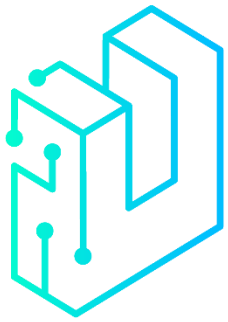

UNDERPIN

DATA SPACE FOR MANUFACTURING

Pan-European data space for holistic asset management in critical manufacturing industries

D3.1 UNDERPIN data space infrastructure, mid-term deployment and integration report

30/11/2024

Co-funded by the European Union

This project has received funding from the Digital Europe Programme under grant agreement No 101123179

Document information			
Project name	Pan-European data space for holistic asset management in critical manufacturing industries		
Project acronym	UNDERPIN		
Grant Agreement No	101123179		
Start / Duration	1/12/2023		
Project Coordinator	MOTOR OIL (HELLAS) DIILISTIRIA KORINTHOU AE		
Deliverable	D3.1 UNDERPIN data space infrastructure, mid-term deployment and integration report		
Work Package	WP3		
Authors	INNOV		
Dissemination level (PU = Public; PP = Restricted to other program participants; RE = Restricted to a group specified by the consortium; CO = Confidential, only for members of the consortium; SEN= Limited under the conditions of the Grant Agreement)	PU		
Type	Document, Report		
Due date (M)	M12	Actual delivery date	30.11.2024

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

Document history

Version	Date (DD/MM/YYYY)	Author(s)	Comments / Description
V1	10/09/2024	Odysseas Kokkinos	Working document, Table of Contents
V2	13/11/2024	Odysseas Kokkinos, Christos Betzelos, Giannis Tsichlas	Semi-complete version
V3	21/11/2024	Odysseas Kokkinos, Petar Ivanov, Aristotelis Ntafalias	First complete version with comments integration
V4	25/11/2024	Odysseas Kokkinos, Petar Ivanov, Aristotelis Ntafalias	Restructuring, segments moved to annex
V5	26/11/2024	Giannis Tsichlas, Christos Betzelos	Infrastructure description
V6	27/11/2024	Petar Ivanov	Central time-series database description
V7	28/11/2024	Vladimir Alexiev, Odysseas Kokkinos	Final Version for submission

List of Authors and Contributors

Name	Organisation
Odysseas Kokkinos	INNOV
Christos Betzelos	INNOV
Giannis Tsichlas	INNOV
Aristotelis Ntafalias	MOH
Petar Ivanov	OT
Vladimir Alexiev	OT
Yannis Stavrakas	ARC
Nikos Kapetanas	ARC

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

DISCLAIMER AND COPYRIGHT © 2023, UNDERPIN CONSORTIUM

This publication has been provided by members of the UNDEPRIN consortium. While the content has undergone review by consortium members, it does not necessarily reflect the views of any individual member. Although the information is believed to be accurate, UNDERPIN members provide no warranty, including implied warranties of merchantability and fitness for a particular purpose. None of the UNDERPIN members, their officers, employees, or agents are liable for any inaccuracies or omissions. This disclaimer extends to any direct, indirect, or consequential loss or damage resulting from the information, advice, or inaccuracies in this publication.

The same disclaimers as they apply to the consortium members equally apply to the European Union employees, officers and organizations.

UNDERPIN has received funding from the Digital European Programme under grant agreement No 101123179.

Table of Contents

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

List of Figures

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

List of Tables

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

Acronyms and Abbreviations

Table 1 Acronyms and Abbreviations

AIT	Austrian Institute of Technology
AP	Authority Portal
CaaS	Connector as a Service
DAPS	Dynamic Attribute Provisioning Service
EDC	Eclipse Dataspace Components
IaC	Infrastructure as Code
IAM	Identity and Access Management
IDS	International Data Spaces
JWT	JSON Web Tokens
IAM	Identity and Access Management
IDS	International dataspaces
MOH	Motor Oil Hellas
MORE	Motor Oil Renewable Energy
OT	Ontotext
UI	User Interface

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

1 Introduction

1.1 Purpose and scope

The document "D3.1 UNDERPIN infrastructure, mid-term deployment and integration report" presents the framework and deployment steps of the UNDERPIN dataspace components, aiming to serve the needs of a dataspace focusing on holistic asset management in critical manufacturing industries. Its primary purpose is to provide stakeholders with insights into the technological foundation required for developing use cases and components that facilitate secure data sharing and interoperability across various systems. The report emphasizes compliance with data protection regulations, such as GDPR, and highlights the importance of secure software development practices.

Within the document a detailed description of the platform architecture is included as well as of the cloud infrastructure deployment steps and the integration strategies employed within the project. It covers security measures implemented across the system, such as HTTPS communications and OAuth2.0 authentication, while also discussing resource requirements for hosting infrastructure. Lastly, it describes the functionality of deployed connectors enabling efficient data exchange and policy-based access control while ensuring that sensitive information is safeguarded and collaboration among organizations is promoted.

1.2 Structure of the document

The structure of the document is as follows:

- Section 1 presents a general overview of the scope and purpose of the document
- Section 2 provides the architectural composition of the deployed system, deployment strategies followed and cloud infrastructure specifications
- Section 3 describes EDC connector functionalities, features and deployment process as well as future advances planned.
- Section 4 describes AP functionalities, features and deployment process as well as future advances planned.
- Section 5 describes DAPS functionalities, features and deployment process as well as future advances planned.
- Sections 6 to 9 give an overview of the additional data space components envisioned as parts of the UNDERPIN dataspace. These include a blockchain client for smart contracts, a central time-series database, and components of the semantic layer for metadata editing, storage and search
- Section 10 provides with the steps taken for dry-run testing the deployed platform
- Section 11 offers a general view of the foreseen advances and developments in the UNDERPIN dataspace components.
- Section 12 wraps up with some concluding remarks.
- Finally, Appendices A and B provide technical details about EDC and AP deployment configurations

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

2 Platform Architecture

2.1 UNDERPIN Component-level Architecture

The UNDERPIN system architecture considers the requirements as laid down in D2.1 and provides a technological basis for future development of use cases and components. Note that the IDS components to form the IDS system layer architecture are seamlessly integrated into the overall system architecture. However, there are services which are not involved in the IDS-based data sharing, and which are therefore not part of the IDS system layer.

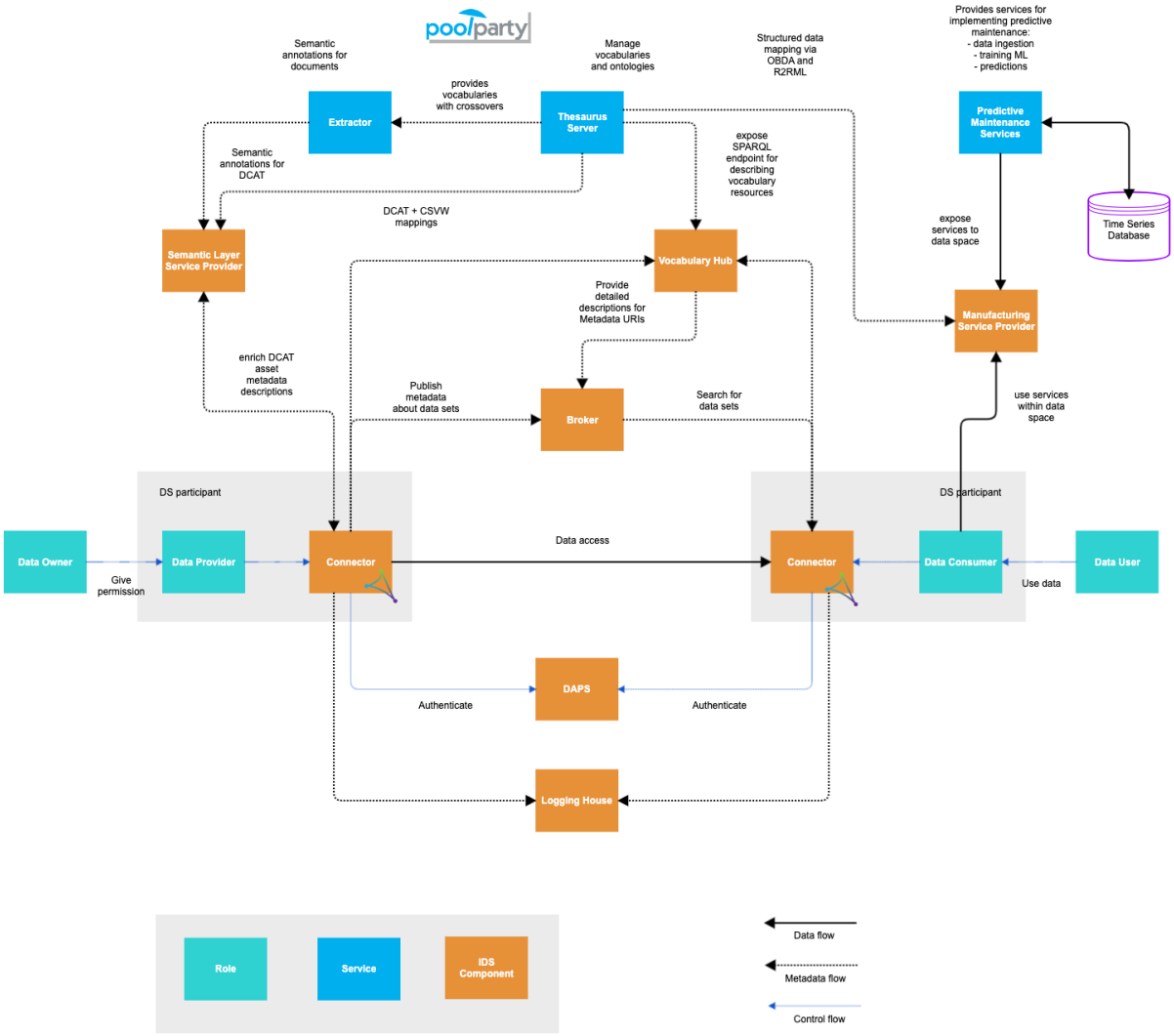

Figure 1 UNDERPIN architecture

2.2 Cloud Infrastructure

Deployment and Integration

Each deployed UNDERPIN component is containerized using Docker[1]. The container orchestration tool used is Kubernetes. Wherever possible, we have also used Helm charts and

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

the “Infrastructure as Code” approach for managing the infrastructure. This approach facilitates scalable, repeatable, flexible and more easily manageable deployment in cloud environments. Communication between components is secured using HTTPS, with appropriate authentication mechanisms such as OAuth2.0 and JSON Web Tokens. Furthermore, each service implements comprehensive logging and monitoring capabilities to ensure high availability and rapid issue resolution.

Compliance and Standards

All components comply with relevant data protection regulations and adhere to industry best practices for secure software development and deployment. The software is designed with scalability, maintainability, and security in mind.

Digital Infrastructure

The first deployment of the data space was carried out by a subcontractor and hosted for a period of three months in their infrastructure, which is built on Microsoft Azure. This deployment included all the required components for a functional data space, such as the Authority Portal (AP), DAPS and EDC connectors. The second deployment is now being managed by INNOV and is hosted on Hetzner infrastructure implementing a Pay-As-You-Go pricing scheme for optimal efficiency. In addition to the core components such as AP, DAPS, and EDC connectors, this deployment includes also other components that we added to UNDERPIN: blockchain for smart contracts, InfluxDB for easier time-series analytics, Elasticsearch for faceted metadata search, GraphDB for storing semantic metadata and PoolParty for editing and viewing metadata.

The cloud-based setup provides robust scalability, flexibility, and high availability, essential for the efficient operation of the project's Data Space infrastructure. For identity and access management, UNDERPIN uses Keycloak[2], an open-source IAM Overall infrastructure integrates all components seamlessly, ensuring secure communication and data exchange. The infrastructure's security measures include:

- HTTPS and TLS: All communications are encrypted using HTTPS and TLS to protect data in transit
- OAuth2 and JWT: These authentication protocols are used to secure API communications and ensure that only authorized entities can access sensitive data and services
- Monitoring and Logging: Comprehensive monitoring and logging are implemented using Prometheus and Grafana, providing insights into system performance and security events

This infrastructure setup ensures that the UNDERPIN project components are deployed in a secure, scalable, and efficient manner, leveraging the provided capabilities of the Hetzner infrastructure and Keycloak to meet the project's operational and security requirements.

Estimated infrastructure resources:

The determined required resources of the hosting infrastructure based on the foreseen needs at peak load for the various dataspace enablers include five servers as follows:

- Dataspace core components (AP, DAPS): 6 cores, 64GB RAM, 1Tb SSD
- Elastic: 6 cores, 128-256GB RAM, 1Tb SSD

[2] https://www.keycloak.org/

- Influx+Telegraf: 6 cores, 128GB RAM, 1Tb SSD
- GraphDB + PoolParty (DCAT editing and search): 6 cores, 64GB RAM, 1Tb SSD
- Analytics: 6 cores, 64-128GB RAM, 1Tb SSD

2.2.1 First deployment on Azure

The first iteration of the UNDERPIN data space was deployed by the subcontractor as a service for a 3-month period on Microsoft Azure[3]. This deployment included only core data space components (DAPS and AP) as well as Connectors-as-a-service.

2.2.2 Second deployment on Hetzner

The second iteration of the UNDERPIN data space is deployed by the consortium on Hetzner servers. Hetzner[4] is a German data-center operator with servers located in Germany and Finland, ensuring all components and data hosted in the data space would remain in the EU.

Based on the infrastructure resources listed previously it was estimated that costs for hosting the whole data space on Hetzner would be 20-25 times lower compared to offers by European third-party Azure hosting providers. Current and future connectors provided by the subcontractor as a service can be configured to connect to the data space on Hetzner, so participants who would prefer to use the subcontractor's services are able to do so.

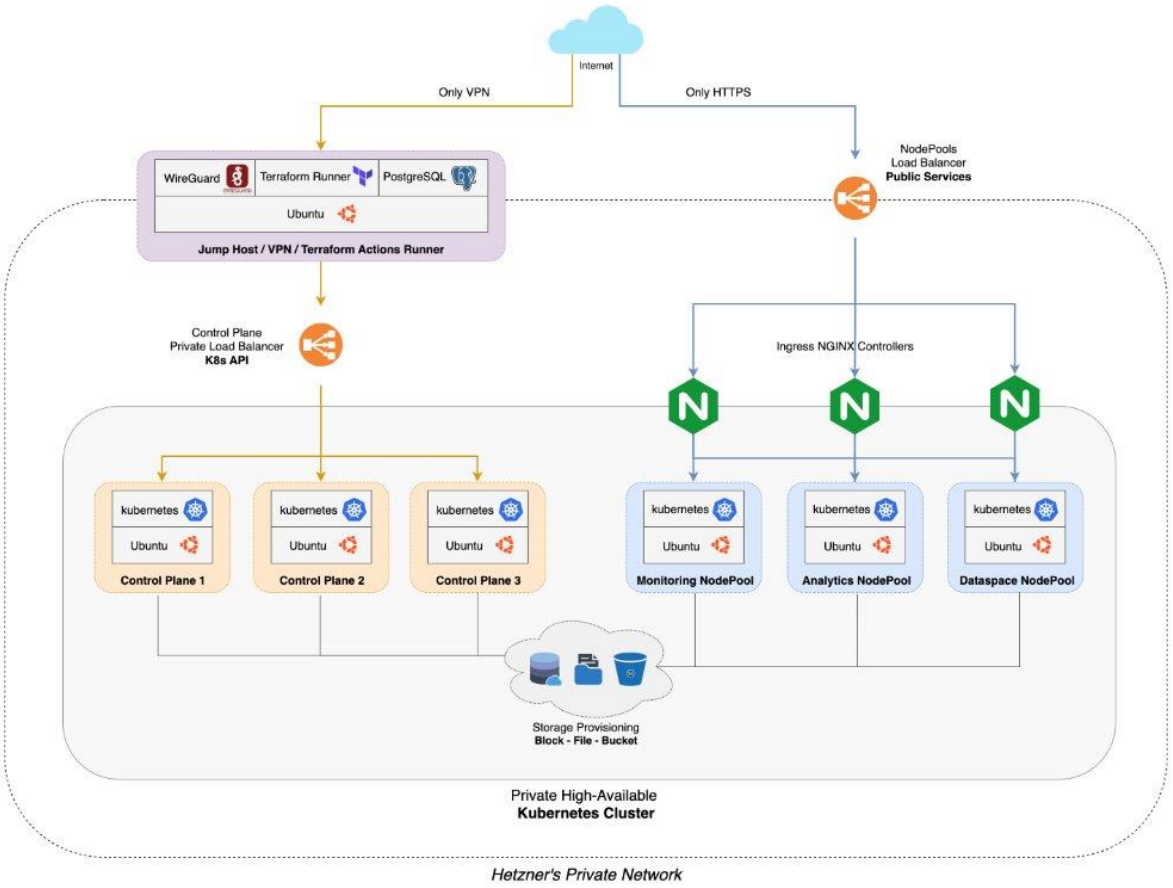

Figure 2 UNDERPIN data space deployment on Hetzner

[3] https://azure.microsoft.com/
[4] https://www.hetzner.com/

In Figure 2, the UNDERPIN infrastructure architecture is depicted in a high-level diagram. The main infrastructure component is a Kubernetes cluster built on top of Ubuntu VMs. The cluster is highly available since it consists of three control-plane nodes. The worker nodes are split into node pools according to each specific use case. The services are exposed using Hetzner's Load Balancer and Ingress NGINX controllers. The management–configuration flow is takes place through an extra VM used as a Jump Host and VPN.

In Figure 3, the UNDERPIN GitOps – CI/CD flow is depicted, following the Infrastructure as Code (IaC) principles. There are 3 git source codes repositories (the figure shows only the 2nd and 3rd as they are more relevant to describe IaC).

- Dataspace Repository: source code for the custom components, such as metadata editor, metadata search, GraphDB and PoolParty integrations into the dataspace.
- Infrastructure Repository: This is a terraform repository responsible for maintaining and reconciling the Hetzner resources (VMs, firewalls, Load Balancers, etc) into the infrastructure. The CI/CD process is developed using the GitHub actions and the Hetzner terraform provider.
- Kubernetes GitOps Repository: This is a Kubernetes repository responsible for maintaining and reconciling the Kubernetes objects into the cluster (Helm charts, plain Kubernetes manifests, etc). The CI/CD process is developed using the ArgoCD for syncing the Kubernetes cluster objects and the HashiCorp Vault for secret management.

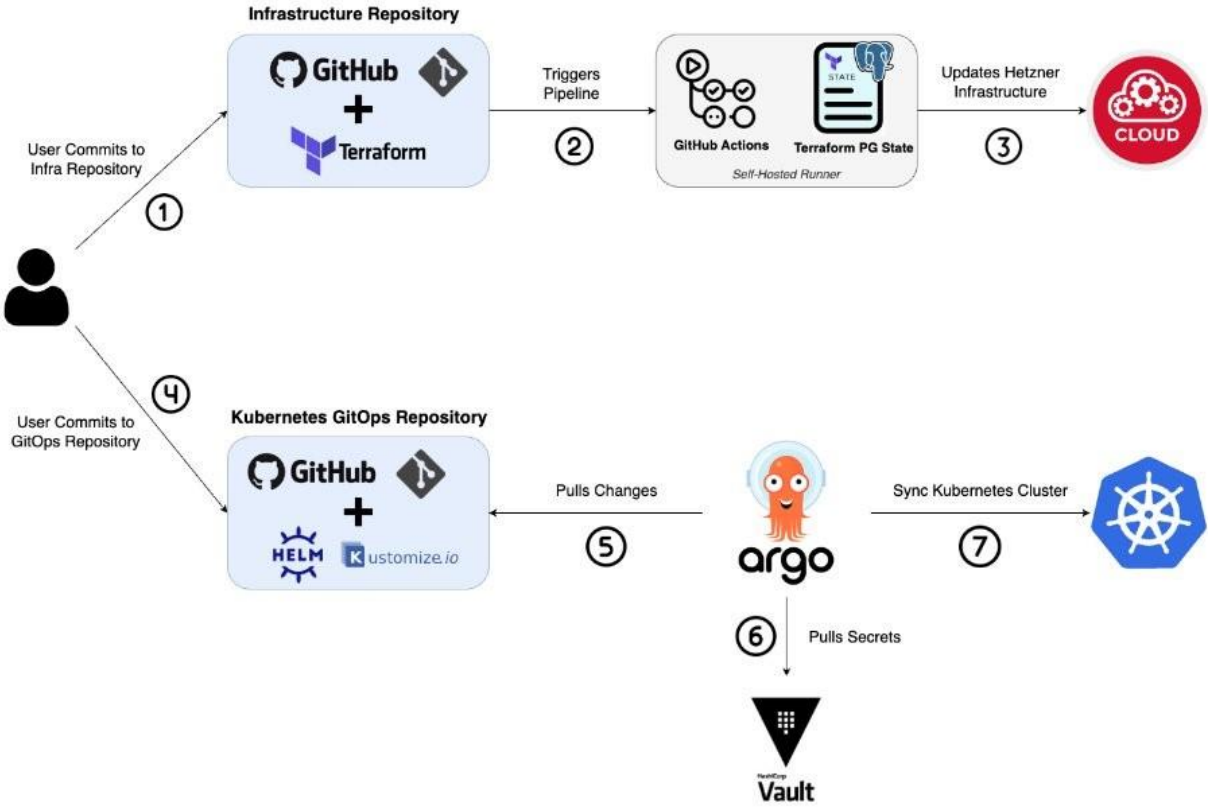

Figure 3 GitOps Structure

3 Authority Portal

3.1 Authority Portal description

The deployed UNDERPIN Authority Portal acts as the central hub for managing dataspace participants and connectors across the dataspace. This component is vital for ensuring controlled access to the dataspace, enabling UNDERPIN dataspace authorities to maintain compliance with regulatory requirements and organizational standards. In addition, it holds the central federated Data Asset Catalog.

URL for UNDERPIN Authority Portal:

https://ap.dataspace.underpinproject.eu

3.2 Authority Portal features

1. **Centralized Identity and Access Management**

The Authority Portal serves as the central hub for managing participants and organizations across the dataspace. This feature enables dataspace authorization mechanisms ensuring that all participants in the dataspace are authenticated and authorized consistently.

2. **Role-Based Access Control (RBAC)**

The Authority Portal supports Role-Based Access Control, allowing authority and organization administrators to assign specific roles to users and manage their access rights based on organizational requirements. This feature ensures that users have access only to the resources they need, reducing the risk of unauthorized access and enhancing overall security.

3. **User-Friendly Interface**

The Authority Portal features an intuitive user interface that simplifies the management of participants and organizations. Its user-friendly design ensures that administrators can easily navigate the authority portal, configure settings, and manage user access without needing extensive technical expertise.

4. **Scalability and Flexibility**

Built to handle both small and large-scale deployments, whether deployed as a service or on-premises, the authority portal is scalable and flexible. It can adapt to the growing needs of organizations as their user base expands, ensuring consistent performance and reliability. The system's architecture supports customization and extension on code-level to meet specific organizational requirements.

5. **Customizable User Interface and Branding Options**

The portal supports customizable user interface options, allowing organizations to tailor the look and feel of the Authority Portal to match their branding guidelines. This customization enhances user experience by providing a familiar interface aligned with organizational aesthetics.

6. **Comprehensive Documentation and User Guides**

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

The Authority Portal comes with detailed documentation and user guides that provide step-by-step instructions for setting up, managing and using the portal. These resources help users quickly become proficient with the system and leverage its full capabilities[5].

7. **Audit-Ready Reporting and Data Export Options**

The portal allows users to generate comprehensive reports, which can be exported for external analysis or audit purposes. This feature supports transparency and accountability, ensuring that dataspaces can easily demonstrate compliance with internal and external standards.

8. **Apache 2.0 License**

Same as in EDC connector (*section 3.2 feature 9 of this document*).

URL for Open-Source repository: https://github.com/Sovity/authority-portal

3.3 Deployment

The configured Dataspace Deployment Environment has a running Keycloak DAPS. To make use of the Data Catalog, the configured Dataspace Deployment Environment has a running Catalog Crawler configured (based on the sovity EDC Connector).

3.3.1 Third Party

- Information about a running instance of the sovity CaaS-Portal
  - URL of the CaaS-Portal, [CAAS_PORTAL_FQDN]
  - URL of the Keycloak for authorizing at the CaaS-Portal, [CAAS_KC_FQDN].
  - Credentials for the CaaS-Portal, [CAAS_CLIENT_ID] and [CAAS_CLIENT_SECRET].
- Running instance of Uptime Kuma
  - It tracks the DAPS, Catalog Crawler and Logging House statuses
  - The statuses are available via the API (`/metrics` endpoint)
    - The output per component:

```
monitor_status{monitor_name="[Component name]", ...} [INTEGER]
```

  - URL of the Uptime Kuma, [UPTIME_KUMA_FQDN].
  - API key for the Uptime Kuma, [UPTIME_KUMA_API_KEY].

3.3.2 Deployment Units

Table 2 AP Deployment units

Deployment Unit	Version / Details
Reverse Proxy / Ingress	Infrastructure dependent
Keycloak Deployment	Version 24.0.4
OAuth2 Proxy	quay.io/oauth2-proxy/oauth2-proxy:7.5.0
Caddy behind OAuth2 Proxy	caddy:2.7
Authority Portal Backend	authority-portal-backend latest version[6]
Authority Portal Frontend	authority-portal-frontend latest version[7]

[5] https://ap-ce.docs.sovity.de/
[6] CHANGELOG.md
[7] CHANGELOG.md

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

Catalog Crawler	ghcr.io/sovity/catalog-crawler-ce version[8]
Postgresql	Version 16 or compatible latest version[9]

3.3.3 Configuration

Deployment configuration for the various AP components can be found in Annex A – EDC Deployment Units Configuration.

3.3.4 Data Catalog Crawlers

- The Data Catalog only displays the Data Catalog as it exists in the database.
- Each deployment environment requires a Data Catalog Crawler.
  - A Data Catalog Crawler is based on the EDC Connector and crawls the catalogs of all connectors in the dataspace.
  - For the deployment an SKI/AKI client ID was used to register the crawler.

3.3.5 Initial Setup

The first user that registers at the portal does not need to be approved and will automatically become an `Authority Admin`.

[8] CHANGELOG.md
[9] CHANGELOG.md

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

4 DAPS

4.1 DAPS description

The deployed DAPS (Dynamic Attribute Provisioning Service) is responsible for issuing and validating dynamic security tokens, which are essential for authenticating and authorizing data exchanges between different connectors within the UNDERPIN dataspace. This component plays a critical role in establishing trust within the dataspace by ensuring that only authorized participants can access and share data, thereby maintaining compliance with security and privacy standards.

URL for UNDERPIN DAPS:

https://keycloak.dataspace.underpinproject.eu/realms/underpin-DAPS

4.2 DAPS features

1. **Dynamic Security Token Issuance**

The DAPS issues dynamic security tokens that are crucial for authenticating and authorizing connectors and communication between connectors in a dataspace. These tokens ensure that only verified and trusted entities can participate in data sharing, providing a secure foundation for interactions.

2. **Trust Framework Compliance**

DAPS operates in compliance with trust frameworks, ensuring that all issued tokens adhere to specific security and privacy standards. This compliance is critical for maintaining a secure dataspace, where data exchanges are governed by a clear set of rules and trusted entities.

3. **Attribute-Based Access Control (ABAC)**

The service supports Attribute-Based Access Control, allowing organizations to define access policies based on participant attributes such as their ID within the dataspace. This flexibility enables more granular control over data access.

4. **Interoperability with Various Systems and Connectors**

DAPS is designed to work seamlessly with a variety of systems and connectors. This interoperability makes it a versatile component for managing security in diverse IT environments, where multiple systems and partners must work together authorized and securely.

5. **Scalability and High Availability**

Designed to support high volumes of data exchanges being based on the industry standard Keycloak, the DAPS is scalable and can handle a growing number of tokens and transactions without compromising performance. Its architecture ensures high availability, providing continuous operation even during peak load.

6. **Configurable Token Lifetimes and Security Levels**

Administrators can configure token lifetimes and security levels based on specific organizational needs or compliance requirements. This feature allows for the customization of security settings to match different risk profiles and operational scenarios, ensuring appropriate levels of protection for various use cases in a dataspace.

This project has received funding from the Digital Europe Programme under grant agreement No 101123179

7. **Secure Communication Channels**

DAPS uses secure communication channels to transmit tokens and related data, ensuring that sensitive information is protected during transit. This feature is critical for maintaining the confidentiality and integrity of security credentials and minimizing the risk of interception or tampering.

8. **Apache 2.0 License**

Same as in EDC connector (*section 3.2 feature 9 of this document*).

URL for Open-Source repository: https://github.com/Sovity/Sovity-daps

4.3 Deployment

DAPService is based on the Keycloak IAM platform. Using Docker Compose everything is automatically build on startup. To use it, we copied the file `.env.example` to `.env` and assigned secure secrets, as described in the file. This setup uses PostgreSQL; however, any database supported by Keycloak can be used. The setup is configured behind a TLS-terminating reverse proxy and the following environment variable adjustments were needed:

```
KC_PROXY=edge
```

```
KC_HOSTNAME=keycloak.example.com
```

Additionally, the `start-dev` command was removed from the `docker-compose.yml`.

Further configuration options can be found in the official Keycloak documentation[10].

4.3.1 Realm Configuration

For least friction, we created a separate realm, commonly named "DAPS". In this new realm, several modifications to pre-existing global defaults were made:

- Changed all pre-existing client scopes to type "None"
- Created a new client scope of type "Optional" with name `idsc:IDS_CONNECTOR_ATTRIBUTES_ALL`

4.3.2 Managing Participants

Participants are represented as Keycloak service accounts. These should have the "DAT Mapper", a custom protocol mapper, configured and assigned to them.

[10] https://www.keycloak.org/server/all-config

5 EDC Connector

5.1 Connector description

As part of the ongoing UNDERPIN project, subcontractor has successfully deployed key components of the UNDERPIN dataspace. These components are designed to facilitate secure and efficient data exchange across different systems and organizations within the UNDERPIN dataspace.

Four connectors have already been deployed in the subcontractor digital infrastructure as CaaS for four different project partners (MOH, MORE, AIT, OT) and serve as the first example of connectors of the UNDERPIN dataspace that can link data-sources to assets and policies and enable data exchange in the dataspace.

5.2 Connector features

1. **Comprehensive Data Exchange Management**

Connectors are digital structures that, when combined with other dataspace components. provide a reliable infrastructure for transferring data across various platforms, ensuring that data are transferred seamlessly while maintaining high standards of integrity and confidentiality. This feature is essential for creating a cohesive data ecosystem where information can be shared and accessed as needed.

2. **Advanced Policy-Based Access Control**

One of the core features of the connectors is their ability to enforce policy-based access controls by using the Open Digital Rights Language (ODRL) ontology. Connector users can define policies that dictate how, when, and by whom data assets can be accessed or shared by using access- and contract-policies. These policies can be customized to meet specific organizational needs, ensuring that data exchange complies with internal guidelines as well as external regulations. This functionality is critical for organizations that need to safeguard sensitive information while enabling collaboration across organizations.

3. **Enhanced Security and Compliance Measures**

Security is a foundational element of the connectors, which are built to adhere to stringent security protocols and privacy standards. They provide mechanisms for authentication and authorization between each other, ensuring that only authorized entities can communicate. Additionally, the connectors support encryption to protect data both in transit and at rest. By maintaining detailed logs of all data transfers in the transfer-history, providing and consuming, the connectors help being compliant with regulatory requirements and internal security policies.

4. **High Interoperability and Compatibility**

The connectors are designed on the EDC Connector framework, which supports a wide range of data exchange protocols and standards. This makes the connectors highly interoperable by using the Dataspace Protocol and compatible with different data-source and data-sink systems, regardless of the underlying technologies. This feature is particularly valuable for organizations that need to integrate with various partners, customers, or vendors, as it simplifies the process of connecting diverse data environments.

5. **Scalability and Adaptability**

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

The already deployed connectors by the subcontractor provide limited scalability and adaptability. Currently they are designed to handle short term foreseen volumes and types of data exchanges, to serve project use cases development. In the future, connectors will be customized and extended on code-level to fit advanced needs and requirements aligning with UNDERPIN growth.

6. **Comprehensive Monitoring and Auditing Capabilities**

The connectors provide monitoring and auditing capabilities, allowing organizations to track all data transfer activities in real-time by using the transfer history. This feature is essential for maintaining transparency and accountability, as it enables dataspace participants to monitor data flows, detect any unauthorized access or anomalies of not working transfers, and take corrective actions if necessary. Detailed logs also support those efforts. At the moment connector specific logs are accessible only through said connector. In future adaptations, this would be remedied by implementing smart contracts, as described in section 3.4 of this document.

7. **User-Friendly Interfaces**

The connectors offer intuitive user interfaces that simplify the process of configuring and managing assets, policies and transfers, making it easier for connector users to oversee and control the data offerings and data transfers without needing extensive technical expertise.

8. **Extensive Backend APIs via API-Wrapper and Management-API**

The connectors offer a range of front- and backend APIs through the API-Wrapper and Management-API respectively, allowing for programmatic access to core functionalities. These APIs enable developers and system integrators to automate tasks such as asset and policy management, data transfers and monitoring, enabling advanced data management and customization.

9. **Apache 2.0 License**

The software is open-source and licensed under the Apache 2.0 License, which offers several significant advantages. This licensing allows users to freely use, modify, and distribute the software, fostering an environment of collaboration and innovation. By being open-source, the software benefits from continuous community contributions, leading to rapid improvements, enhanced security, and quicker identification and fixing of bugs. The Apache 2.0 License also provides a strong legal framework that protects contributors and users by clearly defining the terms under which the software can be used, thereby encouraging widespread adoption and adaptation of the software across different organizations and projects.

URL for Open-Source repository: https://github.com/Sovity/edc-ce

5.3 Deployment

For deploying the EDC connector, it is noted that a running DAPS (presented in Chapter 5) that follows the subset of OAuth2 as described in the DSP Specification is in place. Also, the respective SKI/API pair and .jks file have been generated. Finally, valid Participant IDs / Connector IDs, configured in DAPS have been provided by the subcontractor.

The following units have been deployed and configured as part of the EDC deployment.

Table 3 EDC Deployment units

Deployment Unit	Version / Details
An Auth Proxy / Auth solution of your choice.	v7.5.0
Reverse Proxy that merges multiple services and removes the ports	nginx
Postgresql	13, one for the EDC-data one for the LH-data
EDC Backend	edc-ce[11]
EDC UI	edc-ui[12]

5.3.1 Configuration

Deployment configuration for the various EDC components can be found in Annex A – EDC Deployment Units Configuration.

5.4 Future adaptations

UNDERPIN intends to introduce a Smart Contract implementation as a component to cover the Logging House functionality. By integrating Smart Contracts into the connectors, we can automatically log actions, like the establishment of a contractual agreement or the exchange of data, to the central Logging House.

[11] CHANGELOG.md
[12] CHANGELOG.md

6 Blockchain Node for Smart Contracts

6.1 Blockchain Node description and features

The main role of the blockchain-based network is to serve as a logging house, which means to record information related to important events in the data space. This information can serve to create an overview of the activities in the system or to dig into a specific event to verify the correct working of the system or to verify the compliant behaviour of the participants. An important feature is non-repudiability, i.e. a user cannot claim they did not sign a license agreement, or did not access a dataset. So far, we have identified the following important events:

1. a consumer accepts a contract
2. the provider concludes the contract with the consumer
3. a data access has been requested
4. a data exchange has been initiated

We employ Ethereum as a blockchain ecosystem and one or more smart contracts offer the functionality. There is a clear distinction between smart contracts and contracts in the legal sense. The smart contracts in UNDERPIN will not substitute legal contracts, but will refer to them. The Ethereum network is a private one that is not connected to the global public Ethereum network. As a consequence, the build-in cryptocurrency Ether still exists in the private network but has no market value. The employed consensus mechanism is Proof-of-Authority, which means that only a specified set of nodes can add new blocks to the blockchain. Participants of the data space will have the option to set up their own Ethereum node, but the installation is optional. A participant without its own Ethereum node can connect directly to a dedicated bootnode. The Ethereum network can operate with just a single node, but when it is set up for production it should consist of at least three nodes operated by different organisations. This way, the network has a higher fault tolerance and entries on the blockchain cannot be manipulated easily by a single organisation.

6.2 Blockchain Node integration

There is an API layer for a convenient programmatic access to interact with the smart contracts. Both the Ethereum client software to run a node and the API layer are available as docker containers. The connector can write entries to the blockchain via the API. There might be some security mechanism to allow the connector to be run as-a-service while the blockchain components run on local premises of a participant without risking an unauthorised access to the Ethereum network.

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

7 Central Time-series database

7.1 Time-series database description and features

UNDERPIN is developed as a data space for manufacturing that enables dynamic asset management and predictive / prescriptive maintenance. Consequently, the data assets shared within the data space are time series datasets containing sensor measurements. These datasets need to be processed (sometimes in near-real-time) in order to provide timely predictions of upcoming component malfunctions and failures. This necessitates the use of a dedicated time-series database to store, correlate, query and analyse the data, rather than use disparate spreadsheets or CSV files. For example, when a time-series is broken into hours or days (because of size restrictions or batching requirements), ingesting all these slices into one "table" allows smooth analysis even on the boundaries between slices.

Data providers do not necessarily have the technical expertise to deploy and maintain a dedicated time-series database on their own premises. Furthermore, a wide range of such databases exists, the use of which would require data consumers to adapt their data pre-processing and analysis workflows to each distinct time-series database deployed by a data participant. To address both issues and make the UNDERPIN data space more accessible to future participants, the consortium decided to deploy a central time-series database within the data space.

The central time-series database would be available to all participants to publish their time series data assets in their own separate buckets. Each data asset is to be made available to other data space participants based on the access policies and data offerings that the data provider has set for this specific data asset, and contracts signed by data consumers. Access to the central time-series database is available only to members registered in the Authority Portal; the dataspace users and their access rights are replicated from AP to the time-series database.

The database selected for this purpose is InfluxDB[13], developed by the company Influxdata[14]. It is the leading open-source time series database for metrics, events, and real-time analytics and is made available under Apache 2.0 and MIT licenses.

7.2 Time-series database integration

InfluxDB is available in 3 major versions, of which v3 is currently in pre-release and is still under active development. To ensure stable and extensive functionality, the consortium has decided to use InfluxDB v2. It is made available as a docker image and comes with integrations for Telegraf, Grafana and Chronograf for metrics collection, data visualization and dashboarding, respectively.

Access to InfluxDB's range of functionalities is made available through the InfluxDB UI, a command-line interface, as well as a REST API. Its built-in visualizations and dashboarding functionality make it an excellent choice for presenting the results of predictive analysis.

[13] https://docs.influxdata.com/influxdb/v2/
[14] https://www.influxdata.com/

8 Metadata Broker

8.1 Metadata Broker description and features

The metadata Broker, which is a connector itself, provides endpoints for registration, publication, maintenance, and query of data asset metadata, called Self-Descriptions. The Broker is a central service on the IDS system layer, but contains metadata only, while data sharing is done mutually between two Connectors.

At a later stage, when the metadata Broker is deployed, UNDERPIN will base the metadata Broker on the EDC-CE and develop graph-based metadata search based on Semantic Web standards; in particular the following ontologies:

- Data Catalog (DCAT) for dataset descriptions. We'll use more features of this ontology (e.g. dcat:DataSeries) than are supported in EDC
- CSV on the Web (CSVW) for column-level descriptions, enabling more precise search and facilitating Influx ingest.
- RDF properties for column definitions that can be reused between datasets
- Quantities, Units and Datatypes (QUDT) for describing quantity kinds (e.g. Temperature) and units (e.g. DEG_C)

8.2 Metadata Broker integration

The metadata Broker will result as an extension of the Data Catalog which is integrated within the Authority Portal, allowing users to use both the Authority Portal and Data Catalog seamlessly in one location. The Data Catalog is exclusively available to Data Space members, thereby enhancing the benefits of their membership. Access to the Data Catalog requires registered membership through the Authority Portal.

For the Data Catalog, we aim to provide extended semantic search capabilities beyond the IDS metadata descriptions. For this we will:

- Implement Faceted Search at the level of columns, to enable finding datasets that have data about e.g. "bearing temperatures", as opposed to "bearings" and "temperature" separately. We will use GraphDB and Elastic as backend, and PoolParty as frontend.
- Integrate the k*nowDE* (knowledge-based recommendations) tool from the project partner ARC as a semantic search assistant into the Broker search component.

When successfully integrated, KnowDE will enable semantic exploration and discovery of assets within an EDC dataspace, leveraging both domain-specific and general knowledge. Users will be able to explore dataspace assets through natural language queries, which the system enriches using multiple knowledge bases (DBPedia, ConceptNet or a domain-specific ontology). This would particularly suit organizations who need to help users discover and understand available data assets through semantic search and visual exploration, rather than requiring exact technical queries.

KnowDE could use GraphDB from the project partner OT to store the semantic graphs it needs. To achieve this integration between Sovity components, GraphDB, and KnowDE, we plan to use a containerized flask application (e.g., Docker), with separate containers for KnowDE, GraphDB,

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

and Sovity components, orchestrated using Kubernetes. The overall architecture is depicted in the following picture.

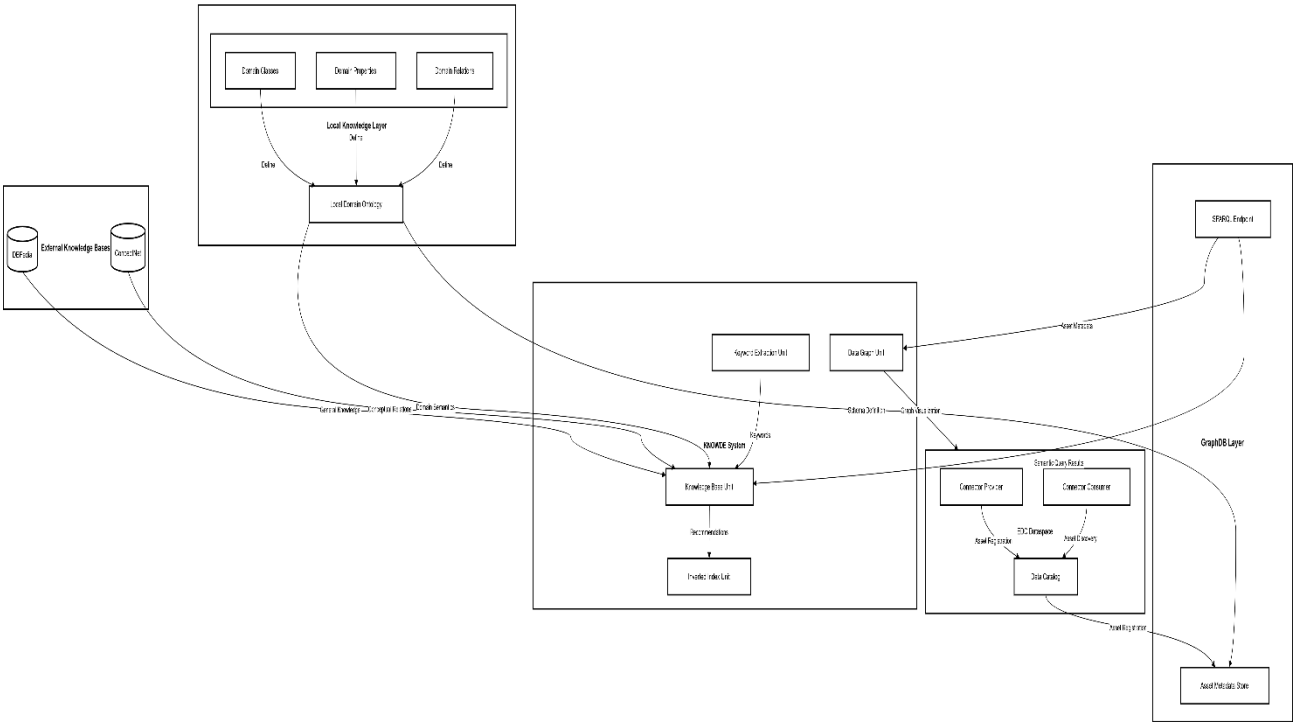

Figure 4 Metadata broker general architecture

The integration of AP and the Catalog is already in place, so that the Data Catalog “knows” the user information (access rights, authentication) of a user accessing the Data Catalog component.

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

9 Vocabulary Hub

9.1 Vocabulary Hub description and features

Interoperability requirements in International Data Spaces (IDS) mandate the use of standardized, commonly understood terms for describing data, services, and contracts, which are organized into vocabularies. Vocabularies can range from simple lists of controlled terms to more complex ontologies, and they need to be shared between parties via digital Catalogs or as Linked Data.

In the IDS context, vocabularies must be machine-readable, include descriptions and titles, and support lookups of terms. These are encoded in RDF (Resource Description Framework) as part of the IDS Information Model, which serves as a shared, minimal vocabulary for all IDS participants. However, to address domain-specific needs, the basic model can be extended with additional vocabularies. Within the UNDERPIN project, the vocabularies used to describe data assets' metadata will be adapted and expanded to conform to semantic standards, as originally envisioned in the IDS Reference Architecture Model[15]. The IDS Vocabulary Hub[16] facilitates this process by hosting, maintaining, and publishing extended vocabularies. It provides IDS-compliant endpoints for seamless integration with IDS components, manages terms and their versions, and supports the administration of data schemes used in IDS use cases.

To this end, IDSA has so far provided an outline of the expected capabilities and features a Vocabulary Hub should support. However, there is not yet a specification document in regards of the interfaces the Vocabulary Hub should implement in regards of the DataSpace Protocol. There exists a reference implementation based on the IDS Messaging Protocol supported by IDS based connectors[17] however this cannot be used with EDC based connectors. Therefore, UNDERPIN will provide a Vocabulary Hub compatible with the EDC-CE connectors, utilizing the inter-connector messaging capabilities of those. As mentioned above the Vocabulary Hub is yet another connector allowing other connectors to interact with it utilizing the same authorization and authentication principles.

The Semantic Vocabulary Hub being developed by OT and SWC includes description of datasets (using DCAT) and their columns (using CSVW and semantic property metadata). It uses OT's GraphDB as a backend, SWC's PoolParty as a frontend, and enables faceted search and provisioning of ingest to InfluxDB (the central time-series database). Access to the data assets' descriptions is provided via REST APIs and a SPARQL endpoint.

9.2 Future adaptations

Through the implementation of the Semantic Layer a set of unified services will be provided to enable semantic interoperability in dataspaces. It is based on the Vocabulary Hub component to provide data descriptions, consolidation and integration to make it easier for participants to make best use of the data available in a dataspace. The Vocabulary Hub will be integrated with the Metadata Broker in the sense that Vocabularies and other relevant resources will be synced to the Metadata Broker, enabling rich semantic search capabilities and enhanced user experience.

[15] https://docs.internationaldataspaces.org/knowledge-base/ids-ram-4.0
[16] https://docs.internationaldataspaces.org/ids-knowledgebase/ids-ram-4/layers-of-the-reference-architecture-model/3-layers-of-the-reference-architecture-model/3_5_0_system_layer/3_5_6_vocabulary_hub
[17] https://github.com/International-Data-Spaces-Association/IDS-VocabularyProvider

Co-funded by the European Union This project has received funding from the Digital Europe Programme under grant agreement No 101123179

10 Dry-run testing

We will conduct a series of systematic dry-run tests to verify the functionality and integration of the EDC (Eclipse Dataspace Connector) dataspace implementation using Sovity components. The test cases focus on validating core subsystems including connector registration, asset management, policy enforcement, contract negotiations, and data transfer mechanisms. These checks are essential to ensure all components are operating as expected within the dataspace ecosystem. The following test cases, documented in the table below (Table 4), provide a structured approach to verify each subsystem's operational status and identify any potential issues.

Table 4 Testing cases

Test Case	**Prerequisites**	**Test Steps**	**Expected Results**
Basic Connector Registration	Sovity Control Plane running - Valid connector configuration	1. Start Connector 2. Check registration status 3. Verify in Sovity Control Plane UI	Connector appears as active in Control Plane dashboard
Asset Registration	Connector running - Test asset prepared	1. Register asset via UI or API 2. Verify asset visibility 3. Check asset metadata	Asset successfully registered and visible in asset index
Policy Definition	Connector running - Asset registered	1. Create usage policy 2. Attach to asset 3. Verify policy association	Policy correctly associated with asset
Contract Definition	Policy defined - Asset registered	1. Create contract definition 2. Link policy and asset 3.Validate contract availability	Contract definition active and queryable
Contract Negotiation	Two connectors running - Valid contract definition	1. Initiate negotiation 2. Monitor negotiation state 3.Verify final agreement	Successful contract negotiation with valid agreement
Data Transfer	Valid contract - Active connectors	1. Request data transfer 2. Monitor transfer progress 3. Verify data integrity	Complete data transfer with integrity check passed
Identity Management	IDS-IDP configured - Valid credentials	1. Authenticate connector 2. Verify token 3. Check token expiry	Successful authentication and valid token received
API Security	Running connector - Test credentials	1. Test unauthorized access 2. Test with valid credentials 3. Verify access logs	All unauthorized attempts blocked, authorized requests successful
Sovity Control Plane Integration	Control Plane deployment - Connector configuration	1. Check connector health status 2. Verify monitoring data 3. Test management functions	Full integration with Control Plane features
Crawler Initial Discovery	Crawler service running - Multiple active connectors	1. Start crawler service 2. Monitor discovery process 3. Verify discovered connectors	All active connectors in network discovered and listed
Crawler Asset Indexing	Discovered connectors - Published assets	1. Trigger asset crawling 2. Monitor indexing progress 3. Verify asset Catalog	Complete index of all available assets across connectors
Crawler Update Detection	Indexed assets - Modified test asset	1. Modify asset in source connector 2. Wait for crawler cycle	Modified asset details reflected in Catalog

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

		3. Check index updates	
Catalog Search Function	Indexed Catalog - Search criteria defined	1. Perform keyword search 2. Test filtering options 3. Verify search results	Accurate search results with proper filtering

Those tests will be carried out promptly, and new tests will be added as new components become available. The project keeps the list of tests together with their status and various comments in a common repository. The list of tests will be updated regularly with new tests and results of tests already carried out.

The dry-run test list of the project can be found at Dry_run_tests.xlsx

11 Future development

The UNDERPIN project aims at enhancing its infrastructure and functionalities through several key future developments, particularly focusing on the expansion of connector capabilities and the development of the semantic layer.

Connector Enhancements:

Current connectors deployed within the UNDERPIN framework are designed for short-term needs but will be customized and extended to accommodate advanced requirements as the project evolves. This includes improving scalability and adaptability to handle increased data volumes and more complex data exchange scenarios. The connectors will also support enhanced policy-based access controls, ensuring that organizations can tailor data sharing practices to meet specific regulatory and operational needs.

Semantic layer integration:

The UNDERPIN Semantic Layer will enable semantic interoperability within the dataspace by providing a central Vocabulary Hub for standard vocabularies and ontologies, accessible via SPARQL. It will include services for document annotation, structured data descriptions, and dataset semantic harmonization through interlinking vocabularies and standards like CSVW and R2RML. Future goals are to enhance the Vocabulary Hub by automating document annotation using machine learning, improving data integration by using Linked Data principles, implementing advanced content analysis for metadata discovery, and optimizing the infrastructure supporting the Semantic Layer for better performance and scalability. These enhancements will create a more dynamic and interoperable environment that will facilitate efficient data management and collaboration among participants.

User Experience Improvements:

Future developments will also focus on refining the user experience by enhancing the interfaces of both the connectors and the Authority Portal. This includes implementing more intuitive navigation, customizable features, and improved management tools that simplify asset and policy configurations for users without extensive technical backgrounds.

12 Concluding remarks

The "D3.1 UNDERPIN infrastructure, mid-term deployment and integration report" encapsulates the progress made in establishing a robust framework for the UNDERPIN project, focusing on secure and efficient data exchange within critical manufacturing industries. Through the deployment of a scalable cloud infrastructure, comprehensive security measures, and advanced connector functionalities, the project lays a solid foundation for future developments aimed at enhancing semantic interoperability and data management capabilities.

As the project moves forward, the integration of further enhancements to the Semantic Layer will facilitate improved logging, data consolidation, and user experience. The Authority Portal's centralized management features will ensure compliance and control over participant access, fostering collaboration across diverse organizations. Overall, UNDERPIN is poised to drive innovation in dynamic asset management through secure data sharing, enabling stakeholders to leverage data more effectively while adhering to regulatory standards and industry best practices. The ongoing commitment to scalability, security, and user-centric design will be pivotal in achieving the project's long-term objectives and fulfilling its vision of a cohesive data ecosystem.

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179

13 Appendix A – EDC Deployment Units Configuration

13.1.1 Reverse Proxy Configuration

To make the deployment work, the connector needs to be exposed to the internet. Connector-to-Connector communication is asynchronous and done with authentication via the DAPS.

- The Sovity EDC Connector is deployed with a reverse proxy merging the following ports:
  - The UI's 8080 port. Henceforth, called the UI.
  - The Backend's 11002 port. Henceforth, called the Management API.
  - The Backend's 11003 port. Henfceforth, called the Protocol API.
- The mapping is as follows:
  - https://[MY_EDC_FQDN]/api/dsp -> edc:11003/api/dsp
  - https://[MY_EDC_FQDN]/api/management -> Auth Proxy -> edc:11002/api/management
  - All other requests -> Auth Proxy -> edc-ui:80
- Regarding TLS/HTTPS:
  - All endpoints are secured by TLS/HTTPS.
  - All endpoints have HTTP to HTTPS redirects.
- Regarding Authentication:
  - The UI and the Management API are secured by an auth proxy.
  - The backend's 11003 port needs to be unsecured. Authentication between connectors is done via the dataspace authority / DAPS and the configured certificates.
- Exposing to the internet:
  - The Protocol API must be reachable via the internet[18].
  - Exposing the UI or the Management Endpoint to the internet requires an intermediate auth proxy, therefore we have restricted the access to the Management Endpoint within our internal network.
- Security:
  - The header size in the proxy has been limited so that only a certain number of API Keys can be tested with one API-request (limited to 64kb).
  - The access rate to the API endpoints has been limited and access is monitored for attacks, i.e. brute force attacks.

13.1.2 EDC UI Configuration

The following config properties have been set for the Sovity EDC UI deployment:

Active Profile

```
EDC_UI_ACTIVE_PROFILE: mds-open-source
```

Management API URL

```
EDC_UI_MANAGEMENT_API_URL: https://[EDC URL]/api/management
```

Management API Key

```
EDC_UI_MANAGEMENT_API_KEY: "ApiKeyDefaultValue"
```

[18] https://github.com/sovity/edc-ce/blob/gitbook/docs/deployment-guide/goals/production/public-endpoints.yaml

Co-funded by the European Union This project has received funding from the Digital Europe Programme under grant agreement No 101123179

Enable config fetching from the backend

```
EDC_UI_CONFIG_URL: "edc-ui-config"
```

13.1.3 EDC Backend Configuration

For the MDS EDC CE Backend deployment the following environment variables were used:

Connector Host Name

```
MY_EDC_FQDN: "my-edc-deployment1.example.com"
```

Participant ID / Connector ID

Must be configured as the value of the "referringConnector" claim in the DAPS for this connector

```
MY_EDC_PARTICIPANT_ID: "MDSL1234XX.C1234XX"
```

Connector Localized Name / Title

```
MY_EDC_TITLE: "EDC Connector"
```

Connector Description Text

```
MY_EDC_DESCRIPTION: "sovity Community Edition EDC Connector"
```

Connector Curator

The company using the EDC Connector, configuring data offers, etc.

```
MY_EDC_CURATOR_URL: "https://example.com"
MY_EDC_CURATOR_NAME: "Example GmbH"
```

Database Connection

```
MY_EDC_JDBC_URL: jdbc:postgresql://postgresql:5432/edc
MY_EDC_JDBC_USER: edc
MY_EDC_JDBC_PASSWORD: edc
```

Management API Key

high entropy recommended when configuring the value (length, complexity, e.g. [a-zA-Z0-9+special chars]{32+ chars})

```
EDC_API_AUTH_KEY: ApiKeyDefaultValue
```

```
# Connector Maintainer

# The company hosting the EDC Connector

MY_EDC_MAINTAINER_URL: "https://sovity.de"

MY_EDC_MAINTAINER_NAME: "sovity GmbH"


# DAPS URL

EDC_OAUTH_TOKEN_URL: 'https://daps.test.mobility-dataspace.eu/token'

EDC_OAUTH_PROVIDER_JWKS_URL:              'https://daps.test.mobility-
dataspace.eu/jwks.json'


# DAPS Credentials

EDC_OAUTH_CLIENT_ID: '_your SKI/AKI_'

EDC_KEYSTORE: '_path to .jks file in container_'

EDC_KEYSTORE_PASSWORD: '_your keystore password_'

EDC_OAUTH_CERTIFICATE_ALIAS: 1

EDC_OAUTH_PRIVATE_KEY_ALIAS: 1


# LoggingHouse Extension settings

EDC_LOGGINGHOUSE_EXTENSION_ENABLED: "true"

EDC_LOGGINGHOUSE_EXTENSION_URL:
https://login.dataspace.underpinproject.eu
```

14 Appendix B – AP Deployment Units Configuration

14.1.1 Reverse Proxy / Ingress

- Authority Portal deployed with TLS/HTTPS.
- The domain under which the Authority Portal is reachable on the internet, referred to as [AP_FQDN].
- Path mapping:
  - Frontend: `https://[AP_FQDN] -> caddy:8080 -> frontend:8080`
  - Backend: `https://[AP_FQDN]/api -> caddy:8080 -> oauth2-proxy:8080 -> caddy:8081 -> backend:8080/api`

14.1.2 Keycloak IAM Deployment

- Keycloak gets the following env variables in the container:

Co-funded by the European Union This project has received funding from the Digital Europe Programme under grant agreement No 101123179

```
# Variables to set privacy policy and legal notice URLs on Keycloak pages
  KEYCLOAK_PRIVACY_POLICY_URL:                https://mobility-dataspace.online/privacy-policy-mds-portal
  KEYCLOAK_LEGAL_NOTICE_URL:                  https://mobility-dataspace.eu/legal-notice
```

- The domain under the Keycloak is reachable on the internet [KC_FQDN] differing from [AP_FQDN].
- The steps to set up the realm were the following
    - sovity theme
        1. Copy sovity-theme[19] directory to {keycloakRoot}/themes/ directory
        2. Import realm.json[20] to create the authority-portal realm
        3. Adjust settings for `oauth2-proxy` client (Clients > `oauth2-proxy` > Settings)
            - `Root URL`: URL of the auth proxy, e.g. `https://authority-portal.example.url`
            - `Home URL`: (Relative) sign in URL of auth proxy, e.g. `/oauth2/sign_in`
            - `Valid Redirect URIs`: (Relative) callback URL of auth proxy, e.g. `/oauth2/callback`
            - `Valid post logout redirect URIs: /*`
        4. Adjust settings for `authority-portal-client` client (Clients > `authority-portal-client` > Settings)
            - `Root URL`: URL of the authority portal, e.g. `https://authority-portal.example.url`
            - `Home URL`: same as `Root URL`
        5. Regenerate client secrets for `oauth2-proxy` and `authority-portal-client` clients
            - Clients > [`client`] > Credentials > Regenerate (Client secret)
        6. Select sovity theme for login & email templates

[19] https://github.com/sovity/authority-portal/tree/gitbook/authority-portal-keycloak/sovity-theme
[20] https://github.com/sovity/authority-portal/blob/gitbook/authority-portal-backend/authority-portal-quarkus/src/main/resources/realm.json

- Select `authority-portal` realm
- Realm settings > Themes > Login theme: Select `sovity-theme`
- Realm settings > Themes > Email theme: Select `sovity-theme`

7. Add email settings (Realm settings > Email)
   - `From` and `Host`

14.1.3 Caddy

The Caddyfile [21] needs to be mounted to `/etc/caddy/Caddyfile` in the Caddy container.

The Caddy gets the following env variables to use in the container:

```
BACKEND_UPSTREAM_HOST: backend

FRONTEND_UPSTREAM_HOST: frontend

AUTH_PROXY_UPSTREAM_HOST: auth-proxy
```

14.1.4 OAuth2 Proxy

- The Authority Portal is deployed with an OAuth2 Proxy in front of the Portal Backend.
- The OAuth2 Proxy is configured to use the Keycloak (IAM) as OAuth2 Provider.
- The contents from resources[22] are copied to a directory the OAuth2 proxy can access (`CUSTOM_TEMPLATES_DIR`)

```
OAUTH2_PROXY_PROVIDER: keycloak-oidc

OAUTH2_PROXY_PROVIDER_DISPLAY_NAME: Keycloak

OAUTH2_PROXY_OIDC_ISSUER_URL: https://[KC_FQDN]/realms/[KC_REALM]

OAUTH2_PROXY_COOKIE_SECRET: [COOKIE_SECRET] # (32-bit base64 encoded secret)

OAUTH2_PROXY_COOKIE_REFRESH: 30s # Access Token Lifespan - 30 seconds

OAUTH2_PROXY_COOKIE_EXPIRE: 30m # Client Session Idle / SSO Session Idle

OAUTH2_PROXY_CLIENT_ID: oauth2-proxy

OAUTH2_PROXY_CLIENT_SECRET: [OA2_CLIENT_SECRET]

OAUTH2_PROXY_EMAIL_DOMAINS: "*"

OAUTH2_PROXY_UPSTREAMS: http://caddy:8081/

OAUTH2_PROXY_API_ROUTES: "^/api/"
```

[21] https://github.com/sovity/authority-portal/tree/gitbook/authority-portal-oauth2-proxy/resources
[22] https://github.com/sovity/authority-portal/blob/gitbook/authority-portal-oauth2-proxy/resources

```
OAUTH2_PROXY_SKIP_AUTH_ROUTES:
"^(/oauth2|/api/registration|/api/config)"

OAUTH2_PROXY_HTTP_ADDRESS: 0.0.0.0:8080

OAUTH2_PROXY_PASS_ACCESS_TOKEN: "true"

OAUTH2_PROXY_SKIP_PROVIDER_BUTTON: "true"

OAUTH2_PROXY_SHOW_DEBUG_ON_ERROR: "true"

OAUTH2_PROXY_REDIRECT_URL: https://[AP_FQDN]/oauth2/callback

OAUTH2_PROXY_SCOPE: openid profile

OAUTH2_PROXY_WHITELIST_DOMAINS: [KC_FQDN]

OAUTH2_PROXY_CUSTOM_TEMPLATES_DIR: [CUSTOM_TEMPLATES_DIR]
```

14.1.5 Keycloak DAPS Client Creation

The Authority Portal requires a client to register new connector certificates. This client has the following settings:

- Section `Authentication flow` (Tab Settings)
  - Everything disabled
  - `Service accounts roles` enabled
- Tab `Client scopes`
  - Added client scope `roles` as a default scope to the client
- Service account roles (Tab `Service Account Roles`)
  - `realm-management > manage-clients` enabled
  - `realm-management > create-client` enabled
  - `realm-management > view-clients` enabled
  - `realm-management > query-clients` enabled

14.1.6 Authority Portal Backend

- Image: `ghcr.io/sovity/authority-portal-backend`
- Environment variables set according to the following documentation

Postgres DB Connection

```
quarkus.datasource.jdbc.url:                "jdbc:postgresql://portal-
db/authority_portal"

quarkus.datasource.username: "postgres"

quarkus.datasource.password: "postgres"
```

Keycloak Client for User IAM

Base URL of the OIDC server (Keycloak). Must contain the `'/realms/{realm}' part of the URL`

```
quarkus.oidc.auth-server-url: "https://[KC_FQDN]/realms/[KC_REALM]"
```

```
# Keycloak Admin Client

# Keycloak Admin Client: Server URL
quarkus.keycloak.admin-client.server-url: "https://[KC_FQDN]"

# Keycloak Admin Client: Realm
quarkus.keycloak.admin-client.realm: "[KC_REALM]"

# Keycloak Admin Client: Client ID
quarkus.keycloak.admin-client.client-id: "authority-portal-client"

# Keycloak Admin Client: Client secret
quarkus.keycloak.admin-client.client-secret: "[AP_CLIENT_SECRET]"

# Keycloak Admin Client: Grant type
quarkus.keycloak.admin-client.grant-type: "CLIENT_CREDENTIALS"


# Log level for backend logging (ERROR, INFO, DEBUG, etc). Docs: https://quarkus.io/guides/logging
quarkus.log.level: "INFO"


# CaaS Portal

# CaaS Portal: URL
authority-portal.caas.sovity.url: "https://[CAAS_PORTAL_FQDN]"

# CaaS Portal: OAuth2 Auth server URL
quarkus.oidc-client.sovity.auth-server-url:
"https://[CAAS_KC_FQDN]/realms/[CAAS_REALM]"

# CaaS Portal: OAuth2 Client ID
quarkus.oidc-client.sovity.client-id: "[CAAS_CLIENT_ID]"

# CaaS Portal: OAuth2 Client Secret
quarkus.oidc-client.sovity.credentials.secret:
"[CAAS_CLIENT_SECRET]"

# Amount of free sovity CaaS per participant
authority-portal.caas.sovity.limit-per-organization: "1"

# Enables the connection (set to false if you don't have the data to  fill out the variables above)
quarkus.oidc-client.sovity.client-enabled: true
```

```
# Must equal the root URL/home URl from the Keycloak configuration -  see above)
authority-portal.base-url: "https://[AP_FQDN]"

# API key to protect config endpoints, like /api/config/log-level
authority-portal.config.api-key: "[AP_CONFIG_API_KEY]"

# Invitation link expiration time in seconds. (Must equal the value  in  Keycloak configuration)
authority-portal.invitation.expiration: "43200"

# Uptime Kuma
# Uptime Kuma URL (/metrics endpoint must be available)
authority-portal.kuma.metrics-url: "https://[UPTIME_KUMA_FQDN]"
# Uptime Kuma API key
authority-portal.kuma.api-key: "[UPTIME_KUMA_API_KEY]"

# Environment Configuration
# - Each Authority Portal can be configured with multiple environments, e.g. test, staging, prod, etc.
# - Following is an example configuration of the "test" environment.
# - Please Note, that the environment "test" is mandatory

# Environment Configuration: Metadata
# Title of the deployment environment configuration
authority-portal.deployment.environments.test.title: "Test"
# Order of environments, from 0 (default) to n (least important)
authority-portal.deployment.environments.test.position: "0"

# Environment Data Catalog Settings
# Time after which offline data offers are hidden from the Data Catalog
authority-portal.deployment.environments.test.data-catalog.hide-offline-data-offers-after: "15m"
# Default page size for the Data Catalog
```

```
authority-portal.deployment.environments.test.data-catalog.catalog-page-page-size: "10"
```

Kuma name for the catalog crawler

```
authority-portal.deployment.environments.test.data-catalog.kuma-name: broker
```

Environment Connector-Dataspace association: Allows certain connectors to be associated as partnered dataspaces

Required: Default Dataspace name

```
authority-portal.deployment.environments.test.data-catalog.dataspace-names.default: "Underpin"
```

Optional: Additional connectors to be given a dataspace name

```
authority-portal.deployment.environments.test.data-catalog.dataspace-names.connector-ids."MDSL1234XX.C1234XX": "Mobilithek"
```

Environment DAPS

Env: DAPS URL

```
authority-portal.deployment.environments.test.daps.url: "https://[KC_DAPS_FQDN]"
```

Env: DAPS realm name

```
authority-portal.deployment.environments.test.daps.realm-name: "DAPS"
```

Env: DAPS Admin Client Client ID

```
authority-portal.deployment.environments.test.daps.client-id: "authority-portal"
```

Env: DAPS Admin Client Client Secret

```
authority-portal.deployment.environments.test.daps.client-secret: "[DAPS_CLIENT_SECRET]"
```

Env: DAPS Kuma name

```
authority-portal.deployment.environments.test.daps.kuma-name: "[DAPS_KUMA_NAME]"
```

Environment Logging House

Env: Logging House URL

```
authority-portal.deployment.environments.test.logging-house.url:
"https://[LOGGING_HOUSE_FQDN]"

# Env: Logging House Kuma name

authority-portal.deployment.environments.test.logging-house.kuma-
name: "[LOGGING_HOUSE_KUMA_NAME]"
```

Adjusting the log level at runtime

The log level can be changed during runtime via a request to the `/api/config/log-level` endpoint. The API key is required for this. Example:

```
curl -X PUT 'https://authority-portal.example.com/api/config/log-
level?level=DEBUG' --header 'x-api-key:
uYtR_wNsvXU4EbV9GioACnj!NHML_HRX'
```

14.1.7 Authority Portal Frontend

Image: `ghcr.io/sovity/authority-portal-frontend`

Set environment variables according to the following table (mandatory)

```
AUTHORITY_PORTAL_FRONTEND_BACKEND_URL: https://[AP_FQDN] # Authority
Portal URL

AUTHORITY_PORTAL_FRONTEND_LOGIN_URL:
https://[AP_FQDN]/oauth2/start?rd=https%3A%2F%2F[AP_FQDN] # Auth
Proxy: Login URL (with redirect to the Authority Portal)

# Following is the URL to signal the Auth Proxy to log out the user.

# Example:
https://[AP_FQDN]/oauth2/sign_out?rd=https%3A%2F%2F[KC_FQDN]%2Frealm
s%2F[KC_REALM]l%2Fprotocol%2Fopenid-
connect%2Flogout%3Fclient_id%3Doauth2-
proxy%26post_logout_redirect_uri%3Dhttps%253A%252F%252F[AP_FQDN]

AUTHORITY_PORTAL_FRONTEND_LOGOUT_URL: (...) # Auth Proxy: Logout URL

AUTHORITY_PORTAL_FRONTEND_INVALIDATE_SESSION_COOKIES_URL:
https://[AP_FQDN]/oauth2/sign_out # Auth Proxy: URL to invalidate
sessions cookies

AUTHORITY_PORTAL_FRONTEND_IFRAME_URL: https://news.
underpinproject.eu # iFrame URL for the "Home" page if it's used

AUTHORITY_PORTAL_FRONTEND_LEGAL_NOTICE_URL:
https://yourdataspace.com/legal-notice # Legal Notice URL

AUTHORITY_PORTAL_FRONTEND_PRIVACY_POLICY_URL:
https://yourdataspace.com/privacy-policy # Privacy policy URL

AUTHORITY_PORTAL_FRONTEND_SUPPORT_URL: https://support.
underpinproject.eu # Support page URL
```

```
AUTHORITY_PORTAL_FRONTEND_ACTIVE_PROFILE: sovity-open-source # UI Branding profile (sovity-open-source)

AUTHORITY_PORTAL_FRONTEND_DATASPACE_SHORT_NAME: ExDS # Short Dataspace name, used in some explanatory texts

AUTHORITY_PORTAL_FRONTEND_PORTAL_DISPLAY_NAME: "Authority Portal" # Portal name displayed in various texts
```

Co-funded by the European Union
This project has received funding from the Digital Europe Programme under grant agreement No 101123179