5G ExPerimentation Infrastructure hosting Cloud-nativE Netapps for public proTection and disaster RElief

Innovation Action – ICT-41-2020 - 5G PPP – 5G

Innovations for verticals with third party services

D4.6: Integration, Verification and Testing Report preliminary version

Delivery date: December 2022

Dissemination level: Public

Project Title:	**5G-EPICENTRE - 5G ExPerimentation Infrastructure hosting Cloud-nativE Netapps for public proTection and disaster RElief**
Duration:	**1 January 2021 – 31 December 2023**
Project URL	https://www.5gepicentre.eu/

This project has received funding from the European Union's Horizon 2020 Innovation Action programme under Grant Agreement No 101016521.

www.5gepicentre.eu

Document Information

Deliverable	D4.6: Integration, Verification and Testing Report preliminary version
Work Package	WP4: Platform integration and VNF development
Task(s)	T4.4: End-to-end platform integration activities T4.5: Lab testing, prototyping and validation
Type	Report
Dissemination Level	Public
Due Date	M24, December 31, 2022
Submission Date	M24, December 30, 2022
Document Lead	Konstantinos C. Apostolakis (FORTH)
Contributors	Luigi D'Addona (IST) Anna Maria Spagnolo (IST) Apostolis Siokis (IQU) Almudena Díaz Zayas (UMA) Jorge Márquez Ortega (UMA) Ankur Gupta (HHI) Kirsten Krüger (HHI) Holger Gäbler (HHI) Jorge Carapinha (ALB) Manuel Requena Esteso (CTTC) Fatemehsadat Tabatabaeimehr (CTTC) Hamzeh Khalili (CTTC) Josep Mangues-Bafalluy (CTTC) Yerasimos Yerasimou (EBOS) Sozos Karageorgiou (EBOS)
Internal Review	Pedro Tomás (ONE) Luigi D'Addona (IST)

Document history

Version	Date	Changes	Contributor(s)
V0.1	17/11/2022	Initial deliverable structure	Konstantinos Apostolakis (FORTH)
V0.2	06/12/2022	50% of the deliverable content, including test cases from partners reported as part of ongoing test case reporting in Task 4.5	Konstantinos Apostolakis (FORTH)
V0.3	09/12/2022	Inputs and clarifications from partners	Luigi D'Addona (IST) Anna Maria Spagnolo (IST) Almudena Díaz Zayas (UMA) Jorge Márquez Ortega (UMA)
V0.4	12/12/2022	80% of the deliverable content	Konstantinos Apostolakis (FORTH)
V0.5	14/12/2022	90% of the deliverable content	Apostolis Siokis (IQU) Kostas Ramantas (IQU) Almudena Díaz Zayas (UMA) Jorge Márquez Ortega (UMA)
V1.0	16/12/2022	Final version for internal review.	Konstantinos Apostolakis (FORTH) Apostolis Siokis (IQU) Ankur Gupta (HHI) Kirsten Krüger (HHI) Holger Gäbler (HHI) Jorge Carapinha (ALB)
V1.1	18/12/2022	Complementary input on VNF chain placement and cross-testbed Management and Orchestration tooling.	Manuel Requena Esteso (CTTC) Fatemehsadat Tabatabaeimehr (CTTC) Hamzeh Khalili (CTTC) Josep Mangues-Bafalluy (CTTC)
V1.2	19/12/2022	Complementary input on Northbound API/Configurator.	Yerasimos Yerasimou (EBOS) Sozos Karageorgiou (EBOS)
V1.3	20/12/2022	1st internal review version with comments and suggestions.	Luigi D'Addona (IST)
V1.4	23/12/2022	2nd internal review version with comments and suggestions.	Pedro Tomás (ONE)
V1.5	29/12/2022	Amendments based on internal review comments.	Manuel Requena Esteso (CTTC) Fatemehsadat Tabatabaeimehr (CTTC) Konstantinos Apostolakis (FORTH) Yerasimos Yerasimou (EBOS)

V2.0	30/12/2022	Final version for submission, after quality review (copyediting; proofreading; formatting).	Konstantinos Apostolakis (FORTH)

Project Partners

Logo	Partner	Country	Short name
	AIRBUS DS SLC	France	**ADS**
	NOVA TELECOMMUNICATIONS SINGLE MEMBER S.A.	Greece	**NOVA**
	Altice Labs SA	Portugal	**ALB**
	Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V.	Germany	**HHI**
	Foundation for Research and Technology Hellas	Greece	**FORTH**
	Universidad de Malaga	Spain	**UMA**
	Centre Tecnològic de Telecomunicacions de Cata-lunya	Spain	**CTTC**
	Istella SpA	Italy	**IST**
	One Source Consultoria Informatica LDA	Portugal	**ONE**
	Iquadrat Informatica SL	Spain	**IQU**
	Nemergent Solutions S.L.	Spain	**NEM**
	EBOS Technologies Limited	Cyprus	**EBOS**
	Athonet SRL	Italy	**ATH**
	RedZinc Services Limited	Ireland	**RZ**
	OptoPrecision GmbH	Germany	**OPTO**
	Youbiquo SRL	Italy	**YBQ**
	ORamaVR SA	Switzerland	**ORAMA**

List of abbreviations

Abbreviation	Definition
API	Application Programming Interface
CNF	Cloud-native Network Function
DB	Database
ExaaS	Experiments as a Service
GA	Grant Agreement
K8s	Kubernetes
KPI	Key Performance Indicator
MANO	Management and Orchestration
MQ(TT)	Message Queueing (Telemetry Transport)
N/A	Non Applicable
(C/V)NF	(Cloud-native/Virtual) Network Function
NFV	Network Functions Virtualization
NS	Network Service
QoE/S	Quality of Experience/Service
TAP	Test Automation Platform
TC	Test Case
UC	Use Case
(G)UI	(Graphical) User Interface
VNF	Virtual Network Function
VPN	Virtual Private Network
WP	Work Package

Executive summary

The present report summarizes the activities of the 5G-EPICENTRE Consortium with respect to system integration (Task 4.4) and testing (Task 4.5), undertaken for the period M1-M24. It follows up on prior documentation on both planning (D4.1) and implementation of the integrated prototype (D4.4), and constitutes a reporting of the framework for carrying out the integration and testing activities throughout the project lifetime. It will be succeeded by a final version due in M34 (D4.7), where the final system integration and testing activities throughout the remainder of the project will be appended to the present document.

The deliverable describes the means by which the integration roadmap set during the first year of the project has been followed, along with how aspects of system integration have been considered inside the scope of the individual technical Work Packages (WPs) for the period M1-M24. The document underlines interlinking within each WP itself, as well as with other WPs, and proceeds to elaborate on the means by which the integration targets have been met for the reported period. As such, the key technical outputs of each WP are listed, along with lessons learned from the integration activities, which include insights that will inform the continuing integration work over the final reporting period.

In addition, the present report follows up on the framework employed for the verification and validation of both individual system components and the overall system itself reported in D4.4, with a report on the test cases defined to verify functionalities of individual system components (and integrations thereof into smaller subsystems). Finally, the deliverable presents the guidelines and principles that will be employed toward assessing usability and acceptance of the final solution among the intended target stakeholders, by means of heuristic evaluation and user testing.

Table of Contents

List of Figures

List of Tables

1 Introduction

The present document reports on 5G-EPICENTRE partner activities in the context of Tasks T4.4: "*End-to-end platform integration activities*" and T4.5: "*Lab testing, prototyping and validation*", undertaken throughout the first two years of the project lifetime (M1-M24). Both activities are coordinated by the project Technical Manager. The deliverable constitutes a register of the technical integration and testing activities, receiving input primarily from the recent deliverable D4.4: "*5G-EPICENTRE experimentation facility preliminary version*", where the actual technical work and outputs of the project technical Work Packages (WPs) are described (as informed by the project elicited requirements and technical architecture reported in D1.3 "*Experimentation requirements and architecture specification preliminary version*"), but also, input from across all technical WPs (where actual activities are proactively taken to ensure a smooth integration procedure) will be reported.

As such, the deliverable constitutes a "snapshot" of the project implementation activities and roadmaps at the time of delivery. This report will be followed-up by an updated (and final) version due in M34 (D4.7: "*Integration, Verification and Testing Report final version*"), which will summarize all relevant activities in the development of the final 5G-EPICENTRE experimentation facility.

The rest of the deliverable is structured as follows:

- Section 2 constitutes the report on iterative systems integration (corresponding to Task 4.4), elaborating on the methods and approaches to carry out integration in the context of each WP. Each Section presents the internal and external interdependencies formed between the corresponding WP Tasks and other technical activities within the project work plan. In addition, the Section recounts the means to facilitate coordination, integration and communication between the multi-party project implementation staff.
- Section 3 compiles the project test documentation at the present point in time, in the form of test cases describing the step-by-step procedures to execute testing with the platform components available.
- Section 4 elaborates on the planned system usability evaluation, to ensure that user-facing processes and capabilities exposed to end users are sufficiently agile and efficient. User experience will be assessed through these mechanisms, and adaptations will be made to address poor User Interface (UI) experience.
- Section 5 concludes the deliverable with outlooks on remaining work in the context of the input Tasks.

The project overall test plan document, based on the IEEE 829 standard [1], is supplied as an appendix in Annex I.

1.1 Mapping of project's outputs

The purpose of this section is to map 5G-EPICENTRE Grant Agreement (GA) commitments, within the formal Task description, against the project's respective outputs and work performed.

Table 1: Adherence to 5G-EPICENTRE's GA Tasks' Descriptions

5G-EPICENTRE Task	Respective Document Chapters	Justification
T4.4: End-to-end platform integration activities "*This Task will deal with the integration of the modules developed in the technical WPs, according to the system architecture (T1.3) and use case requirements (T1.2)*".	Section 2 – Report on iterative system integration (M24)	This Section elaborates on the aspects of system integration that were considered within the scope of the corresponding WP, and reports on the definition and implementation of the WP-developed and exposed interfaces to support them.

"[…]. Integration will hence be addressed using vertical methods in order to have functional entities and horizontal approaches so as to facilitate any necessary customization of the platform, which will iteratively integrate components resulting from technical WPs to deliver incremental releases of the 5GEPICENTRE platform".		
T4.5: Lab testing, prototyping and validation *"The main aim of this task is to manage testing and validation of the separate components to be integrated into the final system"*. *"[…]. The Task will specify high-level tests for the components to be integrated in T4.4, as well as integration tests and method of validating the integrated system"*.	Section 3 – Report on 5G-EPICENTRE testing and validation (M24)	This Section presents the test cases developed for the various interdependent system components, and reports on their validation results.
	Annex I: 5G-EPICENTRE test plan	The Annex presents the test plan document in accordance with the IEEE Standard for Software and System Test Documentation. It outlines how testing is to be approached and managed, by elaborating on the scope, approach and guiding processes for all testing activities.
T4.5: Lab testing, prototyping and validation *"[…]. This task will provide feedback […] through instruments such as surveys, interviews, data collection and experiments. This includes the technical assessment of the reliability, robustness, resilience and technical performances of the integrated release, and will be targeted to the early detection and diagnosis of errors before the platform grows in complexity"*.	Section 4 – Report on usability and user experience assessment (M24)	This Section describes the methodology and guidelines of the planned usability evaluation, which will include the assessment of the system's functions from an (end) user's perspective, so as to feed into the designs of the user interfaces and improve their processes to lower the risk of end user disengagement due to increased system complexity.

2 Report on iterative system integration (M24)

The purpose of this Section is to explain how aspects of system integration have been considered inside the scope of the technical WPs for the period M1-M24. All activities reported in this Section are informed by the project "Integration plan and framework" delivered in D4.1 (M12). The material below is a report on activities to fulfil targets specified in D4.1, and where necessary, the content described in this document shall take precedence (for referential purposes) over the aforementioned document for the remainder of the integration timeline.

2.1 Adherence to the integration plan

In D4.1, the overall integration activity is addressed in two major phases, each addressing specific requirements from both a testbed and use case (UC) perspective. On the one hand, integration refers to the development of a usable 5G-EPICENTRE federated experimentation facility prototype, in accordance to the 5G-EPICENTRE overall functional architecture component diagram, as specified in D1.3. On the other, it refers to a number of preparatory activities that need to be undertaken in the context of both testbeds and use cases to fully exploit the different functional elements for the proof-of-concept demonstration goals set in the context of WP5. The relationship between the two activities is reflected in the main integration roadmap, reiterated from D4.1, in Figure 1.

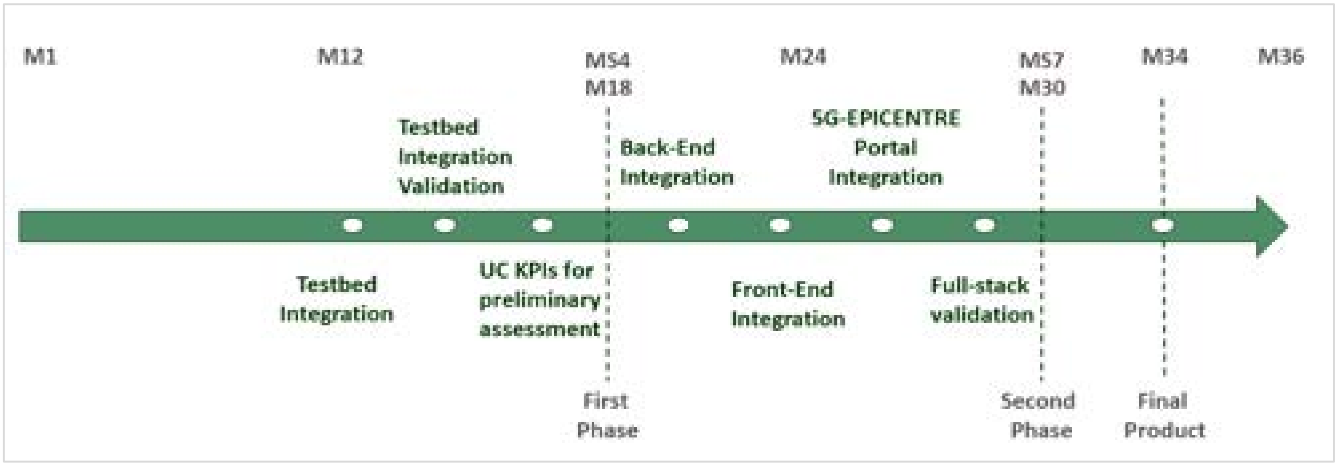

Figure 1: Integration roadmap (retrieved from D4.1).

2.1.1 System integration

System integration (which shall henceforth be referred to simply as *integration*) in the context of Task 4.4, refers to the process of interlinking different technological components toward facilitating a uniform system. Following up on the platform integration overview, reported in D4.4 (Section 2 in that document), the means by which integration is addressed is through the definition and implementation of well-defined interfaces (*i.e.*, with concretely established inputs and outputs). Such interfaces follow the consumer-producer paradigm, allowing a component to either expose or consume methods exposed by other components so that the exchange of data can be facilitated through pre-specified sets of parameters. Through the 5G-EPICENTRE integration approach that follows the microservices architecture paradigm, two components (services) are considered to be "integrated" when either one is able to consume the interfaces exposed by the other. Functional aspects of each individual service are not part of this integration activity, as these are addressed at the individual Task level in the technical WPs.

Toward achieving integration, three major objectives (D4.1) were pursued over the reported period, namely:

- **The identification of interdependencies among the different 5G-EPICENTRE components.** This exercise has been carried out over the course of the first 12 months of the project, and its outputs are reported in D4.1 (Section 5), where the intermediate architectural update to the 5G-EPICENTRE experimentation facility was delivered.
- **The definition of all components' interfaces.** The objective has entailed (i) the preliminary definition of component' interfaces supported over the different reference points of the 5G-EPICENTRE architectural framework; (ii) identification of the interfaces exposed by individual components; (iii) identification of the information elements exchanged over these interfaces; and (iv) definition of parameters sent and received over each interface. A preliminary listing of foreseen interfaces has been addressed in D4.1, with a more concrete elaboration and implementation (including updates to the original schemas) being addressed in D4.4, namely:
  - Front-end – Back-end intercommunication.
  - Front-end – Infrastructure intercommunication.
  - Back-end – Back-end intercommunication.
  - Back-end – Infrastructure intercommunication.
  - Back-end – Synchronization intercommunication (defined as an approach in D4.4, implementation to be elaborated in D4.3 due in M28).
  - Synchronization – Infrastructure intercommunication (defined as an approach in D4.4, implementation to be elaborated in D4.5 due in M30).

  As specified above, component interface development is a core task of the integration WP (Task 4.4), and remains an ongoing activity throughout the course of the project technical work plan duration. A final listing of components' interfaces will be delivered in D4.5 "5G-EPICENTRE experimentation facility final version".
- **The identification of the information flow for the 5G-EPICENTRE facility** to exploit available and emerging infrastructural developments for the purposes of testing and validation. The objective has been addressed through the first definition of experimentation procedures (D4.4), where information flows through the system were identified, namely:
  - **Downstream:** usage of front-end tools to setup and calibrate infrastructure components.
  - **Upstream:** infrastructure components generating data that is presented to the user via a UI.

  The activity remains ongoing work in progress, and the final definition of how the different components will interact amongst themselves, will be reported in D4.5.

2.1.2 UC deployment-related activities

To prepare for the first party experimentation, it has been necessary to facilitate the communication between the UC and the testbed owners to agree on specific deployment plans and roadmaps for each case, as well as to actually deploy the UC vertical application components (*i.e.*, components outside the 5G-EPICENTRE experimentation facility, and maintained externally to the project WPs, with the exception of their containerization) onto the 5G testbed infrastructures. To address these necessities, D4.1 has foreseen both (i) the specification of a roadmap for all testbeds to deliver on cloud-native virtualisation and compliance to the 5G-EPICENTRE architecture; and (ii) the specification of a roadmap for each UC vertical application to implement its Network Application chains and components on top of the offered virtualized infrastructures. The activities are foreseen as part of WP2, as Task 2.1 delivers the Kubernetes (K8s) infrastructure and Task 2.2 is in charge of the deployment of the use cases on top of this infrastructure. Task 4.4 is monitoring the progress of testbed integration, as a side activity of monitoring the main integration roadmap, while individual developments regarding to the UC NFs are monitored as part of Task 4.2.

Much of the integration focus during the elapsed period has been placed on preparing the infrastructures (testbeds) for hosting the ensuing 5G-EPICENTRE components (with support for the entire experimentation facility being foreseen as a milestone for all testbeds to be achieved in M30). Table 2 – Table 5 informs on the status of the testbed milestones, as listed on the roadmaps defined in D4.1.

Table 2: Aveiro testbed milestone roadmap

Month	Milestone	Status (Achieved/Delayed/Not due)	Date of achievement	New date of achievement
M12	Initial K8s deployment	Achieved	01/02/2022	M13
M14	Initial deployment of UCs	Achieved	15/02/2022	-
M15	RabbitMQ[1] integration	Achieved	31/04/2022	-
M18	Message format integrated	Achieved	31/08/2022	M20
M24	Intermediate deployment of UCs	Not due	-	M25
M26	Instantiation of 5G Core in K8s	Not due	-	-
M26	Intermediate validation of UCs	Not due	-	-
M28	Analytics module deployment and integration	Not due	-	-
M30	5G-EPICENTRE experimentation facility integration	Not due	-	-
M30	Final deployment of UCs	Not due	-	-
M34	Final validation of UCs	Not due	-	-

Table 3: Berlin testbed milestone roadmap

Month	Milestone	Status (Achieved/Delayed/Not due)	Date of achievement	New date of achievement
M13	Initial K8s deployment for UCs	Achieved	28/02/2022	M14
M15	Initial deployment of UCs	Achieved	29/03/2022	-
M16	First selection of measurement methods	Achieved	31/04/2022	-
M17	Rabbit MQ integration	Achieved	31/05/2022	-

[1] https://www.rabbitmq.com/

M20	Message format integrated	Achieved	30/09/2022	-
M24	Intermediate deployment of UCs	Achieved	31/12/2022	-
M26	Intermediate validation of UCs	Not due	-	-
M28	Provision of measurement data for the verification of KPIs	Not due	-	-
M30	5G-EPICENTRE experimentation facility integration	Not due	-	-
M30	Final deployment of UCs	Not due	-	-
M34	Final validation of UCs	Not due	-	-

Table 4: Málaga testbed milestone roadmap

Month	Milestone	Status (Achieved/Delayed/Not due)	Date of achievement	New date of achievement
M6	Initial K8s deployment	Achieved	30/05/2021	-
M7	Initial deployment of UCs	Achieved	30/06/2021	-
M12	Measurement’s aggregation	Achieved	01/04/2022	M16
M12	Message format integrated	Achieved	30/01/2022	M13
M12	Integration between use-cases and selected testbed clusters concluded	Achieved	30/04/2022	M16
M14	RabbitMQ integration	Achieved	30/01/2022	-
M18	Analytics module deployment and integration (1st)	Achieved	01/06/2022	-
M20	Intermediate deployment and validation of UCs	Achieved	30/06/2022	M18
M24	Measurement’s aggregation (1st)	Achieved	14/12/2022	M24
M24	Analytics module deployment and integration (2nd)	Achieved	14/12/2022	M24
M30	Measurement’s aggregation (2nd)	Not due	-	-

M30	Analytics module deployment and integration (3rd)	Not due	-	-
M30	5G-EPICENTRE experimentation facility integration	Not due	-	-
M30	Final deployment of UCs	Not due	-	-
M34	Final validation of UCs	Not due	-	-

Table 5: Barcelona testbed milestone roadmap

Month	Milestone	Status (Achieved/Delayed/Not due)	Date of achievement	New date of achievement
M13	Initial K8s deployment for UCs	Achieved	31/01/2022	-
M15	Coordination and initial deployment of UCs	Achieved	30/11/2022	M23
M16	Selection of measurement methods	Achieved	30/04/2022	-
M17	RabbitMQ integration	Achieved	08/06/2022	M18
M20	Integrated Message format	Achieved	30/11/2022	M24
M22	Instantiation of 5GC in K8s	Achieved	15/12/2022	M24
M26	Intermediate deployment and validation of UCs	Not due	-	-
M30	5G-EPICENTRE experimentation facility integration	Not due	-	-
M30	Final deployment of UCs	Not due	-	-
M34	Final validation of UCs	Not due	-	-

2.2 Integration from the perspective of the Work Packages

As previously specified, integration within the context of an individual WP refers to the definition and implementation of interfaces and communication exchange parameters to facilitate the intercommunication among the WP internally developed modules, as well as between that WP and the other 5G-EPICENTRE components/services that are being developed in the context of other WP activities.

The following sub-Sections present an overview of system-level integration activities, addressed both within and across the technical WPs. More specifically, we present an overview of identified interdependencies of all WP-specific components, corresponding to functional architectural blocks in D1.3, highlighting how they might inter-link with other components produced in other Tasks of the same WP, as well as with components external to its WP. We further underline lessons learned from the integration processes over the course of the first two years, distilling insights into higher-level takeaways that will further guide the integration activities throughout the remainder of the project (M25-M36).

2.2.1 Integration activity within WP2

The key technical outputs of WP2 as a whole for the elapsed period (M1-M24) include (but, are not limited to) the deployment of Kubernetes (K8s) in all the testbeds partaking in the 5G-EPICENTRE federation, alongside the development of key components of the 5G-EPICENTRE architecture (D1.3) including:

- The **Experiment Coordinator** is in charge of the communication between the 5G-EPICENTRE Portal and the different testbeds, where the experiments will be executed. The Experiment Coordinator can be interacted with using a northbound interface exposed toward the Experiment Composer (Portal) and a southbound interface exposed toward the testbed. For the latter, several endpoints have been defined for testing and validation purposes (see Section 3):
  - **`/experiment/run (POST)`**: endpoint for creating and queueing a new experiment execution.
  - **`/execution/{id}/logs (GET)`**: endpoint for retrieving log messages generated by an experiment execution.
  - **`/execution/{id}/ results (GET)`**: endpoint for retrieving logs and files generated by the experiment execution.
- The **5G Traffic Simulator Manager** manages and monitors the network traffic measurements to be generated by the various probes/agents deployed in different parts of the testbeds. It exposes the Traffic Simulator Manager interface consumed by the authorized consumer functional blocks (*e.g.*, iPerf[2] agents) on a testbed, through an interface that is propagated by the REST component to distribute the configuration commands to the proper probes at each site. Because of the need for the iPerf agents to consume the payload produced by the 5G Traffic Simulator Manager, an asynchronous mode of communication is preferred. Endpoints for this connection have been defined for testing and validation purposes (see Section 3):
  - **`/start (POST)`**: endpoint for Executing an iPerf command (*i.e.*, with iPerf parameters[3]) on the iPerf agent. This is intended to be the normal way to start a server/client on a remote agent.
  - **`/stop (POST)`**: endpoint for stopping an iPerf server running on a remote agent, identified by a dictionary with keyword "agent_id" and value "ID".
  - **`/retrieve?agent_id="" (GET, POST)`**: This endpoint returns the last result produced by the specified agent. If it does not exist, it returns a JSON with the corresponding error.
  - **`/add_iperf_agent (GET, POST)`**: This endpoint displays a minimal Graphical User Interface (GUI) for a user to enter a new agent.
  - **`/remove_iperf_agent`**: This endpoint displays a minimal GUI for a user to delete an agent.
- The **Publisher** component (novel component not described in D1.3) is a Python REST Application Programming Interface (API) that listens to the application RabbitMQ Message Queueing Telemetry Transport (MQTT) queue and completes experiment metadata in order to send it with a proper format to the Analytics engine module through the Publisher's MQTT queue. It has several interfaces that allow it to: (i) collect monitoring data from Prometheus[4]; (ii) configure appropriate metadata for different experiments; and (iii) publish data coming from a Test Automation Platform (TAP) adaptor. This component is meant to communicate exclusively with the Experiment Coordinator component. The Publisher can be

[2] https://iperf.fr/
[3] https://iperf.fr/iperf-doc.php#3doc
[4] https://prometheus.io/

interacted with using the following endpoints, defined for testing and validation purposes (see Section 3):

 - **`/add_experiment (POST)`**: Sends a JSON containing metadata for the publisher to store. When the corresponding experiment results are read from the MQTT they will be completed with this metadata.
 - **`/remove_experiment (POST)`**: Removes experiment metadata from the list.

- Updating on D1.3, the **Virtual Network Function (VNF) chain placement** approach is composed of two modules: a Manager, developed at the cross-testbed Management & Orchestration (MANO) level; and an Algorithm developed at the infrastructure layer. Each module has several interfaces, as described:
  - i) the Manager interacts with the cross-testbed MANO API server to receive Helm[5] chart (deployment) and after the decision on placement, returns the result for binding to a cluster.
  - ii) the Algorithm interacts with the Kubernetes API server to receive deployment; and after the decision on placement, returns the result for binding step.
  - iii) the Manager subscribes to topic that is published by Publisher.

  Hence, the initial approach for optimal VNF chain placement is considered at two levels:
  - At Karmada level (see Section 2.2.3), where the best possible deployments in either one or another K8s cluster is decided.
  - At K8s level, where the suitable node is identified.

  The contemplated approaches have warranted an architectural update, where the approach no longer falls under the responsibility of the Experiment Player subsystem. Hence, work on T4.3 and T2.3 has been very closely aligned.

- The (first version of) the **Analytics engine** module, which is comprised of several interdependent components and monitors experimental application conditions in order to identify anomalies and predict Key Performance Indicators (KPIs). All interfaces of the Analytics engine are asynchronous, using the RabbitMQ middleware for publishing messages and subscribing to other components' MQTT queues (*e.g.*, the Publisher's).

WP2 is fundamental to the integration procedures in 5G-EPICENTRE. As can be seen from the components developed in this WP, the Tasks responsible for the development of those components are interlinked (*i.e.*, requiring inputs from one another), while other WP Tasks require inputs from Tasks in this WP. More specifically:

Internal WP relationships: In order to take a decision on VNF chain placement among testbeds, both VNF chain placement components developed in Task 2.3 subscribe to metrics from Publisher that is developed in T2.4. Task 2.5 (developing the Analytics engine components) likewise receives input measurements coming from the vertical applications and from the infrastructure via the Publisher module, hence also forming a link with Task 2.4. Furthermore, with respect to the overall integration roadmap (Figure 1), as has previously been elaborated, Task 2.1 delivers the cloud-nativized cores of the testbeds, being responsible for the support of K8s in the respective infrastructures; while Task 2.2 is in charge of the deployment of the use cases on top of these infrastructures.

External WP relationships: Task 2.4 is linked with Task 3.1 for the definition of the interface between the platform Portal and the Experiment Coordinator. Also, the output of the Analytics engine (Task 2.5) is forwarded to Task 3.2 (Insights Tool) for visualization. Moreover, the VNF chain placement Manager receives some essential templates from cross-testbed federation, occurring in T4.3.

Furthermore, input from WP2 has contributed to deliverables D4.1, D4.4 and now D4.6, with the definition of the interfaces of the components developed in WP2 and the specification of the general workflow of the experimentation framework.

[5] https://helm.sh/

2.2.2 Integration activity within WP3

The purpose of WP3 has been fixed on the development of user-facing solutions and applications intending to allow the PPDR vertical service provider to interact with the 5G-EPICENTRE platform components. Pragmatic outputs of this WP include the development of two User Interface (UI) enabling the connection between the UCs vertical applications and the 5G-EPICENTRE infrastructure, namely the Northbound API/configurator and the 5G-EPICENTRE Portal.

In the context of requiring inputs from one another, the Tasks in WP3 are distinctly independent, each provisioning an output related to the overall integrated platform, with its own individual interface to expose, or consume the services needed for fulfilling its role within the experimentation facility.

Following the microservices architectural style, T3.2 and T3.3 can be viewed as weakly associated in the sense of each Task developing part of the integrated functionality envisioned by the 5G-EPICENTRE Portal service, yet each corresponding to a loosely coupled online service, which does not affect either operation or performance of the other. Hence, each can be upgraded (or replaced) by a more novel version, without affecting the functionality of the other.

External WP relationships: Sitting atop the 5G-EPICENTRE architectural stack, the three Tasks are interdependent from other components developed in the context of WP2 and WP4. More specifically:

- Task T3.1 develops the **Northbound API/configurator** (D3.1), which provides a middleware between the UCs vertical applications and the 5G-EPICENTRE infrastructure. In order to implement this functionality, the Task interlinks with both 5G-EPICENTRE testbed and UC owners to determine which are the network parameters that are indeed configurable and can be requested by the UC vertical apps. Hence, all former communications are implemented in the context of WP2, where the testbed virtualized infrastructures are implemented. Communications between the UC owners (T4.2 participants) and component developers (T3.1 leader) was established for determining the parameters that each UC owner requests to have access to through the Northbound API/configurator.
- Task 3.2 core output (D3.2, expected in M30) corresponds to the final destination point of the 5G-EPICENTRE integrated experimentation facility upstream information flow, *i.e.*, the presentation of human-readable information from data generated at the 5G-EPICENTRE infrastructure level, which constitutes part of the integrated functionality envisioned by the 5G-EPICENTRE Portal service. In terms of external relations to WP3, this Task receives input from the Analytics Engine components developed in the context of Task 2.5.
- Task 3.3 has implemented the **experiment planning interface** of the **5G-EPICENTRE Portal**, a web-based, user-friendly interface for defining and requesting onboarding of an experiment to one or more of the 5G-EPICENTRE federated platforms. In terms of the 5G-EPICENTRE integrated experimentation facility information flows, the Task output (D3.3) corresponds to the entry point of the downstream information flow, and provides input to the Experiment Coordinator module developed in the context of Task 2.4.

2.2.3 Integration activity within WP4

WP4 is central to the integration processes, as it is the WP where all integration activities are planned, monitored and coordinated. Three of the WP Tasks (Task 4.1, Task 4.4 and Task 4.5) address integration horizontally, and are responsible for laying out plans and procedures for the integration, actually carrying out those plans, and verifying the whole integration process. As such, these Tasks interact with all technical Tasks across the other two technical WPs.

The remaining two Tasks offer implementation of key 5G-EPICENTRE experimentation facility components, including:

- The centralized **Network Service Repository** (Task 4.2[6]), which realizes a private Helm chart repository developed in JFrog[7], and which stores the catalogue of Helm chart packages (artifacts) that the K8s orchestrator can download. JFrog provides REST APIs, implemented for the connection to/from the private repo, allowing a file to be uploaded to the private repo by using a single curl command. In the context of this Task, an Open API server has been developed, that communicates with the JFrog private Helm repository, supporting retrieval, updating and deletion of Helm charts.
- Task 4.3 implements the prototype **Cross-testbed MANO tooling** to enable cross-orchestration of distributed resources on K8s clusters executing containerised NFs over the different testbeds. The approach is based on the Karmada K8s management system that enables cloud-native applications to run across multiple K8s clusters with no changes to the application. The selection of this tool was based on a thorough analysis and mapping of Karmada components to 5G-EPICENTRE architecture (elaborated in D4.4).

Internal WP relationships: As previously stated, Task 4.1 delivered the overall integration plan that has informed all integration activities throughout the course of the reporting period, while Task 4.4 has been responsible for facilitating integration of the disparate 5G-EPICENTRE facility components. Task 4.5 is responsible for organizing and executing the testing and validation procedures (both at the unit level, as well as at the integration level). With respect to development, a particular link is created between Task 4.3 and Task 4.2, as the cross-testbed federation approach using Karmada makes use of the 5G-EPICENTRE private Helm chart repository for storage of the Network Application chart packages.

External WP relationships: As already discussed, WP4 is a focal point of the entire 5G-EPICENTRE implementation activity, with each WP referring to Task 4.4 regarding the integration of system components. Component-wise, Task 4.3 is linked to Task 2.3 in the analysis of Karmada scheduling and deployment of complex services, where (as specified in the mappings of Karmada to the cross-testbed MANO API defined in D4.4) Karmada's scheduler is being explored for offering the functionalities assigned to the VNF Chain Placement Manager component in the 5G-EPICENTRE architectural stack. The scheduler designed and implemented in Task 2.3 will be integrated with the Karmada's scheduler. The cross-testbed MANO tool implemented in Task 4.3 also publishes metrics related to the federation of the Kubernetes clusters to the Publisher developed in Task 2.4.

2.3 Cross-partner collaboration with respect to integration

To follow-up on the integration points described therein, issues related to integration between Tasks of the same WP, or the integration of Task outputs with other WPs, have been addressed by coordinating the corresponding WP team through a dedicated mailing list, while a string of dedicated, periodic WP calls have been set up (see Table 6), and held over regular intervals throughout each WP implementation activity. Meeting minutes are summarised and kept online via a dedicated space in the project's Confluence collaboration platform.

Furthermore, specific meetings have been held with select members of the Consortium (*e.g.*, Testbed and UC owners) in order to clarify details about integration points and protocols.

[6] It is worth noting that Task 4.2 is responsible for the development and adaptation of all VNFs that will form components of the vertical applications for the first party experiments defined in the project (D1.1), which includes the identification of those vertical application's services and interfaces (between themselves, as well as between the vertical application and the 5G Core network services exposed by the testbeds, or other services provided by the 5G-EPICENTRE platform). This aspect of the work carried out in Task 4.2 is considered external to the iterative system integration activities, as defined in the present document, and is foreseen within the reporting responsibility of D4.2 "Network functions implementation" (M26).

[7] https://jfrog.com/

Table 6: Regular periodic meeting details held in the context of project implementation & integration.

Meeting	When	Bridge provided by
WP2/WP5 monthly	Occurs every 2 weeks on Wednesday until 30/6/2023 from 11:00 AM to 12:00 PM (CET). As of M14, it is held in conjunction with WP5.	Zoom (hosted by UMA).
WP3 monthly	Occurs the second Friday of every month until 30/6/2023 from 14:00 PM to 15:00 PM (CET).	Zoom (hosted by FORTH).
WP4 monthly	Occurs the first Tuesday of every month until 31/12/2023 from 15:00 PM to 16:00 PM (CET)	Zoom (hosted by FORTH).

Other collaboration channels include the project ***Confluence*** space (a place where integration-related documentation, templates and meeting minutes are stored and made accessible to all partners), and the project ***Mattermost*** channels, where bilateral discussions have been held to streamline integration discussions in a more orderly communication form.

2.4 Lessons learned

Integration in 5G-EPICENTRE has been a daunting and risk-prone task for its involvement in a wide variety of technologies, systems, communication and data exchange needs, not to mention the inherent heterogeneity of the testbed platforms partaking in the federation. The federation itself has been a concept that has recently identified Karmada as the solution best fitting the foreseen functions of the cross testbed MANO block in the 5G-EPICENTRE architecture. Proliferation of technologies and their resulting adoption and use are to be expected in the context of large-scale, long-term and multi-partner projects such as 5G-EPICENTRE, and demonstrate that Consortium members remain up-to-date with respect to latest technological trends. Therefore, successful integration of such technologies, and the adoption of new methods and tools by other Consortium members outside of the introducing partner further highlight the Consortium's capacity to (re-)adapt the project technological developments to align with such breakthroughs, with minimal effort and re-structuring of existing solutions.

To ensure this, it has been necessary for the partners to regularly meet and discuss integration aspects and evolving technological requirements and new needs arising from both internal (*e.g.*, elaboration of Karmada as the cross-testbed MANO solution) and external factors (*e.g.*, the recent common definition given to Network Applications across the 5G-PPP projects in the latest 5G-PPP Software Network Working Group White Paper "NetApp: Opening up 5G and beyond networks" [2]). Through carefully selected collaboration tools and efficient project management, this has been made possible within 5G-EPICENTRE throughout the first two years of the project, which introduced the added complexity of the COVID-19 pandemic and restrictions that were put in place for partners travelling and meeting face-to-face.

A key takeaway that partners can apply in future potential activities (both internally and externally to 5G-EPICENTRE) lies with the careful and individual planning that has characterized each testbed's evolution toward enabling a stable 5G-EPICENTRE experimentation facility integration, at the latest in M30. In cases where a testbed owner has also been the owner of a component (or collection of components) that require integration with other platforms, that partner has a distinct head start in the integration of their own components, thus acting as a "champion" for others to follow (cases of this happenstance include UMA being ahead of other testbed owners in terms of the K8s deployment – as reflected in the testbed roadmaps – and CTTC being the beneficiary in charge of the Karmada deployment and testing at the Barcelona testbed). This "champion" partner is responsible for capturing knowledge with respect to issues and difficulties faced during the implementation, integration and deployment of their technologies, and ensuring the transfer of that knowledge to the interested internal organisations. Such planning further allows testbed owners with a head start to also champion other aspects of system integration for the other partners (such has been the case for UMA and IST in trailblazing the deployment of the Analytics engine components on the K8s-based testbed).

3 Report on 5G-EPICENTRE testing and validation (M24)

Testing and validation activities run in parallel to the iterative module integration processes with the aim to support them, by validating component functionality at both the unit and (sub)system level. In the context of Task 4.5, testing focuses at establishing a feedback loop with the individual Task in charge of component / service / module development, to ensure that testing results are used towards refining technical development and introducing enhancements to the overall 5G-EPICENTRE infrastructure.

Activities carried out in the context of Task 4.5 receive input from the 5G-EPICENTRE testing framework elaborated in D4.4 (Section 5), which outlines the provisions in accordance to which the automated and manual testing activities take place. At the core of the testing lies the test documentation, which in 5G-EPICENTRE has produced both a test plan document (see Annex I) and test cases documentation, covering how tests for individual components will be carried out to validate both components and their integration in accordance to the test plan guidelines. Test cases were developed by the Partner technical representatives, that each project Beneficiary involved in technical implementation and integration activities appointed. The Test Case (TC) documentation template was presented in the preceding deliverable D4.4, which was submitted in M18. For the sake of completeness of the present report, it is briefly summarised in Table 7.

Table 7: Testable features plan template distributed to 5G-EPICENTRE technical partners.

Test plan reference	Description	
Test Item	Component name	
Features to be tested	Listing of features (component functions) to be tested, with indication of the level of risk for each one (H: High; M: Medium; L: Low).	
Features NOT to be tested	Listing of features (component functions) that will not be tested as part of this component's test plan, along with reasons for omitting the test.	
Approach	Description of how the test will be carried out.	
Needs	Listing of conditions that need to be met prior to the test in order for it to be carried out successfully.	
Test case(s) (As many as needed for all test cases of a single test item)	**TC[No] – Test case descriptive title** *Test case short description, explaining to the tester what they are testing.*	
	Assumption(s)/ Precondition(s)	Listing of all conditions that should be met prior to the test being carried out.
	Test steps	1. Step-by-step description of the test's execution. 2. …
	Expected result(s)	Expected behaviour of the test item after all test steps have concluded.
Item pass/fail criteria	**PASS**	Define the condition indicating successful completion of the test plan.
	FAIL	Describe unacceptable failure conditions for the test item.

In accordance to the formal Deliverable and Task description, this Section summarises a report on system-level testing, along with its results. The paragraphs below summarise the test scenarios and test cases for the 5G-EPICENTRE technical components, corresponding to the technological maturity expected as of M24.

3.1 Test items

3.1.1 Experiment Planning Interface front-end and back-end components

The following Tables (Table 8 – Table 10) present indicative test plan features for testing the Experiment Planning Interface module of the 5G-EPICENTRE Portal, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 8: Experiment Planning Interface testable features plan (M24).

Test plan reference	Description
Test Item	Experiment Planning Interface
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• As an anonymous user, log into the 5G-EPICENTRE Portal [H]. • As an experimenter, initiate and carry out a new experiment request [H]. • As an experimenter, check notification status after an experiment request has been updated by a testbed owner [L]. • As an experimenter, submit an amended version of an experiment [M]. • As a function developer, request the onboarding of a new Network Application package via the Portal [H]. • As a testbed owner, receive and view a new experiment request [L]. • As a testbed owner, receive and view the contents of a Network Application package onboarding request [H]. • As a testbed owner, notify on the status of an experiment request. [L] • As a testbed owner, notify on the status of a Network Application package onboarding request. [L].
Features NOT to be tested	• Usage of the Experiment Planning Interface on mobile/smart device web browser environment.
Approach	The 5G-EPICENTRE Portal shall be tested only on a desktop web browser.
Needs	A private instance of the Experiment Planning Interface back-end must be set up and running for testing to commence.

Table 9: Experiment Planning Interface test cases (M24).

TC-01 – As an anonymous user, log into the 5G-EPICENTRE Portal. *A user who has previously registered with the 5G-EPICENTRE Portal, should be able to successfully log in.*	
Assumption(s)/Precondition(s)	The user should have registered on the 5G-EPICENTRE Portal with a username and password. The tester uses proper credentials and ensures that they are typed in correctly.

Test steps	1. Load the 5G-EPICENTRE Portal index page. 2. Enter the e-mail address in the designated input field. 3. Enter the password associated to the e-mail address in the corresponding input field. 4. Click on the bright orange 'Sign in' button.
Expected result(s)	The Portal redirects the user to the main Dashboard page (corresponding to the user role associated to the account used to log in).
TC-02 – As an experimenter, initiate and carry out a new experiment request. *This test case illustrates the steps followed by an Experimenter to request the scheduling of an experiment.*	
Assumption(s)/Precondition(s)	The experimenter account should have been registered on the 5G-EPICENTRE Portal with a username and password.
Test steps	1. Log-in (TC-01) using the function developer account credentials. 2. Navigate to the 'My Experiments' page from the left-hand side menu. 3. Click on the bright orange 'Create a new experiment' button displayed at the top right corner of the screen. 4. [Optional] Answer the questions in the network slice configuration "Wizard", and click on the bright orange "Next step" button. 5. [Optional] Opt to skip the wizard and select slice configuration manually. Click on the bright orange "Next step" button. 6. Provide input for all fields. Click on the bright orange "Next step" button. 7. Provide input for all fields in the 'Experiment KPI 1 space ('Parameter to be measured', 'Short description', 'Values'). Click on the bright orange "Next step" button. 8. Review information in the 'Confirmation' tab and click on the bright orange 'Publish' button.
Expected result(s)	The Portal redirects the user to the 'My Experiment' page, where the new experiment request should appear inside the 'Pending experiment requests' layout. A request is forwarded to the designated Testbed Owner(s) for each site that was specified in the experiment form (check with TC-06).
TC-03 – As an experimenter, check notification status after an experiment request has been updated by a testbed owner. *A notification will be generated at Experimenter's dashboard, notifying on the response by a testbed owner regarding one of the pending experiment requests.*	
Assumption(s)/Precondition(s)	The experimenter account should have been registered on the 5G-EPICENTRE Portal with a username and password. An experiment request must have been submitted (TC-02). A testbed owner must have decided on an experiment request (TC-08)

Test steps	1. After log-in (TC-01), navigate to the 'My Resources' page from the left-hand side menu. 2. Observe the bright orange 'Notifications' button to the left of the user avatar image indicating 1 new notification. 3. Click on the 'Notifications' button and read notification status. 4. Click on the notification text.
Expected result(s)	The Portal redirects the user to the experiment page, where the status of the experiment has been updated, and the list of requested revisions has been filled in by the testbed owner.
TC-04 – As an experimenter, submit an amended version of an experiment. *This test case illustrates the steps followed by an Experimenter to amend their request for the scheduling of an experiment**.***	
Assumption(s)/Precondition(s)	The experimenter account should have been registered on the 5G-EPI-CENTRE Portal with a username and password (TC-01). An experiment request must have been submitted (TC-02). A testbed owner must have notified an experiment request (TC-08)
Test steps	1. After log- in (TC-01), check notification status after an experiment request (TC-03). 2. Click on the bright orange pencil-shaped 'Edit' button (appearing next to the 'Submit' button). 3. Execute the steps followed by an Experimenter to request the scheduling of an experiment (TC-02).
Expected result(s)	The Portal redirects the user to the 'My Experiment' page, where the new experiment request should appear inside the 'Pending experiment re-quests' layout. A request is forwarded to the designated Testbed Owner(s) for each site that was specified in the experiment form (check with TC-06).
TC-05 – As a function developer, request the onboarding of a new Network Application package via the Portal. *The test case illustrates the steps followed by a Function Developer to request onboarding of an NF resource**.***	
Assumption(s)/Precondition(s)	The function developer account should have been registered on the 5G-EPICENTRE Portal with a username and password.
Test steps	1. Log-in (TC-01) using the function developer account credentials. 2. Navigate to the 'My Resources' page from the left-hand side menu. 3. Click on the bright orange 'Onboard a VNF' button displayed next to the VNFs list header space. 4. In the ensuing the VNF Onboarding page, enter data into the 'Ti-tle' and 'Category' text input fields.

	5. Select one or more testbed platforms to request onboarding to via the radio buttons provided. 6. Drag and drop, or Click on 'Browse' in the 'Upload the VNF package' space on the left, to upload a compressed archive of files (*e.g.*, .zip file), and wait until the file appears inside the yellow-tinted space. 7. Click on the bright orange 'Submit' button.
Expected result(s)	The Portal redirects the user to the main 'My Resources' page, where the new VNF onboarding request should appear inside the 'Network Functions' layout. A request is forwarded to the designated Testbed Owner(s) for each site that was specified in the onboarding form (check with TC-06).
TC-06 – As a testbed owner, receive and view a new experiment request. *The test case illustrates the steps followed by a Testbed Owner to view a newly arrived experiment request.*	
Assumption(s)/Precondition(s)	The testbed owner account should have been registered on the 5G-EPICENTRE Portal with a username and password. An experiment request must have been submitted (TC-02).
Test steps	1. Log-in (TC-01) using the testbed owner account credentials. 2. Observe the bright orange 'Notifications' button to the left on the user avatar image indicating 1 new notification. 3. Click on the 'Notifications' button and read notification status. 4. [Option 1] Click on the notification text. 5. [Option 2] Click on the bright orange 'OK' button to exit the notifications space. Navigate to the 'Experiment Requests' page from the left-hand side menu and locate the new experiment request (the status of the experiment is indicated as 'Submitted'). Click on the title of that experiment.
Expected result(s)	The Portal redirects the user to the Experiment Overview page.
TC-07 – As a testbed owner, receive and view the contents of a Network Application package onboarding request. *The test case illustrates the steps followed by a Testbed Owner to view a newly arrived resource onboarding request.*	
Assumption(s)/Precondition(s)	The testbed owner account should have been registered on the 5G-EPICENTRE Portal with a username and password. A resource onboarding request must have been submitted (TC-05).
Test steps	1. Log-in (TC-01) using the testbed owner account credentials. 2. Observe the bright orange 'Notifications' button to the left on the user avatar image indicating 1 new notification. 3. Click on the 'Notifications' button and read notification status.

	4. Click on the bright orange 'OK' button to exit the notifications space. 5. Navigate to the 'Resource Requests' page from the left-hand side menu. 6. Locate the new Network Application package request and click on its title. 7. Download the compressed archive from the 'VNF Package' space.
Expected result(s)	A file download is initiated in the user's browser where the contents of the file are received.
TC-08 – As a testbed owner, notify on the status of an experiment request. *The test case illustrates the steps followed by a Testbed Owner to notify on the decision for an experiment request.*	
Assumption(s)/Precondition(s)	The testbed owner account should have been registered on the 5G-EPI-CENTRE Portal with a username and password. An experiment request must have been submitted (TC-02).
Test steps	1. Locate the experiment in the 'Experiment Requests' page and review its contents (TC-06). 2. Re-navigate to the 'Experiment Requests' page. 3. Click on the orange pencil-shaped 'Edit' button in the experiment request record (located on the right). 4. Select a 'Status' from the drop down menu. 5. [Optional] Enter some text in the 'Comments' section. 6. Click on the bright orange 'Submit' button.
Expected result(s)	The Portal redirects the user to the 'Experiment Requests' page, where the status of the experiment is indicated as 'Notified'. A request is forwarded to the Experimenter to update the status of the experiment in their dashboard (check with TC-03).
TC-09 – As a testbed owner, notify on the status of a Network Application package onboarding request. *The test case illustrates the steps followed by a Testbed Owner to notify on the decision for a Network Application package onboarding request.*	
Assumption(s)/Precondition(s)	The testbed owner account should have been registered on the 5G-EPI-CENTRE Portal with a username and password. A resource onboarding request must have been submitted (TC-05).
Test steps	1. Locate the experiment on the 'Experiment Requests' page and review its contents (TC-06). 2. Re-navigate to the 'Experiment Requests' page. 3. Click on the orange pencil-shaped 'Edit' button in the experiment request record (located on the right). 4. Select a 'Status' from the drop down menu. 5. [Optional] Enter some text in the 'Comments' section.

	6. Click on the bright orange 'Submit' button.
Expected result(s)	The Portal redirects the user to the 'Resources Requests' page, where the status of the experiment is indicated as 'Notified'. A request is forwarded to the Function Developer to update the status of the experiment in their dashboard (check with TC-03).

Table 10: Experiment Planning Interface pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• Any of the cases fail to produce the expected result.

3.1.2 Insights Tool

The following Tables (Table 11 – Table 13) present indicative test plan features for testing the Insights Tool of the 5G-EPICENTRE Portal, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 11: Insights Tool testable features plan (M24).

Test plan reference	Description
Test Item	Insights Tool
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• Connection (integration) of the tool to the Analytics engine for reception of metrics [H].
Features NOT to be tested	• Translation of statistical information originating at the Analytics Engine components into boxplot chart (feature is tested during development with mock data). • Translation of use case-specific KPI data analysis originating at the Analytics Engine components into stream graph (feature is tested during development with mock data). • Translation of location data of the user equipment into visual output on the 3D map.
Approach	To access the metrics, the proper RabbitMQ configuration settings must be used. The Analytics engine endpoint to be deployed as a pod in the Malaga testbed K8s cluster (for testing purposes, as the component is to be deployed at the Backend Layer Cloud environment).

Needs	VPN access to the Malaga testbed via Virtual Private Network (VPN) configuration file for OpenVPN[8] client.

Table 12: Insights Tool test cases (M24).

TC-01 – Connection (integration) of the tool to the Analytics engine for reception of metrics. *The test case illustrates the steps to facilitate connection and subscribe to the Analytics engine publishing component, so as to obtain different kinds of metrics.*	
Assumption(s)/Precondition(s)	VPN access to Malaga testbed.
Test steps	1. Launch the application. 2. Click on the bright orange 'Connect' button. 3. Enter the RabbitMQ Host IP address in the 'Broker Address' field. 4. Enter the RabbitMQ Port in the 'Port' field. 5. Click on the bright orange 'Connect' button.
Expected result(s)	The log area should display the message 'Connected to broker on [Port Address]. Periodically, the log should display two types of messages, one carrying statistical information and the other carrying KPI measurements in their respective payloads.

Table 13: Insights Tool pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• The module crashes. • No output produced after predesignated time.

3.1.3 Northbound API/Configurator

The following Tables (Table 14 – Table 16) present indicative test plan features for testing the Northbound API/Configurator, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 14: Northbound API/Configurator testable features plan (M24).

Test plan reference	Description
Test Item	Northbound API/Configurator

[8] https://openvpn.net/

Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• As a UC/ Testbed owner, log into the Northbound API/Configurator interface [L]. • As a UC owner, create a new experiment request [M]. • As a UC owner, set the parameters/KPIs to be requested by the TB and submit a new experiment request [M]. • As a Testbed owner, receive and view a new experiment request [M]. • As a Testbed owner, accept, reject or submit a "revise and resubmit" request for an experiment request [M]. • As UC owner, amend experiment and resubmit [M]. • As UC/ Testbed owner, view list of requests and status [M].
Features NOT to be tested	• N/A.
Approach	UC owners to use the Northbound API/Configurator for sending configuration and scheduling requests to dedicated Testbeds. Testbed owners to use the Northbound API/Configurator for managing the requests and responding.
Needs	Access to the TBs exposed APIs is required.

Table 15: Northbound API/Configurator test cases (M24).

TC-01 – As a UC/Testbed owner, log into the Northbound API/Configurator interface. *A user registered as a UC or Testbed owner in the Northbound API/Configurator UI should be able to login.*	
Assumption(s)/Precondition(s)	The UC/Testbed owner has registered beforehand. The UC/ Testbed owner uses proper credentials and ensures that they are typed in correctly.
Test steps	1. Load the Northbound API/Configurator login page. 2. Enter the user credentials in the corresponding input fields. 3. Click on the 'Log in' button.
Expected result(s)	The log in page redirects the user to the main Configurator page.
TC-02 – As a UC owner, create new experiment request. *A user registered as a UC owner in the Northbound API/Configurator UI should be able to create a new experiment request.*	
Assumption(s)/Precondition(s)	N/A.
Test steps	1. Log in (TC-01) using the user credentials. 2. Click on "New experiment" button.
Expected result(s)	The user should be redirected to a new page where the testbed/parameters can be selected.

TC-03 - As a UC owner, set the parameters/KPIs to be requested by the Testbed and submit a new experiment request. *The user should be able to configure the parameters to be tested and submit to the Testbed.*	
Assumption(s)/Precondition(s)	N/A
Test steps	1. Once the "New experiment" button has been clicked (TC-02), the user should be redirected to the page, where the experiment can be configured. 2. The user is able to select the list of parameters/KPIs that wants to test. These parameters are predefined and are relevant to the user's UC (TC-01). 3. The user is able to select the Testbed for executing the experiment. 4. The users are able to select a proposed date and time for the onboarding of their Network Application. 5. The user submits the new request by clicking the "Submit" button.
Expected result(s)	Following the parameter input by the user and the submission, a configuration file is created, and forwarded to the respective TB.
TC-04 – As a Testbed owner, receive and view a new experiment request. *A Testbed owner should be able to receive and view new experiment requests submitted through the Northbound API/ Configurator.*	
Assumption(s)/Precondition(s)	A new experiment request has been submitted by a UC owner.
Test steps	1. Log in (TC-01) using the user credentials. 2. Click on "Requests" button. 3. View the list of pending requests.
Expected result(s)	The Testbed owner should be able to navigate through the Northbound API/Configurator and view new experiment requests and be able to either accept, reject or send back to the UC owner for revision.
TC-05 – As a Testbed owner, accept, reject or submit a "revise and resubmit" request for an experiment request. *A Testbed owner should be able to either accept, reject or request a revision of the experiment request.*	
Assumption(s)/Precondition(s)	N/A.
Test steps	1. The request of a UC owner is reviewed. 2. The request is either accepted, rejected or revisions are requested via a drop-down menu. 3. A text box is available for providing comments to the UC owner. Filling the text box is mandatory in case the request is rejected or revisions are requested.

Expected result(s)	The Testbed owner should be able to manage all incoming requests and respond to the original requester via the Northbound API/Configurator.
TC-06 – As UC owner, amend experiment and resubmit. *A UC owner should be able to revise the experiment and resubmit to the Testbed owner for approval.*	
Assumption(s)/Precondition(s)	A "revise and resubmit" request has been received by the Testbed owner.
Test steps	1. Log in (TC-01) using the user credentials. 2. Click on the "Requests" button. 3. Click on the request that has a status of "Revisions requested". 4. Amend experiment in order to comply with Testbed owner requests. 5. The user submits the revised request by clicking the "Submit" button.
Expected result(s)	The UC owner should be able to submit a revised request to the Testbed owner.
TC-07 – As UC/Testbed owner, view the list of requests and status. *All users should be able to view the list of requests and their status in the Northbound API/Configurator.*	
Assumption(s)/Precondition(s)	A request involving the user either a UC or Testbed owner has been previously submitted.
Test steps	1. Log in (TC-01) using the user credentials. 2. Click on the "Requests" button. 3. View the list of requests and their status (Accepted, rejected, revisions requested, executed).
Expected result(s)	The user should be able to view a log of past and future requests and their status.

Table 16: Northbound API/Configurator pass/fail criteria (M24).

Criterion	**Description**
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• UC owners are not able to utilise the functionalities provided by the component. • Testbed owners are not able to utilise the functionalities provided by the component.

3.1.4 Network Service Repository

The following Tables (Table 17 – Table 19) present indicative test plan features for testing the Network Service (NS) Repository module, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 17: Network Service Repository testable features plan (M24).

Test plan reference	Description
Test Item	Network Service Repository
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• Successful uploading of a Virtual/Cloud-native Network Function (VNF/ CNF) descriptor and/or NS/Helm chart [L]. • Successful updating of a VNF/CNF descriptors or NS/Helm chart [L] • Successful deletion of a VNF/CNF descriptor or NS/Helm chart [L].
Features NOT to be tested	N/A.
Approach	Use the Network Service Repository to upload/update/delete descriptors/Helm charts (The repository has its own front-end test environment). Use the Network Service Repository to onboard VNF/CNF descriptors or NS/Helm charts to the cross-testbed MANO.
Needs	Access to the Network Service Repository and front-end test environment.

Table 18: Network Service Repository test cases (M24).

TC-01 – Successful uploading of a VNF/CNF descriptor and/or NS/helm chart. *Successful uploading of a VNF/CNF descriptor and/or NS/Helm chart to the Network Service Repository.*	
Assumption(s)/Precondition(s)	A VNF/CNF descriptor and/or NS/Helm chart is available.
Test steps	1. Load the Network Service Repository front-end test environment. 2. Select the files (descriptors/NS/Helm charts) in the Network Service Repository front end test environment. 3. Press the ‘Upload’ button.
Expected result(s)	The file has been uploaded to the repository.
TC-02 – Successful updating of a VNF/CNF descriptor and/or NS/Helm chart. *Successful updating of a VNF/CNF descriptor and/or NS/Helm chart in the Network Service Repository.*	
Assumption(s)/Precondition(s)	A VNF/CNF descriptor and/or NS/Helm chart have been uploaded to the repository.
Test steps	1. Load the Network Service Repository front-end test environment. 2. Select the files (descriptors/NS/Helm charts) in the network service repository.

	3. Press the 'Update' button.
Expected result(s)	The files information have been updated in the repository.
TC-03 – Successful deletion of a VNF/CNF descriptor and/or NS/helm chart. *Successful deletion of a VNF/CNF descriptor and/or NS/Helm chart in the Network Service Repository.*	
Assumption(s)/Precondition(s)	A VNF/CNF descriptor and/or NS/Helm chart have been uploaded in the repository.
Test steps	1. Load the Network Service Repository front-end test environment. 2. Select the files (descriptors/NS/Helm charts) in the network service repository. 3. Press the 'Delete' button.
Expected result(s)	The information files are deleted from the repository.

Table 19: Network Service Repository pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	-

3.1.5 Experiment Coordinator

The following Tables (Table 20 – Table 22) present indicative test plan features for testing the Experiment Coordinator module, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 20: Experiment Coordinator testable features plan (M24).

Test plan reference	Description
Test Item	Experiment Coordinator
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• Testbed experiment execution [H].
Features NOT to be tested	N/A.
Approach	Correct execution of experiments by the Experiment Coordinator.
Needs	-

Table 21: Experiment Coordinator test cases (M24).

TC-01 – Testbed experiment execution. *Correct execution of experiments by the Experiment Coordinator.*	
Assumption(s)/Precondition(s)	The Experiment Coordinator is deployed, listening for connections and with a defined test experiment.
Test steps	1. Request to the endpoint **/experiment/run** with the descriptor of a testcase. 2. Checking HTML 200 response with JSON message containing the execution id of the experiment. 3. Estimated maximum waiting time of 2 minutes. 4. Request to the endpoint **/logs** verifying that it contains the expected output.
Expected result(s)	Output log with the expected content.

Table 22: Experiment Coordinator pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• Any of the cases fail to produce the expected result.

3.1.6 5G Traffic Simulator

The 5G Traffic Simulator is formed by the 5G Traffic Simulator Manager and iPerf Agent's components. These are the components in charge of generating 5G traffic in the network. Through a request to the 5G Traffic Manager, traffic generation requests are directed to the iPerf Agent's deployed in the desired platforms. The intention is for this traffic to be published in the cross-testbed MANO tooling by the iPerf Agent's.

The following Tables (Table 23 – Table 25) present indicative test plan features for testing the 5G Traffic Simulator module, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 23: 5G Traffic Simulator testable features plan (M24).

Test plan reference	Description
Test Item	5G Traffic Simulator
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• Successful publication of messages by the iPerf Agent [M]. • Successful sending traffic between server and client remote iPerf agents[L]. • Successful sending of messages by several clients[M].

Features NOT to be tested	N/A.
Approach	Use the 5G Traffic Simulator charts to the cross-testbed MANO.
Needs	-

Table 24: 5G Traffic Simulator test cases (M24).

TC-01 – Successful publication of messages by the iPerf Agent. *Successful publication of messages intended for the cross-testbed MANO.*	
Assumption(s)/Precondition(s)	One remote iPerf agent will be deployed on the target platform acting as a server and another on a different platform acting as client.
Test steps	1. Set server on target remote iPerf agent using the /start endpoint. 2. Set client on target remote iPerf agent using the /start endpoint.
Expected result(s)	The traffic appears in the MQTT queue.
TC-02 – Successful sending traffic between server and client remote iPerf agents. *Successful sending traffic between server and client remote iPerf agents.*	
Assumption(s)/Precondition(s)	One remote iPerf agent will be deployed on the target platform acting as a server and another on a different platform acting as a client.
Test steps	1. Set server on target remote iPerf agent using the /start endpoint. 2. Set client on target remote iPerf agent using the /start endpoint
Expected result(s)	The server receives the generated traffic.
TC-03 – Successful sending of messages by several clients. *Multiple clients sending messages to the same server.*	
Assumption(s)/Precondition(s)	One remote iPerf agent deployed acting as a server and several acting as clients.
Test steps	1. Set server on target remote iPerf agent using the /start endpoint. 2. Set all clients on targets remote iPerf agents using the /start end-point of each of them
Expected result(s)	The server shows the traffic of all clients.

Table 25: 5G Traffic Simulator pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• Any of the cases fail to produce the expected result.

3.1.7 Analytics engine components

The Analytics Engine is comprised of several interdependent components and monitors experimental application conditions in order to identify anomalies and predict KPIs.

3.1.7.1 Analytics Aggregator

The following Tables (Table 26 – Table 28) present indicative test plan features for testing the Analytics Aggregator module, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 26: Analytics Aggregator testable features plan (M24).

Test plan reference	Description
Test Item	Analytics Aggregator
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• KPIs aggregation [H]. • Quality of Service (QoS)/Quality of Experience (QoE) parameters aggregation [H]. • Notification to the Insights Tool [H].
Features NOT to be tested	N/A
Approach	The tester will use a test tool to pass the Analytics Aggregator the specific input data and receive the output.
Needs	The tester will need a file containing mock data relevant to the specific functionality to be tested.

Table 27: Analytics Aggregator test cases (M24).

TC-01 – KPIs aggregation. *The test provides the Analytics Aggregator with mock data (simulating KPIs values coming from different testbeds) and provides the output of the aggregated data in the expected format.*	
Assumption(s)/Pre-condition(s)	Input data set with KPIs values needed to perform the aggregation.
Test steps	1. The tester launches the test tool providing an input file with a set of suitable KPIs data.

	2. The test tool provides the Analytics Aggregator with the input data simulating KPIs values coming from different testbeds. 3. The Analytics Aggregator aggregates and formats data in the output format for the Insights Tool. 4. The test tool reads and verifies the output of the Aggregator.
Expected result(s)	Output data in the expected format.
TC-02 – QoS/QoE parameters aggregation. *The test provides the Analytics Aggregator with mock data (simulating QoS/QoE parameters coming from different testbeds) and provides the output of the aggregated data in the expected format.*	
Assumption(s)/Pre-condition(s)	Input data set with QoS/QoE parameters needed to perform the aggregation.
Test steps	1. The tester launches the test tool providing an input file with a set of suitable QoS/QoE parameters. 2. The test script provides the Analytics Aggregator with the input data simulating QoS/QoE parameters coming from different testbeds. 3. The Analytics Aggregator aggregates and formats data in the output format for the Insights Tool. 4. The test tool reads and verifies the output of the Aggregator.
Expected result(s)	Output data in the expected format.
TC-03 – Notification to the Insights Tool. *The test provides the Analytics Aggregator with mock data including detected anomalies and provides in output data in the expected format.*	
Assumption(s)/Pre-condition(s)	Input data set with anomalies.
Test steps	1. The tester launches the test tool providing an input file with a set of suitable data about detected anomalies. 2. The test tool provides the Analytics Aggregator with the input data. 3. The Analytics Aggregator produces the output for the Insights Tool. 4. The calculated data is produced as output.
Expected result(s)	Output data in the expected format.

Table 28: Analytics Aggregator pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• The module crashes. • No output is produced.

3.1.7.2 Analytics Driver

The following Tables (Table 29 – Table 31) present indicative test plan features for testing the Analytics Driver module, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 29: Analytics Driver testable features plan (M24).

Test plan reference	Description
Test Item	Analytics Driver
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• Data extraction for KPIs and QoE/QoS monitoring [H].
Features NOT to be tested	N/A
Approach	The tester will use a test tool to pass the Analytics Driver the input data and receive the output.
Needs	The tester will need a file containing a mock data set with known features.

Table 30: Analytics Driver test cases (M24).

TC-01 – Data extraction for KPIs and QoE/QoS Monitoring. *The test will provide the Analytics Driver with a mock data set of measurements (from the UCs and the infra-structure) and will validate the information extracted from the original data and passed to the KPI Monitor and the QoE/QoS Monitor.*	
Assumption(s)/Precondition(s)	Input data set with known features.
Test steps	1. The tester launches the test tool providing an input file with a set of known features. 2. The test script pushes the input data provided into a dedicated queue. Then, the Analytics Driver reads data from the queue and extracts relevant data for KPIs or QoE/QoS calculation. 3. The Analytics Driver pushes result data into the output queues. The test tool reads from the output queues and displays the "tagged" data to the tester.
Expected result(s)	Data properly processed and "tagged" for routing towards KPI Monitor and QoS/QoE Monitor modules.

Table 31: Analytics Driver pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• The module crashes. • No output is produced.

3.1.7.3 KPI Monitor

The following Tables (Table 32 – Table 34) present indicative test plan features for testing the KPI Monitor module, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 32: KPI Monitor testable features plan (M24).

Test plan reference	Description
Test Item	KPI Monitor
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• i-th iteration of KPI calculation and storage [H].
Features NOT to be tested	N/A
Approach	The tester will use a test tool to pass the input data to the KPI Monitor and receive the output.
Needs	The tester will need a file containing mock data relevant to the specific KPI calculation.

Table 33: KPI Monitor test cases (M24).

TC-01 – KPI Calculation *The test will provide the KPI Monitor with mock measurements data and will provide as output the calculated KPI.*	
Assumption(s)/Precondition(s)	Input data set with measurements needed to calculate the specific KPI.
Test steps	1. The tester launches the test tool providing an input file with a set of suitable measurements data. 2. The test tool pushes the input data provided into a dedicated input queue. Then, the KPI Monitor reads data from the queue and calculates the KPI. 3. The calculated KPI is then stored in the internal database.

	4. The calculated KPI is retrieved from the internal database and produced as output.
Expected result(s)	Expected KPI value.

Table 34: KPI Monitor pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• The module crashes. • No output is produced. • KPI not correctly stored/retrieved from the DB (error logged).

3.1.7.4 QoS/QoE Monitor

The following Tables (Table 35 – Table 37) present indicative test plan features for testing the QoS/QoE Monitor module, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 35: QoS/QoE Monitor testable features plan (M24).

Test plan reference	Description
Test Item	QoS/QoE Monitor
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• Anomaly detection: positive case [H]. • Anomaly detection: negative case [H].
Features NOT to be tested	N/A
Approach	The tester will use a test tool to pass the QoS/QoE Monitor the specific input data and receive the output.
Needs	The tester will need a file containing mock data relevant to the specific functionality to be tested.

Table 36: QoS/QoE Monitor test cases (M24).

TC-01 – Anomaly detection: positive case. *The test provides the QoS/QoE Monitor with an input set of parameters for a given time slot including an anomaly and verifies that the system correctly detects it.*	
Assumption(s)/Precondition(s)	Input dataset of parameters for a given time slot including an anomaly.

Test steps	1. The tester launches the test tool providing an input file with a set of suitable parameters and pushes the data into the input queue of the QoS/QoE Monitor. 2. The QoS/QoE Monitor reads data, processes them and performs prediction against trained models. 3. The result (anomaly detected/not detected) is produced as output. 4. The test tool reads the produced result.
Expected result(s)	The module correctly detects the anomaly.
TC-02 – Anomaly detection: negative case. *The test provides the QoS/QoE Monitor with an input set of parameters for a given time slot not including any anomaly and verifies that the system correctly doesn't detect any anomaly.*	
Assumption(s)/Precondition(s)	Input dataset of parameters for a given time slot representing a normal condition of the system (no anomalies).
Test steps	1. The tester launches the test tool providing an input file with a set of suitable parameters and pushes the data into the input queue of the QoS/QoE Monitor. 2. The QoS/QoE Monitor reads data, processes them and performs prediction against trained models. 3. The result (anomaly detected/not detected) is produced as output. 4. The test tool reads the produced result.
Expected result(s)	The module doesn't detect any anomaly.

Table 37: QoS/QoE Monitor pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• Anomalies not correctly detected. • No output is produced.

3.1.8 Cross-testbed VNF chain placement

The following Tables (Table 38 – Table 40) present indicative test plan features for testing the VNF chain placement module, with the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 38: VNF chain placement testable features plan (M24).

Test plan reference	Description
Test Item	VNF chain placement
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• Retrieve API resources (deployment and policies) from API-servers (Manager and Algorithm) [L]. • Subscribe the Publisher [L]. • Optimal Selection of cluster which is belonged to testbed (Manager) and optimal resource propagation among nodes (Algorithm) [M]. • VNF chain re-location[M].
Features NOT to be tested	N/A.
Approach	• Use python-client library to connect the Kubernetes API-server • Define a specific API to connect to cross-testbed MANO API-server • Use python-client library to subscribe to a published topic • Use Artificial Intelligence subfields like Integer Linear Programming and heuristic. to calculate the optimal placement and support the dynamic changes.
Needs	The cross-testbed MANO be installed. Kubernetes cluster be installed. Metrics be Published. Probably access to the NS/Helm.

Table 39: VNF chain placement test cases (M24).

TC-01 – Retrieve the API resources from the api servers (Manager and Algorithm). *Retrieving standard resource template that are in form of deployment from cross-testbed MANO and Kubernetes api-servers.*	
Assumption(s)/Precondition(s)	Cross-testbed MANO is running successfully. Testbed are joined in the federation. Testbed's clusters are running. API resource objects, like service deployment and Propagation policy are available..
Test steps	1. Manager watches the cross-testbed API server for API resources 2. Algorithm watches the Kubernetes API server for un-scheduled pod.
Expected result(s)	A message "resources is received successfully" has to be printed.

TC-02 – Subscribe the MQTT. *Subscribe to the MQTT*	
Assumption(s)/Precondition(s)	Publisher is running. Topics related to metrics are available.
Test steps	1. Manager uses pika[9] client library to subscribe and receives the published metrics in form of .json file. 2. Parse the contents of .josn file.
Expected result(s)	A message “resources is received successfully” plus a list of metrics has to be printed.

TC-03 –Propagate the resource in testbed (Manager) and among nodes (Algorithm) optimally. *The Manager selects the proper testbed at time and Algorithm compute the optimal place of VNF chain among the available resources.*	
Assumption(s)/Precondition(s)	Deployment is available. Metrics for each testbed are available. Un-healthy clusters and nodes are filtered out.
Test steps	Manager: compare standard cross-testbed scheduler and customized scheduler (VNF chain placement). 1. Deploy a service with default scheduler. 2. Terminate the service. 3. Stress the host cluster. 4. Deploy again the service. Algorithm: Compares the standard Kubernetes scheduler and customized scheduler (VNF chain placement). 1. Deploy a service with default scheduler. 2. Deploy a service with custom scheduler.
Expected result(s)	Manager: • (Standard scheduler) Service deploys in the stressed cluster because it does not take informed decisions based on computational resource. • (VNF chain placement) Service deploys in a cluster with more free computational resources. Algorithm: • (Standard scheduler) Latency may not be guaranteed since Kubernetes scheduler does not use these metrics for decision making. • (VNF chain placement) Required latency is fulfilled.

[9] https://github.com/pika/pika

TC-04 –VNF chain re-location. *Manager guarantees KPIs in long-term.*	
Assumption(s)/Precondition(s)	NS/Helm is deployed.
Test steps	1. The cluster is crashed or the network performance being poor and latency increases (Since the purpose of this test is evaluating the Manager's ability to re-locate services, provision of critical conditions in cluster should be done in this test step). 2. Check that latency exceeds the threshold. 3. Manager terminates the deployment from the desired cluster and spin up, in a cluster closer to end user with better condition.
Expected result(s)	The latency is below the threshold.

Table 40: VNF chain placement pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	Manager and Algorithm do not connect the publisher. Manager selects the cluster with more available computing capacity and latency. Algorithm propagates deployment on the resource with greater latency. Manager cannot automatically re-locate the placement.

3.1.9 Cross-testbed orchestration tooling

The following Tables (Table 41 – Table 43) summarise project Beneficiary CTTC's plan for testing the cross-testbed orchestration module, with reference to the test plan and the intention of demonstrating the sequence of undertaken steps to validate each testable feature or function provisioned.

Table 41: Cross-testbed orchestration testable features plan (M24).

Test plan reference	Description
Test Item	Cross-testbed orchestration manager.
Features to be tested [Level of risk: H: High; M: Medium; L: Low]	• Successful registration of the testbeds/platforms in the cross-testbed orchestration manager [L]. • Successful deployment of services through the cross-testbed orchestration manager [M]. • Successful deployment of services through the cross-testbed orchestration manager under workload conditions [M].
Features NOT to be tested	N/A.

Approach	Access the cross-testbed orchestration manager through the Kubernetes kubectl command-line application.
Needs	Karmada software is installed correctly in one of the Kubernetes clusters. Kubernetes cluster in each testbed/platform must be correctly installed. Access to the NS/HELM repositories should be available.

Table 42: Cross-testbed orchestration manager test cases (M24).

TC-01 – Register two testbeds/platforms. *The manager of the cross-testbed federation registers the testbeds/platforms in the cross-testbed orchestration manager.*	
Assumption(s)/Precondition(s)	The Karmada software is correctly installed in one of the Kubernetes clusters. The tester uses proper credentials to access both Kubernetes clusters and the cross-testbed orchestration manager.
Test steps	1. Set proper credentials to access the cross-testbed orchestration manager. 2. Set proper credentials to access both testbed's Kubernetes clusters. 3. Register testbed #1's Kubernetes clusters into the cross-testbed orchestration manager. 4. Register testbed #2's Kubernetes clusters into the cross-testbed orchestration manager.
Expected result(s)	Both Kubernetes clusters belonging to the testbeds/platforms are correctly registered in the cross-testbed orchestration manager.
TC-02 – Deployment of a network service through the cross-testbed orchestration manager. *The user of the federation can deploy a network service through the cross-testbed orchestration manager.*	
Assumption(s)/Precondition(s)	At least two testbeds/platforms are registered in the cross-testbed orchestration manager. The tester uses proper credentials to access testbed's Kubernetes clusters and the cross-testbed orchestration manager.
Test steps	1. Apply the PropagationPolicy object to Karmada. 2. Apply the resource template (Deployment) object to Karmada.
Expected result(s)	The Deployment is applied and the VNFs are deployed in the testbed according to the specification of the PropagationPolicy object.
TC-03 – Deployment of a network service through the cross-testbed orchestration manager under workload conditions.	

The user of the federation can deploy a network service through the cross-testbed orchestration manager under workload conditions.	
Assumption(s)/Precondition(s)	At least two testbeds/platforms are registered in the cross-testbed orchestration manager. The tester uses proper credentials to access testbed's Kubernetes clusters and the cross-testbed orchestration manager.
Test steps	1. Create workload containers in Kubernetes cluster #1. 2. Apply the PropagationPolicy object to Karmada. 3. Apply the resource template (Deployment) object to Karmada.
Expected result(s)	The Deployment is applied and the VNFs are deployed in the testbed #2 (*i.e.*, the less congested Kubernetes cluster).

Table 43: Cross-testbed orchestration manager pass/fail criteria (M24).

Criterion	Description
Successful Completion of this Plan when	All test cases are completed (no errors).
Failure	• Kubernetes clusters are not correctly registered. • Network service is not deployed in any Kubernetes cluster. • Network service is not deployed in the less congested cluster.

3.2 Testing and validation results

In this Section, the validation of each test item defined in the previous Section will be reported. Test results are summarised in Table 44 – Table 55:

Table 44: Experiment Planning Interface front-end and back-end validation results (M24).

Test case ID	Test case name	Result
TC-01	As an anonymous user, log into the 5G-EPICENTRE Portal.	PASS
TC-02	As an experimenter, initiate and carry out a new experiment request.	PASS
TC-03	As an experimenter, check notification status after an experiment request has been updated by a testbed owner.	PASS
TC-04	As an experimenter, submit an amended version of an experiment.	PASS
TC-05	As a function developer, request the onboarding of a new Network Application package via the Portal.	PASS
TC-06	As a testbed owner, receive and view a new experiment request.	PASS

TC-07	As a testbed owner, receive and view the contents of a Network Application package onboarding request.	PASS
TC-08	As a testbed owner, notify on the status of an experiment request.	PASS
TC-09	As a testbed owner, notify on the status of a Network Application package onboarding request.	PASS

Table 45: Insights Tool validation results (M24).

Test case ID	Test case name	Result
TC-01	Connection (integration) of the tool to the Analytics engine for reception of metrics.	PASS

Table 46: Northbound API/Configurator validation results (M24).

Test case ID	Test case name	Result
TC-01	As a UC/TB owner, log into the Northbound API/Configurator interface.	Planned
TC-02	As a UC owner, create new experiment request.	Planned
TC-03	As a UC owner, set the parameters/KPIs to be requested by the TB and submit a new experiment request.	Planned
TC-04	As a testbed owner, receive and view a new experiment request.	Planned
TC-05	As a testbed owner, accept, reject or submit a “revise and resubmit” request for an experiment request.	Planned
TC-06	As UC owner, amend experiment and resubmit.	Planned
TC-07	As UC/TB owner, view list of requests and status.	Planned

Table 47: Network Service Repository validation results (M24).

Test case ID	Test case name	Result
TC-01	Successful uploading of a VNF/CNF descriptor and/or NS/helm chart.	PASS
TC-02	Successful updating of a VNF/CNF descriptor and/or NS/Helm chart.	PASS
TC-03	Successful deletion of a VNF/CNF descriptor and/or NS/helm chart.	PASS

Table 48: Experiment Coordinator validation results (M24).

Test case ID	Test case name	Result
TC-01	Testbed experiment execution.	PASS

Table 49: 5G Traffic Simulator validation results (M24).

Test case ID	Test case name	Result
TC-01	Successful publication of messages by the iPerf Agent.	PASS
TC-02	Successful sending traffic between server and client remote iPerf agents.	PASS
TC-03	Successful sending of messages by several clients.	PASS

Table 50: Analytics Aggregator validation results (M24).

Test case ID	Test case name	Result
TC-01	KPIs aggregation.	Planned
TC-02	QoS/QoE parameters aggregation.	Planned
TC-03	Notification to the Insights Tool.	Planned

Table 51: Analytics Driver validation results (M24).

Test case ID	Test case name	Result
TC-01	Data extraction for KPIs and QoE/QoS Monitoring.	PASS

Table 52: KPI Monitor validation results (M24).

Test case ID	Test case name	Result
TC-01	KPI Calculation.	Planned

Table 53: QoS/QoE Monitor validation results (M24).

Test case ID	Test case name	Result
TC-01	Anomaly detection: positive case.	Planned
TC-02	Anomaly detection: negative case.	Planned

Table 54: Cross-testbed VNF Chain Placement validation results (M24).

Test case ID	Test case name	Result
TC-01	Retrieve API resources from the API servers (Manager and Algorithm).	Planned

TC-02	Subscribe to Publisher.	Planned
TC-03	Optimal Selection on cluster belonged to a testbed (Manager) and optimal resource propagation among nodes (Algorithm).	Planned
TC-04	VNF chain re-location.	Planned

Table 55: Cross-testbed orchestration tooling validation results (M24).

Test case ID	Test case name	Result
TC-01	Register two testbeds/platforms.	PASS
TC-02	Deployment of a network service through the cross-testbed orchestration manager.	PASS
TC-03	Deployment of a network service through the cross-testbed orchestration manager under workload conditions.	PASS

4 Report on usability and user experience assessment (M24)

The purpose of this section is to present how formative usability evaluation will be carried out for testing (and subsequently, improving) user experience of the 5G-EPICENTRE front-end (*i.e.*, user-facing) components. Particular focus is placed on identifying, and eliminating usability issues, which might trigger the risk of end-users disengaging from the procedure of experimentation with the platform components as a result of an improper design of the UI experience. The assessment will be conducted by employing two different assessment methods, namely heuristic evaluation and user testing, the combination of which has been reported to increase the number of usability problems identified (and subsequently corrected) in a user interface, taking into account that each method discovers different usability issues [3] [4].

4.1 Heuristic evaluation

To pass judgement on the platform's front-end components in terms of their usability, the heuristic evaluation approach [5] will be employed. It will be carried out with help from a small group of evaluators (UX experts), who will inspect the developed UIs and determine whether they conform to well-documented usability principles (known as *heuristics* [6]). The process is illustrated in Figure 2.

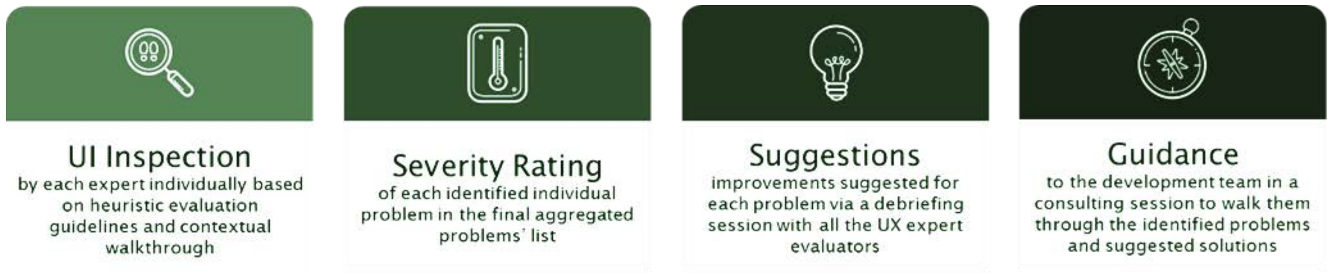
UI Inspection
by each expert individually based on heuristic evaluation guidelines and contextual walkthrough
Severity Rating
of each identified individual problem in the final aggregated problems' list
Suggestions
improvements suggested for each problem via a debriefing session with all the UX expert evaluators
Guidance
to the development team in a consulting session to walk them through the identified problems and suggested solutions

Figure 2: Usability inspection process

Firstly, evaluators will be given a report sheet where they will be able to note down problems they have identified, alongside the usability principle affected by them. Each evaluator will carry out inspection individually and independently, to not cross-contaminate each other's reports.

An integrated report will be compiled from the isolated documents handed by each evaluator, ensuring that any potential duplicate usability problems are omitted from the evaluation report. Once the final report is compiled, each evaluator will rate each problematic usability issue with a severity score (see Table 56), decided based on the following criteria:

- the number of times the problem occurs;
- the severity of the impact that the problem causes;
- the problem persisting despite user workarounds;
- the impact of the problem on the marketability of the test item.

Table 56: Usability severity scores

Score	Scale
0	I disagree that this is a usability problem.
1	The identified problem is cosmetic only.

2	The usability problem is minor.
3	The usability problem is major.
4	The usability problem is catastrophic.

All individual scores will be aggregated and the average will be calculated to finally grade the issue with a conclusive severity rating. Each evaluator will then be consulted on potential solutions to the problems identified.

The Table below (Table 57) presents usability evaluation heuristics that will be employed to assess usability of the 5G-EPICENTRE user-facing applications.

Table 57: 5G-EPICENTRE usability valuation heuristics

Heuristic	Clarification
Visibility of system status	The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
Match between system and the real world	The system should speak users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
User control and freedom	Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
Consistency and standards	Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
Error prevention	Careful design, which prevents a problem from occurring in the first place is even better than good error messages. Either eliminate error-prone conditions, or check for them, and present users with a confirmation option before they commit to the action.
Recognition rather than recall	Minimise users' memory load by making objects, actions, and options visible. Users should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible, or easily retrievable whenever appropriate.
Flexibility and efficiency of use	Accelerators — unseen by novice users — may often speed up the interaction for expert users, such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
Aesthetic and minimalist design	Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors	Error messages should be expressed in plain language (no codes), precisely indicating the problem, and constructively suggesting a solution.
Help and documentation	Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on users' tasks, list concrete steps to be carried out, and not be too large.

The above methodology has been employed widely toward detecting and eliminating usability problems throughout the system design and implementation procedures. In addition, heuristic evaluation is capable of determining issues that user testing is unable to capture and vice versa, hence warranting that both methods are employed to maximise the likelihood of usability problems being addressed prior to a system becoming available to the intended audience.

4.2 User testing

User based evaluation is a method involving a number of representative end users testing the system [5]. There is a plethora of variations of the method, depending on when it is conducted, where, and how [7] [8] [9].

The user-based studies that will be performed in the context of the 5G-EPICENTRE project will be remotely moderated sessions, assessing the platform's effectiveness, efficiency, and overall user experience. To this end, at least 10 users will participate, with expertise in developing Network Applications and carrying out 5G experiments.

The user-based evaluation study will be conducted remotely through a teleconferencing platform. An experienced facilitator will be responsible for the coordination of each testing session. Overall, each session will involve four main phases, as shown in Figure 3:

1. Introduction: the facilitator welcomes the participant, explains the purpose of the study and the testing procedures, and acquires the participant's signed consent to proceed with the study.
2. User observation: this is the main phase of the session during which the facilitator provides the participant with specific tasks that need to be accomplished with the system being tested. The tasks are provided in writing, in a clear and concise format, describing the goal that the user is asked to achieve. As the user interacts with the system to accomplish the task goal, the facilitator records specific usability metrics (*e.g.*, how many errors occurred, how much time the user required in order to complete the task, *etc*.) and observes the overall user behaviour (*e.g.*, positive or negative statements, comments, suggestions, *etc*.), keeping notes throughout.
3. Questionnaire-filling: at the end of each task and after all tasks have been concluded, the facilitator provides a questionnaire for the user to fill-in, aiming to assess their satisfaction with regard to the provided usability and user experience of the platform.
4. Interview: the facilitator asks clarifying questions to the participant, aiming to shed light on particular instances of the session that raised inquiries as well as to understand the participant's overall opinion of the system. In the end, the facilitator thanks the participant for their contribution and responds to any questions they may have.

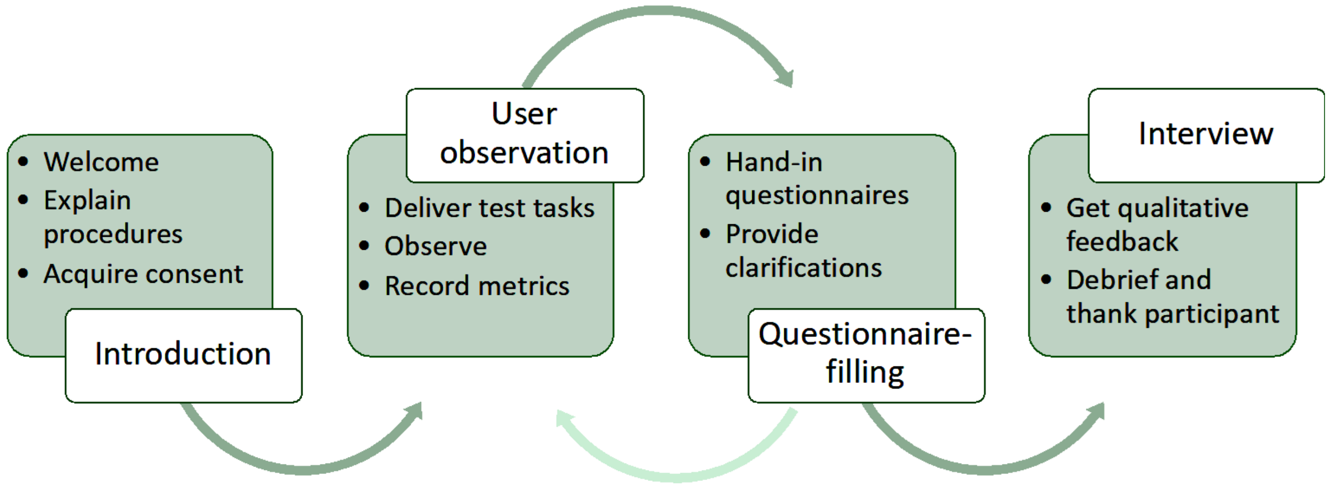

Figure 3: User testing process

After conducting all the user testing sessions, all the recorded data will be analysed in order to identify particular aspects of the platform that need to be improved, as well as overall negative and positive aspects of the user experience. Conclusions will be drawn regarding the usability of the platform and follow-up actions will be planned accordingly.

Finally, it is noted that the user-based evaluation will be conducted after the heuristic evaluation and once all the identified problems – at least those characterized as major or more serious – have been addressed.

5 Conclusions

The deliverable has reported on the current status of system integration and testing activities throughout the course of the integration Task lifetime so far (T4.4, M8-M24), as well as from the start of the project, in the case of the system testing and validation Task (T4.5). It follows up completion of the preliminary version of the 5G-EPICENTRE experimentation facility (D4.4), corresponding to the project milestone MS4, while also reporting on the status of the internally set milestones for preparing the four testbed platforms to accommodate the afore-mentioned service-oriented system.

A key contribution of this report is the detailed reporting of the integration status and activities with respect to both the integration plan put in place in M12 (D4.1), as well as from the perspective of the technical WP activities. The deliverable has thus reported on the project partners' collaboration among Tasks at both the individual WP level, as well as which interdependencies were formed between partner development teams in different WPs. Task 4.4 has thus represented the central hub to coordinate those activities, and the integration targets set in the individual integration roadmaps for both the project and each testbed to be met with success. In this regard, we also provide lessons learned from the integration activities to highlight best practices that will continue to characterize integration activities throughout the final year of the project implementation.

A second key contribution lies with the reporting of the project test plan and individual test case documentations for all interdependent facility test items. In addition, the document presents the partners' plan for carrying out formative usability evaluation of the full system, so that user-facing application interfaces provisioned to the system's target end users can be refined to meet their needs and expectations.

The contents in this deliverable will be updated in M34 with the release of the deliverable D4.7, where the Consortium will incorporate novel insights, test cases and integration outcomes corresponding to project developments over the course of M25-M36.

References

[1] IEEE Standard for Software and System Test Documentation. (2008). *IEEE Std 829-2008*, 1-150. https://doi.org/10.1109/IEEESTD.2008.4578383.

[2] Sayadi, B., Chang, C.-Y., Tranoris, C., Iordache, M., Katsaros, K., Vilalta, R., … & Makropoulos, George. (2022). *Network Applications: Opening up 5G and beyond networks*. Zenodo. https://doi.org/10.5281/zenodo.7123919.

[3] Fu, L., Salvendy, G., & Turley, L. (2002). Effectiveness of user testing and heuristic evaluation as a function of performance classification. *Behaviour & information technology*, *21*(2), 137-143.

[4] Maguire, M., & Isherwood, P. (2018, July). A comparison of user testing and heuristic evaluation methods for identifying website usability problems. In *International Conference of Design, User Experience, and Usability* (pp. 429-438). Springer, Cham.

[5] Nielsen, J. (1994). *Usability Engineering*. Morgan Kaufmann.

[6] Nielsen, J. (1994). Enhancing the Explanatory Power of Usability Heuristics. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (pp. 152-158). 10.1145/191666.191729.

[7] Ntoa, S., Margetis, G., Antona, M., & Stephanidis, C. (2021). User experience evaluation in intelligent environments: A comprehensive framework. *Technologies*, *9*(2), 41.

[8] Fernandez, A., Insfran, E., & Abrahão, S. (2011). Usability evaluation methods for the web: A systematic mapping study. *Information and software Technology*, *53*(8), 789-817.

[9] Vermeeren, A. P., Law, E. L. C., Roto, V., Obrist, M., Hoonhout, J., & Väänänen-Vainio-Mattila, K. (2010, October). User experience evaluation methods: current state and development needs. In *Proceedings of the 6th Nordic conference on human-computer interaction: Extending boundaries* (pp. 521-530)

Annex I: 5G-EPICENTRE test plan

5G ExPerimentation Infrastructure hosting Cloud-nativE Netapps for public proTection and disaster RElief

Innovation Action – ICT-41-2020 - 5G PPP – 5G

Innovations for verticals with third party services

5G-EPICENTRE Test Plan

Lead beneficiary: FORTH

Original draft date: June 2021

Project Title:	5G-EPICENTRE - 5G ExPerimentation Infrastructure hosting Cloud-nativE Netapps for public proTection and disaster RElief
Duration:	1 January 2021 – 31 December 2023
Project URL	https://www.5gepicentre.eu/

This project has received funding from the European Union's Horizon 2020 Innovation Action programme under Grant Agreement No 101016521.

www.5gepicentre.eu

TEST PLAN IDENTIFIER

5G-EPICENTRE_Test_Plan_v1.0.

REFERENCES

Project documentation in support of the test plan include:

- ANNEX 1 (part A) Innovation action NUMBER — 101016521 — 5G-EPICENTRE (Description of the action Amendment 2.0, GA) [ref1].
- Deliverable D1.3: Experimentation requirements and architecture specification preliminary version [ref2].
- Deliverable D3.1: 5G EPICENTRE Northbound API [ref3].
- Deliverable D3.3: 5G EPICENTRE User Interface [ref4].
- Deliverable D4.1: Integration plan and framework [ref5].
- Deliverable D4.4: 5G-EPICENTRE experimentation facility preliminary version [ref6].
- 5G-EPICENTRE live risk registry and mitigation/contingency plan documentation [ref7].
- Deliverable D4.6: Integration, Verification and Testing Report preliminary version [ref8].

INTRODUCTION

The 5G-EPICENTRE test plan defines the various test cases and test scenarios for the novel components of the 5G-EPICENTRE experimentation facility (as described in the preliminary architecture documentation, *i.e.*, [ref2]), since said components are both developed in the context of the project; and integrated into the platform by 5G-EPICENTRE technical partners themselves. It is aimed at establishing the means by which individuals responsible for carrying out the testing of those components can verify them against a set list of performance and behaviour specifications. The plan further identifies the tools by which partners can communicate and coordinate testing activities.

It is important to note that tests within the scope of this particular test plan do not correspond to the actual evaluation of platform KPIs, as specified in [ref1] (which is a process carried out in the context of WP5), nor does it cover testing of the vertical applications corresponding to the first party experiments (since those vertical applications are developed and maintained outside the scope of the project, as part of the first party experimenters' businesses, and are therefore covered by their respective internally maintained test plan documentation).

TEST ITEMS

This test plan covers all novel 5G-EPICENTRE components, and hence does not consider third-party tools brought in to cover specific functionalities envisioned by the final system. Components considered in this test plan may include both Infrastructure elements/services, as well as components delivered in the form of Network Applications (and their potential individual components). Considering the current level of maturity of platform components (reflected in the project work plan, as specified in [ref1]), the elements to be tested include:

- the experiment planning interface front-end and back-end components;
- the insight tool of the front-end layer (Portal);
- the platform northbound interface/configurator network application;
- the network service repository module and API server;
- the platform back-end Experiment Coordinator module;
- the novel publisher component;
- the platform back-end 5G Traffic Simulation manager component (and iPerf agents);

- the cross-testbed VNF chain placement module components;
- the 5G-EPICENTRE analytics engine components.
- the integration of the Karmada components for cross-testbed K8s cluster orchestration tooling.

Future versions of this test plan will include further systems to be tested, as these become available to the integration team, based on their scheduled release milestones, listed in [ref1].

SOFTWARE RISK ISSUES

Software/Technical risks have been identified in the latest version of the project risk management plan [ref7]:

- Poor framework performance during tests, resulting in failure of demonstrators and/or small-scale trials.
- Personal and/or sensitive data is disclosed due to improper data handling.
- Developed components are below the required level of maturity to sustain the difficult application requirements, requiring specialized customized development to achieve the appropriate TRL for their deployment in the experimental phases.
- Third party experimenters disengage from the experimentation procedure because of the process being perceived as cumbersome and not user-friendly.
- Deploying and integration of scheduler in different testbeds, because each testbed has a different equipment, resources, and configuration, hence it may cause some issue in the final deployment.
- Potential compatibility issue of scheduler with the new release of Kubernetes.
- Any potential defects (bugs) that may emerge during testing, which may indicate technical risks within the software delivery.

FEATURES TO BE TESTED

A complete list of features of the experimentation facility that are tested under this test plan will be reported in each test case, following the individual test item plan template showcased in Section 3.1 of D4.6 [ref8].

FEATURES NOT TO BE TESTED

A complete list of features of the experimentation facility that are not going to be tested in the current version of the test plan will be reported in each test case, following the individual test item plan template showcased in Section 3.1 of D4.6 [ref8].

APPROACH

Testing is to be carried out in accordance to the specified test cases, which have been defined by the partners and are available in Section 3.1 of D4.6 [ref8]. Testing teams (or individual testers) have been formed internally within each partner organization. They are responsible for executing the tests in accordance to the test case instructions, and mark each one with the corresponding grade ('Pass', 'Fail', 'Skip').

In case of a failed test case, a bug report should be created in the selected issue tracker.

ITEM PASS/FAIL CRITERIA

Each individual test item's pass/fail criteria are reported in each test case, following the individual test item plan template showcased in Section 3.1 of D4.6 [ref8].

SUSPENSION CRITERIA AND RESUMPTION REQUIREMENTS

Testing should be suspended if any of the failures indicated in the test cases documentation occur. The testing representatives will be responsible to resolve all issues and repeat the test cases until they succeed.

TEST DELIVERABLES

Deliverable D4.6 will constitute the primary test deliverable, reporting on both the defined test cases and their validation result. D4.7 will constitute the updated version, corresponding to the final set of testable features for the 5G-EPICENTRE components.

Test run results will be saved by partners internally. Test results will be reported to the Technical Manager by each Partner technical representative.

REMAINING TEST TASKS

Remaining test tasks not covered in the present test plan document involve the usability and user experience assessment, which will be covered in Section 4 of D4.6 [ref8].

ENVIRONMENTAL NEEDS

A complete list of environmental needs for the test cases of the experimentation facility corresponding to the test items the current version of the test plan will be reported in each test case, following the individual test item plan template showcased in Section 3.1 of D4.6 [ref8].

STAFFING AND TRAINING NEEDS

Testers assigned should have basic knowledge of the platform.

RESPONSIBILITIES

The project Technical Manager is responsible for facilitating the test plan document by gathering its various inputs from the partners responsible for the components listed in the 'Test Items' section. Partner technical representatives have been appointed internally by each project Beneficiary, each responsible for identifying risks; features to be tested; and not to be tested; setting up the environment for the testing of their individual components; provide the required training to the individuals who will carry out the tests (if not carrying out tests themselves, and reporting on the tests results to the Technical Manager for revising of the test plan and project risk registry.

SCHEDULE

Testing will take place as soon as the component/integrated subsystem is deemed ready for integration. Each testing round should be completed within 2 weeks of the starting date.

RISKS AND CONTINGENCIES

Overall risks to the project that might affect the testing process have been identified in the latest version of the project risk management plan [ref7]:

- Withdrawal of consortium member from the project/Unexpected partner underperformance – In the unlikely event of a partner departure from the project, the consortium will identify a suitable replacement. If this is not possible, the tasks allocated to the departing partner will be re-assigned to the rest of the consortium members.
- Failure to meet user requirements, w.r.t. software functionalities, *etc*. –
- Poor quality of data to validate the results and/or cooperation issues between different components of the platform –
- Underestimated development time, problems coping with new technology, unacceptable performance by individual partners –
- Key people involved in project activities depart the project –
- Northbound API information models and implementations delayed –

APPROVALS

The technical representatives appointed internally by each project Beneficiary are responsible for approving the test process for the components under their care, thus allowing the technical development to advance to the next stage.