H2020 5Growth Project
Grant No. 856709

D4.2: Verification methodology and tool design

Abstract

This deliverable constitutes the first one of a series of three, from D4.2 to D4.4, intended to provide the results of the project, addressing not only the verification of pilot outcomes in the context of the successive releases of the experimental environments, but also the establishment of a systematic way to perform this verification, guaranteeing verification quality and contributing these findings and tools for their application elsewhere.

Document properties

Document number	D4.2
Document title	Verification methodology and tool design
Document responsible	Diego Lopez, Sonia Fernandez (TID)
Document editor	Diego Lopez, Sonia Fernandez (TID)
Editorial team	Álvaro Gomes (ALB) - Engin Zeydan, Josep Mangues, Jordi Baranda, Ricardo Martínez, Luca Vettori, Manuel Requena, Lorenza Giupponi (CTTC) - Hugo Martins, Aldo Trindade (EFACEC_E) - Paulo Paixão, Rui Antunes, Pedro Elísio (EFACEC_S) - Diego San Cristóbal, Fernando Beltrán (ERC) – Daniel Corujo, Diogo Gomes, João Paulo Barraca, João Alegria, João Fonseca, Vitor Cunha (ITAV) - Konstantin Tomakh, Denis Kucherenko (MIRANTIS) - Olivier Tilmans (NBL) – Nikolaos Maroulis, Sokratis Barmpounakis, Nikolaos Koursioumpas (NKUA) - Juan Brenes, Giada Landi (NXW) - M. Ajmone Marsan, C. Casetti, C. F. Chiasserini, C. Puligheddu (POLITO) - Koteswararao Kondepu, And rea Sgambelluri, Luca Valcarenghi (SSSA) - Paola Iovanna, Giulio Bottari, Stefano Stracca, Fabio Ubaldi (TEI) - Aitor Zabala, Manuel Jiménez (TELCA) - Diego Lopez, Sonia Fernandez (TID) – Carlos Guimarães, Jorge Martín, Carlos Bernardos, Sergio González (UC3M)
Target dissemination level	PU
Status of the document	Final
Version	1.0
Delivery date	November 30, 2020
Actual delivery date	November 30, 2020

Production properties

Reviewers	Ricardo Martínez (CTTC)

Disclaimer

This document has been produced in the context of the 5Growth Project. The research leading to these results has received funding from the European Community's H2020 Programme under grant agreement Nº H2020-856709.

Contents

List of Figures

List of Tables

List of Acronyms

AGV – Automated Guided Vehicle

API – Application Programming Interface

AR – Augmented Reality

CKPI – Core 5G KPI

CMM – Coordinate-Measuring Machine

CP – Control Plane

CPE – Customer Premises Equipment

CUPS – Control and User Plane Separation

DSS – Decision Support System

E2E – End-to-End

eMBB – enhanced Mobile Broadband

FR – Functional Requirements

I4.0 – Industry 4.0

IIoT – Industrial Internet of Things

KPI – Key Performance Indicator

LV – Low-Voltage

LX – Level Crossing

mMTC – massive Machine Type Communications

MTTR – Mean Time To Repair

NS – Network Service

RAN – Radio Access Network

SKPI – Service KPI

SLA – Service Level Agreement

SO – Service Orchestrator

TMV-WG – Test, Measurement and KPI Validation Working Group

UC – Use Case

UE – User Equipment

UP – User Plane

URLLC – Ultra-Reliable Low Latency Communication

vEPC – virtual Evolved Packet Core

VIM – Virtualized Infrastructure Management

VM – Virtual Machine

VNF – Virtual Network Function

VPN – Virtual Private Network

VS – Vertical Slicer

ZDM – Zero Defect Manufacturing

Executive Summary and Key Contributions

This deliverable constitutes the first one of a series of three, from D4.2 to D4.4, intended to verify pilot results, aligned with the successive releases of the experimental environments developed in WP3 and their integration of the innovations addressed by WP2. These pilot results are verified in a systematic way, supported by a methodology and a series of tools reported here as well.

This deliverable is based on and refines the analysis work performed in the first WP4 deliverable, comprising the identification of core 5G KPIs, the definition of general service KPIs associated to vertical functional requirements, the mapping between the identified core KPIs and the service KPIs.

The main contributions of this deliverable can be listed as:

- Refinement of the 5Growth Service and Core KPIs, and the mapping between the two sets of KPIs.
- Identification of the KPIs addressed by the different 5Growth pilots in the first validation campaign, including a description of how they are measured.
- Description of the measurement components to be used for verification, including the internal and external tools applicable.
- Design of an experiment catalogue focused on guaranteeing their reproducibility and repeatability.
- Analysis of the 5Growth data infrastructure, intended to collect, process, and publish the data produce by the project validation campaigns.
- Definition of metadata formats for the integration of heterogenous data sources, data aggregation mechanisms and data consumers.
- Analysis of the integration patterns for the data infrastructure, including metadata, with the other components of the 5Growth architecture.
- Report the results of the first validation campaign performed by each pilot, including measurements and lessons learned so far.
- Establish a general format for successive WP4 deliverables (D4.3 and D4.4).

The structure of D4.2 provides the general pattern for further contents in D4.3 and D4.4, that will be arranged around the same idea of combining reports on the evolution of methodology and tooling, and the analyses of the verification results. WP4 foresees these contents will evolve during the project lifetime, dedicating more space and detail to result verification as both the methodology matures, and the experimental capacity of the different pilots consolidates.

1. Introduction

WP4 main objective is to evaluate core (network) and service (application) KPIs through 5Growth field trials, validating the applicability of 5G technologies to the different use cases considered by the project pilots. These does not only include the execution of the different verification campaigns, but also the definition of the measurement and testing methodologies applied in the campaigns. WP4 will provide a detailed description of these methodologies and a series of tools to enable the automation of the verification procedures, the processing of the measurement data and the reproducibility of the verification results. This way, WP4 is committed not only to perform pilot verification, but also to do so in a systematic way. The latter would guarantee scientific (engineering) quality and contribute these findings and tools for their application elsewhere.

The work in WP4 is aligned with WP2 and WP3 activities, addressing three innovate/deploy/validate cycles during the project lifetime. WP4 applies the appropriate releases produced by WP3, that incorporate the modules produced in WP2, and selects the applicable experimental environment for the corresponding cycle, comprising:

- Data sources and consumers, including the metadata for them
- External tools to be applied, available through the project tool catalogue
- Specific measurement methods, documented in the deliverables

Pilot use cases are validated in the context of the successive releases of the experimental environments, and reported in the associated deliverable, including the environment characteristics and the outcome of the verifications run during the period.

As the first deliverable in a series of three, focused on reporting verification results, the structure of D4.2 provides the general pattern for further contents in D4.3 and D4.4, that will be arranged around the same idea of combining reports on the evolution of methodology and tooling, and the analyses of verification results. The verification methodology and associated tooling provide the common substrate to the execution of verification campaigns, working to:

- Enhance the quality of verification campaigns.
- Guarantee repeatability and reproducibility as essential features of experimental reports.
- Contribute to a better understanding of the mechanisms to aggregate and process telemetry data for data-enabled network management.

The validation campaigns are intended to collect evidence about the fulfilment of the core 5G and service KPIs in the contexts of the different pilots, analyzing the collected data and producing reports to verify the applicability of 5G technologies in the associated scenarios.

Therefore, this deliverable is structured as follows:

1. A section on the measurement components to be used for verification. It starts with a review of the KPIs already identified. It continues with a discussion on the methods applied by the different pilots and a description of the external tools used for performing (or deriving) the required measurements. It finishes with the design of an experiment catalogue.

2. An analysis of the components of the 5Growth *data infrastructure*, i.e. the supporting infrastructure for collecting, processing and making available the data produced by the project validation experiments. The data infrastructure includes heterogenous data sources, data aggregation mechanisms and data consumers, together with the metadata gluing these components.
3. A discussion on how this data infrastructure can be integrated with the other components of the 5Growth architecture and the different ICT-17 sites.
4. The results of the validation performed at the time of writing, with one dedicated sub-section per pilot. These sections describe those measurements that could be performed for the available use cases, including the experience and lessons learned so far. Successive WP4 deliverables (D4.3 and D4.4) will bring an increasing contribution from these tasks, as the project platform matures.

2. Measurement components

Section 2 presents in detail the measurement components involved in the 5Growth platform, departing from the work presented in deliverable 4.1 (D4.1) [1]. The section starts with a detailed description of the association between the core and service KPIs. Additionally, it explains the measurement procedures where each pilot presents the adopted methodology for measuring and validating the KPIs. Finally, Section 2 describes the tools and presents the measurement tool catalogue used by 5Growth to measure both the core and service KPIs.

2.1. Detailed Core KPIs and Service KPIs associations

This section details the associations between the Core KPIs and the Service KPIs identified in D4.1. Recalling from D4.1, Service KPIs are high-level abstractions of the business requirements and are not directly measurable. An association between both Service and Core KPIs was established to guarantee a Service KPI by measuring their corresponding Core KPIs. Table 1 summarizes the mapping between 5Growth identified Service KPIs and Core KPIs, applied in all the 5Growth pilots. Table 1 does not provide the methodology definition of such associations. However, it provides the supporting material to further explain each association in more detail. For a more detailed discussion, D4.1 provides a reference of each Core and Service KPI entailing its meaning, description, and/or motivation.

Since D4.1 publication, some minor changes have been applied to the association between Core and Service KPIs. Table 1 is updated to reflect such modifications, where the removal of an association is marked with a minus symbol (-), and an addition of a new association is marked with a plus symbol (+). Additionally, CKPI-4 has been removed from the list of Core KPIs. Specifically, this is related to Coverage, which is determined as an unmeasurable Core KPI clashing with 5GR-SKPI-7. Moreover, CKPI-10 and CKPI-11 have been also removed from the Core KPI list. These metrics are measured solely at the user terminals, based on application-specific monitoring probes for each deployment. Besides, the added value of measuring both Core KPIs with respect to the other network-oriented Core KPIs is negligible. Furthermore, two additional Core KPIs (CKPI-2 and CKPI-3) substituting the removed KPIs (CKPI-10 and CKPI-11) are included into the measurement of 5GR-SKPI-3.

TABLE 1: MAPPING BETWEEN 5GROWTH SERVICE KPIS AND CORE 5G KPIS

5GR-SKPIs	Core 5G KPIs										
	CKPI-1	CKPI-2	CKPI-3	CKPI-4	CKPI-5	CKPI-6	CKPI-7	CKPI-8	CKPI-9	CKPI-10	CKPI-11
5GR-SKPI-1					X	X					
5GR-SKPI-2	X	X									
5GR-SKPI-3		+	+	X	X					X	X
5GR-SKPI-4		X	X					X	X		
5GR-SKPI-5	X				X						
5GR-SKPI-6		X	X		X						
5GR-SKPI-7	X	X	X	X	X		X				
5GR-SKPI-8	+	X	X								
5GR-SKPI-9	-		X		X						
5GR-SKPI-10					X						
5GR-SKPI-11	X		+								

The following tables explain how the mappings between Service and Core KPIs in Table 1 are derived, providing details about the relationship of each Service KPI with its associated Core KPIs.

2.1.1. Service Setup Time (5GR-SKPI-1)

Deriving the service setup time from a vertical service can be achieved by measuring two Core KPIs: Availability (CKPI-5) and Slice Creation/Adaptation Time (CKPI-6). Table 2 shows the measurement units of this Service KPI.

TABLE 2: SERVICE SETUP TIME UNIT

Service KPI	Measurable Unit
Service Setup Time	Minutes (min)

The Slice creation or adaptation time impacts directly on the Service KPI, as the service setup time depends on the creation or adaptation of the slice where the vertical service will be accommodated. It could be measured by probing the platform service orchestrator slicing component. Secondly, once the slice where the vertical service is deployed, instantiated, or adapted, the time the vertical service takes in deploying its components is measured with the Availability Core KPI, by probing the availability of all the components. Consequently, the maximum downtime during the setup period is taken.

2.1.2. Synchronization between communication components (5GR-SKPI-2)

Inferring the Synchronization capability between communicating components in a vertical service can be achieved by measuring two Core KPIs for each of the communication links: End-to-end latency (CKPI-1) and Packet loss (CKPI-2). Table 3 shows the measurement units of this Service KPI.

TABLE 3: SYNCHRONIZATION BETWEEN COMMUNICATION COMPONENTS UNIT

Service KPI	Measurable Unit
Synchronization between communication components	QoE metric – Bad/Acceptable/Good

End-to-end latency is measured across the whole communication path to assure that the delay in the traffic flows is always lower to the maximum end-to-end latency required for a correct synchronization. Moreover, to maintain the desired synchronization, the Packet loss Core KPI is also measured to avoid exceeding the defined packet loss tolerance since the synchronization protocols automatically downgrade the transmission speed when packet loss increases. This in turn directly impacts on the synchronization time.

2.1.3. Device Mobility (5GR-SKPI-3)

Deriving the device mobility capability of a vertical service can be achieved by measuring three different Core KPIs, namely, Packet Loss (CKPI-2), Guaranteed Data Rate, and Availability (CKPI-5). Table 4 shows the measurement units of this Service KPI.

TABLE 4: DEVICE MOBILITY UNIT

Service KPI	Measurable Unit
Device Mobility	Boolean: True or False

Packet Loss and Guaranteed Data Rate directly affects the Mobility as the end-user devices can only move inside areas with acceptable network service, i.e., those with a minimum level of packet loss and data rate. When the minimum values for these two Core KPIs are not satisfied, it could indirectly affect the quality of service when moving across antennas, directly impacting the mobility capability of the end-user. Additionally, the availability of the networks traversed can also impact the device mobility. Specific network services demand different availability requirements, limiting their operation to areas where their minimum availability can be guaranteed.

2.1.4. High-resolution Real-time Video Quality (5GR-SKPI-4)

Inferring high-resolution real-time video quality for a vertical service can be achieved by measuring four different Core KPIs: Packet Loss (CKPI-2), Guaranteed Data Rate (CKPI-3), Data Volume (CKPI-8) and Jitter (CKPI-9). Table 5 shows the measurement units of this Service KPI.

TABLE 5: HIGH-RESOLUTION REAL-TIME VIDEO QUALITY UNIT

Service KPI	Measurable Unit
High-resolution Real-time Video Quality	QoE Ranking (1 lowest to 5 highest quality)

Real-time video requires monitoring the packet-loss, the minimum guaranteed data rate, the available data volume, and the jitter. Specifically, real-time video is limited by the minimum data rate necessary for the video transmission at a certain quality. The minimum data rate is defined to avoid possible queuing or dropping of video frames which degrade the quality of the video transmitted. Moreover, the jitter and the packet loss can also directly affect the quality of the real-time video, as the variation between frame latency and retransmissions caused by packet loss. This typically results in video discontinuity [2]. Additionally, the volume of data available to the application can also impact the quality of the real-time video, as the available data volume is essential to determine at which quality a video can be streamed.

2.1.5. Radius of Operation (5GR-SKPI-5)

The radius of operation is directly impacted by two different Core KPIs: Latency (CKPI-1) and Availability (CKPI-5) of the end-to-end communication service. Table 6 shows the measurement units of this Service KPI.

TABLE 6: RADIUS OF OPERATION UNIT

Service KPI	Measurable Unit
Radius of Operation	Kilometres (km)

For a vertical service determining its operational radius, the required minimum latency must be lower than the minimum latency provided inside the radius. Moreover, the required availability of the

service also impacts in the radius of the operation. Specifically, higher radius networks are more complex since more connection points are involved which does increase the probability of errors. This may lead to reduce the availability and thus reducing the radius of operation of the vertical service.

2.1.6. Integrated Multitype Communications (5GR-SKPI-6)

The integrated Multitype Communications is directly impacted by three different Core KPIs; Packet Loss (CKPI-2), Guaranteed data rate (CKPI-3) and Availability (CKPI-5). Table 7 describes the units this Service KPI is measured.

TABLE 7: INTEGRATED MULTITYPE COMMUNICATIONS

Service KPI	Measurable Unit
Integrated Multitype Communications	Fail or Success

The 5GR-SKPI-6 is influenced by multiple Core KPIs due to the broad definition of this Service KPI. When multiple types of communication and protocols can coexist together, it is required to first individually characterize whether the communications and protocols can share a specific communications channel. Therefore, as the packet loss and the guaranteed data rate impact notably in most communications and protocols, they become critical to determine the sought coexistence.

2.1.7. Extensive Network Coverage in Vertical Premises (5GR-SKPI-7)

The extensive network coverage in vertical premises is directly impacted by six different Core KPIs; Latency (CKPI-1): Packet Loss (CKPI-2), Guaranteed Data Rate (CKPI-3), Availability (CKPI-5) and Connection Density (CKPI-7). Table 8 shows the measurement units of this Service KPI.

TABLE 8: EXTENSIVE NETWORK COVERAGE IN VERTICAL PREMISES

Service KPI	Measurable Unit
Extensive Network Coverage in Vertical Premises	Square Meters (m2)

On the one hand, the extensive network coverage is affected by the availability of connecting to a 5G RAN from the vertical premises and any location the actors in the use case would require connecting. On the other hand, the 5GR-SKPI-7 also is influenced by the common set of Core KPIs which directly impact the service performance, such as the latency, packet loss, and guaranteed data rate. Without fulfilling these Core KPIs, the system will not be capable of guaranteeing the minimum.

QoS expected by the vertical, hence limiting the coverage to areas where these Core KPIs can be satisfied. Furthermore, the connection density Core KPI can also impact on the number of users capable of being served in a specific location, impacting on the availability of actors and elements connecting to the network and consequently impacting on the extensive network coverage.

2.1.8. Service Operation Time (5GR-SKPI-8)

The service operation time for a vertical service can be derived by measuring three different Core KPIs: Packet Loss (CKPI-2), Guaranteed Data Rate (CKPI-3) and End-to-end Latency (CKPI-1). Table 9 shows the measurement units of this Service KPI.

TABLE 9: SERVICE OPERATION TIME UNIT

Service KPI	Measurable Unit
Service Operation Time	Minutes (min)

Certainly, Service Operation Time is influenced by the minimum guaranteed data rate: If this core KPI decreases, the overall service operation time is affected. Moreover, in case the packet loss increases, more retransmissions occur, which leads to reducing the data rate. Consequently, delaying the time to complete an iteration of the complete service workflow negatively impacts the service operation time. Additionally, the end-to-end latency could impact the service operation time: an increase in this core KPI can also increase the time it takes to perform a complete iteration of the service workflow.

2.1.9. Service Operation Capacity (5GR-SKPI-9)

Deriving the service operation capacity for a vertical service can be achieved by measuring three different Core KPIs: End-to-end Latency (CKPI-1), Guaranteed Data Rate (CKPI-3) and Availability (CKPI-5). Table 10 shows the measurement units of this Service KPI.

TABLE 10: SERVICE OPERATION CAPACITY UNIT

Service KPI	Measurable Unit
Service Operation Capacity	Absolute number

The service operation capacity can be impacted by the minimum guaranteed data rate, as the data rate defines the number of simultaneous service workflows that can be handled. Moreover, the availability also has an impact on the service operation capacity. On the one hand, with higher availability, fewer errors occur, and thus more workflows could be potentially added. On the other hand, with less availability, potentially more errors rise, and thus fewer workflows can be managed.

2.1.10. Service Availability (5GR-SKPI-10)

Inferring the service operation capacity for a vertical service can be achieved by measuring the Availability KPI (CKPI-5). Table 11 shows the measurement units of this Service KPI.

TABLE 11: SERVICE AVAILABILITY UNIT

Service KPI	Measurable Unit
Service Availability	% time the service is available

As such, in case there is a critical component failure, the whole service availability is impacted. The service availability KPI could be impacted the most by the minimum availability of a component inside the vertical service.

2.1.11. Service Reaction Time (5GR-SKPI-11)

Deriving the service reaction time for a vertical service can be achieved by measuring two different Core KPIs: End-to-end Latency (CKPI-1) and Guaranteed Data Rate (CKPI-3). Table 12 shows the measurement units of this Service KPI.

TABLE 12: SERVICE REACTION TIME

Service KPI	Measurable Unit
Service Reaction Time	Milliseconds (ms)

The end-to-end latency directly impacts the service reaction time, as with higher delay the reaction time increases and vice versa. Moreover, depending on the kind of triggered events, the data rate also impacts on the service reaction time. Particularly, if an event is triggered by an image or video stream, the data rate becomes critical. For example, when an action is required from a teleoperator based on the images available through a video stream, a higher data rate will allow them to process more images and react promptly to events.

2.1.12. KPIs to be considered by each pilot

The KPIs considered by each pilot are listed in the table below, summarizing the specific set of Service KPIs considered at each one.

TABLE 13: SERVICE KPIS CONSIDERED BY EACH PILOT IN THIS VALIDATION CAMPAIGN

	Industry 4.0 (INNO)	Industry 4.0 (COMAU)	Transportation (EFACEC_S)	Energy (EFACEC_E)
5GR-SKPI-1	X			
5GR-SKPI-2	X	X	X	X
5GR-SKPI-3	X	X		
5GR-SKPI-4	X	X	X	X
5GR-SKPI-5	X			
5GR-SKPI-6	X			
5GR-SKPI-7	X	X		
5GR-SKPI-8	X			
5GR-SKPI-9	X			
5GR-SKPI-10			X	X
5GR-SKPI-11				

2.2. Measurement procedures

Before going into detail on how each pilot measures and validates the targeted KPIs, some general considerations for the Service KPIs validation are outlined.

Prior to testing, according to the current technology capabilities, a theoretical canvas can be depicted considering the range of achievable values for the core KPIs. The following figure illustrates the idea when studying the impact on a service of two core KPIs.

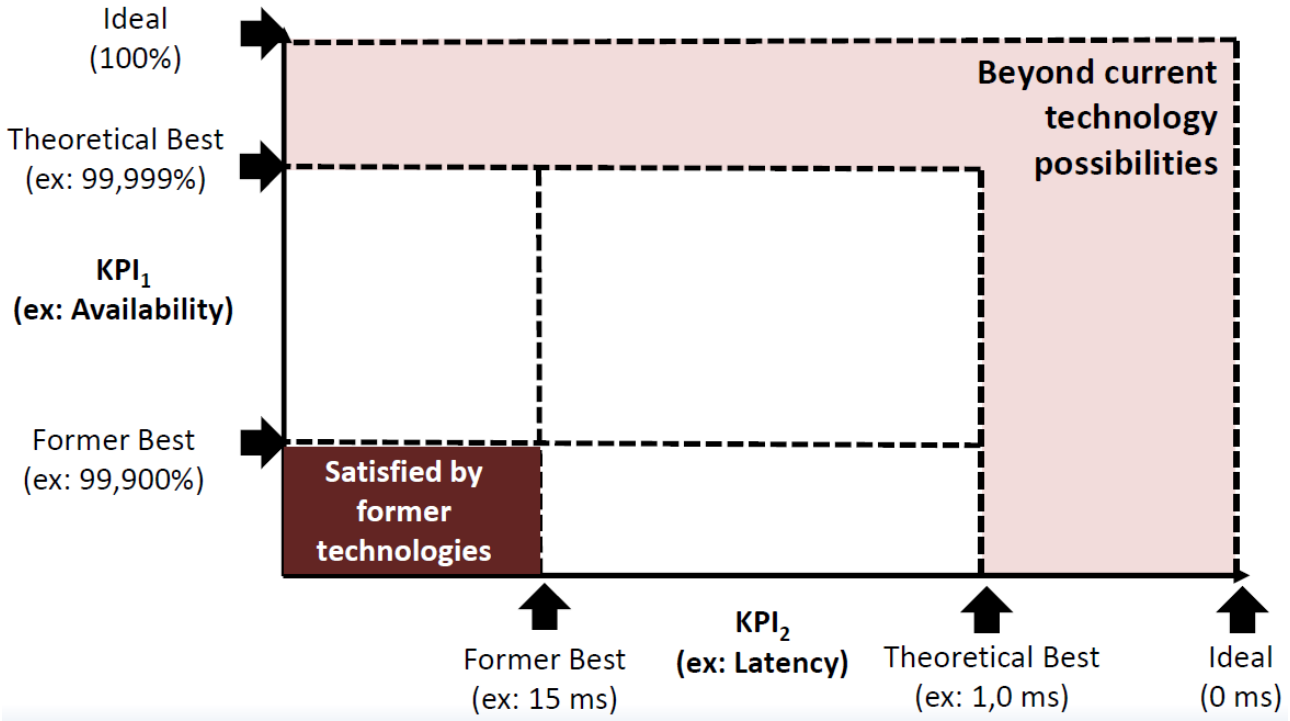


FIGURE 1: TECHNOLOGY CAPABILITY CANVAS

Then, executing tests, metrics are collected and Service KPIs are validated by a mix of objective measurements and subjective perceptions of an expert user. Besides, some Core KPIs can be influenced to emulate different network conditions. This way, it is possible to establish the performance conditions that allow determining service operation as: good (green area in illustrative figure), acceptable (yellow areas) or bad (orange area).

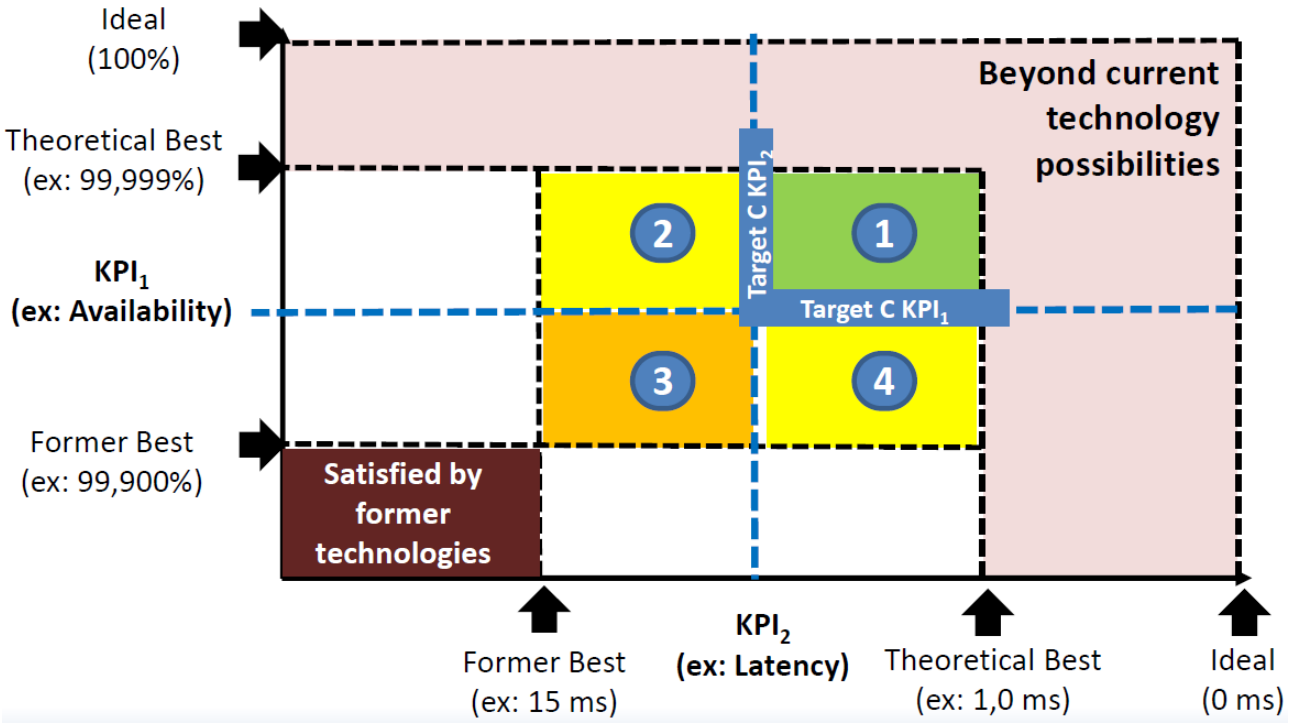


FIGURE 2: GOOD/ACCEPTABLE/BAD PERFORMANCE AREAS

5Growth Service KPIs (*5GR-SKPI-n*) are defined in D4.1 (in Table 2) as the qualitative indicators that allow validating a business and industrial scenario from the vertical point of view. Core 5G KPIs (*CKPI-m*) are also defined in D4.1 (in Table 1) as the technical KPIs that can be directly measured on the use cases' infrastructure and network. In most cases, 5Growth Service KPIs are not directly measurable, and this is the reason why D4.1 (in Table 4) defines a mapping between Core 5G KPIs and 5Growth Service KPIs. Section 5 of D4.1 includes the definition of the sets of specific Service KPIs selected to evaluate each of the use cases (UC) of the different 5Growth pilots. These UC-specific KPIs are named UC Service KPIs (indicated as PxUCy-SKPI-z where x is the pilot number, y is the use case number, z is the KPI number for the specific use case).

The four pilots in the project have a common approach to validate the UC Service KPIs. For each use case, it is reported a table which lists all the related UC Service KPIs. For each UC Service KPI, it is reported the associated 5Growth Service KPIs and the related Core 5G KPIs. The table also reports which Core 5G KPI can be either measured or emulated.

2.2.1. INNOVALIA Pilot

2.2.1.1. How to validate SKPIs identified for UC1

TABLE 14: INNOVALIA UC1 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P1UC1-SKPI-1: Teleworker-	CKPI-1 End-to-end Latency	Y	Emulated	Expressed as Bad/Acceptable/Good. This will be mapped with the measured Core KPIs to

CMM Synchronization (=5GR-SKPI-2)	CKPI-2 Packet Loss	N	Emulated	establish the range of values that imply a Bad, Acceptable or Good synchronization.
P1UC1-SKPI-2: High-resolution Real-time Video Quality (=5GR-SKPI-4)	CKPI-2 Packet Loss	Y	Emulated	Expressed in values from 1 to 5, being 1 the lowest and 5 the highest quality perception. This will be mapped with the measured Core KPIs to establish the range that correlates to each QoE value.
	CKPI-3 Guaranteed Data Rate	Y	Emulated	
	CKPI-9 Jitter	Y	Emulated	
P1UC1-SKPI-3: Service Setup Time (=5GR-SKPI-1)	CKPI-5 Availability	N	Measured	Expressed in minutes. It will be measured by using the timestamps of the activity logs. It will be the difference between the timestamp when the service instantiation was requested and the timestamp when the instantiation confirmation is received. As the NS and VNFs need to be instantiated, it can take some minutes.
	CKPI-6 Slice Creation Time	Y	Measured	
P1UC1-SKPI-4: Radius of Operation (=5GR-SKPI-5)	CKPI-1 End-to-end Latency	Y	Emulated	Expressed in kilometers. To obtain the distance, first, latency is measured, and artificial extra latency is added to find the maximum value of latency that allows a proper remote operation of the machine. That value is then translated into a distance parameter, assuming that every 100 km a packet has to traverse adds half a millisecond to the one- way latency and thus, 1 ms to the RTT.
	CKPI-5 Availability	N	Measured	
P1UC1-SKPI-5: Integrated Multitype Communications (=5GR-SKPI-6)	CKPI-2 Packet Loss	N	Emulated	Expressed as Success or Fail. Tests will be tried adding extra background traffic, and it will be verified whether the guaranteed data rate for each slice is met. In real-life network conditions, there will be other users using the network as well. The 5G network guarantees that the resources are distributed among the users according to their requirements.
	CKPI-3 Guaranteed Data Rate	Y	Emulated	
	CKPI-5 Availability	N	Measured	
P1UC1-SKPI-6: Extensive Network Coverage in the Factory Premises (=5GR-SKPI-7)	CKPI-1 End-to-end Latency	Y	Emulated	Expressed in m^2. The strategy for testing will be trying the performance at increasingly farther distances from the radio antenna to obtain the maximum distance that enables a good performance of the use case. The area can then be calculated as: π x distance^2 x (antenna beam width/360).
	CKPI-2 Packet Loss	N	Measured	
	CKPI-3 Guaranteed Data Rate	N	Measured	
	CKPI-5 Availability	N	Measured	
	CKPI-7 Connection Density	Y	Emulated	

5GR-SKPI-2 Synchronization between Communication Components

The Core KPIs to measure this SKPI are *CKPI-1 End-to-end Latency* and *CKPI-2 Packet Loss*.

This Service KPI requires human feedback. During the tests, workers will report their perception regarding the synchronization, qualifying the experience as Bad (not possible at all), Acceptable (possible to operate but somehow limited) or Good (operation with no limitations). This will be mapped with the measured Core KPIs to establish the range of values that imply a Bad, Acceptable or Good synchronization.

CKPI-1 End-to-end Latency is the main KPI. The lower the latency, the better the operation. Different latencies can be emulated by adding artificial latency.

CKPI-2 Packet Loss is a secondary KPI. An artificial packet loss rate can be emulated while testing, to obtain which would be the upper packet loss limit that would be acceptable for a right remote operation.

5GR-SKPI-4 High-resolution Real-time Video Quality

The Core KPIs to be measured are: *CKPI-2 Packet Loss*, *CKPI-3 Guaranteed Data Rate* and *CKPI-9 Jitter*.

As usual for QoE rankings, workers will report a value from 1 to 5 depending on their quality perception under different environment conditions, being 1 the lowest quality and 5 the highest quality. This will be mapped with the measured Core KPIs to establish the range of values that correlates to each QoE value.

CKPI-3 Guaranteed Data Rate is the main KPI. It can be emulated by demanding different Data Rate values from the video application.

CKPI-2 Packet Loss is a secondary KPI. It can be emulated by adding an artificial packet loss, to establish its impact on the video quality.

CKPI-9 Jitter is a secondary KPI. It can be emulated by specifying a random delay variation in the traffic control shaper.

5GR-SKPI-1 Service Setup Time

The Core KPIs to be measured are *CKPI-5 Availability* and *CKPI-6 Slice Creation Time*.

This Service KPI will be expressed in minutes. It will be measured by using the timestamps of the activity logs. It will be the difference between the timestamp when the service instantiation was requested and the timestamp when the instantiation confirmation is received. As the NS and VNFs need to be instantiated, it can take some minutes. It is expected from INNOVALIA this Service KPI to be lower than 5 minutes to be considered as acceptable.

CKPI-6 Slice Creation Time is the main KPI. It can only be measured, and the target would be that the 5Growth platform (along with 5G-EVE platform) is able to create it as quickly as possible.

CKPI-5 Availability is a secondary KPI. The impact of the availability is that if the network or components are not available, the service is not established. Thus, the availability means the probability of the service to be setup in the guaranteed time. If the availability of the resources is 99%, it can be guaranteed that for 99% of the time, the service setup will be completed in a certain time.

5GR-SKPI-5 Radius of Operation

The Core KPIs to be measured are *CKPI-1 End-to-end Latency* and *CKPI-5 Availability*.

This Service KPI will be expressed in kilometers. To obtain the distance, first, the latency will be measured, and artificial extra latency will be added to find the maximum value of latency that allows a proper remote operation of the machine. That value will be translated to distance, assuming that every 100 km a packet has to traverse adds half a millisecond to the one-way latency and thus 1 ms to the RTT [3].

CKPI-1 End-to-end Latency is the main KPI. The latency can be emulated by adding artificial latency.

CKPI-5 Availability is a secondary KPI. The impact of the availability is similar to the one discussed for 5GR-SKPI-3 above: if the availability of the resources is 99%, it can be guaranteed that for 99% of the time, it will be possible to operate the machine from a certain remote location.

5GR-SKPI-6 Integrated Multitype Communications

The Core KPIs to be measured are *CKPI-2 Packet Loss*, *CKPI-3 Guaranteed Data Rate* and *CKPI-5 Availability*.

It will be expressed as Success or Fail. Tests will be tried adding extra background traffic, and it will be verified if the guaranteed data rate for each slice is met. In real-life network conditions, there will be other users using the network as well. The 5G network guarantees that the resources are distributed among the users according to their requirements.

The main CKPI is *CKPI-3 Guaranteed Data Rate*. In any conditions, the network will try to provide the agreed user data rate for each slice. In this case, there are both an URLLC slice and an eMBB slice. The behavior of the data rate in each slice will be measured under different background traffic conditions. The data rate will just be measured.

CKPI-5 Availability is a secondary KPI. The impact of the availability is that if the network or components are not available, the communication will not be possible, so the availability means the probability of the communication to work. If the availability of the resources is 99%, it can be guaranteed that for 99% of the time, the communication over the network will work.

CKPI-2 Packet Loss is a secondary KPI. In this case, it is worth to measure it in the URLLC slice as it should be ultra-reliable, so even with bad background conditions, the network must be able to deliver those packets.

5GR-SKPI-7 Extensive Network Coverage in Vertical Premises

The Core KPIs to be measured are *CKPI-1 End-to-end Latency*, *CKPI-2 Packet Loss*, *CKPI-3 Guaranteed Data Rate*, *CKPI-5 Availability* and *CKPI-7 Connection Density*.

It will be expressed in m^2. The strategy for testing will be trying the performance at increasingly farther distances from the radio antenna to obtain the maximum distance that enables a good performance of the use case. The area can be then calculated as: π x $distance^2$ x (antenna beam width/360).

CKPI-1 End-to-end Latency and *CKPI-7 Connection Density* will be considered as the main CKPIs. Latency will be emulated by adding artificial latency, and tests will be performed while having more active users in the cell. Testing those different latency and simultaneous user conditions for each of the increasingly farther distances from the antenna, will provide conclusions on how coverage area may be affected.

2.2.1.2. How to validate SKPIs identified for UC2

TABLE 15: INNOVALIA UC2 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P1UC2-SKPI-1: Service Operation Time (=5GR-SKPI-8)	CKPI-1 End-to-end Latency	N	Measured	Expressed in minutes. The total time for the whole operation described will be calculated, but also the selected CKPIs will be measured to verify the right performance at each stage. 1.AGV->CMM: Sending of piece code (Packet Loss as main KPI, if the command is not received, next action will not be triggered) 2.CMM: Downloading inspection program (Data Rate as main KPI, the download speed will impact the most on time) 3.AGV->CMM: Piece in place for inspection (Packet Loss as main KPI, if the command is not received, next action will not be triggered) 4.CMM: Inspection execution (Data Rate as main KPI, as the results are sent in real-time to the workstation) 5.CMM->AGV: Piece can be taken back (Packet Loss as main KPI, if the command is not received, next action will not be triggered) 6-AGV: Piece ready for unloading (Packet Loss as main KPI, if the command is not received, next action will not be triggered)
	CKPI-2 Packet Loss	Y	Measured	
	CKPI-3 Guaranteed Data Rate	Y	Measured	
P1UC2-SKPI-2: Service Operation Capacity (=5GR-SKPI-9)	CKPI-3 Guaranteed Data Rate	Y	Measured	Expressed in number of workflows running simultaneously. To validate this KPI, it needs to be verified that each workflow meets the guaranteed data rate. Also, the availability plays a role as the percentage of time this capacity can be reached.
	CKPI-5 Availability	N	Measured	

P1UC2-SKPI-3: Service Creation Time (=5GR-SKPI-1)	CKPI-5 Availability	N	Measured	Expressed in minutes. It will be measured by using the timestamps of the activity logs. It will be the difference between the timestamp when the service instantiation was requested and the timestamp when the instantiation confirmation is received. As the NS and VNFs need to be instantiated, it can take some minutes.
	CKPI-6 Slice Creation / Adaptation Time	Y	Measured	
P1UC2-SKPI-4: AGV-Edge Control Synchronization (=5GR-SKPI-2)	CKPI-1 End-to-end Latency	Y	Emulated	Expressed as Bad/Acceptable/Good. This will be mapped with the measured Core KPIs to establish the range of values that imply a Bad, Acceptable or Good synchronization.
	CKPI-2 Packet Loss	N	Emulated	
P1UC2-SKPI-5: Network support for user mobility (=5GR-SKPI-3)	CKPI-2 Packet Loss	Y	Emulated	Expressed as Success/Fail. This will require a test where the end device is moving, so it can be evaluated with the traffic sent by the AGVs while they are moving. The traffic should be stable and not be altered too much while the AGVs move. The packet loss will be increased artificially to see how it impacts the ability of the AGVs to move smoothly and the ability of the 5G network to guarantee the data rate.
	CKPI-3 Guaranteed Data Rate	Y	Measured	
	CKPI-5 Availability	N	Measured	
P1UC2-SKPI-6: Concurrency of simultaneous users in a given area (=5GR-SKPI-7)	CKPI-1 End-to-end Latency	Y	Emulated	Expressed in m^2. The strategy for testing will be trying the performance of the AGV at increasingly farther distances from the radio antenna to obtain the maximum distance that enables a smooth performance. The area can then be calculated as: π x $distance^2$ x (antenna beam width/360).
	CKPI-2 Packet Loss	N	Measured	
	CKPI-3 Guaranteed Data Rate	N	Measured	
	CKPI-5 Availability	N	Measured	
	CKPI-7 Connection Density	Not sup-ported	Not supported	

5GR-SKPI-8 Service Operation Time

The Core KPIs to measure are *CKPI-1 End-to-end Latency, CKPI-2 Packet Loss* and *CKPI-3 Guaranteed Data Rate*.

This Service KPI will be expressed in minutes. The total time for the whole operation described will be calculated by measuring the elapsed time between the timestamp registered in the logs at the reception of a new piece code and the timestamp registered in the logs at the reception of the notification of piece ready for unloading. Also, the selected CKPIs will be measured to verify the right performance at each stage. The stages are:

1.AGV->CMM: Sending of piece code (Packet Loss as main KPI, if the command is not received, next action will not be triggered)

2.CMM: Downloading inspection program (Data Rate as main KPI, the download speed will impact the most on time)

3.AGV->CMM: Piece in place for inspection (Packet Loss as main KPI, if the command is not received, next action will not be triggered)

4.CMM: Inspection execution (Data Rate as main KPI, as the results are sent in real-time to the workstation)

5.CMM->AGV: Piece can be taken back (Packet Loss as main KPI, if the command is not received, next action will not be triggered)

6-AGV: Piece ready for unloading (Packet Loss as main KPI, if the command is not received, next action will not be triggered)

The main CKPI is *CKPI-3 Guaranteed Data Rate* at stages 2 and 4. A big amount of data is sent in these stages, so time will depend on the achieved data rate.

CKPI-2 Packet Loss is the main KPI at stages 1, 3, 5 and 6. If packet loss happens, the system will not be aware of the completion of a stage, so the whole operation would be delayed.

CKPI-1 End-to-end Latency is a secondary KPI. The latency will be in the order of tens of ms, so a slightly higher or lower latency will make little difference in the whole time of operation.

5GR-SKPI-9 Service Operation Capacity

The Core KPIs to measure are *CKPI-3 Guaranteed Data Rate* and *CKPI-5 Availability*.

This Service KPI will be expressed in minutes. The total time for the whole operation described will be calculated by measuring the time between the timestamp registered in the logs at the reception of a new piece code and the timestamp registered in the logs at the reception of the notification of a piece ready for unloading. Also, the selected CKPIs will be measured to verify the right performance at each stage.

The main KPI is *CKPI-3 Guaranteed Data Rate*. The number of workflows will be increased until the maximum value that allows the guaranteed data rate is found.

CKPI-5 Availability is a secondary KPI. The impact of the availability in this Service KPI is that if the network or components are not available, the operation will not be possible, so the availability means the probability of the operation to work. If the availability of the resources is 99%, it can be guaranteed that for 99% of the time, the operation over the network will work.

5GR-SKPI-1 Service Setup Time

The Core KPIs to be measured are *CKPI-5 Availability* and *CKPI-6 Slice Creation Time*.

This Service KPI will be expressed in minutes. It will be measured by using the timestamps of the activity logs. It will be the difference between the timestamp when the service instantiation was requested and the timestamp when the instantiation confirmation is received. As the NS and VNFs need to be instantiated, it can take some minutes. It is expected from INNOVALIA this Service KPI to be lower than 5 minutes to be considered acceptable.

CKPI-6 Slice Creation Time is the main KPI. It can only be measured, and the target would be that the 5Growth platform (along with 5G-EVE platform) is able to create it as quickly as possible.

CKPI-5 Availability is a secondary KPI. The impact of the availability is that if the network or components are not available, the service will not be established, so the availability means the probability of the service to be setup in the guaranteed time. If the availability of the resources is 99%, it can be guaranteed that for 99% of the time, the service setup will be completed in a certain time.

5GR-SKPI-2 Synchronization between Communication Components

The Core KPIs to measure are *CKPI-1 End-to-end Latency* and *CKPI-2 Packet Loss*.

This Service KPI requires human feedback. During the tests, workers will report their perception regarding the synchronization, qualifying the experience as Bad (not possible at all), Acceptable (possible to run but somehow limited) or Good (running with no limitations). This will be mapped with the measured Core KPIs to establish the range of values that imply a Bad, Acceptable or Good synchronization.

CKPI-1 End-to-end Latency is the main KPI. The lower the latency, the better the operation. Different latencies can be emulated by adding artificial latency.

CKPI-2 Packet Loss is a secondary KPI. An artificial packet loss rate can be emulated while testing, to obtain which would be the upper packet loss limit that would be acceptable for a right AGVs operation.

5GR-SKPI-3 Device Mobility

The Core KPIs to measure are *CKPI-2 Packet Loss*, *CKPI-3 Guaranteed Data Rate* and *CKPI-5 Availability*.

This Service KPI will be reported as Success or Fail. This will require a test where the end device is moving, so it can be evaluated with the traffic sent by the AGVs while they are moving. The traffic should be stable and not be altered too much while the AGVs move. The packet loss will be increased artificially to see how it impacts the ability of the AGVs to move smoothly and the ability of the 5G network to guarantee the data rate.

CKPI-2 Packet Loss is one of the main KPIs. If packet loss increases significantly, retransmissions will also increase and will have an impact in the smoothness of the movement. This KPI can be emulated by artificially setting a packet loss rate.

CKPI-3 Guaranteed Data Rate is also a main KPI. This KPI can only be measured, but it needs be verified that the guaranteed value is always met.

CKPI-5 Availability is a secondary KPI. The impact of the availability is that if the network or components are not available, the service will be impacted or even interrupted, so the availability means the probability of the network to provide the expected device mobility functionalities. If the availability of the resources is 99%, it can be guaranteed that for 99% of the time, the network will support device mobility.

5GR-SKPI-7 Extensive Network Coverage in Vertical Premises

The Core KPIs to be collected are *CKPI-1 End-to-end Latency*, *CKPI-2 Packet Loss*, *CKPI-3 Guaranteed Data Rate*, *CKPI-5 Availability* and *CKPI-7 Connection Density*.

It will be expressed in m^2. In this use case, the aim of the test is to evaluate the area where the AGVs would have a smooth operation. The strategy for testing will be trying the performance of the AGV at increasingly farther distances from the radio antenna to obtain the maximum distance that enables a smooth performance. The area can then be calculated as: π x distance^2 x (antenna beam width/360).

CKPI-1 End-to-end Latency will be considered as the main CKPI. Latency will be emulated by adding artificial latency. Testing those conditions of different latency for increasingly farther distances from the antenna, will provide conclusions on how coverage area may be affected.

2.2.1.3. How to measure each CKPI

5G EVE platform in 5TONIC makes 5Probe tools available to measure different network metrics. These tools are installed in key parts of the network. Then, a Data Shipper gets these metrics and forwards them to the 5G EVE Monitoring system (Kafka).

The 5Probe extracts KPIs from end-user traffic by analyzing it with programmable Shallow Packet Flow Inspection techniques.

TABLE 16: CKPI MEASUREMENT METHODOLOGY

CKPI	Tool	Measurement methodology
CKPI-1 End-to-end Latency	5Probe	Two time-synched probes are required to measure E2E latency, one at each end. The time it takes for the packet from leaving the source until it reaches the target, will be the latency time in ms.
CKPI-2 Packet Loss	5Probe	Percentage of packets that fail to reach their destination. To measure this KPI, only one probe is needed. When a TCP packet reaches its destination, it sends back an ACK response. The probe verifies for how many packets sent an ACK is received. When an ACK is not received for a certain packet, it is counted as a lost packet.
CKPI-3 Guaranteed Data Rate	5Probe	The probe measures the number of bytes of every packet that goes through the probe per second. The downlink and uplink traffic are measured independently. Only one probe is needed to measure this KPI.
CKPI-5 Availability	5Probe	Refers to the percentage of time that a system is fully operational. Because of the rates at which the various 5G-EVE sites are being tested and upgraded, it would be impractical (and deceptive) to measure the network availability, this being defined as the ratio of uptime over the total time (uptime plus downtime). Instead, we decided to estimate the service availability as one minus the measured packet error rate during the operation of a service.

CKPI-6 Slice Creation Time	Vertical Slicer	This CKPI is measured using the logs from the Vertical Slicer, obtaining the difference between the timestamp when the service creation was ordered and the timestamp when the confirmation is received that the service is available.
CKPI-7 Connection Density	Not supported	This CKPI has not yet been studied by 5GPPP TMV WG and it is not supported in 5G-EVE platform either.
CKPI-8: Data Volume	Indirect from CKPI-3	Using the data rate metrics obtained with the probe and dumped into the database, the volume of traffic in Gbits will be calculated from the sum of all downlink and uplink throughput from the start of the experiment until it ends.
CKPI-9 Jitter	Indirect from CKPI-1	Having the value of latency of each packet, the delta of latency between packets is calculated and the average jitter over a period is obtained.

2.2.1.4. Measurements on the Mobile Network Infrastructure

The measurements in the INNOVALIA pilot are mainly obtained with the aforementioned 5Probe. The probes are placed in the network so they can inspect the packets and obtain the desired metrics. A good place to have them is in the router that is installed in the same rack as the user plane component of the 5G EPC or the 5GCore (5GC). See in Figure 3 an illustration of the generic environment used in the validation campaigns of the INNOVALIA pilot.

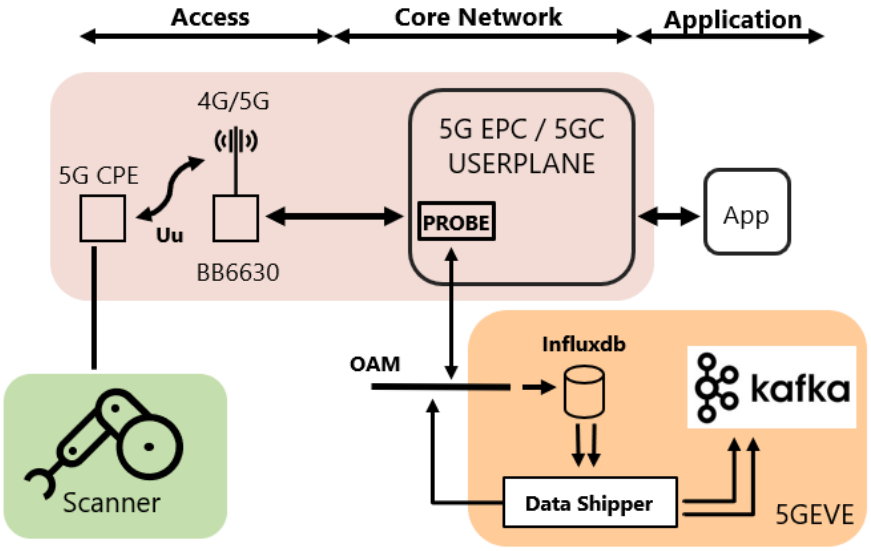


FIGURE 3: GENERIC MEASUREMENT ENVIRONMENT

2.2.2. COMAU Pilot

The following sections reports the test setups and methodologies used to assess the performances of the mobile and of the transport network.

2.2.2.1. Measurements on the Mobile Network Infrastructure

The measurements were performed in a testbed, whose setup is shown in Figure 4, that has seen the introduction of two PC Engines APU2C4 embedded boards (APU_108 and APU_109) at either ends of the mobile infrastructure (i.e., from the 5G CPE to the EPC), connected to the infrastructure devices by Gigabit Ethernet links.

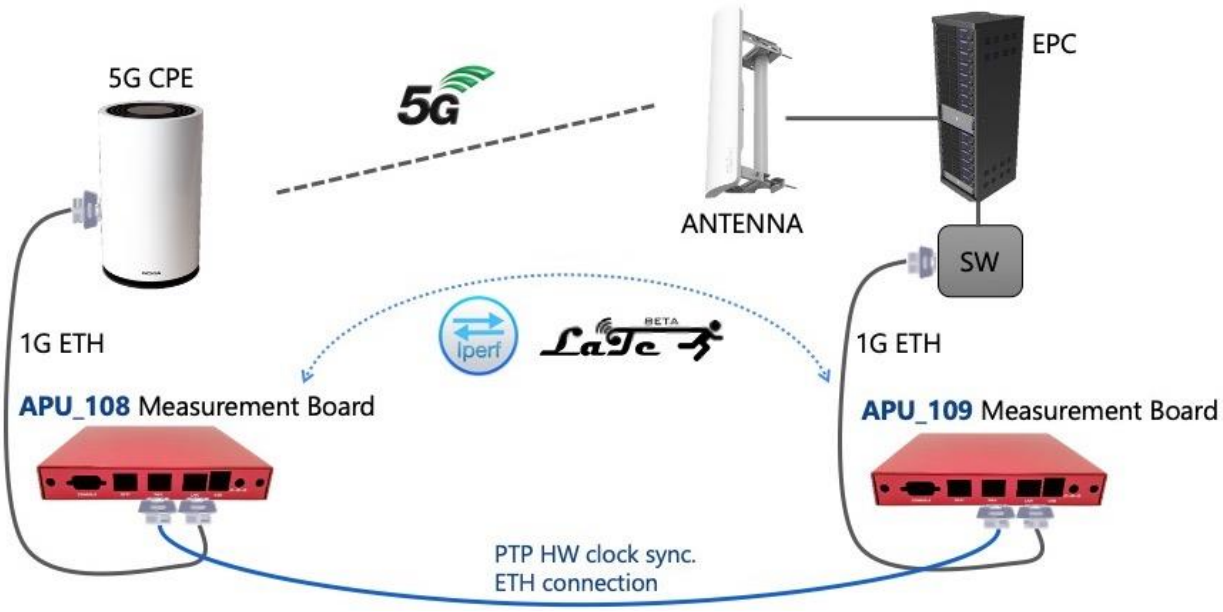


FIGURE 4: MEASUREMENT SETUP IN THE COMAU MOBILE INFRASTRUCTURE

The software tools used in the tests were: iPerf for the bandwidth measurements and a custom tool developed by POLITO, called LaTe (*La*tency *Te*ster) [4] for the latency and packet loss. The LaTe tool leverages a flexible application layer protocol, called LaMP, also developed by POLITO, which can be encapsulated inside any lower layer protocol, and it is designed to be as much self-contained as possible. For instance, it can be encapsulated, without requiring any modification to its rules and packet format, inside UDP over IPv4, or directly inside a raw Ethernet packet. The LaMP protocol works with a client-server paradigm: LaMP requests are sent by a client and received by a server, which replies back to the client. Then, in a typical scenario, LaMP computes the latency or Round-Trip Time (RTT) as the time difference between "send timestamps" and "receive timestamps", which are managed by the applications participating in the measurement session. Typically, the "send timestamp" is inserted inside the LaMP packet by the client sending a request and the "receive timestamp" is instead gathered by the client when a reply, containing a copy of the "send timestamp" from the corresponding request, is correctly parsed. Before any session is started, a connection initialization packet is sent by the client and properly acknowledged by the server, in order to ensure that the measurements can start, and the connection is stable enough. In order to compute one-way latency, LaMP also works in unidirectional mode, in which the client only sends requests to the server, which will be responsible for computing the latency and returning it to the client at the end of the test. This mode requires the clocks of the different devices to be precisely synchronized through the Precision Time Protocol (PTP) or through the Network Time Protocol (NTP).

In the COMAU pilot, a sub-microsecond clock synchronization between the involved hardware devices was guaranteed thanks to PTP (IEEE 1588), via a dedicated Gigabit Ethernet link between the two boards. The usage of PTP was enabled by the Ethernet cards embedded in the APU2C4 devices, namely Intel i210-AT, supporting hardware timestamping and clock synchronization.

LaTe can measure different kinds of latency: in our measurements we have leveraged a particular mode called "User-to-user", i.e., focused on measuring a full end-to-end application layer latency. It works by obtaining the reception timestamp as soon as a LaMP packet is received and correctly parsed by the LaTe application.

The tests reported in this deliverable lasted 1 week, with the APU2C4 measurement boards continuously alternating between iPerf and LaTe measurement. iPerf 2.0.13 was used to measure the throughput in UL and DL, both using UDP and TCP, while measurements were obtained for RTT latency, DL latency, UL latency and packet loss using LaTe. To avoid synchronization problems, probe packets were sent with a random exponential periodicity with minimum value 100 ms and mean 110 ms, with a new periodicity drawn every 10 packets.

2.2.2.2. Measurements on the Transport Network Infrastructure

The transport network is used in this pilot to connect the Baseband Units to the Radio Units of the radio cells, implementing centralized RAN architecture, where Baseband units serving several radio sites are centralized in a single location. Being an NSA (non-standalone) configuration, both a 5G and a 4G radio base stations are needed to connect the User Equipment. The transport network and how it is integrated in the COMAU pilot is described in deliverable 3.2 (D3.2) [5].

Figure 5 shows the structure of the User Plane of the Transport Network deployed for the COMAU pilot.

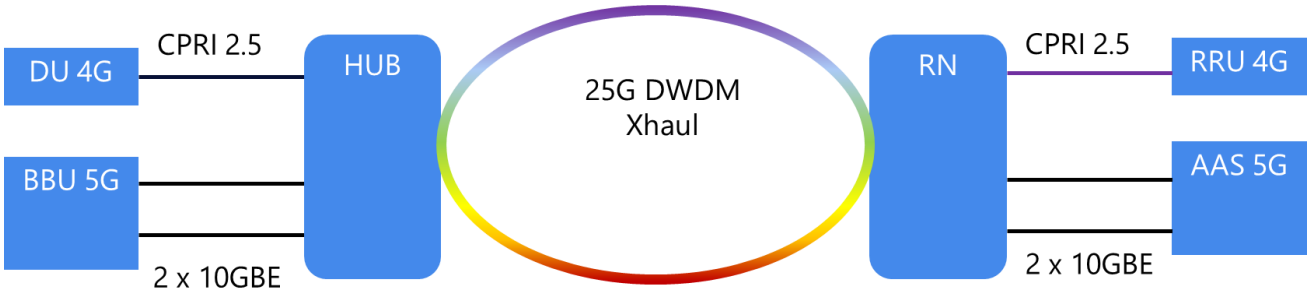


FIGURE 5: TRANSPORT NETWORK USER PLANE

On the left side there are the Baseband Units used for 4G (DU) and for 5G (BBU). Three clients, one CPRI Option 3 (2.5 Gb/s) link for 4G and two eCPRI links (10 GBE) for 5G are connected to a Hub node that performs switching and framing in the digital domain, and DWDM multiplexing functions. On the other end of the DWDM connection, realized with a fiber ring in order to offer physical protection, there is a Remote Node that performs wavelength selection with an OADM (Optical Add

and Drop Multiplexer), and de-framing and switching in the digital domain. The three clients are then connected to Remote Radio Unit (4G) and Advanced Antenna System (5G).

The set of measurements presented here are relative to the User Plane function of the transport network, to show that the performances of the network are adequate to support desired services without disruption. In that perspective, the reference methodology used is that described in IETF's RFC2544 [6].

The purpose of the measurements was to show that the performances of the transport network are adequate to support 5G and 4G fronthaul connections, based on two 10GBE eCPRI links (5G), and one CPRI link without introducing degradation or disruption.

The following tests, which results are reported in Section 5.2, have been performed on the 10GBE links used for eCPRI:

- *Latency*: measures the time taken by a test frame to travel through a network device or across the network and back to the test port. Latency is the time interval that begins when the last bit of the input frame reaches the input port and ends when the first bit of the output frame is seen on the output port.
- *Throughput*: measures the maximum rate at which none of the offered frames are dropped by the device/system under test (DUT/SUT). This measurement translates into the available bandwidth of the Ethernet virtual connection.
- *Back-to-back burst*: measures the longest burst of frames at maximum throughput or minimum legal separation between frames that the device or network under test will handle without any loss of frames.
- *Frame loss*: defines the percentage of frames that should have been forwarded by a network device under steady state (constant) loads that were not forwarded due to lack of resources.

For what concerns the CPRI link, a functional test has been performed: the test showed that regardless the traffic load of the 10GBE clients, the 4G base station is up and running and no alarms are showed by RBS management system.

The test setup used in the lab is illustrated in Figure 6.

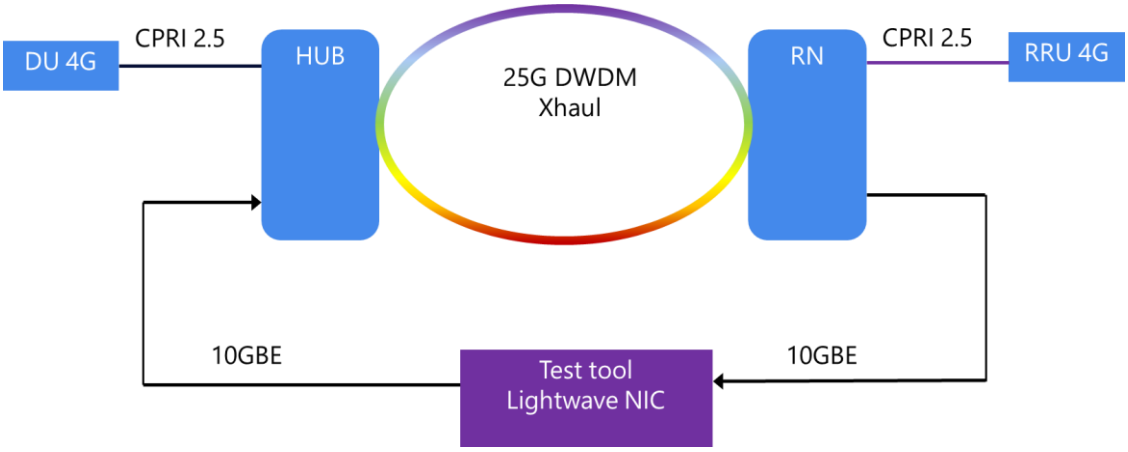


FIGURE 6: TEST SETUP

A test instrument (NIC from Lightwave) has been used to perform RFC2544 compliant measurement on a 10 GBE link, while the other 10 GBE and the CPRI 2.5G links were fully loaded.

The E2E latency experienced by the clients of the infrastructure, is constituted by the sum of the mobile contribution and the transport contribution. Results of the measurements, reported in detail in Section 5.2, demonstrate that the transport contribution is two order of magnitude less than the mobile one. Therefore, the following table related to the KPI validation refers only to the latency test of the mobile infrastructure.

2.2.2.3. How to validate SKPIs identified for UC1

TABLE 17: COMAU UC1 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P2UC1-SKPI-1 – Delay guarantees (=5GR-SKPI-2)	CKPI-1 End-to-end Latency	Y	Measured	Expressed in ms. It results from the computation of the latency or Round-Trip Time (RTT) as the time difference between "send timestamps" and "receive timestamps" used by the LaTe tool. Packet loss computed through the number of timeout events recorded by the LaTe tool.
	CKPI-2 Packet Loss	N	Measured	
P2UC1-SKPI-2 – Support of industrial protocols over 5G networks (=5GR-SKPI-2)	CKPI-1 End-to-end Latency	Y	Measured	Expressed as Bad/Acceptable/Good. It results from the computation of the latency or RTT as the time difference between "send timestamps" and "receive timestamps" used by the LaTe tool. Packet loss computed through the number of timeout events recorded by the LaTe tool.
	CKPI-2 Packet Loss	N	Measured	
P2UC1-SKPI-3 – Extensive on-plant coverage (=5GR-SKPI-7)	CKPI-1 End-to-end Latency	N	Measured	Expressed as Bad/Acceptable/Good. It results from the computation, in different areas of the plant, of the latency or RTT as the time difference between "send timestamps" and "receive timestamps" used by the LaTe tool.
	CKPI-2 Packet Loss	N	Measured	

	CKPI-3 Guaranteed Data Rate	N	Measured	Packet loss computed through the number of timeout events recorded by the LaTe tool. Availability is measured through a mapping of the coverage in the plant. Connection density is emulated through analytical models.
	CKPI-5 Availability	Y	Measured	
	CKPI-7 Connection Density	Y	Emulated	

- The first Service KPI considered for the COMAU pilot *UC1 Delays guaranteed* represents the time between the instant a problem occurs and the moment in which the plant manager decides how to mitigate it. The delay must be minimal, up to being not perceptible. The plant manager must be able to immediately see the results of their decisions. Measurements of this SKPI basically amount to the quantification of the End-to-end latency.
- The second Service KPI considered for the COMAU pilot *UC2 Support of industrial protocols over 5G* reflects the need to tunnel over a 5G link dedicated industrial communication protocols, such as Profinet or Ethernet/IP strongly based on time synchronization of data exchange cycles. They currently work over cable and are used to interconnect sensors, stations, and PLCs. Moving these protocols over a 5G link can be done if requirements about latency and packet loss are met.
- The third Service KPI considered for the COMAU pilot *UC3 Extensive on plant coverage* refers to the steady connection that must be ensured to all connected stations in the plant. To do so, the network must be broadly available throughout the plant and proper QoS must be ensured to all devices. A minimum link quality, with no packet loss, measured from peers connected to the 5G network, is required to guarantee the service. Since a multitude of sensors will be connected in a limited area, the network must offer a minimum connected device density and support integrated multitype communications.

2.2.2.4. How to validate SKPIs identified for UC2

TABLE 18: COMAU UC2 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P2UC2-SKPI-1 – Extensive coverage for high density sensors (=5GR-SKPI-7)	CKPI-1 End-to-end Latency	Y	Measured	Expressed as Bad/Acceptable/Good. It results from the computation of the latency or RTT as the time difference between "send timestamps" and "receive timestamps" used by the LaTe tool. Packet loss computed through the number of timeout events recorded by the LaTe tool. Data Rate in UL and DL measured by iPerf 2.0.13 for TCP and UDP traffic. Availability measured through a mapping of the coverage in the plant. Connection density emulated through analytical models.
	CKPI-2 Packet Loss	N	Measured	
	CKPI-3 Guaranteed Data Rate	N	Measured	
	CKPI-5 Availability	N	Measured	

	CKPI-7 Connection Density	Y	Emulated	

- The Service KPI considered for COMAU pilot *UC2 Extensive coverage for high density sensors* caters for the requirement that equipment monitoring for fault detection and predictive maintenance need networking support from all the sensors, which may simultaneously reporting data from each station of the production line. The core KPIs reflect the main metrics that should be considered in view of the coverage assessment, such as latency, packet loss and guaranteed data rate, along with availability and connection density.

2.2.2.5. How to validate SKPIs identified for UC3

TABLE 19: COMAU UC3 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P2UC3-SKPI-1 – High-resolution Real-time Video Quality (=5GR-SKPI-4)	CKPI-2 Packet Loss	Y	Measured	Expressed in values from 1 to 5, being 1 the lowest and 5 the highest quality perception. This will be mapped with the measured Core KPIs to establish the range that correlates to each QoE value. It results from the computation of: the packet loss through the number of timeout events recorded by the LaTe tool; the Data Rate in UL and DL as measured by iPerf 2.0.13 for TCP and UDP traffic; the jitter as quantified, in each batch of measurements, through the standard deviation of the latency, as well as through the absolute difference between max and min measured latency.
	CKPI-3 Guaranteed Data Rate	Y	Measured	
	CKPI-9 Jitter	Y	Measured	
P2UC3-SKPI-2 – Network Support for Device Mobility (=5GR-SKPI-3)	CKPI-5 Availability	Y	Measured	Expressed as Bad/Acceptable/Good. It results from the availability measured through a mapping of the coverage and of the received radio signal quality.
	CKPI-10 Received Radio Signal Quality	Y	Measured	
	CKPI-11 Buffer Occupancy	Y	Measured	

- The first Service KPI considered for COMAU pilot *UC3 High-resolution Real-time Video Quality*: its evaluation is justified by the fact that applications in support of remote assembly and maintenance operators run on consumer devices such as tablets or smartphones and are based on high-quality video feed. From the vertical's perspective, real-time HD video quality is critical, however, in order to assure a smooth factory operation in case of disruptions. A real-time HD video streaming requires high bandwidth and low packet loss. Moreover, in order for the vertical to enable a real-time satisfactory interaction, there must be a requirement for low latency and jitter. More specifically for this use case, in order for the

vertical to establish a reliable QoE-aware connection between the remote technical experts and the local technicians with no service interruptions, a low packet loss rate should be guaranteed, considering that otherwise it may impact the smoothness of the remote support.

- The second Service KPI considered for COMAU pilot *UC3 Network Support for Device Mobility* refers to the availability of remote support service all over the factory area in order to enable remote maintenance activities. Regardless of the operators' position and movements across the facility, their devices should stay connected to the network all over the plant. Service should be available all over the plant to allow remote maintenance activities close to every station of the facility.

2.2.3. EFACEC_S

For the Transportation pilot in Aveiro, we will address the measurement procedures considering two aspects: firstly, the measurement procedures concerning the operator's network perspective and, secondly, the measurement procedures considering the end user's perspective (i.e., the vertical).

Concerning EFACEC_S the validation campaigns involves the following stages:

a) Preliminary validation at EFACEC_S premises supported by LTE/4G technology – The main goal is to have a baseline information, verify the use cases at functional level and obtain a knowledge based in order to compare the achievements according the progress of the 5G network

b) Preliminary validation at IT Aveiro labs, also supported by LTE/4G technology

c) Validation over 5G network with emulated RAN at IT Aveiro labs

d) Validation over real 5G network at IT Aveiro labs

2.2.3.1. Network Operator Measurement Procedures

An end-to-end test between GbE/10GbE/100GbE ports (depending on the available Ethernet ports) is required to characterize the IP connection through a 5G terminal (DUT) connected to a mobile network backbone, with the test equipment connected to both sides simultaneously. The test shall include several traffic flows, defined according to the worst conditions that can load the 5G terminal. It is also important to make sure that the mobile network backbone is not fully loaded, in order to not have the DUT test results affected by the network behavior, reason why it will be preferable to use a dedicated test backbone, to ensure measurements accuracy and repeatability.
The following measurements and end-to-end KPIs have been considered to benchmark the IP networks for testing IP connections over a 5G backbone:

1. Packet Loss: Track Tx frames, Rx expected frames, Rx frames, Rx bytes frame delta loss %
2. Packet Rate: Tx frame rate, Rx frame rate, Rx rate (bps, Bps, Kbps, Mbps)
3. Packet Latency: Store and forward, cut-through, MEF frame delay, forwarding delay
4. Packet Delay Variation (Jitter): Delay variation measurement (jitter) minimum, average, maximum

5. Packet Inter-Arrival Time: Inter-arrival minimum, average, maximum
6. Sequence: Small error, big error, reverse error, last sequence number, duplicate frames, sequence gaps
7. Time Stamps: First and last timestamp per flow
8. TrueView™ Convergence: Control plane and data plane integrated time stamping for calculating convergence measurements
9. Packet Loss Duration: Estimated time without received packets calculated by frames delta at the expected Rx rate
10. Misdirected Packets: Per-port based count of packets not expected on a Rx port
11. Late Packets: Per-flow count of packets that arrived late; user-defined threshold for late packets
12. Re-Ordered Packets: Per-flow count of packets that were received out of order
13. Duplicate Packets: Per-flow count of duplicate packets that were received
14. In-Order Packets: Per-flow count of packets that were received in order

The test scenario to characterize the IP connection through a 5G terminal (DUT) connected to a mobile network backbone is represented in the figure below.

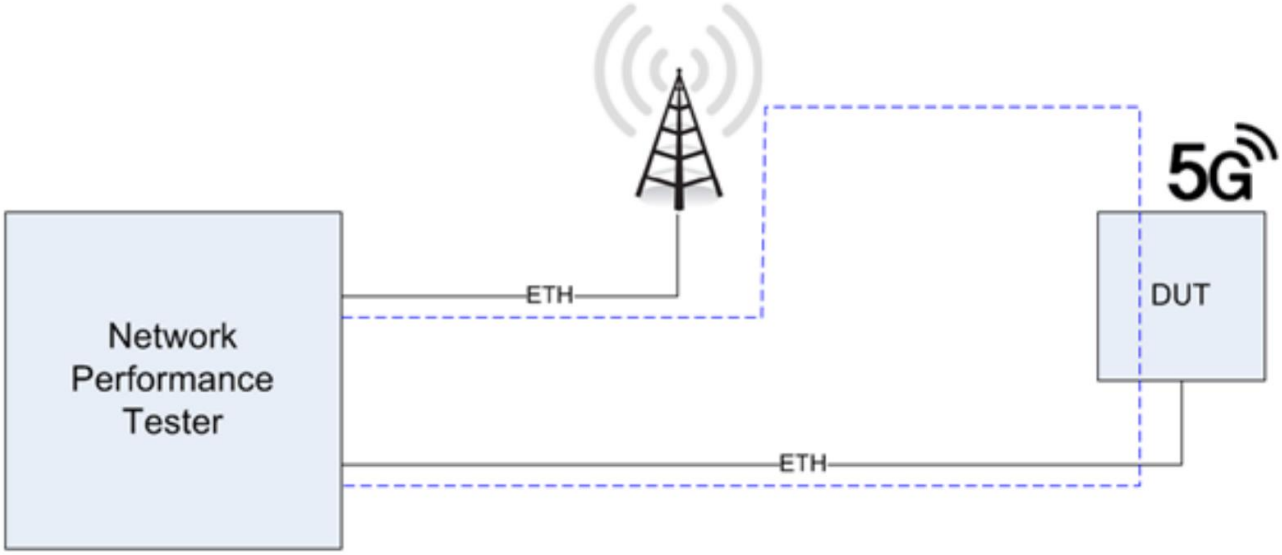


FIGURE 7: BASIC TEST SCENARIO

2.2.3.2. Vertical Measurement Procedures

This section describes the methodology for validating the Service KPIs in the EFACEC_S pilot. For each Service KPI, it has been analysed (i) how they will be measured; (ii) if human feedback is needed; (iii) which Core KPIs are the most relevant; (iv) which Core KPIs cab be emulated in the tests (for example, adding extra latency or forcing packet loss) or which ones can be only be measured.

2.2.3.3. How to validate SKPIs identified for UC1

TABLE 20: EFACEC_S UC1 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P3UC1-SKPI-1: Sync between LX detectors and Controller (=5GR-SKPI-2)	CKPI-1 End-to-end Latency	Y	Measured	Expressed as Bad/Acceptable/Good. This will be mapped with the measured Core KPIs to establish the range of values that imply a Bad, Acceptable or Good synchronization.
	CKPI-2 Packet Loss	N	Measured	
P3UC1-SKPI-2: Communication Availability between Lx Detectors and Lx Controller (=5GR-SKPI-10)	CKPI-5 Availability	Y	Measured	Ability of a product / equipment / system to be in state to perform a required function under given conditions and environment at a given instant of time or over a given interval assuming that the required external resource is provided; In fact, the goal is to measure the time the system will be unavailable. Availability = 1- Unavailability

The first Service KPI considered for EFACEC_S Pilot UC1 Communication between involves the communication of LX Detectors and LX Controller. The communication with these critical components must be established with low latency according to the railway standards [7], in order to assure the proper behavior of the Level Crossing; otherwise, The Level Crossing, including the half-barriers will enter into protection mode, blocking the road traffic. The network needs to ensure a **minimum** end-to-end latency in order to reduce the communication time between railway protection devices.

As such, this use case SKPI applies to the general **5Growth 5GR-SKPI-2** "Sync between communication components", which maps to the core CKPI-1 "e2e latency" and CKPI-2 "packet loss".

Regarding the "CKPI-1 – e2e latency" packets (data messages) between the rail sensor (strike in detector) and the railway crossing controller need to be exchanged with very low latency (< 10 ms) in order to detect that a train is approaching the Level Crossing and to start the safety signalling operation. If the data messages do not reach the destination with low latency a failure will occur and once again the system will operate in protection mode. The e2e latency can be measured at the level crossing controller.

For the "CKPI-2: packet loss", the communication between the strike in detector and the level crossing controller involves two different kinds of packets (data messages): i) events that occur when a new train is approaching the level crossing and ii) other protocolary messages such as keep-alive messages to guarantee a stable communication between the devices. Packet loss must be measured at the level of the crossing controller. In order to be compliant with Railway CENELEC standards and thus assuring a Certified Level crossing, this Core KPI must be guaranteed. If the availability or a failure that occurs does not recovery automatically in the proper time will represent a failure in the Level Crossing, starting this way to an operation in a protection mode (road traffic will be blocked). This situation has a huge impact in operation, can motivate penalties and requires the presence of maintenance people to manually recover from this operation mode.

Communication between these critical components must be established with availability according to the railway standards, in order to assure the proper behaviour of the Level Crossing; otherwise, The Level Crossing, including the half-barriers will enter into protection mode, blocking the road traffic. The network needs to ensure **high levels** of availability and resilience, in order to ensure that the communication between railway protection devices is always up and running.

As such, the second SKPI related to the Use Case 1, applies to the general 5Growth **5GR-SKPI-10** "Service availability", which maps to the core CKPI-5 "Availability". As indicated previously, in order to be compliant with Railway CENELEC standards and thus assuring a Certified Level crossing, this Core KPI must be guaranteed. If the availability or a failure that occurs doesn't recovery automatically in the proper time (<500ms) will represent a failure in the Level Crossing, starting this way to an operation in a protection mode (road traffic will be blocked). This situation has a huge impact in operation, can motivate penalties and requires the presence of maintenance people to manually recover from this operation mode. The availability of this critical communication can be measured at level crossing controller.

The communication availability between LX detectors and LX controller must be established with availability according to the railway standards, in order to assure the proper behavior of the Level Crossing; otherwise, the Level Crossing, including the half-barriers will enter into protection mode, blocking the road traffic. The network needs to ensure **high levels** of availability and resilience, in order to ensure that the communication between railway protection devices is always up and running.

As such, this use case SKPI applies to the general 5Growth **5GR-SKPI-10** "Service availability", which maps to the core CKPI-5 "Availability". As indicated in Section 5.3.1, in order to be compliant with Railway CENELEC standards and thus assuring a Certified Level crossing, this Core KPI must be guaranteed. If the availability or a failure that occurs doesn't recovery automatically in the proper time (<500ms) will represent a failure in the Level Crossing, starting this way to an operation in a protection mode (road traffic will be blocked). This situation has a huge impact in operation, can motivate penalties and requires the presence of maintenance people to manually recover from this operation mode. The availability of this critical communication can be measured at level crossing controller.

2.2.3.4. How to validate CKPIs identified for UC1

CKPI-1 RTT Latency is the main import KPI regarding safety communications. The lower the latency, the better the operation. The tool to be used is the Ping and with the data collected the average value of RTT latency and Jitter shall be calculated.

The value of RTT latency (e2e) and Jitter are going to be collected in two different situations:

1. With train detection application running
2. With train detection application stopped

Two measures will be performed just to compare that the values do not differ a lot because the Ping tool uses such a low amount of data to perform the test and the train detection application volume data exchange is quite small, as well.

Step1 -With Train detection application stopped measure communication latency and jitter for 10 minutes (PING tool shall present values <20ms but application can coop with up to <100ms).

Step 2 - With Train detection application running measure communication latency and jitter during all time of the test (PING tool shall present values <20ms but application can coop with up to <100ms).

To validate the e2e Latency will be also used the network tool described in Section 2.2.3 and the train application simulator. The simulator allows to adjust the desirable latency involved in the communication of LX components (detector and controller). With this parametrization any time the latency vales are not guaranteed the simulator will detect that safety messages are being discarded and communication will be lost.

CKPI-2 Packet Loss

The value of Packet Loss is going to be collected in two different situations:

1. With train detection application running
2. With train detection application stopped

Step -1: Verify that there are no lost packets during 10 minutes of the test.

Expected 0 packets lost from PING tool.

Step-2: Verify that there are no lost packets reported by the application (train detector application)

Expected 0 packets lost from PING tool.

Expected 0 packets lost (data messages) from Train Detector Application/simulator

To validate the Packet Loss will be also used the network tool described in Section 2.2.3.

CKPI-5- Availability

As availability, the standard EN50126 defines as the ability of a product / equipment / system to be in state to perform a required function under given conditions and environment at a given instant of time or over a given interval assuming that the required external resource is provided.

2.2.3.5. How to validate SKPIs identified for UC2

TABLE 21: EFACEC_S UC2 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P3UC2-SKPI-1: High-resolution Real-time Video Quality (=5GR-SKPI-4)	CKPI-2 Packet Loss	N	Measured	Expressed in values from 1 to 5. *Mean Opinion Score (MOS) is a well-known measure of video quality (5-Excellent, 4-Good, 3-Fair, 2-Poor, 1-Bad).*
	CKPI-3 Guaranteed Data Rate	Y	Measured	

	CKPI-9 Jitter	N	Measured	
P3UC1-SKPI-2: Real Time sensors monitoring latency (=5GR-SKPI-2)	CKPI-1 e2e latency	N	Measured	Expressed as Bad/Acceptable/Good. This will be mapped with the measured Core KPIs to establish the range of values that imply a Bad, Acceptable or Good synchronization.
	CKPI-2 packet Loss	Y	Measured	
P3UC3-SKPI-3: Real Time Sensors Monitoring Communication Availability (=5GR-SKPI-10)	CKPI-5 Availability	Y	Measured	Ability of a product / equipment / system to be in state to perform a required function under given conditions and environment at a given instant of time or over a given interval assuming that the required external resource is provided; In fact the goal is to measure the time the system will be unavailable. Availability = 1-Unavailability.

This use case is broken down into two different scenarios. involving (*i*) HD video image transmission from level crossing surveillance camera to the approaching trains' on-board tablets and/or maintenance agents' handheld tablets and (*ii*) Level crossing controller status and alarm events transmission to maintenance agents' tablets.

The main objective of the first scenario is to transmit, supported by 5G communications, HD video images to the approaching train, allowing the driver to get images of the level crossing area, thus preventing accidents caused by cars trapped between the level crossing gates. In the scope of this UC, it is also desired to transmit the same video images to maintenance agents/Command Centre, providing this way, mechanisms to reinforce the level crossing safety.

Regarding the second scenario, the main objective is to allow maintenance agents to monitor (status and alarm event transmission) and manage, remotely, the level crossing controller, through a tablet device (5G communications). Therefore, the first Service KPI, related to the Use case 2 is High-definition Video Surveillance.

The control centre and mobile devices (Train and maintenance agents) need to be able to visualize the HD surveillance video with good quality and low latency; that means MOS of at least 4[1] and in accordance with Video Surveillance Systems [8]. The network needs to ensure the appropriate resources to transmit HD video with a good level of quality and without delay. That means a sustained bandwidth, low end-to-end latency and jitter.

As such, this use case SKPI applies to the general 5Growth 5GR-SKPI-4 "High-Resolution Real-Time Video Quality", which maps to the core CKPI-2 "Packet Loss", CKPI-3 "Guaranteed Data Rate", CKPI-8 "Data Volume" and CKPI-9 "Jitter".

[1] *Mean Opinion Score (MOS) is a well-known measure of video quality (5-Excellent, 4-Good, 3-Fair, 2-Poor, 1-Bad).*

Regarding CKPI-2 "Packet Loss", the link needs low packet loss in order to avoid any loss of quality to the video or its freeze. This UC does not apply to safety critical communications, but it is used to reinforce the safety conditions of the level crossing. However, to have real-time video images to help train drivers or maintenance agent to make decisions, no packet loss in the transmission must be assured. This KPI can be measured at tablet devices (train).

Regarding CKPI-3 "Guaranteed Data Rate", the link needs to guarantee enough data rate to support the video stream. A guaranteed data rate of 160 Mbps is needed to accommodate high video resolution and M-JPEG compression assuring that 2 trains in the same level crossing can receive, simultaneously the LX images. This KPI can be measured at tablet devices (train).

Regarding CKPI-8 "Data Volume", this KPI is not considered due to its low relevance regarding real-time video. The link needs to have enough capacity to hold the transmission of the video at the highest quality possible.

Regarding CKPI-9 "Jitter", interarrival delays need to be avoided to not impact the quality of the video. According to the best practices for real-time video systems applicable to railway scenarios, the latency should be less than 100 ms to attain optimal performance. This latency as a direct relationship with the train speed and the 5G network performance. Latency between 100 and 200 ms causes impact in images visualization and latency higher than 500 ms are unacceptable and any image shall be removable from the displays. This jitter can be measured at tablet devices (train) and at the command center.

The second Service KPI of the use case is related to Real-time status and alarm information need to reach the control centre and maintenance team devices (Tablet) in useful time (less than 100ms). The network needs to ensure that the monitoring is performed with a **minimum** end-to-end latency so that the information can be collected with the appropriate accuracy.

As such, this use case SKPI applies to the general 5Growth **5GR-SKPI-2** "Sync between communication components", which maps to the core CKPI-1 "e2e latency" and CKPI-2 "packet loss".

Regarding the "CKPI-1 – e2e latency" packets (data messages) between the Lx Controller sensors and the maintenance teams need to be exchanged with low latency (< 100 ms) in order to allow the team to work without perceivable delay regarding the sensed information. If the data messages do not reach the destination with enough low latency, the quality of the maintenance team can be impacted. The latency can be measured at Tablet Devices**.**

For the "CKPI-2: packet loss", the communication between the sensors and the maintenance crew involves two different kinds of packets (data messages): i) events that occur when a new train is approaching the level crossing and ii) other protocolary messages such as keep-alive messages to guarantee a stable communication between the devices. The latency can be measured at Tablet Devices (maintenance teams).

The network needs to ensure that the monitoring services are continuously provided with a high level of reliability and availability in order to manage the status and alarms in a proper way, allowing

them to make decisions in near-real-time. As such, the third SKPI of the UC2 applies to the general 5Growth 5GR-SKPI-10 "Service availability", which maps to the core CKPI-5 "Availability".

Regarding the "CKPI-5 – Availability", the KPI can be measured at Tablet Devices (maintenance teams).

2.2.3.6. How to validate CKPIs identified for UC2

The video data are sent by RTSP over UDP, to validate the CKPI-2 and CKPI-3. with iperf3 it is needed to check the number of packets lost in UDP for several baud-rates, during 120s. the number of packets lost must not be more than 0.1%. To check the CKPI-9, with the ping tool, at a 0.2s transmission rate, in a test of at least 15min, for the packet size identical to the RTSP of the video, it is possible to extract the RTT and the associated Jitter.

The average RTT should not exceed 10ms and the Jitter less than 5ms. Additionally, with the video application running the latency measured should not exceed 10 ms. Note that the EFACEC_S video application can provide latency measures in real time.

SKPI2 – CKPI-1 - To validate the CKPI-1, with the ping tool, at a 0.2 s rate, in a test of at least 15min, for the packet size (UDP) 1KB similarly to the message protocol, we can extract the RTT, and by the symmetry the respective latency. The average RTT should not exceed 10 ms. The maximum RTT should not exceed 20ms.

SKPI2-CKPI-2- To validate the CKPI-2, with iperf3 we will check the number of packets lost in UDP for several baud-rates, during 120s, the number of packets lost must not be more than 0.1%.

2.2.4. EFACEC_E

For the Energy pilot in Aveiro, we will address the measurement procedures considering two aspects: firstly, the measurement procedures concerning the operator's network perspective and, secondly, the measurement procedures considering the end user's perspective (i.e., the vertical).

2.2.4.1. Network Operator Measurement Procedures

The network operator's measurement procedures are similar to those applicable to the Transportation vertical.

An end-to-end test between GbE/10GbE/100GbE ports (depending on the available Ethernet ports) is required to characterize the IP connection through a 5G terminal (DUT) connected to a mobile network backbone, with the test equipment connected to both sides simultaneously. The test shall include several traffic flows, defined according to the worst conditions that can load the 5G terminal. It is also important to make sure that the mobile network backbone is not fully loaded, in order to not have the DUT test results affected by the network behavior, reason why it will be preferable to use a dedicated test backbone, to ensure measurements accuracy and repeatability.
The following measurements and end-to-end KPIs have been considered to benchmark the IP networks for testing IP connections over a 5G backbone:

1. Packet Loss: Track Tx frames, Rx expected frames, Rx frames, Rx bytes frame delta loss %
2. Packet Rate: Tx frame rate, Rx frame rate, Rx rate (bps, Bps, Kbps, Mbps)
3. Packet Latency: Store and forward, cut-through, MEF frame delay, forwarding delay
4. Packet Delay Variation (Jitter): Delay variation measurement (jitter) minimum, average, maximum
5. Packet Inter-Arrival Time: Inter-arrival minimum, average, maximum
6. Sequence: Small error, big error, reverse error, last sequence number, duplicate frames, sequence gaps
7. Time Stamps: First and last timestamp per flow
8. TrueView™ Convergence: Control plane and data plane integrated time stamping for calculating convergence measurements
9. Packet Loss Duration: Estimated time without received packets calculated by frames delta at the expected Rx rate
10. Misdirected Packets: Per-port based count of packets not expected on a Rx port
11. Late Packets: Per-flow count of packets that arrived late; user-defined threshold for late packets
12. Re-Ordered Packets: Per-flow count of packets that were received out of order
13. Duplicate Packets: Per-flow count of duplicate packets that were received
14. In-Order Packets: Per-flow count of packets that were received in order

The test scenario to characterize the IP connection through a 5G terminal (DUT) connected to a mobile network backbone is represented in the figure below.

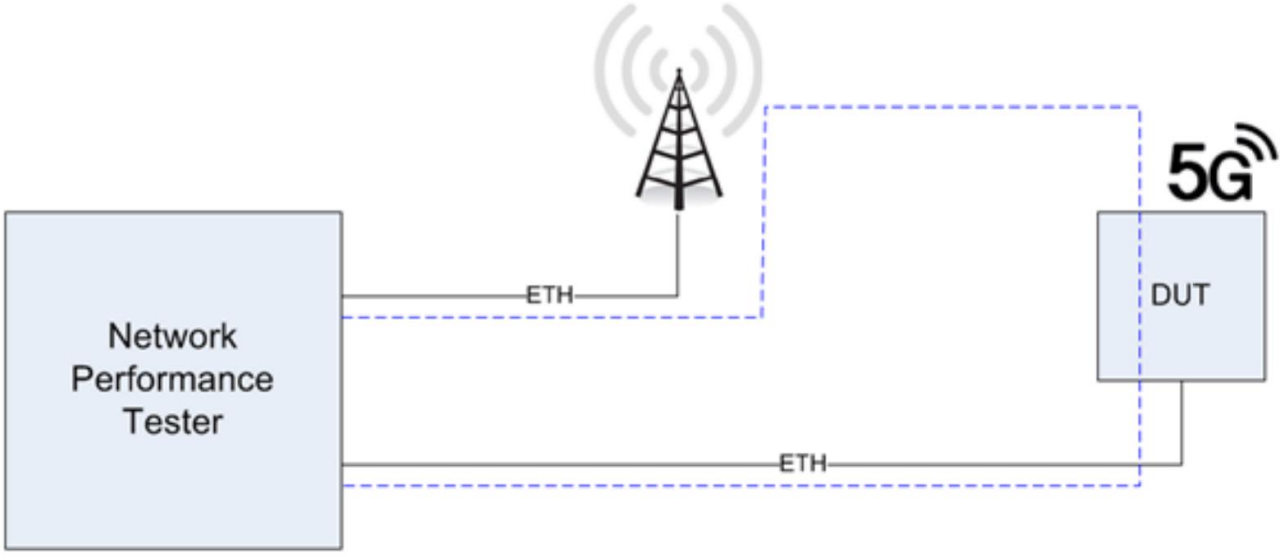


FIGURE 8: BASIC TEST SCENARIO

2.2.4.2. Vertical Measurement Procedures

This section describes the methodology for validating the Service KPIs in the EFACEC Energy pilot. For each Service KPI, it has been analysed: (i) how they will be measured;; (ii) which Core KPIs are the

most relevant; (iii) which Core KPIs cab be emulated in the tests (for example, adding extra latency or forcing packet loss) or which ones can be only be measured.

2.2.4.3. How to validate the SKPIs identified for UC1

TABLE 22: EFACEC_E UC1 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P4UC1-SKPI-1: Monitoring Sensors Information Collection and Visualization End-to-end Latency (5GR-SKPI-2)	CKPI-1: E2E latency	Y	E/M	End-to-end RTT can be evaluated injecting traffic using *ping* with different configurations. End-to-end latency can be measured directly over application traffic, enabling protocol trace on Frontend side. Packet Loss can be evaluated injecting traffic with Iperf and looking to the retransmission rate (TCP)
	CKPI-2: Packet Loss	N	E	
P4UC1-SKPI-2: Monitoring Sensors Information Collection and Visualization Availability (5GR-SKPI-10 & 5GR-SKPI-2)	CKPI-5: Availability	Y	E/M	Network availability can be evaluated injecting control traffic using *ping*, over a large period. Network availability can also be measured by processing the logs of the Frontend keep alive mechanism over the connection to the GSmart.
	CKPI-2: Packet Loss	N	E	
P4UC1-SKPI-3: High-definition Surveillance Video (5GR-SKPI-4)	CKPI-2: Packet Loss	Y	E	Packet Loss can be evaluated injecting traffic with iperf and looking to the packet loss rate (UDP). Jitter can be evaluated injecting traffic using *ping* with different configurations and processing the results.
	CKPI-9: Jitter	Y	E	
P4UC1-SKPI-4: Augmented Reality Information Real-time End-to-end Latency (5GR-SKPI-2)	CKPI-1: E2E latency	Y	E	End-to-end RTT can be evaluated injecting traffic using *ping* with different configurations. Packet Loss can be evaluated injecting traffic with Iperf and looking to the retransmission rate (TCP)
	CKPI-2: Packet Loss	N	E	
P4UC1-SKPI-5: Augmented Reality information Real-time Availability (5GR-SKPI-10 & 5GR-SKPI-2)	CKPI-2: Packet Loss	Y	E	Network availability can be evaluated injecting control traffic using *ping*, over a large period.
	CKPI-5: Availability	N	E	

- The first Service KPI considered for EFACEC Energy pilot UC1 is *Monitoring Sensors Information Collection and Visualization End-to-end Latency*. The Core KPIs to measure are *CKPI-1 End-to-end Latency* and *CKPI-2 Packet Loss*.

  CKPI-1 End-to-end Latency. The lower the latency, the better the real-time operation. In order to be acceptable for real-time operation an upper limit for latency is established at 100 ms.

  CKPI-2 Packet Loss. The SCADA protocol between the Secondary Substation and the Control Centre is TCP based so, packet loss will lead to retransmissions. In order to guarantee a stable

synchronization and to avoid degradation of data transfer performance between the different measurement devices and the control centre, the packet loss rate shall be kept at minimum values.

- The second Service KPI considered for EFACEC Energy pilot UC1 is *Monitoring Sensors Information Collection and Visualization Availability*. The Core KPIs to measure are *CKPI-5: Availability* and *CKPI-2 Packet Loss*.

  CKPI-5 Availability. The impact of the availability is that if the network or some components are not available, the communication will not be possible, so the availability means the probability of the communication to work. The network service needs to be available for continuously sending the monitoring data and video surveillance. If the availability of the network is 99,9%, it can be guaranteed that for 99,9% of the time, the communication over the network will work.

  CKPI-2 Packet Loss. The SCADA protocol between the Secondary Substation and the Control Centre is TCP based so, packet loss will lead to retransmissions. In order to guarantee a stable synchronization and to avoid degradation of data transfer performance between the different measurement devices and the control centre, the packet loss rate shall be kept at minimum values.

- The third Service KPI considered for EFACEC Energy pilot UC1 is *High-definition Surveillance Video*. The Core KPIs to measure are *CKPI-2 Packet Loss* and *CKPI-9: Jitter*.

  CKPI-2 Packet Loss. The video streaming will consist mainly in RTSP and UDP transmissions. Packet loss in UDP will lead to lost video frames. When considering encoders like H.264, the loss of I or P frames can have a strong impact in the video reception. So, in order to guarantee a good quality in the reception and visualization of the video streaming, the packet loss rate shall be kept at minimum values.

  CKPI-9 Jitter. Jitter can have impact in the time synchronization of the reception, affecting the visualization of the video streaming in the Control Centre. To a certain point, however, some video streaming protocols, can effectively mitigate the effects by smoothing variations in timing of packets, and feeding them to the end application at a more regular rate.

- The fourth Service KPI considered for EFACEC Energy pilot UC1 is *Augmented Reality Information Real-time End-to-end Latency*. The Core KPIs to measure are *CKPI-1 End-to-end Latency* and *CKPI-2 Packet Loss*.

  CKPI-1 End-to-end Latency. If the information does not come on time, the AR service will lose integrity, representing wrong or not timely information on top of live video streaming. In order to enable a correct representation of data over the video. an upper limit for latency is established at 100 ms.

  CKPI-2 Packet Loss. In order to guarantee a stable synchronization and to avoid degradation of data transfer performance, the packet loss rate shall be kept at minimum values.

- The fifth Service KPI considered for EFACEC Energy pilot UC1 is *Augmented Reality information Real-time Availability*. The Core KPIs to measure are *CKPI-5: Availability* and *CKPI-2 Packet Loss*.

  CKPI-5 Availability. The impact of the availability is that if the network or components are not available, the communication will not be possible, so the availability means the probability of the communication to work. The network service needs to be available for continuously sending the monitoring data and video surveillance. If the availability of the network is 99,9%, it can be guaranteed that for 99,9% of the time, the communication over the network will work.

  CKPI-2 Packet Loss. In order to guarantee a stable synchronization and to avoid degradation of data transfer performance between the different measurement devices, the control centre and the AR application in the mobile device, the packet loss rate shall be kept at minimum values.

2.2.4.4. How to validate the SKPIs identified for UC2

TABLE 23: EFACEC UC2 SPECIFIC SERVICE KPIS, CORE KPIS AND VALIDATION METHODOLOGY

SKPI	CKPI	Main?	Emul/Meas	Validation methodology
P4UC2-SKPI1: Last-gasp Information End-to-end latency (=5GR-SKPI-2)	CKPI-1 – E2E latency	Y	E/M	End-to-end RTT can be evaluated injecting traffic using *ping* with different configurations. End-to-end latency can be measured directly over application traffic, comparing the time of the power-off event with the time of Last Gasp arrival to GSmart. Packet Loss can be evaluated injecting traffic with iperf (UDP). Packet Loss can be measured comparing the power-off events triggered at the LV Sensor with the monitored power-off events at the operator workstation
	CKPI-2 Packet Loss	Y	E/M	
P4UC2-SKPI2: Last-gasp Information Connectivity Availability (=5GR-SKPI-10 & 5GR-SKPI-2)	CKPI-5: Availability	Y	E	Network availability can be evaluated injecting control traffic using *ping*, over a large period.
	CKPI-2: Packet Loss	Y	E	
P4UC2-SKPI3: Secondary Substation End-to-end Latency (=5GR-SKPI-2)	CKPI-1 – E2E latency	Y	E	End-to-end RTT can be evaluated injecting traffic using *ping* with different configurations. Packet Loss can be evaluated injecting traffic with iperf
	CKPI-2 Packet Loss	Y	E	
P4UC2-SKPI4: Secondary Substation Availability (=5GR-SKPI-10 & 5GR-SKPI-2)	CKPI-5: Availability	Y	E	Network availability can be evaluated injecting control traffic using *ping*, over a large period.
	CKPI-2: Packet Loss	Y	E	

P4UC2-SKPI5: Time Synchronization between Low Voltage Sensors (=5GR-SKPI-2)	CKPI-1 – E2E latency	Y	E	End-to-end RTT can be evaluated injecting traffic using *ping* with different configurations. Packet Loss can be evaluated injecting traffic with iperf
	CKPI-2 Packet Loss	Y	E	

- The first Service KPI considered for EFACEC Energy pilot is UC2 is *Last-gasp Information End-to-end latency*. The Core KPIs to measure are *CKPI-1 End-to-end Latency* and *CKPI-2 Packet Loss*.

  CKPI-1 End-to-end Latency. The network needs to send the last-gasp capable devices information as soon as it is generated with very low latency, to ensure that the device still has power. In this scenario an upper limit for latency is established at 5 ms.

  CKPI-2 Packet Loss. The last gasp information needs to be sent with no packet loss**.** Since transmission is done over UDP, Packet loss will result in unrecoverable transmission of last gasp event.

- The second Service KPI considered for EFACEC Energy pilot UC2 is *Last-gasp Information Connectivity Availability*. The Core KPIs to measure are *CKPI-5: Availability* and *CKPI-2 Packet Loss*.

  CKPI-5 Availability. The impact of the availability is that if the network or some components are not available, the communication will not be possible, so the availability means the probability of the communication to work. The network service needs to be available for continuously sending last-gasp events. If the availability of the network is 99,9%, it can be guaranteed that for 99,9% of the time, the communication over the network will work.

  CKPI-2 Packet Loss. The communication between the Low Voltage sensor and the Gsmart is UDP based so, packet loss will lead to non-recoverable transmission of the last-gasp event. To guarantee a successful transmission of the last-gasp event, the packet loss rate shall be kept near to zero.

- The third Service KPI considered for EFACEC Energy pilot UC2 is *Secondary Substation End-to-end Latency.* The Core KPIs to measure are *CKPI-1 End-to-end Latency* and *CKPI-2 Packet Loss*.

  CKPI-1 End-to-end Latency. As the reconstruction of the power outage event requires very precise analysis, the operator needs to be able to collect information quickly. The information needs to be sent with very low delay from its origin (Low Voltage sensors) to the Control Centre. In order to be acceptable for real-time analysis an upper limit for latency is established at 100 ms.

  CKPI-2 Packet Loss. The communication between the Low Voltage sensors and the Gsmart is UDP based so, packet loss will lead to non-recoverable transmission of the telemetry data from the Low Voltage sensors along the LV power network. In order to guarantee a successful transmission of the LV telemetry, the packet loss rate shall be kept at minimum values.

- The fourth Service KPI considered for EFACEC Energy pilot UC2 is Secondary Substation Availability. The Core KPIs to measure are *CKPI-5: Availability* and *CKPI-2 Packet Loss*.

  CKPI-5 Availability. The impact of the availability is that if the network or some components are not available, the communication will not be possible, so the availability means the probability of the communication to work. The network service needs to be available for continuously sending last-gasp events. If the availability of the network is 99,9%, it can be guaranteed that for 99,9% of the time, the communication over the network will work.

  CKPI-2 Packet Loss. The communication between the Low Voltage sensors and the Gsmart is is UDP based so, packet loss will lead to non-recoverable transmission of the telemetry data from the Low Voltage sensors along the LV power network. In order to guarantee a successful transmission of the LV telemetry, the packet loss rate shall be kept at minimum values.

- The fifth Service KPI considered for EFACEC Energy pilot is *Time Synchronization between Low Voltage Sensors*. The Core KPIs to measure are CKPI-5: Availability and CKPI-2 Packet Loss.

  CKPI-1 End-to-end Latency. The time synchronization commands sent by the GSmart to the low voltage sensors will need to guarantee the synchronization between all sensors equal to or less than 1ms. In order to keep the devices all timely synchronized, the connectivity among them needs to be of very low latency (<20ms) and jitter should be as lower as possible in order to enable a predictive latency, which is an important aspect for time synchronization protocols.

  CKPI-2 Packet Loss. In order to guarantee a stable time synchronization, the packet loss rate shall be kept at minimum values.

2.3. Available tools

2.3.1. End-to-end Unidirectional Link Latency Evaluator

2.3.1.1. General description

These probes have been designed and developed within the project 5Growth by SSSA and have been integrated with the 5Growth monitoring platform by Mirantis.

The End-to-end Unidirectional Link Latency Evaluator is an example of active probe that allows the estimation of a unidirectional link spanning across the network. The tool requires the execution of two probes (i.e., at the source and at the destination of the end-to-end link to be evaluated).

The probe behaviour is based on two main processes:

1. The periodic unidirectional link latency evaluation (the real link latency estimation)
2. The clock offset estimation (i.e., the clock difference) to be compensated for the link latency evaluation.

The general scenario of application is shown in Figure 9, where a simple distributed system, composed by 2 VNFs interconnected by a unidirectional link, is represented. The two cloud entities, instantiated in different premises, present different time clocks (i.e., t0' refers to the time clock of VNF1, while t1 refers to the time clock of VNF2). The management plane is bidirectional and allows the IP reachability among the two VNFs.

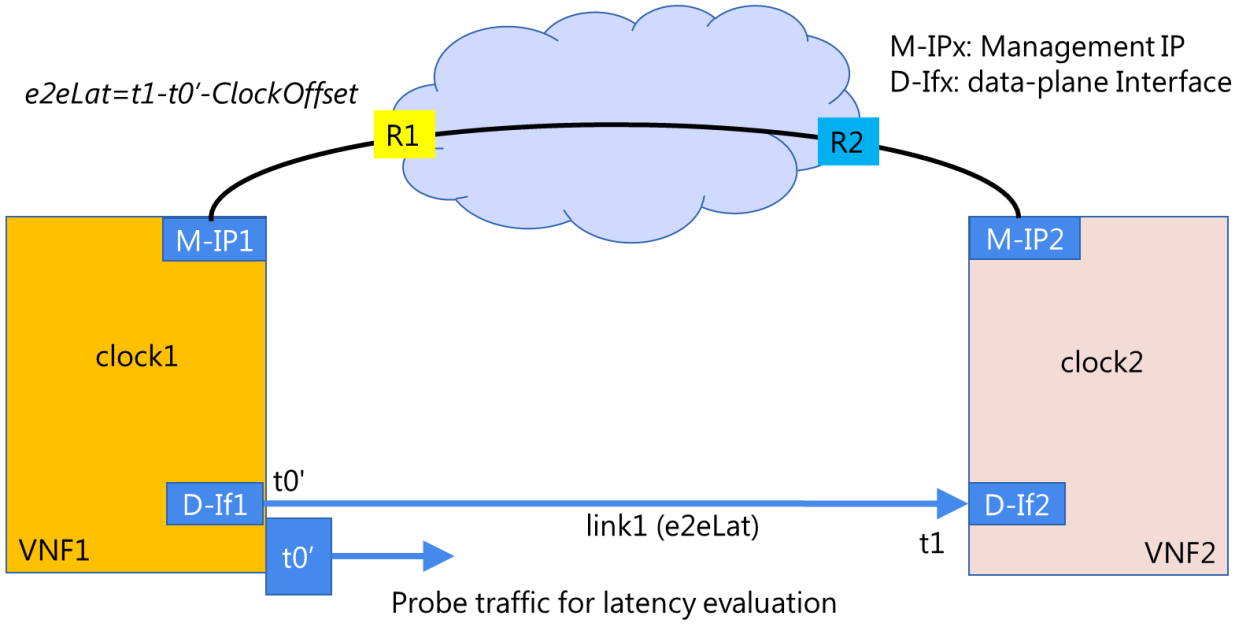


FIGURE 9: END-TO-END UNIDIRECTIONAL LINK LATENCY EVALUATION GENERAL SCENARIO

In this case the unidirectional latency would be:

$$e2eLat = t1 - t0' - ClockOffset \qquad (1)$$

where ClockOffset represents the difference among the time clock of the 2 VNFs.

In order to evaluate the ClockOffset, a dedicated mechanism has been designed (reported in Figure 10). More specifically, ad-hoc packets are periodically sent by the destination VNF reporting the creation time, the receiver (i.e., VNF1) adds the receiving time (referred to its own time clock) and sends back it to VNF1. Then, the ClockOffset can be evaluated by applying (2):

$$ClockOffset = t3' - t2 - \frac{RTT}{2} \qquad (2)$$

where RTT=t4-t2.

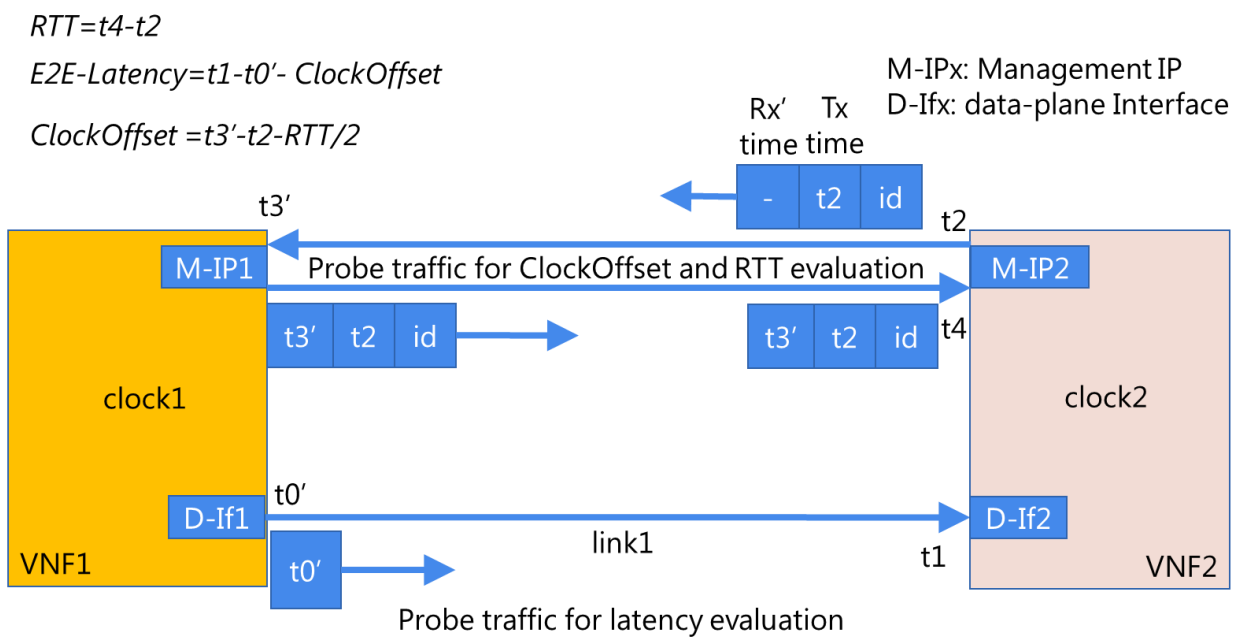


FIGURE 10: CLOCK OFFSET AND RTT EVALUATION

By continuously repeating the three estimations and applying the reported formulas, the probes evaluate the unidirectional end-to-end link latency. In case the data-plane link is bidirectional, the ClockOffset estimation can be carried out on the data-plane, without exploiting the double connectivity.

Current version of the probe does not work in case the management plane is asymmetric (i.e., link latency is not equal to the RTT/2).

The probes have been extended, including specific counters, enabling the evaluation of the packet loss and jitter experienced along the end-to-end link.

Moreover, the original probe implementation has been enhanced, including a Prometheus exporter, in order to enable the communication with the 5Growth monitoring platform, based on Prometheus Server.

2.3.1.2. Configuration

The details of the designed procedure of the integrated system are presented below.

The probes, including the Prometheus exporter, are installed by the 5Growth monitoring platform. Each probe exposes the port 15000 for estimation requests. A second socket is activated on port 14998 (used by the destination probe) in order to process the requests coming from the remote probe.

1) The Prometheus Server requests monitoring data from the probe installed on the VM2, sending a REST GET method request at the port 15000:

   http:// VM2IP:15000/metrics?ip=VM1IP&polling=2

2 parameters are included in the request:

- **ip** is the remote IP address (VM1IP) involved in the evaluation
- **polling** is latency evaluation period represented in seconds

2) The probe at VM2 activates the SRCoffset service on port 14999 for receiving offset information from the probe of VM1

3) The probe at VM2 starts the SRCprobe thread which sends probe traffic from VM2 to the port 14998 on VM1

4) The probe at VM2 returns "no data" with HTTP code 404, as reply to the request issued at step1.

5) The probe at VM1 detects the source IP address of probe traffic (VM2IP) and starts the thread DSToffset on VM1, replying with the proper information to VM2.

6) Then, the probe at VM2 evaluates and collects the link metrics, including the latency, the jitter, the packet loss, the percentage of packet loss and the total transmitted packets.

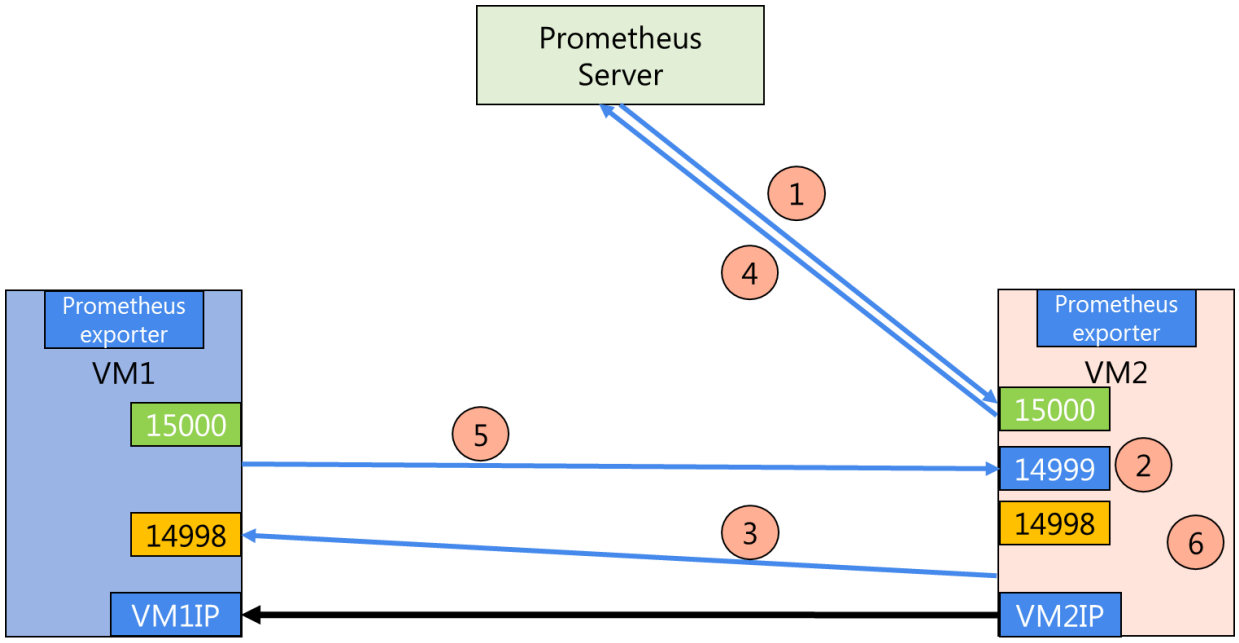


FIGURE 11: WORKFLOW FOR THE INTEGRATED E2E UNIDIRECTIONAL LINK LATENCY EVALUATION PROBES

Once the Prometheus server requests again the link information from the probe at VM2 (sending the GET request http://VM2IP:15000/metrics?ip=VM1IP&polling=2), the probe processes all the metrics collected since the last Server Prometheus request.

An example of the reply is shown in Figure 12, where the average values of link latency, jitter, percentage of packet loss, packet loss and total transmitted packets are shown.

```
# HELP linklatency avg_linklatency
# TYPE linklatency gauge
linklatency{host=" VM1IP "} 0.1
# HELP jitter avg_jitter
# TYPE jitter gauge
jitter{host=" VM1IP "} 0.1
# HELP packetLoss_percent packetLoss_percent
# TYPE packetLoss_percent gauge
packetLoss_percent{host="VM1IP"} 0.0
# HELP packetLoss packetLoss
# TYPE packetLoss gauge
packetLoss{host=" VM1IP "} 0.0
# HELP totalTXPackets totalTXPackets
# TYPE totalTXPackets gauge
totalTXPackets{host=" VM1IP "} 17.0
```

FIGURE 12: EXAMPLE OF THE PROMETHEUS EXPORTER REPLY

If the probe at VM2 does not receive any request during last 300 seconds, it stops the link latency evaluation processes.

2.3.2. Prometheus node exporter

2.3.2.1. General description

The Prometheus node exporter is a software that is installed on Linux machines and provides the Prometheus server by hardware and OS metrics. It provides metrics to the Prometheus server from such components as CPU, disk I/O (filesystem statistics, disk space usage), network I/O (interface statistics such as bytes transferred), etc. This exporter is used in the 5Growth project as a source of metric from the virtual machines (vertical workloads) and physical hosts (5Growth platform nodes).

Prometheus node exporter does not require an additional configuration once installed. The metrics got retrieved by the server by sending the request to the exporter:

REST GET requests with URL http://ip_address:9100/metrics should be sent to the host for getting metrics. The list of all available metrics will be received as a response to this request.

2.3.3. Prometheus Blackbox exporter

2.3.3.1. General description

The Prometheus Blackbox exporter is a software that is installed on Linux machines and enables the possibility to probing endpoints over HTTP, HTTPS, DNS, TCP, and ICMP. This exporter is used in the 5Growth project for measuring core KPI - "Availability". The Prometheus Blackbox exporter allows the possibility to monitor service status and notifies the Prometheus server this information. It also gives information about the probe duration, status, success, etc.

The URL for the request to Blackbox exporter has the format
http://ip_address:9115/probe?target=ip_address_target_host&module=http_2xx

Where:

ip_address - is an ip address host where Blackbox is installed.

ip_address_target_host - is an IP address of the host where is a service

http_2xx - is a probe type.

The Blackbox exporter response is included below:

```
# HELP probe_http_duration_seconds Duration of http request by phase, summed over all redirects
# TYPE probe_http_duration_seconds gauge
probe_http_duration_seconds{phase="connect"} 0.198103346
probe_http_duration_seconds{phase="processing"} 0.206252956
probe_http_duration_seconds{phase="resolve"} 0.12224032900000001
probe_http_duration_seconds{phase="tls"} 0.166810132
probe_http_duration_seconds{phase="transfer"} 0.109561104
# HELP probe_http_redirects The number of redirects
# TYPE probe_http_redirects gauge
probe_http_redirects 1
# HELP probe_http_ssl Indicates if SSL was used for the final redirect
# TYPE probe_http_ssl gauge
probe_http_ssl 1
# HELP probe_http_status_code Response HTTP status code
# TYPE probe_http_status_code gauge
probe_http_status_code 200
# HELP probe_success Displays whether or not the probe was a success# TYPE probe_success gaugeprobe_success 1
```

The probe type should be defined in a BlackBox configuration file. It is possible to define the request's parameters and the expected response. More detailed information about the Blackbox exporter and its configuration file can be found at the Github tool site [9].

2.3.4. Using ELK stack for calculation a slice creation/adaption time.

2.3.4.1. General description

The core KPI "creation/adaptation time" was defined in D4.1. ELK stack was used for measuring this KPI. Filebeat is a component of the ELK stack that is installed on each component of the 5Growth

stack and collects logs. Logstash is a component of ELK stack that has the functionality for calculating the creation/adaption time of a slice. The Logstash was configured in to analyse the Network Service Descriptor (NSD) instantiation at the 5Gr-SO. This configuration analyses the logs produced by the 5Gr-SO and looks for the specific messages. An example of the logs for both the instantiation start and finish is provided below:

[*07/13/2020 05:37:53.561*] INFO SOEc instantiate_ns_process
with nsId fgt-***9f62028***-cbe4-4d07-b89d-e39d07943176, body
{**'additional_affinity_or_anti_affinity_rule'**: None,
'additional_param_for_ns': None,
'additional_param_for_vnf': None,
'flavour_id': **'df_vCDN'**,
'location_constraints': None,
'nested_ns_instance_id': None,
'ns_instantiation_level_id': **'il_vCDN_small'**,
'pnf_info': None,
'sap_data': None,
'start_time': None,
'vnf_instance_data': None}

[*07/13/2020 05:39:55.852*] INFO *****Time measure: SOEc instantiate_ns_process finished for nsId fgt-***9f62028***-cbe4-4d07-b89d-e39d07943175

Logstash analyses the logs with the same nsId, recognizes the timestamp, calculates the difference between these two timestamps, and puts the duration of instantiation to the message.

2.3.5. 5Probe

Passive probes are elements that can sniff the network traffic during the experiments. These probes produce log information with the measurements of some parameters of the sniffed traffic sent and received within the 5G network. These probes are not generating traffic and are not disturbing the 5G network load in any way.

5Probe is a passive probe that extracts information from end-user traffic by analysing the packets with programmable Shallow Packet Inspection using a packet-capture (pcap) library to generate the 5G network metrics.

2.3.6. Data Shipper

Data shippers collect the metrics from the probes and provide it to 5G-EVE IWL (Kafka). These data shippers are deployed in the VNFs/PNFs that belong to a given experiment and are configured to collect the metrics selected in the experiment. At each site there is a local Kafka lightweight framework that eases the process to ship data to a higher layer data collector stack through the Kafka bus. The overall toolchain, according to the selected implementation tools in 5G-EVE, is represented in Figure 13:

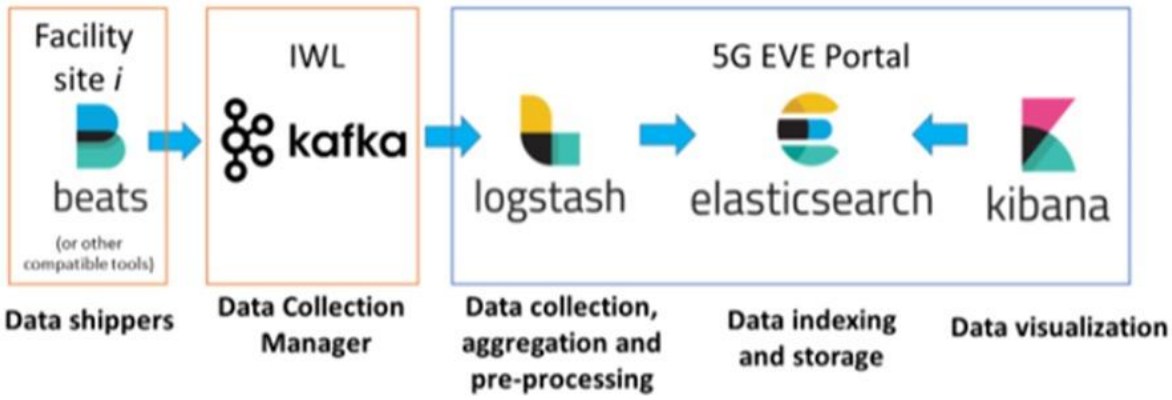


FIGURE 13: ELASTIC STACK TOOLCHAIN WITH BEATS AND KAFKA INTEGRATION

2.3.7. Commercial network testing tools

This section describes those deployment of commercially available network testing tools used at specific sites. The use of these tools for measurement and data collection would require commercial licensing agreements.

2.3.7.1. Tools available at the ALB site

Considering the measure procedures presented in Section 2.2.3, the following test equipment is available, integrated with the 5G-VINNI project.

- IXIA product code 944-1068: Xcellon-Lava 40/100GE2RP,40/100 Gigabit Ethernet, dual speed, load module,2-ports,1-slot with CFP MSA interfaces and full featured L1-3 data planesupportandupto100 routing protocol emulations per port. The load module is compatible with the XGS12SD12-slot, standard performance rack mount chassis bundle (940-0011), XGS12-HS 12-slot, high-speed performance rack mount chassis bundle (940-0006), XG12 12-slot, rack mount chassis (940-0005), XGS2-SD 2-slot,3RU standard performance chassis bundle (940-0010),XGS2-HS 2-slot, 3RU high-speed performance chassis bundle (940-0012) and the XM2-02 desktop chassis(941-0023).
- IXIA product code 944-1142: NOVUS 10/1GE16DP,16-port, SFP+/10GBASE-T Dual-PHY 10GE/1GE/ 100M loadmodule,1-slot Dual-PHY with 16-port search of the SFP+ and 10GBASET RJ45 physical interfaces, L2-7 support. Compatible with the XGS12-SD rack mount chassis (940-0011), XGS12-HSL rack mount chassis (940-0016), XGS2-SD 2-slot standard performance chassis (940-0010), and2-slot high performance XGS2-HSLChassis (940-0014).
- IXIA product code 944-0098: LSM1000XMVDC16-01 Gigabit Ethernet Load Module, 16-Port Dual-PHY (RJ45 and SFP) 10/100/1000 Mbps; The load module is compatible with the XGS12-SD 12-slot, standard performance rack mount chassis bundle (940-0011), XGS12-HS 12-slot, high-speed performance rack mount chassis bundle (940-0006), XG12 12-slot, rack mount chassis (940-0005), XGS2-SD 2-slot, 3RU standard performance chassis bundle (940-0010), XGS2-HS 2-slot, 3RU high-speed performance chassis bundle (940-0012) and the XM2

desktop chassis (941-0023); 1GB Port CPU memory, full featured L2-L7 with FCoE enabled. Fiber Ports REQUIRE SFP transceivers, options include SFP-LX, SFP-SX, and SFP-CU.

- IXIA product code 940-0011: XGS12-SD, 12-SLOT CHASSIS BUNDLE. This chassis bundle includes: Chassis frame assembly with Standard Star Topology Sync, Processor Module running Windows 7 Operating System, Fan assembly module, 6000W power supply module. Includes installed IxOS software.

2.3.8. Example additional tools

The following table lists a selection of tools that could be integrated with the 5Growth monitoring platform providing alternatives to monitor all the CKPIs. In this table, "X" represents the CKPIs that are directly measured by the tool, while "+" represents that the CKPIs are only partially measured.

Five types of tools are identified, according to their deployment patterns:

1. **Client and server**-based tools aim to measure the end-to-end CKPIs, such as the RTT and, throughput, in a stateful fashion (e.g., via TCP sessions).
2. **Single node** tools send (arbitrary) traffic towards any node and measure the response times of either the host node itself or the service the specific node is hosting by crafting the requests. They also measure availability (of the service or node) and can measure jitter.
3. **In-network** tools passively observe the traffic flows and report the statistics about them. These are especially powerful when combined with clear-text transport protocol such as RTP, where a single passive probe can report the exact packet loss rates, RTTs, etc. experienced by the streaming applications by parsing the content of RTCP messages.
4. **In-hypervisor** tools are deployed on the host OS of the servers and provide fine-grained visibility over all the services hosted at the same machine.
5. **Hardware** tools are less flexible, as they cannot be deployed through software containers/package. However, in turn they are more powerful for testing the lower layer of a 5G network such as emulating large number of UEs, RSSI degradation, etc.

TABLE 24: ADDITIONAL TOOL LIST AND CORRESPONDING CORE 5G KPIS

Tools	Core 5G KPIs								Type
	CKPI-1	CKPI-2	CKPI-3	CKPI-5	CKPI-6	CKPI-7	CKPI-8	CKPI-9	
Iperf [10]	X	X	X					X	Client and server
Fping [11]	X	X		+				X	Single node
Moongen [12]	X	X	X	X				X	Single node
Curl [13]	X			X					Single node
Ntopng [14]	+	+	X	X			X	+	In-network
Cilium [15]	X	X	X	X	X		X	X	In-hypervisor
P8800S UeSIM [16]	X	X	X		X	+		X	Hardware

2.4. Experiment Catalogue

The 5Growth monitoring platform provides a framework to monitor both SKPIs and CKPIs status in a consistent and reproducible way. Under the hood, it orchestrates a series of measurement

experiments, each leveraging one or more monitoring tools and translating their resulting data stream into KPI status. The 5Growth experiment catalogue holds all the experiments currently defined for a given site.

Such a catalogue can be used in at least two different ways. First, it complements the slice blueprint with a way to (partially) validate SLAs by tying the blueprint with experiments evaluating (most of) the SKPIs relevant for the vertical which will use that slice. Second, it might be undesirable to run all the experiments at all time, e.g., an experiment measuring the maximal achievable throughput could actively interfere with other active services in the same slice. Thus, the catalogue enables experimenters to run individual experiments at will, e.g., according to a schedule to validate a CKPI that is not expected to vary.

This section first describes the methodology to define an experiment, using a service-based approach. Then, it discusses the abstract interface that tools should support to be used across the experiment in a consistent way.

2.4.1. Defining experiments

The 5Growth platform adopts a service-first approach tailored towards verticals. The experiment catalogue follows the same philosophy. More precisely, the experiment catalogue decouples the measurement intents (i.e., what KPI does the experiment measure), from their realization (i.e., using which tools and at what cost). By enabling to group experiments by their produced KPIs, this approach has two notable benefits. First, multiple experiments measuring the same KPI (but in different ways) improve the level of confidence in the results if they are consistent. Additionally, this also allows hinting possible issues or unexpected side-effects if they show inconsistencies (e.g., comparing idle latencies to latencies under load to detect buffer bloat). Second, it enables to dynamically select the experiment to use based on runtime constraints (e.g., the network is “live” hence “costly” tools cannot be used, some nodes are unavailable).

This service-based design is reflected in the experiment definition process. First, it defines the goal of the experiment, i.e., the targeted measurable KPI. If the measurement goal is a SKPI, it must be further decomposed into the set of specific CKPIs as well as a combination function that will allow evaluating the SKPI status according to the selected CKPIs measurements contributing to it (which is the identity function for a CKPI). Note that this combination function can be seen both as a data consumer and as a producer, following Section 3 nomenclature. Then, the experiment needs to be tied to a set of tools as well as their placement enabling to measure all CKPIs. It is worth noting that this tool selection will implicitly determine the resource cost of the experiment, as well as create the operational constraints (e.g., on the availability of a vantage point such as in-network taps). With these constraints, it will be possible to ultimately determine whether an experiment can be performed at any given time, enabling to automate the execution of experiments and/or schedule them over time.

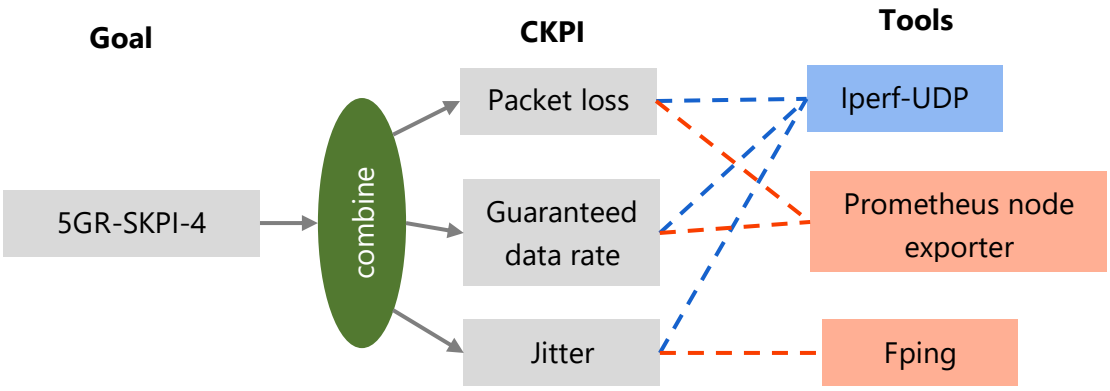


FIGURE 14: TWO SET OF TOOLS APPLICABLE TO EXPERIMENTS MEASURING P3UC2-SKPI-1

We illustrate this process using an example shown in the figure above. More precisely, we aim to define an experiment measuring whether a "High-resolution Real-time Video Quality" SKPI is met. Section 2.2 defined the decomposition into CKPIs that should take place for that SKPI. In our sample network, lab measurements (similar to those described in Section 2.2.1.1) have shown that the quality was deemed "good" when the packet loss rate, data rate, and the jitter were respectively <0.1%, >=15Mbps, <5ms; and "bad" otherwise. This allows to define the SKPI scoring function for our experiment, as a conjunction of thresholds to be checked for each CKPI. This is illustrated in the above figure by the green oval.

In this example network, two sets of tools are available and can be selected, depending on the operational constraints. Both set of tools are deployed on the video sender and receiver as no in-network tool (e.g., nprobe with an RTP/RTCP decoder) is available. On one hand, iperf in UDP mode can be used to perform an experiment when the network is idle/not running production traffic. Indeed, iperf can estimate all the 3 CKPIs by setting its sending rate to the ideal data rate (i.e., 15Mbps), and logging on the video receiver the effective data rate and resulting losses and inter-packet jitter. Iperf, however, comes with a relatively large resource cost, as it consumes the bandwidth normally used by the service itself. On the other hand, the orange pair depicted in the figure (i.e., fping & Prometheus node exporter) can instead be used to monitor a running service instance with a near-zero resource usage, but with the operational constraint that the service must be running. More precisely, fping is used for the end-to-end jitter measurement (i.e., at a cost of a few kbps). The two other CKPIs can then be inferred by comparing the Prometheus TX/RX stats at the video sender and receiver to compute the achieved data rate and loss rate.

These two sets of tools thus enable defining two different experiments in our catalogue for the chosen SKPI. This allows to compare synthetic results (i.e., iperf-generated traffic) with the production results (i.e., by instrumenting a running system) to assess the behaviour of the network under load, as well as the consistency of the SKPI decomposition and service model. Their complementary operational constraints also enable to continuously run one of these experiments to always validate the (feasibility of the) SKPI.

2.4.2. Tool abstraction

Experiments are realized by collecting and analysing measurements from one or more monitoring tool. More precisely, tools' results are meant to be converted into CKPIs, following the nomenclature of Section 2.2 as well as D4.1. This section first presents the tool meta-data which is needed by the 5Growth Monitoring platform to operate it properly. Then, it discusses how the monitoring platform discovers its available tools. Finally, it concludes by listing various tools that could be integrated in 5Growth, complementing the already available tools.

2.4.2.1. Tool metadata

Figure 15 illustrates the five sets of metadata that are needed to integrate a monitoring tool with the 5Growth platform. They are described below at a high-level.

1. **Tool packaging.** This metadata describes the components that are required to use the tool with the 5Growth monitoring platform such as the eventual build script (e.g., a container image, statically linked executable), the probe exporter that will ensure that the produced data is compatible with the 5Growth data ingestion pipeline.

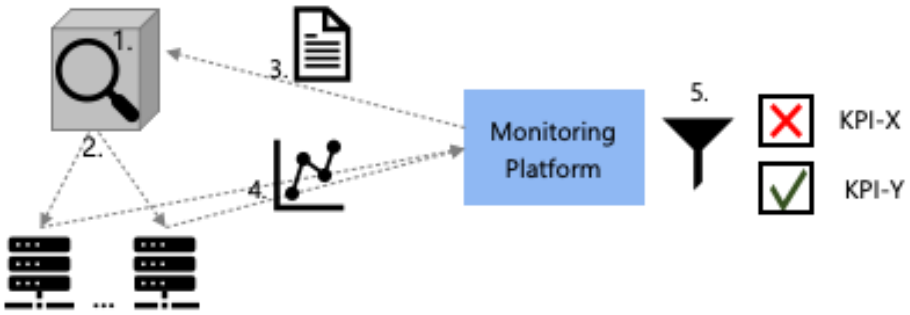


FIGURE 15: ABSTRACT TOOL LIFECYCLE AND METADATA

2. **Tool deployment.** This set of metadata controls where (i.e., on which node) the tool can be used (e.g., hardware appliances are available in fixed locations, Prometheus exporters are only on VNFs), and how to deploy it there (e.g., install script, activation via RPC). Once a tool is deployed on a node, the 5Growth Monitoring platform Orchestrator can then start to use it, until removed (e.g., hardware appliances are always deployed, VNF exporters' lifecycle follow the one of their VNF and are thus dynamically discovered).
3. **Tool configuration.** This defines how the monitoring platform can set up the tool for a given experiment (e.g., target address and tool parameter).
4. **Tool measurements.** This defines how the monitoring platform instructs the tool to start exporting its measurements (as well as how to stop it), as the tool starts its the role of data source (cf. Section 3.1). This metadata set also defines how resource-intensive (e.g., CPU, RAM, throughput) a given tool is, depending on its configuration.
5. **CKPI adapters.** This is arguably the most important set of metadata for a tool, as it defines the tool to CKPI mapping. More precisely, it defines the data-consumers (cf. Section 3.3) that

may be needed to convert the measurements produced by the tool into the expected format for the given CKPI.

2.4.2.2. Tool discovery

We normally distinguish between two classes of tools that can be made available to the 5Growth data infrastructure.

1. **Node capabilities.** This class of tools encompasses all tools that are statically deployed on a given network node. For example, the physical hardware appliance presented in Section 2.3.5, the Prometheus node exporter that reports statistics about a given server. These tools have a fixed set of capabilities, documented as part of the network topology. As such, the monitoring platform can be statically configured to use them.
2. **VNFs.** This class contains tools that can be deployed on demand on a target node (e.g., remote deployment, daemon activation). Making them visible to the monitoring platform thus requires the definition of a tool registration protocol. We leverage for this the already existing Prometheus probe infrastructure, enabling to seamlessly add and remove tools from the data ingestion pipeline.

3. Data infrastructure

The figure below shows a high-level overview of the different modules in a general data engineering platform. It mainly consists of five modules: Data Connect, Data Ingest, Data Analysis, Data Storage and Data Visualization. Note that the interconnection between each of these modules are shown loosely and there may be several interfaces connecting those modules depending on the use case.

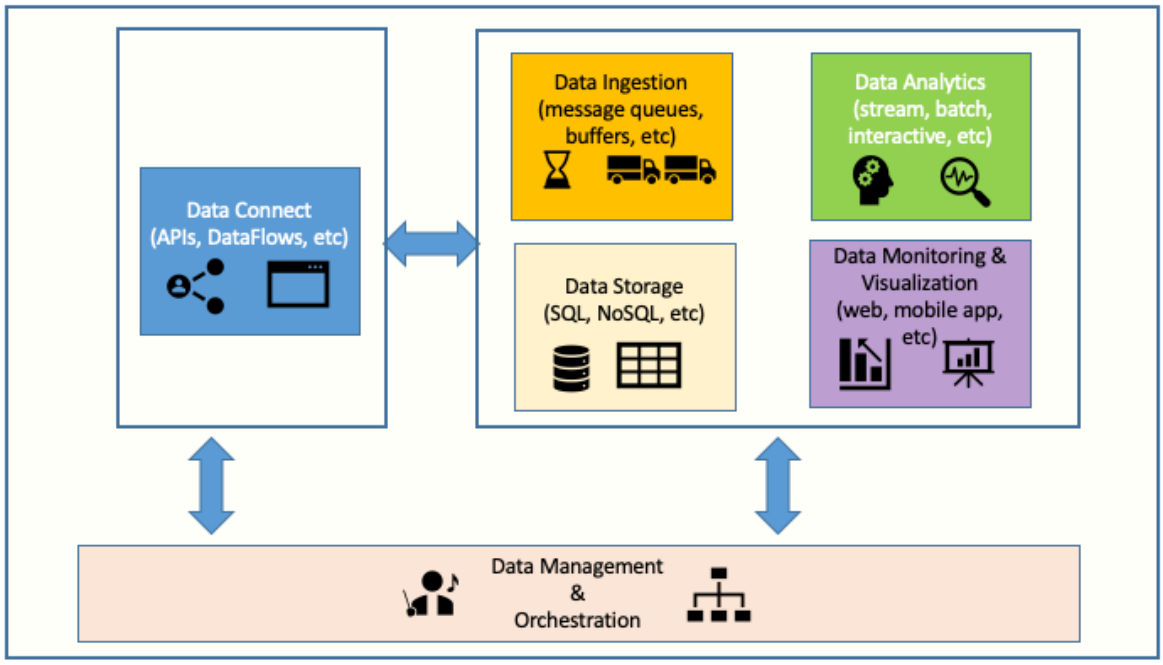


FIGURE 16: GENERAL DIAGRAM OF DATA ENGINEERING PIPELINE COMPONENTS

A summary of the characteristics of open-source frameworks in data engineering pipeline is shown in the following table.

TABLE 25: FRAMEWORKS AND CHARACTERISTICS

Frameworks		Characteristics
Data Connection		- API service to receive data from its sources. - Used to put data into temporary storage
Data Ingestion		- Acts as a distributed pub-sub messaging system - Enables ingestion of data into cluster - Manages data (transformation and enrichment) - Messaging system (asynchronous and in real-time) - Avoids overwhelming the data pipeline
Data Processing & Analytics	**Processing**	- Analyzes data in batch, interactive, streaming or real time - Model/Inference/Prediction Serving - Distributed data structures (RDDs, DataStreams and DataSets)
	Log Analytics	- Continuous log stream processing (parsing, joining, augmenting)

		- Real-time application performance monitoring - Generate new derived data as the results of queries from logs, metrics, web applications, data stores
	Query	- Enables query capabilities over big data (e.g. Hadoop cluster), NoSQL databases and cloud storage - Provide OLAP capability to big data - Supports SQL (or a subset) functionality - Streaming, batch and hybrid modes
	Search	- Execute search engine over big data
	Analysis	- Classification, Clustering, Regression, Deep Learning, Distributed Model Training - Distributed Reinforcement learning, Hyperparameter search
Data Storage		- Graph databases - In-memory and analytics databases - Distributed storage systems
Data Visualization & Monitoring		- 1D, 2D and 3D visualization - Temporal and spatial analysis - Monitoring pipelines, dashboards, alerts and notifications
Management, Orchestration and Scheduling		- Resource management, defining and scheduling workflows/tasks and services across cluster, data center and cloud - Container orchestration, auto-scaling

3.1. Data sources

The validation of the 5Growth KPIs is performed using information coming from a variety of data sources. We classify these data sources in three main categories:

- **5Growth integrated data sources:** Include all the data sources that have already been fully integrated with the 5Growth Monitoring Platform and which, therefore, can be natively used for the KPI validation.
- **ICT-17 data sources:** Include the data sources provided by the monitoring systems available in the ICT-17 platforms the 5Growth architecture interacts with (i.e. 5G EVE and 5G-VINNI). These data sources will be used during the trialing campaigns of the use cases either entirely or partially deployed in the ICT-17 sites. Their integration with the 5Growth Monitoring Platform requires the implementation of specific data adaptation blocks, in charge of translating between the information/data models and monitoring data collection procedures adopted in the 5G EVE and 5G-VINNI platforms and the ones at the 5Growth Monitoring Platform.
- **Other external data sources:** Embrace data sources generated from any other element external to the 5Growth environment and its related ICT-17 sites, which will be used during the use case KPI validation. As for the previous case, some adaptation and processing of this kind of monitoring data is required to enable their usage on the Monitoring Platform.

The following figure provides a high-level overview of the data sources and the different mechanisms used to feed the information to the Monitoring Platform. The following subsections provide further detail about each category.

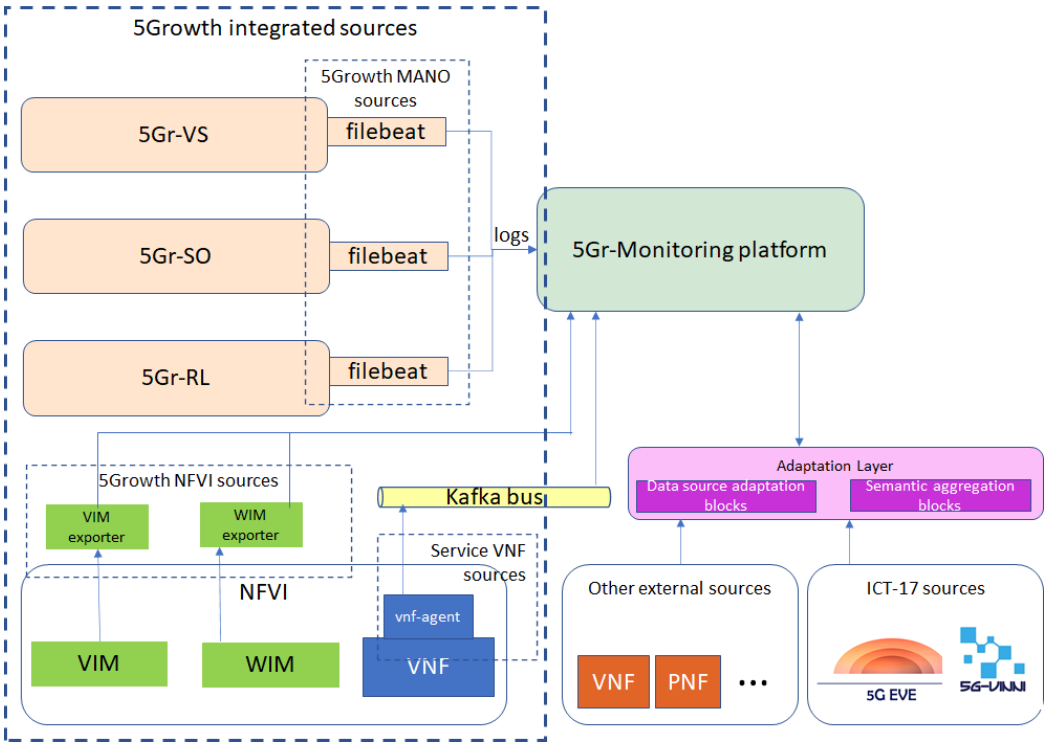


FIGURE 17: 5GROWTH DATA SOURCES

3.1.1. 5Growth integrated data sources

5Growth supports the integrated data sources providing information from different parts of the platform, as illustrated in the previous figure:

- 5Growth NFVI.
- Service VNFs.
- 5Growth MANO stack.

All the 5Growth data sources feed the monitoring platform, which is responsible for processing the information and provides the required capabilities for querying of monitoring data or alert notifications towards the rest of the platform components. The 5Growth Monitoring Platform is flexible enough to accommodate additional types of data sources in the future. For example, we may envision an architecture where the 5Growth forecasting and prediction mechanisms generate data that are also collected in the monitoring platform for further processing. In this case, the forecasting and prediction modules would become another type of 5Growth integrated data source.

The 5Growth NFVI monitoring embraces the information coming mainly from the VIM and WIMs. This information is mostly used to monitor the usage of the NFVI computing, storage, and networking resources. In particular, 5Growth inherited the 5G-TRANSFORMER NFVI data sources

[17]: Openstack, Opendaylight, ONOS, Kubernetes which are fully compatible with the 5Growth Monitoring Platform through the adoption of dedicated "data exporters".

5Growth also uses monitoring the information produced by the service VNFs themselves. This information is usually used to provide: (i) VNF instance specific information (e.g., CPU, memory usage, statistics related to incoming/outgoing traffic on the virtual network interfaces, etc.); (ii) application-specific information used to extract service metrics and KPIs (e.g., the number of end-users connected to a given service). The 5Growth monitoring platform provides the required mechanisms to support automated installation and configuration of the VNF exporters on the VNFs during the service instantiation phase (via RVM agents as described in Section 4.2). In Section 2.3, it is detailed one of the tools developed in the project to obtain the link related information.

As in the 5G-TRANSFORMER project the monitoring jobs in the 5Growth Monitoring Platform are configured so that the information coming from the VNF instances is associated with some specific labels, namely: the VNF instance id, the network service instance id and the VNFD id. In 5Growth the labels are extended to support the exporter id (used to identify the actual VNF exporter for retrieving the information). All these labels can be used then to aggregate and query the information available on the monitoring platform.

The 5Gr-VS, 5Gr-SO and 5Gr-RL also act as data sources to the monitoring platform. The log files of each of these components can be processed to obtain metrics related to the service lifecycle management such as service the creation and the service termination times. In this case, filebeat agents are used to mediate the collection of the information available in the log files. We summarize the exporter for the different data sources and provide examples for the different metrics in the table below.

TABLE 26: 5GROWTH DATA SOURCE EXAMPLES

5Growth data source	Exporter	Example metrics
5Growth NFVI	VIM exporter	• Status of Openstack resources (i.e., Openstack compute node load, number of VM instances, etc) • Kubernetes status (i.e., cluster status, pod status)
	WIM exporter	Inter-PoP transport network connection statistics (i.e., port and node status, packet loss, received packets, etc.)
Service VNFs	VNF exporter	• Service specific metrics (i.e., vCDN cache chunk hits, vCDN number of active users) • Per VNF instance resource usage (i.e., maximum, and average usage of CPU, disk or memory, link data rates)
5Growth MANO	filebeat	Lifecycle management metrics (i.e., service deployment time, service termination time, etc)

3.1.2. ICT-17 data sources

5Growth will execute the use case trialling campaigns using both the ICT-17 5G EVE and 5G-VINNI platforms. Each platform provides different monitoring and KPI validation capabilities which shall be used during 5Growth trialling campaigns to validate the KPIs of the targeted use cases. This section details the different sources of information available for each platform, how to specify the metrics that need to be collected and how the data is retrieved by each platform. It also describes the available interfaces and mechanisms to consume the produced monitoring and KPI information, including the data aggregation mechanisms supported by each platform.

3.1.2.1. 5G-VINNI data sources

The 5G-VINNI platform uses as its service platform the SONATA Service Platform [18]. Figure 18 shows a high-level picture of the service platform architecture with the highlighted Monitoring Manager.

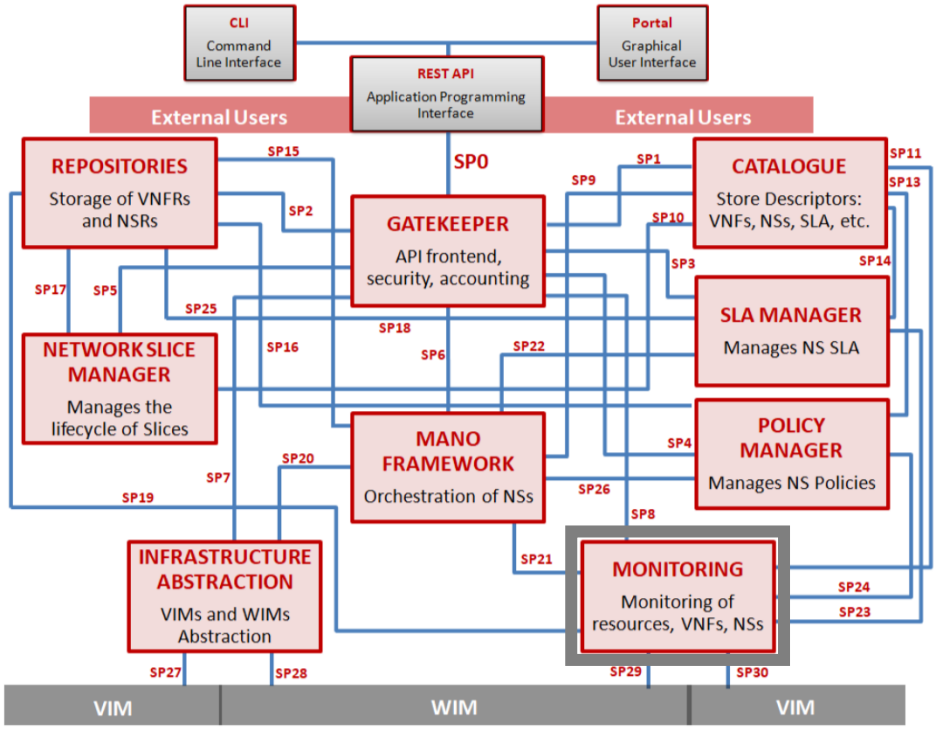


FIGURE 18: FINAL VERSION OF THE SONATA SERVICE PLATFORM

As represented, the Monitoring Manager collects the monitoring data from a number of other components (i.e., interfaces labelled as SP11, SP19, SP21, SP23, SP24, SP29 and SP30), stores it, and makes it available to the external systems through the Gatekeeper (shown at the top as the interface named SP8).

A more detailed architecture of the Monitoring Manager is shown in Figure 19 (see [18]).

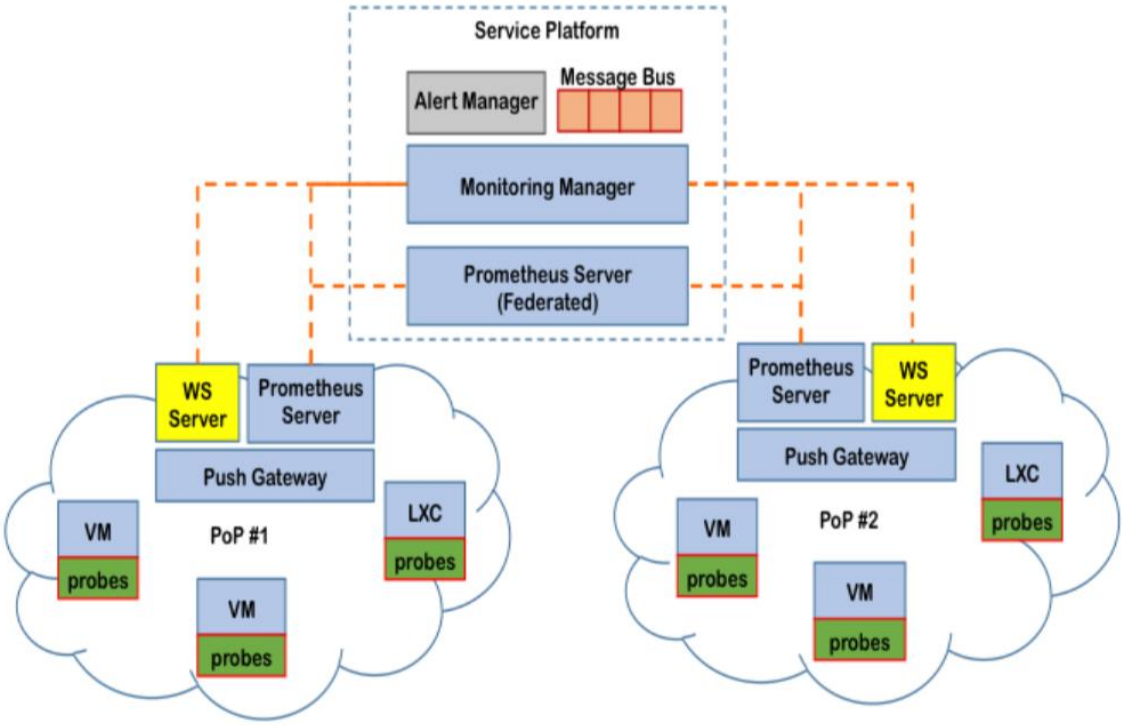


FIGURE 19: A ZOOM IN INTO SONATA MONITORING MANAGER ARCHITECTURE

Data collection / monitoring

There are two distinct kinds of metrics that can be collected in a SONATA SP, both coming from the VIMs, either VM- or Kubernetes-based interface (named SP30, in the above figure), and from the WIMs (interface named SP29, above):

- Infrastructure metrics: these metrics are collected from the NFVI exporter, based on Ceilometer. This collects data from VMs and from Kubernetes clusters hosting VDUs and/or VNFs, and from the SDN exporter which collects data from an ODL controller to monitor the performance of the SDN-enabled infrastructure network (see [19], [20]).
- Service specific metrics: this kind of metrics is collected from the specific probes the service developer has included in the service VNFs/VDUs. VNFDs of those VNFs should include a reference to the relevant metrics (see [21] for an examples). Upon a successful service instantiation, the Monitoring Manager is notified (by the MANO Framework, flow SP21 in the previous figure) about these metrics that need to be collected from then on.

Among the infrastructure monitoring metrics supported by the probe include total/used cores, total/used Instances, total/used RAM size, total/used Floating Ips, etc.

For Kubernetes-based VIMs, probes need to use the sidecar design pattern, as shown in Figure 20.

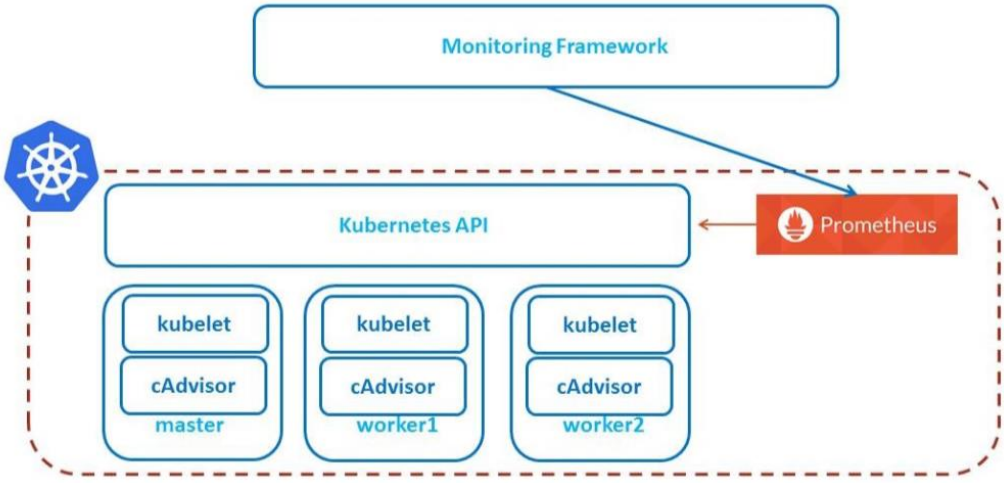


FIGURE 20: THE SIDECAR DESIGN PATTERN, ENABLING COLLECTION OF MONITORING DATA FROM KUBERNETES

```
...
monitoring_parameters:
      - name: "vm_cpu_perc"
        unit: "Percentage"
      - name: "vm_mem_perc"
        unit: "Percentage"
      - name: "vm_net_rx_bps"
        unit: "bps"
      - name: "vm_net_tx_bps"
        unit: "bps"
...
```

FIGURE 21: A SIMPLE EXAMPLE OF MONITORING PARAMETERS SPECIFIC TO A SERVICE

```
...
    monitoring_parameters:
      - name: "cpu_util"
        unit: "Percentage"
      - name: "memory_usage"
        unit: "MB"
    snmp_parameters:
      version: "v2"
      auth_protocol: "None"
      security_level: "None"
      username: "public"
      port: 161
      interval: 10
      oids:
        - oid:
"1.3.6.1.4.1.8072.1.3.2.3.1.2.19.115.105.112.112.111.83.101.114.118.101.114.83.101.
115.115.105.111.110.115"
          metric_name: "sippoServerSessions"
          metric_type: "gauge"
          unit: "number"
          mib_name: "NET-SNMP-EXTEND-MIB"
        - oid:
"1.3.6.1.4.1.8072.1.3.2.3.1.2.22.115.105.112.112.111.83.101.114.118.101.114.67.111.
110.102.101.114.101.110.99.101.115"
          metric_name: "sippoServerConferences"
          metric_type: "gauge"
          unit: "number"
          mib_name: "NET-SNMP-EXTEND-MIB"
...
```

FIGURE 22: A MORE COMPLEX, SNMP-BASED EXAMPLE OF SERVICE MONITOTING PARAMETERS

Data models and storage

Available metrics are dynamic because they depend on the entities to be monitored in each deployment. Table 27 shows a list of the metrics from a specific deployment which monitors the VMs, Containers and an OpenStack-based VIM.

TABLE 27: EXAMPLES OF METRICS

Container metrics	VM metrics	VIM metrics
cnt_block_in_MB	vm_cpu_perc	vim_maxImageMeta
cnt_block_ou_MB	vm_disk_total_1k_blocks	vim_maxPersonality
cnt_cpu_perc	vm_disk_usage_perc	vim_maxPersonality
cnt_created	vm_disk_used_1k_blocks	vim_maxPersonalitySize
cnt_mem_limit_MB	vm_mem_free_MB	vim_maxSecurityGroupRules
cnt_mem_perc	vm_mem_perc	vim_maxSecurityGroups
cnt_mem_usage_MB	vm_mem_total_MB	vim_maxServerGroupMembers
cnt_net_rx_MB	vm_net_rx_MB	vim_maxServerGroups
cnt_net_tx_MB	vm_net_rx_bps	vim_maxServerMeta
cnt_status	vm_net_rx_pps	vim_maxTotalCores
	vm_net_tx_MB	vim_maxTotalFloatingIps
	vm_net_tx_bps	vim_maxTotalInstances
	vm_net_tx_pps	vim_maxTotalKeypairs
	vm_up	vim_maxTotalRAMSize
		vim_totalCoresUsed
		vim_totalFloatingIpsUsed
		vim_totalInstancesUsed
		vim_totalRAMUsed
		vim_totalSecurityGroupsUsed
		vim_totalServerGroupsUsed

All this monitoring data is stored in Prometheus, as shown in Figure 19, above.

Monitoring data exposure

Exposing the monitoring data to external entities / functions is done through the defined SONATA API via the following endpoints:

- `/api/v3/monitoring/data`: returns data in JSON.
- `/api/v3/monitoring/graphs`: returns HTML graphs (e.g., enabling its insertion into HTML pages).

This monitoring data can also be used internally. For example, flows SP29 and SP30 in Figure 18 above connect the Monitoring Manager with the SLA Manager and the Policy Manager. This sharing is done through a message broker (i.e., RabbitMQ, in SONATA) that is shared between these modules. This solution allows a very efficient triggering of events that then spills into other functional areas (in this case, SLA Management and Policy Management).

3.1.2.2. 5G EVE data sources

The 5G EVE platform allows the definition and execution of experiments, with the objective of validating service KPIs in different environments and contextual situations. During the experiment definition phase, the experiment designer needs to specify which metrics need to be collected, and

how these metrics are to be used to calculate the service KPIs. There are two main types of metrics defined on the 5G EVE platform:

- Infrastructure metrics: metrics related to the performance of the 5G network and collected directly from the infrastructure on each 5G EVE site facility, through dedicated network probes or monitoring functionalities available in the RAN controllers. Examples of such metrics include latency, uplink/downlink bandwidth, and availability. Their collection is automatically managed by the 5G EVE platform, without the need to implement any dedicated agent.
- Application metrics: metrics related to the vertical service and produced by the service applications or by some additional appliance defined in the experiment. The 5G EVE platform relies on data shippers installed in the service VMs to collect this kind of metrics.

In both cases, when the experiment is deployed, the different data shippers are automatically configured by the 5G EVE platform to push the metrics into the monitoring system, sending the data into site-specific Kafka buses and assigning them topics which facilitate their processing (e.g., indicating the particular experiment, service and originating source). In case of application metrics, the script defining the steps of the experiment configuration and execution must include, respectively, the commands to configure and to start the data shippers in charge of collecting the related monitoring data. This approach allows to automate the collection of application metrics during the lifecycle of the experiment, without any need of manual intervention.

During the execution of a 5G EVE experiment, the Result Analysis and Validation (RAV) component of the 5GEVE platform will automatically start collecting the metrics made available on the Kafka buses, in order to compute the KPIs defined in the experiment. Such KPIs are also pushed on the Kafka bus and made available on the 5G EVE monitoring platform. Moreover, at the end of the experiment, the RAV produces a report containing both metrics and KPIs, validated according to the thresholds defined in the experiment descriptor. This report can be visualized on the web GUI of the 5G EVE Portal.

The experimenter can also access the 5G EVE Portal to monitor the progress of the experiment during its execution and visualize in real-time the graphs for the collected metrics and KPIs, as shown in Figure 23 . Still from the 5G-EVE Portal, the experimenter can download the CSV files containing the values of the collected metrics and KPIs. Such files can be used as data source for the 5Growth Monitoring Platform e.g., using an approach based on filebeat as in the case of the 5Gr-VS/SO/RL logs. However, at the moment, the download of these files cannot be automated and needs a manual intervention. At the time of writing, the 5G EVE project is evaluating the possibility to provide REST APIs for requesting the download of such CSV files in a programmable manner.

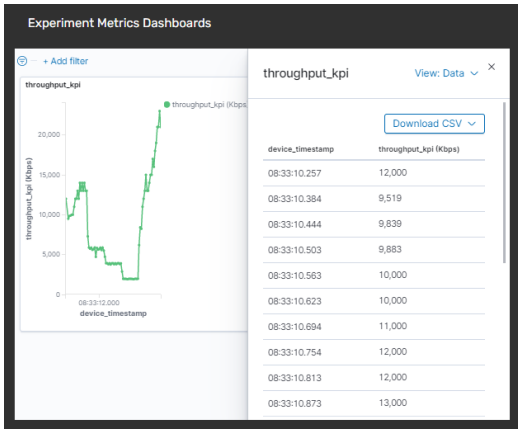


FIGURE 23: 5G EVE SAMPLE REPORT

3.1.3. Other external data sources

The 5Growth KPI validation processes also foresee the possibility of using data sources which are outside of the scope of the ICT-17 and the 5Growth integrated sources. The information produced by these data sources will likely need some processing and adaptation to be used together with the Monitoring platform. An example of external data sources is the case of YANG-based agents.

3.1.3.1. YANG-based devices

5Growth vertical use cases present some requirements which are difficult to address by legacy network management protocols. Therefore, it is necessary to address the deployment and network management in a more automated manner, thus avoiding the high degree of interaction with the user currently existing in legacy deployments. In this context, model-driven network management is consolidating, based on the handling of YANG-based devices. Therefore, this type of data sources must be considered in the context of the 5Growth project as potential external data sources.

YANG-based devices are based on the idea of applying data model languages to formally describe data sources, defining the semantic, syntaxis, structure and restrictions of the management objects. Precisely, the language used for the data modelling is YANG. It is a technology that allows creating data models, defining the organization of these data within the model as well as the restrictions on them. From an architectural point of view, both client and server are driven by the content of YANG modules. As a result, the client is aware of the data supported by the server (data source, device) and the server knows which are the data rules and the behaviour it should have.

YANG-based devices include the definition of the modules as metadata, available to the protocol engine processing the incoming requests from the clients. Apart from processing the incoming requests, this engine uses the metadata to parse and verify any request, perform the requested operation, and return the result to the client.

Regarding its use, a specification written in YANG language is known as *YANG module*, forming a set of such modules what is usually known as *YANG model*. Therefore, a YANG model is focused on the

specification of the data a client manages and observes, using standard operations. Moreover, each YANG module defines the data hierarchy that can be used by the transport protocol operations.

They key aspects regarding YANG-based devices is that data models defined on the device are independent from the transport protocols and data coding languages used for accessing the information.

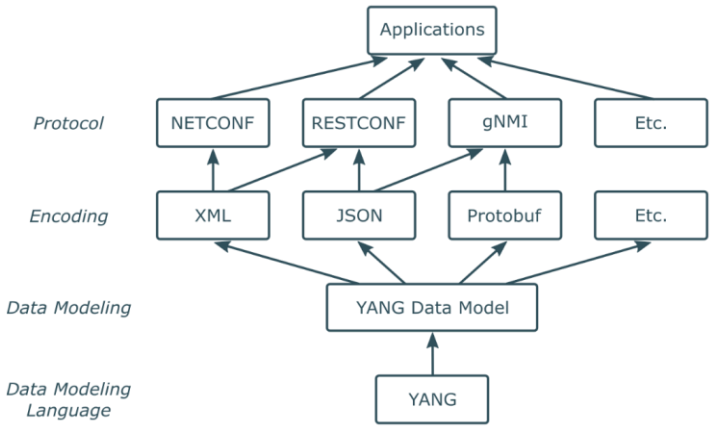


FIGURE 24: MODEL-DRIVEN DATA MANAGEMENT COMPONENTS

Figure 24 depicts the different components that are available for accessing and coding YANG-based devices. Once the YANG data model is defined and implemented, a client can select the best encoding (XML, JSON, Protobufs, etc.) and a transport protocol (NETCONF, RESTCONF or gRPC/gNMI). Considering this type of data sources are not associated to the transport protocol to access the information, this provides more flexibility in collecting data from the different network devices, functions and technologies (e.g., optical, transport, etc.) associated to the targeted 5Growth use case scenarios.

3.2. Data aggregation

Data aggregation is the backbone of both stream and batch processing and is also named as data ingestion in Figure 16: General diagram of data engineering pipeline components general data engineering pipeline diagram show in Figure 16. Data aggregation systems mainly provide a data integration layer for decoupled event-driven applications. Through data aggregators, new producers and consumers can be added, as well as extending the complexity of applications. An example of a general data ingestion architecture with Kafka as the central broker and Zookeeper as context registry is given in Figure 25.

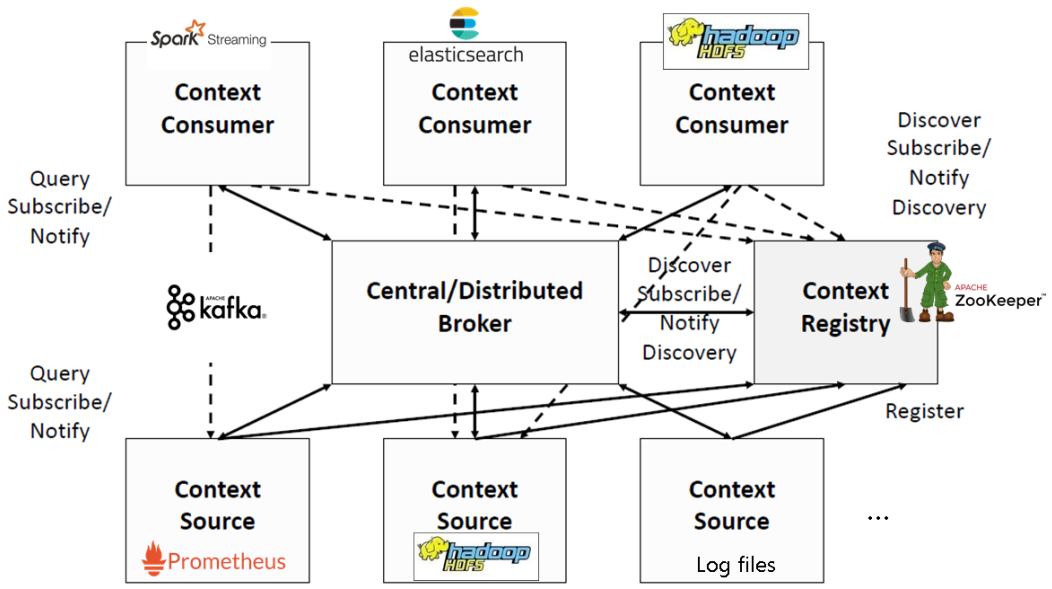


FIGURE 25: AN EXAMPLE OF DATA AGGREGATOR ARCHITECTURE

In terms of communication during data ingestion, two types of message delivery patterns may emerge. One is **queuing** and the other one is **streaming**.

In **queuing**, the ordering is not important as all the events are taken from the message queue at the device/user into a system. Some examples are payment, transaction processing where they are mostly used for mission-critical systems. Message queues provide asynchronous communication protocols where sender and receiver do not need to interact with message queues at the same time and are common features of data ingestion frameworks. Platforms and protocols such as RabbitMQ, JMS, AMQP and others enables asynchronous data integration among multiple systems by acting as a central hub. They are best suited for message queuing applications, e.g., for real-time transactional services with no tolerance to data loss. Hence, consistency and durability are key features of these systems. On the other hand, these systems may suffer from scalability issues during peak traffic (e.g. RabbitMQ is not designed as a distributed system). This is also true for web service/API based architectures.

In **streaming** based communication pattern, event ordering is important so that the behaviour can be analyzed from the sequence of ordered events. Some examples are user behaviour analysis, anomaly detection, web traffic log analysis. In streaming, the loss of data in some parts can be tolerable compared to queuing systems as long as the proper ordering of events are kept. These are mostly suited for OLAP oriented use cases (e.g., business intelligence, dashboards, machine learning, etc).

In large scale deployments, real time data delivery becomes important. During real-time data delivery, delivery time or speed becomes a key metric, and the data engineering architecture should be responsive to events as they happen. For this reason, real-time API is beginning to emerge as both producers and consumers are interested in fast experiences and more instantaneous data transactions. There are various options between multiple APIs when it comes to developing event-driven APIs e.g. open source protocols such as MQTT, WebSockets or streaming protocols such as

Kafka. When establishing connections between the producers and consumers with real-time APIs, in addition to utilized protocols, defining an appropriate streaming semantics becomes key for the considered use cases. In the part, we summarize three fundamental streaming semantics.

3.2.1. Streaming semantics

At Most Once

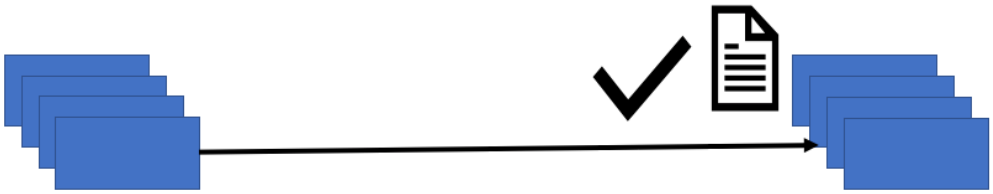
a. Message is pulled once
b. May or may not be received
c. No duplicates
d. Possible missing data

At Least Once

a. Message is pulled one or more times and processed each time
b. Receipt is guaranteed
c. Likely duplicates
d. No missing data

Exactly Once

a. Message is pulled one or more times but processed once
b. Receipt is guaranteed
c. No duplicates
d. No missing data



A summary of the comparison of the three different consistency guarantees based on reliability requirements of the streaming request is given in the following table.

TABLE 28: COMPARISON OF THREE DIFFERENT CONSISTENCY GUARANTEES

Consistency Guarantees	Description	Example Applications
At Least Once	- The message is pulled once or multiple times and processed each time (likely duplicates are received). - During transmission, it is not possible for message to be dropped or lost, hence the receipt is guaranteed. - No data is missed. - Duplicate messages can be allowed and ignored after further processing considering the timestamps of the records. - Typical usage at scale.	- Scaling service applications - Tracking a user's (or device) location via cell phone (or vehicular) records.
At Most Once	- The message is pulled once (no duplicates are allowed), hence the message may or may not be received. - Applicable for use cases where possible missing data is tolerable - Typical usage at scale	- Periodic sensor data (e.g. temperature, humidity) sent in high frequency intervals. - Scaling service applications
Exactly Once	- Message is pulled one or more times but processed once compared to at least once (no duplicates are allowed). - Like at least once, receipt is guaranteed, no missing data is allowed. Hence, there is only one complete successful process. - Especially useful for one-time processing scenarios such as where exactly once processing is crucial even when a broker or instance of function fails. - The complexity of the system increases with exactly once transmission mode of operation which introduces latency. - The benefits of exactly-once-processing should be weight against the drawbacks of additional latency cost. - Some data stream processing frameworks such as Flink Data Stream, Spark Streaming can enable exactly once processing at the expense of increased latency	- Critical business infrastructure - OLTP applications (Transactional message streaming, billing systems, payment processing, etc) - Mission-critical applications. - Other guaranteed lossless event processing applications (Image or video upload and processing in cloud).

	when there is deep pipelines and high volumes of input. - Distributed transactions at scale is difficult.	

3.3. Data consumers

Stream processing is all about processing flows of events. In addition to the data producers that generate new events such as network or application metrics, data consumers need to process these events. The capabilities of data consumers can vary depending on the requirements. There are multiple stream processing tools that can process records from messaging queues such as Apache Spark, Apache Flink, Google Cloud Dataflow, etc. In general data consumers can read and aggregate incoming streaming data, do some processing or send automatic alerts in real time all based on events arriving in a predefined windowing period. Figure 25 gives some examples of data consumers which can be Spark Streaming, HDFS or ElasticSearch depending on the data pipeline building process given in Figure 16.

3.3.1. Data subscription modes

In addition to choosing the best suitable protocols defined above, subscription modes also need to be considered. In this part of this section, we describe different subscription modes.

Partition based

In this subscription mode, data consumers subscribe to each partition of a topic as given in Figure 26.

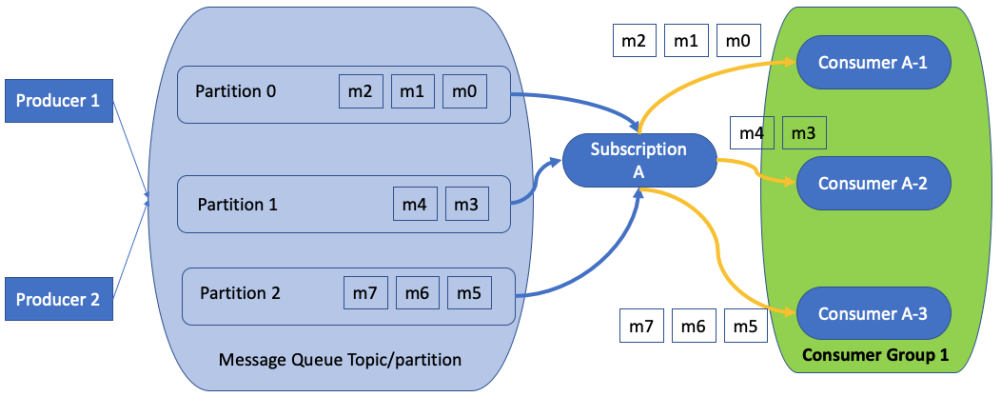


FIGURE 26: PARTITION-BASED SUBSCRIPTION MODE

Shared (consumer group based)

In this subscription mode, each data consumers subscribe to partitions as group-based and data consumers subscribe to each partition of a topic and as given in Figure 27. For example, consumer group 1 can subscribe to different partitions than consumer group 2.

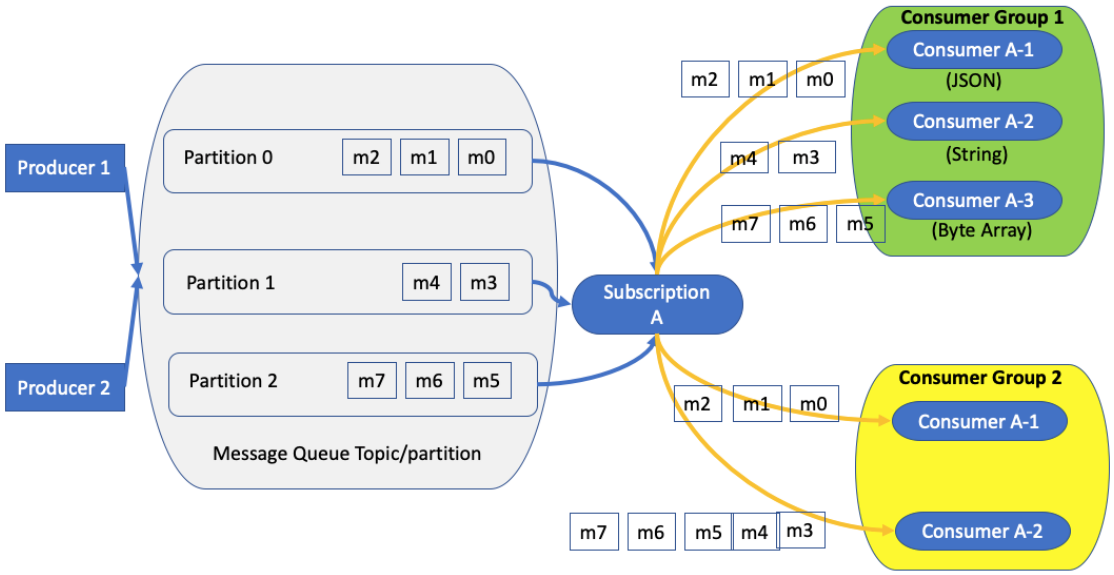


FIGURE 27: GROUP-BASED SUBSCRIPTION MODE

Shared (round-robin based)

In this subscription mode, each data consumer subscribes to a topic and gets delivered in round-robin message delivery manner as illustrated in Figure 28. Delivering messages to multiple consumers in a round-robin approach allows scaling the number of consumers beyond the number of partitions.

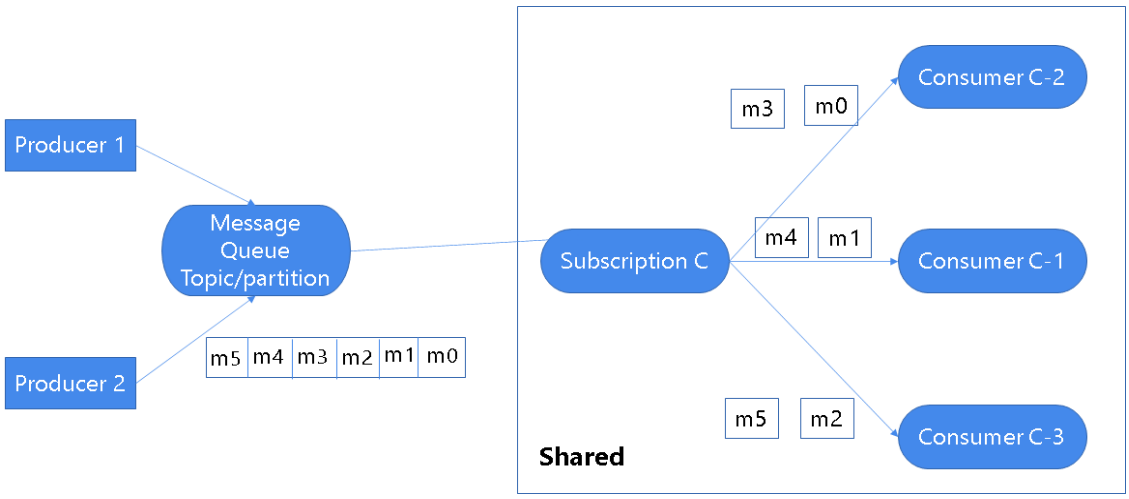


FIGURE 28: SHARED SUBSCRIPTION MODE

Key shared

In this subscription mode, each data consumer subscribes to a topic and the data delivery is done through the key sharing concept as shown in Figure 29. Multiple consumers can attach to the same subscription where the delivery of the messages are distributed across consumers who has the same key. It allows higher scaling as well as ordering guarantees at key level.

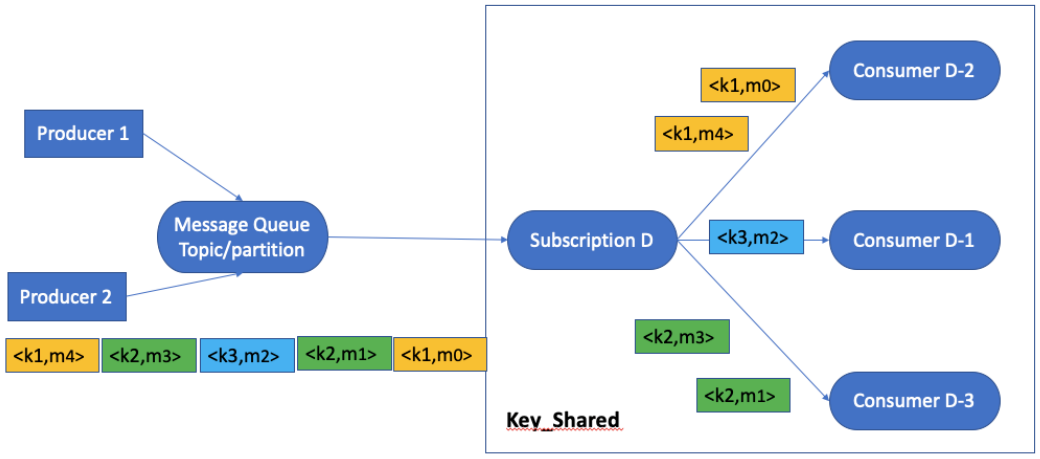


FIGURE 29: KEY-SHARED SUBSCRIPTION MODE

Exclusive

In this subscription mode, each data consumer subscribes to a topic in an exclusive manner and only a single consumer is allowed to subscribe to the data as depicted in Figure 30.

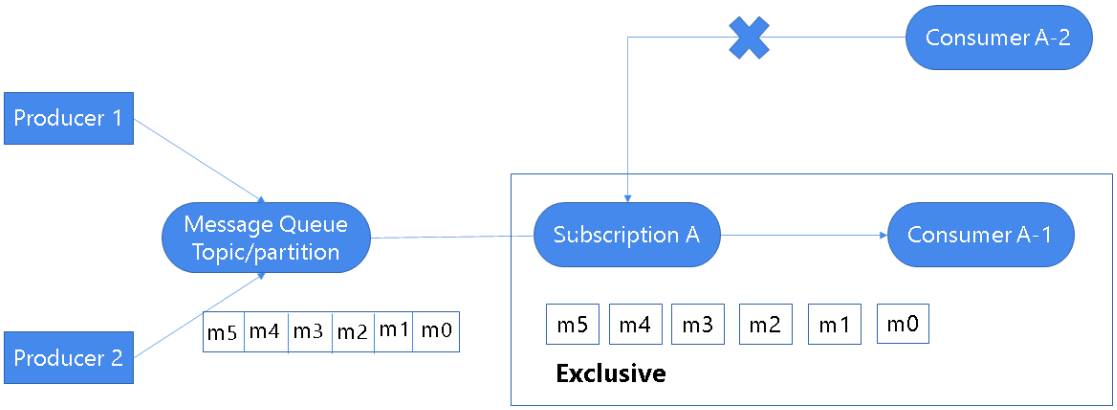


FIGURE 30: EXCLUSIVE SUBSCRIPTION MODE

Failover

In this subscription mode, multiple consumers are attached to a single topic for resiliency and failure recovery purposes as given in Figure 31.

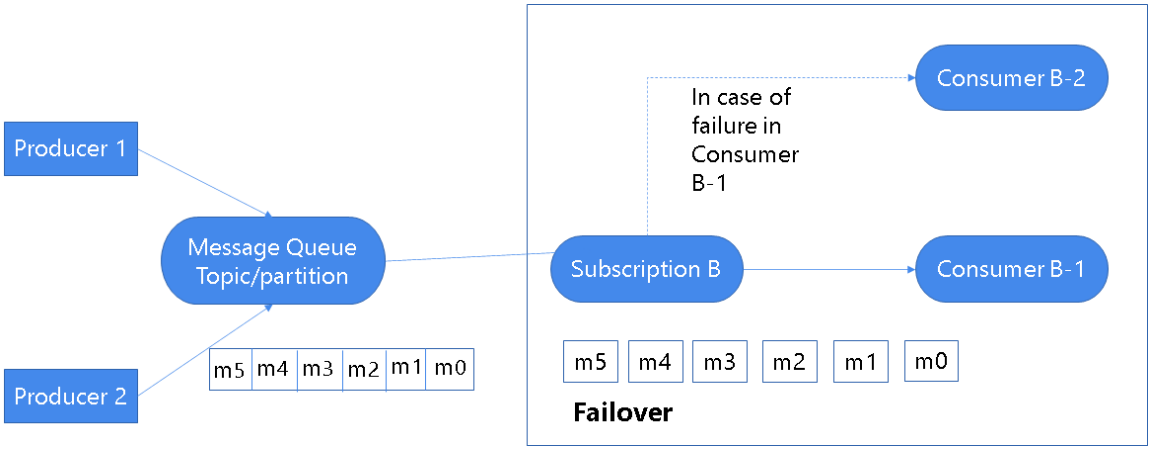


FIGURE 31: FAILOVER SUBSCRIPTION MODE

Note that some existing data aggregation frameworks such as Apache Kafka [22] and Pulsar [23] provide some of the above functionalities. For messaging/streaming applications, exclusive and failover subscription modes are mostly suitable for scenarios when ordering guarantees are needed at the partition level. For queuing applications, shared and key shared subscriptions are used.

Additionally, there are various options to select different types of window operations depending on use case. Table 29summarises them.

TABLE 29: OPTIONS TO SELECT WINDOW OPERATIONS DEPENDING ON USE CASE

Windows	Types	Description
Time-driven Windows	**Tumbling Windows**	There is no overlapping between the windows, e.g. observe strict T seconds intervals
	Sliding Windows	Slide the window with the overlapping time intervals, e.g. monitor activities in the last T seconds.
	Session Windows	Based on the activity, e.g. during user session until pre-defined time-out period where user is inactive
Data-driven Windows		Windows every N elements (count window) and is only based on counts of events and is data independent.

For streaming applications, the domains of time involved need to be understood clearly. As a matter of fact, the Table 29 windowing concepts (tumbling, sliding and session window) are closely tied with the time characteristics which is another important concept for data engineering pipeline and frameworks. In any data processing, the following key concepts in time are important:

- **Event time:** represents the time when event is created at source of the system.
- **Ingestion time:** represents the time when event enters dataflow.
- **Processing time:** represents the time when event was processed/observed in the system.

In an ideal scenario, the gap between event time and processing time should be zero (real-time applications). However, this may not be the case due to several reasons such as network congestion, software logic, etc. Note that depending on the use case, time characteristics can be important (e.g. anomaly detection, predictive maintenance applications).

3.3.2. An analysis of data consumers in project pilots

3.3.2.1. INNOVALIA Pilot: Mapping with streaming semantics and time windowing

In this pilot, there are two use cases.

I. **Connected worker for remote operation of quality equipment**: This use case enables remote operation in real time of the CMM. To this end, several service properties must be monitored such as video quality, delay of the image seen in display and good reliability of the end-to-end connectivity. Two data engineering pipeline related applications are:
   1. **Data collection application**: The 5Gr-VS interacts with the 5G-EVE framework for the Network Service instantiation and for the use of the metric collection platform.
   2. **Visualization applications**: In this application, first, the virtualization of the CMM, to enable the remote control of the machine and the virtual representation of the status of the machine (position of the sensor, etc). Second, a real time visualization of the environment (a video stream of the machine) to show the movement of the sensor and avoid collisions, and a more detailed video visualization of the surface the sensor is scanning (with a light camera attached to the sensor). Third, a 3D virtual display of the machine in real time is done.

II. **Connected worker - Augmented ZDM Decision Support System (DSS)**: In this use case, when a manufactured piece comes out of the production line, the piece code will be identified. Two data engineering pipeline related applications are:
   1. **Handling measurement results**: Test the transmission of the measurement results to the visualization and storage device. CMM is measuring the piece and sending the results to the visualization and storage device. The results are sent in real-time towards the device. They can be visualized in real-time and they are stored for future analysis.
   2. **Monitoring slice creation time**: Several slices will be working with different network performance requirements. It is important to monitor the total time that takes since the user requests a slice until the connectivity is established.

3.3.2.2. COMAU Pilot: Mapping with Streaming Semantics and Time Windowing:

In this pilot, there are three use cases. The main objective in use cases 1 and 3 is to deliver the uninterrupted video streaming to multiple users with high-definition formats.

I. **Digital twin apps**: This use case allows the plant manager to receive live information about the production line through a digital representation of the factory (digital twin of the actual robot is under study). Three data engineering pipeline related applications are:
   1. **Augmented reality application**: It is performing real-time reporting on the robot status with no delays and perfect synchronization between the physical and virtual robots. First, the coordinates of the robot position are periodically sent. Later, positioning data from the robot controller is gathered in real-time (which estimates

the actual robot position via the encoders installed on the axes motor). Finally, the robot behaviour is replicated on the augmented reality device (i.e., in the HoloLens)

2. **Control software of the robot**: This is a software controlling the robot (same computer with AR) for transmitting movement commands
3. **Application in the 5G smartphone (optional)**: Used to drive the robot from remote locations.

II. **Telemetry/monitoring apps**: They provide deeper information, monitoring the status of the equipment installed in the stations (i.e., robots, conveyors, motors, and so on), by installing a high number of sensors that use 5G communication technologies. Data flows from all the connected pieces of machinery and robots to gateway level and from gateway to IoT at TIM premises. Three data engineering pipelines related to applications are:

1. **Failure detection (or predictive maintenance)**: Issues such as vibration patterns affecting the robots could be revealed by monitoring the process in real time. New prediction algorithms for both maintenance and scheduled activities can be applied to the provided data.
2. **Sensor analytics (correlating sensor data)**: To process sensors (for pressure, temperature, vibration (IMU), photocells to recognize piece presence, etc) with long lasting batteries and communicating few data in real time. Correlating data from these sensors would enhance both monitoring and failure prevention.
3. **A better visualisation layer**: Further implementation of more effective predictive analytics procedures are to be explored in future works.

III. **Digital tutorials and remote support**: It enables a high-resolution interface to remotely train and assist the worker on certain tasks, providing high data rates to stream video flow and remote support tool. A data engineering pipeline related application is:

1. **AR via HoloLens**: Augmented Reality Device (HoloLens) is used both to visualize procedures and remote instructions as well as for the high-quality video streaming. End instructions and procedures are sent to the technician (i.e., a digital tutorial) from remote factory to COMAU premises. An application is used to both receive video streaming and send maintenance suggestions and procedures which Leverage the AR capabilities of the HoloLens to superimpose graphical elements to the reality in front of the technician.

3.3.2.3. EFACEC_E Pilot: Mapping with streaming semantics and time windowing:

In this pilot, there are two use cases.

I. **Advanced monitoring and maintenance support:** This use case assists both the remote operator in the control centre, and the crews dispatched to the field to better assess the severity and the impact of the outage they are facing.

1. **Alarm monitoring system:** Low voltage controller received faults from low voltage sensors and sends an alarm to the SCADA/ADMS System in the Control Centre about

the faults. During repairment, augmented reality devices will be used by the maintenance team.

2. **Live video feed:** In the event of an alarm, or an intrusion inside the Secondary Substation, a video surveillance camera will provide a live FHD video streaming to the control centre operator, and also to the mobile device of the maintenance crew.

II. **Advanced critical signal and data exchange across wide smart metering and measurement infrastructures:** This use case is based on advanced monitoring and fault detection in the low voltage grid. The last gasp feature of smart devices in the low voltage grid will help the utility to determine which customers are affected.

1. **Last-gasp event transmission:** Low latency communication is required to ensure that the last-gasp events are successfully transmitted before the low voltage sensors turn off. Zero packet loss and high availability are also mandatory requirements.
2. **Correlation analysis and big data processing:** Low latency communication is required to improve the general quality of the acquired data (i.e., currents, voltages, powers, etc), benefiting advanced control applications dependent on the correlation analysis and big data processing.

3.3.2.4. EFACEC_S Pilot: Mapping with streaming semantics and time windowing:

In this pilot, there are two use cases.

I. **Safety critical communications (train detection):** This use case is on using a 5G network to relay the detection of an incoming train, towards a Level Crossing (LX) controller.

1. **LX controller data collection:** The LX controller needs to collect sensor information to automatically define the position of the half-barriers, the traffic lights and the protection signal, and be prepared against failures in the communication. For instance, in case of communication failures, the level crossing must be set to its safe mode (protection mode).

II. **Non-safety critical communications (video surveillance):** This use case reinforces the safety conditions and considers the video stream of live footage from the LX towards the incoming train and maintenance crews in the area.

1. **Monitoring software applications:** Monitor and control of the LX through various software applications.
2. **Event and alarm reception application:** Reception of the LX events and alarms both at Command Centre (CC) and tablet device (maintenance staff).

3.3.2.5. Required Tools

Based on the above discussion, we propose different monitoring data consumer tools, using the time windowing and streaming semantics options:

- **Latency monitoring tool**: It checks every pre-defined streaming time-windowing interval whether the latency is below the required service requirements.
  - First, determine the streaming time windowing option, e.g., tumbling, sliding, session.

  - Second, collect the latency measurements over the time windowing period. Collecting latency measurements is critical for AR application. This requires exactly once streaming semantics
  - Third, calculate the average latency per second over the collected measurements.
  - Fourth, compare the calculated value with the fixed latency requirements of service (5-7 ms for digital twins and telemetry use cases)
  - Finally, notify the infrastructure owners if discrepancy occurs
- **Throughput monitoring tool**: It checks every pre-defined window stream interval whether the throughput is below the required service requirements.
  - First, determine the streaming time windowing option, e.g., tumbling, sliding or session-based windows.
  - Second, collect the throughput measurements over the time windowing period. Collecting throughput measurements is non-critical -> Requires at least once streaming semantics
  - Third, calculate the average throughput per second over the collected measurements.
  - Fourth, compare the calculated value with the fixed throughput requirements of service (1 Gbps for digital twins and 100 Mbps telemetry use cases)
  - Finally, notify the infrastructure owners if discrepancy occurs
- **Availability monitoring tool**: It checks every pre-defined window stream interval whether the availability is below the required service requirements.
  - First, determine the streaming data windowing option, e.g., observe number of times the service is unavailable.
  - Second, collect the non-availability times over the data windowing period. Collecting availability measurements is critical -> Requires exactly once streaming semantics
  - Third, calculate the non-availability times over the collected measurements.
  - Fourth, compare the calculated value with the fixed service requirements or 1 out of 1000 times unavailability, (five 9s for digital twins, telemetry and digital tutor)
  - Finally, notify the infrastructure owners if discrepancy occurs
- **Slice connectivity time monitoring tool**: It checks every pre-defined window stream interval whether the slice connectivity time (both creation and removal of slices) is below the required service requirements.
  - First, determine the streaming time windowing option, e.g., tumbling, sliding, session-based windows.
  - Second, collect the slice connectivity time measurements over the time windowing period. Collecting slice connectivity measurements is critical. This requires exactly once streaming semantics
  - Third, calculate the average slice connectivity time over the collected measurements.
  - Fourth, compare the calculated value with the fixed slice connectivity requirements of service (less than 5 min for both slice connection and removal in connected worker augmented ZDM DSS (INNOVALIA) use case)
  - Finally, notify the infrastructure owners if discrepancy occurs
- **Service Operation Time Monitoring Tool**: Checks every pre-defined window stream interval whether the service operation time (the total time it takes for the CMM to be ready to measure

the piece since the code is read at the production line in piece identification experiment of connected worker augmented ZDM DSS (INNOVALIA) use case,) is below the required service requirements.
  - First, determine the streaming time windowing option, e.g. tumbling, sliding, session-based windows.
  - Second, collect the service operation time measurements over the time windowing period. Collecting service operation measurements is non-critical -> Requires at least once streaming semantics
  - Third, calculate the average service operation time over the collected measurements.
  - Fourth, compare the calculated value with the fixed service operation time requirements of service (120 seconds for in connected worker augmented ZDM DSS (INNOVALIA) use case)
  - Finally, notify the infrastructure owners if discrepancy occurs.
- **Network Slice Deployment Time Monitoring Tool:** Checks every pre-defined window stream interval whether the network slice deployment time (the time between the service request and the service availability when SONATA orchestrator is used to instantiate the network service in safety critical communications (EFACEC_S) use case) is below the required service requirements.
  - First, determine the streaming time windowing option, e.g. tumbling, sliding, session-based windows.
  - Second, collect the network slice deployment time measurements over the time windowing period. Collecting service operation measurements is non-critical -> Requires at least once streaming semantics
  - Third, calculate the average network slice deployment time over the collected measurements.
  - Fourth, compare the calculated value with the fixed network slice deployment time requirements of service (under 60 seconds for safety critical communications (EFACEC_S) use case)
  - Finally, notify the infrastructure owners if discrepancy occurs

3.4. Metadata

In any complex enough managed entity, the number of potential monitoring data sources and their characteristics becomes extremely high. Network infrastructures constitute a paradigmatic example of these complex entities, especially if a full end-to-end (E2E) control is intended for the infrastructure and network services running on it. To address this problem, data aggregation mechanisms are an essential component providing a consistent and manageable set of information for further processing.

Different data sources provide different means to access their data. For instance, while some push methods, so aggregation can occur by receiving data passively from the data source, in other cases a polling mechanism requiring explicit requests at a continuous pace of requests is applied, while other requires to actively collect data from a database which is shared concurrently by other processes. A similar heterogeneity needs to be considered when dealing with data transport, with

data sources applying a great variety of access control, confidentiality and integrity methods. In addition, in many cases timing constraints on both the data and their processing need to be considered. The validity of data depends on the time when they are collected and accessed, and the correctness of a process depends on whether it completes on time.

In conclusion, data received from the network may come in many different formats, from unstructured to strictly based on standard models, being transported according to a great variety of mechanisms, accessed using diverse credentials and be available at different time frames. This heterogeneity increases the difficulty in designing suitable solutions openly applicable and reusable in different scenarios. Such an open integration of data sources and consumers of different nature requires a data semantic framework able to adapt data in an efficient way, suitable to be provided to the data consumers in 5GR architecture. This approach allows flexible usage of produced data and it enables organizations run different types of analytics from dashboards and visualizations to big data processing, real-time analytics, and machine learning.

In this context of metadata definition, it appears the ETSI ISG CIM (cross-cutting Context Information Management) standards. CIM considers the exchange of data and metadata across systems is a crucial enabler for different applications, allowing these applications to better collect information from different origins, filter information from "data lakes" and to create derivative information or decisions. CIM considers provisioning of provenance, data quality, access control and other features as part of its NGSI-LD protocol [24]. The NGSI-LD API uses linked open data and property graphs to reference data definitions (ontologies).

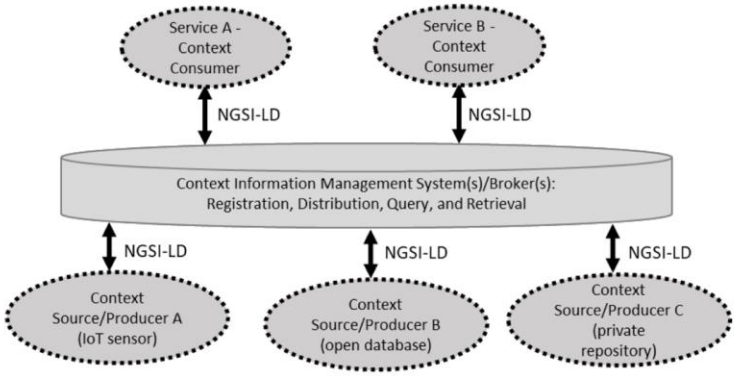


FIGURE 32: INTERCONNECTION OF CONTEXT INFORMATION

Context information is exchanged among applications, context producers, and context brokers. Although the NGSI-LD API is intended to be primarily an API and CIM does not define a specific architecture, Figure 33 depicts a prototypical architecture for implementing a CIM-based architecture.

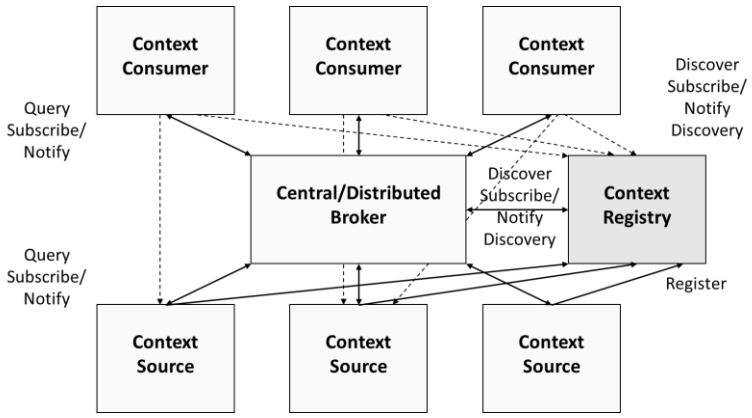


FIGURE 33: CIM REFERENCE ARCHITECTURE

The main components that are shared across the different CIM-based architectures are the ones shown in Figure 33. There are Context Producers that use operations to update the context information in the Broker and there are Context Consumers that request context information from the Broker, either using synchronous one-time query or asynchronous subscribe/notify operations. The main difference appears when the Broker implementation takes places. Depending on the selected choice (either central or distributed), this component has a different role. In the central architecture, the Broker stores all the context information and answers all the requests from its storage. In the distributed architecture, the Context Sources register themselves with the Context Registry, not the Context Broker, providing information about what context information they can provide. The Distribution Broker then queries or subscribes to each relevant Context Source, if possible, it aggregates the context information retrieved from the Context Sources and provides them to the Context Consumer.

The CIM approach is based on the information model that is shown in Figure 34. Context information is considered to be any relevant information about entities, their properties, and their relationships with other entities. Entities may be representations of real-world objects but may also be more abstract notions such as a legal entity, a network function, or a group of other entities. Relationships between entities can be modelled as having specific properties, and the whole set of entities and relationships are modelled according to a labelled property graph, providing a theoretical foundation for automated reasoning about the characteristics of the systems they represent.

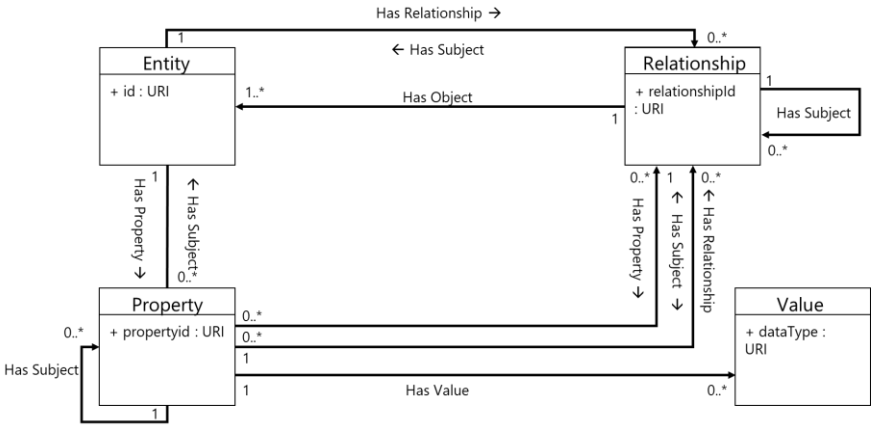


FIGURE 34: CIM INFORMATION MODEL

The application of the CIM framework has been focused on IoT applications so far but, given its support for binding context sources and consumers, we are incorporating network monitoring data as another class of context to de addressed within it, relying primarily on YANG models, though other modelling mechanisms (e.g., SNMP MIBs, time series databases like Prometheus, IPFIX data flows, etc.) are being considered as well. The following figures illustrate the information models for the different classes of metadata being considered at this stage, together with examples of their description using NGSI-LD.

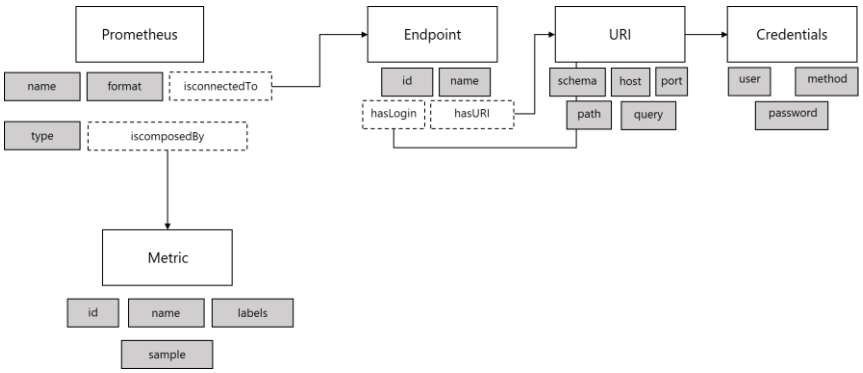


FIGURE 35: INFORMATION MODEL FOR PROMETHEUS-BASED DATA SOURCES

A sample Prometheus-based data source would look like this:

```
{
    "id": "urn:ngsi-ld:Prometheus:1",
    "type": "Prometheus",
    "name": {
        "type": "Property",
        "value": "admin"
    },
    "prometheusType": {
        "type": "Property",
        "value": "PrometheusRouter"
    },
    "format": {
        "type": "Property",
        "value": "JSON"
    },
```

```
    "isComposedBy": {
        "type": "Relationship",
        "object": "urn:ngsi-ld:Metric:1"
    },
    "isConnectedTo": {
        "type": "Relationship",
        "value": "urn:ngsi-ld:Endpoint:1"
    },
    "@context": [
        {
            "Prometheus": "http://example.org/Prometheus",
            "prometheusType": "http://example.org/prometheusType",
            "format": "http://example.org/format"
        },
        "https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld"
    ]
}
{
    "id": "urn:ngsi-ld:Metric:1",
    "type": "Metric",
    "name": {
        "type": "Property",
        "value": "node_cpu_seconds_total"
    },
    "labels": {
        "type": "Property",
        "value": {
            "cpu": "0",
            "mode": "guest"
        }
    },
    "sample": {
        "type": "Property",
        "value": "0"
    },
    "@context": [
        {
            "Metric": "http://example.org/Metric",
            "labels": "http://example.org/labels",
            "sample": "http://example.org/sample"
        },
        "https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld"
    ]
}
. . .
```

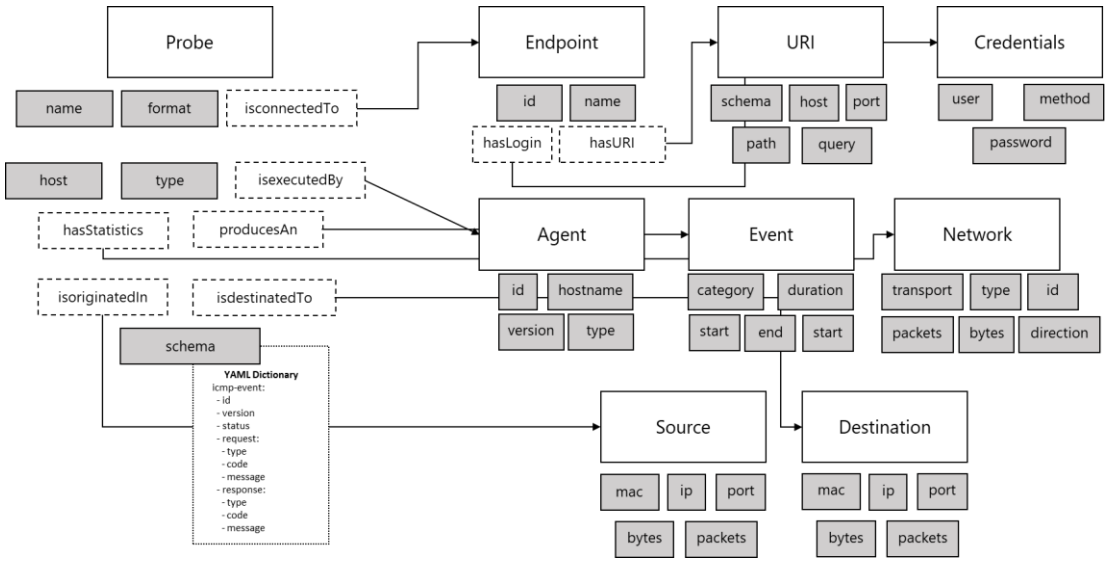


FIGURE 36: INFORMATION MODEL FOR JSON-BASED PROBES

A sample JSON-based data source would look like this:

```
{
    "id": "urn:ngsi-ld:Probe:1",
    "type": "Probe",
    "name": {
        "type": "Property",
        "value": "pinger-1"
    },
    "format": {
        "type": "Property",
        "value": "json"
    },
    "isConnectedTo": {
        "type": "Relationship",
        "object": "urn:ngsi-ld:Endpoint:1"
    },
    "host": {
        "type": "Property",
        "value": "ovs1"
    },
    "probeType": {
        "type": "Property",
        "value": "icmp"
    },
    "isExecutedBy": {
        "type": "Relationship",
        "object": "urn:ngsi-ld:Agent:1"
    },
    "hasStatistics": {
        "type": "Relationship",
        "object": "urn:ngsi-ld:Network:1"
    },
    "producesAn": {
        "type": "Relationship",
        "object": "urn:ngsi-ld:Event:1"
    },
    "isOriginatedIn": {
        "type": "Relationship",
        "object": "urn:ngsi-ld:Source:1"
    },
    "isDestinatedTo": {
        "type": "Relationship",
```

```
        "object": "urn:ngsi-ld:Destination:1"
    },
    "schema": {
        "type": "Property",
        "value": ""
    },
    "@context": [
        {
            "Probe": "http://example.org/Probe",
            "format": "http://example.org/format",
            "isConnectedTo": "http://example.org/isConnectedTo",
            "host": "http://example.org/host",
            "probeType": "http://example.org/probeType",
            "isExecutedBy": "http://example.org/isExecutedBy",
            "hasStatistics": "http://example.org/hasStatistics",
            "producesAn": "http://example.org/producesAn",
            "isOriginatedIn": "http://example.org/isOriginatedIn",
            "isDestinatedTo": "http://example.org/isDestinatedTo",
            "schema": "http://example.org/schema"
        },
        "https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld"
    ]
}
. . .
```

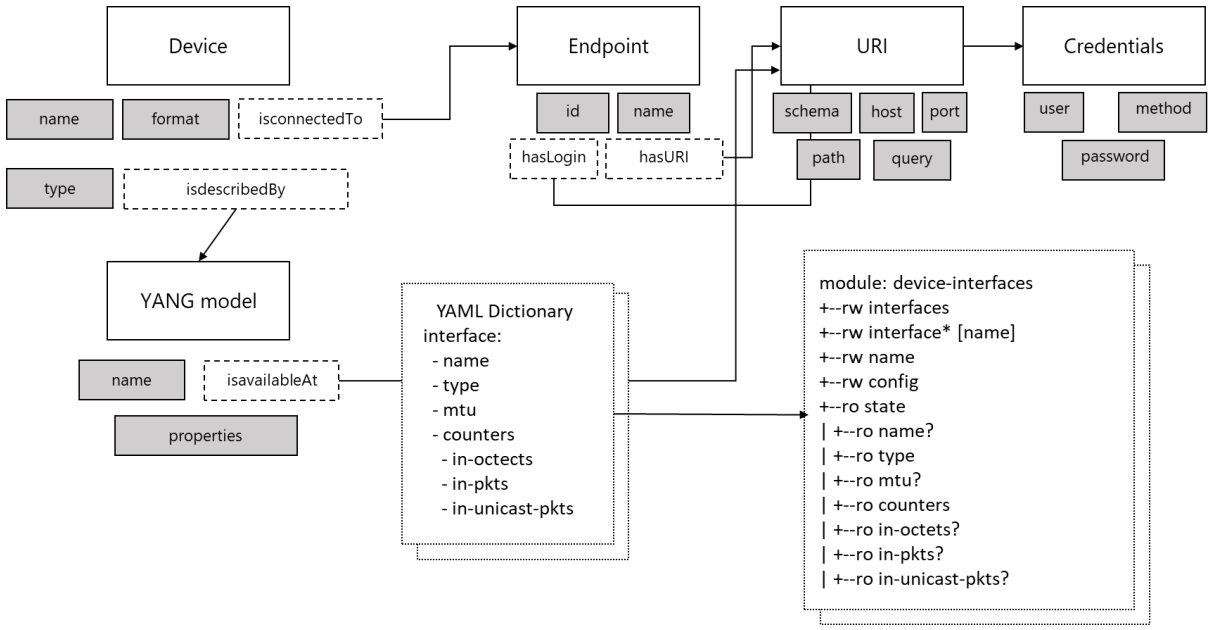


FIGURE 37: INFORMATION MODEL FOR YANG-BASED DATA SOURCES

And a sample YANG-based data source would look like this:

```
{
    "id": "urn:ngsi-ld:Device:1",
    "type": "Device",
    "name": {
        "type": "Property",
        "value": "Cisco ASR 1001-X"
    },
    "deviceType": {
        "type": "Property",
        "value": "Router"
    },
    "format": {
        "type": "Property",
```

```
        "value": "JSON"
    },
    "YANGModel": {
        "type": "Property",
        "value":
"https://raw.githubusercontent.com/YangModels/yang/c719b4547b46545d4926b3134e84226606a98a
aa/standard/ietf/RFC/ietf-interfaces@2018-02-20.yang"
    },
    "isConnectedTo": {
        "type": "Relationship",
        "value": "urn:ngsi-ld:Endpoint:1"
    },
    "@context": [
        {
            "Device": "http://example.org/Device",
            "deviceType": "http://example.org/deviceType",
            "format": "http://example.org/format",
            "YANGModel": "http://example.org/YANGModel",
            "isConnectedTo": "http://example.org/isConnectedTo"
        },
        "https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld"
    ]
}
{
    "id": "urn:ngsi-ld:Endpoint:1",
    "type": "Endpoint",
    "name": {
        "type": "Property",
        "value": "yang-admin"
    },
    "hasLogin": {
        "type": "Relationship",
        "object": "urn:ngsi-ld:Credentials:1"
    },
    "hasURI": {
        "type": "Relationship",
        "value": "urn:ngsi-ld:URI:1"
    },
    "@context": [
        {
            "Endpoint": "http://example.org/Endpoint",
            "hasLogin": "http://example.org/hasLogin",
            "hasURI": "http://example.org/hasURI"
        },
        "https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld"
    ]
}
. . .
{
    "id": "urn:ngsi-ld:Credentials:1",
    "type": "Credentials",
    "user": {
        "type": "Property",
        "value": "admin"
    },
    "method": {
        "type": "Property",
        "value": "Basic Auth"
    },
    "password": {
        "type": "Property",
        "value": "YWRtaW46YWRtaW4="
    },
    "@context": [
        {
            "Credentials": "http://example.org/Credentials",
```

```
            "user": "http://example.org/user",
            "method": "http://example.org/method",
            "password": "http://example.org/password"
        },
        "https://uri.etsi.org/ngsi-ld/v1/ngsi-ld-core-context.jsonld"
    ]
}
. . .
```

4. Integration with the 5Growth architecture

The work in WP4 is aligned with the activities conducted in both WP2 and WP3, addressing three innovate/deploy/validate cycles during the project lifetime. WP4 applies the appropriate releases produced by WP3, that incorporate the modules produced in WP2, and selects the applicable experimental environment for the corresponding cycle.

This section discusses the mechanisms required to integrate the measurement methodology and the data infrastructure described above within the general 5Growth architecture for the cycle reported in this document. We analyze the interplay of the metadata definitions and data infrastructure descriptors, the integration with the 5Growth Monitoring Platform, the interaction with the ICT-17 platforms on which experiments will run, and finally how the data infrastructure has to be applied for the different innovations explored within WP2.

4.1. Experiment descriptors

As described in Section 2.4, the 5Growth platform provides a framework to validate vertical SKPIs and CKPIs basing the experiments in the usage of the 5Growth experiment catalogue. Therefore, it is needed to provide a consistent way of defining these experiments across the different vertical use cases. Given the complexity of handling the different vertical requirements in an NFV-based platform, it is also convenient to support the appropriate abstractions to provide usability.
This experiment stage is carried out by means of a model-based approach in which vertical users have an experiment descriptor available, defining the concrete parameters that are going to be verified. In order to carry out the actions and operations described in the experiment descriptor; it is needed to make an internal translation of it to be spread across the needed components in the 5Growth architecture. As it was previously defined in Section 2.4 and Section 3.4, 5Growth adds to its architecture the metadata concept to achieve the required level of abstraction in the definition of the different components that take part in the experiment stage.
In terms of implementation, this experiment descriptor includes both the service descriptor(s) associated to the experiment and also the different measurements that need to be performed for the validation. Thus, by making use of this descriptor, the system references the different data sources (i.e., tools, devices, ICT-17 platforms, etc) that can provide the required measurements. In general, this experiment descriptor will be consumed by the 5Growth data aggregator that exposes its capabilities through an API. Then, this data aggregator will handle the different connections needed with the rest of the 5Growth architecture components. As these experiments are also going to make use of some resources available in ICT-17 infrastructures, the experiment descriptor should take this integration into consideration. Considering the different methods for integrate 5Growth with ICT-17 platforms defined on D3.2 Section 3, the experiment descriptors will be adapted depending the scenario as follows:

- Regarding 5G-EVE integration there are two different options. In option 2 (defined in D3.2 Section 3.3), the experiment descriptor will be part of the 5G-EVE blueprint.
- Regarding 5G-VINNI, 5Growth data aggregator will received manually the experiment descriptor though its API and then this component should be able to interact with the Vertical API to get the necessary information.

4.2. Monitoring Platform

4.2.1. Monitoring platform overview

5Growth Monitoring platform is a logging, monitoring, and alerting system, integrated into the 5Growth infrastructure. It provides a collection of metrics and logs from VNFs, analyzing and tracking of parameters and thresholds. 5Growth Monitoring platform is based on the 5G-TRANSFORMER Monitoring Platform with some extensions to its functionality. This ends up with a more advanced architecture including a centralized messages queue, enhanced probes model (that allows gathering server or client-centric metrics), enhanced configuration capabilities (supporting client-based monitoring rules), and the possibility to act as a trigger for the scaling process. The Monitoring platform becomes as the data source for other systems such as the AI/ML platform. A more detailed description of the 5G-TRANSFORMER architecture can be found in the 5G-TRANSFORMER D4.3 [17].

The 5Growth Monitoring system includes the following elements:

- The **Prometheus server** is a platform that collects metrics from monitored targets. Prometheus data is stored in the form of metrics, where each metric has a name that is used for referencing and querying it. Prometheus stores data locally on disk, which helps for fast data storage and fast querying. Prometheus has PromQL - the query language used to create dashboards and alerts.
- **Grafana** provides graphs and visualizations of metrics. Grafana is multi-platform open-source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources. It is expandable through a plug-in system. End users can create complex monitoring dashboards using interactive query builders.
- **AlertManager** processes Prometheus alerts. AlertManager can include logic to silence alerts and to forward them to email, Slack or other notification services.
- **Prometheus Pushgateway** and **Prometheus MQ Agent** are used to provide Prometheus data from MQ Kafka.
- An **HTTP server** holds RVM agent archives and initial script for virtual machines.
- **Logstash** is the data collection pipeline tool. It collects data inputs and feeds them into Elasticsearch. It aggregates all types of data from different sources and makes it available for further use.
- **Elasticsearch** allows storing, searching, and analyzing big volumes of data. It is mostly used in these applications as the underlying engine for implementing search task functionality. It has been adopted in search engine platforms for modern web and mobile applications. Apart from a quick search, the tool also offers complex analytics and many advanced features.
- **Kibana** is used for visualizing Elasticsearch documents and helps developers to have a quick insight into it. Kibana dashboard offers various interactive diagrams, geospatial data, and graphs to visualize complex queries. It can be used for search, view, and interact with data stored in Elasticsearch directories. Kibana helps to perform advanced data analysis and visualize the data in a variety of tables, charts, and maps.

- **Kafka MQ** is a bus for metrics and control information for RVM agents.
- **RVM agent** is an agent that is installed on VM during VM creation through cloud-init. It provides the next features:
    - Bash script code execution and providing the execution result.
    - Keepalive mechanism.
    - RVM agent Prometheus Collectors management (create/delete).

    All these functions are managed by Config Manager
- **Config Manager** is a lightweight REST adapter exposing a single, simplified REST API for the configuration of the whole monitoring system, acting as the "management" NBI of the Monitoring Platform.

The following figure shows the vertical service monitoring architecture. The Config Manager is responsible for translating the platform-independent requests for monitoring-related management actions into requests directed to Prometheus, Grafana, AlertManage, Kibana, ElasticSearch, LogStash. Config Manager also generates scripts for RVM agent installation.

Elasticsearch offers the possibility to collect logs from any application and configure how these logs are parsed, allowing the storage and analysis of the logs to infer specific metrics and KPIs, such as availability of applications and servers, setup times, fault detection, etc. This functionality is not allowed by the Prometheus platform.

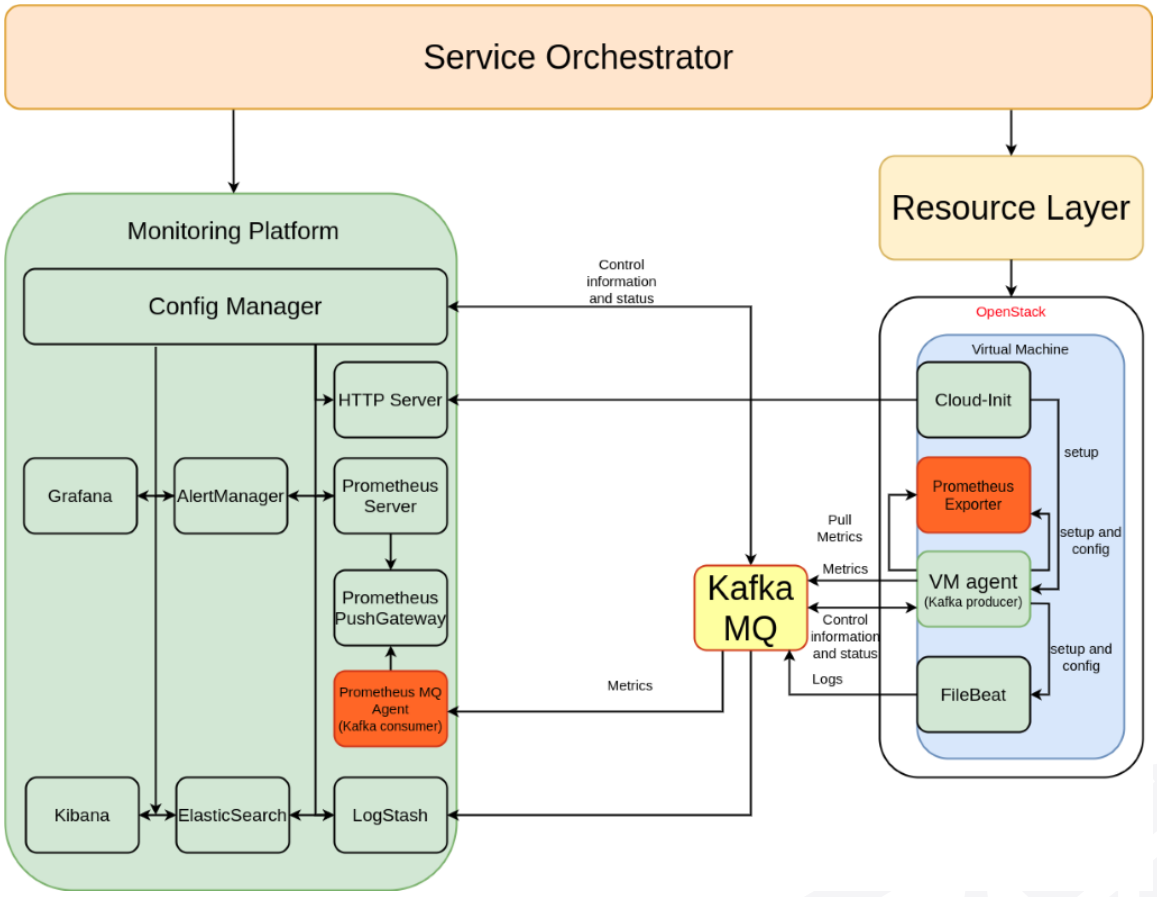


FIGURE 38: VERTICAL SERVICE MONITORING ARCHITECTURE

The procedure of how 5Gr-SO creates VM and manages the RVM agent looks like the following. 5GR-SO analyzes NSD and calculates how many and where VNFs should be created. Figure 2 shows the RVM agent installation workflow.

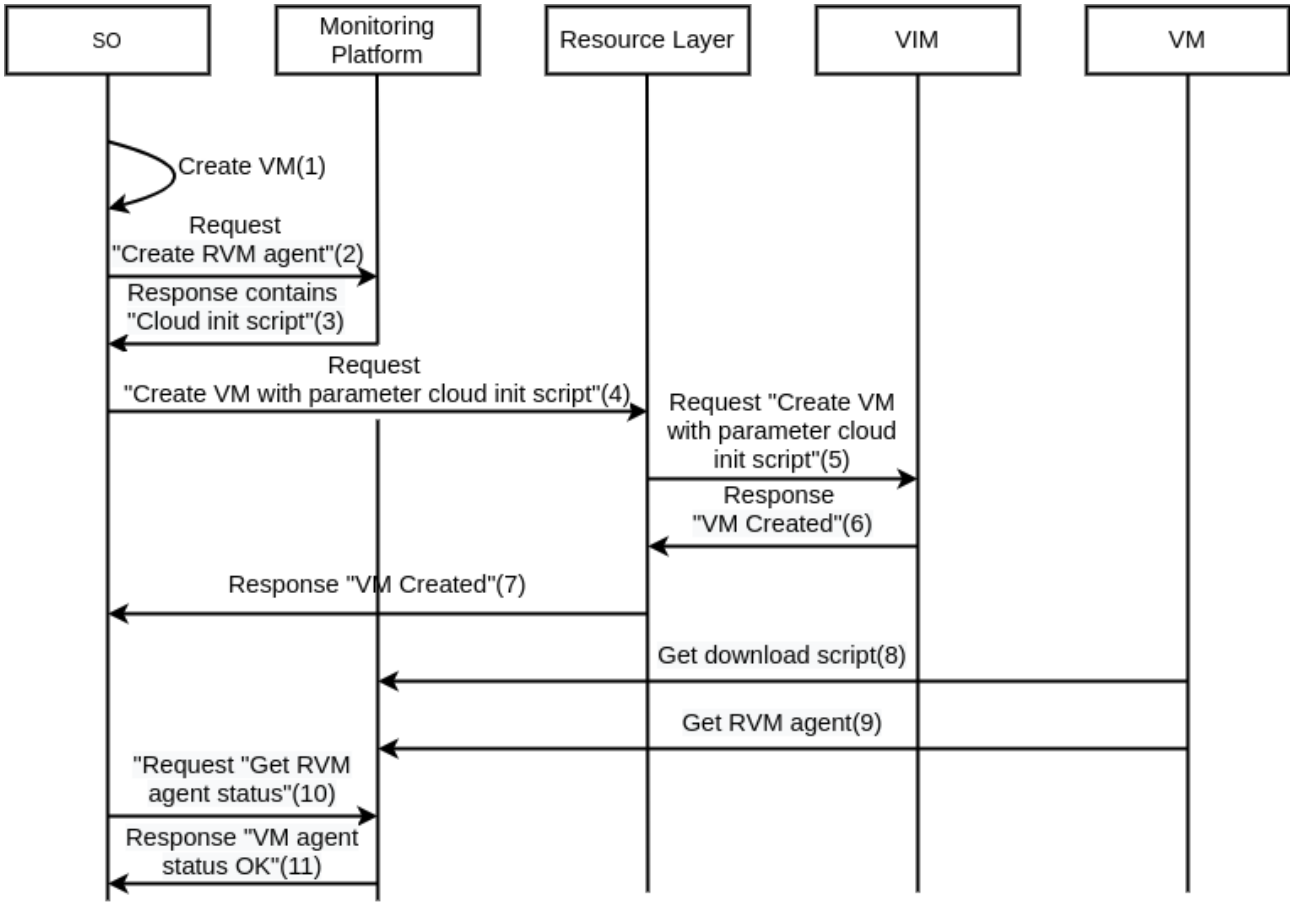


FIGURE 39: AN RVM AGENT INSTALLATION WORKFLOW

Workflow:

1. The 5Gr-SO takes the task "Create VM".
2. The 5Gr-SO makes a request "Create RVM agent" to the monitoring platform.
3. The monitoring platform returns the response to the 5Gr-SO with a cloud-init script.
4. The 5Gr-SO makes a request "Create VM" with parameter cloud-init to the Resource Layer.
5. The Resource Layer sends the request to VIM (Virtual Infrastructure Manager)
6. The VIM creates a VM and returns a response "VM created" to the Resource Layer
7. Resource Layer transmits the response to the 5Gr-SO "VM created".
8. When Virtual Machine starts, it loads the script.
9. The script detects the type of the operating system, downloads, and starts the necessary an RVM agent.
10. The 5Gr-SO repeats the request "Get VM agent status" to the Monitoring Platform.
11. The 5Gr-SO repeats the previous request until it receives the response "VM agent status OK" from the monitoring platform.

This time interval includes the following processes: get VM created, get the Operation system started, and get VM agent started until it sends keepalive messages to the Monitoring Platform.

4.2.2. Monitoring data exposure

The Monitoring Platform (MP) can be a source of information for an AI/ML platform or forecasting module. MP provides monitoring information for the following algorithms: auto-scaling, self-healing, automated traffic management, forecasting, inference, anomaly detection, control-loops stability. The MP should have the flexibility to provide data for different analyzing and other monitoring systems. There are mechanisms to exchange the monitoring information between domains. The monitoring system consists of two subsystems: Prometheus stack which collects metrics and Elastic stack which collects logs.

The Prometheus stack of the MP has many ways to share monitoring information with other MP and systems:

- Prometheus has an API for federation. Other monitoring domains can receive all metrics from the Prometheus database.
- Prometheus API supports Prometheus Query Language. It gives the possibility to other systems to get information for some specific metric and a specific time interval.
- There is the use case present when data in the specific data format should be exposed to the Kafka topic. Other systems (for example AI/ML) can get this information in real-time from the Kafka topic. Data Scraper was developed and integrated into the monitoring platform for this purpose. This object is configured through MP API. Data Scraper collects the last specific metrics from Prometheus and puts these metrics to defined Kafka's topic. Other systems (for example AI/ML) can get this information in real-time from the Kafka topic for analyzing.

The Elastic stack has many ways of exchanging information between other stacks. Elastic provides the ability to search across clusters by using Cross-cluster search (CCS) and replicate data across clusters by taking advantage of Cross-cluster Replication (CCR) and expose a Search API to query in the same way users query through the UI. All Elastic stack's suitable methods with 5Growth for data exposure are described below:

- Cross-cluster search lets you run a single search request against one or more remote clusters. Cross-cluster search filters and analyzes log data stored on clusters in different data centers. The Elastic node that receives a request from an operator is named the coordinating node. The coordinating node in the cluster is the main interface with systems requesting searches, it receives and parses the requests. Moreover, the coordinating node will forward each of the search requests to each of the remote clusters, including the local cluster. Each cluster performs the requested search independently, applying its own cluster-level settings to the request. Furthermore, each cluster sends its search results back to the coordinating node. Finally, after collecting results from each cluster, the coordinating node returns the results from the cross-cluster search as a response to the search request.

- Cross-cluster replication has several multi-cluster architectures; it was identified that is reasonable to use Centralized reporting architecture for a multidomain monitoring platform. The centralized reporting architecture has the advantage of minimizing the network traffic and latency in querying multiple geo-distributed Elasticsearch clusters, while preventing search loads from interfering with the indexing loads by offloading the search loads to a secondary cluster. Using a centralized reporting cluster is useful when it is not feasible to query across the network, either due to increased costs or latency. In this configuration, the cluster will replicate data from many smaller clusters to the centralized reporting cluster. One of the main drawbacks of using cross-cluster replication is that it requires a Platinum subscription for the Elasticsearch platform.
- Elasticsearch exposes the interfaces that are used by its UI components through a REST API which can be called directly to configure and access Elasticsearch features. The REST API is named Search API and allows to execute a search query and retrieve the search hits that match a specific query. The Elasticsearch API could be a technical enabler to exchange data with other systems from the Elastic stack.

4.2.3. The mapping between the Monitoring Platform and KPIs

KPI methodology gives the possibility to evaluate service requirements and understand whether the system provides all the required features. A set of KPIs were defined for 5Growth and described in the table 1 of D4.1. Each KPI has a description of a monitored parameter as shown in the table. The Monitoring Platform can provide measuring core KPIs. All other KPI types are based on Core KPIs. Each KPI should have a defined threshold for controlling the QoS. 5Gr-SO creates the VNFs and configures them according to the NSD. The MP is also configured by the 5Gr-SO. To do that, the 5Gr-SO makes a request to the MP for creating the monitored parameters. It also configures the thresholds for targeted metrics and defines the corresponding action if any of these metrics comes out of range. Each of the KPIs and how they can be measured will be discussed below.

CKPI-1 End-to-end Latency

Latency as key network parameter has a huge influence on both services and customer applications. The End-to-end Latency is a quite complex parameter to be measured. Usually, it requires a source of etalon time (NTP server) for measurement one-way time delay between two-point. Also, it requires installed specific software tools on both points. An End-to-end Unidirectional Link Latency Evaluator is presented above in section End-to-end Unidirectional Link Latency Evaluator2.3.1. This mechanism provides the measurement of unidirectional link latency and packet loss evaluation in distributed systems. This algorithm was used to deploy the Prometheus exporter with the aim to provide information about latency, jitter, and packet loss for a unidirectional link. The Prometheus exporter in conjunction with the aforementioned algorithm were integrated into the monitoring platform. The parameters for the Prometheus exporter were defined in NSD by the 5Gr-SO. To do that, the 5Gr-SO installs and configures the Prometheus exporters on both points by using the Monitoring platform.

The End-to-end Unidirectional Link Latency Evaluator can provide KPIs measurements such as **CKPI-2 Packet Loss** and **CKPI-9 Jitter**.

CKPI-4 Coverage and **CKPI-7 Connection Density** are measured within the RAN segment, which is required to provide such information to the Monitoring system.

CKPI-5 Availability is the percentage of time during which a specific component (e.g., application, server, network function, etc.) is responding to the requests received with the expected QoS requirements. That is, it is the ratio between the up time of a specific component over the total time since the component was deployed. There are two ways to evaluate this KPI.

The first way is from the client side. The idea of a client probe usage is shown in Figure 3. Some deployment with Virtual Server is installed in the Cloud. The Virtual server has a Prometheus exporter for collecting customers'/clients' metrics. This exporter receives metrics from clients and saves them in memory. The Monitoring Platform grabs the collected metrics from the exporter by an explicit request which are afterwards processed.

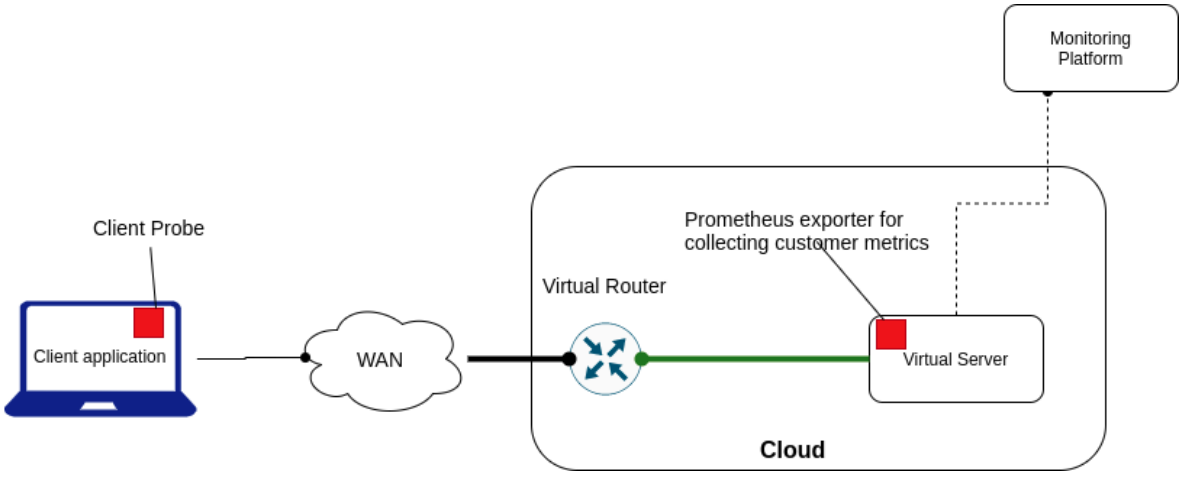


FIGURE 40: CLIENT-SIDE PROBE SCHEMA

The client probe can work in two modes: as a third-party client-side monitoring and as native client-side monitoring.

In the third-party client-side monitoring mode the client probe sends a request to the Prometheus exporter for collecting customers metrics. It receives a reply and calculates the latency time. Then it sends a request to the Prometheus exporter with the information about the latency time. The current latency measuring method shows the indirect quality of the client-side service. It gives the possibility to measure the quality of a network connection. Client probe can send information about the client hardware: CPU usage, memory usage, SNR level, etc. The amount and type of information depends on the software and hardware platform where the client application is implemented. This mode has the disadvantage that it uses extra network bandwidth.

Native client-side monitoring mode is the mode where the client probe gets metrics from a client application such as latency, speed of data loading, buffer usage, data processing time on the client side, etc. The request latency, in this case, shows the true QoS. The speed of data loading shows the quality of service for the data channel. Data processing time on the client-side shows the performance of client hardware if a client has enough performance to use the service.

The client application should be capable of client probe to pass application metrics. The client probe sends collected metrics to the Prometheus exporter for collecting customer metrics. The monitoring platform receives metrics from the Prometheus exporter as a reply on request. This mode does not use extra network bandwidth.

Bellow you can see example of the application metrics of a service:

latency_ms{client_host="192.168.122.1",session_id="1839"} 66.4

loading_ms{client_host="192.168.122.1",session_id="1839"} 146.4

parsing_ms{client_host="192.168.122.1",session_id="1839"} 49.9

bw_kbps{client_host="192.168.122.1",session_id="1839"} 30805.8

Where:

client_host is the IP address of the client who uses a service.

session_id is an identifier of a session.

latency is a time interval between the moment client's application requests the data and when it receives the first packet of the data.

loading is a time interval between the moment when client's application receives the first packet of the data and when it receives the last packet of the data.

parsing is a time interval during which client's application processes the data.

bw_kbps shows the bandwidth used when downloading the data, unit of measurement is Kbps.

The time diagram of data processing is shown below

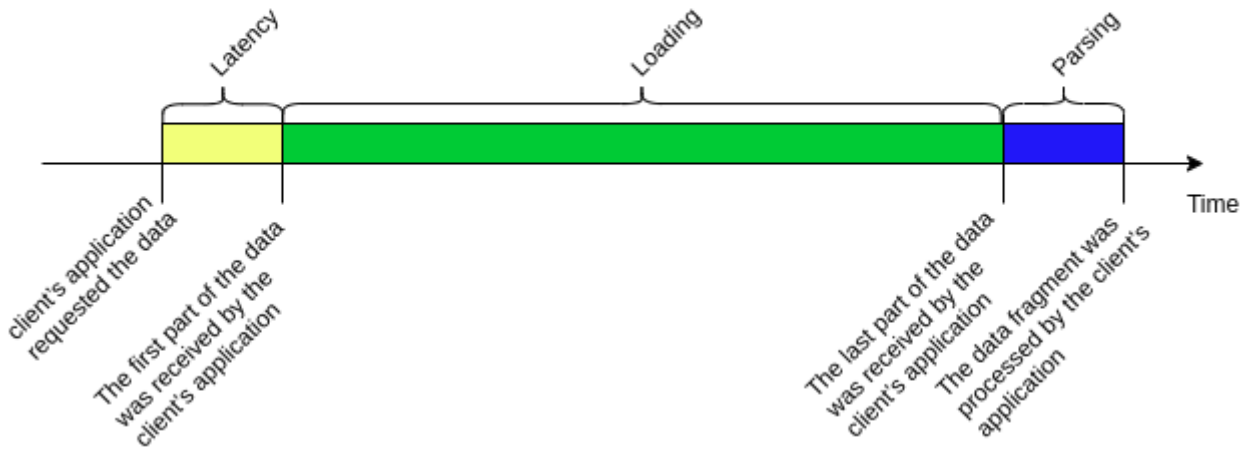


FIGURE 41: TIME DIAGRAM OF DATA PROCESSING

The second way is controlling this KPI from Server Side. The Server-side probe is a dedicated server which does requests to the virtual server as a client and reports to the monitoring platform about service status. The Server-side probe gives the possibility to evaluate the quality of the service without network channel influence. Metrics from this server is collected by monitoring platform. The server-side probe is shown in figure 5. Service-side has the same functions and modes as the client-side probe.

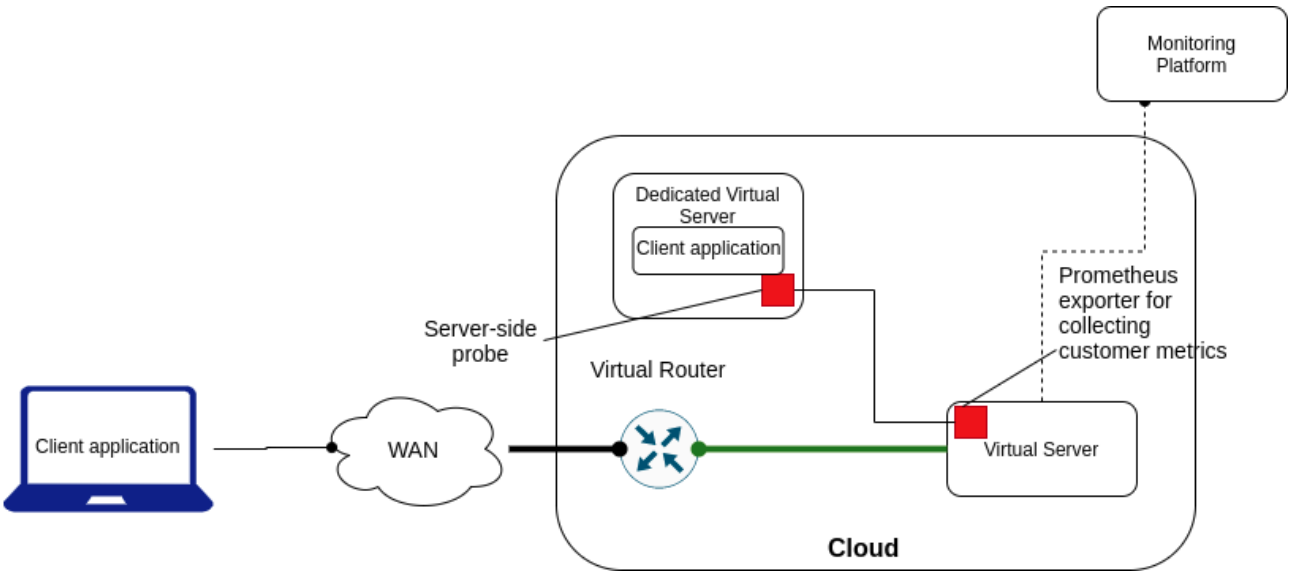


FIGURE 42: SERVER-SIDE PROBE SCHEMA

CKPI-6 Slice Creation/adaption Time. This KPI can be measured by using 5 Growth stack logs. Each component of the 5Growth stack generates logs during his work. Filebeat can be installed on each component of 5Growth. Filebeat is a component of the ELK stack that collects logs and pushes it to Elasticsearch where it can be processed and can be shown in Kibana's Dashboard. For Example, ELK stack can collect 5Growth-SO logs and calculate the time between two-time moments. The first-time moment is 5Growth-SO received request to deploy service and the second-time moment 5Growth-SO finished service deployment and reported about it.

CKPI-7 Connection Density The description of this KPI is the number of users/devices that can be connected simultaneously to the use case network infrastructure without degrading the performance of the users/devices that are already connected. This KPI should be get from the RAN domain since this network segment has information about the current and total radio resource usage.

CKPI-8 Data Volume is the total quantity of information transferred over a given interface during specific use case operations, measured in bits. The information transferred over a given interface can be collected by already integrated tools to the monitoring platform.

CKPI-10 Received Radio Signal Quality and **CKPI-11 Buffer occupancy** should be received from the RAN domain.

4.3. ICT-17 interaction

The initial work developed within WP3 with respect to the integration of 5Growth platform and ICT-17 platforms (i.e., 5G-EVE and 5G-VINNI) is focusing on identifying the multi-domain interactions between the different platforms for supporting the 5Growth vertical pilots and use cases. On its initial stage, this interaction among the different platforms is targeting high-level service lifecycle management operations (such as the creation, instantiation, update, and termination), disregarding any interaction for exchanging monitoring data.

Nevertheless, requesting and retrieving monitoring information from the ICT-17 platforms is still possible through manual processes. Figure 43 depicts the high-level workflow on how to request and retrieve data from both 5G-EVE and 5G-VINNI platforms. Details about these two data sources can be found in Section 3.1.

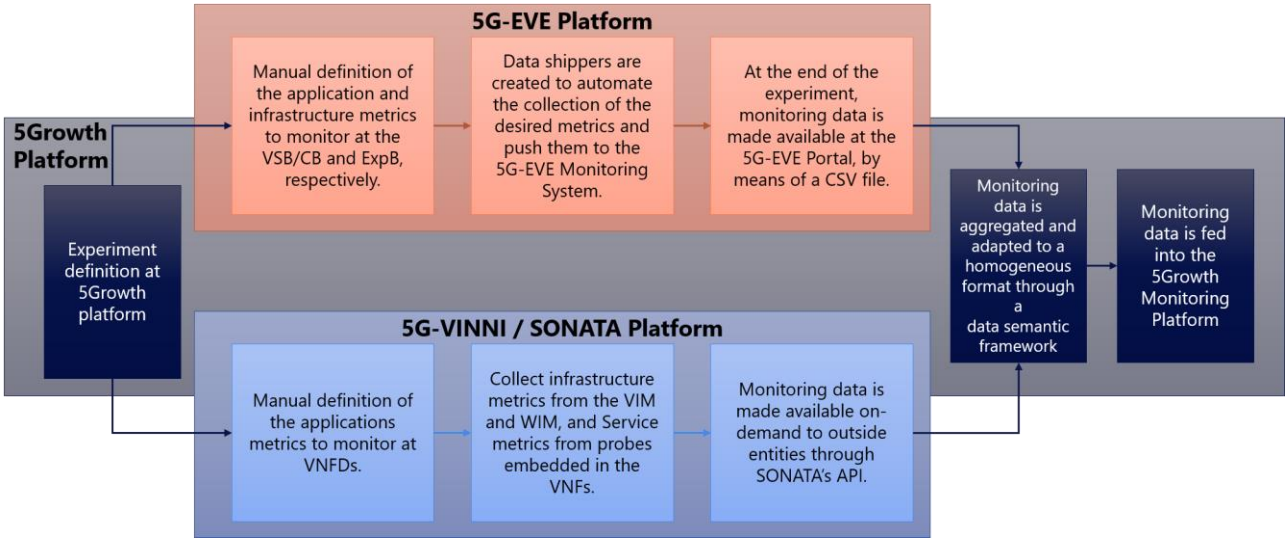


FIGURE 43: HIGH-LEVEL WORKFLOW FOR MONITORING DATA BETWEEN 5GROWTH AND ICT-17

Monitoring data collected from the ICT-17 sources are aggregated through a semantic data aggregation framework able to adapt data in an efficient way, suitable to be provided to the data consumers in the 5Growth architecture (as described in Section 3.4). At this point, monitoring data from the ICT-17 platforms are available for any internal usage within the 5Growth Platform.

There are a set of considerations that can be taken in the future for improving this workflow:

1. In case 5G-EVE and 5G-VINNI / SONATA platforms enable the dynamic definition of the metrics to monitor, automated workflows in the interaction with the 5Growth platform might be envisioned. Thus, monitoring metrics could be relayed by the 5Growth to the ICT-17 platforms during service request interactions.
2. The 5G-EVE project is evaluating the possibility to provide REST APIs for requesting the download of such CSV files in a programmable manner.
3. 5G-EVE Portal enables to monitor the progress of the experiment during its execution and visualize in real-time the graphs for the collected metrics and KPIs.

Building upon these considerations, less manual and more automated workflows can be envisioned, where data can be collected on-demand during the use case execution. Such scenario opens the way for a more dynamic application of the 5Growth innovations developed within WP2, that can not only use information generated by the different 5Growth building blocks but also by the ICT-17 platforms. The outcomes of this deliverable are going to be leveraged by WP3, and namely T3.2, to assess the feasibility of extending the current interactions between 5Growth and ICT-17 platforms to support the exchange of monitoring data.

4.4. Data infrastructure and innovations

4.4.1. WP2 innovation module as data producers/consumers

WP2 Innovations are grouped into three main innovation clusters: Architecture (I1-I7), Algorithms (I8-I10) and Framework (I11-I12). To perform many of the activities conducted on these innovation clusters heterogenous data need to be consumed and produced where the Monitoring Platform (I2) arises as the main data source. Thus, the Monitoring Platform acts as the "monitoring gateway", given that 5Growth's architecture implements an architecture where monitoring is centralized. Centralizing all the responsibility of the monitoring tasks into the monitoring platform releases the innovation designers/developers from the extra burden of dealing with monitoring tasks (e.g., subscribing, feeding data, etc.).

All the 5Growth WP2 Innovation Modules feed the monitoring platform. The monitoring platform is responsible for processing the information and provides the required capabilities for querying the monitoring data or alert notifications towards the rest of the components of the platform. The 5Gr-MP is flexible enough to accommodate additional types of data sources.

The 5Growth platform is based on the 5G-TRANSFORMER as the main baseline architecture. Thus, the 5Growth Vertical-oriented monitoring system (5Gr-VoMS) leverages the deployment made for the 5G-TRANSFORMER Monitoring Platform. The figure below illustrates an overview of the functional architecture of the 5GT-SO Monitoring Platform. Further details about the 5G-TRANSFORMER architecture can be found in 5G-TRANSFORMER D4.3 [17].

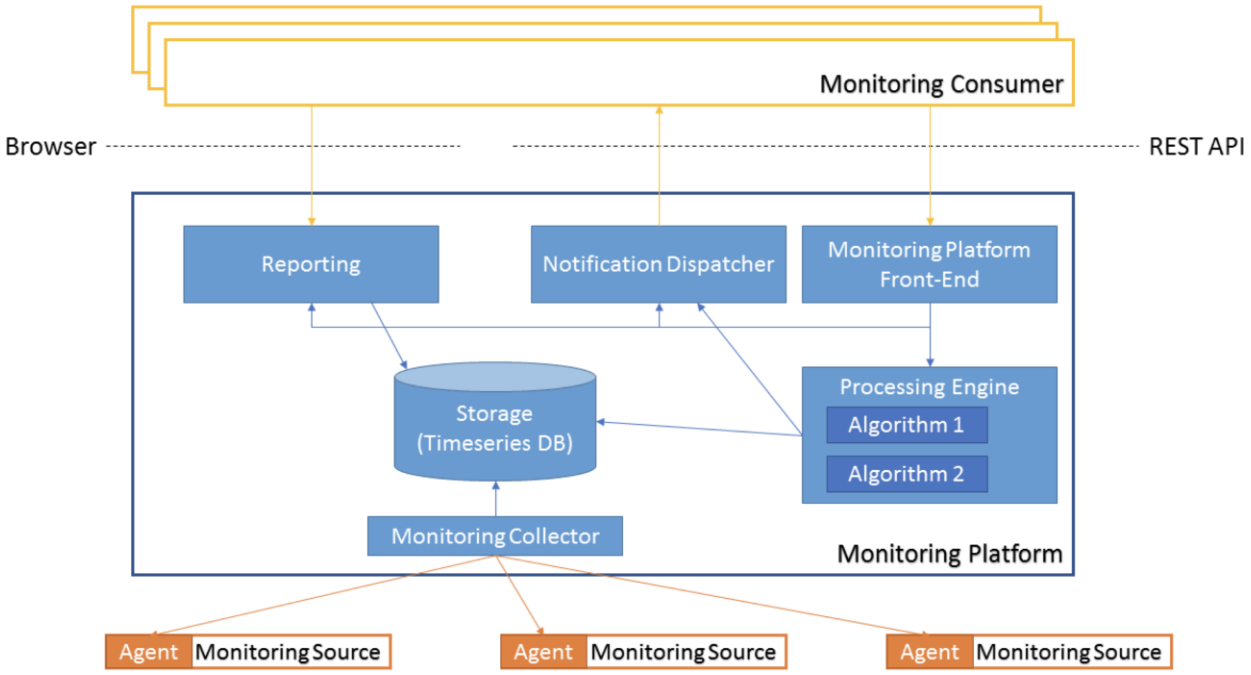


FIGURE 44: THE 5G-TRANSFORMER MONITORING PLATFORM MODULES

4.4.1.1. Monitoring sources - WP2 innovation modules as data producers

This section identifies those WP2 innovation modules that provide data to the Monitoring Platform through the Southbound Interface (SBI) and more specifically to the Monitoring Collector Module of the MP. The 5GT Monitoring Platform (hence the 5Growth platform) has been implemented by integrating a Prometheus backend with the 5GT-SO. Therefore, the agents at the monitoring sources are implemented as Prometheus exporters, while the Monitoring Collector, the storage and the processing engine components are all implemented by the Prometheus core engine. Thus, each WP2 innovation module that produces data for the Monitoring Collector of the MP, is communicating through dedicated agents that translates between different information models, monitoring mechanisms or message protocols. More specifically, these WP2 innovation modules are:

- Innovation 5: Control and Management. AI/ML Support
  - Data regarding SLA compliance and to support Vertical Slicer operations
- Innovation 9: Anomaly Detection
  - This Innovation module will create new types of events and alerts towards Vertical Slicer
- Innovation 10: Forecasting and Inference
  - Provide insights in order to lower the number of required resources (e.g. proactively allocate VMs, containers, connections) and to minimize KPIs (e.g. energy consumption etc.)
- Innovation 11: Security and Auditability

- The integration of Innovation 11 mechanisms can be applied to support the verification of orchestration SLAs and extended to incorporate monitoring actions and even measurement themselves and provide them to the I2 MP.

4.4.1.2. Monitoring consumers – WP2 innovation modules as data consumers

The post-processing data which are stored in the internal database, are available for the WP2 innovation consumers, which can retrieve them through queries (through the Monitoring Service front-end) or notifications (through the Notification Dispatcher) at the Northbound API (NBI). The 5Growth architecture integrates a Kafka messaging queue to the Monitoring Platform, where WP2 innovation modules could collect information in real-time from a predefined Kafka topic.

The list below identifies the WP2 innovation modules that need to communicate through predefined interfaces with each of the two NBI interfaces (i.e., the Monitoring Service front-end and Notification Dispatcher).

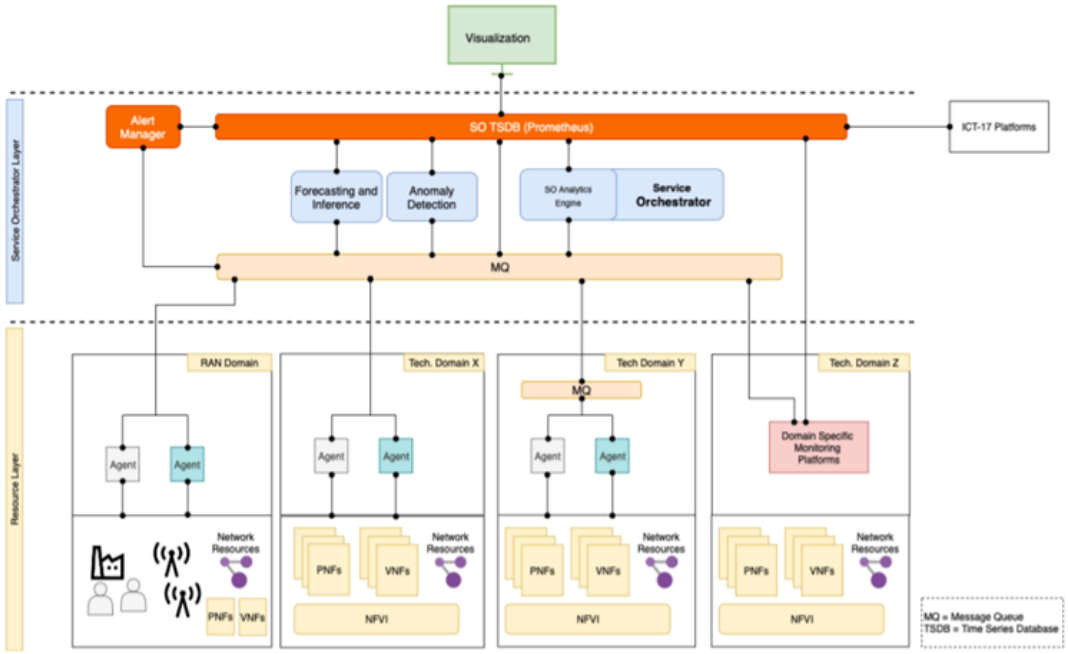


FIGURE 45: INFORMATION COLLECTION THROUGH THE KAFKA MQ

- Innovation 1: On Service level – Monitoring and Telemetry
  - RAN resources associated to the network slice (Vs-Mo)
- Innovation 2: Monitoring Platform
- Innovation 3: Orchestration and Monitoring Functions
  - Management of the monitoring functions, lifecycle management and metadata aggregation.
- Innovation 4: Control and Management - On inter-domain and federation
  - Closed-loop includes the process of collecting monitoring data from the services and networks, performing real-time data analytics for identifying events

and alarms, so as to take proper orchestration decisions for optimization and re-configuration of the system.

- Innovation 5: Control and Management. AI/ML Support
  - From data made available by the telemetry and monitoring platform, this innovation focuses on identifying proper training and testing datasets, in collaboration with the verticals.
- Innovation 6: End-to-End Orchestration on inter-domain and federation
  - Request and access to monitoring and measurement data, which in turn, is closely linked with auditability (Innovation 11)
- Innovation 7: End-to-End Orchestration on NG-RAN
  - RAN metrics through RI-Mo for RAN functions, UE profiling etc.
- Innovation 8: Smart Orchestration
- Innovation 9: Anomaly Detection
  - I9 will utilize the Monitoring Platform in order to analyse the network, computing and storage resource utilization, slice specific KPIs, RAN measurements, data traffic patterns, mobility patterns (if available).
- Innovation 10: Forecasting and Inference
  - Data required for time series analysis, artificial intelligence, machine learning, and statistical inference (historic data of VM requests, containers, connections etc.).
- Innovation 11: Security and Auditability
- Innovation 12: CI/CD

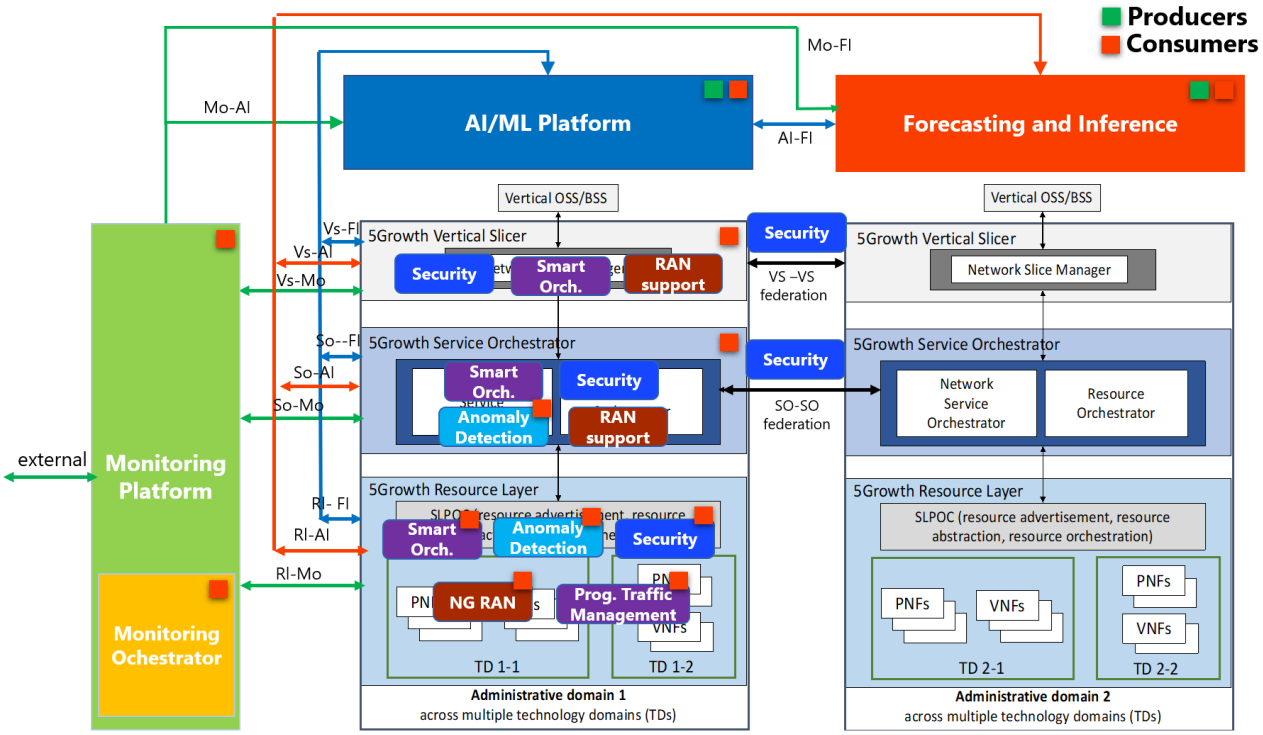


FIGURE 46: DATA PRODUCERS AND CONSUMERS IN THE 5GROWTH ARCHITECTURE

4.4.1.3. WP2 innovation modules as notification producers (NBI) – Graphical interfaces and dashboards

The MP reporting component (NBI) is configured via REST API through the unified Monitoring Platform front-end, providing the description of the "per-service" dashboard(s) to be made available to the users (e.g., in terms of type and format of monitoring parameters to be visualized). When the vertical connects to the URL of a generated dashboard, the reporting component then generates the graphs on-demand by querying the monitoring timeseries database.

Innovation 2 (on Monitoring Platform), besides the collection of the performance metrics expressed in the NSD, it is also in charge of the visualization using dashboards based on Kibana. Thereby, Kibana is used for visualizing Elasticsearch documents and helps developers to have a quick insight into it. Kibana dashboard offers various interactive diagrams, geospatial data, and graphs to visualize complex quires. It can be used for search, view, and interact with the data stored in Elasticsearch directories. Moreover, Kibana helps to perform advanced data analysis and visualize the data in a variety of tables, charts, and maps.

The list below showcases the WP2 innovation modules that could utilize Kibana for visualizing and help developers to have better insights

- Innovation 1: Real-time Network Slice status.
- Innovation 3: Real-time status update on the Monitoring Functions.
- Innovation 4: Visualisation of the real-time data analytics of identified events and alarms.
- Innovation 5: Real-time charts and graphs of the related network metrics and detected anomalies/events.
- Innovation 10: Visualisation of the data from the time series analysis, through line-plots and charts.
- Innovation 12: Visualisation of the status of the containers and VMs from Kubernetes through the Dashboard.

4.4.2. Interfaces - Monitoring Platform and interconnections between modules

4.4.2.1. Interfaces for vertical service monitoring

As mentioned above a significant amount of the 5Growth interfaces are inherited from the 5G-TRANSFORMER platform. In what follows, we summarize such inherited interfaces of 5G-TRANSFORMER relevant for the monitoring vertical services in 5Growth:

- 5Growth Vertical Slicer (5Ge-VS):
    - Ve-Vs: This interface is located at the northbound of the VS and interacts directly with a vertical. The interface provides different mechanisms to a vertical for monitoring the performance and failures of a deployed vertical service. In detail, it allows retrieving a monitoring parameters of a vertical service instance, enables creating subscriptions and notifications of monitoring parameters and allows creating alarms related to failures on the deployed instances.

- 5Growth Service Orchestrator (5Gr-SO):
  - Vs-So (-MON): This interface lies between the southbound of the VS and the northbound of SO. It has a reference point for monitoring network services such as VNFs and failures. Such an interface enables monitoring through either queries or subscriptions/notifications about the performance metrics and network service failures.
  - So-So (-MON): This interface allows federated SOs to retrieve monitoring data through the so-called east/west interface. It provides the same functionality as the one described above to federated SOs.
- 5Growth Resource layer (5Gr-RL):
  - So-MTP (Resource Monitoring Management): This interface's reference point allows the interaction between the SO on the southbound and the MTP in the northbound. It provides the required interworking procedures including the primitives and parameters for supporting the 5GT-SO Monitoring Service capability. On the one hand, it allows the functionality for continuous performance monitoring of the network service instance. On the other hand, it provides a fault management functionality to recover and restore interrupted network services.

4.4.2.2. Interconnection between WP2 innovation modules

The AI/ML Platform innovation module (I5) is responsible of hosting the AI/ML algorithms, in order to adapt to individual instances of slices, to detect patterns of events, which can then be associated with future situations, trends or problems. AI/ML algorithms will be made available by the following innovations:

- Innovation 2: Vertical-oriented Monitoring System
- Innovation 4: Control-loop Stability
- Innovation 8: Smart Orchestration and Resource Control
- Innovation 9: Anomaly Detection
- Innovation 10: Forecasting and Inference

Moreover, if needed by the innovations mentioned above, the targeted I5 AI/ML-based platform will exploit those functionalities tackled within the I3 such as dynamic instantiation of probes. This enables the execution of specific AI/ML algorithms that can take advantage of improved monitoring.

The architecture offered by this innovation will constitute the fundamental building block for all the supported innovations, which will be provided with a common set of features and interfaces suited for the needs of all of them.

4.4.2.3. Interfaces with external platforms

EMP-MO: Used by Monitoring Orchestrator (I3) to gather information from *External* Monitoring Platforms. This interface facilitates the integration with external platforms to support monitoring for both multi-domain and federation use cases. Thus, the WP2 innovation modules, will produce/consume data from external non-5Gr platforms, through the 5Gr Monitoring Platform.

MO-MP: Connects the Monitoring Orchestrator to the Monitoring Platform. The MP opens this interface for the Monitoring Orchestrator for instantiating/ provisioning monitoring functions by the Monitoring Orchestrator.

MO-MF: Used by the Monitoring Orchestrator to manage the monitoring functions, lifecycle Management and metadata aggregation.

MF-MP: Used by the Monitoring Platform to receive raw monitoring data from monitoring functions

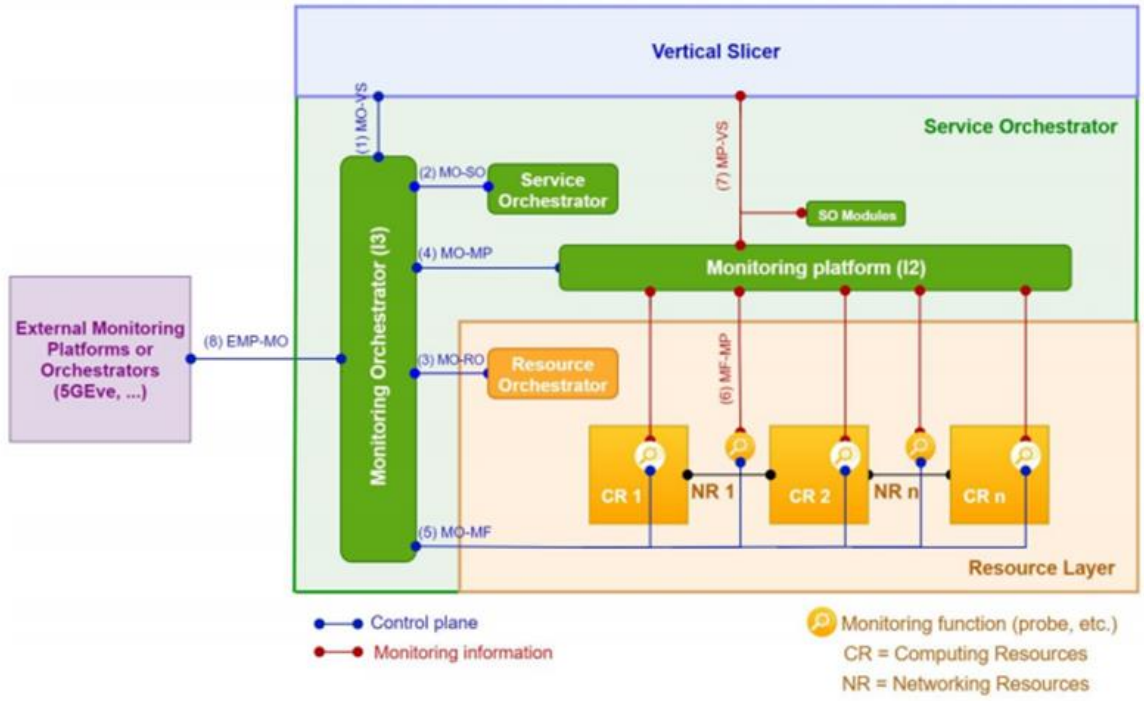


FIGURE 47: INTERFACE BETWEEN THE 5GROWTH ARCHITECTURE AND EXTERNAL ICT-17 PLATFORMS

4.4.2.4. Messaging queues

Messaging Queues (MQs) can be used as an interface for the exchange of information between different technologies and components of the architecture. In this way, internal and external components (e.g., ICT-17 projects, federated domains, etc.) can read/publish information in a common manner, avoiding the definition, creation, and implementation of new APIs.

WP2 Innovation specific Kafka topics will be implemented, in order for each Innovation module to have access to data streams from the I2 Monitoring Platform.

5. Initial pilot validation

Within 5Growth, pilot use cases are validated in the context of the successive releases of the experimental environments and reported in the corresponding deliverable. Once we have described the methodology for these experiments, the characteristics of the data infrastructure in use, and their integration with the whole 5Growth framework, this section focuses on the actual report for the initial validation results, mainly executed at ICT-17 premises.

5.1. Industry 4.0 pilot – INNOVALIA

This section includes the results obtained of the testing performed around the use cases within the INNOVALIA pilot.

5.1.1. Use Case 1: Connected worker remote operation of quality equipment

Some preliminary first testing was performed with the following setup:

- The devices on each end were connected to a different vendor 5G Customer Premises Equipment (5G CPE).
- At the time of these tests, the lab had only one 5G cell, so both 5G CPEs were connected to the same cell. This means that radio resources are shared if both CPEs are transmitting at the same time.
- Joystick and scanner devices were emulated by software applications on regular PCs.
- The video solution used was not the definitive solution designed by Interdigital, as it had not been possible yet to install it in the 5TONIC lab due to COVID-19 restrictions. These tests were done with a raspberry pi camera.

Following picture (Figure 48) shows the equipment connected to the 5G network in the 5TONIC lab and the IP assignment for the interconnectivity.

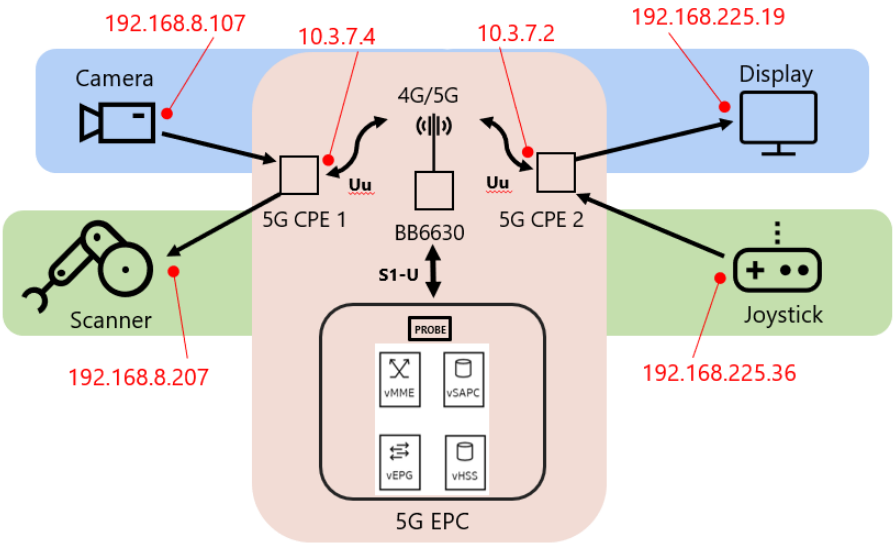


FIGURE 48: INNOVALIA UC1 FIRST EXPERIMENT SETUP

The executed tests were:

- pings from joystick to scanner to obtain End-to-End (E2E) latency on the network.
- iperfs from camera to display and from joystick to scanner to obtain peak throughput on the network.
- actual video streaming tests to measure video application E2E latency and data rate. The video application demanded 20 Mbps and the latency measurements showed 45 ms, as it includes image processing.

The results obtained in this first experiment are summarized in the following table.

TABLE 30: INNOVALIA UC1 FIRST EXPERIMENT MEASURED METRICS

CKPI	Measured metrics	5G NSA (1) Results	5G NSA (2) Results
CKPI-1 End-to-end Latency	RTT Joystick to Scanner Latency (ping path: `Joystick -> CPE 1 -> RAN UL -> 5G EPC -> RAN DL -> CPE 2`)	14 ms	14 ms
	RTT Camera to Display Latency (network) (ping path: `Raspberry PI -> CPE 1 -> RAN UL -> 5G EPC -> RAN DL -> CPE 2`)	15 ms	15 ms
	Video streaming application RTT Latency	45 ms	45 ms
CKPI-3 Guaranteed Data Rate	Peak throughput (scanner operation)	52 Mbps	90 Mbps
	Peak throughput (video streaming)	52 Mbps	90 Mbps
	Video streaming application data rate	20 Mbps	20 Mbps

(1) 3,5 GHz frequency band with TDD pattern 7:3 and 50 MHz bandwidth

(2) Adding to (1) an extra LTE 20 MHz bandwidth

A second round of testing was performed in the 5TONIC lab, following a similar setup than the one described above, but using the actual INNOVALIA scanner, and connecting an additional PC to the scanner to run the virtualized app. For this experiment, the 5G SA network was used. At the time of the experiment, the machine-to-machine communication between 2 CPEs was not working in the SA network in 5Tonic, so the joystick was connected through cable to the 5Tonic lab network, and there was connectivity end to end using one segment of RAN. Figure 49 illustrates the idea. The virtualized application is deployed on a PC in the user device end as an intermediate step towards the deployment of the application on the network edge.

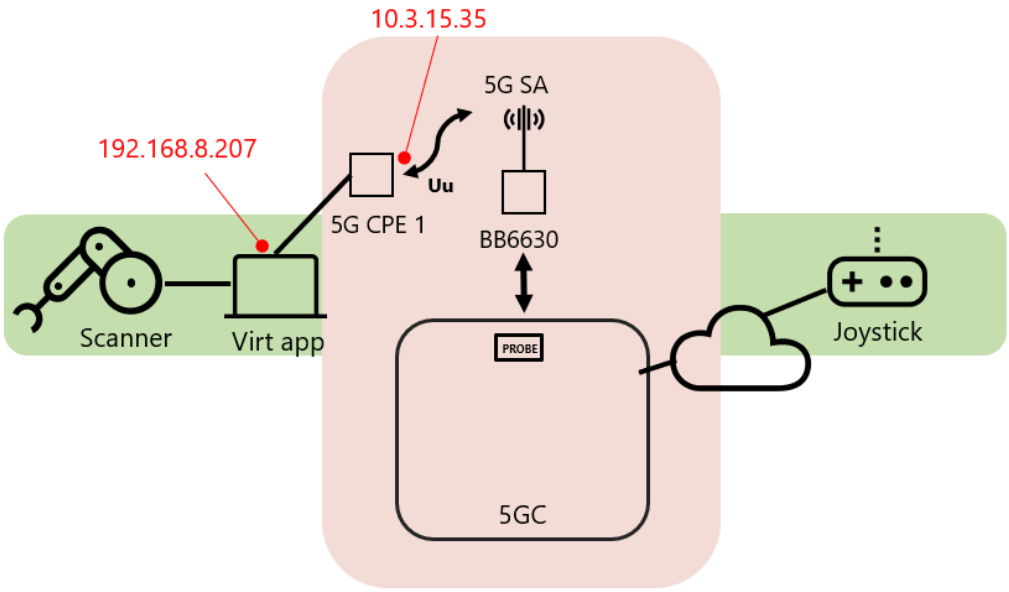


FIGURE 49: INNOVALIA UC1 SECOND EXPERIMENT SETUP

The test executed was the sending of commands from the joystick end, to move the scanner on the other end.

The results obtained are reported in the following table. The traffic of this application is not continuous, the packets are sent when needed. In the tests done, the application demanded a maximum of 7 Mbps (averaged per second).

TABLE 31: INNOVALIA UC1 MEASURED METRICS WITH THE ACTUAL SCANNER

CKPI	Measured metrics	5G SA (1) Results
CKPI-1 End-to-end Latency	Joystick-scanner operation app RTT Latency	15 ms
CKPI-3 Guaranteed Data Rate	Scanner->joystick operation app data rate UL	7 Mbps
	Joystick->scanner operation app data rate DL	0.5 Mbps
	Peak throughput UL	18 Mbps
	Peak throughput DL	330 Mbps

(1) 3,5 GHz frequency band with TDD pattern 4:1 and 40 MHz bandwidth

5.1.2. Use Case 2: Connected worker: Augmented Zero Defect Manufacturing (ZDM) Decision Support System (DSS)

No testing has been performed yet regarding this use case. Due to delays in the pilot caused by the COVID-19 crisis, it was decided to put the focus on the UC1 applications testing. The metrics obtained from the network in UC1 testing, are also valid for UC2. When a successful deployment of at least basic UC1 functionalities on the 5Growth stack is validated, then UC2 applications will be tried.

5.2. Industry 4.0 pilot – COMAU

The measurements more relevant for each use case, collected according to the methodology described in Section 2.2.2, are presented below.

5.2.1. Use Case 1: Digital twin apps

We look at the *latency* as the main metric for Use Case 1. Latency was measured using LaTe and encapsulating LaMP in a UDP payload of size 24 B. RTT, DL and UL delays were measured using the same kind of probe packets travelling in both directions, but the actual measurements of each of these quantities was carried out independently. The measurements could not be carried out continuously because other use case testing was underway, therefore the probing was conducted during "measurement windows" in which the network was devoid of other interfering traffic. During an active measurement window, each latency test ran for 18 seconds, involving the exchange of, roughly, 165 packets (packet transmission times have a little random jitter added to avoid synchronization issues).

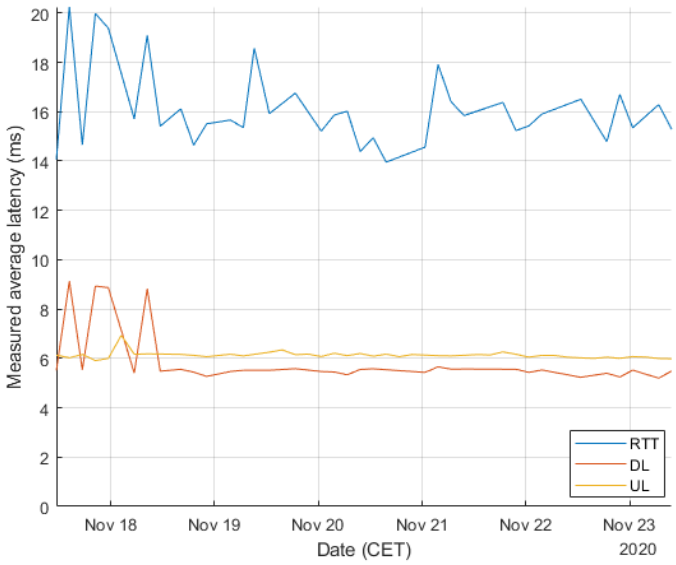


FIGURE 50: AVERAGE LATENCY

As shown in Figure 50, over the whole measurement week, the average RTT was 16.51 ms, the average DL latency was 5.42 ms and the average UL latency 6.1 ms. It should be noted that RTT > UL+DL, with the extra latency probably due to internal operations in the CPE and in the EPS switches.

In addition to the latency measurements on the mobile infrastructure, according to the methodology reported in 2.2.2.1, the latency introduced by the optical transport layer has been assessed as detailed in 2.2.2.2.

As for transport, the used fiber span of 8.8 km introduces a delay of 44 µs due to light propagation in glass (5ns/m). The measurement shows that the latency of the transport network is essentially due to the fiber span (44 µs); the additional delay due to the digital processing (i.e., switching and framing) is below the instrument precision (1 µs). Latency is independent of packet size or line load.

The next figure reports the output of the instrument.

RFC Through Put Test

Stream Id	# 16	MAC Dest Addr	00-00-00-00-00-00
Trial Duration	10 (Secs)	Dest Port	Packet(same)
BW Ceiling	100.00 %	BW Floor	10.00 %

Accept Lost Rate – 0 % Resolution Rate – 1.00 %

Latency Iterations - 20

				Latency (usec)		
Size	Rate	Tx Packets	Rx Packets	Min	Max	Avg
64	100.00	148809559	148809559	44	44	44
128	100.00	84463760	84463760	44	44	44
256	100.00	45291819	45291819	44	44	44
512	100.00	23497278	23497278	44	44	44
1024	100.00	11973693	11973693	44	44	44
1280	100.00	9615773	9615773	44	44	44
1518	100.00	8117317	8117317	44	44	44
9000	100.00	1385868	1385868	44	44	44

FIGURE 51: LATENCY TEST RESULTS

A shown in the next figure, the actual throughput is equal to the theoretical one, regardless the frame length (100%).

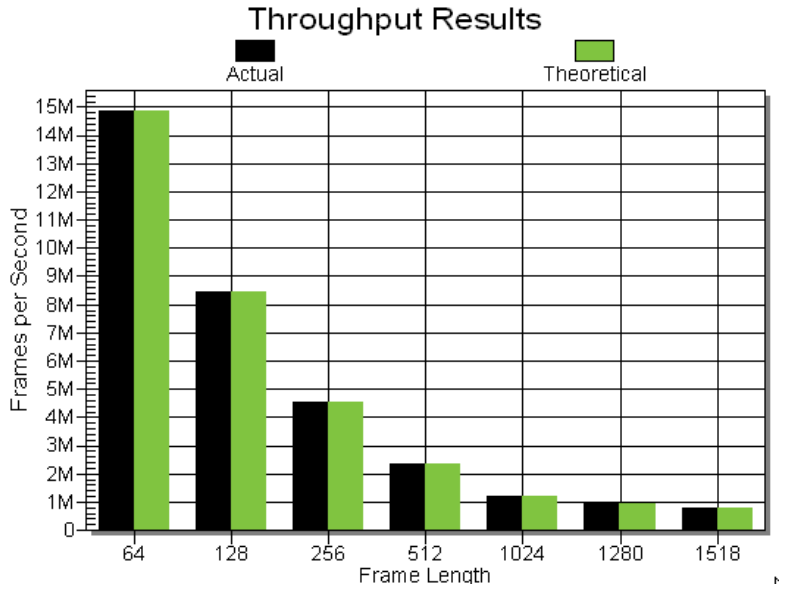


FIGURE 52: THROUGHPUT TEST RESULTS

No frame loss has been detected in the optical communication, regardless the frame length or the traffic load. The back-to-back burst value is the number of frames in the longest burst that the DUT will handle without the loss of any frames. The test shows that no frames are lost regardless the burst length. In the following figure the results are shown for a 10-seconds long burst; longer bursts showed the same result.

Frame Size (bytes)	Max Possible Number of frames	Avg Observed B2B Frames
64	148809520	148809520
128	84459440	84459440
256	45289840	45289840
512	23496240	23496240
1024	11973180	11973180
1280	9615380	9615380
1518	8127420	8127420
9000	1385800	1385800

FIGURE 53: BACK-TO-BACK BURST TEST RESULTS

In the end, the tests performed on the transport infrastructure, based on RFC2544 standard, prove that the network performances are more than adequate to support the architecture adopted in the pilot. The latency introduced by switching and framing layer is negligible for the application of interest. No frame loss or limited throughput has been observed during the tests. These considerations, related to transport performances, also applies to the other two use cases.

5.2.2. Use Case 2: Telemetry/monitoring apps

In Use Case 2, beside the *latency* reported in the previous subsection, the *packet loss* also plays a role in ensuring that telemetry data are always timely reported. The packet loss was recorded by LaTe while measuring the latency during each 18-second test (over 160-170 packets). The number of tests that failed due to a single-packet (out of 170) timeout were 4 out of 9505, leading to a packet loss of around $2*10^{-6}$, a value consistent with the bandwidth dips seen in throughput measurements (see next subsection), leading one to believe that they were likely due to the correlated temporary malfunctions.

5.2.3. Use Case 3: Digital tutorial and remote support

The *throughput* is the most important metric for Use Case 3. The TCP throughput was measured by iPerf with a socket application buffer size of 1000B and is shown in Figure 54 (downlink) and Figure 55 (uplink). Each plot shows the evolution of each metric during the measurement windows in the whole testing week. The plots report the values measured in consecutive 9-second "probe tests", at the end of which the throughput was averaged and the corresponding value inserted in the plot. It should be noted that the TCP transient was removed from each probe test (i.e., first two seconds of each probe test) in order to focus on the steady state of the network behaviour. Overall, the TCP DL average throughput measured over the week was approximately 821 Mbit/s. A handful of probe tests returned throughput values below 800 Mb/s (2 out of 9505), but they can be considered outliers. The TCP UL average throughput measured over the week was 59.01 Mbit/s.

FIGURE 54: DOWNLINK TCP THROUGHPUT

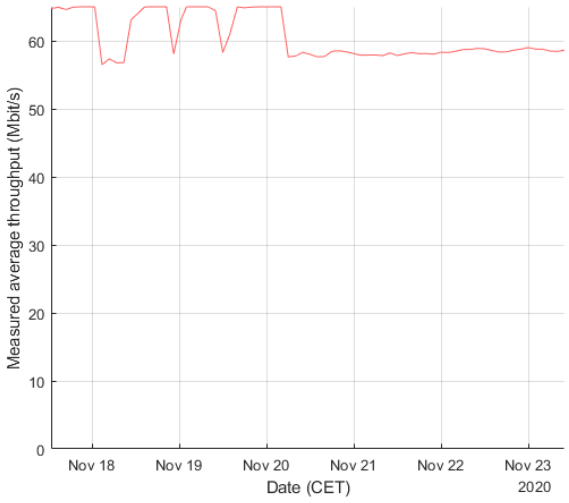


FIGURE 55: UPLINK TCP THROUGHPUT

The UDP throughput is shown in Figure 56 (downlink) and Figure 57 (uplink). The size of UPD packets was fixed at 1000B, which yielded a higher throughput than TCP, but more oscillations.

The UDP DL average throughput over the week was around 944 Mbit/s, while the UL average throughput over the week was 61.12 Mbit/s.

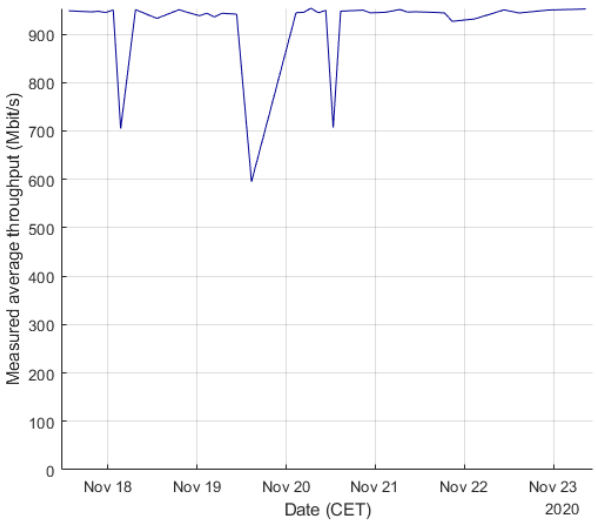


FIGURE 56: DOWNLINK UDP THROUGHPUT

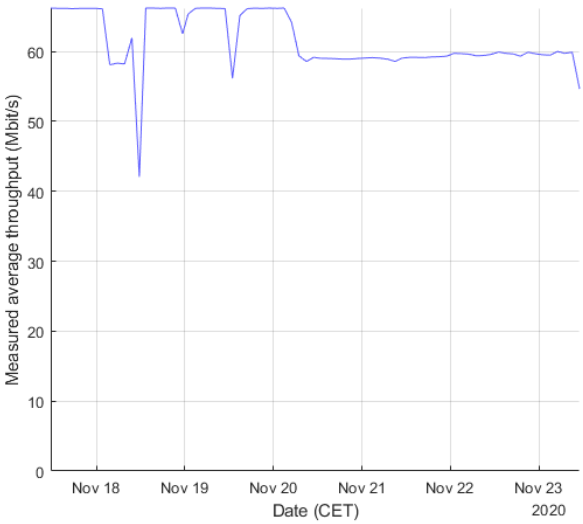


FIGURE 57: UPLINK UDP THROUGHPUT

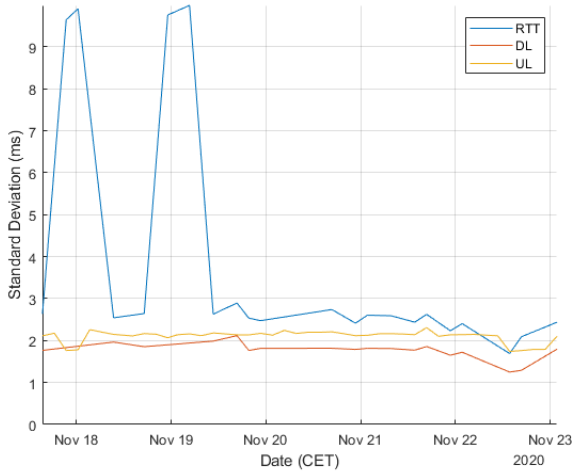


FIGURE 58: STANDARD DEVIATION OF LATENCY

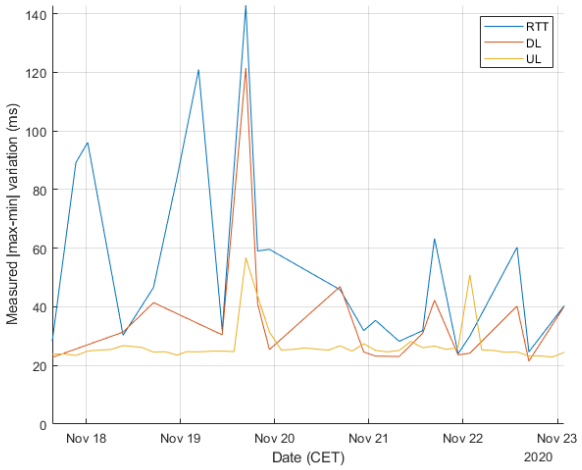


FIGURE 59: LATENCY VALUES EXCURSION

Finally, we have assessed latency variations that concur to the evaluation of jitter. In Figure 58, we show the average standard deviation of the latency in each batch of tests. Average standard deviation over the week was: 2.78 ms (RTT), 1.67 ms (DL), 2.11 ms (UL). It is worth observing that standard deviation peaks correspond to peaks in measured maximum latency. In Figure 59, the absolute value of the difference in the max-min latency amplitude (latency excursion) during each single test is reported. The recorded absolute maximum variations over the week were: 140.26 ms (RTT), 120.98 ms (DL), 57.04 ms (UL). It should be pointed out, though, that these maximum variations are relative to peaks which happened quite rarely, considering the total of 9505 tests.

5.3. Transportation pilot - EFACEC_S

5.3.1. Use Case 1: Safety critical communications

As described in 2.2.3, the first considered scenario for validations was deployed at EFACEC facilities (Lab) and the communications between the train detector sensor and the Level crossing controller was supported by LTE/4G. These measurements were performed to obtain information of the impact of using VPN to secure data communication and to verify if there were some difference according with the chosen path (LX controller to train detector or train detector to LX controller). A new campaign in the same conditions was performed at IT Aveiro premises with results aligned to the ones previously obtained.

The 5G network with emulated RAN scenario was deployed at IT Aveiro labs and the communications between the train detector sensor and the Level crossing controller was supported by a 5G core and an emulated RAN. Next figures show the RTT collected values with different packet size of 64, 200, 1000, 5000 and 10000 bytes. and a histogram representing 1000 captures for 64 bytes.

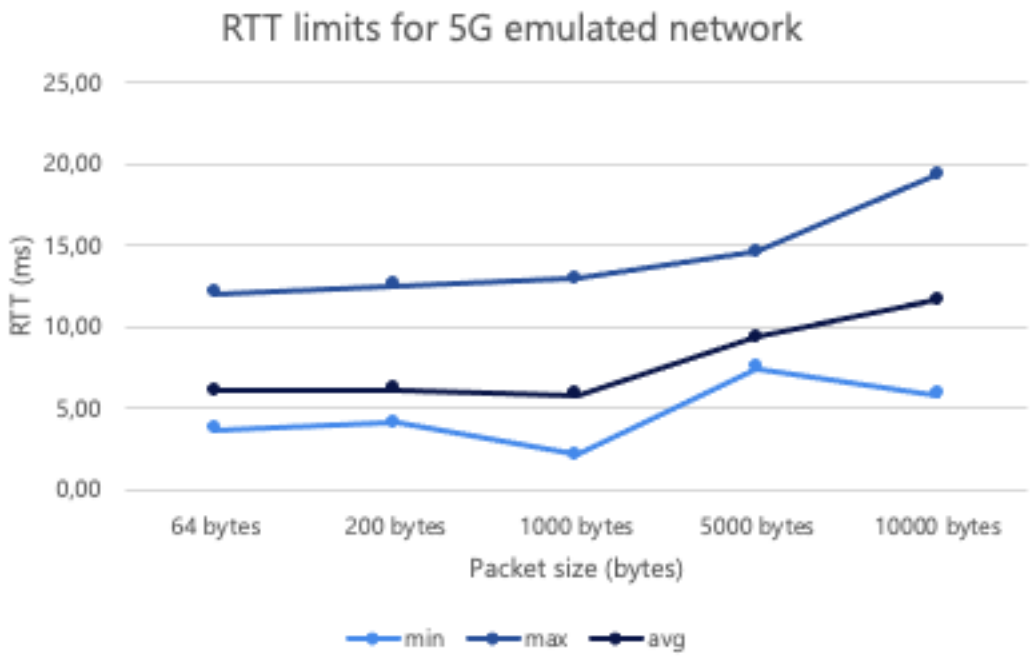


FIGURE 60: RTT VALUES FOR 5G NETWORK WITH EMULATED RAN

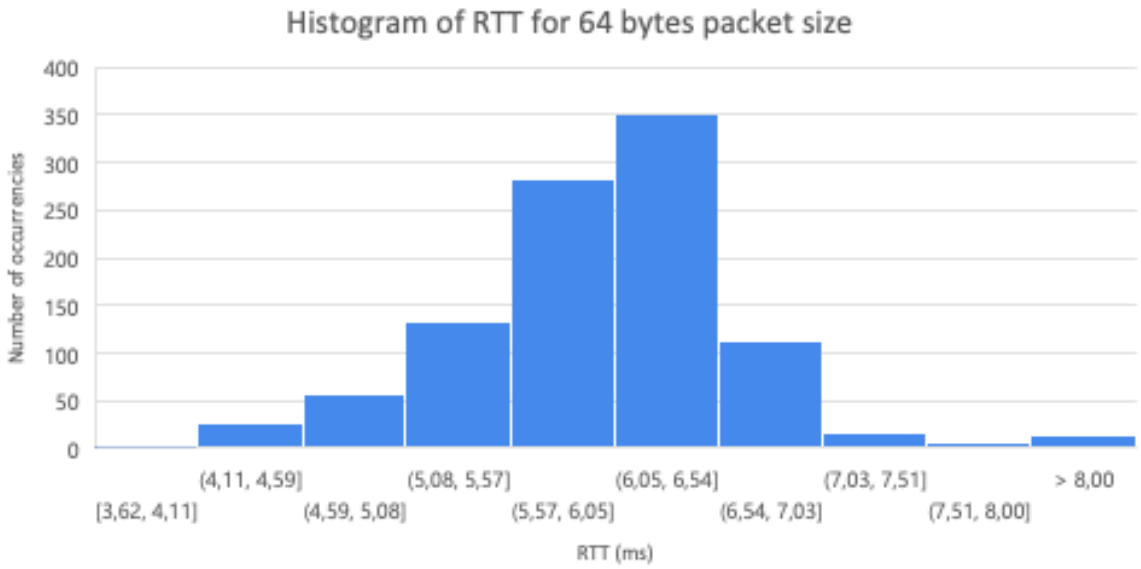


FIGURE 61: HISTOGRAM OF RTT FOR 64 BYTES PACKET SIZE

The values measured for the RTT were around 7ms, which is higher than what is specified for 5G networks, even when there was no radio delay, since this component is emulated.

All the tests were performed with the 5G network with emulated RAN dedicated for this test scenario and even so the RTT values sweep between 3.6 ms up to 12 ms which could be explained by buffering limitations or processing limitations on the 5G network with emulated RAN server side.

Regarding the performance tests, also throughput values were collected, and next tables shows the measurements on the UDP and TCP bitrates.

TABLE 32: THROUGHPUTS MEASURED FOR UDP PROTOCOL IN EFACEC_S UC1

Bitrate		Bitrate receiver		Jitter		Lost Datagrams	Total Datagrams	Percentage
6	Mbits/sec	5,97	Mbits/sec	0,389	ms	0	5778	0%
50	Mbits/sec	49,7	Mbits/sec	0,078	ms	0	48148	0%
100	Mbits/sec	99,5	Mbits/sec	0,096	ms	8	96296	0,01%
150	Mbits/sec	149	Mbits/sec	0,069	ms	679	144443	0,47%
200	Mbits/sec	174	Mbits/sec	0,054	ms	20400	192539	11%
250	Mbits/sec	176	Mbits/sec	0,731	ms	70010	240740	29%
300	Mbits/sec	122	Mbits/sec	0,063	ms	168182	288905	58%
350	Mbits/sec	120	Mbits/sec	0,082	ms	220740	337014	65%
400	Mbits/sec	121	Mbits/sec	0,004	ms	265160	385183	69%
450	Mbits/sec	122	Mbits/sec	0,044	ms	312890	433341	72%
500	Mbits/sec	122	Mbits/sec	0,021	ms	362862	481482	75%

TABLE 33: THROUGHPUTS MEASURED FOR TCP PROTOCOL IN EFACEC_S UC1

Bitrate		Tx retries		Note
8,3	Mbits/sec	3586	packets	Test executed over 60 seconds

For the throughput analysis with the UDP protocol, the results are around 120Mbps and a lost package value of 0.47% with a jitter value of 0.069ms. The jitter calculated with Ping can vary from 0.5 up to 0.9 ms and calculated with iPerf UDP protocol can vary from 0.004 up to 0.8 ms which are similar and very good values for any network.

The throughput analysis for the TCP protocol the results show a data rate of 8.3 Mbps over 1 minute which is also considerable low as well, only comparable to some 4G/LTE networks. Even with this low data rate, the retransmissions that occurred were 3586.

The Jitter values are also presented in the tables representing very good values for any network.

When comparing the RTT results between this 5G network with emulated RAN and the commercial 4G/LTE network the RTT value is $1/10^{th}$ of the average measure under a 4G/LTE network. The throughput using TCP is very similar to the one obtained using the 4G/LTE network but, when using UDP the throughput is much higher than the values obtained under the 4G/LTE network. The values of the measurements obtained in this test are very good in order to implement this use case requirements.

5.3.2. Use Case 2: Non-safety critical communications

In this use case all data traffic is characterized by UDP to transport RTSP packets.

The measurements for the initial LTE-based validation showed technology limitations for this video use case, mainly regarding the bandwidth. For values greater than 4 Mbit/s a great number of packet loss is detected causing a degradation of video quality. Jitter in this scenario was measured to be around 60ms for 99% of packets, increasing the end-to-end latency introduced by the jitter buffering needed at the receiver.

Concerning the scenario of 5G network with emulated RAN, the environment was deployed at IT Aveiro labs, with the communication between the level crossing video camera and the train video console supported by a 5G core. The performance of the 5G emulated network, in terms of UDP traffic, is very good and there is only some packet loss starting at 64 Mbit/s, representing, only 0.16%. The average latency observed in RTSP packets was 6.3 ms, and jitter below 4.5 ms. However, the radio link and VPN are missing, which will certainly increase the values in a real scenario.

The actual validation campaign was performed in the test scenario with a full RAN and 5G core deployed at IT Aveiro lab, according to the diagram in the figure below.

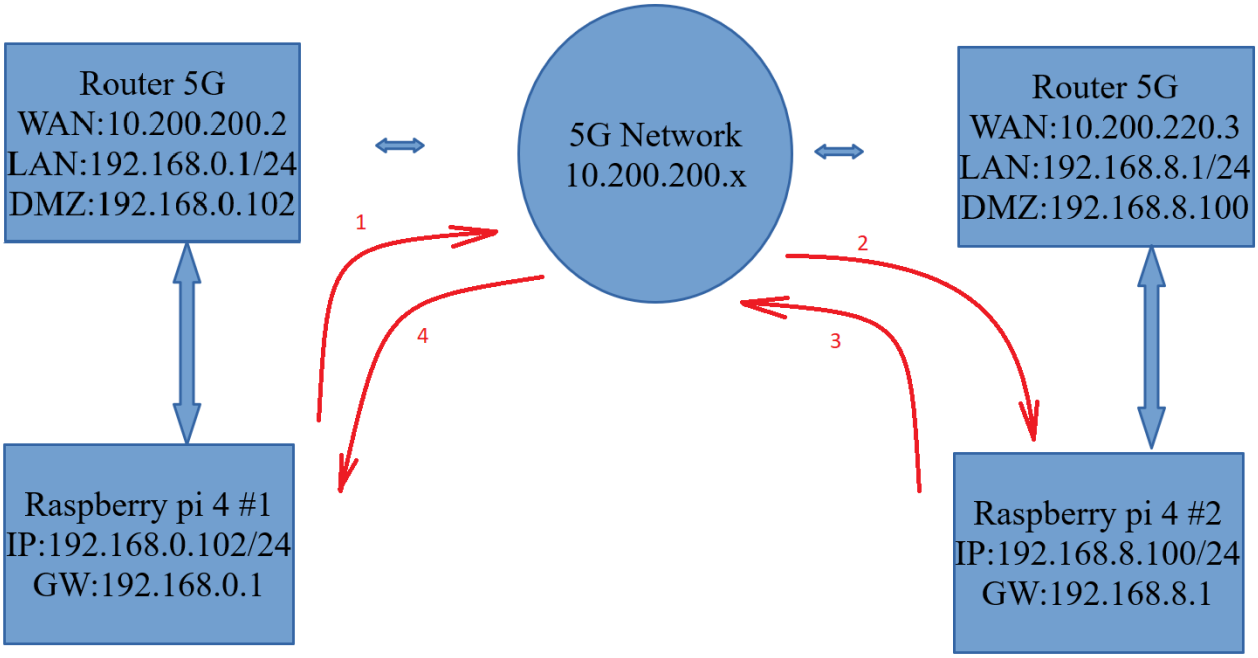


FIGURE 62: BLOCK DIAGRAM OF THE LAB TEST ENVIRONMENT

RTT values are considered as the sum of times measured for the four paths identified with a number in the figure:

1. Time that ICMP request datagram takes from equipment Raspberry Pi4 #1 to 5G Core network base station.
2. Time that ICMP request datagram takes from 5G Core network base station to the equipment Raspberry Pi4 #2.
3. Time that ICMP response datagram takes from equipment Raspberry Pi4 #2 to the 5G Core network base station.
4. Time that ICMP response datagram takes from 5G Core network base station to the equipment Raspberry Pi4 #1.

The Latency value for this test scenario (Raspberry Pi4 #1 – raspberry Pi4 #2) is the time that a datagram takes from the origin equipment up to the final equipment and in this case, we would assume that is half of the RTT value (path 1 + pah 2 – assuming that ICMP request datagram time is the same as the ICMP response time).

The average latency value for this test scenario = 28ms. This result is roughly in line with the latency tests performed between the CPE and the 5G Core (Internet Gateway) in which the average value obtained for RTT delay was roughly 50% compared to the CPE-to-CPE tests.

The following figure shows the RTT histogram of the 5G network.

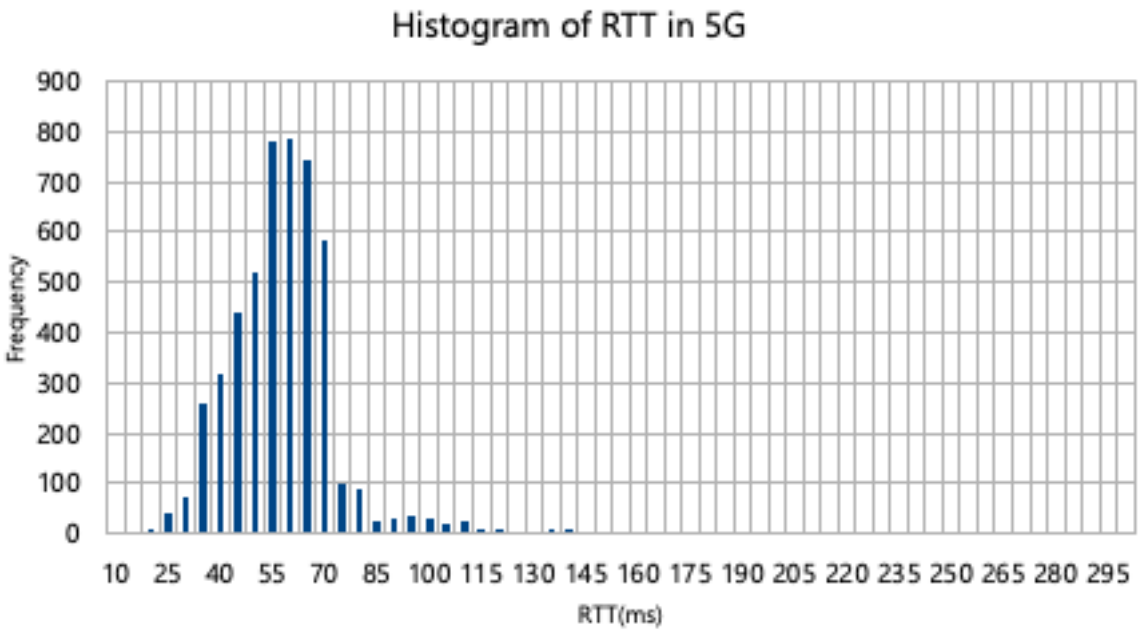


FIGURE 63: HISTOGRAM OF RTT IN THE REAL 5G NETWORK

The RTT average in this scenario is 56 ms. The average latency value for this test scenario = 28ms. This result is roughly in line with the latency tests performed between the CPE and the 5G Core (Internet Gateway) in which the average value obtained for RTT delay was roughly 50% compared to the CPE-to-CPE tests.

Next table shows the bandwidth test with iperf3 for UDP data traffic.

TABLE 34: BANDWIDTH FOR UDP TRAFFIC PROTOCOL IN EFACEC_S UC2

Test	Bandwidth (Mbits/s)	Jitter (ms)	Lost/Total Datagrams	
UDP-2Mb/s	1.99	6.747	0/1950	0%
UDP-4Mb/s	3.98	3.299	0/3900	0%
UDP-8Mb/s	7.98	1.283	0/7800	0%
UDP-16Mb/s	15.8	2.206	13/15578	(0.083%)
UDP-32Mb/s	30.2	0.593	1059/31152	(3.4%)
UDP-64Mb/s	48.2	0.255	14396/62390	-23%
UDP-100Mb/s	49.9	0.152	47802/97482	-49%
UDP-128Mb/s	50.6	0.256	73741/124780	-59%
UDP-256Mb/s	44.0	0.100	205052/249496	-82%

The performance of the 5G network, in terms of UDP traffic is usable for values around 30Mbps with 3.4% of lost datagrams. The maximum UDP throughput is around 50Mbps but the lost datagram is very high (around 50%). It should be noted that the limiting factor of data throughput is the radio uplink. Upstream and downstream throughputs were measured separately, and the results were in

the order of 60 Mb/s and 400 Mb/s, respectively. In any case, a more symmetrical result can be obtained through RAN configuration and will be tested in the future.

According the collected measurements some optimization were performed in 5G Network and the new results are the following:

- RTT (average) – 40 ms and the minimum achieved value was 20 ms
- Latency (average) – 20 ms minimum achieved value was 7 ms
- Jitter (average) – 2.6 ms
- Maximum uplink throughput +/- 70Mbps (UDP)
- Throughput (average) – 55 Mbps (TCP with 0 retries)

Note: According with ASOCS analysis the high jitter in low bitrates is caused by the Scheduling Request Periodicity, related to the request of scheduling radio resource for uplink transmission by the UE.

5.4. Energy pilot - EFACEC_E

This section includes the results obtained up to the date concerning the testing campaigns performed on EFACEC_E Vertical pilot.

5.4.1. Use Case 1: Advanced Monitoring and Maintenance Support for Secondary Substation MV/LV Distribution Substation

As detailed in the previous deliverables, the re-planning of the objectives for M18 made to mitigate the impact of Covid-19 lead to the deployment of part of the Use Case 1 in the IT Lab instead of the real Secondary Substation. This way, the first test campaigns were carried out in the transitory deployment depicted below.

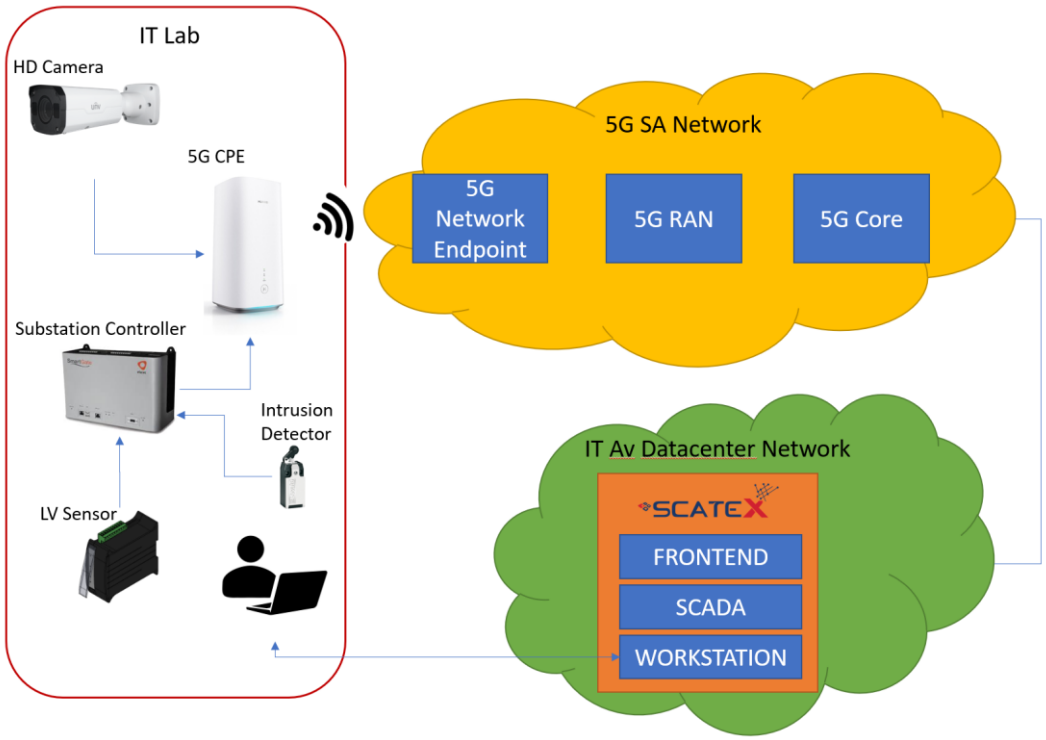


FIGURE 64: LAB ENVIRONMENT FOR EFACEC_E UC1 SUPPORTED BY 5G SA NETWORK

Two 5G network setups were tested. In first place, 5G network with emulated RAN, and more recently, 5G SA Network with ASOCS real RAN and Open5GCore.

5.4.1.1. Test Campaign with 5G network with emulated RAN

The first deployment setup used a 5G network with emulated RAN while the 5G Network with real RAN (ASOCS) was not available. The following diagram depicts the setup used in this test campaign, with the IP assignment for the interconnectivity.

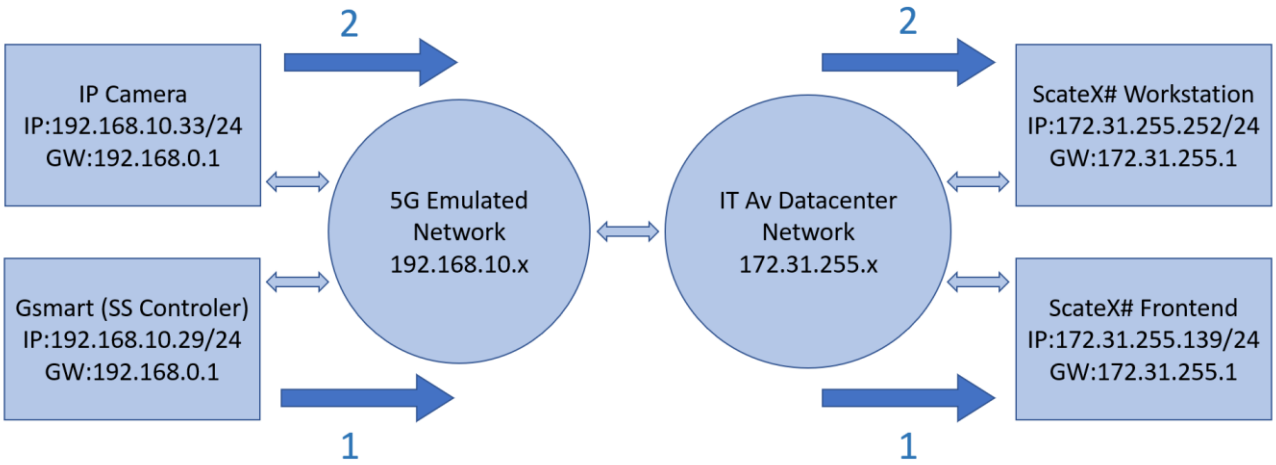


FIGURE 65: EFACEC_E UC1 SETUP USING 5G NETWORK WITH EMULATED RAN

To analyse the performance of the network two tools were used:

a) Ping (ICMP) to evaluate the end-to-end Round-Trip Time
b) iPerf to measure throughput capabilities of the network under UDP and TCP transport layer protocols

The results obtained in this first campaign are summarized in the following figures and tables.

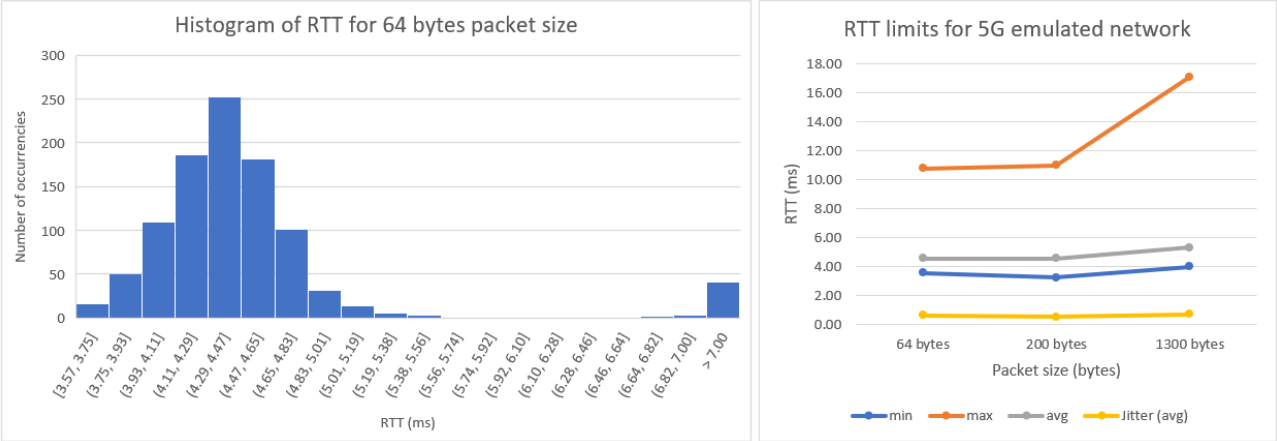


FIGURE 66: END-TO-END RTT TEST RESULTS

TABLE 35: THROUGHPUT TEST RESULTS (UDP)

Bitrate		Bitrate receiver		Jitter		Lost Datagrams	Total Datagrams	Lost/Total
1	Mbits/sec	1.0	Mbits/sec	0.219	ms	0	927	0.00%
6	Mbits/sec	6.0	Mbits/sec	0.169	ms	0	5561	0.00%
10	Mbits/sec	10.0	Mbits/sec	0.126	ms	0	9272	0.00%
20	Mbits/sec	20.0	Mbits/sec	0.088	ms	19	18537	0.10%
30	Mbits/sec	30.0	Mbits/sec	0.076	ms	25	27813	0.09%
40	Mbits/sec	39.9	Mbits/sec	0.082	ms	157	37120	0.42%
50	Mbits/sec	50.1	Mbits/sec	0.104	ms	41	46367	0.09%
60	Mbits/sec	60.0	Mbits/sec	0.091	ms	21	55776	0.04%
70	Mbits/sec	69.8	Mbits/sec	0.102	ms	16	64798	0.02%
80	Mbits/sec	79.6	Mbits/sec	0.100	ms	100	74105	0.13%
90	Mbits/sec	85.9	Mbits/sec	0.097	ms	3605	83388	4.32%
100	Mbits/sec	81.1	Mbits/sec	0.100	ms	4347	34360	12.65%

TABLE 36: THROUGHPUT TEST RESULTS (TCP)

Interval	Transfer	Bandwidth	Tx retries	
0.00-60.00 sec	669 MBytes	93.5 Mbits/sec	1	sender
0.00-60.00 sec	669 MBytes	93.5 Mbits/sec		receiver

5.4.1.2. Test Campaign with 5G SA network with real RAN and real core

The second test campaign already used the real RAN (ASOCS) and a real 5G core (Open5GCore). The following diagram depicts the setup used in this test campaign, the IP assignment for the interconnectivity.

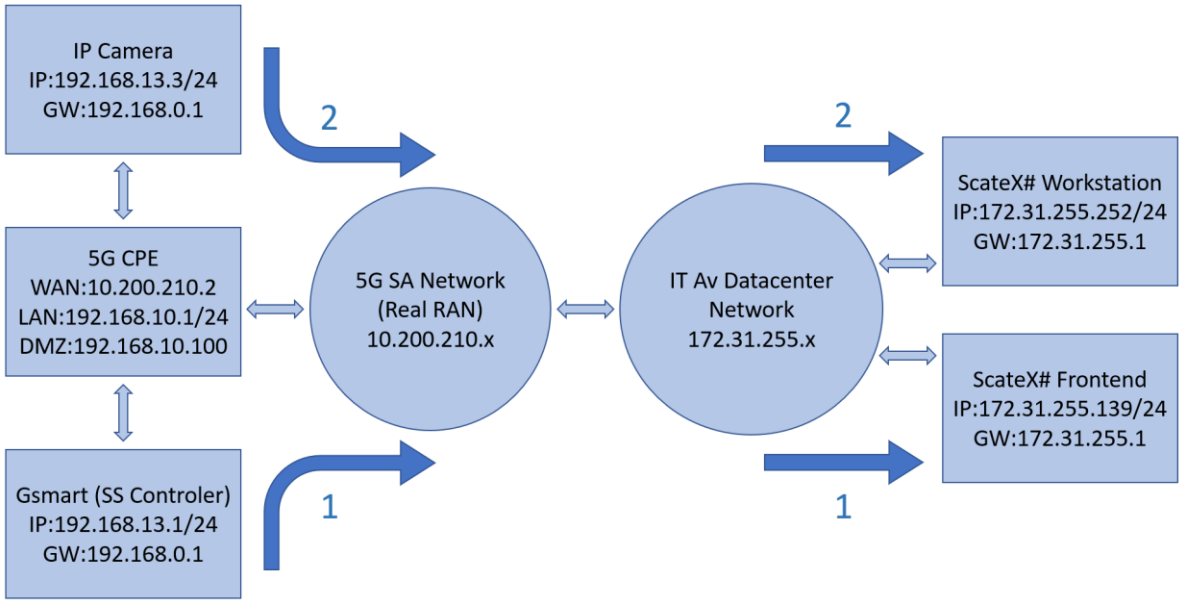


FIGURE 67: EFACEC_E UC1 SETUP USING 5G SA NETWORK WITH ASOCS RAN AND OPEN5GCORE

To analyse the performance of the network two tools were used:

c) Ping (ICMP) to evaluate the end 2 end Round-Trip Time
d) iPerf to measure throughput capabilities of the network under UDP and TCP transport layer protocols

The results obtained in this campaign are summarized in the following figures and tables.

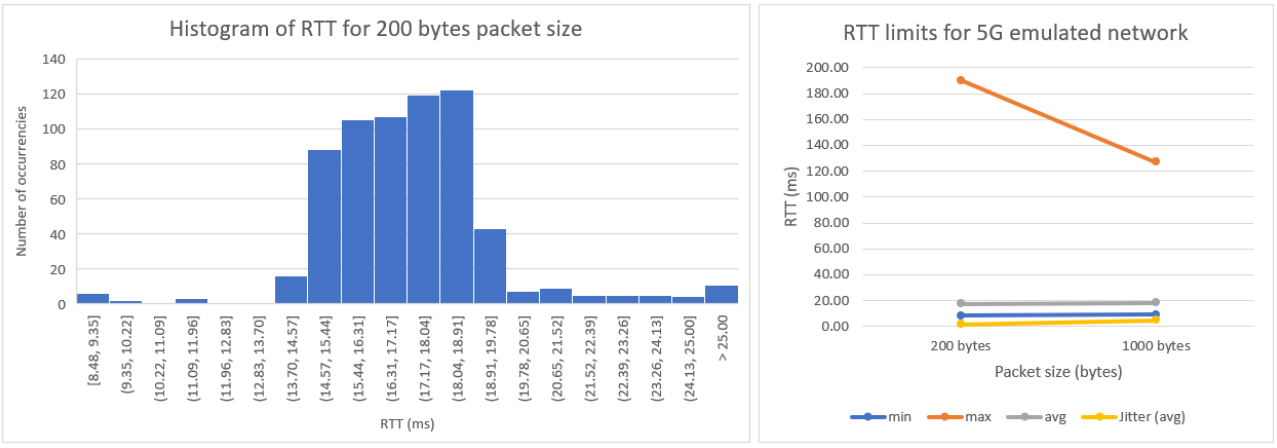


FIGURE 68: END-TO-END RTT TEST RESULTS

TABLE 37: THROUGHPUT TEST RESULTS (UDP)

Bitrate		Bitrate receiver		Jitter		Lost Datagrams	Total Datagrams	Lost/Total
1	Mbits/sec	1.0	Mbits/sec	1.698	ms	0	927	0.00%
6	Mbits/sec	6.0	Mbits/sec	0.243	ms	0	5561	0.00%
10	Mbits/sec	10.0	Mbits/sec	0.174	ms	0	9274	0.00%
20	Mbits/sec	20.0	Mbits/sec	0.165	ms	0	18537	0.00%
30	Mbits/sec	30.0	Mbits/sec	0.132	ms	0	27840	0.00%
40	Mbits/sec	39.9	Mbits/sec	0.137	ms	0	37120	0.00%
50	Mbits/sec	49.7	Mbits/sec	0.131	ms	2	46491	0.00%
60	Mbits/sec	59.6	Mbits/sec	0.225	0	0	55717	0.00%
100	Mbits/sec	67.2	Mbits/sec	0.148	ms	41404	92350	44.83%

TABLE 38: THROUGHPUT TEST RESULTS (TCP) - 1 CONNECTION

Interval	Transfer	Bandwidth	Tx retries	
0.00-60.00 sec	358 MBytes	50.1 Mbits/sec	180	sender
0.00-60.00 sec	357 MBytes	50.0 Mbits/sec		receiver

TABLE 39: THROUGHPUT TEST RESULTS (TCP) - 8 SIMULTANEOUS CONNECTIONS

Interval	Transfer	Bandwidth	Tx retries	
0.00-60.00 sec	238 MBytes	66.5 Mbits/sec	717	sender
0.00-60.00 sec	237 MBytes	66.4 Mbits/sec		receiver

Note: According with ASOCS analysis the high jitter in low bitrates is caused by the Scheduling Request Periodicity, related to the request of scheduling radio resource for uplink transmission by the UE.

5.4.2. Use Case 2: Advanced critical signal and data exchange across wide smart metering and measurement infrastructures

At this time, according to the Energy Vertical planning as indicated in the project roadmap, the Use Case 2 is not deployed yet.

6. Conclusions

This document closes the initial develop-validate cycle of the 5Growth project, providing

- A further analysis of network and service KPIs, including the mapping between them.
- A description of the available measurement methodology and tooling, including the activities addressed within WP3 for supporting the interaction of the 5Growth platform with existing ICT-17 platforms, and the application and adoption of the devised WP2 innovations
- The results of the first verification campaigns conducted by the different pilots

The structure of D4.2 provides the general pattern for further contents of WP4 deliverables for the next two cycles planned for the project. These deliverables will be structured around the same idea of combining reports on the evolution of methodology and tooling, and the analyses of verification results. WP4 foresees these contents will evolve according to the scheme depicted in the figure below.

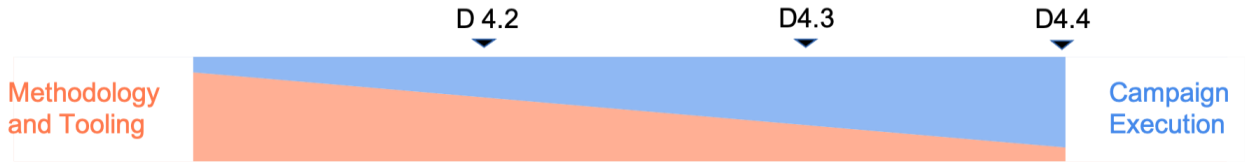


FIGURE 69: FORESEEN EVOLUTION OF WP4 DELIVERABLE CONTENT

Where the amount of information focused on methodology and tooling will decrease, while the one reporting results and analyses from verification will increase, as the project matures.

7. References

[1] 5Growth, D4.1, Service and Core KPI Specification, May 2020.

[2] I. Joe, "Packet loss and jitter control for real-time MPEG video communications" *Computer Communications* vol. 19 no. 11, pp. 901–914, Sep 1996 DOI: 10.1016/S0140-3664(96)01124-3

[3] "Latency in Optical Fiber Systems", available online at https://www.commscope.com/globalassets/digizuite/2799-latency-in-optical-fiber-systems-wp-111432-en.pdf?r=1

[4] F. Raviglione, M. Malinverno, C. Casetti, "A Flexible, Protocol-agnostic Latency Measurement Platform," IEEE VTC-Fall 2019.

[5] 5Growth, D3.2, Specification of ICT17 in-house deployment, April 2020.

[6] RFC2544, "Benchmarking Methodology for Network Interconnect Devices", available online at https://tools.ietf.org/html/rfc2544

[7] CENELEC, EN 50159 "Railway applications - Communication, signalling and processing systems - Safety-related communication in transmission systems", September 2010

[8] IEC, IEC62676 "Video Surveillance Systems for use in security application in terms of security, integrity, availability and latency", April 2014

[9] "Prometheus blackbox prober exporter", available online at https://github.com/prometheus/blackbox_exporter

[10] R. McMahon, Iperf 2.0.14, available online at https://sourceforge.net/projects/iperf2/

[11] D. Schweikert, fping 5.0, available online at https://fping.org/

[12] Paul Emmerich et al., "MoonGen: A Scriptable High-Speed Packet Generator", in proc. IMC'15, October 2017 ISBN: 978-1-4503-3848-6

[13] D. Stenberg, curl, available online at https://curl.se/

[14] ntop Inc, ntopng, available at https://www.ntop.org/products/traffic-analysis/ntop/

[15] Cilium Project, cilium, https://cilium.io/

[16] KeySight, P8800S UeSim, available online at https://www.keysight.com/en/pd-3066542-pn-P8800S/uesim-ue-emulation-ran-solutions?cc=BE&lc=dut

[17] 5G-TRANSFORMER project, "Final design and implementation report on service orchestration, federation and monitoring platform", available online at http://5g-transformer.eu/wp-content/uploads/2019/05/D4.3_Final_design_and_implementation_report_on_service_orchestration_federation_and_monitoring_platform_report.pdf

[18] SONATA project, "SONATA NFV Platform",available online at https://www.sonata-nfv.eu/content/agile-development-testing-and-orchestration-services-5g-virtualized-networks

[19] SONATA and 5GTANGO projects, "5GTANGO/SONATA Monitoring Infrastructure", available online at https://github.com/sonata-nfv/tng-monitor-infra/wiki

[20] SONATA and 5GTANGO projects, "Kubernetes cluster monitoring", available online at https://github.com/sonata-nfv/tng-monitor-infra/tree/master/K8s_mon

[21] SONATA project, "vrouter VNF Descriptor", available online at https://github.com/sonata-nfv/tng-industrial-pilot/blob/master/sdk-projects/tango-vrouter-svc/sources/vnf/vrouter/vrouter-vnfd.yml
[22] Apache Kafka, "A distributed streaming platform", available online at https://kafka.apache.org/
[23] Apache Pulsar, "Distributed pub-sub messaging system", available online at https://pulsar.apache.org/
[24] ETSI White Paper No.31, "NGSI-LD API: for Context Information Management", January 2019. ISBN: 979-10-92620-27-6