AI-empowered Edge Cloud Continuum for self-aware cognitive computing environments

HORIZON Innovation Actions - HORIZON-CL4-2022-DATA-01-02

D2.5: Validation and Benchmarking plan

Delivery date: 06/2023

Dissemination level: Public

Project title:	**AI-empowered Edge Cloud Continuum for self-aware cognitive computing environments**
Duration:	**1 January 2023 – 31 December 2025**
Project website:	**https://www.cognifog.eu/**

Funded by the European Union under the Grant Agreement No. 101092968. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. The European Union cannot be held responsible for them.

Document information

Deliverable	D[#.#]: [Deliverable title]
Work Package	WP2
Task(s)	T2.3 Definition of Validation and Benchmarking plan
Type	Report
Due Date	M06, 06/2023
Submission Date	30/06/2023
Document Lead	THALES
Contributor(s)	ALL
Internal Review	LMS INTRA

Document history

Version	Date	Changes	Contributor(s)
V0.1	03/05/2023	Initial deliverable structure	THALES
V0.2	17/05/2023	50% of the deliverable content	ALL
V0.3	15/06/2023	Full Draft	ALL
V1.0.1	20/06/2023	Internal Review Version	INTRA
V1.1	22/06/2023	Revision after intra review	THALES
V1.1.1	23/06/2023	Internal Review Version	LMS
V1.2	26/06/2023	Final Version for Quality Review	eBOS
V2.0	29/06/2023	Final version for submission	THALES

Consortium

Logo	Partner	Country	Short name
	COMMISSARIAT A L ENERGIE ATOMIQUE ET AUX ENERGIES ALTERNATIVES FR	France	CEA
	ATOS IT SOLUTIONS AND SERVICES IBERIA SL ES	Spain	ATOS
	CYSEC SA CH Associated	Switzerland	CYSEC
	EBOS TECHNOLOGIES LIMITED CY	Cyprus	EBOS
	NETCOMPANY-INTRASOFT SA LU	Luxembourg	INTRA
	FUNDACIO PRIVADA I2CAT, INTERNET I INNOVACIO DIGITAL A CATALUNYA ES	Spain	I2CAT
	KENTYOU FR	France	KENTYOU
	PANEPISTIMIO PATRON	Greece	LMS
	TELEMATIC MEDICAL APPLICATIONS EMPORIA KAI ANAPTIXI PROIONTON TILIATRIKIS MONOPROSOPIKI ETAIRIA PERIORISMENIS EYTHINIS	Greece	TMA
	THALES France	France	THALES
	SIEMENS AKTIENGESELLSCHAFT DE	Germany	SIEMENS
	Urban Technology Alliance Association CH Associated	Switzerland	UTA

List of abbreviations

Abbreviation	Definition
Tb	**Terrabit** is a unit of digital information storage and transmission capacity. It represents a quantity of data equal to 1 trillion bits.
Tflops	**TeraFLOPS** measures a computer system's processing speed in trillions of floating-point operations per second, indicating its high computational power and performance.
SLA	**Service Level Agreement** is a contractual agreement between a service provider and a client that outlines the specific level of service that the provider is expected to deliver.
IoT	**Internet of things** refers to the vast network of interconnected physical devices, vehicles, and objects embedded with sensors and software, enabling them to collect and exchange data over the internet for enhanced functionality and smarter decision-making in various sectors.
RAN	**Radio Access Network** represents the essential component of a mobile telecommunications system that wirelessly links mobile devices to the core network, facilitating users' access to voice and data services.
API	**Application Programming Interface** is a set of rules, protocols, and tools that allows different software applications to communicate and interact with each other. APIs define how different software components should interact, specifying the types of requests that can be made and the data formats to be used.
QoS	**Quality of Service** refers to techniques and mechanisms used to ensure that a network or service meets predefined performance standards, prioritizes resources effectively, and delivers a consistent and satisfactory user experience.
CI/CD	**Continuous Integration/Continuous Deployment** is a software development practice that automates building, testing, and deploying applications, enabling faster and more frequent releases while maintaining quality and stability.
CLI	**Command-Line Interface** is a text-based interface where users can interact with a computer system or software by entering commands through a terminal or command prompt.
ROS	**Robot Operating System** is an open-source framework for building robotic systems that enables flexible and modular development and communication between different components of a robot.
OCI	**Open Container Initiative** is an open-source project that establishes industry standards for container formats and runtimes, ensuring compatibility and portability of containerized applications across different platforms.
ADAS	**Advanced Driver Assistance Systems** refers to a set of technologies implemented in vehicles to enhance safety and improve driving experience.
OS	**Operating System** is software that manages computer hardware resources and provides a platform for software applications to run on.

Executive summary

This document presents Deliverable D2.5 of Task 2.3, the initial version of the Benchmarking and Validation Plan for the COGNIFOG project. The plan outlines a comprehensive methodology for evaluating and comparing the performance of the COGNIFOG cognitive platform across different use cases. It aims to guide the validation process, ensuring effective testing and assessment of the platform's capabilities.

In line with the project objectives, this plan addresses potential risks and challenges that may arise during the validation process. It recognizes technical risks, resource constraints, and pilots-specific risks, and proposes risk mitigation strategies to anticipate and manage them. The validation execution process involves systematic testing and evaluation, focusing on the defined KPIs. To gather relevant data for analysis and performance assessment, monitoring and data collection will be an integral part of the validation process.

By implementing this Validation and Benchmarking Plan, the COGNIFOG project aims to ensure the robustness and effectiveness of the cognitive platform in real-world scenarios. The plan provides guidance for the project team to validate the platform's performance, identify areas for improvement, and drive innovation in the cloud-to-edge continuum. This deliverable plays a crucial role in the successful deployment and adoption of the COGNIFOG cognitive platform across various domains and use cases. It serves as a comprehensive guide for the validation and benchmarking activities, laying the foundation for the project's future advancements and achievements.

It is important to note that this document represents the initial version of the Benchmarking and Validation Plan, serving as a foundational framework for subsequent iterations. As the project progresses, the plan will be refined and enhanced based on the feedback and insights gained during the validation activities.

Table of Contents

List of Tables

1 Introduction

The cloud to edge continuum represents a paradigm shift in computing, where data processing and analysis occur seamlessly across a distributed network of cloud servers and edge devices. This approach enables real-time decision-making, reduced latency, and improved efficiency for a wide range of applications, such as Internet of Things, artificial intelligence, and autonomous systems.

The COGNIFOG project aims to harness the power of this distributed computing landscape by developing a cognitive platform that seamlessly integrates cloud and edge resources. This platform leverages advanced machine learning algorithms, natural language processing, and intelligent data analytics to enable real-time, context-aware decision-making in diverse environments.

The validation and benchmarking plan outlined in this deliverable will play a crucial role in ensuring the quality and performance of the developed cognitive platform. By employing rigorous testing methodologies, the functionality and correctness of the platform are verified and its performance and scalability in various cloud to edge continuum scenarios are assessed and validate its interoperability and compatibility with existing infrastructure and systems.

This validation and benchmarking plan identify any potential areas for improvement and optimization, enabling the enhancement of the platform’s capabilities and ensuring its seamless integration into real-world applications. The deliverables are validated according to established criteria, instilling confidence in the platform’s reliability and paving the way for its successful deployment in a wide range of industries and domains.

The COGNIFOG project's validation and benchmarking plan will provide a robust framework for assessing the quality and performance of the cognitive platform developed for the cloud to edge continuum. With a focus on functionality, robustness, performance, scalability, and compatibility, this plan will enable to refine and optimize the platform, ensuring its successful integration into the distributed computing landscape and driving innovation in a multitude of applications.

2 Objectives and Key Performance Indicators

2.1 Objectives and goals of the benchmarking plan

In order to assess and evaluate the different capabilities of the COGNIFOG platform, a specific set of Key Performance Indicators has been established that must be fulfilled. To ensure accuracy and transparency in the assessment process, this document contains a comprehensive description of each objective, along with a detailed explanation of how these objectives will be measured during the validation and benchmarking phase.

The platform's successful performance depends on essential aspects or functionalities that are represented by each KPI. Measurable and quantifiable indicators enable the effective and efficient gauging of the platform across various dimensions

The document provides a clear breakdown of each objective, outlining the desired outcome and the specific metrics that will be used to measure its attainment. These metrics encompass a range of factors, including performance, reliability, scalability, security, and resource utilization, among others. These metrics are carefully selected to capture the critical aspects of the platform's capabilities and to assess its ability to meet user requirements and industry standards.

During the validation and benchmarking phase, rigorous testing methodologies will be employed to collect data and evaluate the platform's performance against the defined KPIs. Various scenarios, workloads, and usage patterns will be simulated to comprehensively assess the platform's responsiveness, stability, and scalability. The collected data will be analysed against the predetermined metrics, allowing to objectively measure the platform's performance and identify any areas that require improvement.

The objectives and measurement methodologies are documented in this detailed manner, ensuring that the assessment process is systematic, repeatable, and accountable. This comprehensive approach enables a holistic understanding of the COGNIFOG platform's capabilities, enabling informed decisions and optimizations based on the identified strengths and areas for improvement.

2.2 Performance KPIs for benchmarking

2.2.1 Collaborative mission in urban area[i]

Table 1- Collaborative mission in urban area KPIs

KPIs	Description	Unit	Value
Deployment time (K1 & K4)	Edge and IoT deployment time: • New application (containers) should be deployed in few minutes to far edge boards • New application should be deployed in less than a minutes at edge • OS and platform should be deployed in less than an hour	Minute Minute Hour	**10** **1** **1**
Flexibility (K6)	• Deploy at least 100 micro applications that will be monitored by COGNIFOG platform tools	Count Count	**100** **10**

	• Deploy Platform with using 10 variability level domains (e.g., Running the platform in VMs or in k8s or using MQTT or ZeroMQ)		
Scalability (trial specific)	• Perform large scale monitoring and data collection from multiples devices (Up to 100 000 and growing at rate up to 500/h) • Perform large scale monitoring of OODA loops of actuators (Up to 1000 and growing at rate up to 10/h) • Process and aggregate the data at the edge to minimize the Edge to cloud bandwidth (Less than 10Tb/s) • Optimize the edge computing process to minimize the edge computing power needed at edge (Less than 100 Tflops by edge cluster)	Hour Count/hour Tb/s Tflops	**500** **10** **<10** **100**
Resiliency / Platform robustness (K7)	• Perform up to 100 pretesting scenarios to test platform robustness (connectivity disruption, server failure, intrusion) • Perform up to 5 full chaos engineering scenarios combining multiple hardware failure and connectivity issues	Count Count	**100** **5**
Time predictability (trial specific)	• Satisfy and monitor 10 hard real-time classes (Hard real-time means that tasks must be executed within a maximal time constraint) • Three SLA types of time classes must be defined (Strategy, Operative and Tactical)	Count Count	**10** **3**

2.2.2 E-health services in the Edge-Cloud Continuum

Table 2 - E-health services in the Edge-Cloud Continuum KPIs

KPI Name	Description	Unit	Value
Edge and IoT autonomous installation time (K1)	Provision and install an edge server with 2 stations within 3 hours.	Hour	**3**
Setup time of infrastructure (K2)	Same as above: edge server should also be provisioned within 3 hours.	Hour	**3**
Service establishment time (K3)	Vastly depends on area of installation. For an area with 4G connectivity establish a full installation within 3 hours.	Hour	**3**
Single touch orchestration (K4)	New stations should be provisioned within 1 hour	Hour	**1**
Device integration (K6)	For an edge server model, there is no comparison with the current centralized method.	Minute	**<=10**
Scale-up latency (K5)	Setup an added edge server and stations within 1 hour.	Hour	**1**
Data integrity / security (K8)	Moving data from cloud to edge provides better security. Ensure automated encrypted backup to cloud.	Hour	**Every 24**

Number of test users (trial specific)	Emulation of 100 stations connecting with a single edge server running the backend. Websockets should be maintained.	Count	**>100**
Number of test apps (trial specific)	The following apps could be integrated: • Mobile application connecting to backend. • Backend running on an edge server with a secured Linux distribution. • Automated backup application for encrypted backup of edge server database to cloud. • Third – party smart city dashboard. • Connectivity dashboard for overall topology. • CI/CD for backend on edge servers	Count	**5**
Reduce in latency of response (trial specific)	Stations connecting to edge servers is a new concept, however compared to connectivity with a central backend, the latency improvement is obvious.	Percent	**3-5% reduction**
Robustness (K7)	Develop the following communication tests • Station with edge server communication under (emulated) bad connectivity • Station with edge server under lack of any connectivity • Backend with Hub under lack of any connectivity • Stress test: emulation of multiple stations communicating with edge server.	Count	**>3 trials**

2.2.3 Automated Edge-Cloud Continuum for smart manufacturing

Some of Trial 3 baseline KPIs have not been addressed or measured yet. However, the investigation and implementation of methods to address those will be a main activity in the next WP2 period. During the WP2 activities the current KPIs will be measured and the projection for improvements will be addressed in the next version of the benchmarking and validation plan deliverable.

Table 3 - Automated Edge-Cloud Continuum for smart manufacturing KPIs

KPIs	Description	Unit	Value
Latency of response (trial specific)	Reduction of response times among the various use case components (computing, bandwidth limitation, frequency etc.)	ms	**TBD**
Infrastructure setup time (K2)	Reduction of the elapsed time to setup a specific cloud-to-edge-to-IoT infrastructure	Month	**12**
Service establishment time (K3)	Reduction of the elapsed time to setup a specific network service before it being operational	Month	**TBD**
Scale-up latency (K5)	Reduction of the time before scale-up operation is completed in case of a lack of resources	-	**TBD**
Edge CPU utilization (trial specific)	Reduce the amount of Edge CPU usage by deploying some processes on Cloud	Percent	**TBD**
sSecurity (K8)	To protect the network and sensor data from unauthorized access, manipulation, or disruption.	True/ False	**False**

3 Validation methodology and experimental design

3.1 Validation and benchmarking methodology

This section outlines the methodologies used to validate and benchmark the deliverables of the COGNIFOG project. It is essential to acknowledge that not all methods presented in the validation and benchmarking methodology are universally applicable to every use case. The approach to validating and measuring Key Performance Indicators (KPIs) may vary from trial to trial, taking into account the specific context and requirements of each case. The flexibility to adapt and tailor the methods to suit the unique characteristics of each trial is crucial in ensuring accurate and meaningful results.

The following table presents the validation methodologies for each KPI type:

Table 4 - Validation and benchmarking methodology

KPI	Methodology
K1: Edge and IoT[ii] autonomous installation time	• Set up the hardware and make all necessary connections. • Launch the automated installation process for the platform. • Measure the time it takes to complete the platform installation. • Use a continuous integration and continuous deployment (CI/CD) tool such as ATOS industrial grade DEVOPS to create a pipeline that builds and deploys containerized applications to a target environment such as Kubernetes[iii]. • Configure the pipeline to trigger on every code change in a repository that contains a Dockerfile. • Use a tool such as Prometheus to collect metrics from Kubernetes cluster, such as pod creation time, pod readiness time, and service availability time. • Use a tool such as Grafana to visualize and analyse the metrics collected by Prometheus and calculate the time of application deployment as the difference between the code change time and the service availability time.
K2: Setup time of infrastructure	• Set up cloud/online server infrastructure and networking configurations • Launch automated installation process of the required tools and packages to prepare the execution of the platform • Use a continuous integration and continuous deployment (CI/CD) platform • Upload the software services to be deployed in the platform • Configure and execute the deployment pipelines • Monitor the execution and the health of the deployed services using a tool such as Portainer • Measure installation time of Operating System on Cloud and Edge nodes • Measure installation time of the Container as a Service runtime. • Measure installation time of COGNIFOG components framework. Once the different nodes are connected to each other (COGNIFOG ecosystem is set in place) and all the COGNIFOG components are deployed, the infrastructure setup will be completed

K3: Service establishment time	• Slice instantiation and activation time. However, the final instantiation time of the new slices does not depend entirely in the Slice Manager, but it also involves the core instantiation stages and the speed of the RAN to cope with the changes. • In terms of the intelligent part, the decision of the inference can be computed using tools to monitor and measure the time, it takes from the model to generate an answer, once it has been fed with all the information surrounding the network.
K4: Single touch orchestration	• Configure the application's desired state and non-functional requirements. • Utilize the COGNIFOG platform to automatically read and understand these requirements. • The platform automatically places and deploys the application based on specified constraints. • Ensures that the platform optimize resources allocations and meets the application's non-functional requirements. • Use COGNIFOG platform to automatize applications orchestration to maximize the QoS.
K5: Scale-up latency	• Measure the time required to onboard a new infrastructure within a COGNIFOG ecosystem already set up. • Measure the time required to onboard a new device within a COGNIFOG ecosystem already set up. • Measure the time elapsed since a scale-up order is given with the time it takes for the actual state of the application to comply with it.
K6: Device integration time	• Choose a range of representative devices or sensors that are commonly integrated into the COGNIFOG platform. These devices should be representative and cover different types, protocols, and communication interfaces. • Outline a standardized integration process, including the steps and actions required to integrate a device into the cognitive platform. This process should cover device discovery, authentication, configuration, network connection and software installation. • Set up the hardware and make all necessary connections. • Record the time it takes to complete each integration process for the representative devices. Start the timer when the integration process begins and stop it when the device is successfully integrated and operational.
K7: Resiliency	RAN controller operation • The system uses error control protocols to guarantee to deliver the information correctly or to generate an alert when the data is not compatible. IoT edge Gateway • Establish forwarding policies for incoming traffic • Set up the default configuration of outbound network interfaces • Monitor the connection performance of active interfaces • If necessary, apply new configuration according to policies and network performance Application and platform: • Monitor the health of applications (response time, response errors, container state, etc.).

	• Compare the desired state of the services (containers) with its actual state. • Perform Chaos Engineering scenarios: Network bandwidth, latency, disconnection issues and hardware failure.
K8: Data integrity / security	**1. Requirements Analysis:** • Identify the specific data integrity and confidentiality requirements for COGNIFOG based on the project's objectives and stakeholder needs. • Define the desired security goals, such as data integrity, encryption, access control, and secure communication. **2. Test Planning**: • Develop a comprehensive test plan that outlines the testing objectives, scope, and test scenarios. • Identify the types of data to be tested (sensitive, personal, confidential) and the relevant security measures to be validated. • Determine the testing tools, techniques, and methodologies to be used. **3. Test Design:** • Design test cases that cover different aspects of data integrity and confidentiality. • Define the expected outcomes for each test case based on the security requirements and specifications. • Identify the test data sets, including both valid and invalid data, to simulate various scenarios. **4. Test Execution:** • Execute the defined test cases, ensuring proper test environment setup. • Monitor and record the results of each test, including any anomalies, errors, or failures encountered. • Document the steps followed during the testing process for future reference. • **Data Integrity Testing:** – Validate the accuracy and completeness of identified sensitive data within COGNIFOG. – Perform tests to verify that data remains intact and unaltered during storage, transmission, and processing. – Check for any data corruption, loss, or unauthorized modifications. • **Data Confidentiality Testing:** – Verify the effectiveness of security measures in protecting sensitive data from unauthorized access. – Test encryption and decryption mechanisms to ensure data confidentiality. – Evaluate access control mechanisms to ensure that only authorized users can access sensitive data. **5. Vulnerability Assessment:** • Conduct vulnerability assessments and penetration testing to identify potential security weaknesses and vulnerabilities.

	• Test the resilience of the system against common security attacks, such as data breaches or unauthorized access attempts. • Address any vulnerabilities found by implementing appropriate patches, fixes, or security measures. **6. Performance Testing:** • Assess the performance impact of the implemented security measures on data integrity and confidentiality. • Validate that the security measures do not significantly degrade the overall system performance

3.2 Definition of tools and framework to be used for validation

This section provides an overview of the tools and frameworks that will be used for the validation process in the COGNIFOG project. These tools and frameworks are essential for ensuring the thorough evaluation and assessment of the functionality, performance, scalability, reliability, compliance, user experience, security, and accuracy of the cognitive platform. It should be noted that not all of the tools and frameworks listed in the validation section may be applicable or useful for every use case. Recognizing this, an activity is planned for the upcoming WP2 period to identify and select the appropriate tools that align with the specific requirements of each use case. This process ensures that the chosen tools and frameworks are well suited for the validation activities, maximizing their effectiveness and relevance in achieving accurate and meaningful results.

Table 5 - Benchmarking and validation tools and framework

KPI	Technologies tools and framework
K1: Edge and IoT autonomous installation time	• Continuous integration frameworks: Gitlab CI, Jenkins • Monitoring tools: Grafana, Prometheus • Supervision stack: Elasticsearch, Kibana, Loki, fluentbit • Kubernetes cluster: Rancher RKE2/K3S • Robot operating system framework: ROS • Document storage for ROS: MongoDB
K2: Setup time of infrastructure	• Code repository: GitHub/Gitlab • Cloud native container registry: Harbor • OCI containers • Continuous integration: Jenkins, Gitlab CI and Github Action • Container management tool: Portainer • Automatic code review tool: SonarQube
K3: Service establishment time	• Scripting to measure the execution time.
K4: Single touch orchestration	• Container orchestrator: Kubernetes/K3S • Kubernetes extension to work at the edge: Kubeedge • Cluster federation: OCM/Rancher Fleet
K5: Scale-up latency	These tools will give information about the desired state and the actual state of the applications, the time difference between both states will be the scale-up latency. • Kubernetes API

	• Multi cluster management tools and GitOps framework: Rancher Fleet / ArgoCD / OCM • Monitoring tool: Prometheus
K6: Device integration time	• Framework for robot software development: ROS • Microcontroller SDK: (Arduino IDE, MCUxpresso, Pico SDK, PlatformIO) • Device twin in Kubernetes: Kubeedge device twin
K7: Resiliency	IoT edge Gateway • Network speed measurement: Speedtest CLI • Scripting to execute selection/bonding of network interfaces Apps and platform resiliency: • Cloud Native Chaos Engineering framework (Chaos Mesh and Litmus) • Kubernetes API (state of the services and containers) • Prometheus (applications response time, response errors…)
K8: Data integrity / security	• Gitlab CI/CD • Identity and access management solution: Keycloak • Crypto backends: OpenSSL, TPMs, HSMs • Vulnerability scanning: Trivy • Secrets management: Hashicorp Vault • Threat identification framework: STRIDE

3.3 Potential risks and challenges

3.3.1 Technical risks

This section addresses potential technical issues that may arise during validation, including software bugs or compatibility challenges between building blocks. It explains the strategies and measures put in place to mitigate identified risks and challenges, ensuring that the project team is well prepared to handle potential issues effectively. By proactively addressing these risks, the project team is equipped to effectively address and overcome any technical obstacles that may emerge throughout the project lifecycle.

Table 6 - Validation technical risks

Risks	Risk Description	Risk Mitigation
Limited Availability of Datasets/Schemas due to governmental or corporate policy sensitivities	The limited availability of datasets can significantly hinder the development and performance of cognitive models. Without access to diverse and representative data, the models may lack the necessary training to accurately understand and analyse real-world scenarios. This can lead to suboptimal performance, reduced accuracy, or biased outcomes when making decisions or providing insights.	• If access to real-world datasets is limited, the team can consider generating synthetic data that closely mimics the characteristics and patterns of the desired data. • Concentrate on utilizing publicly available datasets that align with the project's objectives. Open datasets, research repositories, and data-sharing initiatives can provide valuable resources • Ensure that the project adheres to ethical guidelines

		and practices when accessing and using sensitive data. Implement appropriate consent mechanisms and anonymization techniques
Complex integration / components' diversity	The technical building blocks provided by COGNIFOG partners could not cover all the use case requirements: • Interoperability between components • Feature completeness	• Apply fully defined, well designed and documented data model • Identify the missing component • Identify the partner able to provide it (by adopting the efforts and resources) • Identify an extern subcontractor that can help the consortium solving the problem • Fill missing features with alternative software
COGNIFOG components' maturity	The technical building blocks provided by COGNIFOG partners have different TRL level	• Well define the APIs the functionality of each building block and its role in the use case scenario • Well define the dataset used • Well define the functionality of each building block and its role in the use case scenario
Software Bugs	Bugs can range from minor issues that affect user experience to critical defects that affect the functionality and reliability of the system. Software bugs can arise due to coding errors, inadequate testing, incomplete requirements, or compatibility issues.	• Code Reviews and Static Analysis: Conduct thorough code reviews to identify potential bugs and improve code quality. • Bug Tracking and Management: Utilize a bug tracking system, such as Jira or Bugzilla, to log, track, and prioritize identified bugs • Comprehensive Testing: Implement a robust testing strategy that includes unit testing, integration testing, system testing, and regression testing. • Continuous Integration and Continuous Deployment (CI/CD): Implement CI/CD pipelines to automate the build, testing, and deployment processes.

3.3.2 Pilot's specific risks

This section focuses on identifying and addressing potential risks that are specific to the pilot phase of the project. Pilots involve real-world testing and implementation of the cognitive platform in a limited deployment, allowing for valuable insights and feedback. However, certain risks unique to this phase may arise, and it is essential to recognize and mitigate them to ensure successful pilot execution. This section outlines the specific risks associated with the pilot phase and presents strategies and measures to mitigate these risks effectively.

Collaborative mission in urban area:

Table 7 - Collaborative mission in urban area specific risks

Risks	Risk Description	Risk Mitigation
Limited number of devices	Thales has a limited number of devices and cannot deploy them on a large scale in geographically dispersed locations.	Use hardware in the loop simulation tools for scaling representation.
Components availability	In the context of collaborating with various partners, it will be necessary to integrate different technical components. This implies that the components are delivered and available.	Define and follow a well-organized development plan for the software components with concrete deadlines and software release cycles. Find alternative components available with comparable functionality. This solution is not always possible.
Cloud provider cost	Using resources from a cloud provider can be expensive if there is intensive utilization of resources.	Using a private cloud instead of a public cloud with pre-existing resources requires additional costs in terms of administration and management.
Components compatibility	The risk of component interoperability poses challenges in ensuring seamless integration and functionality.	It is necessary to adhere as closely as possible to standards and reach agreement on the inter-component communication layer. This will be done by applying fully defined, well-designed and well-documented data models and interfaces and using common communication protocols and data structures (JSON structured messages, REST APIs, Message Broker, etc.)

E-health services in the Edge-Cloud Continuum:

Table 8 - E-health services in the Edge-Cloud Continuum specific risks

Risks	Risk Description	Risk Mitigation
Limited connectivity to internet	Edge servers will need to be able to connect to the central hub	Design with this scenario in mind and test cases without internet connectivity.

Complex integration	Multitude of layers and devices. Websockets should work.	Use well-documented APIs and test each integrated layer before incorporating.
Security issues with API's	APIs must be secured	Use well documented authentication methods to access endpoints
Privacy issues with API's	APIs in some cases will need to be less accessible or anonymize data	Authentication methods should be role-based. Security by design.

Automated Edge-Cloud Continuum for smart manufacturing

Table 9 - Automated Edge-Cloud Continuum for smart manufacturing specific risks

Risks	Risk Description	Risk Mitigation
Components compatibility	Achieve interoperability of the different HW and SW assets	Communication layer to be well addressed and standards integration
Connectivity	Edge and cloud implementation	Offline and online validation
Security and privacy	Adverse network access and data loss	All interfaces must be secure, tested and validated and authentication methods should be integrated

3.4 Validation execution process

The validation process for the three trials combines a general validation model with trial specifics and is carried out within WP5. It consists of three aspects mapped to the work packages tasks.

3.4.1 Operational plan common to all pilots

The comprehensive operational management of the COGNIFOG project, as outlined in Task T5.1, establishes the overarching framework for the trials. Its primary objective is to ensure that the pilots collectively exercise all components of the COGNIFOG platform. These pilots serve multiple purposes, not only supporting the demonstration of various COGNIFOG components and integration cases but also providing valuable insights into the performance of use cases under real-life operational conditions.

The operational plan, shared across all trials, plays a vital role in operationalizing, monitoring, and managing the pilots. It defines the necessary strategies and procedures for establishing an effective environment to conduct the experiments successfully. Additionally, it facilitates the collection of feedback and experimental data, which form the foundation for assessing and validating the proposed KPIs. By incorporating the operational plan, the project ensures that the trials encompass the full breadth of the COGNIFOG platform, allowing for a comprehensive evaluation of its capabilities.

The plan's focus on creating an optimal environment for conducting experiments and collecting relevant data underscores its importance in validating and benchmarking the platform's performance across diverse use cases.

3.4.2 Pilots roll-out

While each pilot within the COGNIFOG project possesses unique characteristics, a common operational plan will be implemented for all pilots, following a structured and staged path. This approach ensures consistency and facilitates effective coordination across the different pilot implementations.

Before deployment and testing in the pilots, the instantiations will undergo integration and testing in controlled laboratory conditions, as specified in Task 4.3. This initial phase allows for thorough integration and validation of the technology solutions developed within Work Package 4 (WP4).

The pilots serve as crucial platforms for assessing various aspects, including real-time data processing, heterogeneous orchestration of distributed computing resources, fault tolerance, and cyber-security protection mechanisms. Additionally, the pilots provide an opportunity to validate the overall framework of the COGNIFOG project. To ensure consistency and adherence to best practices, the pilots will follow the plan and guidelines outlined in Task 5.1, incorporating the lab-integrated technology solutions from WP4. The pilots will also adapt to the specific requirements of each use case and the site where the setup phase will be executed.

The execution of the pilots will occur in two distinct phases. The initial phase will focus on preliminary setup efforts, ensuring that the necessary infrastructure and configurations are in place. Subsequently, the second and final phase will involve more detailed and comprehensive experimental work, aiming to thoroughly evaluate the platform's performance and functionalities.

Throughout the setup and operational management of the pilots, detailed documentation will be maintained in Deliverable D5.1. This documentation will capture the setup procedures, operational guidelines, and any necessary adjustments made to accommodate the requirements of each stakeholder's premises. Furthermore, the pilots will be continuously monitored, generating logs that will facilitate the assessment of KPIs relevant to the project's objectives.

3.4.3 Evaluation of results technical support and evaluation guidelines

3.4.3.1 Evaluation Framework

A standardized evaluation framework is implemented to assess the technical results of the COGNIFOG project under realistic conditions at the pilot sites. This framework encompasses various aspects of the pilot, including system performance, security, developer perspectives, and end-user experiences.

The evaluation process involves collecting data during the pilot execution to gauge the achievement of KPIs and determine their effectiveness. If the KPIs are not met or fail to reach satisfactory levels, feedback will be provided to Work Package 4 (WP4). WP4 will then analyse the limitations and bottlenecks identified and propose new features or improvements to address them.

While operational and functional KPIs will be specific to each pilot, there will be common KPIs that assess the capacity of COGNIFOG in areas such as time predictability, fault tolerance, energy management, security, and control-oriented meta-orchestration over infrastructures. These common KPIs will be uniformly implemented across all pilots, ensuring globally consistent measurements and diagnostics.

The technical results obtained from the pilots will be systematically collected and analysed according to the specified methodology in section 3.1. Additionally, specific tools defined in section 3.2 will be employed to facilitate the analysis process. This comprehensive approach enables a thorough assessment of the project's technical outcomes, providing insights into performance, functionality, and user satisfaction.

Alongside the suggested methodology, certain "building blocks" can play a crucial role in validating specific KPIs, particularly the applications related to runtime monitoring which will help to validate resiliency.

The runtime monitoring proposed by CEA is a Polygraph-oriented method. The approach is based on defining accurately what is a ”correct” behavior of an application. To this end, the Polygraph model is used[iv]. The timing properties derived analytically from the Polygraph design are used to instantiate a set of software monitors. The monitors are compiled into the application and inserted into the application execution flow; they track dataflow-relevant events. In previous works, the approach was applied on some applications representative of real-time computing loads associated with advanced driver assistance systems (ADAS). For example, let's consider a Camera actor that generates image frames at a fixed rate e.g. 15 frames per second. These frames are analyzed by two parallel sub-chains. On one hand, lane markings are detected on every frame using classic computer vision algorithms, and illustrates a time-critical chain. On the other hand Objects detection is performed using a deep neural network, and processes only a subsampled set of the input frames, in a in a soft real-time way. The Display sink actor serves as visual output, and additionally monitors the end-to-end reaction deadline.

During the execution of this example, Polygraph dataflow relevant events were all monitored. These events were: actor job start, job end, message sent, message received. Using these events, the execution times of each actor job was calculated as well as the input and output latency of each actor (time of arrival of input data, and time of departure of output data, relative to original input timestamp).

For hardening purposes, we will leverage the Xanthos operating system in order to benefit of its safe by design and low footprint code base. This way, we will maximize the application reliability by being functionally verified offline by Polygraph tools and monitored online on a hardened OS.

In addition, the technological component "Skipper" will enhance the Polygraph-oriented approach to provide a deterministic approach for modeling, analysis, and optimization of execution in an Edge to Cloud continuum. One key objective is to converge deterministic and statistical approaches in order to offer deterministic guarantees for the safety- critical part and statistical guarantees for the non-safety-critical part of systems using the same methods and models. Based out of Polygraph, Skipper will ensure the performance and reliability of systems in Cloud Native ecosystems from embedded to Cloud, in a mixed-criticality approach that takes into account the specific requirements of both safety-critical and non-safety-critical systems.

3.4.3.2 Refinement process

Agile methodology[v] through various processes will be applied to ensure proper operations and collaboration between the involved partners to achieve efficient integration of all the components. Following the development activities and according to the readiness of the services, integration biweekly meetings will be established focusing to promote close collaboration and maintaining alignment among all technical members. Additionally, this approach will facilitate the smooth integration since it offers close monitoring of the progress and the status of the working activities. Following those practices an issue tracking repository will be held to help to the quick identification of the problems and their prompt solution.

Also, an integration matrix will be circulated to serve as a comprehensive guide, outlining the connections and dependencies between the different components. By the time all the systems' interactions and their interfaces are defined, integration templates will be provided to the technical teams, where the concrete steps followed during the integration testing phase will be reported, streamlining the integration process. The deployment views of the COGNIFOG solution will be carefully designed in separated representations having one general view and three other views each one corresponding to the pilots to establish a clear understanding of the involved services, their interfaces, and communications.

The Integration Plan will be organized into multiple phases, typically spanning 2 to 3 months, enabling teams to work on specific objectives within the defined timeframe. The integration process starts with simple activities and progressively advances towards more complex workflows considering the identified interactions on the integration matrix and report the results on the integration templates. It is also crucial to initiate integration with small-scale tests and gradually scale up to verify performance and scalability aspects. The first phase involves bilateral testing activities focusing on connectivity and communication aspects between the services and identifying and resolving any encountered issues.

Moving forward, the second phase will include unit tests results for the components and bilateral tests in terms of communication and exchanging simulated data based on the defined data models. Any functional issues that will arise during this phase will be thoroughly recorded for resolution to be fixed for the next integration round. In the third phase, the purpose will be to test complex workflows, communications, and data exchange involving multiple components. This encompasses employing final data models and real-life use-case scenarios, conducting end-to-end tests, as well as stress and scalability tests to evaluate system performance. Also, if any performance or functional issues appear will be noted for subsequent encounter. In the final phase, the scenarios of the use-cases and pilots will be meticulously tested to validate the proper operation and workflow of all the involved components. Extensive end-to-end tests will be conducted to ensure seamless functionality. Through this systematic approach, beginning with small-scale tests and gradually progressing towards comprehensive scenarios, the technical partners can effectively address integration challenges and achieve a well-integrated system.

4 Conclusions

In conclusion, the Benchmarking and Validation Plan for the COGNIFOG project provides a comprehensive framework for evaluating and comparing the performance of the cognitive platform across various use cases. Throughout this document, a systematic testing and validation methodology has been outlined, key performance indicators have been identified, and the tools and frameworks to be employed have been defined.

This plan enables the effective assessment of the performance of the COGNIFOG platform and ensures its robustness and effectiveness in real-world scenarios. The validation process, accompanied by monitoring and data collection, will provide valuable insights and metrics for analysis and performance assessment.

Moreover, the plan acknowledges and addresses potential risks and challenges, presenting risk mitigation strategies to proactively manage technical risks, resource constraints, and pilots-specific risks. These challenges are anticipated and addressed, enhancing the validation process and ensuring its success. As an iterative plan, this initial version is recognized as subject to refinement and enhancement as the project progresses. Feedback and insights gained during the validation activities will drive continuous improvement and optimization of the cognitive platform's building blocks.

Ultimately, the successful execution of this Validation and Benchmarking Plan will contribute to the deployment and adoption of the COGNIFOG platform across diverse domains and use cases. The performance is rigorously evaluated, areas for improvement are identified, and innovation is driven, realizing its full potential in revolutionizing the cloud-to-edge cognitive platform domain.

This document serves as a guiding framework, empowering the project team to carry out comprehensive validation and benchmarking activities. By adhering to this plan, we can validate the platform's performance, make informed decisions for optimization, and ensure its readiness for real-world deployment and adoption.

The knowledge gained from this validation process positions well to drive innovation, foster collaboration, and contribute to the advancement of the COGNIFOG project. Through collective efforts, the COGNIFOG cognitive platform can be successfully integrated, transforming the cloud-to-edge continuum and shaping the future of cognitive computing.

References

[i] Militano, Leonardo, Adriana Arteaga, Giovanni Toffetti, **and** Nathalie Mitton. "The Cloud-to-Edge-to-IoT Continuum as an Enabler **for** Search **and** Rescue Operations." Future Internet 15, no. 2 (2023): 55. https*://doi.org/10.3390/fi15020055.*

[ii] Aguiari, Davide. "Exploring Computing Continuum **in** IoT Systems : Sensing, Communicating **and** Processing at the Network Edge." Phdthesis, Sorbonne Université ; Università degli studi (Cagliari, Italie), 2021. https*://theses.hal.science/tel-03382756.*

[iii] Carrión, Carmen. "Kubernetes as a Standard Container Orchestrator - A Bibliometric Analysis." Journal of Grid Computing 20, no. 4 (December 6, 2022): 42. https*://doi.org/10.1007/s10723-022-09629-8.*

[iv] Dubrulle P., Gaston C., Kosmatov N., Lapitre A., Louise S. (2019) A Data Flow Model with Frequency Arithmetic. In: H¨ahnle R., van der Aalst W. (eds) Fundamental Approaches to Software Engineering. FASE 2019. Lecture Notes in Computer Science, vol 11424. Springer, Cham. https://doi.org/10.1007/978-3-030-16722-6 22

[v] Oyelami, Oyewale Adedayo, Alexander Poth, Johannes Hintsch, **and** Andreas Riel. "Quality Assurance **and** Traceability **in** Containerized Continuous Delivery Process," 368, 2019. https*://doi.org/10.1007/978-3-030-28005-5_28.*