A Holistic Approach to Sustainable, Digital EU Agriculture, Forestry, Livestock and Rural Development based on Reconfigurable Aerial Enablers and Edge Artificial Intelligence-on-Demand Systems

CHAMELEON D.4.1 CHAMELEON Plug-n-Play Platform v1

Work package	WP4: CHAMELEON Multipurpose Onboard
Task	Task 4.1: Plug-n-play platform
Authors	**George Stivaktakis, Kapouranis Dimitrios, Stylianos Klados – Adrestia R&D**
Dissemination level	Public (PU)
Status	Final
Due Date	31/12/2023
Document date	29/12/2023
Version number	1.0
Funded by the European Union	Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency. Neither the European Union nor the European Research Executive Agency can be held responsible for them.

Revision and history chart

Version	Date	Main author	Summary of changes
0.1	17/08/2023	Kapouranis Dimitrios	Draft outline
0.2	04/12/2023	Stivaktakis George	Initial Draft Version v0.2
0.3	05/12/2023	Stivaktakis George	Initial Draft Version v0.3
0.4	05/12/2023	Stylianos Klados	First Draft Version 0.4
0.5	11/12/2023	S.Huriez S. Cayla	Peer review
0.6	13/12/2023	Stylianos Klados	First Pre-Final Version
1.0	18/12/2023	Christiana Themistokleous	Final Version

Table of contents

List of abbreviations and acronyms **4**

1 Executive Summary **5**

2 Introduction **6**

2.1 Structure of document 6

3 Chameleon Plug-n-Play Platform Design **7**

3.1 Objectives 7

3.2 Technologies (State of the Art) 7

3.2.1 Container-Based Deployments Without Orchestration 7

3.2.2 Orchestation Tools 8

4 Initial Version of the Plug-n-Play Platform **10**

4.1 Chameleon architecture 10

4.2 Plug and play architecture 11

4.2.1 Server 11

4.2.2 Client 12

4.2.3 Worker 13

4.3 Sequence of Actions 13

5 Conclusions and Next Steps **16**

6 References **17**

Index of figures

Figure 1: Chameleon Architecture 10
Figure 2: Internal Kubernetes Architecture 12
Figure 3: Plug-n-Play Processing Sequence Diagram 14

LIST OF ABBREVIATIONS AND ACRONYMS

Abbreviation	Meaning
UAV	Unmanned Aerial Vehicle
PnP	Plug-n-Play
VPN	Virtual Private Network
WiFi	Wireless Fidelity

1 EXECUTIVE SUMMARY

This document presents the first version of the Chameleon Plug-n-Play Platform and the current implementation status. The choices that led to the initial design of the platform are presented. Furthermore, the initial platform architecture is presented and described, as well as its relationship with other Chameleon components. The Plug-n-Play Platform is a core component of the Chameleon Architecture as it connects the cloud and the UAVs via a gateway and enables the deployment of bundles. Therefore, the worker part of it is integrated into the UAVs and the server part, which is in the cloud, communicates with the Chameleon Store to deploy the selected bundle. Moreover, sequence diagrams with an example use case are showcased. Finally, the next implementation steps are presented.

2 INTRODUCTION

This document serves as documentation for the initial version of the Chameleon Plug-n-Play Platform. CHAMELEON is working on developing a software solution that will enable the deployment of versatile software packages on a variety of unmanned aerial vehicles (UAVs) to meet the requirements of a wide variety of sectors and purposes. As a result, a seamless integration between the UAVs and their deployment infrastructure is required. The Plug-n-Play Platform will serve as a platform that will consistently deliver the software that the UAVs require in order to perform their actions. Moreover, this platform will ensure the integrity and persistence of the software that needs to be delivered to the UAVs.

For reliable software exchange between the Platform and the UAVs, the design and initial implementation of the Platform is based on the latest container-based orchestration software, which ensures the persistence of the software. The State of the Art of those technologies and the choice of the orchestration software are elaborated on this document.

Before elaborating upon the initial implementation of the Plug-n-Play Platform, its role and importance within the CHAMELEON Architecture is presented. The deliverable continues by presenting and elaborating on the initial version of the Plug-n-Play Platform. Moreover, the main operational process of the Plug-n-Play Platform is showcased and elaborated. It must be noted that this initial implementation version might be subject to change as the CHAMELEON project evolves and other components of the overall CHAMELEON Architecture are developed.

2.1 STRUCTURE OF DOCUMENT

The rest of the document is structured as follows:

- Section 1 presents the executive summary of this document.
- Section 2 serves as an introduction to this document.
- Section 3 showcases the state of the art that the Plug-n-Play Platform is based on.
- Section 4 provides implementation details for the initial version of the Plug-n-Play Platform.
- Section 5 concludes the document and presents the next steps.

3 CHAMELEON PLUG-N-PLAY PLATFORM DESIGN

The CHAMELEON Plug and Play Platform, includes the CHAMELEON Server, Gateway, and the Worker (UAV). Container-based development ensures application consistency across various computing environments in the CHAMELEON Plug-n-Play (PnP) Platform. However, managing these container deployments without orchestration presents significant challenges. To address these challenges, the CHAMELEON PnP Platform incorporates orchestration tools. These tools provide automated scalability, adjust container counts in response to workload changes, and offer self-healing capabilities to maintain continuous operation. A thorough analysis of orchestration tools, including Docker Swarm [1], Apache Mesos [2], and Kubernetes [3], guided the selection process for the CHAMELEON PnP Platform.

3.1 OBJECTIVES

The primary objectives of the CHAMELEON Plug and Play Platform are:

- Performance: By utilizing container-based development, the platform aims to ensure that applications function efficiently across various computing environments.
- Management of Containers: Addressing the complexities of managing container deployments without orchestration and simplifying this process.
- Ensure Availability: By implementing self-healing capabilities, the platform ensures continuous operation and system availability.
- Scalability: An orchestration tool integrated into the platform makes scaling easier (an important factor in a UAV-based environment).

3.2 TECHNOLOGIES (STATE OF THE ART)

In this section, the technological aspects of container-based development are explored, emphasizing the challenges associated with managing such deployments without orchestration, particularly in UAV-centric operations like the CHAMELEON project. The benefits offered by orchestration tools are then highlighted, showcasing how they address these challenges with features such as automated scalability and self-healing capabilities. Lastly, it concludes with an analysis of various orchestration tools and an explanation behind selecting Kubernetes as the preferred orchestration tool for the CHAMELEON PnP Platform.

3.2.1 CONTAINER-BASED DEPLOYMENTS WITHOUT ORCHESTRATION

In a container-based deployment without orchestration, applications are encapsulated within containers. These containers offer an isolated environment for each application, ensuring that they run consistently across different computing environments. This is particularly advantageous because containers are lightweight, require fewer resources, and can start up much faster. In this model, each container runs a single application or service, and the dependencies for that service are bundled with it. This setup greatly simplifies deployment and version control, as each container can be updated, deployed, and rolled back independently. However, this approach was not chosen for the CHAMELEON PnP Platform because managing these containers manually becomes increasingly complex as the number of containers grows.

Without an orchestration tool, tasks like deploying new containers, updating existing ones, monitoring their health, ensuring they're running on suitable hosts, and scaling them based on load; all need to be done manually. This method proves especially problematic for UAV-centric projects like CHAMELEON, where such manual management is untenable. To overcome the above challenges, significant benefits are introduced by integrating an orchestration tool:

- *Automated Scalability*: The tool dynamically adjusts the number of containers in response to workload changes, ensuring efficient resource utilization.
- *Resource Optimization*: This leads to superior optimization, allowing more applications to operate efficiently on the same hardware, minimizing resource waste.
- *Self-healing Capabilities*: With continuous monitoring, the platform automatically replaces or restarts containers that fail or don't meet health criteria, ensuring high availability and resilience.
- *Load Balancing and Traffic Management*: The system effectively manages load distribution and traffic, maintaining system stability and performance through even workload and network traffic distribution.

Recognizing these advantages, the decision was made to employ an orchestration tool for the CHAMELEON PnP Platform, significantly enhancing its efficiency and reliability.

3.2.2 ORCHESTATION TOOLS

In the following section, a brief analysis will be conducted on some orchestration tools, namely Docker Swarm, Apache Mesos, and Kubernetes. This analysis aims to evaluate their capabilities and suitability for the CHAMELEON PnP Platform, ultimately guiding the decision-making process in selecting the most appropriate tool for efficient and effective container management.

DOCKER SWARM

Docker Swarm is a container orchestration tool that offers native clustering capabilities and turns a group of Docker hosts into a single, virtual Docker host. It's known for its ease of setup and straightforward operations, making it a popular choice for simpler container orchestration needs. However, Docker Swarm was not the ideal choice for the CHAMELEON PnP Platform. While it excels in simplicity and ease of integration with existing Docker environments, it falls short in handling more complex, large-scale deployments and complicated orchestration tasks. Its limited capabilities in handling complex networking and fewer self-healing mechanisms make it less suitable for a project as dynamic and demanding as the CHAMELEON PnP Platform, which requires a more comprehensive and scalable solution.

APACHE MESOS

Apache Mesos is a cluster manager that provides efficient resource isolation and sharing across distributed applications or frameworks. It excels in large-scale cluster management and is capable of running not just containerized workloads but also non-containerized and data-intensive applications. Despite its strengths, Apache Mesos was not selected for the CHAMELEON PnP Platform. Its complexity in setup and management poses a significant challenge, especially for teams looking for a more straightforward approach to container orchestration. Additionally, Mesos might require extra components to handle container orchestration tasks effectively, which could lead to a more complex system architecture than necessary.

KUBERNETES

Kubernetes is an open-source platform for automating container operations such as deployment, scaling, and management. It stands out for its robustness and flexibility, making it a highly preferred solution for complex container orchestration. The decision to select Kubernetes as the orchestration tool for the CHAMELEON PnP Platform was influenced by its robust automated scaling features, enabling dynamic resource adjustment in line with operational demands a huge factor for the real-time responsiveness required in UAV operations. Furthermore, Kubernetes boasts a vast ecosystem and extensive community support, offering a plethora of tools and resources. This selection's implications and in-depth utilization will be further elaborated in the implementation section.

4 INITIAL VERSION OF THE PLUG-N-PLAY PLATFORM

This chapter delves into the architecture of the CHAMELEON project, analysing the synergy between its three primary components: the Cloud Platform, the Gateway, and the UAVs. It explores the role of the Cloud Platform as the central hub, managing service bundles and ensuring efficient post-processing. Afterwards exploring the Gateway, enhanced by Kubernetes, acting as a link between the Cloud Platform and the UAVs, streamlining data transmission and service deployment. The UAVs, integral to the project's operational efficacy, are tasked with receiving and executing these bundles. This narrative aims to provide a detailed understanding of each component's objectives, infrastructure, and their roles within the CHAMELEON's PnP architecture.

4.1 CHAMELEON ARCHITECTURE

The architecture of the CHAMELEON project, showcased in *Figure 1*, consists of three main components: the Cloud *Platform*, the *Gateway*, and the UAV.

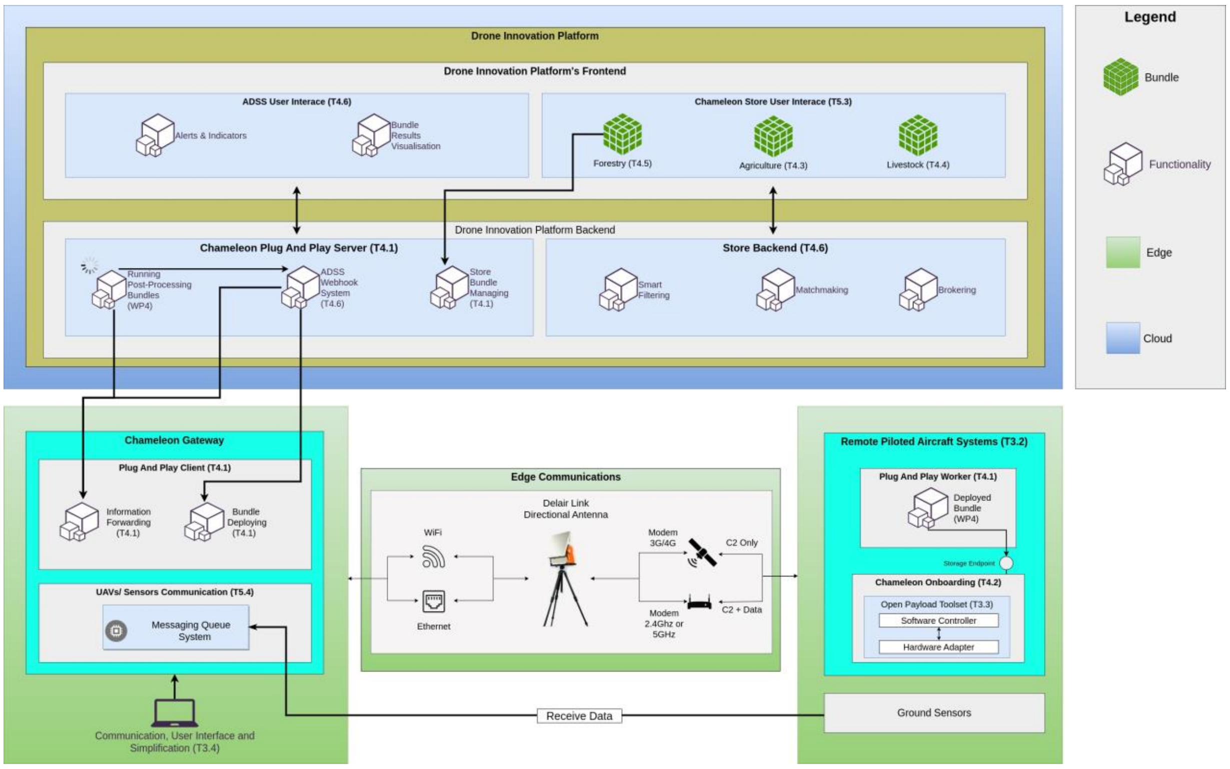
Drone Innovation Platform
Drone Innovation Platform's Frontend
ADSS User Interace (T4.6)
Alerts & Indicators
Bundle Results Visualisation
Chameleon Store User Interace (T5.3)
Forestry (T4.5)
Agriculture (T4.3)
Livestock (T4.4)
Drone Innovation Platform Backend
Chameleon Plug And Play Server (T4.1)
Running Post-Processing Bundles (WP4)
ADSS Webhook System (T4.6)
Store Bundle Managing (T4.1)
Store Backend (T4.6)
Smart Filtering
Matchmaking
Brokering
Legend
Bundle
Functionality
Edge
Cloud
Chameleon Gateway
Plug And Play Client (T4.1)
Information Forwarding (T4.1)
Bundle Deploying (T4.1)
UAVs/ Sensors Communication (T5.4)
Messaging Queue System
Communication, User Interface and Simplification (T3.4)
Edge Communications
Delair Link Directional Antenna
WiFi
Ethernet
Modem 3G/4G
C2 Only
Modem 2.4Ghz or 5GHz
C2 + Data
Receive Data
Remote Piloted Aircraft Systems (T3.2)
Plug And Play Worker (T4.1)
Deployed Bundle (WP4)
Storage Endpoint
Chameleon Onboarding (T4.2)
Open Payload Toolset (T3.3)
Software Controller
Hardware Adapter
Ground Sensors

Figure 1: Chameleon Architecture

- *Cloud Platform* serves as the central hub for the entire system. It is a server that handles the management and processing of service bundles selected by users. Equipped with technologies for data handling and webhook integration, the platform is responsible for the post-processing of bundles, ensuring they are tailored for specific tasks.

- The CHAMELEON *Gateway* handles the communication between the *Cloud platform* and the UAVs. It facilitates the deployment of processed service bundles to the UAVs, ensuring a reliable and efficient transmission of data.
- The CHAMELEON UAVs are essential operational components. They are designed to receive and execute containerized software from the Gateway, handling service bundles and data collection tasks. The integration of UAVs is marked by versatility, allowing a diverse range of models to be seamlessly incorporated into the system, ensuring adaptability and flexibility.

Together, these components form a cohesive architecture. *The Cloud Platform* with its processing capabilities, the *Gateway* with its role in data transmission, and the versatile UAVs all work in unison to create a streamlined and efficient system.

4.2 PLUG AND PLAY ARCHITECTURE

The CHAMELEON PnP Platform encapsulates 3 components: the Server, Client, and Worker, each playing a pivotal role in the system's architecture. The Server manages Store Bundles and facilitates real-time data interactions. The Client, enhanced by Kubernetes, orchestrates service delivery and data transmission. Finally, the Worker, primarily UAVs, executes the bundles and maintains communication within the Kubernetes environment.

4.2.1 SERVER

This section outlines the objectives and infrastructure of the *Server* component within the CHAMELEON project's PnP Platform. Integral to the Cloud Platform, the server handles tasks such as *Store Bundle Management*, *Webhook*, and *Post-Processing of Bundles*.

OBJECTIVES

The primary objectives of the *Server* within the CHAMELEON project's PnP Platform are to manage the store bundles and log if something unexpected happens while deploying them. This server is designed to handle tasks including *Store Bundle Management*, *Webhook*, and *Running Post-Processing of Bundles*. The webhook technology underscores the server's goal to facilitate real-time data exchange. Furthermore, the server's processing capabilities are central to its objective of delivering optimally refined bundles for UAV deployment.

INFRASTRUCTURE

The *Server* component of the CHAMELEON PnP Platform is designed with a triad of functionalities: *Store Bundle Management*, *Webhook*, and *Running Post-Processing of Bundles*. The integration of *Webhook* technology into the CHAMELEON *Plug n Play Server* is a decision that enhances the system's ability to communicate data in real-time. This technology establishes a reactive architecture, enabling the server to instantly respond to events or changes rather than relying on periodic polling. This immediacy in data communication maintains a dynamic and seamless flow of information across the entire architecture, ensuring that every component, from the server to the UAVs, operates on the latest data. Furthermore, the server's ability to manage bundles ensures that the bundle is deployed or, at the very least, logs any errors that occur in the PnP Platform.

4.2.2 CLIENT

The *Client* component is designed in order to address specific operational goals, notably enhanced by the utilization of Kubernetes. Serving dual roles in *Bundle Deployment* and *Information Forwarding*, particularly for UAVs, the *Gateway* is an element that streamlines operations and data transmission. This section will detail how Kubernetes, as the control plane within the *Gateway*, facilitates efficient orchestration of containerized applications, ensuring seamless service delivery and robust system performance.

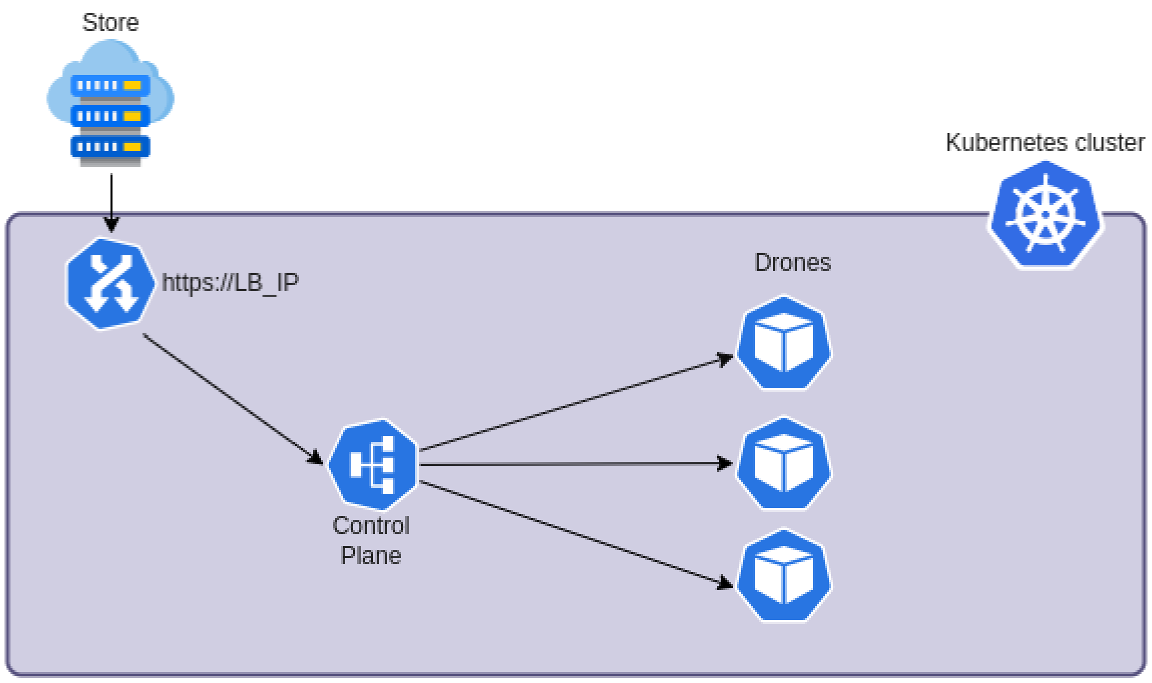

Figure 2: Internal Kubernetes Architecture

OBJECTIVES

The *Gateway* component of the CHAMELEON project, which includes the *Client, as shown in* Figure 1, is designed with specific objectives in mind, now orchestrated by Kubernetes, which was discussed[8]. The *Gateway* serves a dual purpose: efficient *Bundle Deployment* and *Information Forwarding*, as well as a message queue system to the UAV. A key objective is to assign the *Gateway* as the control plane of the Kubernetes cluster, ensuring optimal orchestration of containerized applications. This integration aims to streamline the deployment of service bundles to UAVs in order to have seamless data transmission and maintain system integrity through load balancing and automated scaling.

INFRASTRUCTURE

The infrastructure of the *Client* in the CHAMELEON project, is significantly enhanced through Kubernetes. This inclusion positions the *Gateway* as the Control Plane within the Kubernetes cluster, presented in Figure 2. The *Gateway,* in its capacity as the Control Plane, is in charge of scheduling and reacting to cluster events, such as modifying the pod's status to pending, successful, or failed. It manages the orchestration of containers, ensuring that the deployed services are running efficiently and reliably. Additionally, the infrastructure incorporates a message queue system. This system manages communication and data flow, particularly in high-load scenarios, ensuring that no data is lost during transmission. The use of Kubernetes enables automated scaling, self-healing capabilities, and load balancing, making the *Gateway's* management of containerized applications and services more streamlined and resilient. In conclusion, the Control Plane's role ensures that the system's resources are optimally utilized and that the services are available and responsive to user demands.

4.2.3 WORKER

In the CHAMELEON project, the *Worker* component, represented by the UAVs, is illustrated in *Figure 3*. Functioning as worker nodes within the Kubernetes cluster, these UAVs are designed to execute service bundles deployed by the *Gateway*. A key aspect of their functionality is the robust network bridge connection with the Gateway, vital for receiving instructions and relaying mission results in a Kubernetes-managed environment, as shown in Figure 2.

OBJECTIVES

The *Worker* component of the CHAMELEON project, integrated into the UAVs, is tasked with objectives fundamental to the system's operational success within a Kubernetes environment. These workers (UAVs) are designed to execute the service bundles deployed by the *Gateway*. The main objective of the *Worker node* is to ensure communication with the *Gateway*, via a network bridge. This connection allows UAVs to receive instructions and communicate back the results or any errors that may have occurred.

INFRASTRUCTURE

The infrastructure of the UAVs in the CHAMELEON project is engineered to support their role as worker nodes, inside the Kubernetes cluster, as shown in Figure 2. These UAVs possess the capability to execute assigned bundles and transmit operational data. Central to their infrastructure, which complements the Kubernetes architecture, is the network bridge, which forms a link with the Gateway. This bridge is needed for the UAVs in order to receive deployment instructions and communicate operational data back to the Gateway within the Kubernetes framework. The system's adaptability to various forms of network connectivity ensures that any connection (VPN, Wi-Fi, or Ethernet Bridge) is sufficient to establish a stable connection between the Kubernetes Control Plane and UAVs. This flexible networking approach is key to the UAVs' ability to function seamlessly as worker nodes, allowing them to execute tasks and maintain constant communication with the central system, regardless of their operating environment.

4.3 SEQUENCE OF ACTIONS

This chapter analyses the main operational process of the CHAMELEON PnP platform, through a sequence diagram, depicted in Figure 3. The description outlines how an end-user deploys a bundle and its interaction with the CHAMELEON PnP platform.

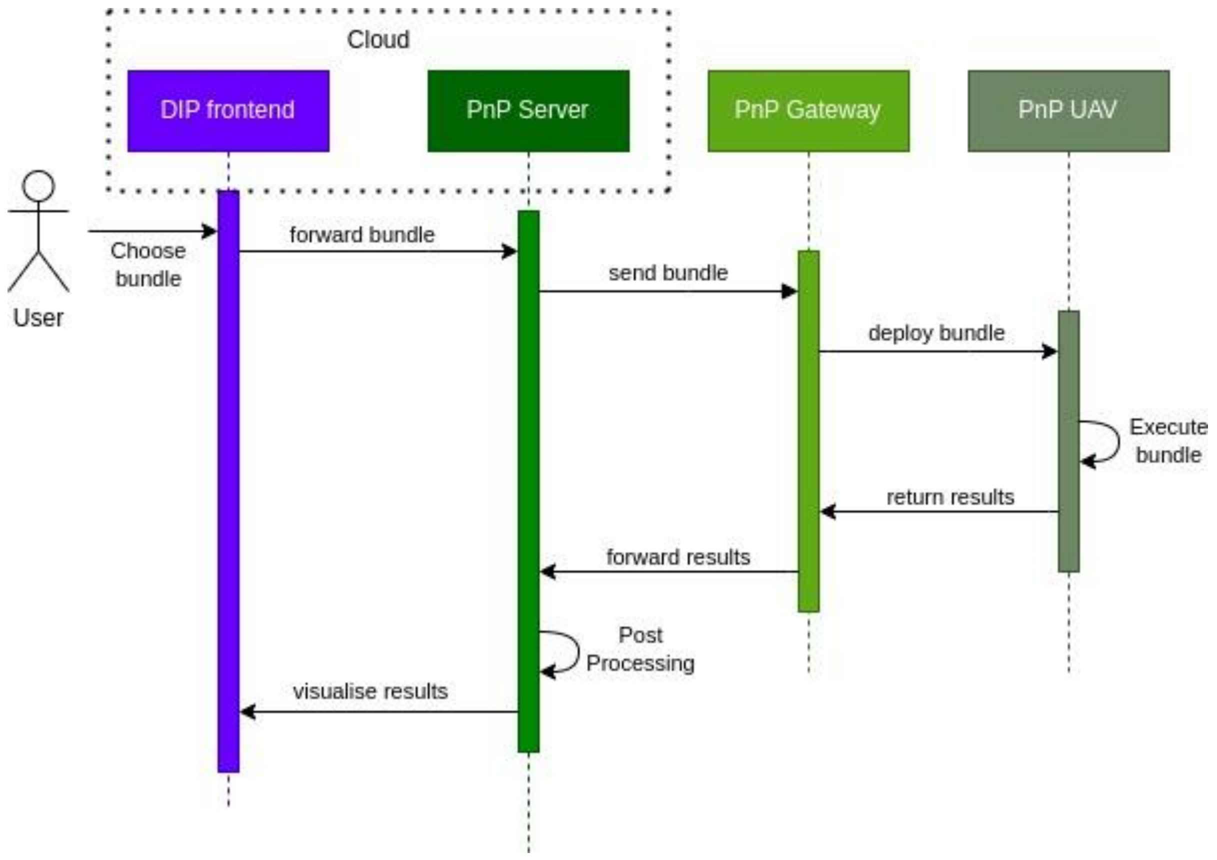

Figure 3: Plug-n-Play Processing Sequence Diagram

- *Initial User Interaction*: The sequence initiates with a user choosing a specific 'bundle' from a cloud-based 'Store'. This bundle represents a set of instructions, or a task designated for the UAVs to execute.
- *Task Forwarding*: Once the bundle is selected, it is forwarded to the CHAMELEON PnP *Server*, which acts as the control centre within the cloud infrastructure. The server is responsible for managing these bundles and directing them to the appropriate destinations.
- *Gateway Transition:* The *Server* then sends the bundle to the CHAMELEON PnP *Gateway*. The *Gateway* serves as the Control Plane of the PnP Platform, ensuring that the communication and task hand-off run smoothly.
- *UAV Deployment*: Upon receipt, the *Gateway* dispatches the bundle to the intended *Worker* (UAV). The UAV, equipped with the required capabilities, deploys the bundle by carrying out the instructions contained within it.
- *Execution and Feedback*: After the UAV has completed the task, it returns the results of the execution back to the *Gateway*. These results are then sent back to the CHAMELEON PnP *Server*.
- Post-Processing: Involves refining data results, ensuring that the information delivered to the front end is accurate. This stage may include filtering no needed data and formatting the results.

- *Completion*: Finally, the *Server* returns the results of the task execution back to the user. This step closes the loop, providing the user with the outcome of the task and allowing for further actions based on the results received.

5 CONCLUSIONS AND NEXT STEPS

This document presented the architecture and the initial development steps of the Chameleon Plug-n-Play Platform. The choices for the design of the platform are elaborated. The relationship with other CHAMELEON components is also presented. Finally, the main operational actions are described. The development process, along with the overall architecture, may be subject to change as the CHAMELEON project and related components continue their development.

The next steps in the CHAMELEON Project involve integrating the platform with other CHAMELEON components, if necessary, and testing and implementing the first on-premises integration alongside the UAVs.

6 REFERENCES

[1] D. S. Developers, "Docker Swarm," Docker, [Online]. Available: https://docs.docker.com/engine/swarm/. [Accessed 05 12 2023].

[2] A. M. Developers, "Apache Mesos," Apache, 24 11 2020. [Online]. Available: https://mesos.apache.org/. [Accessed 05 12 2023].

[3] K. Developers, "Kubernetes," The Linux Foundation, 09 09 2014. [Online]. Available: https://kubernetes.io/. [Accessed 05 12 2023].

A Holistic Approach to Sustainable, Digital EU Agriculture, Forestry, Livestock and Rural Development based on Reconfigurable Aerial Enablers and Edge Artificial Intelligence-on-Demand Systems

The Members of the CHAMELEON Consortium:

Contact:

Project Coordinator: **Pantelis Velanas** Acceligence Ltd.	pvelanas@accelligence.tech

Disclaimer