an UE when it sends or receives any event/message. This factor represents a very powerful tool where it is possible to associate network events to specific locations, monitor a route made by a generic user, checking network parameters during its movement or even know which kind of device has better performance in both specific and general situations.

2.2. GPEH Traces Collector Feature

The developed work is based on a traces data collection feature, specifically General Performance Event Handling (GPEH) from Ericsson. This tool is placed at the RNC and collects all the protocols received and sent by it as well as the ones generated internally.

![Figure 1: GPEH location at RNC.](image)

The Ericsson feature is used to create and log events in a 15 minutes Result Output Period (ROP), i.e., 15 minutes of all layer 3 signalling exchanged in the network belonging to that RNC [7]. A ROP is divided in a set of files, named Main Processors (MPs), with all communication events exchanged in the network on a specific area, covered by a RNC. The events on the MPs are stored in parallel, i.e., each MP file has events from minute zero to fifteen and not sequentially. The first and last files contain specific information and not the normal sequence of events. While the first one works as a reference file, last file contains INTERNAL, IMSI events, representing the UEs that performed communication on that area, even if it was only signalling. For the same IMSI (UE) it could exist more than one INTERNAL, IMSI event in last file, depending on the activity of the UE. If a UE established two phone calls in a 15 minutes period belonging to the same ROP, its last file will contains at least two INTERNAL, IMSI events for that UE, with same IMSI but different triggering time.

All other events are stored and distributed through the remaining files but not explicitly associated to a UE.

2.3. GPEH - RNC and Protocols

Ericsson’s GPEH feature is strategically placed at RNC because of the functions performed by this network element from which useful information related to signalling and data transmission over the radio interface can be taken. Some of these functions encompasses Call Admission Control, where the RNC calculates the current traffic load for each individual cell and decides whether the interference level is or not acceptable, rejecting the call in negative case (blocked call), radio links set-up or release and maintenance or Power Control for the efficient operation of a CDMA network, where the transmitted power of all users is controlled. Handover is also managed by the RNC which uses measurement values supplied by Node B and UE to decide whether a different cell is better suited for a current connection.

The information of each of these processes is reported via layer 3 protocols, specifically, the ones described below:

- **RRC** - The RRC sub-layer controls the signalling between UE and UTRAN providing signalling transfer services to higher layers [8]. Signalling messages are encapsulated within RRC messages for transmission over the radio interface; on the UTRAN side, the RNC configures the broadcast channels that are broadcast in each cell and, on request, it sets up the radio bearers over which applications can exchange data between UE and UTRAN.

To carry out all its tasks, the RRC layer collects measured values from all other layers, in order to generate configuration instructions for them, using suitable algorithms. So, it is possible to sum-up its services in Broadcast, Paging and Notification as well as Dedicated Control services.

- **NBAP** - is another control signalling protocol, which controls the interface resources and providing the means for Node Bs and RNC to communicate. One peer entity of the NBAP resides at the Node B and the other at the RNC, which controls the Node B. It also has an interface with the RRC protocol to provide RRC, the necessary information to operate. NBAP main functions are summed up as follows:
– Advertises RNC about changes that physical procedures performed at the Node B level, could influence the conditions of the logical resources owned by RNC.

– Establishes and maintains a control connection to initiate set-up and release of dedicated user plane connections, i.e., connections to user data transmission.

– Provides Node B with ability to report failure or restoration of a transmission on radio links.

- **RANAP** - is the only signalling protocol defined between the UTRAN and CN and it is used by both CS and PS technologies to access the services provided by UTRAN. It controls the resources between UTRAN and UE, being on top of this interface signalling transport layers. One RANAP entity resides on the RNC and the other in the MSC or SGSN (CN elements). It provides the means for the CN to control the establishment, modification and release of the Radio Access Bearers (RABs) between UE and CN. When analyzed its messages, it could be found information relative to relocation of Serving RNC (SRNC) due to UE mobility or set up and release of its connections. Paging UEs, signalling due to mobility management and UEs tracing are also features of RANAP protocol.

- **RNSAP** - is a control plane protocol providing control signalling across the interface between RNCs. It is executed by two RNCs where one takes the role of SRNC, that handles the connection to the UE and another acts as Drift RNC (DRNC) which may borrow resources from a certain cell when asked by the SRNC. The RNSAP is responsible for bearer management signalling and is used to set up radio links, allowing the SRNC to control those radio links using dedicated resources in a Drift RNS (DRNS). Thus, inside this protocol messages, it can be found information relative to radio and mobility signalling between RNSs, including support of handover, radio resource control and synchronisation. Summed up, radio link set-up, addition or deletion as well as measurement reporting and global resources management through information exchange between RNCs, are the main functions of this protocol.

- **SABP** and **PCAP** are also layer 3 protocols from which messages are collected by GPEH feature but with less relevance in this work. However, as a brief insight, SABP protocol deals with the communication between RNC and Cell Broadcast Centre (CBC) domain from CN. So it defines cell broadcast information that is transmitted to UEs [9]. PCAP protocol deals with communication between RNC and Stand-Alone SMLC (SAS). SAS is a network element, that can be integrated in the RNC, which function is to handle positioning measurements and calculation of UEs position. This way, PCAP provides signalling services between RNC and SAS [10].

2.4. **GPEH - Event Types**

According to the protocols described above, GPEH feature presents two type of events, the **RNC Internal Events** and the **Inter-node Events**. **RNC Internal Events** are generated internally at specific triggering conditions for each event due to either measurements coming from nodes and from UE or RNC algorithm events. As example, the INTERNAL_ IMSI event already referred, is triggered every time a new UE establish any type of connection in the RNC coverage area. **Inter-node Events** refers to layer 3 described protocols. Accordingly, these events can belongs to the RRC, NBAP, RANAP, RNSAP, PCAP and SABP protocols and only information established by GPEH is hold from them, allowing to record specific information as well as the entire protocol message content.

Besides this distinction there common fields to all of them, like the event name and identification.

3. **Implementation**

This project implementation encompasses the two main objectives described in Section 1.3 Some use cases were also defined, which results achievement was a priority. These use cases are the cell footprint that is a representation in terms of RSCP level of the areas covered by a cell and best server, where it is aimed to show the areas were a cell is best server, i.e., where most of the users is served by that cell. The Best Server use case should also be acquired in terms of RSCP. Dropped calls detection and location is also a proposed use case.

3.1. **Network Events Selection**

Cell footprint and best server use cases, concern on RF metrics, specifically the RSCP. To get them, it was detected an event, the **RRC_MEASUREMENT_REPORT**, from the RRC protocol, that is sent by UEs with this metric. Since both situations depends on RSCP, the **RRC_MEASUREMENT_REPORT** event allows the production of both use cases. This event has also important fields that must be collected in order to use as input to the OTD Geo-positioning Algorithm.

To identify Dropped Calls a **RNC Internal Event** can be used, the **INTERNAL_SYSTEM_RELEASE**. This event is triggered every time one or several RABs or standalone **RRC_CONNECTION_RELEASE** (RRC protocol message to disconnect an UE) appears and the release cause is not a normal one.

With such events identification, it is possible to develop an application that will take the relevant infor-
ation from them and produce the desirable network performance results.

3.1.1 Events IMSI Identification

The events identification and association by user, is done by IMSIs attribution, contained in last ROP file, to events that do not contain this information. In order to achieve this objective, it is important to understand how this event is generated within the network. INTERNAL, IMSI events are triggered when an IMSI is associated to a new UE_ CONTEXT (UE identifier in the area), at the reception of RANAP_COMMON_ID after a successful RRC connection setup or at the reception of an IMSI in a RANAP_RELOCATION_REQUEST. The existence of common fields and in much of the time unique, between INTERNAL, IMSI events and the others, like UE_ CONTEXT and RNC_MODULE_ID can also help on this association.

3.1.2 Phone Call Organization

Phone calls and other types of communications definition, is also of interest in order to trace a user, observing the communication conditions during that period. This procedure is initiated with the RRC protocol where an RRC connection establishment is initiated by the UE that sends a RRC_CONNECTION_REQUEST to the RNC. After this, the RNC sends a RRC_CONNECTION_SETUP and, if everything is as it is supposed to, the procedure continues with the message RRC_Connection_SETUP_COMPLETE, concluding the RRC connection establishment.

To release a connection, the signalling RRC release procedure is used. This procedure is initiated at the RNC side, which sends the message RRC_CONNECTION_RELEASE when the UE state is in agreement with such situation. When the release message is received by the UE it answers with the message RRC_CONNECTION_RELEASE_COMPLETE and the procedure finishes with the UTRAN releasing all the UE dedicated resources.

Another factor that can be used to a Phone Call identification, after a successful IMSI attribution to events, is the existence of three unique fields within events belonging to a same communication. These fields are UE_CONTEXT, RNC_MODULE_ID and IMSI that have equal values within events from a same communication and cannot be repeated in different ones, at the same time.

3.2 Application Development

The developed application encompasses three major steps. A parsing component where selected traces are read and stored in suitable data structures, a data flux within OTD Geo-positioning Algorithm components, supplying all needed data for them to work and a final binning process that will allow a comprehensive geographical visualization of the results. Also to highlight, the application was developed taking into account the already mentioned practices and principles in order to produce an high quality code.

3.2.1 Programming Practices and Principles

Starting by the steps to achieve an high quality development, Extreme Programming (XP) and S.O.L.I.D. Principle were followed.

Extreme Programming includes Test Driven Development (TDD) process in which three activities are tightly interwoven: coding, testing, in the form of unit tests and design, in the form of refactoring. Unit testing refers to a program developed to exercises some specific part of the source code in order to assure that the expected result from it is being achieved; refactoring consists on improving the internal structure of an existing program source code, while preserving its external behavior. XP also covers Code Review and Pair Programming concepts. The first is a practice where after a developer finishes his work another one review the code. This review consists on searching for bugs or errors, bad code structure, design and tests reliability. Pair Programming involves having two programmers working at a single machine. One developer operates the keyboard while the other watches, learns, asks, talks, make suggestions and tries to help on bug avoidance or finding.

While Extreme Programming mainly focus on errors and bugs avoidance or early detection, S.O.L.I.D. Principle, besides also contributes for that factor, relies on good design and structure of the code. The principal concern here is to have a decoupled and modular code, by facilitating code changes as well as features addition. It is also important to facilitate the comprehension by another developers, when reading or working with the code. S.O.L.I.D. Principle is in fact composed by a set of principles that allows to achieve those targets: Single Responsibility Principle, Open-Closed Principle, Liskov Substitution Principle, Interface Segregation Principle and Dependency Inversion Principle [III].

3.2.2 Application Components

Data Structures Definition

In order to organize all the information collected from GPEH, it is important to define well suit data structures to store that information. Four different data structures, were initially established. All events are considered TraceEvent data structures, where only common information to all of them was stored. From this class another two were derived with more specific information. The NbapTraceEvent and RrcMeasureTraceEvent data structures contain specific information from its respective events, NBAP_RADIO_LINK_SETUP, REQUEST and RRC MEASUREMENT_REPORT. The information collected here is crucial to the OTD Geo-positioning algorithm and is joined together in another data structure, the fourth
one, called MeasurementReport. So, TraceEvent and MeasurementReport are the main classes and implement the interface IPoint, in order to "obligates" the existence of latitude and longitude coordinates fields, and the interface ITimmedPoint to the triggering event time field existence.

![Diagram](image)

**Figure 2:** Data Structures Hierarchical Organization.

**IMSI Addition Component**

To allow events identification it was followed the knowledge acquired in Section 3.1.1. To accomplish this task Internal IMSI events were previously loaded from ROP and stored. Then, per each event, the following process is done: from IMSI repository, the events with RNC_MODULE_ID and UE_CONTEXT fields equals to the ones of the received event are chosen. After this, time relations were used to correctly add the IMSI to the event.

**Phone Call Definition Component**

In this component, since events already have IMSI at this stage, it was used the triple identifier approach, with the comparison between IMSI, UE_CONTEXT and RNC_MODULE_ID fields. This way, all events with same values for this triple were stored together, as a phone call (or any other communication type). This definition does not need begin or end call events. When, for a sequence of events belonging to a certain IMSI, there is a change in the UE_CONTEXT or RNC_MODULE_ID fields, it means that a different communication was established by the UE of that IMSI. With the creation of groups of events with same IMSI, RNC_MODULE_ID and UE_CONTEXT, different phone calls were being collected with the desired UE identification.

**Geolocation Algorithm Integration**

The integration of a Geolocation algorithm is an important step on this work, since it will allow to associate performance metrics to locations. An OTD based Geolocation algorithm developed at Celfinet was used [12]. This algorithm is composed by three components, a Geolocation component that works and locates only MeasurementReport instances, and two other components, Smooth and Interpolation, that will help to provide the location to all other events.

So, the first step is to feed the Geolocation component with the MeasurementReport objects previously created. After processed, two sets of MeasurementReports are obtained, the located and non-located ones. Then it is applied the Smooth component to the positioned MeasurementReports to erase time and velocity inconsistencies, avoiding absurd or incoherent locations. The, all events location is got, by interpolating the positioned and smoothed MeasurementReports with all the other non-positioned events. After this process, it does not make sense to continue working with different types of data structures and accordingly, a common one was used from this step on, called NetworkEvent.

**Binning Component**

The Binning process is a way of work the information contained in the processed and located traces, with the objective of improve the results geographical visualization quality. If all events from ROP were independently represented in a graphical environment, in order to see its location in a map, it would be a huge mess, since there would be too many points. So, Binning represents a much more efficient way of doing this representation. This method stands in grouping events from a certain area and applying a mathematical method (median, mean...) to attribute a parameter value (depending on the use case to be observed) to that region based on that parameter value in each event.

The objective is to divide the map into a grid. The squares of the grid have a certain area, defined by the user. When the Binning method is applied to events, it converts them latitude and longitude coordinates into Cartesian ones, indicating in which square of the grid the event belongs. Let us imagine that there are six events with the following RSCP values: Event 1: -110 dBm, Event 2: -90 dBm, Event 3: -115 dBm, Event 4: -70 dBm, Event 5: -85 dBm and Event 6: -90 dBm, which respective location is as shown in Figure 3.

If the Binning method is the median, Events 1 and 2 will originate a median value of -100 dBm, Event 3 of -115 dBm and Events 4, 5 and 6, a median of -85 dBm. The result would be as shown in Figure 4.
4. Results

After concluded all the processes, a platform developed in Cellinet, called Labs, was used to geographical representation of the application outputs. It is important to highlight that the binning area used corresponds to the entire world as a grid, where each square represents an area of 50x50 m$^2$. The data used to achieve these results is collected from a generic RNC responsible for Node Bs at Lisboa city. It was collected on April 30, 2015 within 11.30h and 12.15h representing three ROPs.

Although obtained various use cases visualization, here will be focused the ones concerned with the signal level in terms of RSCP.

4.1. Use Cases Visualization

Using the referred visualization platform and the results from application processing, cell footprint and best server, in terms of RSCP, use cases were produced, as shown in Figures 5 and 6 respectively.

4.2. Use Cases Results Validation

Drive tests were used as comparison since represents one of the most reliable data from network performance. However, use cases traces based validation is a complex process and difficult to achieve. The number of processed ROPs due to lack of available data, representing only 45 minutes of communications in a generic day, cannot be representative of the real scenario that is verified in Lisbon city. It must also be considered that the OTD based Geo-location Algorithm used to achieve the events location has an associated positioning error as shown in Table 1. Drive tests also have a problem due to the fact that they rely on outdoor environment metrics collection, while traces data is from both outdoor and indoor environments. So, even when huge differences are found between the comparison of these two types of data, it does not mean that, in this case, traces processing went wrong. To minimize this problem, it was considered that 80% [13] of the traffic is indoor and in general, it represents an average attenuation of 10 dBm [14] when compared with drive tests data (outdoor).

<table>
<thead>
<tr>
<th>Results</th>
<th>OTD Geo-location Algorithm Error [m]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geolocation Median</td>
<td>249.2</td>
</tr>
<tr>
<td>Error [m] Mean</td>
<td>290.8</td>
</tr>
<tr>
<td>Filter Error Median</td>
<td>216.1 ($\sim$-13.3%)</td>
</tr>
<tr>
<td>[m] Mean</td>
<td>295.6 ($\sim$-19.6%)</td>
</tr>
</tbody>
</table>

Qualitative and quantitative analysis were performed from certain Lisbon areas, most of them gar-
dens, in order to avoid as much as possible build-
ingings and hence, indoor traffic. The best results were
achieved in Jardim da Tapada das Necessidades. In
terms of qualitative analysis, in Figure 7 it is possible
to see some similarities (although also some dif-
fences) between traces and drive tests trough the
garden. It was noticed some signal degradation in cer-
tain areas in both cases but in general, traces seems
to report higher signal degradation.

Figure 7: Best Server RSCP comparison between
traces (left) and drive tests (right) in Jardim da Tapada das Necessidades.

In terms of quantitative analysis, RSCP values of
all binning samples present in the images in the com-
parison between drive tests and traces in Jardim da Tapada das Necessidades, were used. Cumula-
tive Distribution Functions (CDFs) were produced to
traces, drive tests and drive traces with indoor attenu-
ation. Figure 8 shows the RSCP trends for the three
scenarios. As expected, the difference between drive
tests and traces is higher when indoor attenuation is
not considered. On the other hand, when used the ap-
proximation with indoor attenuation, the results are
really close. The average RSCP level difference is of
1.3 dB - Table 2 - between the two data types. This
is a good result but how it was explained, more data
should be processed from traces, from an extended
period, in order to be completely sure about the re-
sults.

Figure 8: RSCP level CDFs from drive tests, drive
tests with indoor attenuation and traces in Jardim da Tapada das Necessidades.

Table 2: Comparison between Jardim da Tapada das Necessidades traces and drive tests median and average RSCP error values.

<table>
<thead>
<tr>
<th>Error Comparison between Traces and Drive Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>Median</td>
</tr>
<tr>
<td>Average</td>
</tr>
</tbody>
</table>

4.3. Algorithm Analysis

4.3.1 Processing Time Optimization

From the components developed during this project,
Parser is probably the one with more impact in terms
of processing time, which was initially more than 1.30
h. Some optimization was done reducing this time to
14 minutes. Phone call definition through the usage
of a unique triple, UE CONTEXT, RNC MODULE,
ID and IMSI was a faster process when compared with
the RRC protocol messages sequence analysis, which
is a more complex method. Substitution of List by
Dictionary data structures, which presents a O(1) ac-
cess complexity against O(N) in Lists, was another
important optimisation. Early events filtering as well
as its information, avoiding useless processing, hard-
disk drive accesses reduction and increasing integers
usage instead of strings, also contributed to such re-
sult.

4.3.2 Succeed Applied Methodologies

All practices already mentioned in terms of code de-
velopment and of course, a consciousness of the con-
cepts from S.O.L.I.D. Principle, which were carefully
followed, allowed to achieve an high quality code as
was desirable.

SonarQube is a tool that allows to measure the code
quality, providing four metrics to this evaluation. It
helps to understand if S.O.L.I.D. Principle practices
and also some of Extreme Programming ones were
correctly applied. Such metrics have to do with code
prone to bugs appearance, how vulnerable to outside
threats it is, the code coverage in terms of unit tests
and Code Smells. Code Smells is a quality metric
that allows to understand how much degraded is the
code becoming in terms of complexity, confusion and
bad structure. These factors, when not fulfilled can
also originate problems and bugs. Examples of such
degradation are the introduction of duplicated code,
uncovered code and too big classes. Such tool results
are shown in Figure 9.

In terms of Code Coverage, besides not shown, a
very good value of 83% of coverage, was obtained.
The resulting metrics presented in Figure 9 are great
results. For example, the 14 vulnerabilities have all to
do with “Console.WriteLine” commands, whose func-
tion is print characters in the terminal. It is an op-
eration that can be easily erased and its use is most
of the times for debugging. In terms of Code Smells, 174 of them were detected 174, which implies a debt of 4 days. This means that the changes to perform in order to erase their existence has a estimation of completion of 4 days. It could seems too much time, but taking into account that this is a 8500 lines code and comparing with other projects, it is a really good result. Finally, duplications are minimal on the program, existing only 1.8% of them, corresponding to 10 blocks.

4.3.3 Modular Code

All the applied programming methodologies and techniques, contribute to the creation of a Modular Code. It defends the enforcement of logical boundaries between components, improving decoupled code and minimizing dependencies, what results in easier code maintainability. With this achievement, modifications in a module or component does not change the remaining code and hence, other components functionality. It is possible to add functionalities without changing the ones already developed. Another important feature from modular programming is code reusing, in a way that allows one module to be applied in different situations. New features addition and when bugs were discovered, shows the modular way in which the code is constructed. The consequences of modifications are really scarce and the unit tests existence allows to detect components behaviour changes rapidly.

5. Conclusions

This project presented two main problems to work on. Firstly, traces data analysis was done, to understand which type of metrics of network performance can be taken from them, in order to remotely analyse the network and the service quality that is being provided to customers. Besides avoiding the need to be in the field, this kind of analysis allow to save money by the operators, which need to pay people and buy software to have someone in the field collecting such metrics.

Secondly, the challenge was to develop an application that processes all the relevant information from traces. The application should make use of Geolocation algorithms, which determine the location of the collected data, producing network performance indicators that are associated to geographical positions. KPIs are also interesting measures to be acquired from such processing. The main objective was not only the application development but also to optimise it as much as possible. This optimization is desirable in order to reduce the processing time since the ideal scenario was to get data processed from each group of traces in 15 minutes, which is the period of each group reception. The fact that this application will work as a basis to future features addition, demands an high quality code production. Code robustness and good structure, bugs and errors avoidance, ease on its detection and future usage by other developers, for both taking part of the application features or to add new ones, required several programming practices and principles to be followed.

Starting with the protocol analysis, a set of use cases was initially defined. However, only cell footprint, best server RSCP and dropped call use cases were discussed here. After understanding the protocols organization in UMTS and analyzing the messages collected by the GPEH feature, it was possible to detect which ones fit the mentioned use cases. An interesting fact is that usually, only one message from the protocol is necessary to construct the use case. That was confirmed in the dropped call one, where the event INTERNAL_SYSTEM_RELEASE detection allows its immediate identification. To the remaining use cases, the event MEASUREMENT_REPORT can be used since it has all the necessary fields to achieve the desirable results. Traces analysis also helped in events IMSI attribution. This is relevant, because allowed to define a full communication from a generic UE and tracing it, as well. For the full communication definition it was concluded that it makes more sense to use the unique triple IMSI, UE_CONTEXT and RNC MODULE_ID to associate all events belonging to the same communication.

Concerning the application development, defined use cases production was accomplished, as shown in Figures 5 and 6. In terms of these results validation, it is difficult to get reliable indicators due to the limited number of processed data along with comparable situations from real world in the specific conditions where these results were obtained. Besides limitations in the comparisons, there is also the positioning error, from the Geolocation algorithm, which is not directly part of this thesis but influences the results. However, in terms of best server RSCP, some comparisons were performed with drive tests even though from different dates. Some coherence was found in the comparisons, in both qualitative and quantitative analysis. In Tapada das Necessiades, considering drive tests with indoor attenuation, only 1.33 dB and 4 dB difference was verified in comparison with processed traces, for the average and median values respectively. Nonetheless, more data is needed in order to assert the results validity. This missing data relies in higher number of processed ROP but also in external information, that allows to get a real world
use case to proceed to such verifications.

In terms of code quality, due to tools, practices and principles followed, a modular code was built, with no duplications, decoupled and tested. All the principles detailed during this document are visible in the code. This situation contributes to a better understanding by other developers when using the application, to improve it or just to use it and get results. Features addition is simplified due to either modular code or tests existence, which contributes to bugs avoidance when changes are performed. The step that probably is farther from what was desirable, is the processing time of the application. Besides the parser component, which got a considerable optimisation work, going down from one hour and a half processing time to fourteen minutes, the Geolocation algorithm components take too long to finish, resulting in almost two hours and forty five minutes to complete one ROP processing. However, this time is relative since it depends on the number of events to process and has a great dependence of the workstation where the application is running.

This project allowed the creation of a framework to traces processing and analysis. It was developed in a way that easily grants the addition of new features, like new events processing, new use cases, new Geolocation algorithms and its usage with other telecommunication technologies, 2G and LTE, or with traces from different vendors.

References


[6] 3GPP, 650 Route des Lucioles - Sophia Antipolis Valbonne - FRANCE. "3GPP TS 32.600 V10.0.0 (2010-06) 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication management; Configuration Management (CM); Concept and high-level requirements (Release 10)", 06 2010.


