Document title: Arrowhead fPVN Deliverable D2.1

Version	Status	Date
1.0	final	2023-12-21

Author	Contact
Pal Varga	pvarga@tmit.bme.hu

Deliverable D2.1
The first version of fPVN microservice architecture – Arrowhead v5.0 GSoSD

Work package leader: Pal Varga
pvarga@tmit.bme.hu

Karl-Johan Gramner
karl-johan.gramner@bnearit.se

Editor: Pal Varga
pvarga@tmit.bme.hu

Abstract

This document constitutes deliverable D2.1 of the Arrowhead fPVN project.

WP2 will develop the microservice paradigm-based platform for flexible production value networks. WP2 addresses the maturing of released Eclipse Arrowhead core services, provisioning of communication interoperability, autonomic identification of data-model non-interoperability properties, dynamic instantiation of data-model translation services and improvement of micro-service management based on policy and rule-based management.

Table of contents

1. Introduction

1.1 The elements of Arrowhead fPVN deliverable D2.1

The deliverable D2.1 of Arrowhead fPVN consists of the following documents.

1) Deliverable D2.1 Main document (Arrowhead_fPVN_D2.1_v0.5)
2) Generic System-of-Systems Description (GSoSD) – The Arrowhead Core Systems – v5.0 (GSoSD Arrowhead Core Systems 5.0_v08)
3) Service Registry System – System Description (serviceregistry_sysd_v50_r1)
4) Service Orchestration System – System Description
    i. (serviceorchestration_sysd_v50_r1)
5) Authorization Core System – System Description (authorization_sysd_v50_r1)

Furthermore, the Roadmap discussions and results can be found on github:

https://github.com/eclipse-arrowhead/roadmap/tree/main/5.0%20Draft

1.2 Overview of WP2 aims and tasks

WP2 aims to build, mature and extend a microserves/SOA-based platform to support flexible production value networks in heterogeneous and complex settings. It consists of four tasks, namely:

- Task 2.1 Maturing of the Arrowhead Framework
- Task 2.2 Autonomous handling of data access and non-interoperable properties
- Task 2.3 Microservice design methodologies for Industry5.0
- Task 2.4 Governance of microservice-based architectures

Their main goals are the following.

1.3 The main purpose of Task2.1

For Task 2.1, the main purpose is to extend and mature the Eclipse Arrowhead framework. The task addresses robustness and stability of the Eclipse Arrowhead core system identified as fundamental to enable the objectives of Arrowhead fPVN, e.g., ServiceRegistry, Orchestration, Authorisation, Translation interoperability adaptors. In order to reach this, the partners will work on various coordinated subtasks to reach the following targets:

- Bringing Mandatory and supporting core systems and services to TRL7-8
- Providing seamless interconnections with established IoT platforms
- Providing Microservices in Support of Interoperability
- Conceptualizing and implementing Autonomous Contracting & Invoicing for information sharing and access in fPVN.

- Data access management and data access control for third parties: authentication, authorization and object level authorization.

1.4 The main purpose of Task2.2

Regarding T2.2, the main purpose is to provide concepts and solutions on data handling-related issues in the non-interoperable SOA environment and provide means for handling the arising security and privacy related issues of microservice-based architectures. The task will investigate updates to the Eclipse Arrowhead architecture and relevant core systems to enable dynamic instantiation of translation services or interoperability adaptors. The task will provide a non-interoperability service, providing information about which communication, security, and data model where a non-interoperability has identified. The main target areas to focus on here are:

- Autonomous detection of non-interoperable properties
- Autonomous instantiation of necessary translation
- Embedding Intrinsic Security, Privacy, and Data Integrity in local and global fPVN clouds

1.5 The main purpose of Task2.3

Task T2.3 focuses on exploring cyber-physical systems-of-systems (CPSoS) and industry-led design patterns in microservice-based architectures. Its goal is to develop conceptual tools and design strategies to integrate these components and methods in specific use-cases. Additionally, the task considers the impact of human interaction on microservices/SOA-based System-of-System architectures. The objective is to promote the concept that microservices can be essential components in the data exchange processes of Industry 5.0. This task includes creating microservices for constructing Industry 5.0 CPSoS architectures, compiling a design methodology guide for microservices in industrial contexts, and providing technological support and background for industrial pilot projects.

1.6 The main purpose of Task2.4

Task 2.4 aims address management and governance aspects of microservice architectures within industrial environments. Its scope encompasses developing solutions in several key areas related to governance of such architectures throughout their lifecycle. Thie includes the following:

- Managing changes and versions in microservice-based architectures
- Deployment strategies and autonomous scaling methods for microservice-based Systems of Systems (SoS)
- Governance methods during run-time, design-time, and deployment-time

Naturally, Task 2.4 is committed to researching and offering comprehensive updates and extensions in these areas.

2. WP2 Objectives

WP2 aims to fulfil the General Objective #2 of the Arrowhead fPVN project, namely creating and maturing a microservices/SOA platform that enables of dynamic deployment and autonomous utilization of information translation in PVNs.

The main objectives of WP2 are:

- Maturing of released Eclipse Arrowhead core services to TRL7-8.
- Defining an innovative microservices based architecture, meeting a comprehensive assessment of performance KPIs.
- Defining a set of solutions for middleware virtualization
- Provisioning of communication interoperability through new, updated and extended protocol translators and legacy technology adaptors targeting TRL6-7
- Provisioning of autonomic identification of data model non-interoperability properties
- Provisioning of dynamic instantiation of data model translation services (from WP4)
- Provisioning of improved microservices management based on policy and rule-based approach.

Objectives	**Work towards achieving objectives**
Microservices/SOA enabling of dynamic deployment and autonomous utilization of information translation in PVNs	The definition of the roadmap for Eclipse Arrowhead v5.0 provided by WP2 serves as a basis for core system refinements, microservice creation and refinements, and CPSoS creations. WP2 also worked together with the technical work packages WP3 and WP4 in order to create a proper interoperability platform, allowing translations and adaptions for various major digital data models and languages.

3. Recent results on maturing the Eclipse Arrowhead framework

3.1 The main issues of GSoSD v5.0

The Eclipse Arrowhead v5.0 main concepts have been discussed by the Arrowhead Roadmap definition group.

The latest approved version of the Generic System of Systems Description (GSoSD) is v0.8 at the moment, and it is part of this deliverable D2.1.

GSoSD review issues and questions are tracked in the Eclipse Arrowhead roadmap github: https://github.com/eclipse-arrowhead/roadmap/issues

There were 10 outstanding issues (numbered as #1 .. #10) in relation to the GSoSD of Eclipse Arrowjead v5.0, for which the resolution discussion and concept elaboration went on through months of the regular Roadmap meetings. These are the following:

#1: Should Arrowhead systems be "lazy" (passive) or "active" (proactive)?

> **#1 resolution:** both are feasible and required modes of operation, hence both are introduced in the GSoSD document

#2: What "specification documents" should exist apart from the GSoSD, the Concepts Reference and the Foundational Principles documents? What is the scope of each document?

> **#2 resolution:** introduced in the GSoSD document

#3: How do we name clouds, systems, services, operations, devices and other components? Should there be both human-readable names and machine-readable names, or can one type of name be used for both?

> **#3 resolution:** the issue is moved to v5.1 roadmap

#4: How do we name the Orchestration System (Core) and the Choreography System (Support) such that we avoid unnecessary confusion from the Microservices communities?

> **#4 resolution:** introduced in GSoSD document

v4.x (Current) name	v5.x (Endorsed) name
Orchestration System	Service Orchestration System
Choreography System	Workflow Orchestration System

#5: Should all current kinds of orchestration be supported by the same one orchestration system?

> **#5 resolution:** there are valid use-cases for various kinds of service orchestrations – their ways of operation are described in the GSoSD document.

#6: The Role of an Authentication System?

> **#6 resolution:** complete description is introduced in the GSoSD document.

#7: What authorization mechanisms should be supported by the Authorization System? OAuth 2.0? OpenID Connect (together with the "Authentication System")?

> **#7 resolution:** the mechanisms are not defined as technology dependent in the concept of Eclipse Arrowhead v5.0. The main cornerstones are described in the GSoSD document.

#8: What terms to use for Services and Systems? Microservices and microsystems?

> **#8 resolution:** The naming convention is described in the GSoSD document. Arrowhead uses microsystems and microservices, although The terms microsystem and microservice will be used interchangeable with system and service to simplify writing.

#9: Should Core systems be useful on their own, or can they depend on each other directly?

> **#9 resolution:** Systems must be planned and introduced as independent, and their independence should be kept as much as possible, even if the complexity of the core systems and the local cloud grows. Details are introduced in the GSoSD document.

#10: What kinds of backwards compatibility should Arrowhead version 5 offer with regards to version 4?

> **#10 resolution:** introduced in GSoSD document

3.2 Summary of major conceptual changes for Eclipse Arrowhead v5.x

3.2.1 Working Model for Components

The philosophy of “Arrowhead thinking” should encourage different implementations - in addition to the reference implementation - that are more suited for more specific use cases. Moreover, the arrowhead concept should allow the users to select only those components that could be useful for a particular use case without requiring other dependent components. Thus, every arrowhead component (core and support system) are required to have an individual way of working strategy when no other arrowhead component is necessary to perform its tasks. Beside to this, arrowhead components could continue to support more complex way of working strategies where dependencies between components may be required. In this spirit the next generation of arrowhead **will not contain mandatory components** (only “recommended”) and **the components will have independent working modes as well**. These changes will allow both: establishing local clouds where the application systems are more autonomous and have more responsibilities, but also local clouds where the application systems could focus only to performing its micro services and let arrowhead framework manage all the other necessary tasks and responsibilities as much as possible.

3.2.2 Security-related solutions

Encouraging secured local cloud deployments is still a key target of the arrowhead community, therefore the arrowhead concept **will allow to apply a wider range of security related solutions** and let the framework implementations to decide what exact solution(s) to support until the following major requirements are respected:

Application systems must have an identity in the local cloud.

Application systems' identity must be authenticated when joining to the local cloud.

Service consumption requests must be authorized during the orchestration process (when orchestration is required).

3.2.3 Authentication

In order to provide the implementation possibility of a chosen solution, **a new recommended component, the Authentication Core System will be introduced**, which has the role of creating, verifying and withdrawing application system identities.

3.2.4 Authorization

Authorized service consumption is still a key feature of the arrowhead concept, however the use case scenarios and the user feedbacks have proven that the peer to peer authorization rules are quite cumbersome, therefore a more flexible and general solution is required. In order to meet with these requirements, the arrowhead concept **will allow to define authorization rules in a wider range as well** and also **will allow the provider systems to define their own rules**.

3.2.5 Operations

In order to ensure a proper granularity level of the services and also different permission levels to a service the **arrowhead concept will no longer allow the service design without operations**. Every framework implementation must represent the services according to the service-operation design pattern, where a service is a set of operations related to the service domain and the operations are exposed via one or more interface. Providing a service means that each and every operation of it must be implemented by the provider system. No partial implementation is allowed.

3.2.6 Device related data in Service Registry

Even though there is no fundamental need for having device information stored for a service, it could be required in some of the use cases, therefore the **arrowhead concept will allow the existence of device related data in Service Registry Core System, but only with optional manner**.

3.2.7 Orchestration of Services and Workflows

In order to ensure that arrowhead users from various domains adopt the role and terminology of the core systems more easier, some slight renaming will be beneficial. Bearing this in mind **the current Orchestration Core System will be renamed to Service Orchestration Core System and the Choreographer Core System will be renamed to Workflow Orchestration System**.

3.3 Major changes in reference implementation for Eclipse Arrowhead v5.0

In order to realize the conceptual changes, the current reference implementation of arrowhead will apply the following changes.

3.3.1 General

The ‘core-java-spring’ reference implementation is relying on an interconnected data storage between the core and supporting systems. This particularity must be changed due to the requirement of “independent working modes”, therefore each and every core and support system will have its own database. This fundamental change in the implementation also means that database record identifiers are no more possible to being used for identifying the entities (system instances, services definitions, etc…), so entities must have unique identifiers (names) at cloud level.

3.3.2 Authentication

The current authentication mechanism is built on X.509 certificates. While it is working well and reliably it requires a bit more experience and preliminary work to utilize them in a secured cloud deployment. Since the conceptual requirement is “applying a wider range of security related solutions” and also the arrowhead community is targeting both, keep forcing secured clouds, but also make the usage easier and more flexible, an additional simple-token based security level will be introduced. Also, as stated in the conceptual changes a new Authentication Core System will be developed and all these authentication related tasks will be outsourced to this new core system. With these changes additional authentication solutions will be possible to integrate at later stages.

3.3.3 Service Registry Core System

In the new version of Service Registry Core System the services will be represented according to the service-operation design pattern.

In addition to service and system data, device data registration will also be possible by the application systems.

The interface representation of a service operation will be much flexible and more expressive to ensure that translation support services could rely on the interface data stored by Service Registry Core System.

3.3.4 Authorization Core System

The current peer to peer authorization rules are proven to be inadequate, therefore a policy based solution will be implemented and the followings will be expressible:

- this service/operation is accessible anyone within the local cloud
- this service/operation is accessible anyone within the local cloud, except these consumers (blacklist)
- this service/operation is accessible anyone from the given list (whitelist)
- this service/operation is accessible anyone within the local cloud having the specified meta data
- this service/operation is accessible anyone from the given neighbor cloud list (intercloud whitelist)

Also, provider system will be allowed to define their own service authorization policies with the stipulation that rules made by cloud operators always have priority.

Authorization rules are currently not covering the events handled by the Event Handler Support System, therefore the new policy based authorization will be extended to event types as well.

The current token based authorization is using Json Web Tokens (JWT), which require the providers to being able to decode the token by themselves and read its content. In addition to JWT a simpler token based authorization will be introduced, where the providers will have the possibility to "ask" the Authorization Core System to validate the token received from a consumer system.

3.3.5 **Service Orchestration Core System**

In the current reference implementation the push orchestration is not available, which will change in the next major version and consumer systems will have the possibility receive push orchestration results from this core system.

In order to give room for the implementation of several orchestration strategies the consumer systems will have the possibility to define the required orchestration strategy in the orchestration form. The next version of Service Orchestration Core System will support the following strategies:

- static-store (default)

  Peer to peer orchestration rules made by a higher entity. This is the default, when strategy is not defined.

- flexible-store

  System attribution based orchestrion rules made by a higher entity.

- dynamic

  Unlike the rule based strategies, this way of orchestration is looking for the perfect match(es) on the fly and based on the actually registered service providers.

3.3.6 **Authentication Core System**

In order to enable an Authorisation Service to fulfil its purpose in a secure manner, the systems that consumes authorization must be properly identified. The Authentication System should be responsible for carrying out this task. This system should issue identifiers that can be distributed and validated in a secure manner. This identifiers can be set at deployment time, for example as a certificate, or issued dynamically, for example in the form of a token. The Authentication System should be able to provide validation of the identifiers, to make sure that systems are the one they claim to be.

A limitation in the scope of Arrowhead Core Framework v5.0 is that any system Authentication is performed while User authentication is still under investigation.

Authentication methods are further described in the GSoSD.

4. WP2 technical report M1-M6

In this first part of WP2 activities, the partners focused on setting the requirements and roadmap for the Eclipse Arrowhead v5.0 (maturing: T2.1), and to develop initial plans for the other three tasks (T2.2-2.4). The partners refined objectives, and established success criteria for each task.

As Task 2.1 aims to mature the Eclipse Arrowhead framework, we refined the roadmap for enhancing core systems, facilitating seamless interconnections with IoT platforms, providing interoperability microservices, and implementing autonomous mechanisms. Task 2.2 focuses on handling non-interoperable properties, enabling autonomous translation, and embedding intrinsic security, privacy, and data integrity measures – for which we started the initial planning. In Task 2.3 we investigated microservice design methodologies for industrial CPSoS, and started the initial planning of developing microservices for CPSoS architectures – especially within Arrowhead. This later leads to the creation of a design methodology cookbook, through which WP2 can provide technology support for industrial pilots. Task 2.4 focuses on governance aspects; where the initial work has also started. In this task, the stakeholders are addressing change and version management, deploying and scaling microservice-based SoS, implementing run-time, design-time, and deployment-time governance methods, and enhancing microservice management capabilities within the Arrowhead framework.

WP2 actively participated in the joint KDT project Deep Tech workshop – involving Arrowhead fPVN and the AIMS5.0 project –, in St. Pölten 2023 September.

The first six months of WP2 were marked by successful requirement setting, initial planning of the roadmap, and defining the main engineering approaches. The partners are now proceeding to complete detailed plans and commence task implementation.

4.1 Task 2.1 Maturing of the Arrowhead Framework

The main activities are summarized in the following points.

- **Bugfixes and vulnerability fixes of Eclipse Arrowhead v4.6.x**
  Since the release of version 4.6.0 several used libraries have offered new updates fixing some security vulnerabilities. In Eclipse Arrowhead v4.6.1 we utilize the most recent versions of these libraries. Also, Docker support is reworked in order to offer a more flexible experience. The documentation is transferred into a more useful Github wiki format. Various bugfixes are also included. Eclipse Arrowhead 4.6.2 is going to released in the recent future with more updates and bugfixes.

- **Setting the Roadmap for v5.0**
  There are monthly Roadmap meetings about the Arrowhead Framework's future improvements. On these meeting we identified several discussion points that need to address in Eclipse Arrowhead v5.0

(https://github.com/eclipse-arrowhead/roadmap/issues/63).
The current results of these meetings are various documents including a General System of Systems Description (GSoSD) and the System Description (SysD) documents of some recommended core systems.

- **SysD v5.0 initial version for Service Registry, Authorization and Service Orchestration Systems**
  A System Description (SysD) describes the behavior of a system without any implementation details. These documents also contain the conceptual changes from the previous version. The initial version of SysDs for Service Registry, Authorization and Service Orchestration systems have been written.
  The Service Registry core system enables service discovery within an Eclipse Arrowhead Local Cloud.
  The Authorization core system manages and authorizes connections between various systems using authorization rules within a Local Cloud.
  The Service Orchestration core system finds matching providers for the consumer's specification within a Local Cloud and optionally, in other Arrowhead clouds by collaborating with other core/support systems.

As a sub activity of Task 2.1, a meeting was held in Vienna in September where Sinetiq, BME and AITIA participated. The purpose of the meeting was to set common ground for a commercial version of the Arrowhead Framework Components. The agenda encapsulated a short presentation of the companies and the current status of the Arrowhead engagement, a discussion of principles when addressing the Arrowhead Core System challenges and how to approach the Arrowhead fPVN community in a joint collaboration.

4.2 **Task 2.2** Autonomous handling of data access and non-interoperable properties

In Eclipse Arrowhead v4.6 there is no implemented support of non-interoperability handling from the core systems. This currently means that in case the consumer receives an unsuccessful Orchestration result, it may ask the Translator about translation options (data formats and/or protocols) – then, equipped with this information, the Consumer may request another Orchestration.

This is going to evolve significantly for v5.0; there has been several options started to get conceptualized. The following options are under consideration.

- **Plan option a) for v5.0: Translation chaining**
  The partners have designed a new translation concept proposal which utilizing the micro service architecture of the Arrowhead Framework in a way where the communication protocol and data model translation tasks are outsourced from a central core system to multiple provider systems.
  If translation is required, the new Translation support system could establish a translation bridge which combines an appropriate protocol translator provider with multiple data model providers for transforming input and output data. After the bridge is built the consumer can connect a generated access point of the bridge which made the

necessary conversions before using the original provider service. The provider's response is transformed as well by the bridge before the consumer receives it.

- **Plan option b) for v5.0: Protocol and data format identification**
  As data model-related tasks are ongoing in WP3, and translation-related tasks are also ongoing in WP4, the domain knowledge for data model and protocol identification is available in the technical WPs. The identification of data models and protocol versions from live messages or transactions must be possible through rule-based or machine learning-based methods. This should also be a step 0 for certain translators: distinguishing the data format and protocol version of the messages sent to them. The principles of such an identifier module, and the way it receives input and generates output are to be defined in the next project phase.

Furthermore, the conceptualization of handling arising security and privacy related issues are started in Task 2.3.

4.3 Task 2.3 Microservice design methodologies for Industry5.0

Requirements are gathered from WP3, and collaborations with the use cases together with WP3 and WP4 have started.

The primary goal of Task 2.3 is to design and develop design patterns for microservice-based architectures for CPSoS in Industry5.0. To achieve this, we started the initial work on the following key areas:

- **Microservices for building Industry5.0 CPSoS architectures**
  As part of the Eclipse Arrowhead v5.0 roadmap, the partners started to design microservices fitting the requirements of CPSoS architectures. These microservices will serve as essential building blocks for enabling seamless integration and data exchange within complex industrial systems.

- **Design methodology cookbook of microservices for fPVN use-cases**
  Task2.3 has started the discussions on the Arrowhead fPVN cookbook that is going to outline design methodologies for microservices-based CPSoS. This cookbook will provide practical guidance to developers and architects who are building microservice-based solutions for industrial applications. As part of these activities, the GSoSD has been born.

- **Creating the reference GSoSD definition for v5.0 of the recommended core systems**
  This GSoSD outlines the 5th generation of the Arrowhead Core systems, which offer functionality expected to be needed for the majority of use cases where Arrowhead is applied. It contains a high-level architectural description of what problems the Core systems solve and how they interact.

The primary purpose of the document is to present the functionality offered by the Arrowhead Core systems. Its architectural aspects are defined in the abstract, by which we mean that no specific implementations or technologies endorsed by it.

- **Technology background and support for the industrial pilots**
  Task2.3 will provide in-depth technology support to the industrial use-cases who are utilizing microservices. This includes technical assistance, training, and ongoing consultation to ensure they achieve the expected results. The communication channels are set up for this support activity.

4.4 Task 2.4 Governance of microservice-based architectures

During the first 6 months the task 2.4 has focused mainly on the design- and run-time governance methods. While still on a conceptual level some progress is being made towards having a shared registry for service definitions in the Arrowhead community. This will enable an easy handover between use cases and technology providers, interoperability since interfaces are made public and later used to verify the translator's interoperability. Progress is being made towards having a first demo in the coming months. Providing verifiable identifies for service interfaces, deployed systems and vendors is an enabler for other governance methods, policy rules and other management capabilities.

To mature the Arrowhead Framework from a commercial perspective, the community has created the "Market Oriented Action Group", MOAG. During the Arrowhead fPVN project the role the MOAG is to spread the word about Arrowhead in the commercial ecosystem, share best practices in commercializing Arrowhead-based solutions, and, catalyze dissemination of new results. This activity also results in bringing mature Arrowhead-based solutions from TRL8 to 9 and 10.

As part of the governance activities, a meeting was organized between commercial core contributors in Vienna, in September. The main subject of the meeting was sharing experiences selling the Arrowhead concept in the industry and what needs to change to make it more feasible to use. The outcomes were feedbacked to task 2.1 as outlined previously. Going forward in the Arrowhead fPVN project, further similar meetings are planned.

During the early phases of the fPVN project, the Eclipse Arrowhead Framework strengthened its position the Eclipse IoT Working Group (https://iot.eclipse.org/), an organization or organizations collaborating on new technology development in the IoT space. The Arrowhead Working group within Eclipse – manifesting in Task2.4 actvities currently – plans to use intermediate supporting tools and activities from the Eclipse foundation. As the Eclipse Arrowhead aims to be an active member in the IoT working group, the community is planning to co-locate one Arrowhead workshop during EclipseCon in the autumn of 2024.

5. Collaboration with other technical WPs

WP2 had regular collaboration meetings with WP3 and WP4 to define the requirements about the integration of the T3.1 – T3.3, and of the T4.1 – T4.3 results into the microservice-based AH Eclipse architecture. These requirements cover the following topics:

- Ensure Syntactic, Semantic and Pragmatic Interoperability between systems.
- Ensure interoperability between major standards.
- Ensure the integration possibilities of AI-based Translator modules.
- Ensure the integration possibilities of Model-based Translator modules.

Based on the T3.1 survey results, the participants reported more than 27 major industrial data models, and more than 10 data representation formats. The reported formats could be separated into three groups:

1. The major of the representation formats are XML and JSON, including predefined or standardized schemas.
2. EXPRESS language-based models.
3. Model-based representation (e.g., UML).

As a first interim result of the collaboration between Work Packages,, the requirement of WP3 and WP4 towards WP2 is that the Arrowhead Eclipse core system should provide the following features:

- Translation subsystems based on predefined schemas. The schemas should be uploaded by a provider or consumer, and should be handled as metainformation by the AH system components.
- Schema-related orchestration.
- Automatic recognition of the interoperability problems, and possible solutions with translator instantiation.
- Extending the orchestration parameters with context-specific information to support the Pragmatic Interoperability aspect.

6. Conclusion

This is the first deliverable – D2.1 – of the WP2 "Microservices" work package of the Arrowhead fPVN project.

One of the major results of the first six months of WP2 is that the partners defined the first version of the microservice architecture for Arrowhead fPVN, in the document GSoSD v5.0. This work is the result of the roadmap discussion elaboration and definition for Eclipse Arrowhead v5.0, including enhancing the core systems, defining microservice development, and CPSoS creation. Additionally, WP2 collaborated with WP3 and WP4 to establish a robust interoperability platform, enabling translations and adaptations for various digital data models and languages.

Specifically, the stakeholders addressed bug fixes, vulnerability fixes, defined the Eclipse Arrowhead v5.0 roadmap, created reference GSoSD definitions for v5.0 of recommended core systems, and developed an initial version of SysD v5.0 for Service Registry, Authorization, and Service Orchestration Systems – taking into account microservice-related requirements for industry5.0 needs. By working together with WP3 and WP4, the WP2 partners established the basic concepts for the identification of non-interoperable properties. Additionally, regarding design- and run-time governance methods, progress was made towards establishing a shared registry for service definitions within the Arrowhead community.

7. Appendixes

Besides this “Deliverable D2.1 Main document (Arrowhead_fPVN_D2.1_v0.5)” the other document of this deliverable are the following:

1) Generic System-of-Systems Description (GSoSD) – The Arrowhead Core Systems – v5.0 (GSoSD Arrowhead Core Systems 5.0_v08)
2) Service Registry System – System Description (serviceregistry_sysd_v50_r1)
3) Service Orchestration System – System Description
    i. (serviceorchestration_sysd_v50_r1)
4) Authorization Core System – System Description (authorization_sysd_v50_r1)

Furthermore, the Roadmap discussions and results can be found on github:

https://github.com/eclipse-arrowhead/roadmap/tree/main/5.0%20Draft

8. References

-

9. Revision history

9.1 Contributing and reviewing partners

Contributions	Reviews	Participants	Representing partner
Architecture Definition			AITIA
Architecture Definition			BME
Architecture Definition			SINETIQ
Architecture Definition			LTU
Architecture discussion and review			EUROTECH
Architecture discussion and review			BEIA
Architecture discussion and review			LEONARDO

9.2 Amendments

No.	Date	Version	Subject of Amendments	Author
1	20/11/2023	0.1	Initial version	Pal Varga
2	25/11/2023	0.2	Raw version of the major v5.0 descriptions	Tamas Bordi
3	30/11/2023	0.3	Task contributions	Per Olofsson, Pal Varga
4	12/12/2023	0.4	WP3-4 collaborations	Tamas Tothfalusi
5	15/12/2023	0.5	Finalization for review	Pal Varga
6	21/12/2023	1.0	Review suggestions implemented	David Rutqvist, Pal Varga

9.3 Quality assurance

No	Date	Version	Approved by
1	15/12/2023	0.5	Jerker Delsing
2	21/12/2023	1.0	Jerker Delsing