H2020-ICT-688712

Project: H2020-ICT-688712

Project Name:

5G Applications and Devices Benchmarking (TRIANGLE)

Deliverable D3.6
TRIANGLE testbed calibration and baseline

Date of delivery:	31/12/2018	Version:	1.0
Start date of Project:	01/01/2016	Duration:	18 months

Deliverable D3.6

TRIANGLE testbed calibration and baseline

Project Number:	ICT-688712
Project Name:	5G Applications and Devices Benchmarking
Project Acronym	TRIANGLE

Document Number:	ICT-688712-TRIANGLE/D3.6
Document Title:	TRIANGLE testbed calibration and baseline
Lead beneficiary:	Keysight Technologies Denmark
Editor(s):	Keysight Technologies Denmark
Authors:	Keysight Technologies Belgium (Michael Dieudonne), Keysight Technologies Denmark (Hua Wang, Marek Rohr, German Corrales Madueño), Universidad de Malaga (Almudena Díaz, Bruno García), DEKRA Testing and Certification S.A.U (Carlos Cárdenas, Janie Baños, Oscar Castañeda, Juan Carlos Mora, Pablo Romero), Quamotion (Frederik Carlier, Bart Saint Germain).
Dissemination Level:	PU
Contractual Date of Delivery:	31/12/2018
Work Package Leader:	UMA
Status:	Final
Version:	1.0
File Name:	TRIANGLE_Deliverable_D3.6_FINAL

Abstract

This deliverable describes the calibration of the testbed. The calibration of the testbed is crucial before any execution of the measurement campaign. It requires several procedures: The individual measurement instruments need to be calibrated. The traceability of the measurement techniques such as cabling loss and impacts from various measurement agents shall be characterized and compensated. Based on that, a baseline reference can be developed. Due to the dynamics introduced in the scenario, another important task is to understand how the measurements converge and to identify the number of test iterations needed to obtain a stable test result. To calibrate the ETL process, human panel experiments are performed to verify the consistency of testbed computed QoE score with human experienced QoE score. Finally, a case study of using the TRIANGLE testbed to benchmark the performance of two mobile applications, namely ExoPlayer and SkyTube, is presented.

Keywords

Testbed calibration, characterization, compensation, convergence and repeatability, human panel validation, case study

Document history

V1.0	Initial release of the document
V1.1	General rewriting of the document based on the feedback received by the expert panel at the project final review. Changes are: Section 4.2 rewritten New appendix I

Executive summary

The TRIANGLE project is aimed to develop a fully controlled End to End (E2E) testbed that allows extensive laboratory testing of services against different use cases and scenarios, thus enabling E2E Quality of Experience (QoE) evaluation for new mobile applications and devices in a repeatable manner. One of the most important tasks before using the TRIANGLE testbed to execute any test certification is the calibration of the testbed. This deliverable describes the step-by-step procedures required in the testbed calibration.

The framework of QoE evaluation using the TRIANGLE testbed can be decomposed into two basic blocks, i.e., the measurement block and the ETL process block. The measurement block is responsible for performing measurements for various Key Performance Indicator (KPIs) related to specific use cases and test cases. As individual KPIs are measured in different dimensions and scales, the ETL process block is responsible for normalizing the measured KPIs into a standard 1-to-5 scale, referred to as synthetic mean opinion score (MOS), for further QoE computations.

The calibration related to the measurement block includes the calibration of individual measurement instrument, the characterization of testbed performance (e.g., latency and throughput), and the compensations of cabling loss and impacts from various measurement agents. Another important factor affecting the measurement results is the dynamics introduced in the scenario. Statistical analysis is thus needed to ensure the consistency and repeatability of the measurement results.

The calibration of ETL block consists of calibration of the conversion process from measured KPIs into synthetic MOS values and the calibration of QoE computation. Human panel experiments are introduced at this moment to verify that the TRIANGLE commutated QoE score matches with the human measured QoE score.

The focus of this deliverable is to provide a step-by-step guidance of calibrating the TRIANGLE testbed. Also, two mobile streaming applications have been tested as a case study to verify the capabilities of the testbed as well as to benchmark the performance of the tested mobile applications.

Table of Contents:

List of Figures

List of Tables

List of Abbreviations

3GPP	3rd Generation Partnership Project
AEC	Apps Energy Consumption
APP	Application
AUE	Apps User Experience
DASH	Dynamic Adaptive Streaming over HTTP
DC	Direct current
DL	Downlink
DRA	Device with Reference Apps
DUT	Device Under Test
E2E	End to End
ETL	Extract, Transform and Load
eNB	Enhanced Node-B
E-UTRA	Evolved Universal Terrestrial Radio Access
FDD	Frequency Division Duplex
GPRS	General Packet Radio Service
GSM	Global System for Mobile communications
GUI	Graphical User Interface
IoT	Internet of Things
IP	Internet Protocol
KPI	Key Performance Indicator
LAN	Local Area Network

LTE	Long Term Evolution
MOS	Mean Opinion Score
NR	New Radio
OTA	Over the Air
QoE	Quality of Experience
QoS	Quality of Service
RES	Device Resource Usage
RF	Radio Frequency
RSRP	Reference Signals Received Power
RTT	Round Trip Time
Rx	Receiver
TAP	Test Automation Platform
TC	Test case
TCP	Transmission Control Protocol
TDD	Time Division Duplex
TS	Test Specification
Tx	Transmitter
UDP	User Datagram Protocol
UE	User Equipment
UL	Uplink
VM	Virtual Machine

1. Introduction

5G will bring a new level of services to the end user: faster transmissions, lower latency, higher reliability, massive connectivity, and adaptability to various kinds of services anywhere whenever needed. One crucial task is to ensure that the services promised by 5G will work. By working we mean that 5G must deliver the service (e.g., watching video streaming, playing online games, using a new breakthrough 5G service, etc.) with certain requirements (e.g., throughput, latency, reliability, etc.) to the end user who consumes it in certain network scenarios (e.g., in a train, walking in a crowded city centre, etc.).

The combination of uses cases and network configurations makes the theoretical validation of the Quality of Experience (QoE) for a given 5G service impossible. Field testing seems to be the logical solution, but it is very costly and results are unpredictable given the lack of control over the testing environment, e.g., changing radio propagation conditions, network conditions, etc. The TRIANGLE project aims at building a fully controlled End-to-End (E2E) testbed with a homogeneous testing framework that allows extensive laboratory testing of services against different scenarios and configurations, thus enabling E2E QoE evaluation for new mobile applications and devices in a repeatable manner.

An important pre-requisite before executing any test campaign is to ensure the testbed provides accurate measurement results. This requires various calibration steps. First, the calibration of the instruments used for making measurements. Second, the traceability of the measurement technique. Third, statistical analysis of the measurement results to make sure that the test results converge. And fourth, the conversion from the measured KPIs into the synthetic Mean Opinion Score (MOS).

The TRIANGLE testbed is composed of various measurement instruments (e.g., UXM mobile network emulator, Power Analyser, commercial handset, etc.), software components (e.g., test automation software, data performance tool, mobile test automation tool, etc.), and other hardware. The calibration of the individual measurement instruments shall be done according to standard procedures by calibration laboratories with recognized traceability. Note that calibration establishes a relation between the quantity values (with measurement uncertainties) provided by measurement standards and corresponding quantity values (with associated uncertainties) of the instrument being calibrated. Laboratories must ensure metrological traceability of the measurement standards so that the result can be related to a reference through a documented unbroken chain of calibrations.

The traceability of the measurement technique, mostly software components (e.g., various measurement agents), includes not only the verification that the measurement process is performed according to the method, but also that the measurements are accurate. During the TRIANGLE testbed development, it has been tested and verified that the behaviour of software components will not vary over time and the computations are correct. However, the impact of the software components on the measurement results (e.g., power consumption and resource usage) shall be characterized and separated from the application under test. This should be seen as an offset adjustment of the measurement system, whereby a set of operations are carried out on the system so that it provides prescribed results.

Another important thing that has to be considered in the measurement methodology is the cabling loss: measurements take place on the device in (RF) connectorized mode. The cable that connects the testbed and the device (UE terminal) introduces an attenuation and its characteristics may change over time. Therefore, the measurement methodology includes a process to compensate for the extra losses in the cable. This path compensation process is

loosely referred to as path calibration. This "*path calibration*" or path compensation is actually an adjustment of the measuring system to be done prior to making any measurement and is often mistakenly called "self-calibration". It shall be performed every time the cable changes and on a periodic basis to compensate for the drift of the cable characteristics over time.

The overall measurement process cannot be calibrated, as there is no established traceable calibration standard (measurement standard as for length, time, etc.). Therefore, a baseline reference has been developed. This serves as the calibration standard against which to calibrate the testbed. In other words by comparing the results of the measurements on a known device and application with the baseline the measurement results of the testbed are considered fit for their intended use.

Another important factor affecting the measurement results is the dynamics introduced in the network scenario. The network scenarios define the properties of the radio channel (e.g., channel model, Doppler frequency, received signal power) and the network conditions (e.g., amount of available frequency/time domain resources for scheduling), which have great impact on the overall performance. Because of the statistical nature of the dynamics introduced in the network scenario, the measurement results may vary from one iteration to another. To ensure the accuracy of the results it is also important to characterize the consistency and repeatability of the measurement results, i.e., does the testbed have consistent results when the test is performed at different days and/or different places? Also, Statistical analysis is needed in order to understand whether the average is a good estimation of the measured value and how the measurements converge towards a stable test result, as well as to identify the number of test iterations needed to obtain a good estimation in each network scenario. The latter is especially important to strike a good balance between the accuracy of the measurements and the time required to perform the measurements.

The above mentioned calibration steps, i.e., the calibration of the instruments, the traceability of the measurement technique, and the convergence and repeatability analysis, can be categorised into the calibration of "measurements". A number of KPIs are calculated during the measurement phase. The next step following the measurement is the ETL (Extract, Transform and Load) process which first converts the measured KPIs into the synthetic MOS values, and then aggregates the MOS values to obtain the QoE score for each measured domain. This ETL conversion process from measured KPIs to the final QoE score has to be calibrated as well. As there is no standardized calibration procedure for this process, human panel experiments are a proper choice to verify if the testbed generated QoE score matches the human measured QoE score.

Once the measurement instruments have been calibrated, the testbed has been "calibrated" against the baseline, the number of test iterations needed for the testbed results average to converge has been determined, and the ETL process has been calibrated, we could say that the testbed has been fully calibrated and is ready for running test campaigns. In this deliverable we tested two example applications, namely ExoPlayer and SkyTube. The results demonstrate the measurement capabilities of the testbed and how to benchmark the performance of the two well-known applications.

A diagram of the TRIANGLE framework with all necessary calibration blocks is shown in Figure 1.

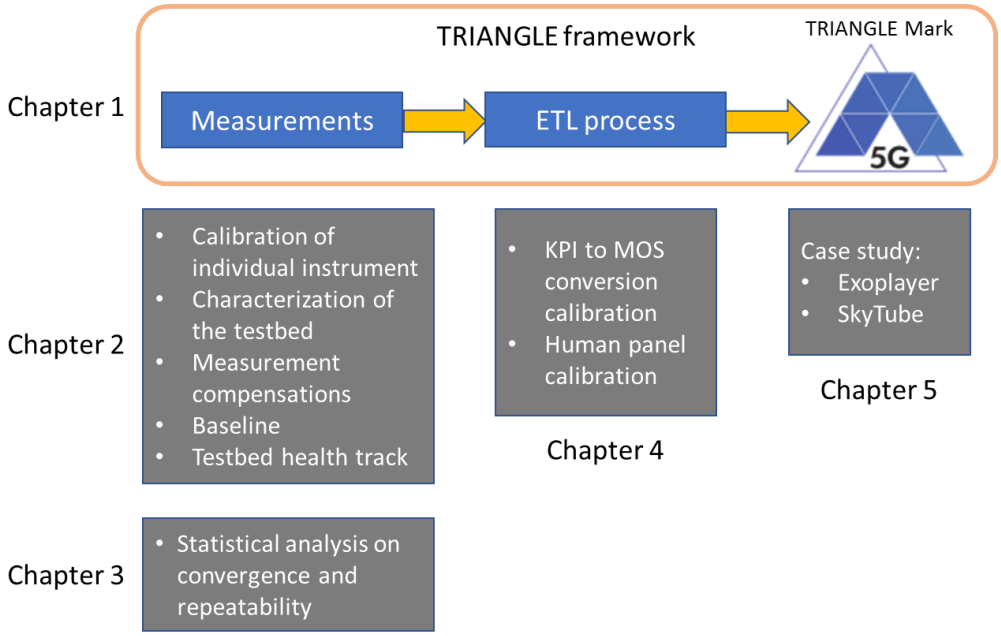

Figure 1: General structure of the deliverable

The remainder of this deliverable is structured as follows:

Chapter 2 presents the necessary procedures required for the testbed calibration, which includes the calibration of individual measurement instrument, the latency and throughput baseline characterization of the testbed, the characterization and compensation of cabling loss and measurement agents. Once the calibration procedures have been performed, the testbed is ready to take measurements.

Chapter 3 investigates the convergence and repeatability of the measurement results with the objective to identify the number of test iterations needed to obtain a good estimation of the result for each use case and in each network scenario. Toward that end, two replicas of the testbed, located in DEKRA Spain and Keysight Denmark, have been built. Five traffic profiles, namely synthetic data (TCP and UDP streams), YouTube video streaming, Spotify audio streaming, Voice MOS, and Google Earth Virtual Reality, have been tested to characterize the performance of each network scenario defined in the project.

Chapter 4 presents the calibration of the ETL process, which consists of conversion from measured KPIs into synthetic MOS values and the computation of QoE score. The human panel experiments are performed to verify how human measured QoE matched the measured QoE, by exposing the same video to humans and to the automated processing of the testbed under various scenarios.

Chapter 5 presents a case study of using the TRIANGLE testbed to benchmark the performance of two mobile applications for content streaming, namely ExoPlayer and SkyTube. The two applications are evaluated under various scenarios and different reference phones. The detailed analysis of how the measured KPIs are converted and averaged to the domain score and to the final TRIANGLE mark is illustrated, as well as high level evaluations of the two apps running in different phones.

Chapter 6 concludes the deliverable.

2. Testbed Calibration

The main architecture of the TRIANGLE testbed is shown in Figure 2, which is composed of various instruments (e.g., UXM mobile network emulator, Power Analyser, etc.), tools (e.g., test automation software, data performance tool, mobile test automation tool, etc.) as well as other hardware. Each component is controlled through TAP [1], which is used to orchestrate different components and run the test cases. The radio access emulator is provided by the UXM Wireless Test Set [2], which also includes a channel emulator for the emulation of various complex radio channels. The testbed also includes a set of reference devices which are connected to the testbed via calibrated cabling. The purpose of using cabling instead of using Over-the-Air (OTA) connection is to enable accurate control over the radio channel conditions emulated by the UXM. To accurately analyze the power consumption during the experiments, the devices are directly powered by N6705B power analyser [3]. Besides the instruments, the testbed also uses software tools such as the data performance tool [4] and the mobile test automation tool [5]. The performance tool is used to monitor memory and CPU usage, as well as to derive quality of service (QoS) measurements. The mobile automation tool is used to automate user interactions with the application, e.g., login, start/stop/pause/resume playing, etc. The detailed description of the testbed, the orchestration of the framework, and the components of the testbed are described in [6].

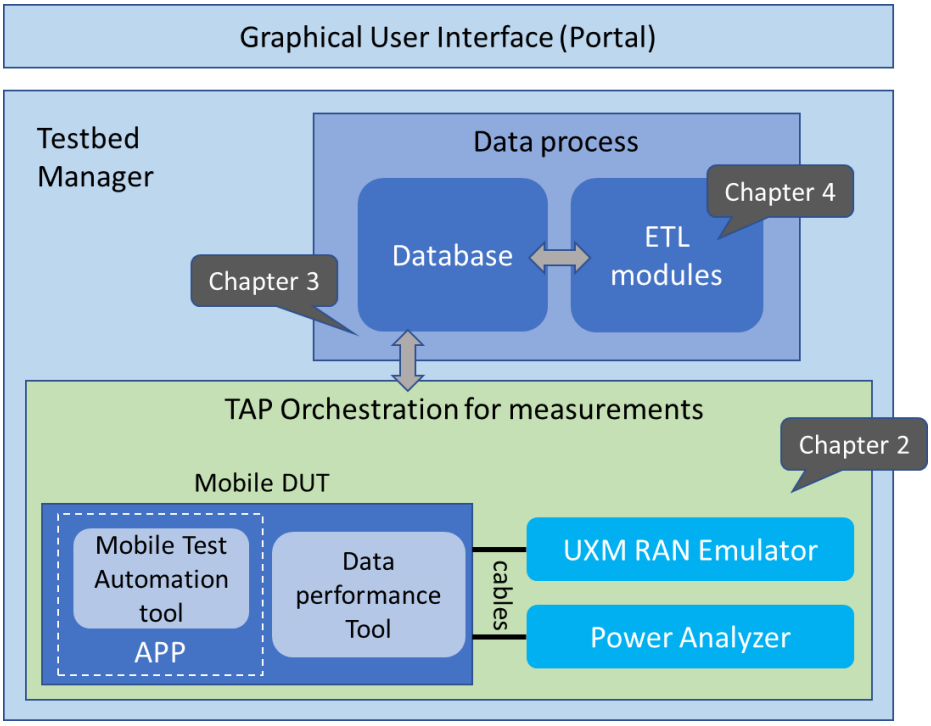

Figure 2: TRIANGLE testbed architecture.

An important pre-requisite before executing any test campaign is to calibrate the performance of the testbed, which involves various steps as described in the introduction. In this chapter, the calibration of the testbed is focused on the measurement part, which consists of calibrating the individual instrument independently, as well as characterizing the overall performance of the testbed against the baseline by running a set of measurements to estimate compensation values for certain measurements.

2.1 Calibration of individual measurement instrument

This section describes the calibration requirements of the individual measurement instrument in the testbed.

2.1.1 Factory calibration of instruments

The test instruments used in the TRIANGLE testbed need to be individually calibrated and verified. In the present configuration of the testbed, the instruments are the UXM mobile network emulator and the N6705 Power Analyser. The manufacturer shall provide the instruments with a proper factory calibration certificate.

2.1.2 Third party calibration of instruments

Calibration, or metrology calibration, performed by accredited third party laboratories. Calibration is the relationship between a measurement standard of known uncertainty and test equipment of unknown uncertainty. To perform calibrations a laboratory must be able to establish a full chain of traceability from its standard to a nationally recognized standard held by an organization such as NIST in the United States. The national organization also must have international traceability.

The calibration also determines the uncertainty of the measurement instrument.

In the present configuration of the testbed, the measurement instruments subject to third party calibration are the UXM mobile network emulator and the N6705 Power Analyser. The recommended calibration frequency is two years.

2.2 Characterization of the testbed

This section describes the characterization of the overall performance of the testbed in terms of the latency and the maximum achievable throughput in the testbed. These two have been selected as the most relevant parameters to stablish the baseline of the testbed and are served as calibration reference. The latency measurement shall also be used to apply the corresponding offset adjustments.

2.2.1 Testbed latency

In order to characterize the latency that different devices under test (DUT) will experience in the testbed, the round trip time (RTT) needs to be measured at multiple levels, such as the delay to reach known servers within the private LAN containing the testbed, the latency in-between the different instruments and VMs within the TRIANGLE LAN, as well as additional latency brought in by TRIANGLE network impairments.

Latency can be evaluated from the perspective of a test PC (running TAP), as well as from the perspective of an Android DUT connected to the testbed.

The relevant measurements related to latency are:

- Latency from UE to Internal data server (within TRIANGLE testbed)

  This latency is controllable and can be calibrated.

- Latency from UE to External (uncontrolled) data server

  This latency is not under control as it is service provider dependent, but it should be characterized.

To facilitate the measurement of the delay, a calibration test case has been developed and the corresponding TAP test steps and TAP plugins have been developed. An example test step setting for the testbed latency characterization is shown in Figure 3. The IP address in field “Server 1” indicates the address of the local data server used for latency testing, while the IP address in field “Server 2” and “Server 3” refer to the address of the external data servers used for latency testing (one with short and one with long geographic distance). Once the latency is measured, the values are saved in the Calibration YAML file. The YAML file will be used as input to the ETL process, and used in the KPI calculations.

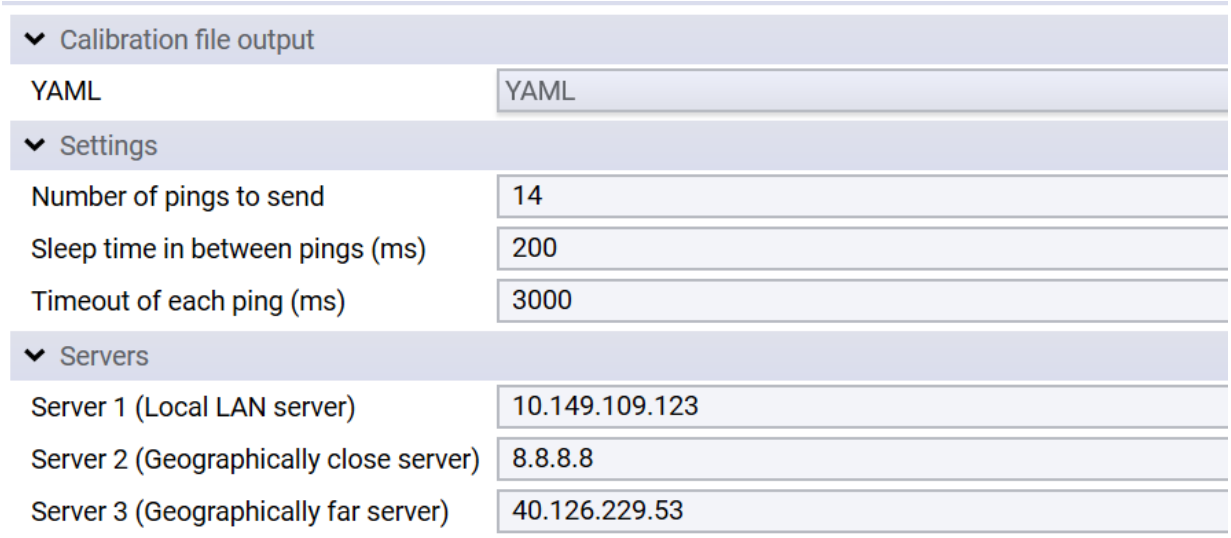

Figure 3: Test step settings for the testbed latency characterization.

2.2.2 Throughput

The maximum supportable throughput of the testbed needs to be characterized. The configured data rate in each test case should not exceed the maximum throughput of the testbed. Evaluation of the maximum throughput of the testbed include:

- Throughput test within the testbed LAN (100 Mbps or 1 Gbps local LAN link)
- Throughput test towards known servers outside the testbed LAN

2.3 Cabling loss compensation

The DUTs used in the TRIANGLE testbed have to be modified (connect RF cables at the antenna ports and the battery) to allow connectivity with various external instruments (e.g., UXM, power analyser) via cables. The purpose of using cabling instead of using Over-the-Air (OTA) connection is to enable accurate control over the radio channel conditions emulated by the UXM. However, this may lead to a difference in the desired RF signal power and the actual RF signal power reaching the modem due to the cabling loss and connection loss. The cabling loss needs to be measured and compensated during the calibration procedure.

The measurement of the cable loss is performed in downlink, per DUT RX antenna and per LTE band. The calibration is based on UE measurements which are fed back to the UXM via RSRP reports.

A TAP test plan has been developed for the measurement and compensation of the cabling loss.

This is achieved via a TAP test step which:

- Generates RF DL compensation per connector, per band, per cell
- Measures the difference between the configured eNB DL power and measured RSRP power at UE side per connector, per band, per cell
- Writes the values in the Calibration YAML file, for the device under test or the device used to host the application under test
- Applies to UXM Control Panel

An example configuration of TAP test step for cabling loss measurement is illustrated in Figure 4.

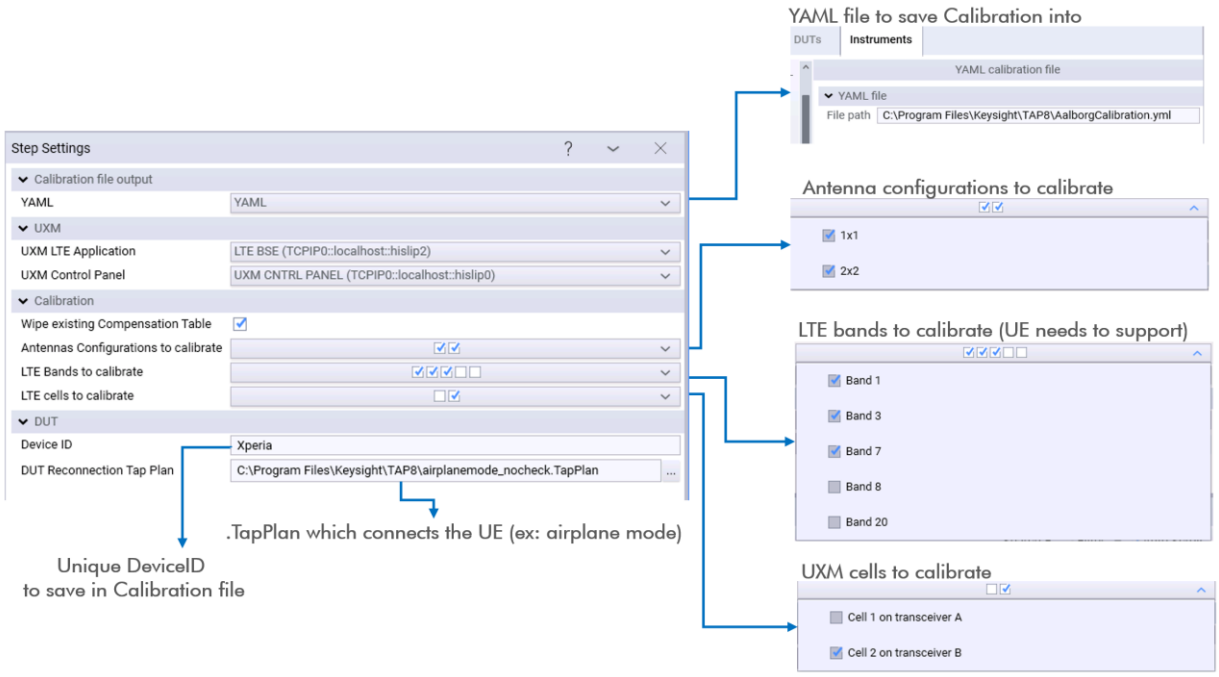

Figure 4: Example configuration of TAP test step for cabling loss measurement.

Once the cabling loss characterization is performed for a device, at the beginning of a test campaign, a TAP test step is created and executed for the compensation of cabling loss. This test step reads the DL RF calibration from the YAML file for a specific device, and then applies to the UXM Control Panel the offset values for all previously calibrated connectors, bands, transceivers. This approach guarantees identical received pilot power for all LTE devices in the testbed, which compensates for the soldered RF modifications of the devices.

An example configuration of TAP test step for cabling loss compensation is illustrated in Figure 5.

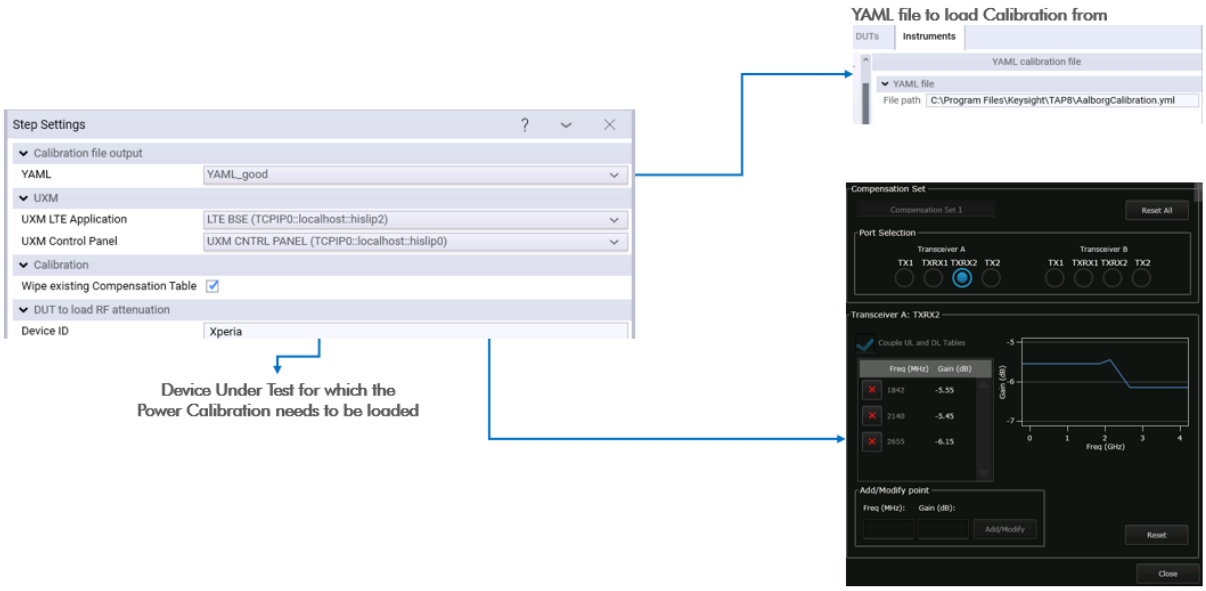

Figure 5: Example configuration of TAP test step for cabling loss compensation.

2.4 Measurement compensation

When testing an application (App under test) the measured energy consumption and resource usage does not only come from the App itself. Even if all other Apps are disabled in the device of which the App under test is running, other processes that cannot be disabled (e.g., measurement agents) consume energy as well as resources. Therefore, it is important to distinguish the energy consumption and resource usage of the App under test from all the other factors (e.g., reference device and the measurement agents).

The characterization of energy consumption and resource usage from other factors include device DC power consumption and measurement agents resource usage. Appropriate compensations need to be performed after the characterization process.

2.4.1 Calibration test cases for App under test

In order to measure the energy consumption and resource usage from the reference device and the measurement agents, proper calibration test cases have to be defined. For App under test, the calibration test scenarios for different use cases and test cases are listed in Table 1.

Table 1. Calibration test scenarios for App under test

Use case	Test case	Calibration characteristics			
		Screen	Activity	Traffic	Traffic Direction
Common	001,002,003	Screen ON	Active	Low/None	DL
Content Streaming	001,002,003	Screen ON	Active	Medium	DL
	004	Screen ON / Screen OFF	Active	High	DL
	005	Screen OFF	Active	Medium	DL

Live Streaming	001	Screen ON	Active	Medium	DL
	002	Screen ON	Active	Medium	UL
	003	Screen OFF	Active	Medium	UL
Social Networking	001,002,003	Screen ON	Active	Low	UL
	004,005,006	Screen ON	Active	Low	DL
High Speed Internet	001, 003	Screen ON / Screen OFF	Active	High	DL
	002, 004	Screen ON / Screen OFF	Active	High	UL
	005	Screen OFF	Active	High	DL
	006	Screen OFF	Active	High	UL
Virtual Reality	001, 002	Screen ON	Active	Medium	DL
Augmented Reality	001,002,003	Screen ON	Active	Low	DL
Gaming	001,002,003	Screen ON	Active	Medium	DL

It can be noted from Table 1 that in none of the test cases the device is in idle state, and that in most of the test cases the screen is ON. However, some of the use cases and/or test cases do not need the screen to be on (e.g., audio streaming). Thus, the screen OFF condition should also be covered in the calibration test case.

Therefore, the number of calibration test cases that are to be implemented is only 2: device active with screen ON, and device active with screen OFF.

However, there is a need to quantify the impact of the measurement agent as well:

- In AEC domain, the measurement can run with the measurement agent killed (hence the need for baseline values without the measurement agent)
- In RES domain, the measurement must run with the measurement agent up and logging.

Thus, there are two more variants of the calibration test cases above: with measurement agent running & logging and with measurement agent killed.

This brings the total number of calibration test cases to four:

- Device active with screen ON and measurement agent running
- Device active with screen ON and measurement agent killed
- Device active with screen OFF and measurement agent running
- Device active with screen OFF and measurement agent killed

Note that the traffic characteristics (amount of data and direction) is for purely informative purpose and should not be included in calibration baseline, as the App itself will generate the traffic.

2.4.2 Device DC power consumption

To accurately analyze the power consumption during the experiments, proper hardware modifications are needed for the device under test, such as:

- To fake or shortcut the UE battery connector and replace it with power cables
- To modify a USB charging cable with power cables, in order to track power leakage/charging over USB

A device modified in this way shall be directly connected and powered by a power analyser and have its DC power consumption recorded in the calibration test cases. The DC power consumption for the device will then be characterized and compensated in the KPI calculations.

2.4.3 Measurement agents resource usage

In order to automate the test and to capture the application resource usage during the test, the UE runs several measurement agents such as Data Performance tool developed by DEKRA and the Mobile Test Automation tool developed by Quamotion. The data performance tool tracks and logs during test execution, and can also be used to monitor CPU/GPU/memory usage. The mobile automation tool is used to automate user interactions with the application, e.g., login, start/stop/pause/resume playing, etc.

Similar to the device DC power consumption characterization, in order to quantify how many resources are used specifically for the application under test, the impact of these measurement agents needs to be characterized (especially in terms of power consumption) and compensated whenever a new test reference device is added.

2.4.3.1 DEKRA agent impact characterization

The impact of DEKRA agent can be characterized by running the four calibration test cases described in Section 2.4.1. An example of the calibration measurements of power consumption for DEKRA agent is shown in Figure 6. The values in the orange box indicate the power consumption with DEKRA agent killed, and the values in the green box indicate the power consumption with DEKRA agent up and logging.

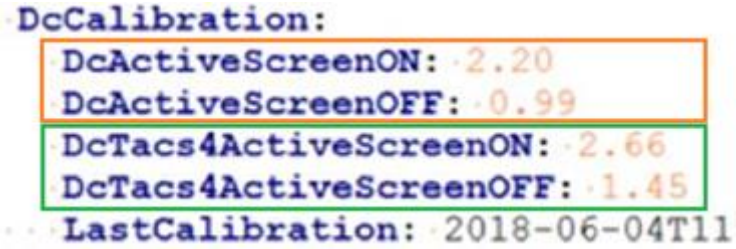

Figure 6: Example of baseline measurements of power consumption for DEKRA agent.

It can be seen from Figure 6 that for that specific test device, the DEKRA agent consumes 0.46W of power when it is running & logging. This value shall be subtracted from the test campaign measurement results in the ETL process (kind of an offset adjustment).

Naturally, as it is the DEKRA agent which logs the CPU, GPU and RAM usage, it is not possible to quantify the usage of these metrics when it is not running.

2.4.3.2 Quamotion agent impact characterization

The impact of running the Quamotion agent on the UE as a mechanism to emulate clicks, searches and presses on the device screen has to be characterized, in order to quantify whether it should be compensated or not in the ETL process.

In this section, the measurements of device resource usage and power consumption for setting up the Quamotion device session will be ignored, as this happens before the application flow starts, followed by the useful measurements.

Across the Quamotion API, the operations which potentially require the highest CPU and power usage are element searches across the application graphical user interface (GUI). This method is called “Find-Element” in the API.

A test case has been developed to analyze the impact of Quamotion’s Find-Element calls. After launching the Quamotion agent, it remains idle for 60 seconds. Then the agent performs 51 successive Find-Element calls. After that, the agent remains idle for another 60 seconds. Then it performs a new round of 51 successive Find-Element calls with 5 seconds of sleep interval, followed by 60 seconds idle period and App start.

Figure 7 shows the Impact of Quamotion’s Find-Element API call on current consumption and CPU usage. It can be seen that there is minor impact on the current consumption when performing the Find-Element searches with the Quamotion agent. But for CPU usage, the impact of performing the Find-Element searches is quite obvious, where the spikes are very visible.

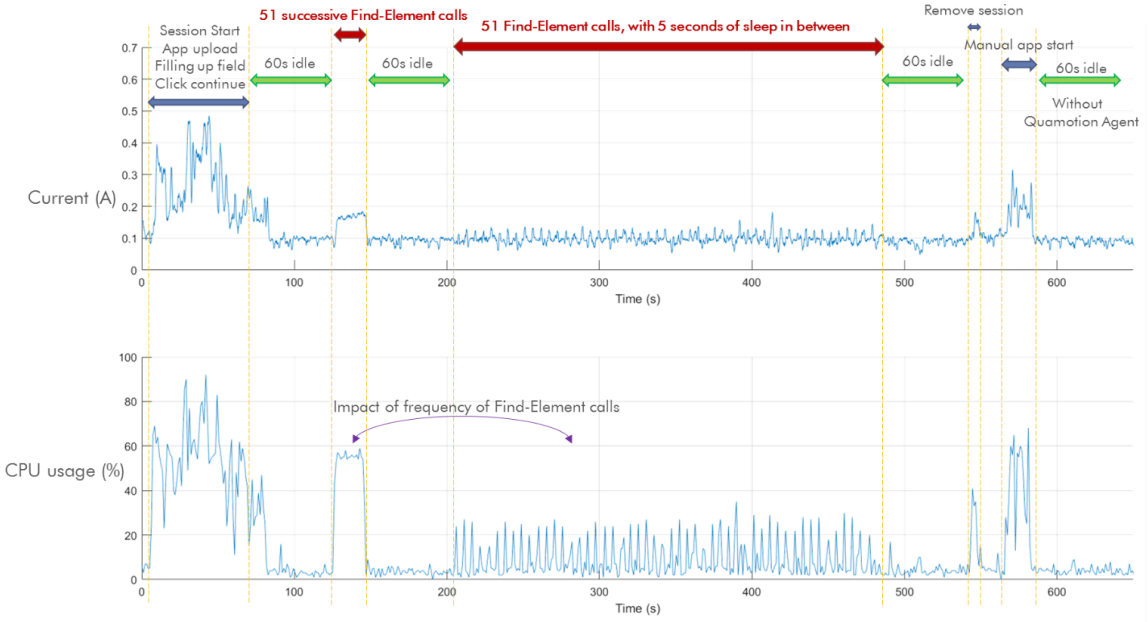

Figure 7: Impact of Quamotion's Find-Element call on current consumption and CPU usage.

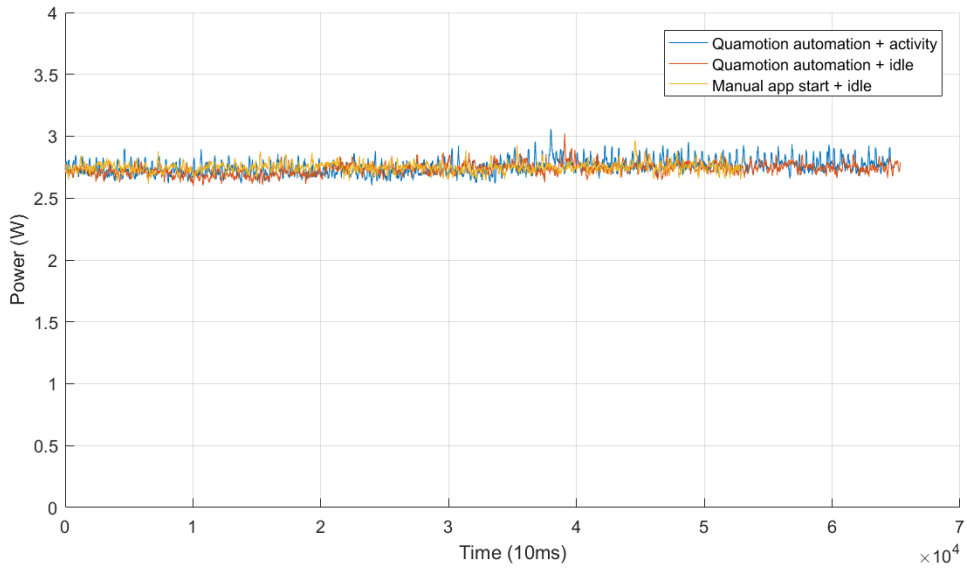

Figure 8: Power consumption with (active/idle) and without Quamotion agent.

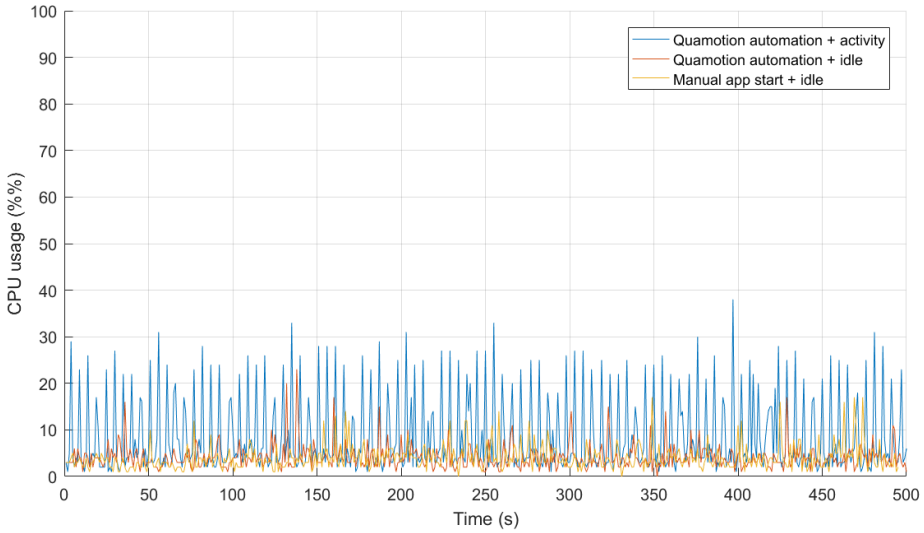

Figure 9: CPU usage with (active/idle) and without Quamotion agent as measured with the DEKRA agent

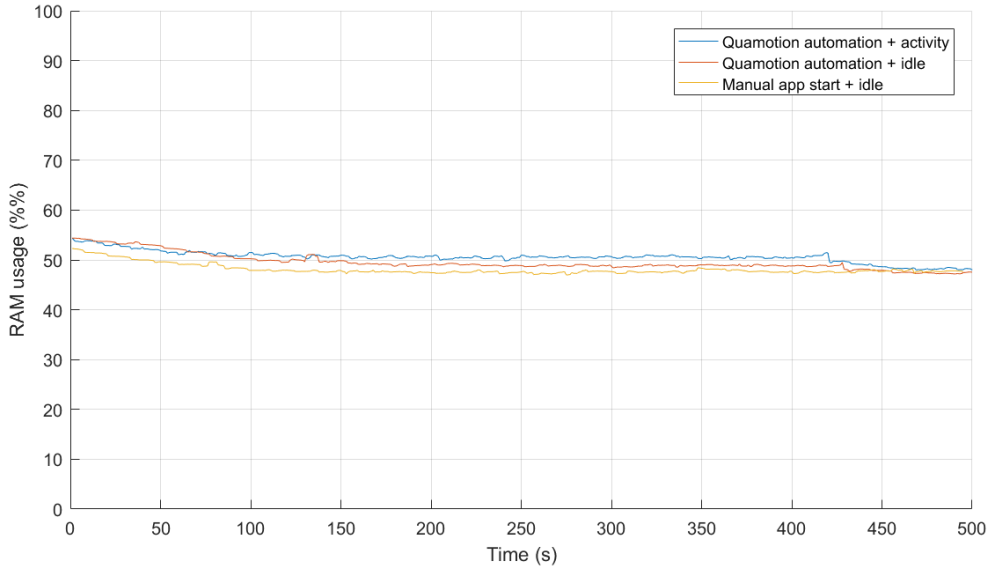

Figure 10: RAM usage with (active/idle) and without Quamotion agent as measured with the DEKRA agent

Figure 8 to Figure 10 show the Impact of Quamotion agent on power consumption, CPU usage, and RAM usage respectively. It can be seen from the figures that the highest impact the Quamotion agent exerts is on the CPU usage, especially when the Find-Element requests are sent frequently and successively. Table 2 summarizes the measurement results of Quamotion agent's impact in different domains.

In most of the test cases defined in the TRIANGLE project, an application test flow normally uses very few Find-Element calls. Typically, the application under test is started, configured, then left to run, such as in many test cases from the Content Streaming use case.

Therefore, in a conclusion, the impact from the Quamotion agent and application control on the power consumption and resources usage can be neglected.

Table 2. Energy consumption and resource usage with and without Quamotion agent

	Power consumption	CPU usage	RAM usage
Quamotion + activity	2.75 W	8.35 %	50.4 %
Quamotion + idle	2.74 W	4.08 %	49.1 %
Manuall app start + idle	2.75 W	4.00 %	48,1 %

2.5 Baseline or testbed calibration standard reference

Once the calibration procedure described in previous sections (i.e., from Section 2.1 to Section 2.4) has been completed, the measurement part of the testbed has been calibrated. As a result, two calibration tables, namely a testbed calibration table and a device compensation table, should be generated. The testbed calibration table summarizes the characterization of the testbed as described in Section 2.2, while the device compensation table summarizes the measurement of cabling loss and the impact of measurement agents on energy consumption and resource usage, as described in in Section 2.3 and Section 2.4 respectively. These two tables serve as the baseline reference against which to calibrate the testbed. Now we can say that the measurement part of the testbed is in calibrated state.

An example of testbed calibration table and device compensation table generated from the DEKRA testbed is shown in Table 3 and Table 4, respectively.

Table 3. Testbed calibration table

Parameter	Value
Testbed Name	DEKRA
Internal Latency (RTT)	25 ms to server A
External Latency (RTT)	36 ms to server A (8.8.8.8)
Max Throughput	204.87 Mbps to server A (local) 92.87 Mbps to server B (remote)

Table 4. Device compensation table for App under test

Parameter	Value
Testbed Name	DEKRA
UE ID	356397081286791
UE Description	Samsung Galaxy S7
UE Supported Bands	LTE Bands 1, 2, 3, 4, 5, 7, 8, 12, 13, 17, 18, 19, 20, 25, 26, 28, 38, 39, 40, 41

UE Supported Modes	FDD/TDD
RF Loss Compensation	-2.0499 dB on Band 1 (TXRX1) -2.3500 dB on Band 1 (TXRX2) -1.8500 dB on Band 3 (TXRX1) -1.9500 dB on Band 3 (TXRX2) -4.3700 dB on Band 7 (TXRX1) -4.5499 dB on Band 7 (TXRX2) -1.4500 dB on Band 8 (TXRX1) -1.5499 dB on Band 8 (TXRX2) -1.3500 dB on Band 20 (TXRX1) -1.5499 dB on Band 20 (TXRX2)
DC Power consumption (idle)	1.025 mW/s (Screen On) 0.566 mW/s (Screen Off)
DC Power Consumption (active)	1.029 mW/s (Screen On) 0.667 mW/s (Screen Off)
DC DEKRA Power Consumption	1.467 mW/s (Screen On) 1.204 mW/s (Screen Off)
DC Quamotion Power Consumption	2.256 mW/s (Screen On)
Total DC Power Compensation	3.465 mW/s
DUT Latency	-
DUT CPU Usage (idle)	10.191%
DUT CPU Usage (Active)	10.407%
DUT RAM Usage	1.596 MB
DUT CPU with DEKRA	16.554%
DUT GPU with DEKRA	-
DUT RAM with DEKRA	1.367 MB
DUT CPU with QUA	17.17%
DUT GPU with QUA	-
DUT RAM with QUA	1.783 MB

2.6 Testbed health tracking

Before running a complete test campaign, a short test campaign which consists of a subset of calibration test cases shall be executed to check and validate the health of the testbed. The obtained testbed calibration results should be compared with the baseline results. If there are noticeable differences (e.g., >20% variations) between the calibration results and no environmental changes have been made in the testbed, necessary actions should to be taken

to check where the difference is coming from, e.g., perform the testbed calibration procedure described between Section 2.1 and Section 2.4. This procedure is defined as testbed health tracking procedure. If there are no differences the testbed is considered to be ready for use.

The testbed health tracking procedure should be repeated whenever an event which could potentially cause changes in the environmental settings of the testbed happens or a new element has been introduced into the testbed. Examples of such events include:

- New device replaced/added in the testbed
- Change of internet access provider to the testbed
- RF cabling change, new shielding box, new splitters…
- UXM software modifications
- Test parameter change, e.g. RF band change
- EPC modifications
- etc.

At this stage, the relevant sections of testbed calibration process as defined in this document shall be repeated. The generated calibration results should be compared to the previous baseline results, and/or replace the previous baseline results if necessary.

2.7 Summary

The testbed needs to be calibrated and proper compensations for certain measurements have to be made against the baseline before execution of any test campaign. This chapter describes the necessary calibration procedures needed for the TRIANGLE testbed, which include:

- Calibration of the individual measurement instruments

  The metrology calibration of the individual measurement instruments shall be done according to standard procedures by calibration laboratories with recognized traceability.

- Characterization of the testbed

  The characterization of the overall test performance includes latency and throughput characterization.

- Cabling loss compensation

  The cable that connects the testbed and the device (UE terminal) introduces additional attenuation and its impact on the measurement results has to be characterized and compensated. It shall be performed very time a new device (e.g., phone) or a new cable is connected to the testbed.

- Measurement compensation

  When testing an App, the measured energy consumption and resources usage are not only due to the App itself. It is important to distinguish the measurement results of the App under test from all other factors, which include the device DC power consumption and the impact from the measurement agents, i.e., DEKRA agent and Quamotion agent. It shall be performed very time a new device (e.g., phone) is connected to the testbed.

Once the testbed calibration procedure is done, the outcome is a testbed calibration table and a device compensation table, which are served as the baseline reference against which to calibrate the testbed and will be used as input to the ETL process in the KPI calculations.

Whenever an event which could potentially cause changes in the environmental settings of the testbed happens or a new element has been introduced into the testbed, the testbed calibration procedure should be repeated.

3. Convergence Study

When the testbed is fully calibrated and fit for its purpose, a complete test campaign can be executed. Typically, in a test campaign, a test case will be executed in various predefined network scenarios, each of which consists of a number of sub-scenarios. A complete list of network scenarios defined in the project can be found in D2.2 [7]. Because of the dynamics introduced in the network scenarios, the measurement results may vary from one iteration to another. Therefore, TRIANGLE has developed the test cases in such a way that the measurements are replicated several times in each run of the test case. Assuming the values are affected by a random process, the error is eliminated by averaging the results over various iterations of the same test. The resulting quantity value, the average, is expected to decrease the associated measurement uncertainty. Thus, it is important to understand how the average of the measurements converge towards a stable test result and to identify the number of test iterations needed to obtain a good estimation of the result in each network scenario.

3.1 Test duration

Test cases, which specify the test conditions, the generic app user flow, the raw measurements that shall be collected during the execution, etc., have to be defined before launching any test campaign. One of the first and important steps in defining a test case is to set the duration of the test. Because the defined test case will be executed in each of the supported network scenarios sequentially, the duration of the test will depend on the network scenario.

As an example, we consider a test case of downloading files sequentially, which belongs to the high speed internet use case. The test case is specified in Table 5. The test duration is determined by the download file size, which can to be 20 MB or 50 MB. The selected network scenario is Urban Pedestrian (UR-PE) with sub-scenario 1 and 2. In sub-scenario 1, the received RSRP decreases linearly from -90 dBm to -105 dBm to emulate walking away from the cell center, and the duration of sub-scenario 1 is around 60 seconds. In sub-scenario 2, the received RSRP is fixed at -105 dBm, but with high number of users to emulate the environment of a busy green area, such as a park, and the duration for sub-scenario 2 is around 30 seconds. The starting sub-scenario in each test iteration is randomly selected.

Table 5. File Download Test Case Description

Identifier	DRA/HS/001 (Mobile Device User Experience with Reference Apps/High Speed Internet/001)
Test Case	Download two files sequentially
File Size Configuration	2 x 20 MB or 2 x 50 MB
NW Scenario	UR-PE with sub-scenario 1 (~60 s) sub-scenario 2 (~30 s)
KPI	Average throughput

Figure 11 and Figure 12 show the measured average throughput in each test iteration with different file size (2x20 MB or 2x50 MB), respectively. The duration of the configured network scenario of UR-PE is around 90 seconds. For 2x20 MB file size, the test duration (file download

time) is around 40 seconds, which is shorter than the duration of the network scenario. While for 2x50 MB file size, the test duration is around 100 seconds, which is longer than the duration of the network scenario. It is clearly shown in the figure that the measured throughput in the different iterations has less variations with large file size. That is because with large file size, the test duration is long enough to cover all the sub-scenarios (2 sub-scenarios in this case). For smaller file size, e.g., 20 MB, the test duration cannot cover both sub-scenarios. As the starting sub-scenario is randomly selected in each iteration, the variation of the measurement result is more obvious when the test duration is shorter than the duration of the network scenario.

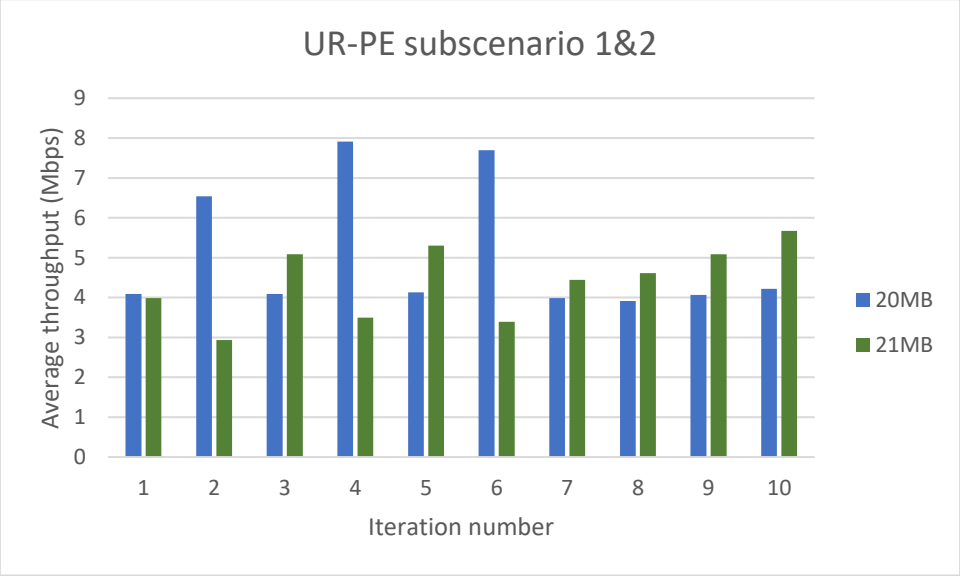

Figure 11: Average throughput in each iteration with 2x20 MB file size in UR_PE scenario

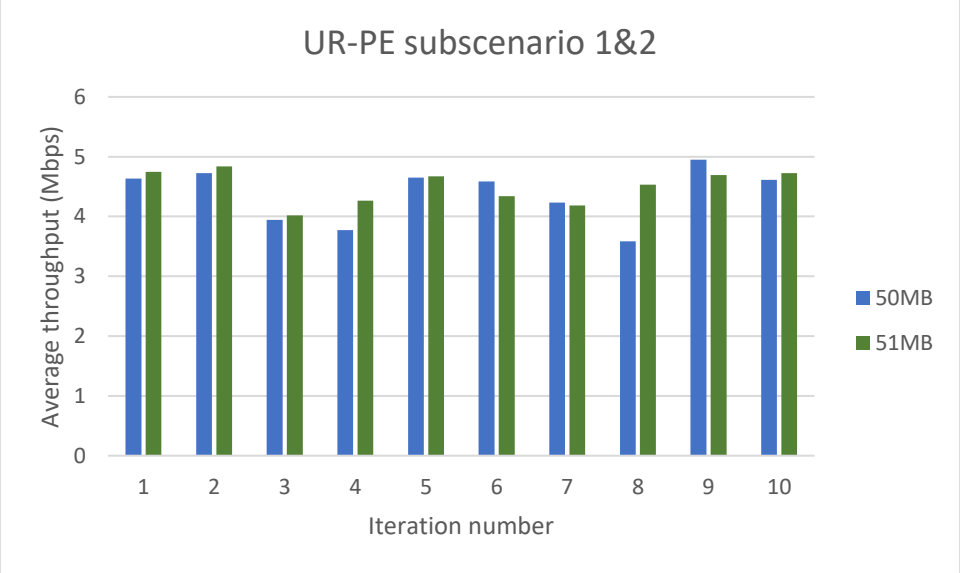

Figure 12: Average throughput in each iteration with 2x50 MB file size in UR_PE scenario

A general rule of thumb is that the test duration has to be longer than the duration of the configured network scenario so that all the sub-scenarios can be properly covered.

The network scenarios developed in TRIANGLE are listed in Table 6. The network scenarios define the properties of the radio channel (e.g., channel model, Doppler frequency, received signal power) and the network conditions (e.g., amount of available frequency/time domain resources for scheduling), which have great impact on the overall performance.

Table 6. Pre-defined Network Scenarios in TRIANGLE

SC	Network Scenario	Number of sub-scenarios	Duration of the scenario
Urban			
UR-OF	Urban-Office, indoor conditions	5	150 s
UR-PE	Urban-Pedestrian, outdoor slow mobility	4	150 s
UR-DN	Urban-Driving-Average driving	2	100 s
UR-DT	Urban-Driving-Traffic jam	2	60 s
UR-DE	Urban-Driving-Emergency driving	2	40 s
UR-IB	Urban-Internet Cafè, Busy Hours	3	90 s
UR-IO	Urban-Internet Cafè, Off-Peak	3	90 s
Sub-Urban			
SU-FE	Suburban-Festival	2	60 s
SU-ST	Suburban-Stadium	2	60 s
SU-SB	Suburban-Shopping Mall, Busy Hours	2	60 s
SU-SO	Suburban-Shopping Mall, Off-Peak	2	60 s

Each network scenario is further broken down into a number of sub-scenarios, which emulate even more accurately the radio channel variations and number of concurrent cellular users within a single network scenario. For instance, during the lab emulation of the Suburban-Festival network scenario, the frequency and time domain resources available to the application under test will be dynamically changing, to capture downlink-starved situations (as can be experienced at a festival entrance where many attendees download their e-tickets) as well as uplink-starved conditions (usual when a popular band is playing and the crowd is live-streaming the performance on social media). In addition to emulating realistic cell capacity, the testbed imposes RF conditions and network impairments (packet latency, packet losses, etc.) to fully represent field conditions in a controlled lab environment. Detailed configurations of each network scenario defined in the project can be referred to [7].

With the information of each predefined network scenario described in Table 6, the test duration of application/device under test can be determined accordingly. For example, for content distribution streaming services (e.g., video playback), all predefined network scenarios are applicable. Therefore, the recommended test duration for content distribution streaming service

would be longer than 150 seconds (i.e., the longest scenario among all). The same rule applies to other test cases, e.g., high speed internet uploading/downloading, etc.

3.2 Convergence and repeatability study

One of the objectives of this study is to investigate the consistency of the results, i.e., does the testbed maintain its characteristics and have consistent results when the test is performed at different days and/or different places. For that end, we have built two replicas of the testbed, one located in DEKRA Spain and another in Keysight Denmark. The measurements were taken under static conditions, under various radio channel conditions, and under different traffic loads, according to the networks scenarios defined in the project.

To characterize the performance of each network scenario, the following five traffic profiles have been tested: synthetic data (TCP and UDP streams), YouTube video streaming, Spotify audio streaming, Voice MOS, and Google Earth Virtual Reality.

The duration of each test iteration should be longer than the duration of the configured network scenario to make sure that each test iteration covers all the sub-scenarios. As the longest network scenario lasts for 150 seconds according to Table 6, the minimum duration of each test iteration was set to 180 seconds.

3.2.1 Synthetic Data

The configured traffic profile for synthetic data is listed in Table 7, which includes TCP and UDP streams. The TCP stream is configured with different number of streams and with the maximum possible data rate. Here “the maximum” means the TCP stream adapts to the maximum throughput of the network with TCP congestion control. The UDP stream is configured with different data rates, which correspond to low rate and high rate.

Table 7. Configured Traffic Profile for Synthetic Data

Traffic profile	Test duration in each iteration	KPI
TCP Max. x1 stream	180 s	Throughput
TCP Max. x4 streams	180 s	Throughput
UDP @ 128 kbps	180 s	Throughput, delay, jitter, packet loss
UDP @ 2 Mbps	180 s	Throughput, delay, jitter, packet loss

To determine how many iterations are needed in each network for the average value to converge, two independent performance evaluations have been carried out at DEKRA Spain and Keysight Denmark.

Figure 13: Convergence performance with TCP traffic in different network scenarios

Figure 13 and Figure 14 show the throughput and data convergence performance with TCP x1 stream traffic in urban and suburban scenarios, respectively. The x-axis is the number of iterations and the y-axis in the convergence subplot is the absolute difference between the total average (15 iterations) and after X number of iterations. The throughput subplot clearly shows

that there are fluctuations from test iteration to iteration due to the changing radio channel conditions. The convergence subplot indicates that the number of test iterations needed for the average to converge depends on the network scenario. For some scenarios with fast changing channel and network conditions more iterations are needed to achieve a stable result within 5% variation. Generally, for TCP traffic, more than 10 iterations are recommended. For UDP traffic, quite stable results can be achieved with a few number of iterations if the traffic load is lower than the channel capacity. However, for UDP traffic with traffic load close or higher than the channel capacity, higher number of iterations (e.g., 15 iterations) are needed to average out the variations. It is worth mentioning that although the absolute throughput performance measured at the two testbeds (DEKRA and KEYD) are different due to the use of different phones, the convergence performance from the two testbeds are comparable with each other.

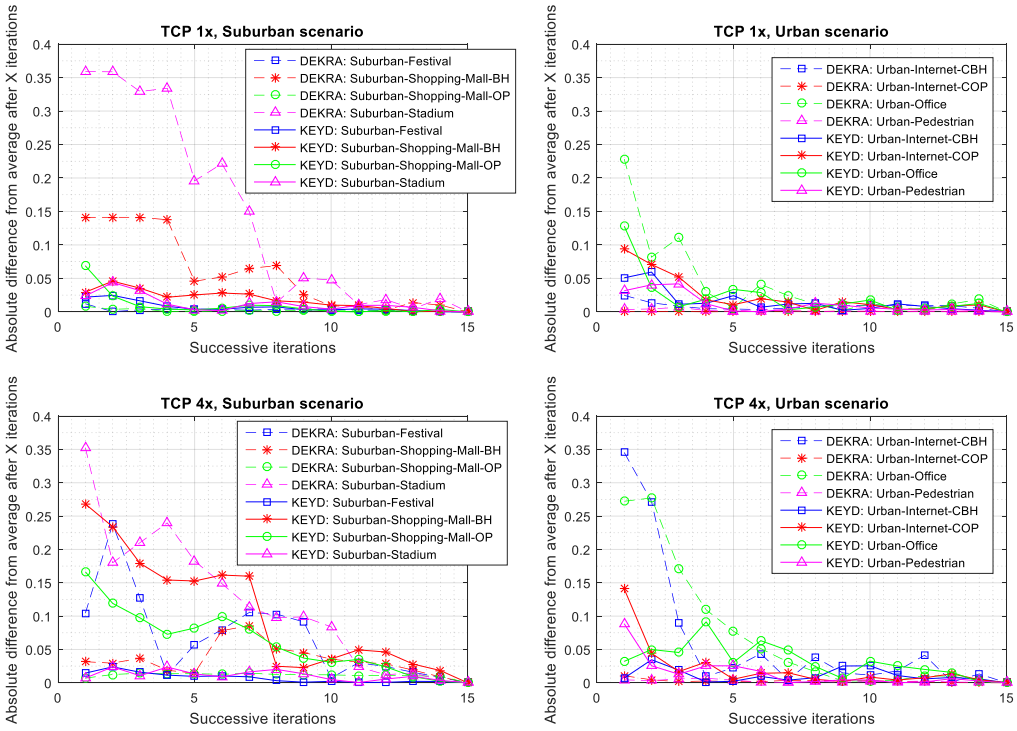

Figure 13: Convergence performance with TCP traffic in different network scenarios

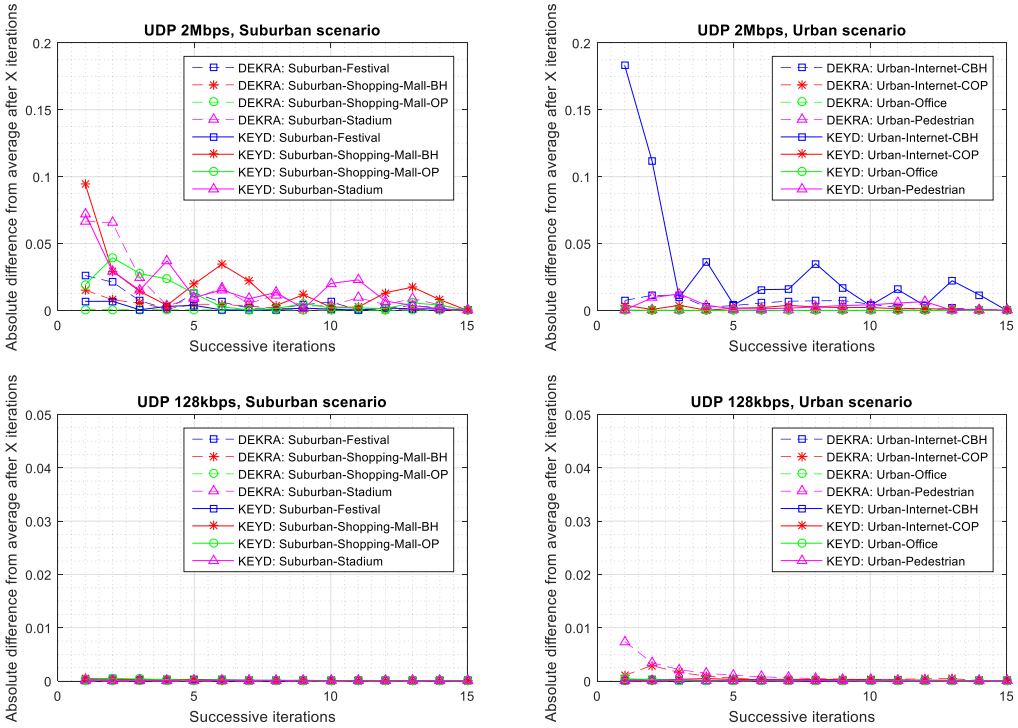

Figure 14: Convergence performance with UDP traffic in different network scenarios

3.2.2 YouTube Video Streaming

The configuration of YouTube video streaming used in the study is listed in Table 8.

Table 8. Configuration of YouTube

YouTube Configuration	Test duration in each iteration	KPI
1080p @ 30fps 1440p @ 30fps 1080p @ 60fps	300 s	Video resolution Re-buffering time

Figure 15 shows the average video quality (i.e., video resolution) and the total re-buffering time for YouTube video streaming under different network scenarios. The performance of the video quality is quite stable and the average value is stable within 5% variation after 10 iterations. However, the performance of the total re-buffering time has more fluctuations than the average video quality and therefore is the KPI that determines the number of needed iterations for the average to converge (e.g., more than 15 iterations). Again, the specific number of iterations is scenario dependent.

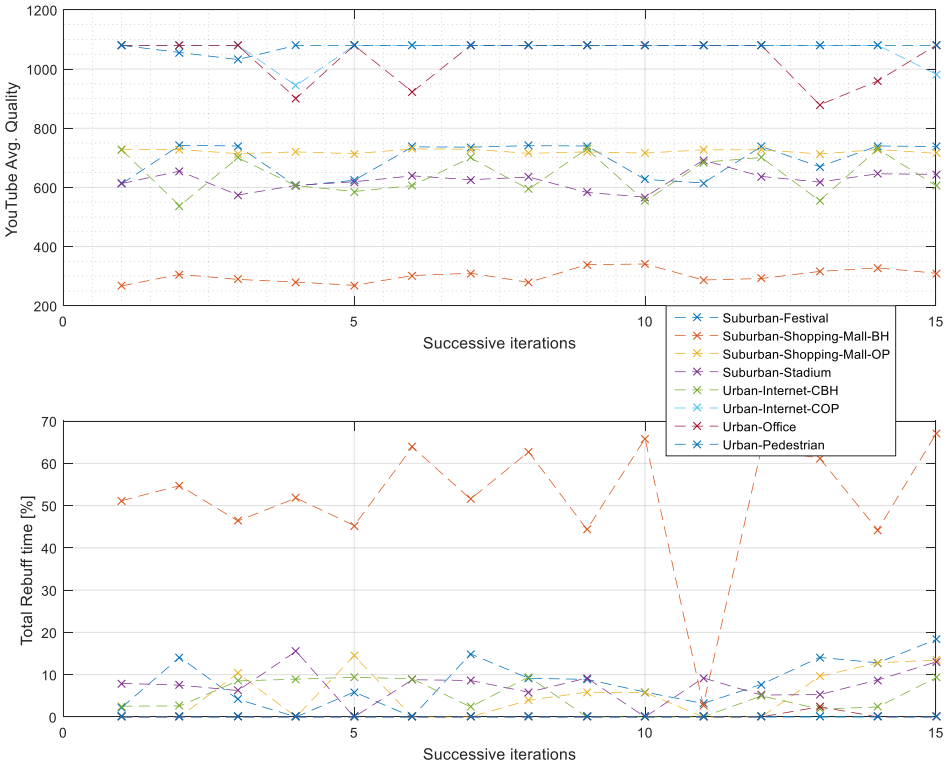

Figure 15: Average video quality and total re-buffering time in different network scenarios

3.2.3 Spotify Audio Streaming

The configuration of Spotify audio streaming used in the test is listed in Table 9.

Table 9. Configuration of Spotify

Spotify Configuration	Test duration in each iteration	KPI
HQ Audio (192 kbps)	300 s	Total audio re-buffering time

Figure 16 shows the total duration of the re-buffering time for the audio playback in different network scenarios. It is shown that across the tested network scenarios, Suburban-Shopping-Mall-BH performs the worst, which is expected due to the harsh channel condition and network configuration in this scenario. Furthermore, it can be observed that even though the measurements are fluctuating from one iteration to another, the average converges fast. The number of iterations needed to converge is again scenario dependent, e.g., 5 iterations for non-challenging network scenarios, and 10 iterations for challenging network scenarios.

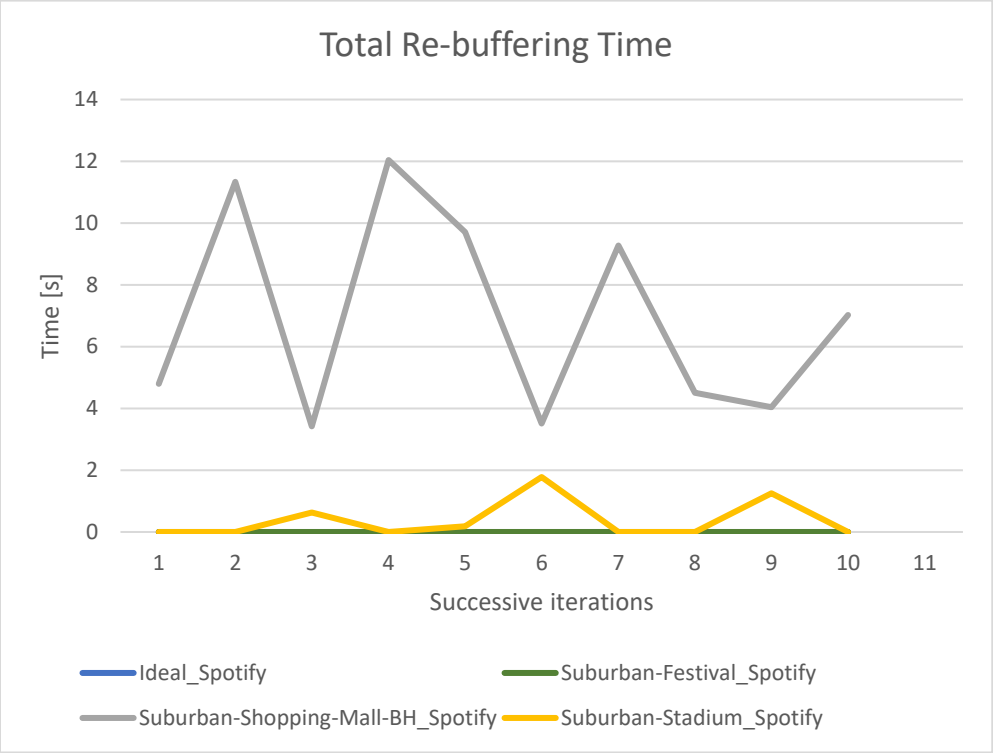

Figure 16: Total re-buffering time for Spotify audio streaming in different network scenarios

3.2.4 Voice MOS

The measurement setup for voice quality evaluation is illustrated in Figure 17. In this experiment, a voice quality analyser has been introduced in the validation environment. This equipment implements the ITU-T POLQA algorithm for objective MOS (Mean Opinion Score) evaluation. This element is external to the TRIANGLE test bed and provides extra measurement capabilities. Therefore, in the current study this equipment is considered as a calibration instrument for the characterization of the network scenarios implemented in TRIANGLE.

Skype app has been selected as reference application for this experiment. A video call has been the selected use case.

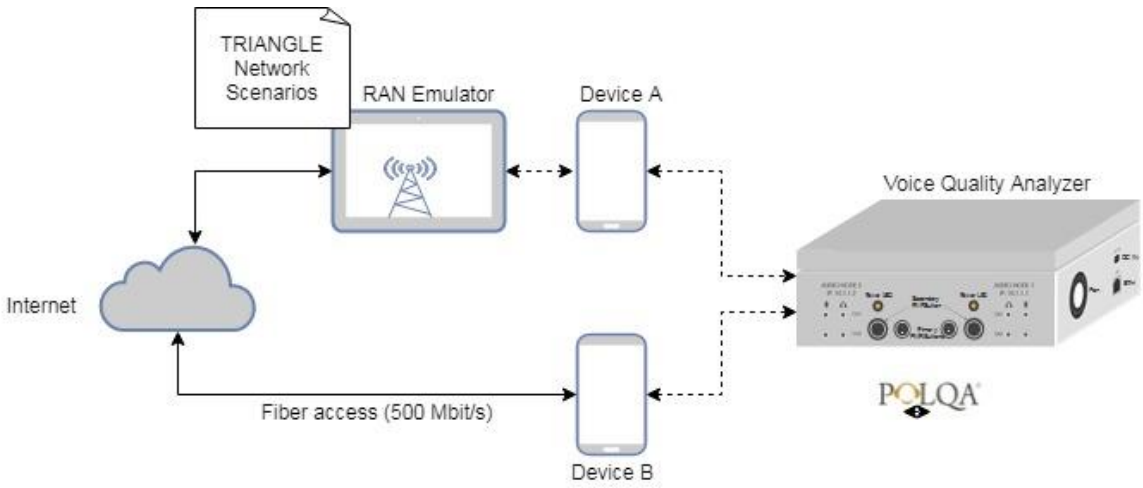

Figure 17: Voice quality evaluation setup

During the video call, if the camera does not capture any motion picture, the data rate for the video call is very low as the video codec tries to optimize the transmission with minimum data transmission rate. In order to force video data transmission over the air, the camera on both devices has been set aiming at a random light flashing source.

Figure 18 and Figure 19 show the MOS value (for the voice) and the voice delay, as perceived in each of the terminals, under different network scenarios. It can be seen that for the use case of a Skype video call, the measured results for all network scenarios are quite consistent across the test iterations. Therefore, less number of iterations, e.g., 5 iterations, are needed for the average to converge.

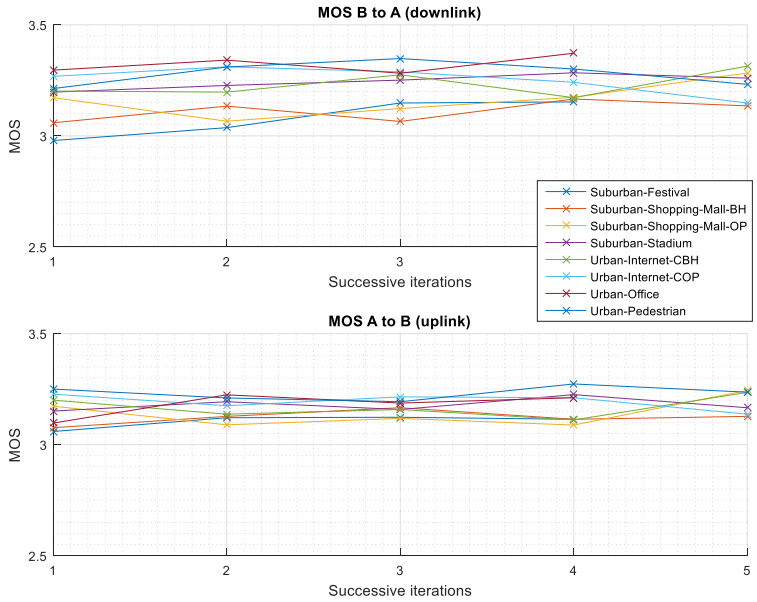

Figure 18: POLQA-WB MOS A to B (uplink) and B to A (downlink)

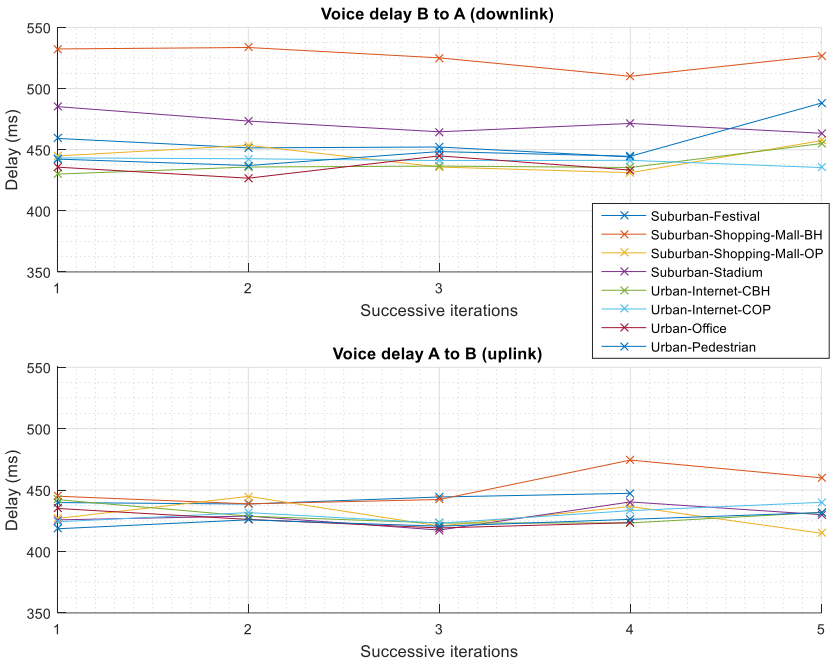

Figure 19: Voice delay A to B (uplink) and B to A (downlink)

3.2.5 Google Earth Virtual Reality

The measurement setup for VR app evaluation is illustrated in Figure 20. In this experiment, the robotic platform developed for the TRIANGLE test bed has been used.

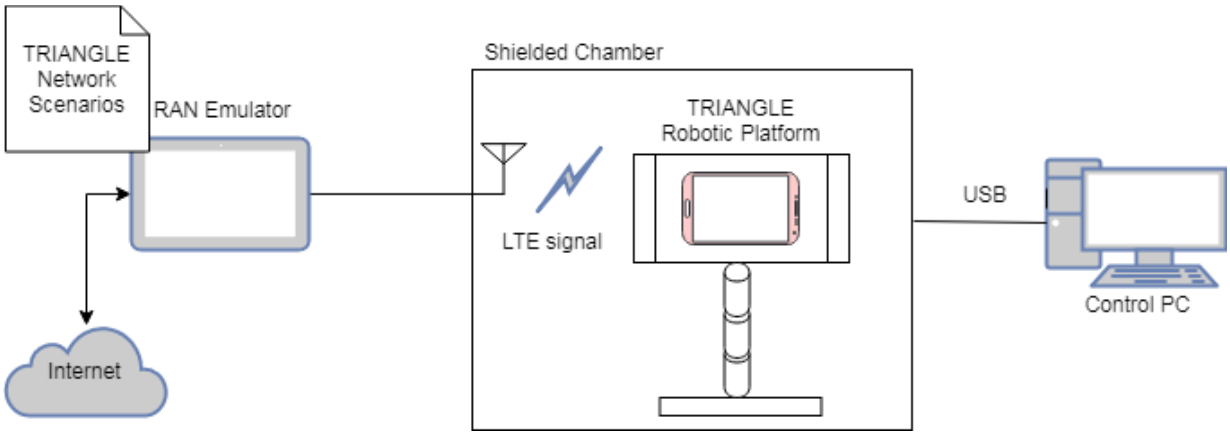

Figure 20: VR Evaluation Setup

For the other experiments described in this document the project team handcrafted the RF cabling between the test phone and the UXM unit. The link is clean and stable while the test phone stays in horizontal position and properly fixed in a platform. However, for VR experiments the phone must be held by the robotic platform which will move the phone in the three-

dimensional axis of space. Unfortunately, the RF cabling is not that robust and some RF leaks could occur, which would tamper the measurements and conclusions of this study. Therefore, the TRIANGLE project team has decided to conduct the experiments radiated inside a shielded chamber. This decision has some implications: (1) the channel models (introduced by UXM) do not work because the signals will now be uncontrolledly reflected inside the chamber and (2) there will be high insertion loss added to all scenarios. Taking aside the channel model and signal levels, the experiments have been mainly focused on the LTE scheduler of the network scenarios.

The following modification has been made in the network scenarios in order to conduct the VR experiments:

- Uplink modulation fixed to MCS15
- AWGN removed
- Channel models removed
- Minimum UXM transmission power level used in all scenarios fixed to -80 dBm. The signal levels still varies in the network scenarios but the lowest signal level has been limited to -80 dBm. This has been done because otherwise, the LTE link would break due to the insertion loss introduced by propagation over the air.

The automated test script consists of the following step sequences: (1) Open Google Earth App, (2) Browse through the app menus, (3) Click “start experience”, (4) Measure time to load virtual scenario. The test duration in each iteration is around 120 seconds. In parallel, the UXM runs the selected network scenario, which consists of several sub-scenarios each of which lasts for 30 seconds, and randomly selects a sub-scenario to begin with.

Figure 21 shows the measurement results obtained in different network scenarios. The KPI depicted in the Y-axis is the “time to load the virtual scenario”, which is defined as the elapsed time since the user click “start experience” until the virtual world is fully rendered in the test phone screen [7]. Each scenario was repeated 10 times, i.e., 10 iterations. There are some variations in the measured KPI, especially in some challenging network scenarios such as Suburban-Shopping-Mall-busy hours and Suburban-Festival. Because “time to load the virtual scenario” is only measured at the start of the App, the variation is caused by the random selection of the first sub-scenario in the selected network scenario, which means that the starting sub-scenario is not always the same in each test iteration.

For the use case of Google Earth Virtual Reality, the Suburban-Shopping-Mall-busy hours and Suburban-Festival scenarios provide unacceptable user experience (i.e., waiting time longer than ten seconds), while the rest of the network scenarios are fine. The number of iterations recommended for testing this use case in the TRIANGLE testbed is 10.

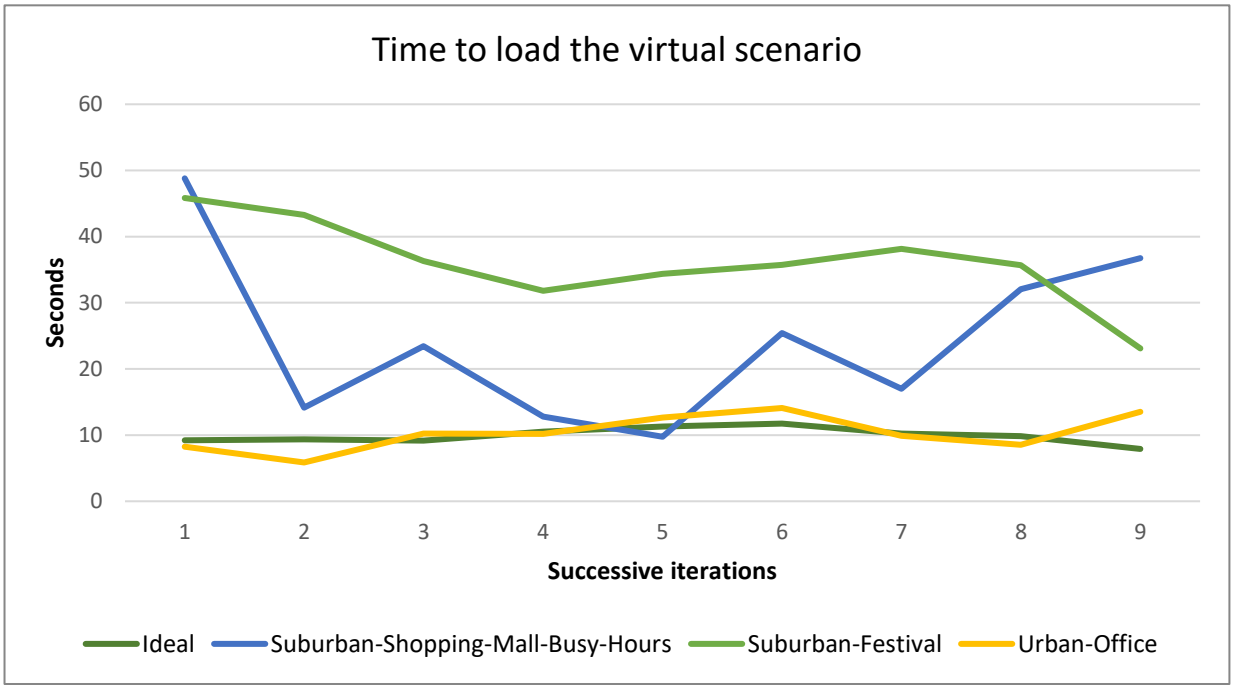

Figure 21: Time to load virtual world for Google Earth VR in different network scenarios

3.3 Summary

Because of the dynamics introduced in the network scenario, the measurement results may vary from one iteration to another. Some statistical analysis is needed to ensure that the testbed can provide consistent and repeatable measurement results before executing any test campaign. A comprehensive study on the convergence performance of the testbed has been presented in this chapter, covering various traffic profiles as well as network scenarios. The measurement results show that the number of iterations needed for the testbed to converge depends both on the traffic configuration and the specific network scenario. Even with the same traffic configuration, it is necessary to configure different number of iterations with respect to the configured network scenario. Nonetheless, a key finding is that the system does converge with feasible number of iterations, i.e., in most cases results converge after 15 iterations. Table 10 summarizes the recommended number of iterations for each tested traffic profile.

Table 10. Number of iterations recommended

Traffic profile	Test duration in each iteration	Number of iterations recommended
TCP	180 s	≥10
UDP with traffic load lower than channel capacity	180 s	≥10
UDP with traffic load higher than channel capacity	180 s	≥15
Video streaming	300 s	≥15
Audio streaming	300 s	≥10

Video call	300 s	≥5
Virtual reality	120 s	≥10

4. ETL Process Calibration

Once the testbed is calibrated (Chapter 2) and the number of test iterations needed for the testbed results average to converge has been determined (Chapter 3), the testbed is ready to generate measurement results for the defined test cases under specified network scenarios. As individual KPIs are measured in different dimensions and scales, they are normalized into a standard 1-to-5 scale (from lowest to highest), as typically used in mean opinion score (MOS), and are referred to as “synthetic MOS”. This process has been adopted by NGMN [9] and it is based on previous work carried out by ITU [10]. Once all KPIs are translated into “synthetic MOS” values, they can be weighted averaged. As the TRIANGLE QoE computation is based on the postprocessing of the synthetic MOS values in different domains, the correct transformation of KPIs into synthetic MOS values is of significant importance.

4.1 KPI to MOS conversion calibration

Each measured KPI is individually converted into synthetic-MOS value for homogeneous comparison and aggregation. The conversion is based on the normalization function and normalization parameters. The normalization function can be either linear interpolation or logarithmic interpolation depending on the measured KPI. The normalization parameters are KPI_min and KPI_max, which define the worst and the best reference benchmarking values of a specific KPI, respectively. The details of the KPI normalization process can be referred to [7].

For the calibration of KPI to MOS conversion, a test campaign is created for each test case with sufficiently large number of iterations (e.g., >20) in each supported scenario in order to get enough samples per KPI per scenario. Then the normalization function and normalization parameters can be extracted throughput statistical analysis (e.g., mean, deviation, cumulative distribution function, etc.) of the measured KPI samples with the objective to effectively convert the measured KPIs into a standard 1-to-5 scale. The converted MOS values should capture the behaviour of the measured KPI in different scenarios.

An example of KPI to MOS conversion calibration for Non-Interactive Playback test case is given, with test configurations listed in Table 11.

Table 11. Test configurations for KPI to MOS conversion calibration

Application under test	ExoPlayer
Reference test phone	Samsung Galaxy S7
Use case	Content streaming
Test case	Non-Interactive Playback
Number of iterations per scenario	25
Measured KPIs in the AUE domain	• Time to load first media • Video resolution • Playback cut-off • Content stall
Measured KPIs in the AEC domain	• Average power consumption
Measured KPIs in the RES domain	• Average CPU usage • Average memory usage

Figure 22 and Figure 23 show the measured KPIs and the corresponding normalized KPIs (i.e., synthetic MOS). The bars correspond to the measurement results obtained from different iterations. In this example, 25 iterations are configured for each scenario. The figures in the left are the measured KPIs in different domains, while the figures in the right correspond to the normalized KPIs by using the normalization function and normalization parameters outlined in Table 12. It can be seen that with the parameters listed in Table 12, the KPIs from raw measurements can be effectively mapped into synthetic MOS values for further comparison and aggregation.

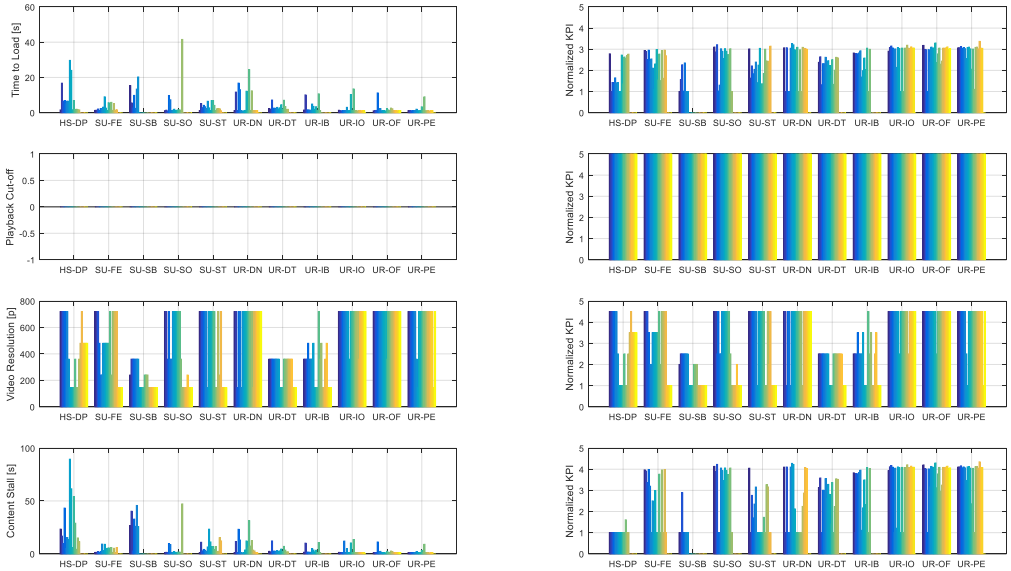

Figure 22: Measured KPIs and normalized KPIs (synthetic MOS) in the AUE domain under different scenarios.

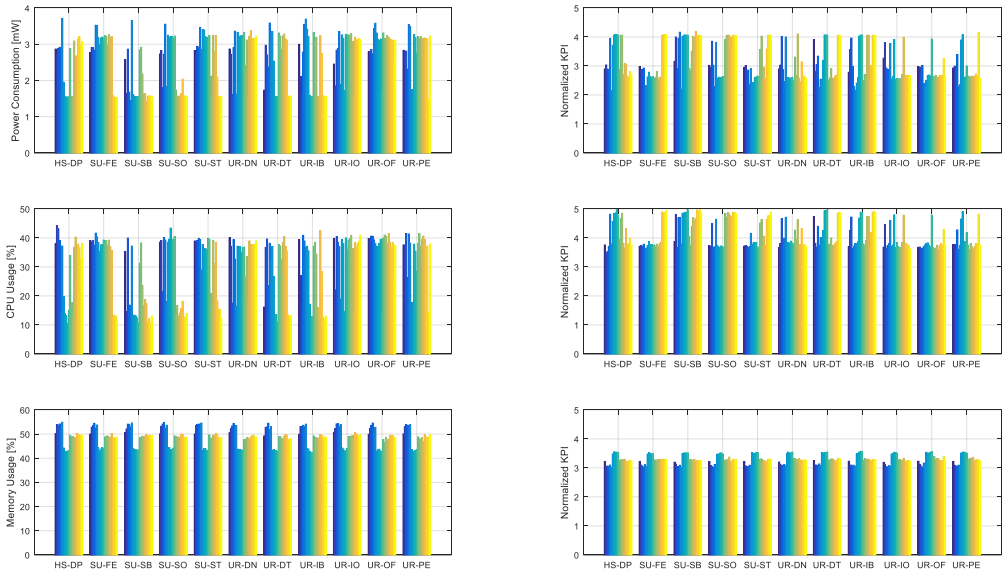

Figure 23: Measured KPIs and normalized KPIs (synthetic MOS) in the AEC and RES domain under different scenarios.

Table 12. KPI Normalization function and parameters for Non-Interactive Playback

Test case	Domain	KPI	Normalization	KPI_min	KPI_max
Non-Interactive Playback	User Experience	Time to load first media frame	Logarithmic	0.1 ms	20 ms
Non-Interactive Playback	User Experience	Playback Cut-off	Linear	0.5	0
Non-Interactive Playback	User Experience	Content stall time	Linear	0 s	5 s
Non-Interactive Playback	User Experience	Video resolution	Linear	144p	1080p
Non-Interactive Playback	Energy Consumption	Average power consumption	Linear	5 mW	0.5 mW
Non-Interactive Playback	Resource Usage	Average CPU usage	Linear	100 %	10 %

Non-Interactive Playback	Resource Usage	Average memory usage	Linear	100 %	10 %

4.2 Human panel calibration

An important pre-requisite before executing any test certification is the calibration of the ETL process, which is responsible for the conversion from the measured KPIs into the synthetic MOS values, and the computation of the QoE score in each measured domain. It is important to ensure that the testbed generated QoE score matches with the actual human perception.

As the assessment of QoE is quite subjective and there is no standardized calibration procedure for this process, we apply the human panel experiments as a proper approach to calibrate and verify if the testbed generated QoE score matches with the human experienced QoE.

As a case study, we selected ExoPlayer as the application under test. ExoPlayer is an open source media player originally developed by Google with support for DASH adaptive playbacks. This application falls into the use case of content streaming defined in the TRIANGLE project. There are 8 test cases defined in TRIANGLE for testing the use case of content distribution streaming service [8]. As an example, we only focus on the test case of "Non-Interactive Playback", which is related to the application user experience (AUE) domain. The test configurations and the measured KPIs are summarized in Table 13.

Table 13. Test Configuration for ExoPlayer

Application under test	ExoPlayer
Reference test phone	Samsung Galaxy S7
Use case	Content streaming
Test case	Non-Interactive Playback
Domain	Application user experience (AUE)
KPIs in the AUE domain	• Time to load first media (Video Start): The time elapsed since the user clicks play button until the media reproduction starts. • Video resolution (Video Quality): Used video resolution. • Playback cut-off (Video End): Probability that successfully started stream reproduction is ended by cause other than the intentional termination by the user. • Content stall (Smoothness): The elapsed duration of content stalls while playing the content.
Network scenarios	All supported scenarios (11 in total) [8]

The methodology for the human panel calibration is described as follows: A set of videos have been reproduced for each network scenario under the same testing conditions as in the testbed. The reproduced videos are of 2-3 minutes long and have random order with no presence of a reference clip. A group of people with diverse genders and ages are requested to watch the reproduced videos in a random order. A score sheet was given to each experimenter to give ratings in terms of synthetic MOS scores (a standard 1-to-5 scale, ranging from lowest to highest) with respect to each measured KPIs as well as an overall QoE score for each reproduced video. Details of the score sheet is given in Appendix I. The collected human panel experiment results for each measured KPI and the overall QoE score in each network scenario are averaged over the number of experimenters. The human panel experiment results are then compared with the testbed generated results for calibration. The calibration is based on the mean squared error (MSE) function with the objective to minimize the difference between the human experienced synthetic MOS score and the testbed generated synthetic MOS score.

Two rounds of human panel test have been conducted. The first round consists of 8 experimenters and the second round consists of 10 experimenters. Different videos have been chosen in the two rounds of test, but are reproduced under the same network conditions and are measured with the same KPIs listed in Table 13.

Figure 24 shows the comparison between the human experienced score and the testbed generated score before/after calibration for each measured KPI in different network scenarios. For simplicity, 4 out of 11 tested scenarios, i.e., Urban Office (UR-OF), Suburban Shopping Mall Busy Hours (SU-SB), Urban Driving Traffic Jam (UR-DT), and Suburban Festival (UR-FE), are selected to be shown in the figure, representing good, bad, and median network conditions. It can be seen that although the synthetic MOS scores generated by the testbed before human panel calibration (i.e., solid blue line) is quite similar to the human experienced score, there is still some deviation between the two. Therefore, a fine tuning of the ETL process with respect to the normalization function and normalization parameters is needed to ensure that the testbed generated score matches with the human experienced QoE.

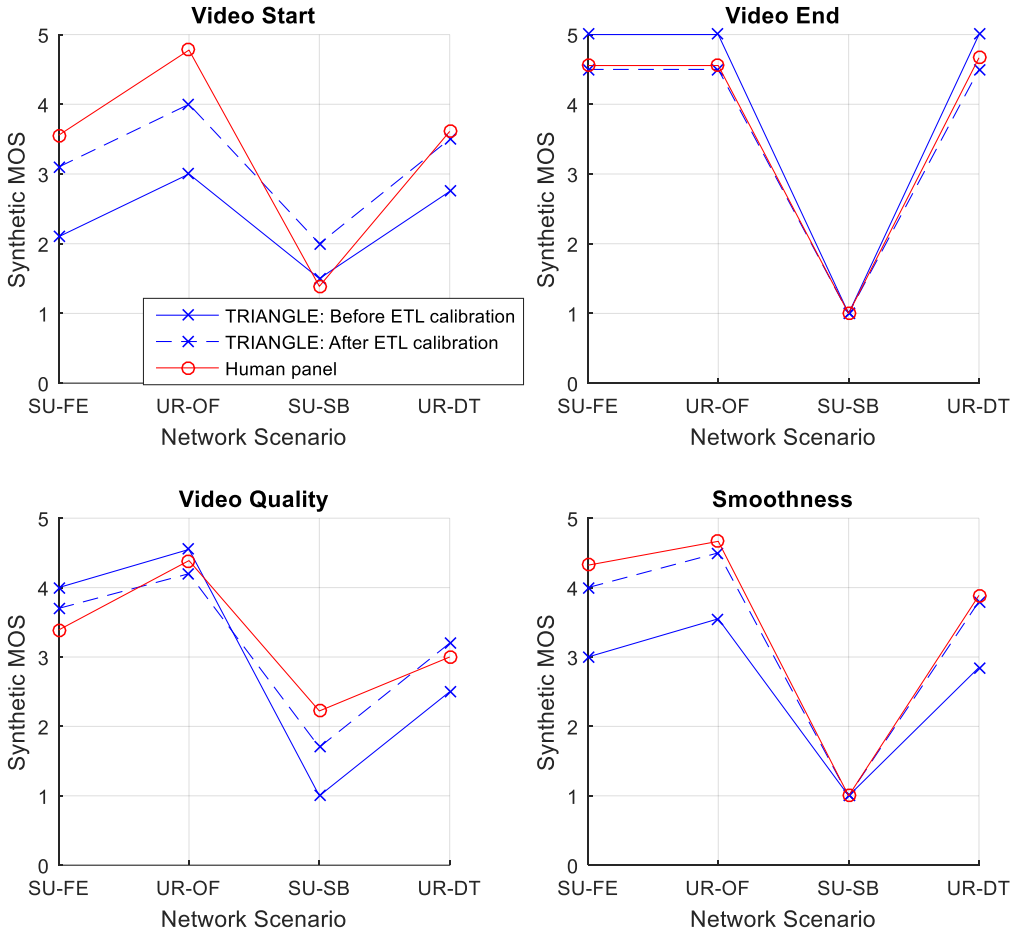

Figure 24. Comparison between the testbed generated score (before and after ETL calibration) and the human experienced score for each measured KPI in different network scenarios

In order to calibrate the ETL calculated synthetic MOS score (i.e., via normalization function and normalization parameters for each measured KPI) to match with the human perception, an objective function has to be defined. Here the objective function is defined as to minimize the mean squared error (MSE) between the human experienced synthetic MOS score and the testbed generated synthetic MOS score over all measured KPIs and scenarios, stated as:

$$\min_{f_k,\mathrm{KPI}_k^{\min},\mathrm{KPI}_k^{\max}} \sum_{s\in\mathrm{Scenarios}} \sum_{k\in\mathrm{KPIs}} \left(\theta_{k,s} - \mu_{k,s}\right)^2$$

$$\mu_{k,s} = f_k\left(\mathrm{KPI}_k^{\min}, \mathrm{KPI}_k^{\max}, \lambda_{k,s}\right) \quad (1)$$

where $\lambda_{k,s}$, $\mu_{k,s}$, and $\theta_{k,s}$ denote the measured KPI, the testbed generated synthetic MOS score via the ETL process, and the human experienced synthetic MOS score for KPI k under network scenario s, respectively. $f_k()$ and $\mathrm{KPI}_k^{\min}, \mathrm{KPI}_k^{\max}$ denote the normalization function and normalization parameters for a specific KPI k, respectively. The normalization function can be

linear/logarithmic interpolation, or other functions depending on the measured KPI. The normalization parameters define the worst and the best reference benchmarking values of a specific KPI, i.e., KPI_min and KPI_max.

The solution to the optimization problem mentioned in (1) can be found by using brute-force algorithm. As an example, for the considered "Non-Interactive Playback" test case described in Table 13, the calibrated ETL configurations of normalization functions and normalization parameters for each measured KPI are summarized in Table 14.

Table 14. Calibrated ETL configurations for KPI normalization

Test case	Domain	KPI	Normalization	KPI_min	KPI_max
Non-Interactive Playback	User Experience	Time to load first media frame	Logarithmic	0.1 ms	19.7 ms
Non-Interactive Playback	User Experience	Video resolution	Step	144p	1080p
Non-Interactive Playback	User Experience	Playback Cut-off	Linear	0.3	0.1
Non-Interactive Playback	User Experience	Content stall	Linear	1.1	8.2

The performance of calibrated testbed generated synthetic MOS scores for different KPIs and scenarios are plotted with dashed blue line shown in Figure 24. It can be seen that by fine tuning the ETL process for each measure KPI, the calibrated testbed generated synthetic MOS scores can better align with the curves obtained from the human panel experiments.

Once each measured KPI has been normalized, the domain evaluation can be performed by aggregating the associated KPIs through a domain-specific function (e.g., weight average). For example in the considered test case of "Non-Interactive Playback", the AUE domain synthetic MOS score for a specific network scenario can be obtained by averaging over the four measured KPIs, as shown in Figure 25(a). Furthermore, a single value representing the QoE score for the AUE domain can be obtained by averaging the domain synthetic MOS score over the tested network scenarios, as shown in Figure 25(b). Again, it can be seen that after ETL process calibration, the testbed generated synthetic MOS score matches quite well with human perception. Specifically, the accuracy of the QoE computation from the ETL process comparing to the human perception has been improved from 87% to 97%, in the AUE domain for "Non-Interactive Playback" test case.

It is worth mentioning that the QoE score obtained from human panel experiment is performed on desktop screens in a shadow office without much environmental noise, e.g., limited background noise, no strong sunlight reflection on the screen, in order to reduce the negative impact on the QoE scoring. In scenarios where environmental noises are not neglectable and may impact the human perception of QoE, a scaling factor ranging from 0 to 1 has to be

multiplied with the testbed generated QoE score (i.e., effective QoE) to match with human perception. The choice of the scaling factor depends on the severity of the environmental noise.

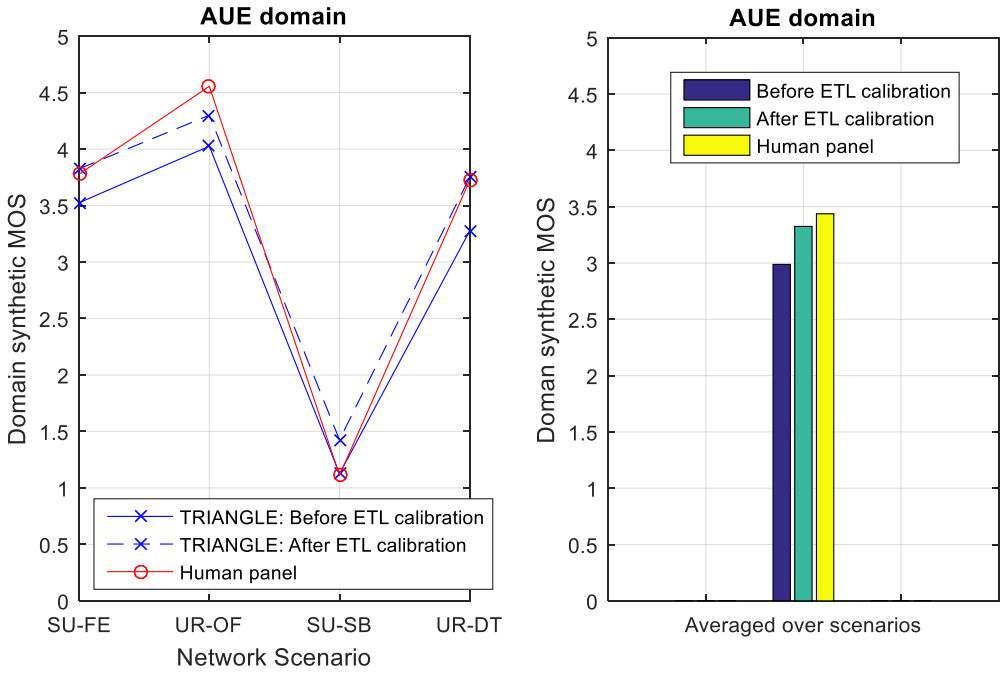

Figure 25. (a) AUE domain synthetic MOS score for specific network scenarios. (b) QoE score for the AUE domain

To summarize, in this section we have presented the framework for calibrating the ETL process with human panel test. We have selected test case “Non-Interactive Playback” which belongs to the content streaming use case as a case study, and is focused on the AUE domain. Two groups of people have been invited for the human panel test to calibrate the testbed generated QoE score with the actual human perception. It is demonstrated that the QoE score generated by the testbed matches quite well with the human perception. With fine tuning of the normalization functions and parameters, the accuracy of the testbed can be further improved.

Due to the lack of time, in this project we have only performed human panel calibration on one test case. But the presented calibration framework and the optimization objective function described in Eqn. (1) is valid for all other test cases. The only difference is that the measured KPIs may differ from one test case to the other. Another limitation in the current calibration work is that so far there is only limited number of participants in the human panel test (18 participants in total). As the measurement of QoE is quite subjective, a larger number of participants, mixed in gender and age, are necessary for the statistical analysis of the QoE score in different KPIs/domains. By having more human panel participants, the accuracy of the ETL conversion process can be improved with fine tuning of the normalization functions and parameter using Eqn. (1).

For the next step, it would be beneficial to: 1). Take a third round of human panel test for test case “Non-Interactive Playback” to check if the calibrated ETL process can better align with human perception, and to further calibrate the ETL process from the third round and compare the accuracy gain achieved from different rounds. Find out how many human experimenters are needed to calibrate the ETL process so that the testbed can generate converged QoE results

comparing with human perception. 2). Apply the human panel calibration in other use cases and test cases as well and fine tune the normalization functions and parameters for the related KPIs, with the findings obtained from step 1).

5. Case Study

Once the measurement instruments have been calibrated, the number of test iterations needed for the testbed results average to converge has been determined, and the ETL process has been calibrated, the testbed has been fully calibrated and is ready for running test certifications. This chapter presents the actual usage of the testbed to verify the measurement capabilities of the testbed and to benchmark the performance of the two-well-known content streaming mobile applications, namely Exoplayer and SkyTube.

5.1 Test case selected for content streaming

The applications under test (e.g., Exoplayer and SkyTube) fall into the use case of content streaming defined in the TRIANGLE project [7]. There are in total 6 test cases defined for content streaming. Here we only focus on the most relevant test case related to the user experience (AUE) domain, which is *Non-Interactive Playback* (i.e., video playing). More specifically, this test case corresponds to CS/001 in the TRIANGLE defined test cases [8]. Table 15 to Table 17 specifies the test conditions, the generic app user flow, and the raw measurements, which shall be collected during the execution of the test for test case CS/001 in three different domains (AUE/CS/001; AEC/CS/001; and RES/CS/001).

Table 15. AUE/CS/001 Test Case Specification

Identifier	AUE/CS/001 (App User Experience/Content Streaming/001)
Test case	Non-Interactive Playback
Objective	Measure the user experience KPIs by the AUT while executing the feature media file playing from the Content Distribution Streaming Services use case
Steps	1. The Test System commands the AUT to replay the Application User Flow 2.1: Play three reference media files. 2. The Test System measures the initial buffering, the number and duration of re-buffering occurrences and the video resolution.
Measurements (Raw)	• Time to load first media frame: The time elapsed since the user clicks play button until the media reproduction starts. • Playback Cut-off: Probability that successfully started stream reproduction is ended by a cause other than the intentional termination by the user. • Content Stall (s): The elapsed duration of content stalls while playing the content. • Video resolution: Used video resolution.

Table 16. AEC/CS/001 Test Case Specification

Identifier	AEC/CS/001 (App Energy Consumption/Content Streaming/001)
Test case	Non-Interactive Playback
Objective	Measure the energy that is consumed by an AUT while executing the feature media file playing from the Content Distribution Streaming Services use case.

Steps	1. The Test System commands the AUT to replay the Application User Flow 2.1: Play three reference media files. 2. The Test System measures the current consumed during the reproduction of the three reference videos.
Measurements (Raw)	• Current consumption: Record current samples during the measurement time and calculate the average current consumption.

Table 17. RES/CS/001 Test Case Specification

Identifier	RES/CS/001 (App Resource Usage/Content Streaming/001)
Test case	Non-Interactive Playback
Objective	Measure the usage of device resources of the AUT when executing the feature media file playing.
Steps	1. The Test System commands the AUT to replay the Application User Flow 2.1: Play three reference media files. 2. The Test System measures the use of Host Device resources during the reproduction of the three reference videos.
Measurements (Raw)	• **Playback average Memory usage**: Average amount of memory used during the measurement in MB. • **Playback average CPU usage**: Average percentage of CPU used during the measurement.

The measured KPIs for test case Non-Interactive Playback are listed in Table 12. As individual KPIs are measured in different dimensions and scales, they are normalized into synthetic-MOS values for homogeneous comparison and aggregation. The normalization function and parameters for each measured KPI are also outlined in Table 12.

5.2 Example detailed TRIANGLE QoE computation with ExoPlayer

ExoPlayer is an open source media player originally developed by Google with support for Dynamic Adaptive Streaming over HTTP (DASH) adaptive playbacks. DASH clients have the capability of adapting to the changing network conditions by choosing different video segment to download (videos are encoded at different bitrates). The ExoPlayer's default adaptation algorithm is basically throughput-based together with some other parameters controlling how often and when switching should occur. The test case configured for ExoPlayer is Non-Interactive Playback.

During the testing the testbed was configured with the different network scenarios defined in Table 6. In these scenarios, the network configuration changes dynamically following a random pattern, resulting in different throughput rates. The expected behaviour of the application under test is that it can adapt to the varying network conditions by decreasing or increasing the video resolution. However, the objective of the testing carried out in the TRIANGE testbed is not just to verify if the video streaming client can adapt to the network conditions, but also to check if this adaptation can improve the user experience.

Figure 26 shows the measured KPIs and the corresponding normalized KPIs (i.e., synthetic MOS) for one test case (Non-Interactive Playback) in the user experience domain (AUE/CS/001), obtained in Urban Office (UR-OF) scenario and Suburban-Shopping Mall Busy Hours (SU-SB) scenario. The UR-OF scenario corresponds to a scenario with relative good radio channel and network conditions, while the SU-SB scenario is one of the scenarios with more challenging radio channel and network conditions. The bars correspond to the measurement results obtained from different iterations. Note that the raw measurement KPI for playback cut-off is either 0 or 1, indicating if the video playback reaches the end or not. It can be seen that with the KPI normalization process listed in Table 12, the KPIs from raw measurements can be mapped into synthetic MOS values for further comparison and aggregation. Besides, it can also be observed that there are some fluctuations of the measurement results from iteration to iteration due to the variation of the radio channel conditions configured in each scenario. In this case study, the test case runs 15 iterations, as suggested in Section 3.3.

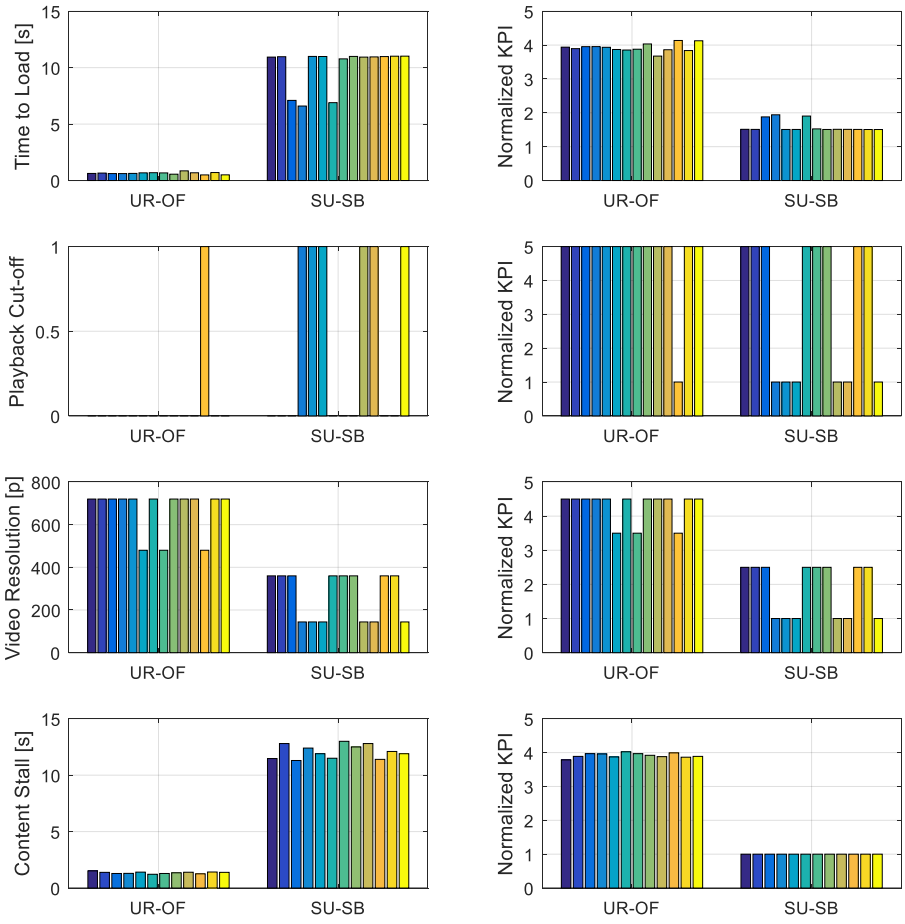

Figure 26: Measured KPIs and normalized KPIs (synthetic MOS) in the AUE domain of UR-OF and SU-SB network scenarios, obtained in different iterations.

The synthetic MOS values obtained from different iterations are then averaged. Figure 27 shows the averaged synthetic MOS values over iterations per KPI per network scenario in the user experience domain. The MOS values for video resolution, time to load, and content stall varies in different scenarios due to different radio channel and network conditions configured in the emulated base station, which leads to different RAN throughput in different scenarios. However, the MOS value for playback cutoff is always high, indicating that the video can be successfully played regardless of the scenarios. Overall, it can be concluded that the application under test (Exoplayer) is able to adapt to the changing conditions of the network, maintaining an acceptable user experience of video resolution, rebuffering times and time to load.

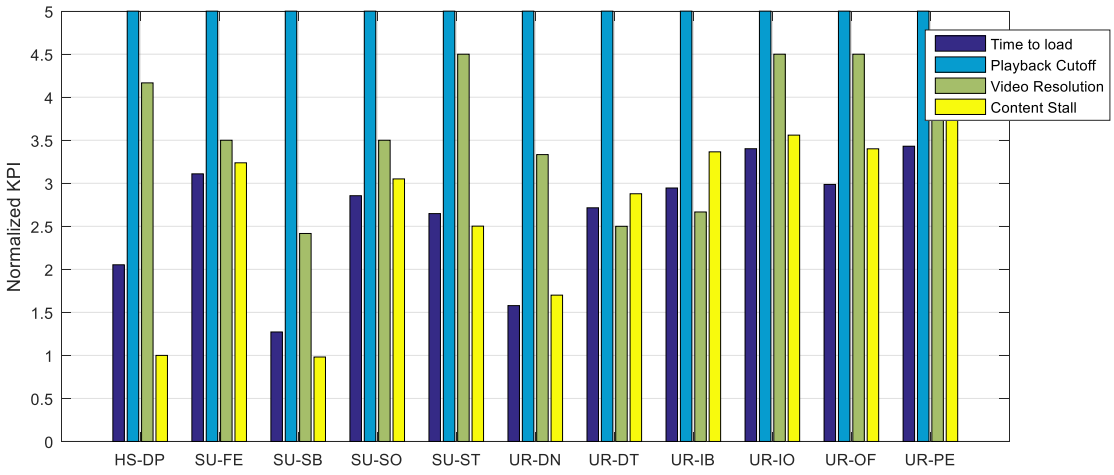

Figure 27: Synthetic MOS per KPI in the AUE domain per network scenario

From a tester's point of view, it is useful to have a single value representing the QoE score for each domain. By obtaining the synthetic MOS values per KPI per scenario in each domain, the synthetic MOS score for domain evaluation can be performed by aggregating the associated KPIs through a domain-specific function (e.g. the weighted sum). For example, the synthetic MOS score for the user experience domain per scenario is calculated by averaging over the four associated KPI from Figure 27.

Figure 28 shows the synthetic MOS score for user experience (AUE) domain in each network scenario. Scenarios with relatively good channel quality and network conditions show higher MOS score (e.g., UR-OF, UR-IO) while scenarios with relatively bad channel quality and network conditions show lower MOS score (e.g., SU-SB). Based on Figure 28, a single value representing the QoE score for the user experience domain can be obtained by averaging the synthetic MOS score over all network scenarios.

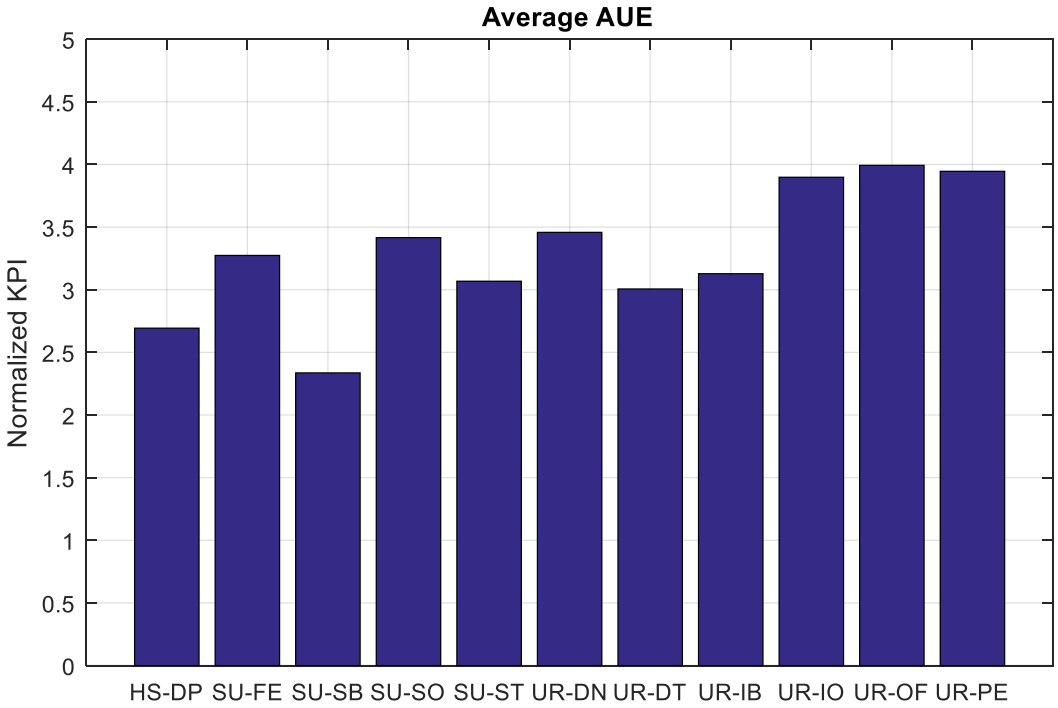

Figure 28: Synthetic MOS score in the AUE domain per network scenario.

Similar evaluations can be performed in other domains, such as energy consumption (AEC) domain and resource usage (RES) domain. Figure 29 shows the synthetic MOS score for the AEC domain and RES domain per network scenario. For test case Non-Interactive Playback, the synthetic MOS score in the AEC domain is somehow inversely proportional to the score in the AUE domain. That is because the energy consumption is closely related to the throughput, which depends on the configured network scenario. The scenario with better channel and network conditions means higher throughput, which results in better video playback quality but higher power consumption. The scores in the RES domain are correlated with the score in the AEC domain, but the variation is quite small.

A final synthetic MOS score in each domain is obtained by averaging the synthetic MOS score over all network scenarios. Figure 30 shows the spider diagram in three domains for the application under test. The scores in the AEC domain and RES domain are quite high, while the score in the AUE domain is lower due to the low synthetic MOS values obtained for time to load, video resolution, and content stall.

The final TRIANGLE mark for the Exoplayer can be obtained as a weight average over the measured domains, representing the overall quality of experience as perceived by the user. It is worth mentioning that in this case study, we only consider one test case “Non-Interactive Playback”. In case of multiple test cases, further aggregation over all test cases are needed.

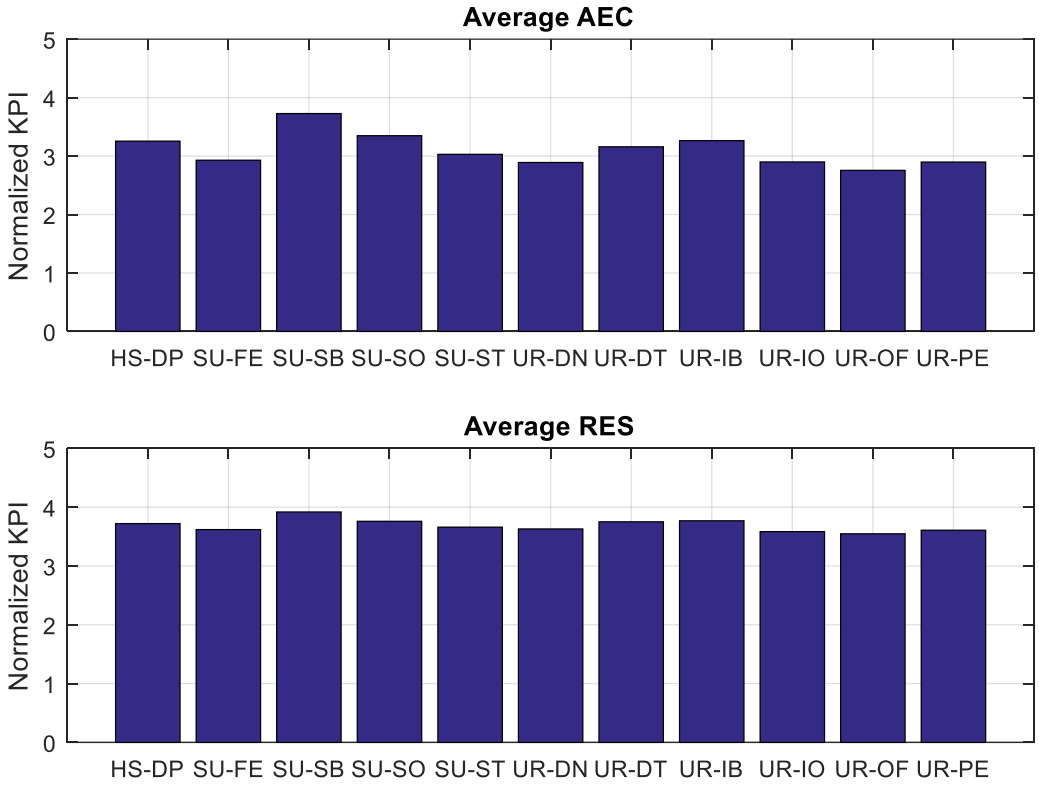

Figure 29: Synthetic MOS score in the AEC domain and RES domain per network scenario.

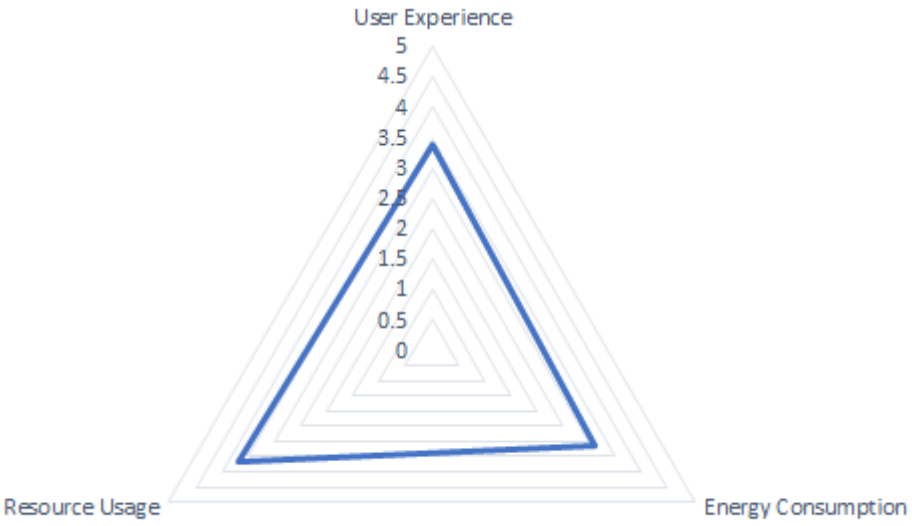

Figure 30: Spider diagram for the application under test

5.3 Mobile application under test

In this section we present the performance evaluations of two mobile applications (ExoPlayer and SkyTube) under two reference phones (Samsung S4 and S7). As both applications belong to the category of content streaming, the selected test case is Non-Interactive Playback with KPIs of interest listed in Table 18. Detailed analysis of each KPI is skipped in this section and we only focus on the overall performance analysis in each domain.

Table 18. Test configurations of application under test

Application under test	ExoPlayer, SkyTube
Reference test phone	Samsung Galaxy S4, S7
Use case	Content streaming
Test case	Non-Interactive Playback
Number of iterations	15
KPIs in the AUE domain	• Time to load first media • Video resolution (not available for SkyTube) • Playback cut-off • Content stall
KPIs in the AEC domain	• Average power consumption
KPIs in the RES domain	• Average CPU usage • Average memory usage

5.3.1 ExoPlayer

Figure 31 shows the averaged synthetic MOS score per network scenario in different domains for ExoPlayer under test. The application is tested with two reference phones, i.e., S4 and S7. If we look at the performance in the AUE domain, there is not much difference between the two reference phones, meaning that no big difference from user experience perspective. If we look at the AEC and RES domain, firstly the performance of energy consumption and resource usage is inversely proportional to the performance of user experience domain, due to the fact that the higher the throughput, the higher the energy consumption and resource usage. Secondly, it is obvious that the tested App consumes more energy and resource in S4 than in S7, probably due to different hardware and software systems in the two phones.

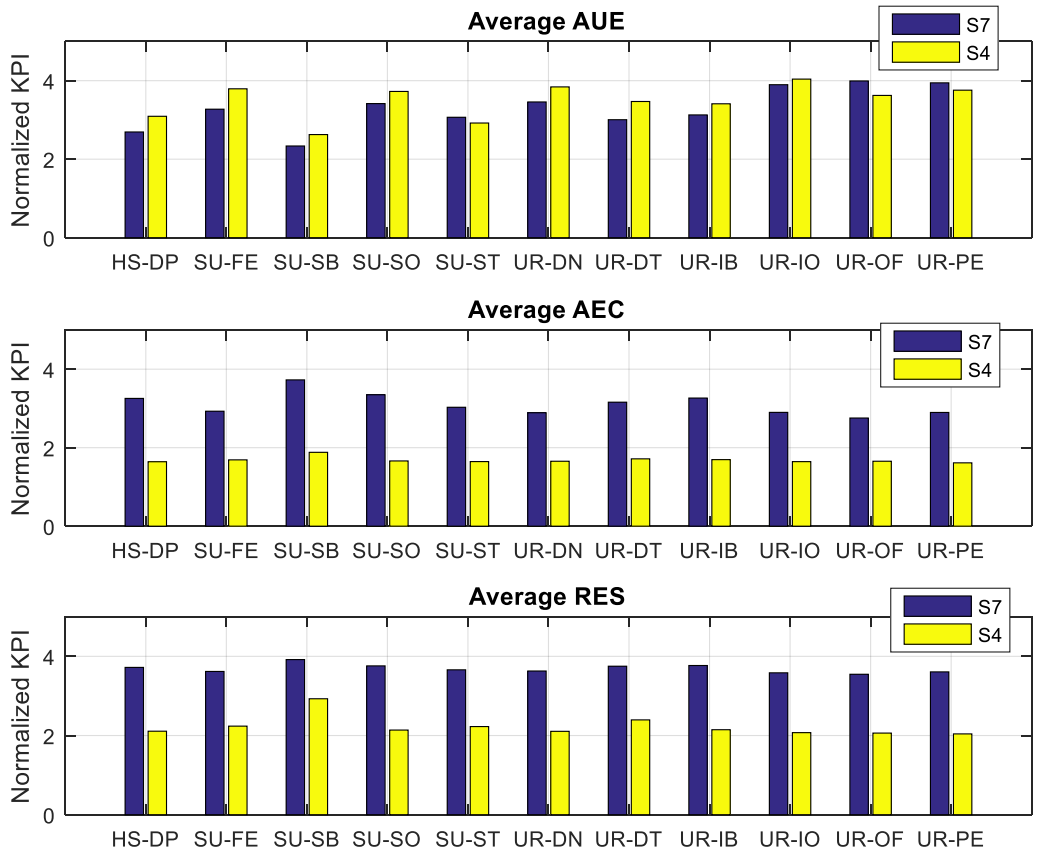

Figure 31: Synthetic MOS score per network scenario in different domains and different phones, for ExoPlayer under test.

Figure 32 shows the spider diagram of the averaged MOS score in different domains, for the application under test with different reference phones. It tells us that the ExoPlayer performs equally well in the AUE doman under the two reference phones, but consumes more energy and internal resources in S4 as compared to S7.

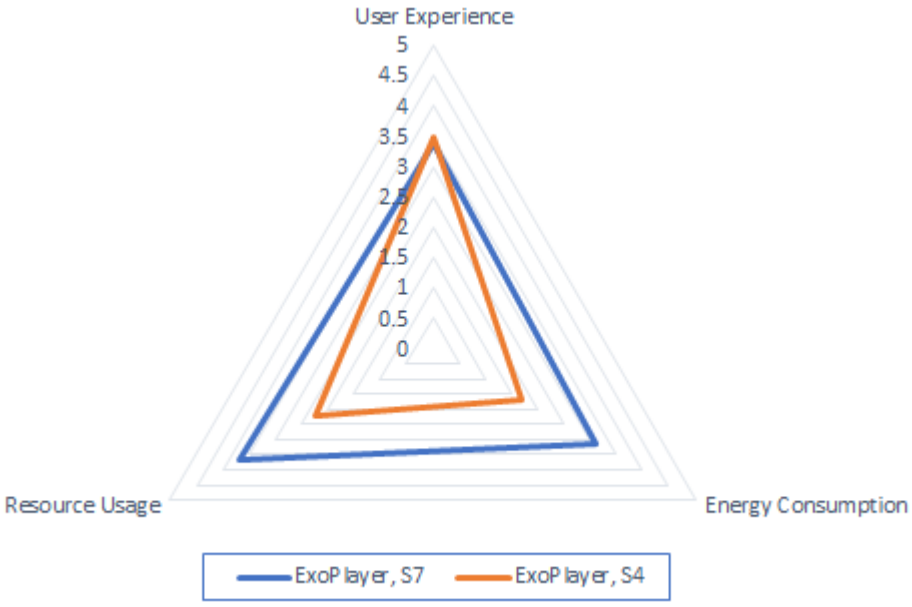

Figure 32: Spider diagram for the ExoPlayer tested on different phones

5.3.2 SkyTube

Figure 33 shows the averaged synthetic MOS score per network scenario in different domains and different reference phones, for SkyTube under test. Similar performance is observed in the AUE domain for the two reference phones. In the AEC and RES domain, less energy consumption and resource usage is also observed when using S7, but the difference is much smaller as compared to the ExoPlayer.

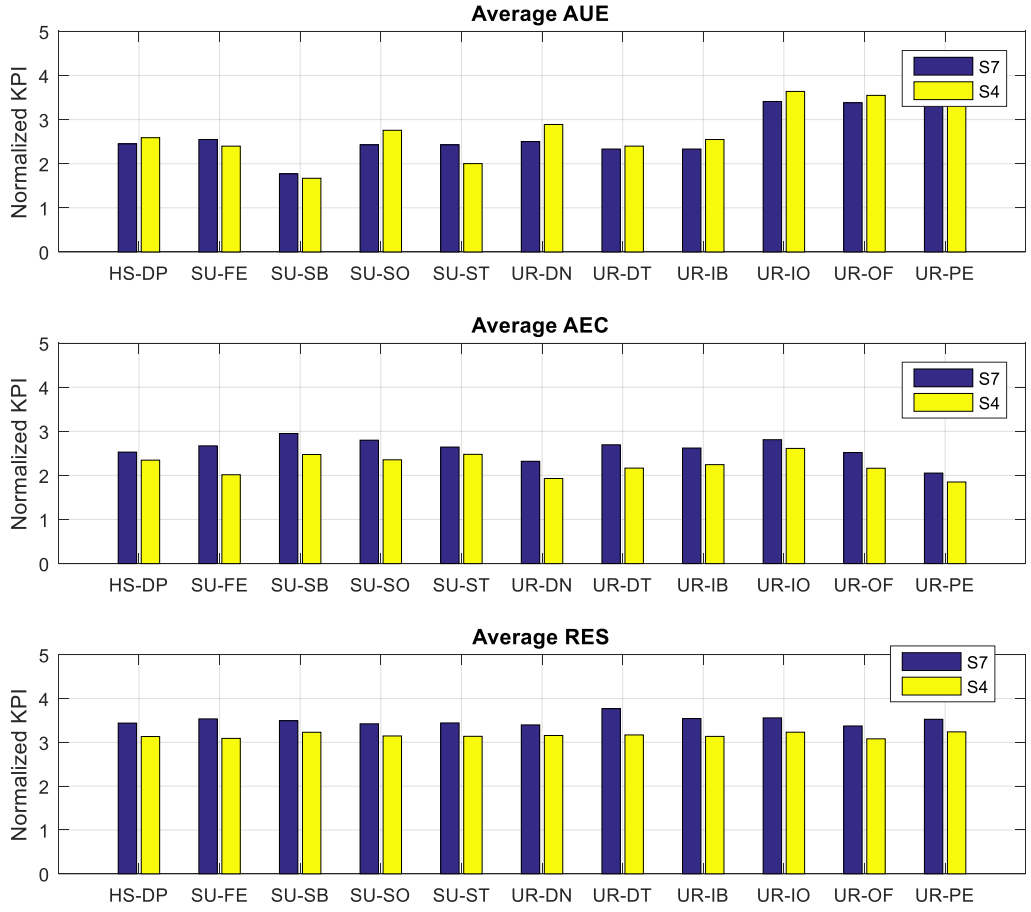

Figure 33: Synthetic MOS score per network scenario in different domains and different phones, for SkyTube under test.

Figure 34 shows the spider diagram of the averaged MOS score in different domains, for the application under test with different reference phones. It is shown that the SkyTube performs equally well in the AUE domain under the two reference phones, and exists marginally better performance in the AEC and RES domain when using S7, as compared to S4.

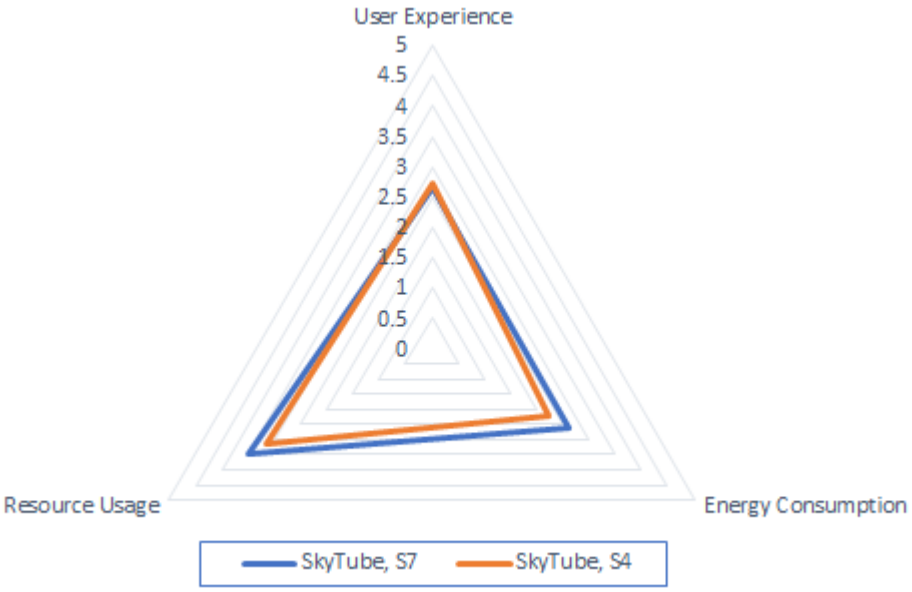

Figure 34: Spider diagram for the SkyTube tested on different phones

5.4 Summary

This chapter presents a case study of using the TRIANGLE testbed to evaluate the performance of two content streaming mobile applications, namely ExoPlayer and SkyTube. The test case selected is Non-Interactive Playback. The applications are tested in three domains (i.e., AUE (AUE/CS/001), AEC (AEC/CS/001), and RES (RES/CS/001)) under all supported scenarios (i.e., 11 scenarios). Each app is tested with two different reference phones (i.e., Samsung S4 and S7).

We first give a detailed analysis of ExoPlayer on S7, explaining how the raw measured KPIs are normalized into synthetic MOS values and how the normalized KPIs are averaged over iterations, over scenarios, over domains, and finally converted into the TRIANGLE mark. In the second part, we present the performance evaluations of ExoPlayer and SkyTube on S4 and S7. The results show that for ExoPlayer, similar performance is achieved in the AUE domain on both phones, but quite different performance is observed in the AEC and RES domain between the two phones. For SkyTube, similar observations are found, but with smaller difference in the AEC and RES domains between the two phones. The measurement results provide some insight into how the application performs in different domains and what potential improvement could be done for the application.

6. Conclusions

The objective of the TRIANGLE project is to develop a fully controlled E2E testbed that allows extensive laboratory testing of services against different use cases and scenarios, thus enabling E2E QoE evaluation for new mobile applications and devices in a repeatable manner. An important pre-requisite before executing any test campaign is to ensure that the testbed can provide accurate measurement results. This deliverable describes various steps required in the calibration of the testbed.

The TRIANGLE testbed is composed of various measurement instruments (e.g., UXM mobile network emulator, power analyser, commercial handset, etc.) as well as software tools (e.g., various measurement agents). The first step is to calibrate the individual measurement instrument. The calibration shall be done according to standard procedures by calibration laboratories with recognized traceability. The second step is to characterize the overall performance of the testbed in terms of latency and throughput, and to measure and compensate for the additional loss introduced by cabling. The third step is to characterize the performance of various software components (mainly the measurement agents). The impact of those software components shall be characterized and separated from the measurement results such as power consumption and resource usage. Once the above mentioned calibration procedures have been done, the outcome (e.g., a testbed calibration table and a device compensation table) shall be served as the baseline reference against which to calibrate the testbed and will be used as input to the ETL process in the KPI calculations.

Because of the dynamics introduced in the network scenario, the measurement results may vary from one iteration to another. Another important task in verifying the performance of the testbed is to ensure that the testbed can provide consistent and repeatable measurement results. The convergence performance of the testbed has been evaluated with various traffic profiles under all supported scenarios. The results show that different number of iterations are needed to obtain consistent results, while the exact number depends on the selected traffic profile and network scenario. A key finding is that the system does converge with feasible number of iterations, i.e., in most cases results converge after 15 iterations.

Another important calibration procedure is the calibration of ETL process, which is responsible for the conversion from measured KPIs to MOS values, and the calculation of the QoE score. Human panel experiments are employed at this stage to verify that the QoE score generated by the testbed matches the subjective perception from human experimenters.

Once the testbed is fully calibrated and the number of test iterations needed for the testbed results average to converge has been determined, two example content streaming mobile applications, namely ExoPlayer and SkyTube, have been tested. The results demonstrate the measurement capabilities of the testbed and provide some insight into how the application performs in different domains.

Appendix I. TRIANGLE Human Panel Validation Score Sheet

TRIANGLE Human Panel Validation

Video Quality Score Sheet

Rating	Label
5	Excellent
4	Good
3	Fair
2	Poor
1	Bad

Name (optional):

Age: ☐10-20 ☐20-30 ☐30-40 ☐40-50 ☐50-60 ☐>60

Sex: ☐Male ☐Female

Glasses: ☐Yes ☐No

Description

There are four videos to watch, please watch one video and rate the different categories and then proceed to the next video. You can use the rating table in this page to score the different categories from 1 to 5. If you don't understand one of the categories, please ask of leave it blank.

Category A) Please rate how satisfied from 1 to 5 you are with the time it took the video to start.

Category B) Please rate if the video ended correctly, e.g., 1 if the video didn't end or 5 if ended correctly.

Category C) Please rate the video image quality from 1 to 5.

Category D) Please rate how smooth the video was?

Video Start				Video End				Video Resolution				Smoothness			
#1	#2	#3	#4	#1	#2	#3	#4	#1	#2	#3	#4	#1	#2	#3	#4

Please also rate the overall score of the video, being 5 if the video playback was excellent (please do not take the debug information on the top of the image into account).

Overall Score			
#1	**#2**	**#3**	**#4**

References

[1] Keysight Technologies, “KS8400A Test Automation Platform 2017 Developer’s System Software – Technical Overview”, [Online] Available: http://literature.cdn.keysight.com/litweb/pdf/5992-1909EN.pdf?id=2796881.

[2] Keysight Technologies, “E7515A UXM Wireless Test Set Getting Started Guide”, [Online] Available: http://literature.cdn.keysight.com/litweb/pdf/E7515-90001.pdf?id=2459161.

[3] Keysight Technologies, “N6705 DC Power Analyzer and Source Measurement Unit (SMU) Modules – Product Fact Sheet”, [Online] Available: http://literature.cdn.keysight.com/litweb/pdf/5989-8615EN.pdf?id=1981218.

[4] DEKRA TACS4 Performance, website https://performance.tacs4.com/home/

[5] Quamotion Mobile app test automation, website http://quamotion.mobi/Mwc/Index

[6] EU H2020 TRIANGLE project, “Implementation report on the testing framework Rel 3 and specification of Rel 4”, Triangle Deliverable D3.4, [Online] Available: https://www.triangle-project.eu/wp-content/uploads/2018/04/TRIANGLE_Deliverable_D3.4_Final.pdf

[7] EU H2020 TRIANGLE project, “Formalization of the certification process, requirements and use”, Triangle Deliverable D2.2, [Online] Available: http://www.triangle-project.eu/wp-content/uploads/2017/07/TRIANGLE-D2.2-appendices.zip

[8] EU H2020 TRIANGLE project, “Final Test Scenario and Test Specifications”, Triangle Deliverable D2.6, [Online] Available: https://www.triangle-project.eu/wp-content/uploads/2018/11/TRIANGLE_D2-6.pdf.

[9] NGMN 5G White Paper, NGMN, Feb. 2015.

[10] ITU-T Recommendation P.10/G.100 (2006) Amendment 1.