D3.4: COROB platform release

Revision: v1.0

Work package	WP3
Task	T3.3
Due date	30/09/2025
Submission date	06/10/2025
Deliverable lead	INTRA
Version	1.0
Authors	INTRA, TU/e
Reviewers	Iñaki Sainz (LOR)
Abstract	Digital platform with interoperability capabilities in cognitive HMI, data management.
Keywords	Interoperability, Asset Administration Shell

CHANGE CONTROL

Date	Version	Editor	Changes to document
2025/09/01	v0.1	INTRA	Outline
2025/09/29	v1.0	INTRA	Full Draft

Important remarks:

- The contributors listed in this table and on the front page are the report's primary editing authors. It is important to note that all COROB partners are contributing critical technical contributions to this ongoing work.

www.corob-project.eu

Grant Agreement No.: **101120640**
Call: **HORIZON-CL4-2022-DIGITAL-EMERGING-02**

Topic: **HORIZON-CL4-2022-DIGITAL-EMERGING-02-05**
Type of action: **HORIZON-IA HORIZON Innovation Actions**

DISCLAIMER

Co-funded by the European Union

Project funded by

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Federal Department of Economic Affairs, Education and Research EAER
State Secretariat for Education, Research and Innovation SERI

COROB (*Cooperative robotics powered by AI and data for flexible production cells*) project is co-funded by the EU's Horizon Europe programme under Grant Agreement number 101120640. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them. This work has received funding from the Swiss State Secretariat for Education, Research, and Innovation (SERI).

Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

COPYRIGHT NOTICE

Project co-funded by the European Commission in the Horizon Europe Programme		
Nature of the deliverable:	**DEM**	
Dissemination Level		
PU	*Public, fully open, e.g. web (Deliverables flagged as public will be automatically published in CORDIS project's page)*	✓
SEN	*Sensitive, limited under the conditions of the Grant Agreement*	
Classified R-UE/ EU-R	*EU RESTRICTED under the Commission Decision No2015/ 444*	
Classified C-UE/ EU-C	*EU CONFIDENTIAL under the Commission Decision No2015/ 444*	
Classified S-UE/ EU-S	*EU SECRET under the Commission Decision No2015/ 444*	

* *R: Document, report (excluding the periodic and final reports)*

DEM: Demonstrator, pilot, prototype, plan designs

DEC: Websites, patents filing, press & media actions, videos, etc.

DATA: Data sets, microdata, etc.

DMP: Data management plan

ETHICS: Deliverables related to ethics issues.

SECURITY: Deliverables related to security issues

OTHER: Software, technical diagram, algorithms, models, etc.

EXECUTIVE SUMMARY

This document complements the work performed under "Task 3.3 User Experience design & Platform Deployment" with focus on the data management. It documents the overall platform and its configurability. In section 1 an introduction and a business perspective is given followed by the solution architecture in section 2. Sections 3 and 4 addresses the platform's installation, configuration, and usage. Section 5 presents assessment outcomes (assessments are presented in section 8 Annexes) in terms of comparison with similar open-source initiatives and finally Section 6 concludes this document highlighting the advantages of the COROB Platform.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS & GLOSSARY

AAS	Asset Administration Shell
API	Application Programming Interface
COROB	Cooperative robotics powered by AI and Data for flexible production cells
I/O	Input/Output
MQTT	Message Queuing Telemetry Transport
OPCUA	Open Platform Communications Unified Architecture
REST	REpresentational State Transfer
JSON	JavaScript Object Notation
corobHost	The machine name that host the COROB platform.

1 INTRODUCTION

The COROB platform is a digital solution that bridges the gap between shopfloor and data-driven services. It enables the digitization of companies enabling them to manage data in an interoperable, scalable, and vendor-independent way.

The platform connects devices at the edge by supporting standard industrial protocols and integrates them into a semantic layer using the Asset Administration Shell (AAS). This creates a digital representation (digital twin) of each asset ensuring that information is accessible in an interoperable way, independent of the underlying technology.

For companies, this means:

- **Digitalization**: shopfloor data and systems can be connected seamlessly without costly custom integration.
- **Interoperability**: assets from different vendors communicate through a common semantic layer.
- **Data sovereignty and security**: companies maintain control of their data, enabling secure sharing.
- **AI and analytics**: Shopfloor data are maintained structured that can be used directly for dashboarding, analytics, or AI based applications.

2 ARCHITECTURE

The architecture of the platform is composed of three main layers:

1. **An IoT Edge Layer** – connects to shopfloor assets and collects data through the industrial communication protocols also handling the historical data storage.

2. **Semantic Layer (AAS Repository)** – provides digital twins of shopfloor assets and ensures interoperability through standardized data models.

3. **Security Layer** – Secure access control, and interaction with external applications both on historical but also on contextual information.

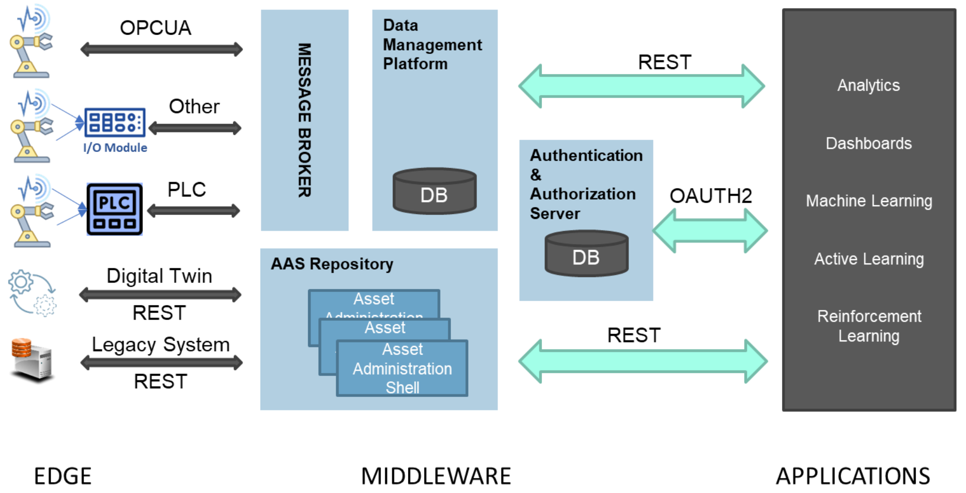

Figure 1 COROB Platform

By combining these layers, the COROB platform offers a **turnkey solution** for companies to experiment and adopt Industry 4.0 principles, becoming flexible, efficient, and intelligent production systems.

3 DEPLOYMENT

The COROB platform has been designed for flexibility and scalability. It can be deployed on a variety of environments ranging from on-premise industrial sites to cloud infrastructures. The solution is provided as containerized services, making it easy to install and scale depending on the size and requirements each case.

The deployment approach allows companies to start small—connecting a few devices and assets—and then expand to larger and more complex environments without re-engineering the system. The only prerequisite is the availability of a container runtime (e.g., Docker).

4 CONFIGURATION & USAGE

Using dedicated graphical tools and APIs, platform actors (users, applications) can:

- Define the digital twins of shopfloor assets in the AAS repository.
- Configure connections between shopfloor assets and their digital models supported by the out of the box support for standard protocols[1] (MQTT, OPCUA, etc.).
- Monitor and visualize the resulting data streams in a simple web-based interface.

This process ensures that data from heterogeneous equipment is updated, and stored in a structured way. The rest of this section shows the afore mentioned actions in a detailed way and can be used as a tutorial/roadmap.

4.1 CREATING THE AAS

The first step for using the platform as the interoperability layer is to create the digital twin data models (AAS) that are capable to describe each use case. Apart from the APIs to create the AAS and its Submodels/properties the platform offers integration with AASX package explorer which is a graphical tool to create AASs. The integration can retrieve data from i) the AASX Server and ii) AASX package explorer in “Rest Server Mode”. Moreover, offers a JSON import functionality according to the “Details of the Asset Administration Shell - Part 1” which describes the serialization of the AAS in various formats including JSON.

An example of the process is found below:

Export the AAS as a JSON file from AASX Package Explorer

1. Open the AASX Package Explorer.
2. Create a new AAS (resource AAS, product AAS).
3. Go to File and click Save as…
4. From the Save as type dropdown menu choose AAS JSON file (*.json), select a destination folder and save the file.

Import the AAS JSON file to the platform

1. Open the Swagger UI (http://localhost:8082/swagger-ui/index.html)
2. Navigate to Import Asset Administration Shell From JSON file (exported by AASX Package Explorer).
3. Click on Import Json AAS and then on the Try it out button.
4. Browse for the JSON file exported by the AASX Package Explorer (step 5 of the previous process).
5. Click Execute, if the import is successful a 201 status code should be returned.

Apart from the API to interact with the AAS repository a user interface has been developed to monitor the state of the AAS repository (check the hosted models).

This is accessible through the URL: http://corobHost:8000

[1] Currently the COROB platform supports the following protocols: OPCUA, MQTT, REST, S7COMM, KAFKA, Yaskawa TCP/IP Protocol, Modbus.

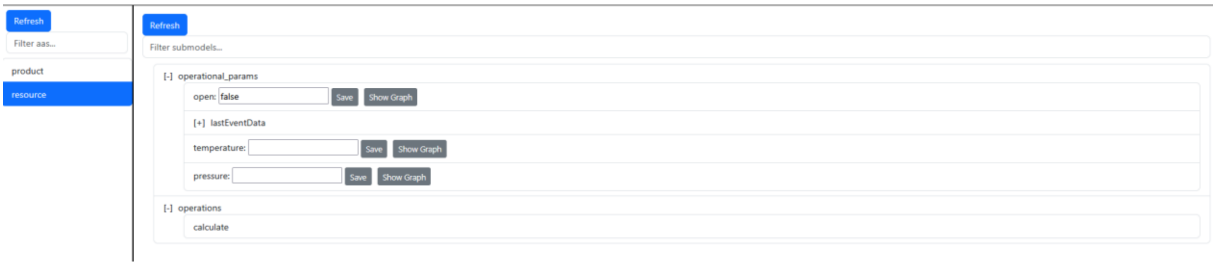

Figure 2 AAS Browser UI

4.2 CONFIGURING THE AAS

The AAS is the digital twin of its corresponding Asset. The configuration steps are provided as the linkage of the AAS to its physical counterpart (robot, cameras, sensors, quality assessment modules etc.). For the configuration a dedicated UI (Edge Connector) is provided that binds the AAS properties with the Asset by choosing an appropriate communication mechanism (OPCUA, MQTT and others[2]). This mechanism further enhances interoperability since once performed the applications talks with AAS API to the Asset behind the AAS without be concerned with the exact underlying communication protocol.

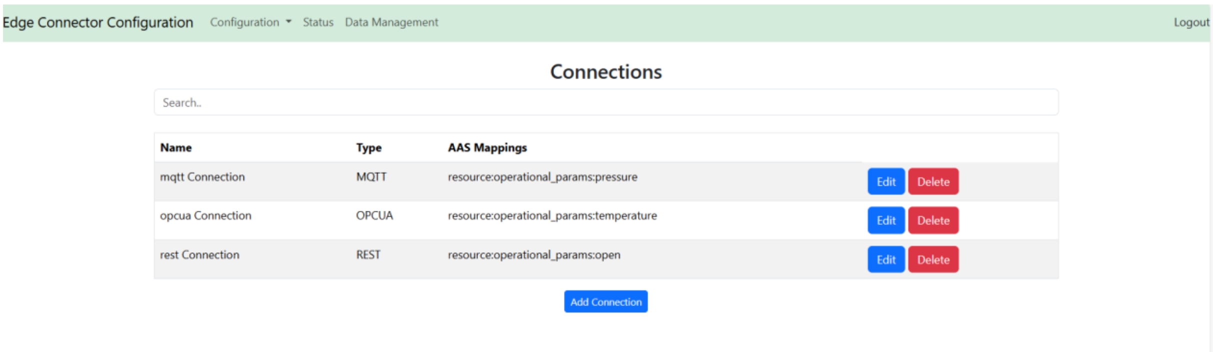

Figure 3 Connections management screen

[2] Currently the COROB platform supports the following protocols: OPCUA, MQTT, REST, S7COMM, KAFKA, Yaskawa TCP/IP Protocol, Modbus.

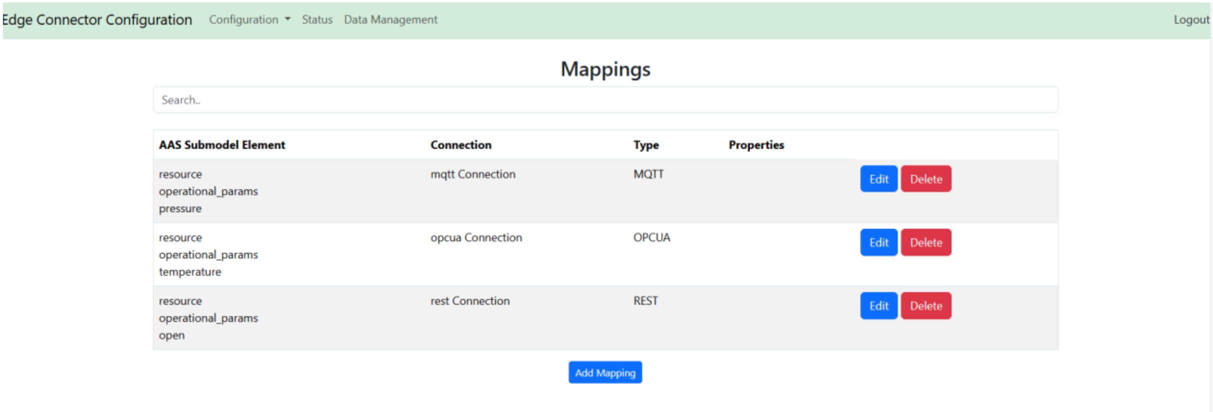

Figure 4 Mappings management screen

An example of an OPCUA configuration is provided below:

OPCUA connection configuration:

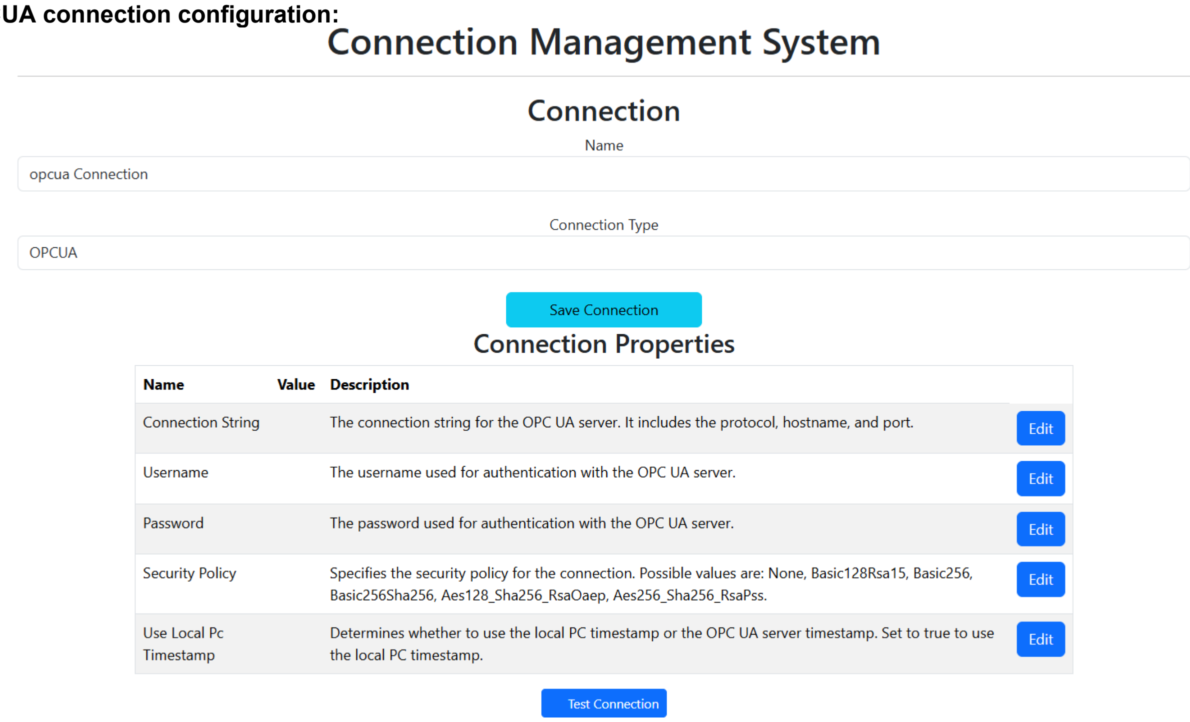

Figure 5 OPCUA Connection properties

1. Open the Edge Connector interface (http://corobHost:8089).
2. From the Configuration dropdown menu select Connections.
3. Click on the Add connection button.
4. Name the connection.
5. For connection type, select OPCUA.
6. Click on save connection.
7. After that the connection properties should appear.
8. In the Connection String field write the address of the OPCUA server (i.e. “opc.tcp://host.opcua:5353/OPCUAServer”).
9. In the Username field write the username to access the OPCUA server.
10. In the Password field write the password to access the OPCUA server.
11. Click on Test Connection button, a message should appear confirming the establishment of the connection.

OPCUA node mapping configuration

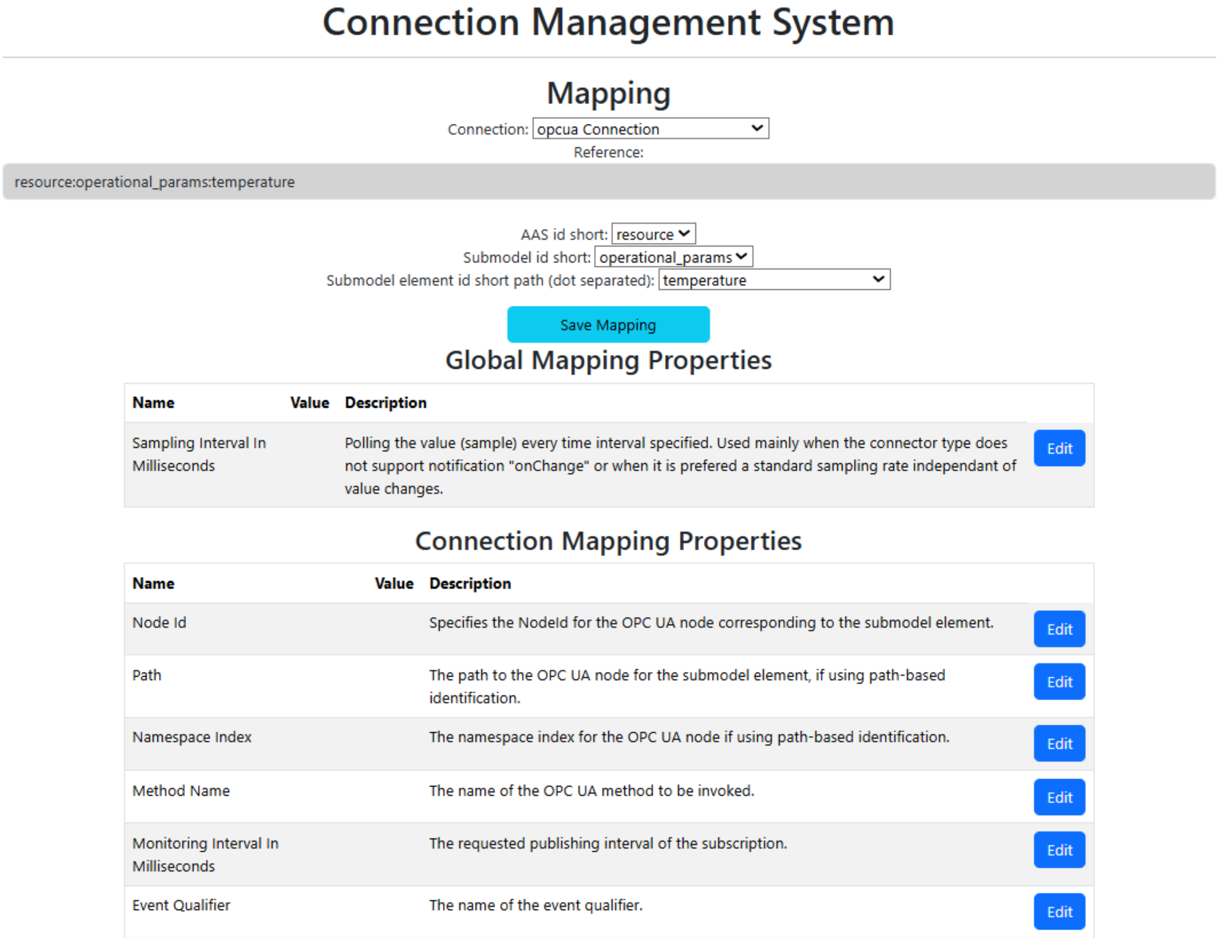

Figure 6 Mapping properties for an OPCUA connection

1. From the Configuration dropdown menu, select Mappings.
2. Click on Add Mapping.
3. For connection, select the newly created opcua connection.
4. For AAS id short, select from the drop down menu a desired AAS.
5. For Submodel id short, select from the drop down menu a desired submodel.
6. For Submodel element id short path, select from the drop down menu a desired property.
7. Click Save mapping. Connection mapping properties should appear.
8. In the field Node Id write a complete identifier (i.e. ns=6;s=MyLevel).
9. In the field Monitoring Interval In Milliseconds write in milliseconds the interval permitted by the server and suitable in the desired use case (i.e. 1000 for each second).
10. Navigate the status tab and restart the connection.

4.3 TESTING THE DEPOYMENT AND CONFIGURATION

Using the "AAS Browser UI" (http://corobHost:8000) you may browse all the deployed AAS. In case that a specific AAS property is bound through a connection to a sensor or a data source than this property will be automatically updated and its value history will be automatically persistent and available for dashboarding and/or analytics. The UI offer a limited functionality of displaying timeseries chart.

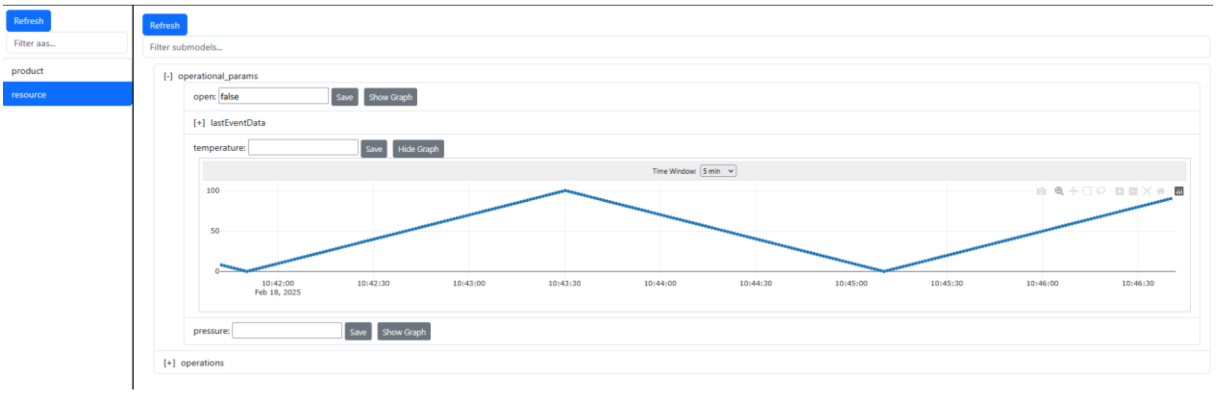

Figure 7 Timeseries chart of the AAS Browser UI

4.4 USING THE API

The platform provides two APIs to interact with other systems (applications), a REST API implementation of the Asset Administration Shell API ("Specification of the Asset Administration Shell Part 2: Application Programming Interfaces – v2.0") and an API to interact with historical values for individual AAS properties.

The afore mentioned APIs are documented by using swagger under the respective urls:

http://corobHost:8082/swagger-ui/index.html

http://corobHost:8083/data/swagger-ui/index.html

Finally, the above APIs are regulated through an inhouse OAUTH2 server restricting access in a fine-grained way based on an internal RBAC system.

5 ASSESMENT

In the Annexes sections (Sections 8.1, 8.2 and 8.3) we provide a comparison of COROB's platform with other open source AAS servers.

The COROB platform stands out for:

- Protocol support.
- Efficient performance, capable of handling sustained and high-concurrency conditions.
- Visualization tool, enabling access to platform's insights, in a visual way, without third party applications.

6 SUMMARY

This document is provided as a proof of delivery of the COROB platform. The platform has been already provided to our FSTP to work on the integration but also has been deployed to our pilot cases on premise. A set of data models have already been created for our internal test cases and updated according to FSTP's requirements. The data models are linked to various degrees with each cell's components (robots, welding machines, sensors etc.) and performance stress tests are underway.

The main advantage of the COROB's platform is that provides a turnkey solution able to handle multiple aspects of the digitization challenges from efficient data collection, data storage up to data analytics and dashboarding without compromising performance (as depicted in section 4). The AAS repository part of the platform enables interoperability providing single source of truth accessed in the same way independent of the actual underlying communication protocols. Moreover, opens perspectives to its users both in terms of data ownership and data sovereignty but also in the current AI world to feed data for training and decision support purposes.

The COROB platform addresses most industrial challenges (interoperability, semantics, data sovereignty) and simplifies the transition from legacy systems to Industry 4.0 while benchmarking confirms the platform's readiness for industrial adoption.

7 REFERENCES

[1] *Kaya F, Şanlı E, Albayrak Ö, Ünal P, Kirci P. Asset Administration Shell Tool Comparison: A Case Study with Real Digital Twins Used in Petrochemical Industry. Sensors. 2025; 25(7):1978. https://doi.org/10.3390/s25071978*

8 ANNEXES

8.1 FUNCTIONAL & THEORETICAL ASSESMENT

The "Functional & Theoretical Assesment" table below has been taken from the literature [1] and we appended a column for the platform (1st column).

Table 1 Functional & Theoretical Assessment

Criteria/aspects	COROB Platform	AASX Server	Eclipse Basyx	NOVAAS
Programming Languages	Java	C# (.NET Core, Blazor, .NET Framework)	Java (primary), Python, C++, Rust, .NET	Node-RED (low-code)
Metamodel Support	V2	V3	V3	V3: Partial
Dashboard	YES	NO	YES	YES
Data Monitoring	YES	NO	YES	YES
Import/Export AASX File	Import only (JSON)	Export only	YES	NO
Persistence	Postgres, *H2 Database*	In-memory by default	• Supports MongoDB or SQL for persistence • Offloads data from memory	AAS state in-memory with importing on start
Cloud Deployment	Cloud-ready via Docker	Cloud-ready via Docker	Kubernetes deployment	Primarily designed for on-prem/edge
Remote Access	• Web UI • API	Web UI	Web UI	Web dashboard
Edge/Device Deployment	Java-based deployment on PCs and gateways	Suitable for industrial devices	Java-based deployment on PCs and gateways	Runs on Raspberry Pi and similar Linux-based gateways
CPU and Concurrency	Multi-threaded	Multi-threaded	Multi-threaded	Single-threaded event loop
Startup Time	Moderate (a few seconds due to container startup and optional DB loading)	Fast (<2 s)	Moderate (a few seconds due to container startup and optional DB loading)	Fast (<2 s)
Distribution Modes	• Java JAR (CLI) • Docker	• CLI executable • Docker	• Java JAR (CLI) • Docker • SDK	• Docker image • Run by importing Node-RED flows

8.2 PERFORMANCE BENCHMARKING

A benchmarking test below shows performance assessment. For benchmarking two different scenarios have been used to evaluate the platform performance, one for sustainable usage (1 user, 1000 requests) and another for high concurrency (100 users, 10 requests per user). The results are shown below.

Table 2 Scenario 1 Results

Platforms	Average Response Time	Minimum Response Time	Maximum Response Time	Standard Deviation	90% line	95% line	99% line	Error %	Throughput	Received KB/sec	Sent KB/sec	Average Bytes
COROB Platform	4	3	12	0.68	5	5	6	0%	226.6546	97.55	216.92	440.7
AASX Server	1	1	17	0.75	2	2	3	0%	599.5204	124.7	174.47	213
Eclipse BaSyx	4	3	8	0.83	5	6	6	0%	230.8936	65.32	61.78	289.7
NOVAAS	3	2	30	1.41	4	5	6	0%	307.5031	78.68	94.29	262

Table 3 Scenario 2 Results

Platforms	Average Response Time	Minimum Response Time	Maximum Response Time	Standard Deviation	90% line	95% line	99% line	Error %	Throughput	Received KB/sec	Sent KB/sec	Average Bytes
COROB Platform	4	3	15	1.34	5	6	10	0%	968.9923	417.31	927.36	441
AASX Server	1	1	8	0.89	3	3	5	0%	994.0358	206.77	289.28	213
Eclipse BaSyx	5	3	111	4.83	7	9	14	0%	968.9923	274.42	259.28	290
NOVAAS	76	3	124	29.89	112	117	122	0%	591.716	151.4	181.44	262

8.3 RESOURCE UTILIZATION BENCHMARKING

Finally, a monitor of the CPU and memory usage during the test is presented below.

	IDLE CPU	IDLE MEMORY	SCENARIO1 CPU	SCENARIO1 MEMORY	SCENARIO2 CPU	SCENARIO2 MEMORY
COROB Platform	0.45%	1.18GB	5%	1.18GB	0.91%	1.19GB
AASX Server	0.2%	60.3MB	0.47%	75.1MB	0.3%	186MB
Eclipse BaSyx	1.2%	1.62GB	207%	1.74GB	406%	1.6GB
NOVAAS	0.59%	223MB	1%	270MB	140%	280MB