Co-funded by the European Union

DEPLOYTOUR
European Tourism Data Space

Call for proposals	DIGITAL-2023-CLOUD-DATA-AI-05	**Type of action**	DIGITAL-SIMPLE
Grant Agreement No.	101173388	Start date	1 October 2024
Project duration	36 months	End date	30 September 2027

Contact: projects@anysolution.eu

Website: www.deploytour.eu

Project consortium – Coordinator: ANYSOLUTION			
POLITENICO DE MILANO	BEN - POLIMI	NTT DATA SPAIN	BEN - NTTDES
AMADEUS DATA PROCESSING GmbH	BEN - ADP	AMADEUS GERMANY GmbH	AE - AMADEUS GERMANY
EONA-X	BEN – EONA-X	ITALIAN MINISTRY OF TOURISM	BEN - MITUR
FUNDACIÓN TECNALIA RESEARCH & INNOVATION	BEN - TECNALIA	NECSTOUR	BEN - NECSTOUR
CITY DESTINATIONS ALLIANCE	BEN - CityDNA	INTELLERA	BEN - INTELLERA
ARCTUR	BEN - ARCTUR	INSTITUTO TECNOLÓGICO DE INFORMÁTICA	BEN - ITI
GMV SOLUCIONES GLOBALES INTERNET	BEN - GMV	AVORIS CENTRAL DIVISION	BEN - AVORIS
AUSTRIA TOURISM (OSTERREICH WERBUNG)	BEN – AUSTRIA TOURISM	EUROPEANA	BEN - EF
TURISMO ANDALUCIA	BEN – EPGTDA SA	AMADEUS SAS	BEN – AMADEUS SAS
PLEXUS TECH	BEN – PLEXUS TECH	TECNOLOGÍAS PLEXUS SL	AE - PLEXUS
FRAUNHOFER	BEN - FRAUNHOFER	HIBERUS TECNOLOGIAS DIFERENCIALES SL	BEN - HIBERUSTECH
HIBERUS IT DEVELOP	AE - HIBIT	UNPARALLEL INNOVATION	BEN - UNPARALLEL
PLEIADES CLUSTER	BEN - PLEIAD	UNI SYSTEMS SYSTIMATA	AE - UNIS
THE DATA APPEAL COMPANY	BEN – DATA APPEAL CO	INDUSTRY INNOVATION CLUSTER SLOVAKIA	BEN - ICC
TOURISM BOHINJ SLOVENIA	BEN – TURIZEM BOHIN	LAPLAND UNIVERSITY OF APPLIED SCIENCES	BEN – LAPLAND UAS
DISSET CONSULTORES	BEN - DISSET	UNIVERSITY ILLES BALEARS	BEN - UIB
ADQUIVER	BEN - ADQUIVER	AE-ADQUIVER DATA & ADVANCED ANALYTICS, SOCIEDAD LIMITADA	ADDATA
TRENITALIA	BEN - TRIT	TOURISM PORTUGAL	BEN – TURISMO PT
UNIVERSITY NOVA LISBOA	BEN - UNL	LIBELIUM LAB	BEN – LIBELIUM
MODUL UNIVERSITY VIENNA GMBH	AP - MODUL	YPOURGEIO TOURISMOU	AP - MINTUR
AGENCIA D ESTRATEGIA TURISTICA DE LES ILLES BALEARS	AP - AETIB	STICHTING BREDA UNIVERSITY OF APPLIED SCIENCES	BEN - BUAS
FIWARE FOUNDATION EV	AP - FIWARE		

Co-funded by the European Union

D2.2 ETDS Prototype

Document Identification			
Status	Final	**Due Date**	31/01/2026
Version	1.0	**Submission Date**	31/01/2026

Related WP	WP2	**Document Reference**	D2.2
Related Deliverable(s)	D2.1, D2.4, D2.5	**Dissemination Level (*)**	PU
Lead Partner	NTTDES	**Lead Author**	
Contributors	ITI AMADEUS NTTDES PLEXUS EONA-X LIBELIUM FIWARE ANYSOLUTION	**Reviewers**	ITI

Co-funded by the European Union

Document Information

List of Contributors	
Name	Partner
Jordi Arjona	ITI
Anastasia Koufaki	Amadeus
Delia Moreno	NTTDES
Laura Sande	PLEXUS
Alex Sandro Antonio de Sousa	PLEXUS
Dr. Dolores Ordóñez	ANYSOLUTION
Tayrne Butler	ANYSOLUTION
Antonio Sánchez	ANYSOLUTION
Peio Oiz Arruti	ANYSOLUTION
Jonathan Huffstutler	EONA-X
Juan F. Inglés	LIBELIUM
Juanjo Hierro	FIWARE

Document History			
Version	Date	Change editors	Changes
0.1	02/12/2025	Delia Moreno	Initial version
0.2	09/01/2026	Jordi Arjona	First review
0.3	21/01/2026	Anastasia Koufaki	Second review
0.4	27/01/2026	Anastasia Koufaki	Third review
1.0	28/01/2026	Peio Oiz Arruti	Final Review

Quality Control		
Role	Who (Partner short name)	Approval Date
Project Coordinator	ANYSOL	31/01/2026

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
EIF	European Interoperability Framework
ETDS	European Tourism Data Space (the ecosystem of Tourism data spaces)
MVDS	Minimum Viable Data Space
EDC	Eclipse Dataspace Components
ODRL	Open Digital Rights Language
UI	User Interface
IDSA	International Data Spaces Association
VCs	Verifiable Credentials
DSP	Dataspace Protocol
SDs	Self-Descriptions
DSSC	Data Spaces Support Centre
DIDs	Decentralized Identifiers
DCP	Decentralized Claim Protocol
DOME	Distributed Open Marketplace for Europe
DSIF	Data Space Interoperability Framework
FDC	FIWARE Dataspace Components
ITB	Interoperability Test Bed (ITB)

Glossary

Term	Description
Business model	A description of the way an organisation creates, delivers, and captures value. Such a description typically includes for whom value is created (customer) and what the value proposition is. Normally, a tool called the Business Model Canvas is used to describe or design a business model, but alternatives better suited to specific situations, such as data spaces, are available.
Data Model	A structured representation of data elements and relationships used to facilitate semantic interoperability within and across domains, encompassing vocabularies, ontologies, application profiles and schema specifications for annotating and describing data sets and services. These abstraction levels need not be hierarchical; they can exist independently.
Data Product	Data sharing units, data and metadata packaging, and any associated license terms. Explanatory Texts: • We (the DSSC) borrow[s] the definition from the CEN Workshop Agreement Trusted Data Transactions. • The definition of data products is still evolving in the data space community. • The data product may include, for example, the data product's allowed purposes of use, quality and other requirements the data product fulfils, access and control rights, pricing and billing information, etc.
Data Product Offering	An offering, in a general sense, refers to data, services, or a combination of both that a data provider offers to data recipients, and includes attributes such as description, provider, creator, pricing, license, data format, current version, previous version, and access rights.
Data Service	A collection of operations that provides access to one or more datasets or data processing functions. For example, data selection, extraction, and data delivery.
Dataset	A collection of data published or curated by a single agent or identifiable community.
Data Source	System or entity that generates information and provides data and metadata, but they are not yet integrated into the governance of the dataspace.
Data Space	An interoperable framework, based on common governance principles, standards, practices and enabling services, that enables trusted data transactions between participants. Explanatory Texts: ● Note for users of V0.5 and V1.0 of this blueprint: we (the DSSC) have[as] adopted this new definition from CEN Workshop Agreement Trusted Data Transactions, to converge with ongoing standardisation efforts. Please note that further evolution might occur in future versions. For reference, the previous definition was: “Distributed system defined by a governance framework that enables secure and trustworthy data transactions between

	participants while supporting trust and data sovereignty. One or more infrastructures implement a data space and enable one or more use cases." ● Note: some parties write dataspace in a single word. We (the DSSC) prefer[s] data space in two words and consider that both terms mean the same.
Data Space Agreement	A contract that states the rights and duties (obligations) of parties that have committed to (signed) it in the context of a particular data space. These rights and duties pertain to the data space and/or other such parties.
Data Space Building Block	A description of related functionalities and/or capabilities that can be realised and combined with other building blocks to achieve the overall functionality of a data space. Explanatory Texts: ● In the data space blueprint, the building blocks are divided into organisational and business building blocks and technical building blocks. ● In many cases, the functionalities are implemented by Services.
Data Space Component	A specification for a software or other artefact that realises one service or a set of services that fulfil functionalities described by one or more building blocks. Explanatory Text: For technical components, this would typically be software; for business components, it could consist of processes, templates, or other artefacts.
Data Space Component Architecture	An overview of all the data space components and their interactions, providing a high-level structure of how these components are organised and interact within data spaces.
Data Space Connector	A technical component that is run by (or on behalf of) a participant and that provides participant agent services, with similar components run by (or on behalf of) other participants. Explanatory Text: A connector can provide functionality beyond strictly connectivity. The connector can offer technical modules that implement data interoperability functions, authentication, interfacing with trust services and authorisation, data product self-description, contract negotiation, etc. We use "participant agent services" as the broader term to define these services.
Data Space Pilot	A planned and resourced implementation of one or more 'use cases' within the context of a data space initiative. A data space pilot aims to validate the approach for a full data space deployment and showcase the benefits of participating in the data space.
Use Case	A specific setting in which two or more participants use a data space to create value (business, societal or environmental) from data sharing. Explanatory Texts: ● By definition, a data space use case is operational. When referring to a planned or envisioned setting that is not yet operational, we can use the term use case scenario. ● A use case scenario is a potential use case envisaged to solve societal, environmental or business challenges and create value. The same use case scenario, or variations of it, can be implemented as a use case multiple times in one or more data spaces.

Data Spaces Blueprint	A consistent, coherent and comprehensive set of guidelines to support the implementation, deployment and maintenance of data spaces. Explanatory Text: The blueprint contains the conceptual model of data space, the data space building blocks, and the recommended selection of standards, specifications, and reference implementations identified in the data spaces technology landscape.
DSSC Asset	A sustainable open resource that is developed and governed by the Data Spaces Support Centre (DSSC). The assets can be used to create, deploy, and operationalise data spaces, and to enable knowledge sharing around them. The DSSC also develops and executes strategies to ensure continuity of the main assets beyond project funding.
End User Product	The data product offering value for the end users of the dataspace 'use cases', i.e., business apps, training models, etc.
Resource	A dataset, a data service or any other resource that a metadata record may describe in a catalog.

Executive Summary

DEPLOYTOUR is a three-year project starting in October 2024 that aims to develop and deploy the European Tourism Data Space (ETDS). It is preceded by two preparatory actions, DATES and DSFT, whose recommendations this deliverable has adopted.

In this context, Deliverable D2.2 presents the detailed design and implementation strategy for the ETDS prototype, which will be progressively implemented throughout the DEPLOYTOUR project. As a first step in this process, the deliverable defines a Minimum Viable Data Space (MVDS) a functional implementation focused on core components and technical validation. While the MVDS provides the initial deployment baseline, it does not represent the full ETDS prototype. Instead, it serves as a foundational milestone toward the broader, federated prototype that will evolve over the course of the project. Drawing from the preparatory work conducted in D2.1 (Interoperability & Data Sharing) and D2.4/D2.5 (ETDS Architecture), this deliverable moves the project into its technical execution phase.

The document provides a comprehensive overview of the architectural approach, identifying the main building blocks and interaction flows that structure the ETDS. It outlines a modular ETDS stack based on Eclipse Dataspace Components and FIWARE Dataspace Components, which can be combined. Future phases will address interoperability between different connectors. In parallel, the potential alignment with other European interoperability frameworks, such as SIMPL, will be explored to strengthen cross-domain integration and reusability.

Finally, the deliverable details the deployment and integration setup, including the use of Kubernetes-based infrastructure, conformance testing via the Interoperability Test Bed (ITB), and alignment with key European standards and frameworks such as Gaia-X, IDSA, and the Data Governance Act. The selected technology stack relies on consolidated open-source components to ensure openness, replicability and long-term sustainability.

1 Introduction

The European tourism sector, dynamic and fragmented, increasingly demands secure and interoperable data sharing to support innovation, sustainability, and competitiveness. The DEPLOYTOUR project responds to this need by establishing the foundations for an ETDS, in line with the European Strategy for Data and the relevant legislative and technical frameworks.

This document, Deliverable D2.2 – ETDS Prototype, represents a key transitional deliverable within the DEPLOYTOUR initiative. It defines the design and implementation strategy for the ETDS prototype, which will be progressively developed over the course of DEPLOYTOUR. As a first step in this process, the deliverable introduces the Minimum Viable Data Space (MVDS), a focused implementation designed to validate the core architectural components and technical flows in a controlled environment. While the MVDS provides the foundation for initial deployment and testing, it does not reflect the full extent of the ETDS prototype. Rather, it acts as a stepping stone toward the broader, federated infrastructure that the project seeks to realise.

To support flexibility and promote adoption across diverse technological environments, the project includes two complementary implementations of the MVDS: one based on the EDC and another using the FDC. Although these versions are not interoperable at this stage, interoperability will be progressively addressed in subsequent phases of the project.

1.1 Objectives

The main objectives of this deliverable are: · Establish the design principles guiding the development of the ETDS prototype and ensure their practical application in building a secure, interoperable, and modular foundation aligned with relevant European frameworks.

- Define the essential components and interaction flows of the ETDS prototype, providing a shared architectural basis for secure, decentralised, and interoperable data sharing among tourism stakeholders.
- Set up the technical environment for deploying the MVDS prototype using open, scalable and standards-compliant tools, ensuring support for integration, testing and alignment with European interoperability frameworks.

Therefore, this document establishes the conceptual baseline and shared understanding necessary to develop the first functional version of the ETDS prototype and to initiate its progressive evolution.

1.2 Purpose and scope of the Document

The purpose of this deliverable is to describe the conceptualisation and preparation for the development of the first version of the ETDS prototype. It serves as a transitional step between the architecture definition and implementation, translating the reference design into a concrete plan for an MVDS that will demonstrate the technical feasibility of the European Tourism Data Space.

The scope of D2.2 ETDS Prototype includes: · The identification of the technical requirements to be addressed in this first release.

- The definition of the prototype's conceptual architecture, core components, and interaction flows.
- The selection of development tools, technologies, and processes that will guide implementation. · The initial setup of the technical environment to host and integrate the selected components.

While this deliverable focuses primarily on the EDC-based implementation, it acknowledges the parallel development of an FDC-based variant, in line with DEPLOYTOUR's dual-stack strategy. Interoperability between the two implementations will be addressed in subsequent stages.

This document does not present a working prototype; instead, it defines the design, scope, and implementation strategy of the MVDS that will be developed throughout the upcoming phases of the project.

1.3 Relation to other deliverables

Deliverable D2.2 connects directly to the previous and future results of the project as follows:

- D2.1 Interoperability & Data Sharing established the conceptual basis for interoperability within the European Tourism Data Space. It gathered requirements from pilot initiatives, identified data-sharing mechanisms and standards and outlined the semantic and technical frameworks to be adopted. These insights provide the essential groundwork for defining the prototype's components.
- D2.4/D2.5 ETDS Architecture translated the interoperability framework into a high-level architectural model for the ETDS, detailing the MVDS and its building blocks. D2.2 now takes this one step further by operationalising the architecture, specifying which components will be developed in the first version of the prototype (MVDS) and how they will interact in practice.

In this sense, D2.2 serves as a bridge deliverable between the conceptual and architectural design phases. This ensures technical and strategic continuity across the different stages of the DEPLOYTOUR project and consolidates a shared understanding of what the initial ETDS prototype will encompass.

1.4 Structure of the document

The rest of the document is organised as follows:

- Chapter 2 introduces the ETDS prototype within the broader DEPLOYTOUR framework, outlining the key design principles and conceptual architecture that shape its development.
- Chapter 3 describes the main components of the prototype and how they interact to enable secure and decentralised data sharing. It also details the key interaction flows, as well as the tools, technologies, and methodologies selected to build the first version of the system.
- Chapter 4 addresses the integration and deployment of the prototype. It presents the integration strategy alongside other relevant frameworks. It provides an overview of the technical environment, runtime configuration, and infrastructure supporting testing and operations during the pilot phase.
- Chapter 5 summarises conclusions and outlines the next steps toward the implementation of MVDS.

The Annex provides supporting materials, references, and complementary technical details that contribute to a better understanding of the prototype's design and its future development.

2 Overview of the ETDS Prototype

The ETDS prototype begins with a Minimum Viable Data Space (MVDS) that implements the reference architecture in a controlled pilot environment, prioritising integration testing over scale or high availability. Its core connector stack is based on the Eclipse Dataspace Components (EDC), with the FIWARE Dataspace Components (FDC) regarded as a possible complementary variant rather than a fixed element of this initial build. The MVDS is deliberately confined to the essential building blocks that ensure data sovereignty, trust, and interoperability in sharing. At the same time, advanced capabilities (e.g., large-scale federation or complex semantics) are intentionally excluded from this first phase to maintain a solid, scalable foundation. Let's keep in mind that the ETDS Prototype refers to a federation of data spaces and data-sharing initiatives in the tourism sector.

As part of a dual implementation strategy, the ETDS prototype is being developed in two parallel variants: one based on EDC and another based on FDC. Both follow the same architectural principles and adhere to European interoperability frameworks developed independently. While interoperability between these two implementations is not ensured in this initial version, future phases of the project will work towards compatibility and convergence across both stacks, aiming to enable seamless collaboration between them.

In addition, the project will explore potential alignment with other European initiatives, such as SIMPL, as part of its ongoing effort to reinforce interoperability and reuse across data spaces and ensure consistency with the broader European data ecosystem.

2.1 General design principles

The European Tourism Data Space (ETDS) architecture is founded on a set of generic design principles that ensure the system is interoperable, trustworthy, and aligned with European standards. These principles draw heavily from established frameworks – notably the International Data Spaces Association (IDSA) Reference Architecture Model (IDS-RAM) and the Gaia-X Trust Framework – and leverage the technical capabilities of the Eclipse Dataspace Components (EDC) and FIWARE Dataspace Components (FDC). In particular, the Eclipse Dataspace Components (EDC) serve as the primary foundation providing core building blocks (connectors, catalogs, identity services, etc.). At the same time, the FIWARE Dataspace Components (FDC) are being considered as a complementary framework to ensure openness and prevent technology lock-in. These design principles additionally respect EU regulations (such as the Data Act and Data Governance Act), the EU Digital Identity Framework and Reference Architecture, and the European Interoperability Framework (EIF), ensuring that the prototype is not only technically robust but also legally and organisationally compliant with European requirements.

At a high level, the IDSA reference architecture model and related initiatives (e.g., DSSC Blueprint principles and, recently, the Data Spaces Interoperability Framework – DSIF use cases) emphasise an open, federated architecture for cross-sector data sharing that preserves data sovereignty and trust while maximising interoperability. Similarly, Gaia-X provides guidelines for self-sovereign identity, security, and compliance in federated data ecosystems. Building on these, the ETDS's generic design principles can be summarised under several key themes:

- Interoperability and Use of Open Standards. Use of common protocols and vocabularies (IDSA's Dataspace Protocol -DSP, W3C/DCAT, ODRL, etc.) to enable seamless data sharing across systems.
- Data Sovereignty, Trust and Security, Participants retain control over their data (e.g., via access and/or use control ODRL-based policies) and identities through self-sovereign identity technologies - Distributed Identifiers (DID), Verifiable Credentials

(VCs) standard protocols for issuance and exchange of VCs - and trusted governance frameworks (Gaia-X compliance).

- Federated and Decentralized Architecture. Avoid central data silos: each participant runs its own connector; minimal central services are used only as needed, aligning with "decentralisation by default".
- Modularity and Extensibility of Components. Architecture composed of modular building blocks (e.g., connector, catalog, identity hub, etc.) with well-defined interfaces. Components can be added or replaced via EDC and/or FDC extensions to adapt to different needs.
- Security, Resilience, and "Design for Failure". Security-by-design with strong authentication/authorisation (OID4VC/DCP based on DIDs and VCs, OAuth2), and the enforcement of authorisation policies (ODRL-based policies), together with continuous monitoring to ensure data is accessed and used in compliance with established policies and contracts. The system is cloud-agnostic and resilient: components are decoupled to limit the impact of failures, with failover mechanisms, retry/circuit breakers, and observability to ensure robust operations.

These principles collectively ensure that the ETDS prototype will facilitate voluntary, secure, and efficient data sharing among tourism stakeholders, in line with European data space objectives. In the following subsections, each principle is discussed in detail, including how it maps to specific EDC and FDC standards or components.

2.1.1 Interoperability and Use of Open Standards

Interoperability is the cornerstone of any data space: it allows different organisations, systems, or actors to exchange data in a coordinated way, ensuring aspects such as reliability, trust, compliance, common semantics, and technical considerations, among others. To facilitate this interoperability, the ETDS adopts common open standards and protocols to maximise compatibility both within the tourism sector and with other data spaces (e.g. mobility or culture). Key aspects include:

- Dataspace Protocol (DSP): The ETDS connector stack will support the IDSA's Dataspace Protocol – a standardised set of message types and API bindings for catalog querying, publishing data product offerings, contract negotiation, and transfer process control at the control plane. By using DSP (formerly IDS Protocol), any connector in ETDS can interoperate with connectors in other compliant data spaces that rely on DSP. EDC-based connectors, therefore, the ETDS Connector is DSP-compliant starting with version 0.7.1, facilitating technical interoperability with other DSP-compliant connectors.
- Standardised Vocabularies for Metadata: ETDS will use the W3C DCAT (Data Catalog) vocabulary version 3 to describe data assets and services DCAT provides a standardised structure for publishing metadata such as descriptions, formats, publishers, and access conditions, enabling consistent cataloguing and discovery of data offerings. This allows easy discovery and integration of data from different providers.
- Domain-Specific Vocabularies: The ETDS will produce domain-specific vocabularies to facilitate the indexation and discovery of assets in the data space by providing common definitions of key concepts within the tourism sector. These controlled vocabularies will harmonise the meaning of domain terms used in the metadata, reducing ambiguity and improving data reuse across organisations.
- Standardised Policy Language: For expressing access/usage agreements and restrictions, ETDS uses the W3C ODRL (Open Digital Rights Language). ODRL is a widely adopted machine-readable language for access/usage policies. The EDC's Policy Engine natively supports the enforcement of ODRL policies during contract

negotiation—contract offers in ETDS are defined with ODRL policies that specify usage terms and eligibility criteria. This means consumers and providers in the ETDS interpret policies (e.g. “data can only be used for nonprofit purposes” or “not outside the EU”) consistently. The reliance on ODRL also aids interoperability with other platforms that use the same policy language. On the other hand, the ODRL-OPA component from FDC can be used with APISIX to enforce policies during data-plane exchanges.

- APIs and Data Models: Wherever possible, ETDS utilises open API specifications and data models. For example, when sharing real-time data via APIs, REST and JSON/JSON-LD standards are used to ensure broad compatibility. Common semantic models might be adopted for key concepts (e.g., destinations, events) to facilitate cross-organisational understanding. In the future, a Vocabulary Hub component could host shared ontologies to ensure semantic interoperability across the data space. Synergies with the Smart Data Models initiative endorsed by FIWARE and OASC (Open and Agile Smart Cities) will be explored.

By adhering to these standards, the ETDS ensures that different software implementations can work together. The future ETDS connectors essentially behave as “IDSA-compliant” connectors and align with technology convergence initiatives such as DSIF (Data Space Interoperability Network); therefore, they will be able to integrate into a broader ecosystem of European data spaces. This interoperability is critical given tourism's intersections with other domains (transport, culture, etc.). For instance, a dataset published in the Mobility Data Space using DCAT and DSP could be directly discoverable and usable by a tourism connector in ETDS, and vice versa, thanks to the shared standards.

Mapping to EDC and FDC: The Eclipse Dataspace Components (EDC) framework serves as the core foundation for the ETDS connector due to its strong compliance with standard protocols (e.g., the Dataspace Protocol) and alignment with the IDS Reference Model. In parallel, we are evaluating the FIWARE Dataspace Components (FDC) as a complementary solution — FDC offers compatibility with initiatives like DOME (Distributed Open Marketplace for Europe) and ensures that the identities of users within consumer organisations can be managed in a decentralised manner. They may either directly access data services/applications or support finer-grained access control during the data-sharing process. That could be incorporated at a later stage. As standards evolve (e.g., IDS protocol updates or new EU interoperability specifications), EDC and FDC updates will incorporate those changes into the ETDS. The open standards approach also reflects the EIF (European Interoperability Framework) principle of openness, promoting the use of open data formats and interfaces.

2.1.2 Data Sovereignty, Trust and Security

A foundational goal of European data spaces is to enable data sharing while guaranteeing trust among the participants and each participant’s control over their data – often termed data sovereignty. Therefore, trust and sovereignty are cornerstones of data spaces. In the ETDS, trust and sovereignty are maintained through careful identity and access management built on self-sovereign identity principles and robust security measures. Data sovereignty is preserved by enabling users to define both use and access control policies based on ODRL, which the policy engines of data space connectors based on EDC and FDC can interpret. Below, key concepts related to self-sovereign identity and data sovereignty, as well as those related to trust and security, are explained in more detail.

- Decentralised Self-Sovereign Identity (SSI): ETDS participants (organisations, data providers/consumers) manage their own identities using decentralised identifiers. Specifically, the ETDS uses the W3C Decentralised Identifier (DID) standard for participant IDs and Verifiable Credentials (VCs) for certifications/attributes. Instead of a central authority owning all identities, each participant has a DID (e.g., represented

as a DID document on a web-accessible registry) and holds VCs issued by trusted issuers. For example, a tourism agency in ETDS might have a VC stating that it is a certified ETDS member, issued by the ETDS governing body. The EDC and FDC Identity & Trust frameworks support this: both rely on a DID Registry component to resolve DIDs to DID documents, and on an Identity Hub (decentralised wallet) where each participant stores its VCs or EU Digital Identity-compliant wallets for end users.
    - This means when a consumer and a provider interact, they exchange DIDs and VCs rather than just usernames/passwords. A receiving connector can validate a caller's credentials without querying a central identity provider, thereby enhancing decentralisation. Each participant remains in control of its own credentials, and trust is established through a network of trusted VC issuers (trust anchors) accepted in the tourism data space. This SSI approach aligns with the Gaia-X principle of self-described identity: participants present Self-Descriptions (composed of VCs) to prove their identity and attributes. In the future, depending on its maturity, it may also be possible to leverage the Gaia-X registry to facilitate interoperability for user organisations and across data spaces.
- Trust Framework Compliance (Gaia-X): The ETDS design is compliant with the Gaia-X Trust Framework, which defines rules for assuring trustworthy and transparent data sharing. In practice, this means ETDS participants must provide certain standard information about themselves and their data offerings in the form of Self-Descriptions (SDs). An SD is essentially a package of VCs (participant attributes, service attributes, etc.) that can be verified. The EDC includes extensions for Gaia-X compatibility – it can generate and verify Gaia-X compliant Self-Descriptions for participants and datasets. For example, information such as a company's legal name, registration number, and adherence to specific codes of conduct can be part of its credentials. By validating these, connectors ensure they interact only with known, trustworthy organisations. This fosters a web of trust where every data transaction in the ETDS is between verified parties, discouraging illegitimate actors. It also helps with compliance: organisations must meet certain criteria (e.g., be legally identifiable and agree to the data space terms) before they can join and exchange data, thereby reinforcing accountability and trust.
- Access and Usage Control: Data sovereignty also implies that data providers retain control over how others use their data. In ETDS, this is enforced through fine-grained access policies and consent. Using the ODRL-based policies (as mentioned in the previous section), a provider can specify exactly who can access data and under what conditions (e.g., "only tourism agencies with a certain trust level can access this API, and data must be deleted after 30 days"). The EDCs and FDC's Policy Engine evaluate these rules in real time at the control plane and during exchange. Crucially, policy decisions can be based on the requester's trusted attributes obtained from their VCs – e.g., checking that "Data Consumer X is accredited as a public authority" before allowing access. This dynamic, attribute-based access control ensures sovereign usage enforcement: the provider's conditions travel with the data and are always checked by connectors. Additionally, because both parties digitally sign contracts and can be revoked if the terms are breached, there is a clear, verifiable agreement governing each data exchange.
- Secure Communication: All network communication in the ETDS is secured using state-of-the-art protocols (e.g., TLS for transport security, etc.). The EDC connectors authenticate each other using their digital identities (DIDs/VCs or OAuth2 tokens). Notably, EDC also supports OAuth 2.0 for scenarios where a centralised authentication server might be used within an organisation. In the tourism data space, OAuth2 could be used for an internal user portal, but inter-organisational exchanges rely on a

decentralised identity mechanism. The combination of mutual TLS and credential exchange prevents eavesdropping and impersonation – only authorised parties can initiate data transfers.

- Logging and Audit: To build trust, participants need assurance that all actions are recorded and auditable. The ETDS includes Audit Services (as part of the governance building blocks), which log all data transactions immutably. This audit trail (maintained in append-only logs) can be inspected by the data space operator or authorities to detect misuse or breaches of contract. For instance, if a provider suspects a consumer violated a term, the logs can help trace what was accessed and when. This transparency is essential for trust in a federated environment.

ETDS' design ensures that each participant stays in control of their data and identity. Data sharing occurs based on clear mutual trust, established through verified identities (DIDs/VCs) and governed by ODRL-defined access and usage policies. Together, these measures fulfil the "secure and sovereign" aspect of the data space, as advocated by IDSA and Gaia-X. From an implementation perspective, the EDC and FDC frameworks provide the necessary components (Identity Hub, DID registry, VCs, policy engines) to realise this trust fabric. By using EDC and possibly FDC, the ETDS benefits from an already tested security architecture aligned to these European identity standards, significantly reducing development effort and increasing confidence in the prototype's security.

2.1.3 Federated and Decentralised Architecture

Another core design principle of ETDS is a federated architecture: data remains with the parties that own it, rather than being aggregated in a central platform. This principle is directly inherited from the IDS Reference Architecture and data mesh concepts, which prescribe decentralisation by default for data spaces.

In practical terms, this means the ETDS is built as a network of distributed data connectors operated by each participating entity, rather than a monolithic system:

- Connectors at the Edge: Each data provider or consumer in the tourism data space runs its own instance of a Data Space Connector (designed based on the ETDS Connector stack, which is initially based on EDC and possibly FDC as a second step, see section 3). This Connector is the participant's gateway to the data space. It interfaces with the participant's internal systems or data sources, and it handles all interactions (catalog publishing, negotiating contracts, data transfer initiation) with other participants' connectors. Because every participant has an independent connector, data exchange is fundamentally peer-to-peer. For example, if a museum shares data with a travel agency, the museum's connector and the agency's connector communicate directly to negotiate and execute the data transfer – there is no central server through which the data flows. This aligns with the "connect once, reach many" ideal of self-service data meshes, where each organisation only needs to integrate with their own connector to be connected to all others.
- Minimal Central Services: While the architecture is primarily decentralised, some federation services can be provided centrally when necessary for the ecosystem. These do not handle actual data transfers but facilitate network-wide functions. In ETDS, such optional central components might include:
    - A Federated Catalog (or discovery service) that aggregates high-level metadata of all offerings in the data space. This acts like a "yellow pages" – participants' connectors periodically sync with it so that new entrants can find available datasets without crawling each connector individually. Importantly, even this is optional: participants can discover data via peer-to-peer catalog queries as well. The federated catalog just improves efficiency at scale. EDC supports deploying a centralized Federated Catalog service if needed, essentially a read-

only index of catalogs. FDC, on the other hand, supports deploying a centralised Federated Marketplace service if needed, enabling the setup of contracts from a one-stop shop and easing monetisation.

- A Registration Service for onboarding, which keeps the list of all trusted participants in the data space. When a new organisation joins ETDS, it registers with this service (likely run by the data space operator), which then issues the organisation's "membership" Verifiable Credential. Connectors use the registration service to obtain the up-to-date participant list and to verify if a caller's DID is a current member (preventing interaction with revoked or rogue participants).
- Possibly a Certification/Clearing House role to validate compliance (though in EDC/IDS, these are usually outside the core data exchange flow). These central components are kept lightweight and not in the data path. Their failure or unavailability should not impede already onboarded participants from exchanging data in the short term (for resilience). The architecture thus follows IDSA's guidance: use centralised services only "where applicable and useful" (for bootstrapping or enabling specific business models), but keep the default mode decentralised.

- Domain-Driven Decentralisation: Following data mesh principles, data remains with data owners. In ETDS, this means, for instance, that hotel data stays in hotel IT systems, airline data stays with the airline, etc., and each participant exposes what it wants through its connector. This avoids creating a large data lake and respects organisational boundaries. It also ensures ownership of the source of truth – data is fetched from the source via the connector, reducing duplication and inconsistency. Similarly, it facilitates the creation and enforcement of data sovereignty by supporting access and usage policies set up by data owners.
- Peer-to-Peer Negotiation: A consequence of federation is that contract negotiation is a direct dialogue between participants. If a tourism agency wants to access a dataset from a national tourism board, the two connectors will communicate to negotiate terms (possibly fully automated). They iterate through offer and counter-offer messages defined by the DSP until they reach an agreement, resulting in a contract signed by both parties. This process does not require a central broker to intermediate; the architecture itself is the broker network. That said, a participant could delegate negotiation to an intermediary service if desired (for example, via a data marketplace interface), but that would be built on top of these basic capabilities.
- Decentralised Data Transfer: After agreements, the data plane operation – the actual data transfer – also happens directly between the consumer and providers. The ETDS Connector's Data Plane component can directly stream or transmit data from provider to consumer without routing through any third party. This is essential for performance and security (no unnecessary data copies). In ETDS, two modes are possible once a contract is in place:
  - Consumer-Pull: The consumer connector pulls data via the provider's Data Plane (e.g., acting as a proxy to the provider's API). This is suitable for on-demand queries where the consumer periodically fetches updates.
  - Provider-Push: (Future capability) The provider pushes data to a known consumer endpoint or storage. In either case, the exchange is point-to-point.
- No Central Data Storage: The ETDS operator does not store all data centrally; it only hosts the supporting federation services (like participant registry or maybe a federated catalog). Only metadata can be stored centrally. Thus, participants do not have to "hand over" their datasets to a platform to share them; they stay in control and share on demand via connectors. This also greatly reduces legal complexity, since data is not

being transferred to a central entity for redistribution – instead, it is shared directly under bilateral terms.

The federated design inherently improves scalability (as each participant adds its own capacity) and prevents single points of failure or control. It also aligns with European values of autonomy: each organisation maintains autonomy over its IT and data. Technically, the ETDS Connector architecture, based on EDC and FDC, will bebuilt to support such distributed deployments – multiple control planes and data planes interacting in a network. The cloud-agnostic nature of EDC and FDC further supports federation: connectors can run on any infrastructure (on-premises servers, cloud providers, etc.) as each participant prefers. This heterogeneity among participants' backends is addressed by maintaining interoperability through common protocols.

The ETDS is not a single application, but an ecosystem architecture. It embraces decentralisation, much like the internet itself, with central directories but no central data store. This principle is vital for reducing barriers to entry (each participant can join with minimal dependence on others) and ensuring the overall system's resilience. Should any one participant or central service go offline, the rest of the network continues to function – a hallmark of a robust federated system.

2.1.4 Modularity and Extensibility of Components

To accommodate the diverse needs of the tourism sector and to future-proof the data space, the ETDS architecture follows a modular design. Each major function of the data space is implemented as a distinct component (building block) with well-defined responsibilities and interfaces. This modularity allows for extensibility – the ability to plug in new components or update/replace implementations as the system evolves – and flexibility in deployment (different setups can include or omit certain components as needed).

The main components (as identified in the ETDS high-level architecture include, but are not limited to:

- Control Plane (Connector Core): Manages data offers, handles negotiations, and orchestrates data transfers. This is the centrepiece of a participant's ETDS Connector stack. Built on EDC and FDC, the control plane is inherently modular: EDC itself is designed in a micro-modular fashion, with features such as transfer protocols, storage backends, and identity providers added via extensions. For example, if ETDS decides to support a new protocol or cloud storage, an appropriate EDC extension can be integrated without overhauling the entire system. FDC, on the other hand, enables compatibility with a federated marketplace compliant with DOME.
- Data Plane: Executes the actual data transfer once authorised. The Data Plane can have different implementations (for streaming, file transfer, etc.). For pure B2B data sharing at the organisational level, EDC currently provides an HTTP-based data plane for REST and is extending support to other protocols (e.g., S3, Kafka). In ETDS, participants can deploy the default data plane and later extend it for large file transfers or real-time streams as needed. Because the data plane is separated from control logic, improvements here will not break other parts. For direct interactions between users at consumer organisations and services offered by providers (not only data access services but also services for processing and visualising data), FDC provides support for controlling these exchanges via REST APIs.
- Federated Catalog: Manages the metadata about available datasets (assets) and makes them searchable. The metadata model is defined following the DCAT standard. The catalog component aggregates the definitions of the assets shared by each participant and offers a search index. This aggregation can be performed following either a pull (i.e., central catalog crawls the assets from the participant connectors) or a push (participant connectors inform the central catalog about the assets being

offered) approach. Ideally, the central catalog would respond to a participant's query only with those assets that could be accessed by her based on her credentials and the access policies defined by the data owner. However, this is still in an incipient stage and needs to be explored in greater depth. If a more advanced search capability or a different metadata standard is required in the future, that component can be upgraded independently, as long as it continues to expose the standardised interface (e.g., other connectors expect a DCAT feed).

- Vocabulary Hub: (Planned) A service to host shared vocabulary, ontologies, metadata models and data models for the tourism domain. While it is not an ETDS
- Connector component, it is a building block that can greatly enhance interoperability. Its existence is modular – it can be integrated to offer semantic services (model discovery, mapping suggestions), but the core connector operations do not depend on it. Thus, it can be introduced when semantic harmonisation becomes a priority. Its implementation can be as simple as a public repository where participants can access or download these models or definitions to apply them locally in their deployments, structure or enrich their data, and then share.
- Identity & Trust Services: Including DID Registry and Identity Hub (VC wallet). These can be operated per participant or as shared services. For example, each participant might use the default DID web method (storing their DID document on their own web server). Alternatively, a consortium might run a joint blockchain or database for DIDs. The architecture permits either – it is abstracted behind the DID resolution interface. Similarly, participants could each host their own identity hubs or use third-party services. This flexibility ensures the identity subsystem can evolve (e.g., if new SSI standards or EU digital identity wallets emerge, ETDS can incorporate those by adding support in this module).
- Policy Engine: Enforces access and/or usage control policies. It is usually embedded as a library or extension within other components (e.g., connector, catalog, etc.). If the policy logic needs to be updated (for example, to support new types of access or usage control policies), only this engine module needs to be modified. The rest of the architecture calls it to evaluate policies. Moreover, note that even when the operator offers a policy engine as an extension for participant connectors, the policy engine can be different for each participant. Its mission is to evaluate the policies that a participant has defined for its own assets. Thus, as a participant establishes new policies, she may need to extend her local policy engine to evaluate them.
- Administration Services: e.g., Registration Service for onboarding as discussed, Audit Service, Billing Service, Data Space portal. Each of these addresses a separate concern (membership management, logging/auditing, usage accounting). They can be developed and improved in parallel. For instance, a billing component might not be needed initially (if data sharing is free). Still, once monetisation is introduced, a billing module can be plugged in without restructuring the whole architecture. Similarly, a Data Space portal can be offered by the operator to assist participants with configuration aspects (e.g., related to identity), to host the Vocabulary Hub or to provide an interface for the federated catalog, among others.
- User Portals/Dashboards: Though not core “data exchange” components, the design foresees participant and operator portals for usability. These sit on top of the connectors (using their APIs) and can be iterated on or replaced independently (e.g., one organisation might integrate connector APIs into its existing dashboard instead of using a provided portal).

This modular breakdown follows the conceptual model recommended by the Data Spaces Support Centre (DSSC) Blueprint and IDSA reference architecture: it separates Participant-facing components (connectors, identity wallet) from Dataspace-wide components (registries,

federated services) and from Support components (semantic, monitoring, etc.). Each module can be implemented by different technologies or vendors as long as they adhere to the interfaces. In ETDS's prototype phase, many modules leverage EDC's or FDC's default implementation (since both frameworks provide reference implementations for components such as connector's verifiers, catalog, identity hub, policy engines, etc.). But the architecture does not lock the project into one technology. For example, if a specialised Vocabulary Hub from another project is better, it can be integrated in place of (or alongside) an EDC component.

EDC and Extensibility: The Eclipse Dataspace Components are designed to be extensible. All core features in EDC (transfer protocols, data formats, authentication mechanisms, storage, etc.) are provided as plug-in extensions to a common framework. The ETDS can benefit from this by adding custom extensions if needed. This is precisely the basis for combining EDC and FDC. For instance, if the tourism data space needs to integrate with a legacy SOAP/XML web service, a custom data plane extension could be written for that format without altering other parts of EDC. This modular architecture also eases maintenance and scalability – each component can be scaled out (run in multiple instances) or updated independently. Connectors and data planes can be containerised and scaled on cloud infrastructure as needed without affecting the operation of other components, since they communicate via stable APIs.

Modularity ensures that the ETDS architecture is not monolithic – it is flexible and adaptable. New capabilities can be introduced by adding new modules, and existing ones can be upgraded with minimal impact overall. This design principle safeguards the prototype against technological change: as standards evolve or as new requirements emerge (e.g., support for new data types, new trust services, etc.), the ETDS can evolve by module substitution rather than a complete redesign. It is an embodiment of "future-proof design".

2.1.5 Security, Resilience, and "Design for Failure"

The ETDS's architecture is engineered with a "design for failure" philosophy, meaning it anticipates that things will occasionally go wrong – whether it is network outages, component crashes, or malicious attacks – and includes mechanisms to handle such situations gracefully. Coupled with this are practices ensuring performance and scalability, so that the data space can reliably serve a growing number of participants and data exchanges.

Resilience and Fault Tolerance: The distributed nature of the ETDS already lends resilience (since there's no single point whose failure brings down everything), but within each component and interaction, additional mechanisms improve reliability:

- Component Decoupling: The architecture aims to minimize tight coupling. For example, the control plane is separate from the data plane – if a data transfer fails, it does not crash into the control logic. Connectors are also decoupled from each other, interacting via asynchronous protocols. If one participant's connector is temporarily down, others can retry later or route around it. Decoupling "limits the blast radius" of an issue, isolating failures to one part without cascading.
- Retry and Circuit Breakers: Connectors use robust communication patterns like retries for transient errors and circuit breakers to avoid overload. Suppose a provider's connector is momentarily unreachable; the consumer's connector can automatically retry the request after a backoff. If a certain service (like the catalog query) is failing repeatedly, the connector might pause calls to it (circuit-break) to not waste resources and to give it time to recover.
- Redundancy: In critical deployments, important services can be run in redundant configurations. For central services (e.g., the participant registry or federated catalog), the ETDS operator could have a failover instance in another region. Participants

themselves might cluster their connector for high availability. Because EDC and FDC components are cloud-agnostic, one could deploy multiple replicas behind a load balancer or use Kubernetes for self-healing restarts. · Monitoring and Observability: The design includes an Observability & Alerting layer as part of the architecture. All connectors and services emit telemetry (metrics, logs, traces) which are collected in Telemetry backends using open standards (e.g., Open Telemetry). This means operators have real-time insight into system health: performance metrics (throughput, latency), usage statistics, error rates, and detailed logs of each transaction. With this observability, issues can be detected and localized quickly – for instance, if one connector version is causing errors, or a specific data transfer is stuck, the traces will show exactly where. Alerting can be set up on critical conditions (like an inability to reach the registry or a spike in failed policy checks). In a distributed system like ETDS, end-to-end tracing is especially valuable to follow a transaction through multiple services and pinpoint bottlenecks.

- Scalability Mechanisms: As more participants join and traffic increases, the system should scale without a drop in service quality. ETDS benefits from horizontal scalability: new participants add more connectors, and central services can be scaled by increasing resources or instances. The use of cloud-native technologies (containers, orchestration) allows the data space to scale elastically. For example, if the query load on the federated catalog grows, that service can be scaled to multiple instances and the load distributed, thanks to stateless design. Meanwhile, individual connectors handle only the load related to their organisation's data, which is naturally partitioned.
- Performance Optimisations: The architecture allows caching where appropriate. Connectors cache discovered catalog entries locally to improve search speed and reduce repetitive queries. They also cache verified credentials and DID documents for participants, so that not every interaction requires re-fetching from the source (within safe expiration limits). Such caching improves responsiveness and reduces external calls, thereby increasing resilience if external identity or catalog services are slow.
- Robust Security Posture: Security is tightly integrated (as discussed under Trust), which also contributes to resilience. Using mutual authentication and authorisation keeps rogue actors out, reducing the risk of malicious overload or data leakage. Connectors are hardened through secure coding practices from EDC and FDC and can leverage cloud security controls (e.g., network isolation, firewalls). Regular security audits help maintain trustworthiness over time.
- Failure Handling in Data Flows: If an ongoing data transfer fails mid-stream (due to consumer crash or network cut), the system can recover. Since contracts remain valid until revoked, a consumer can re-request the data after a failure without renegotiating the contract, provided the contract period is still valid. In future, more sophisticated "bulk transfer" support will allow automatic restart of partially completed transfers.
- Design Testing: A principle implicit in "design for failure" is to test the system under failure scenarios (chaos engineering). While building the prototype, we assume such tests (e.g., disconnecting a connector, dropping messages) are done to verify that the system tolerates them as expected.
- Embracing this principle leads to an architecture capable of self-recovery and graceful degradation. For example, if the central catalog is offline, connectors can still use their last-known cache to answer queries, so data sharing does not halt completely. Or if one data provider's node goes down, only that provider's data is temporarily unavailable, not the entire tourism data space. This resilience is crucial for a production system that runs 24/7 across many independent organisations.
- EDC and FDC components are designed to be cloud-deployable and are used in various projects, benefiting from ongoing improvements. The cloud-neutral design (running on Azure, AWS, on-premises, etc.) avoids dependence on a single

infrastructure provider, aligning with Europe's push for no vendor lock-in and portability. The development in both frameworks also actively incorporates community feedback on performance and security, meaning ETDS will build on a mature, continuously improving codebase.

2.1.6 Summary

The Generic Design Principles outlined above provide a blueprint for the ETDS Prototype's architecture. They ensure that, as the European Tourism Data Space is built out, it will enable sovereign data sharing across the tourism ecosystem in an interoperable, secure, and resilient manner. These principles are generic; they could apply to any data space. However, in ETDS, they are tailored to the specific context of tourism (e.g., emphasising trust and compliance due to sensitive personal data, such as traveller information, and interoperability given tourism's cross-domain nature). By adhering to these design principles, the ETDS will be well-positioned to integrate with other European data spaces and scale from prototype to a production ecosystem that realises the EU's vision for a common data space in tourism. The use of EDC and FDC, in alignment with IDSA/GAIA-X, means the prototype will be built on solid, standard foundations, reducing risk and facilitating future enhancements.

2.2 Overall Prototype Architecture Overview

Building upon the design principles outlined in the previous section, the ETDS prototype architecture is structured as a federated ecosystem. This design decouples the governance and trust mechanisms from the actual data exchange, ensuring that while trust is centralised, data remains distributed and sovereign.

The high-level logical view of the MVDS architecture is depicted in Figure 2. The architecture is organised into three distinct layers: the Data Space Federation Services (Governance Layer), the Participants (Data Layer), and the Shared Infrastructure (Support Layer).

2.2.1 The Data Space Federated Services (Governance Layer)

At the top level of the hierarchy sits the Data Space Federated Services. In the prototype, this entity does not act as a data aggregator or a middleman for data traffic. Instead, it serves as the root of trust and discovery for the ecosystem. Its primary responsibilities are:

- Trust and Identity: It hosts the Participant Registry and Trust/Credentials Verification services. It acts as the trusted anchor that issues credentials to eligible participants, allowing them to prove their identity to others without requiring prior bilateral agreements.
- Discovery: It provides a Federated Catalog. While data remains at the source, the Authority aggregates metadata (descriptions of data products) to facilitate discovery. This allows consumers to search for datasets across the entire network from a single point of entry.
- Telemetry and Audit: It collects operational metrics and audit logs to ensure the stability of the network and compliance with the dataspace rules, without inspecting the payload of the data transfers.

2.2.2 The Participants (Sovereign Nodes)

The core of the architecture consists of the Participants, defined by their roles as Providers or Consumers (noting that a single entity can play both roles).

Each participant operates a Connector (Gateway), which acts as their sovereign agent within the dataspace. Following the reference architecture, the Connector is strictly divided into two logical planes:

- The Control Plane: This component handles the intelligence of the system. It manages identity (communicating with the Authority), publishes or queries metadata, and orchestrates the Negotiation process. When a Consumer requests data, the Control Planes of both parties communicate to agree on the usage of policies and access rights.
- The Data Plane: Once a contract is negotiated by the Control Plane, the Data Plane executes the actual Data Exchange. It interfaces directly with the Provider Backend Systems (data sources) and the Consumer Systems (sinks/apps).

2.2.3 Interaction Flows and Separation of Concerns

A critical architectural feature is the separation of the interaction flows:

- Negotiation Flow (Control Plane): Represented in figures by the dashed line, this flow handles the "handshake" between participants to establish trust and agreement.
- Data Exchange Flow (Data Plane): Represented by the solid double arrow, this flow occurs peer-to-peer between the Provider and the Consumer. The data flows directly from the source to the sink via a secure, encrypted channel. Crucially, the Data Space Authority is never involved in this flow, ensuring that no central entity has access to the business data.

2.2.4 Shared Infrastructure

Underpinning these interactions is the Shared Infrastructure layer. In the MVDS prototype environment, this layer provides the ecosystem-wide runtime support services, including centralised logging, persistent storage for system state, key management services, and the networking backbone that interconnects the distributed components. (see Fig. 1 below)

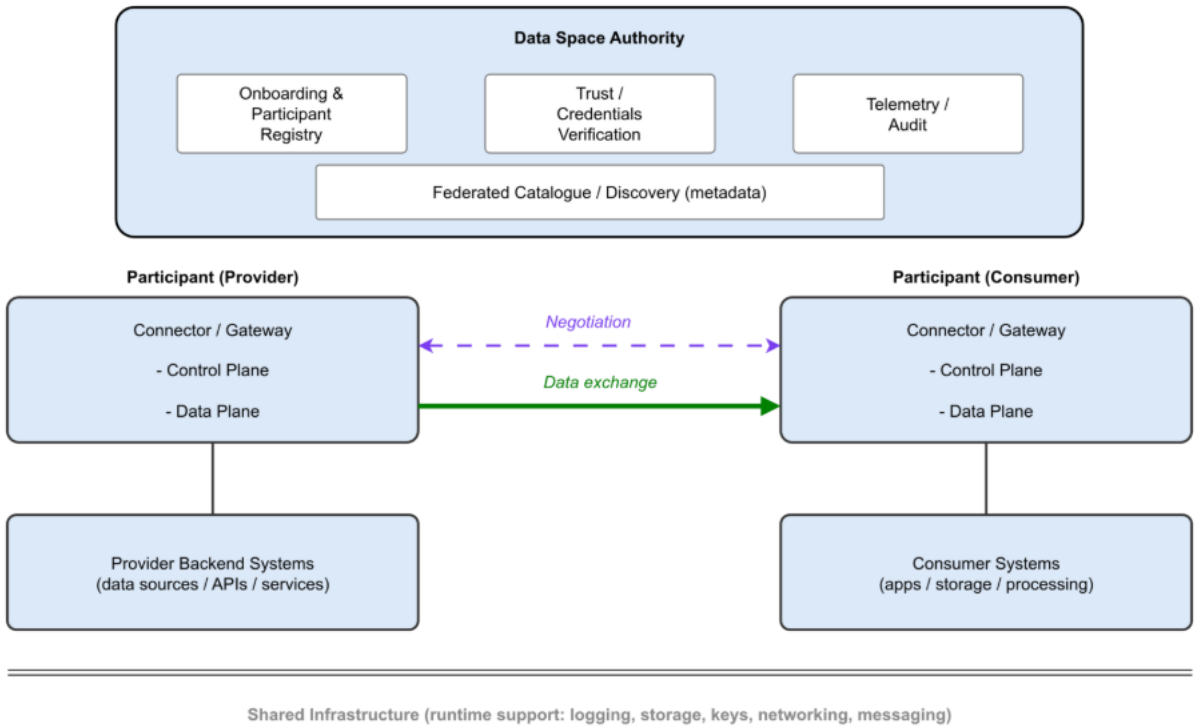

Figure 1 Shared infrastructure

3 Definition of Prototype Components

While this section describes the initial MVDS of the ETDS Prototype, our main objective at this stage is to deploy a robust MVDS. The core components of the MVDS infrastructure will be the set of Data Space Federation Services operated by the Data Space Governance Authority and the technology stack for connectors deployed by participants in the Data Space (the ETDS connector stack).

The consortium has revised in detail stacks such as EDC, FDC, and SIMPL. For this first MVDS, the consortium will mainly work to preserve the EDC and FDC stacks allowance. SIMPL is not considered at this stage due to its low maturity, but it is not discarded. It is important to note, however, that this election does not constrain or limit the stacks that could be used by the ETDS participants, the only requisite being that, for technical interoperability at the control plane, IDS DSP-compliant connectors are used. Therefore, approaches like SIMPL should be integrated as they mature.

The ETDS Connector stack offers a modular, extensible approach that enables secure identity management, federated catalogs, contract negotiation, and controlled data sharing among participants, while meeting the basic requirements for an operational MVDS. MVDS components of the ETDS Connector stack focus on implementing the essential services that guarantee sovereign data sharing. The design ensures scalability and flexibility, allowing traditional components and advanced features to be integrated as the project evolves. This approach provides a solid foundation for future expansion toward a fully functional ETDS.

The following subsections describe the components identified as essential to the MVDS architecture and the main interaction flows between them.

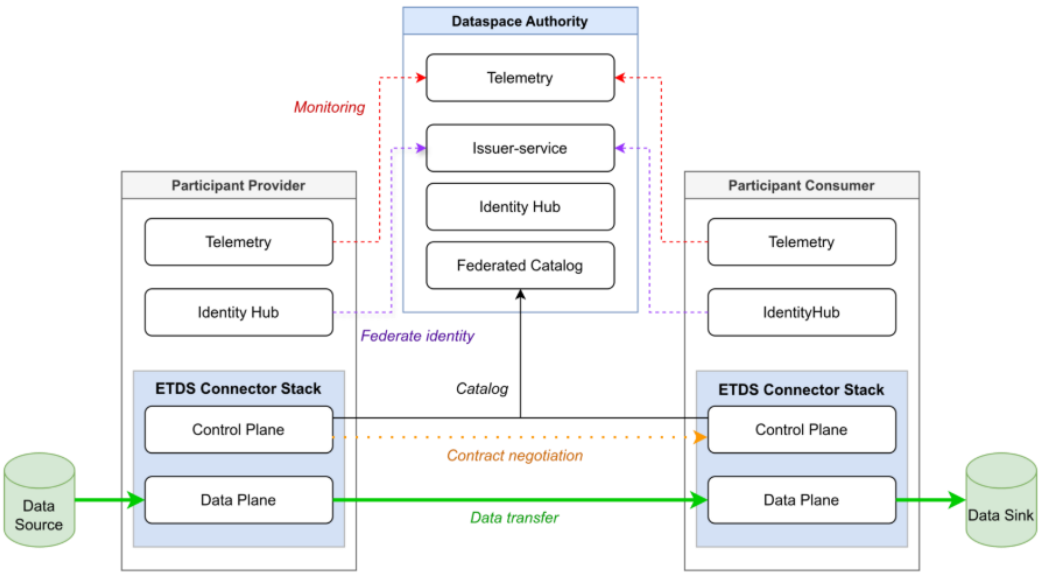

Figure 2 High-level logical architecture

3.1 Description of Main Component

3.1.1 Data Space Federation Services

The Data Space Governance Authority, which could be seen as the data space operator, has several key missions. First, it must devise and offer an onboarding process that allows new participants to join the data space, ensuring they can be trusted and providing mechanisms to facilitate their interactions with other participants (e.g., tokens, signatures). An essential global

service, it hosts a list of services that identify which participants are valid or trusted and which verifiable credentials schemas they should follow. Alternatively, it may be responsible for maintaining or handling credential issuance for the service. Secondly, it can operate a federated catalog service that allows other participants to discover the assets being shared in the data space easily. This catalog may be implemented in different ways, such as crawling the different assets offered by the participants, given that the operator may be the only one knowing the entire list of participants, or being the participants the ones informing the catalog instance every time they publish or retire a data product offering. Similarly, this catalog, as well as other services from the data space authority, may be provided through a data space portal that serves as the reference site for the data space, hosting and offering, for instance, metadata, data models, or business vocabularies, among others.

Finally, among its services or functionalities, the Data Space Governance Authority can provide a telemetry service that collects and stores information and statistics from the different connectors. This information could be later shared with the rest of the participants through the data space portal.

The Data Space Governance Authority should provide, at least, the following Data Space Federated Services:

- authority-identityhub: Implements DIDs and W3C VCs used in the Data Space. It exposes OIDC-compliant APIs for authentication and integrates with Vault for cryptographic material.
- authority-issuerservice (optional): Handles the lifecycle of VCs for participants and connectors using JSON-LD/JWT formats.
- authority-federatedcatalog: Aggregates metadata from participants via the DCAT-AP schema, providing SPARQL endpoints or REST APIs for asset discovery and semantic search.
- authority-telemetryservice: Collects operational events (such as contract states and transfer metrics) via an Event Hub and exposes APIs for compliance dashboards.
- authority-telemetrycsvmanager: Generates compliance reports (e.g., monthly CSVs) for auditing purposes.

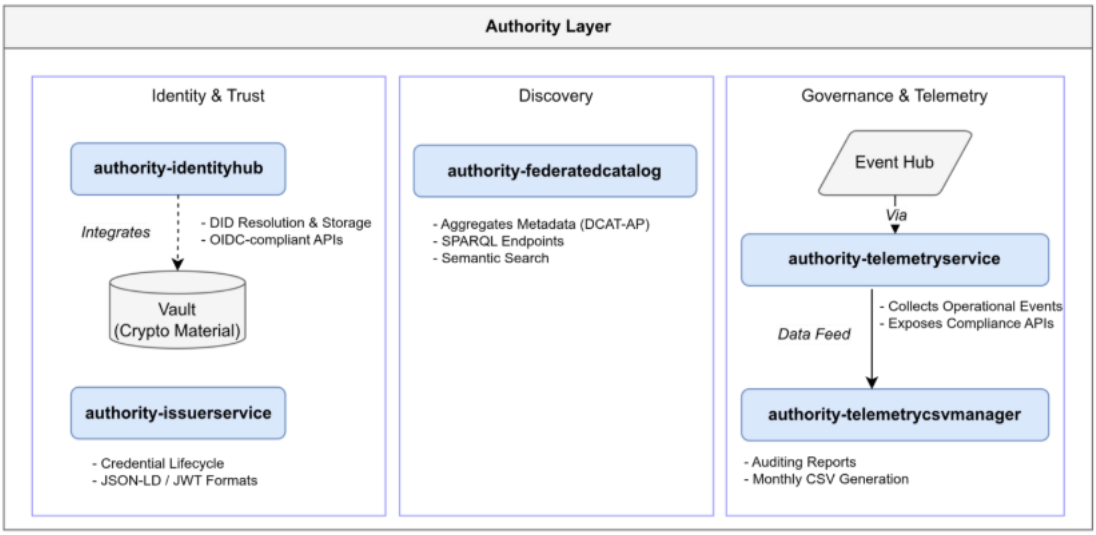

Figure 3 Data Space federation services

3.1.2 Data Provider

A data provider is a participant that publishes data offerings in the data space. Note that a participant does not need to be exclusively a data provider or consumer; they can play both roles. However, for the sake of clarity, we separate these roles to describe the main services and functionalities associated with each role. A data provider is expected to dispose of a data space connector that allows for the publication of data offerings, for the handling of contract negotiations based on offerings upon request, or to start and control the secure consumption of the data.

The DSP protocol must be supported by the connector deployed by the provider for this purpose, but TM Forum Open APIs may be supported in addition (optional). DCP should be supported for authentication when using DSP, but OID4VC may also be supported for authentication when using DSP or TM Forum Open APIs.

The connector is also expected to facilitate the definition and enforcement of access and usage policies once transfer processes have been started.

Note that providers must be able to prove that it is a participant of the data space –providing valid credentials that can be validated by another participant. Finally, as an optional functionality, it may offer telemetry information to the data space authority. It should provide, at least, the following services:

- provider-controlplane: Implements the IDS Connector logic to register data assets and usage policies. It validates VCs via OIDC and negotiates contracts using the DSP Protocol or EDC Contract Negotiation API.
- provider-dataplane: Executes the actual data transfer post-agreement, supporting both PULL (REST API) and PUSH (SFTP, HTTPS) modes. It ensures security via mTLS and integrity checks.
- provider-backend: Interfaces with external data sources (databases, object storage) and hosts domain-specific APIs for consumers.

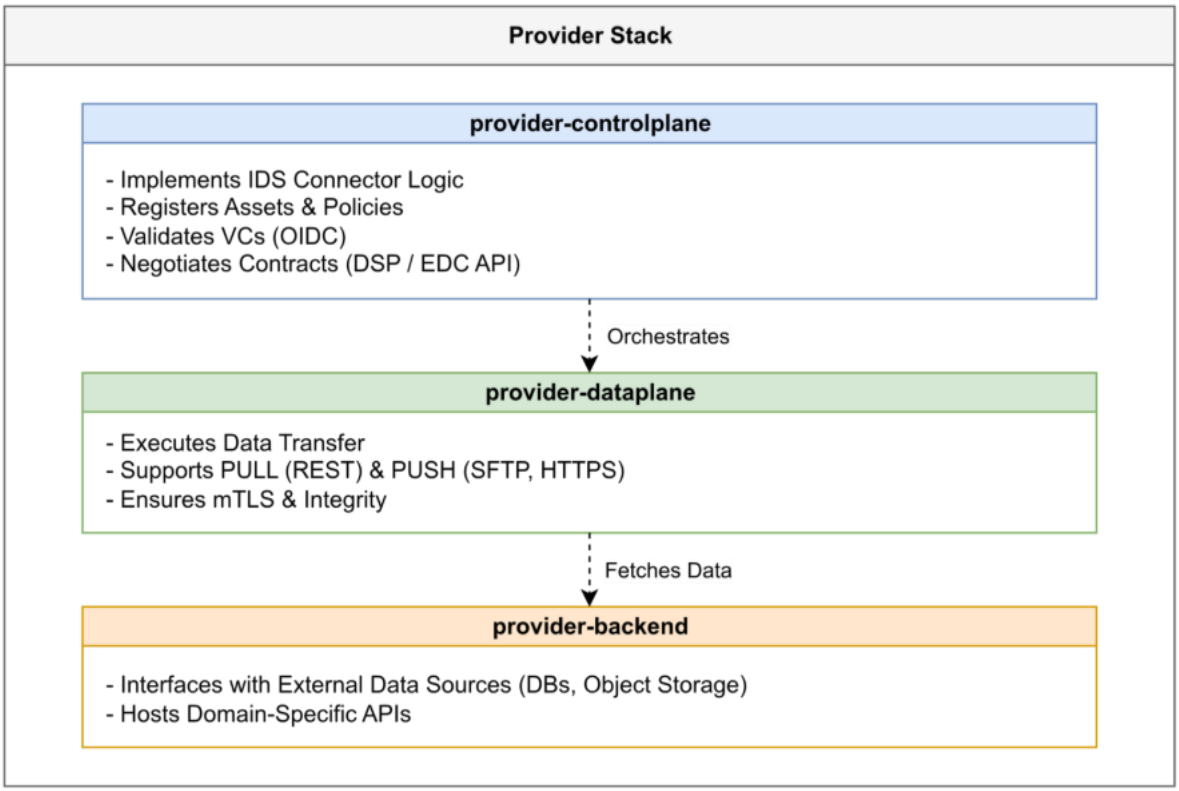

Figure 4 Provider stack services

3.1.3 Data Consumer

A Data Consumer organisation is also expected to use a data space connector to interact with a data provider, request access to a data offering, negotiate it, and initiate the process by which the final transfer will take place. This implies, among other things, that a participant must be able to prove it is a valid and trustworthy entity. It must be able to provide credentials or attestations that a data provider can use to evaluate data access or usage control policies. Based on this evaluation, the data provider can decide whether the participant is allowed to access services associated with its data. The participant must also be able to establish a transfer process under which the actual exchange of data will take place. As with the data provider, it may offer a telemetry service that shares usage statistics with the data space governance authority.

Multiple scenarios for exchange will be supported at the data plane once the transfer process is initiated: exchange can be pure B2B data exchange or may involve users linked to the consumer organisation. FDC components that implement OID4VP can be used when authentication with VCs is required for these users. Note that users within the consumer organisation do not need to deploy a connector; they can use digital wallets to store their VCs.

A data consumer should offer at least the following services:

Consumer-controlplane: Responsible for discovering assets via the Federated Catalog, negotiating contracts, validating credentials, and maintaining contract states.

Consumer-dataplane: Handles the secure ingestion and retrieval of data files or the invocation of APIs offered by the provider. For now, only file transfers are supported; stream transfers will be included in the future. It also integrates with consumer-side analytics or storage systems.

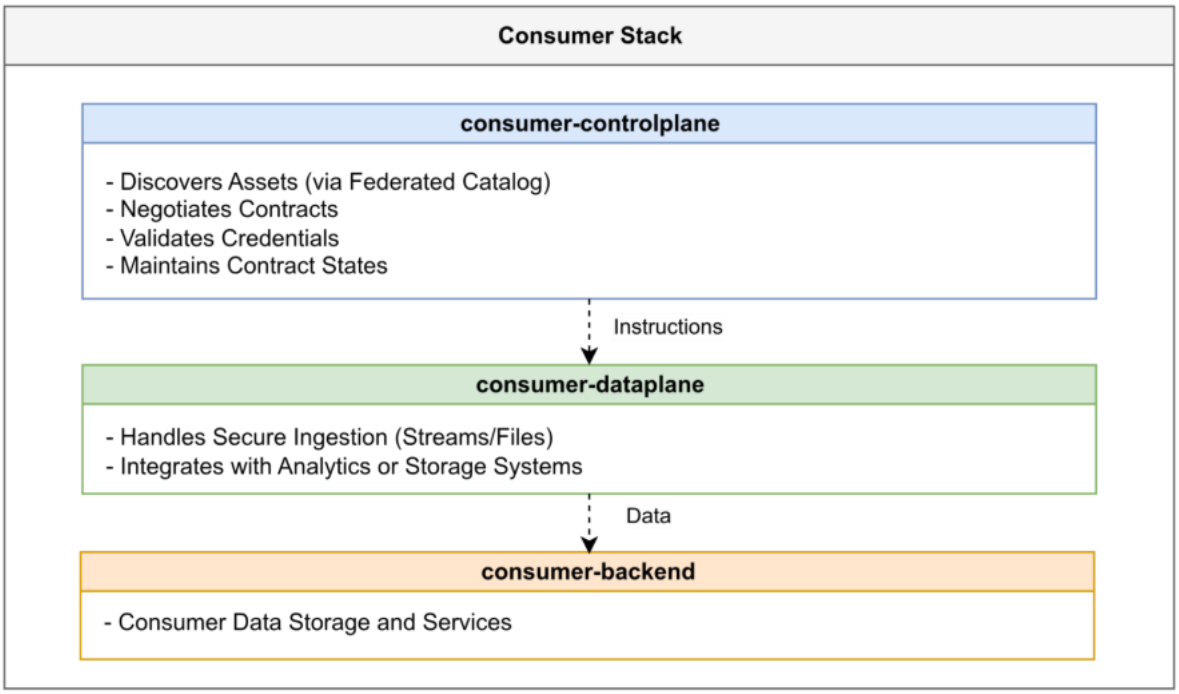

Figure 5 Consumer stack services

3.1.4 User Interface (UI)

The frontend architecture is designed to ensure scalability and prioritise integration with backend services. The UI development relies on the following core technologies:

- Framework: Angular 18.
- UI Library: Bootstrap.
- Authentication: AuthGuard.
- API Integration: Auto-generated OpenAPI Clients.
- Testing: Jest + Playwright.

Table 1 shows the website's navigation structure. The following paragraphs outline the UI pages, their routes, user roles, authentication requirements, and core features. Comprehensive specifications regarding functions can be found in Annex 1.

Section	Route	Component
Home	/	HomePageComponent
Catalog	/catalog	CatalogPageComponent
My Datasets	/dataset-management/dataset	DatasetPageComponent
My Policies	/dataset-management/policy	PolicyPageComponent
My Offers	/dataset-management/contract-definition	ContractDefinitionPageComponent
My Negotiations	/contract-management/contract-negotiation	ContractNegotiationPageComponent

My Transfers	/contract-management/contract-transfer	ContractTransferPageComponent

Table 1 UI Navigation structure

Home (Landing page)

Route: /

Role: all.

Authentication: not required.

Functionalities: Landing page with a login button to initiate authentication.

Federated Catalog Page

Route: /catalog

Role: Consumer and provider.

Authentication: Required (Login and route protection using an authentication guard).

Functionalities: Display datasets available in the federated catalog, navigation functionalities, and dataset detail view.

Dataset Management Page

Route: /dataset-management/dataset

Role: Provider.

Authentication: Required (via AuthGuard).

Functionalities: Participant dataset management (creation, editing, deletion), quick actions for each asset, and navigation.

Asset properties:

- Data Address Configuration: Endpoint, HTTP method, storage.
- Metadata: title, description, categories, publisher, license, temporal coverage, geographical coverage, data format...

Policy Management Page

Route: /dataset-management/policy

Role: Provider.

Authentication: Required (via AuthGuard).

Functionalities: List all ODRL policies defined by the participant. Create, edit, and delete policies. Preview policy in ODRL JSON-LD format. Policy validation and syntax checking.

Contract Definition Management Page

Route: /dataset-management/contract-definition

Role: Consumer and provider.

Authentication: Required (via AuthGuard).

Functionalities: List all contract definitions (Asset + Policy combinations). Create, edit, and delete contracts; view contract details; display eligibility criteria for consumers.

Contract Negotiation Page

Route: /contract-management/contract-negotiation

Role: Consumer and provider.

Authentication: Required (via AuthGuard).

Functionalities: Display all contract negotiations (past and active) and their state. Filtering by state. Cancel ongoing negotiations. Initiate data transfer after agreement. Display cryptographic signatures and verification status; export the contract as JSON or PDF.

Contract Transfer Management Page

Route: /contract-management/contract-transfer

Role: Consumer and provider.

Authentication: Required (via AuthGuard).

Functionalities: List all data transfer processes (past and active) and their state. Display transfer progress. Actions per transfer (suspend, resume, cancel, download transferred data). Monitor real-time transfer metrics.

3.2 Interaction Flows between Components

The operation of the MVDS prototype relies on a set of orchestrated interactions that clearly distinguish the Control Plane from the Data Plane. The Control Plane involves the provider, the consumer, and the dataspace services, which act as orchestrators—such as the federated catalog, contract negotiation services, and coordination of the overall transfer lifecycle. In contrast, the Data Plane establishes a peer-to-peer connection directly between provider and consumer to execute the actual data exchange. This decentralised approach allows participants to remain autonomous while interacting through shared dataspace services, ensuring interoperability and compliance with the dataspace rules. Based on the architectural design, the interactions among components are organised into five key workflows, described in the following subsections.

3.2.1 Federated Identity and Trust Establishment

Trust is the non-negotiable prerequisite for any interaction within the dataspace. Rather than relying on a central gatekeeper for every transaction, the architecture employs a Decentralised Self-Sovereign Identity (SSI) model. Before any data exchange can occur, participants must establish a verifiable digital identity:

- Credential Issuance (Onboarding): The Issuer Service (hosted by the Dataspace Authority) acts as the root of trust. It validates the legal and technical status of new participants and issues of W3C Verifiable Credentials (VCs). These credentials assert attributes (e.g., "Certified Tourism Provider") and are cryptographically signed by the Data Space Governance Authority.
- Storage (Identity Hub): Once issued, these credentials are not stored centrally. Instead, they are held in the participant's own Identity Hub (a decentralised digital wallet). This ensures that participants retain sovereignty and control over their own identity data.
- Runtime Verification (the purple flow in Figure 6): When a Data Consumer initiates a transaction:
    - Resolution: The connectors exchange Decentralised Identifiers (DIDs) to locate each other's DID Documents and public keys via the resolution services.
    - Presentation: The Consumer generates a Verifiable Presentation (VP) derived from its VCs and presents it to the Provider.

- Validation: The Data Provider validates the cryptographic signature of the VP against the Authority's public key (Trust Anchor) to ensure the credentials are valid and have not been revoked.

- Outcome: This process enables mutual authentication and establishes a secure, encrypted communication channel (mTLS) without the Authority needing to be a bottleneck in the actual data transfer.

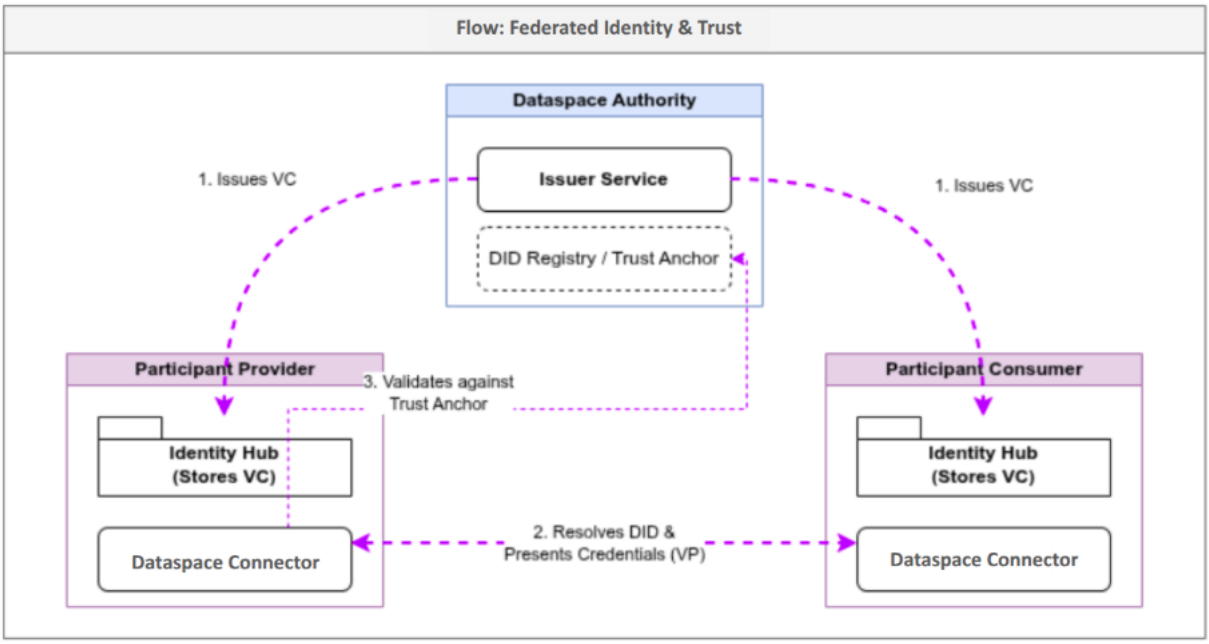

Figure 6 Federated identity and trust flow

3.2.2 Catalog Synchronisation and Discovery

This flow decouples data storage from data discovery, ensuring that while data remains distributed at the source (on-premises or in the cloud), it is visible and searchable across the entire network. This mechanism is critical for the "Federated" nature of the dataspace:

- Metadata Publishing (Data Provider): The Data Provider defines its "Data Assets" (what is being shared) and "Contract Definitions" (rules of use) within its local catalog and makes it accessible via DSP. It maps internal metadata to the dataspace metadata model, which is defined following DCAT-AP principles (note: this is not the DCAT vocabulary itself), ensuring semantic interoperability.
- Aggregation (Authority Side): The Federated Catalog (hosted by the Authority) acts as a central index or "crawler." It periodically synchronizes with the catalogs of registered connectors or receives metadata push updates. It aggregates this information into a searchable index without ever storing the actual business data.
- Discovery (Consumer Side): When a Data Consumer needs data, it does not query every node in the network. Instead, it sends a search query (e.g., via REST or SPARQL) to the Federated Catalog. The Catalog returns a list of available datasets matching the criteria, along with the Connector Endpoint Address and the Protocol required to initiate a negotiation. This step effectively resolves *who* has the data and *how* to reach them.

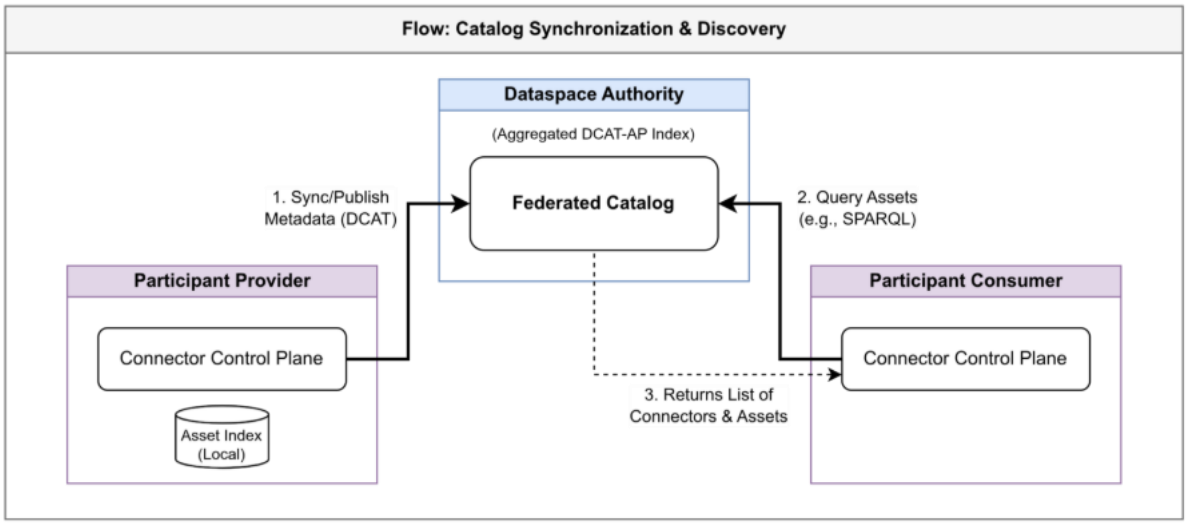

Figure 7 Catalog synchronisation and discovery flow

3.2.3 Contract Negotiation

Once the Data Consumer has identified a desired dataset or data service within a data product offered by a Data Provider, the system transitions to a direct peer-to-peer negotiation phase. This process is fully automated and handled exclusively by the Control Planes through EDC components using the IDSA Dataspace Protocol – Contract Negotiation subprotocol. This process comprises the following steps:

- Initiation: The Data Consumer Control Plane initiates the sequence by sending a Contract Request to the Provider. This request specifies the target asset and the desired usage policy (e.g., "I want to read this data").
- Policy Enforcement: Upon receiving the request, the Data Provider Control Plane triggers its Policy Engine. It evaluates the request against the ODRL policies stored on the asset. Common restrictions include:
  - Temporal constraints: "Access valid for 48 hours."
  - Spatial constraints: "Data must remain within the EU."
  - Purpose constraints: "For research use only."
- Handshake & Agreement: If the request complies with the rules, the connectors exchange messages to finalize the terms. The state transitions from REQUESTED to OFFERED, then to AGREED.
- Binding Contract: The outcome is a Contract Agreement, cryptographically signed by both parties. This digital contract is stored for audit purposes and serves as the "authorisation key" that allows the Consumer to request the actual data transfer in the next stage. Crucially, strictly no business data is transferred during this phase; it is solely a logical negotiation to acquire the right to access the data.

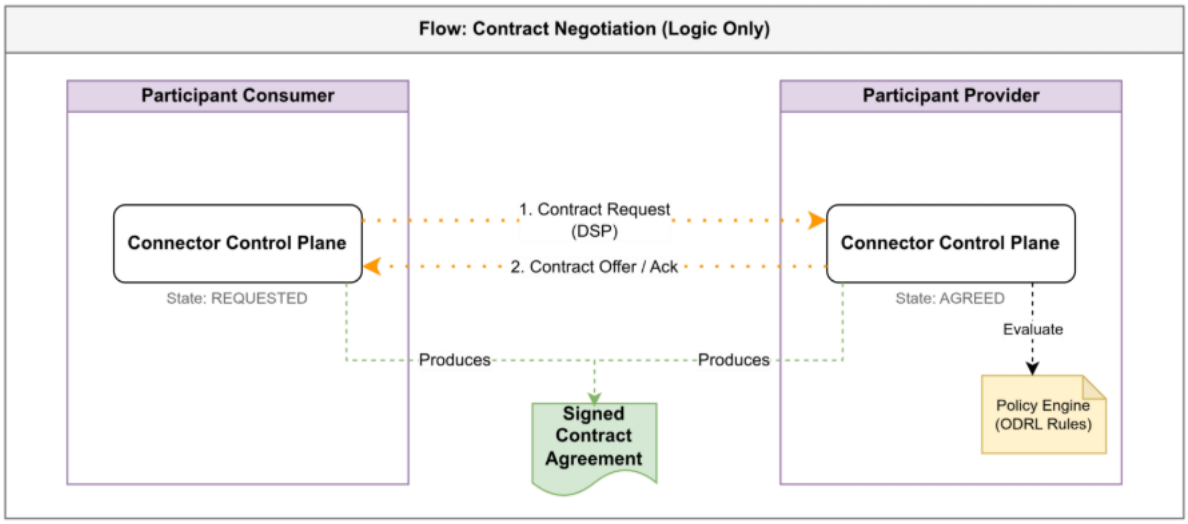

Figure 8 Contract negotiation flow

3.2.4 Creation of Transfer Process

Once a contract has been established the Data Product Consumer request the creation of a Transfer Process to the Data Product Provider under which actual exchange will take place. This process is fully automated and handled exclusively by the Control Planes through EDC components using the IDSA Dataspace Protocol – Transfer Process subprotocol. This process comprises the following steps:

- Token Provisioning (Data Plane Authentication): Once the contract is signed, the Consumer requests the creation of a Transfer Process. This means that the Provider Control Plane returns an Endpoint Data Reference (EDR) that contains a data-address (service endpoint) through which actual exchanges will be carried out together with either a) a temporary, cryptographically signed access token which allows access to the specific asset for a limited time or b) info how to obtain this access token (e.g., using OID4VP on a given verifier endpoint). The second option allows users to support scenarios in which users within the Consumer organisation gain direct access to data services and authenticate via OID4VP (for natural persons such as employees or customers, using VCs stored in their wallets).
- Policy Enforcement (initial application and configuration): prior to returning an access token, the Provider checks whether agreed ODRL policies apply and, if so, will eventually configure authorisation modules at the data plane to enforce these policies during the whole transfer process.

3.2.5 Secure Data Transfer

This is the operational core of the dataspace: the only phase where actual business data traverses the network. Unlike the previous logical flows, this process is handled exclusively by the Data Planes, ensuring high performance and strict isolation from Control Plane components.

- Direct Connection (Peer-to-Peer): The Data Consumer Data Plane uses the token obtained in the previous step to perform requests to the Provider Data Plane. The Provider validates the token without needing to contact the central Authority and check whether the request is compatible with defined policies. When OID4VP is used and the Gaia-X ODRL profile for VCs is used, the authorisation module may extract info from

credentials in the token which, combined with info about requested operations and data payload, can be checked against ODRL policies.

- Abstraction and Streaming: The Data Provider Data Plane acts as a secure gateway. It connects internally to the raw Data Source (e.g., an SQL database, an API, or an S3 bucket), fetches the data, and streams it to the Data Consumer Data Plane. The Consumer component then may write the incoming stream to its local Data Sink.
- Security & Sovereignty: The transfer channel is secured using mTLS (Mutual Transport Layer Security), ensuring encryption in transit. Crucially, this pipe is established directly between the two participants. The Dataspace Authority and other intermediaries are physically excluded from this loop, guaranteeing that no third party can access, view, or store the raw business data.

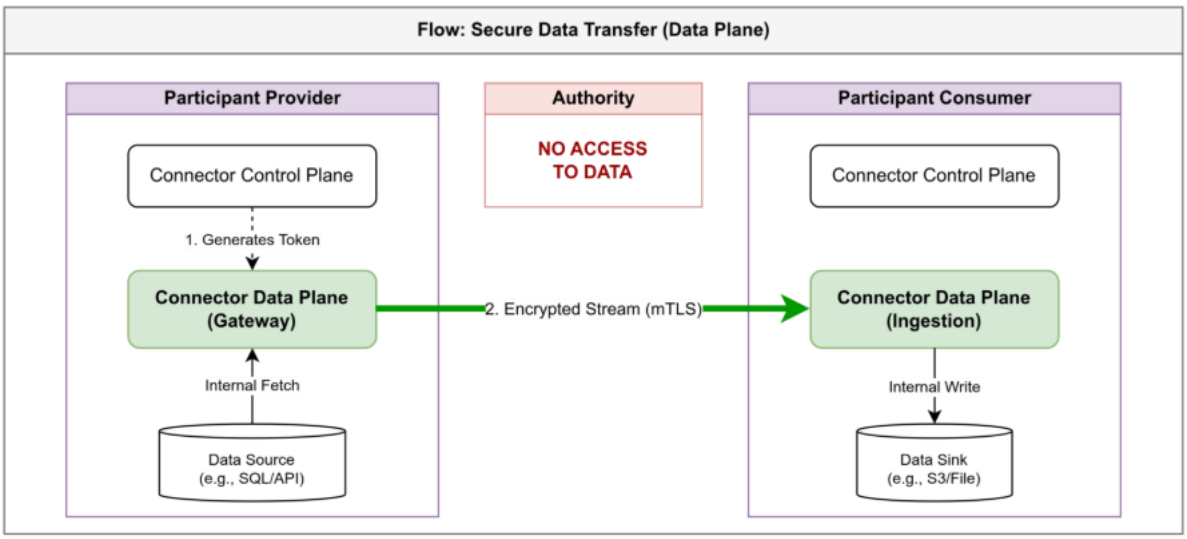

Figure 9 Secure data transfer flow

3.2.6 Telemetry and Monitoring

To ensure the health and compliance of the ecosystem.

- All components (Provider EDC and FDC, Consumer EDC and FDC, and Authority Services) emit operational logs and metrics.
- This data is pushed to the central Telemetry Service (Authority), allowing the operator to monitor network stability and generate compliance reports via the Telemetry CSV Manager.

Considering the five core component interaction flows—identification, discovery, negotiation, transfer, and telemetry—the data-sharing process between two dataspace participants follows the workflow outlined in Figure 10.

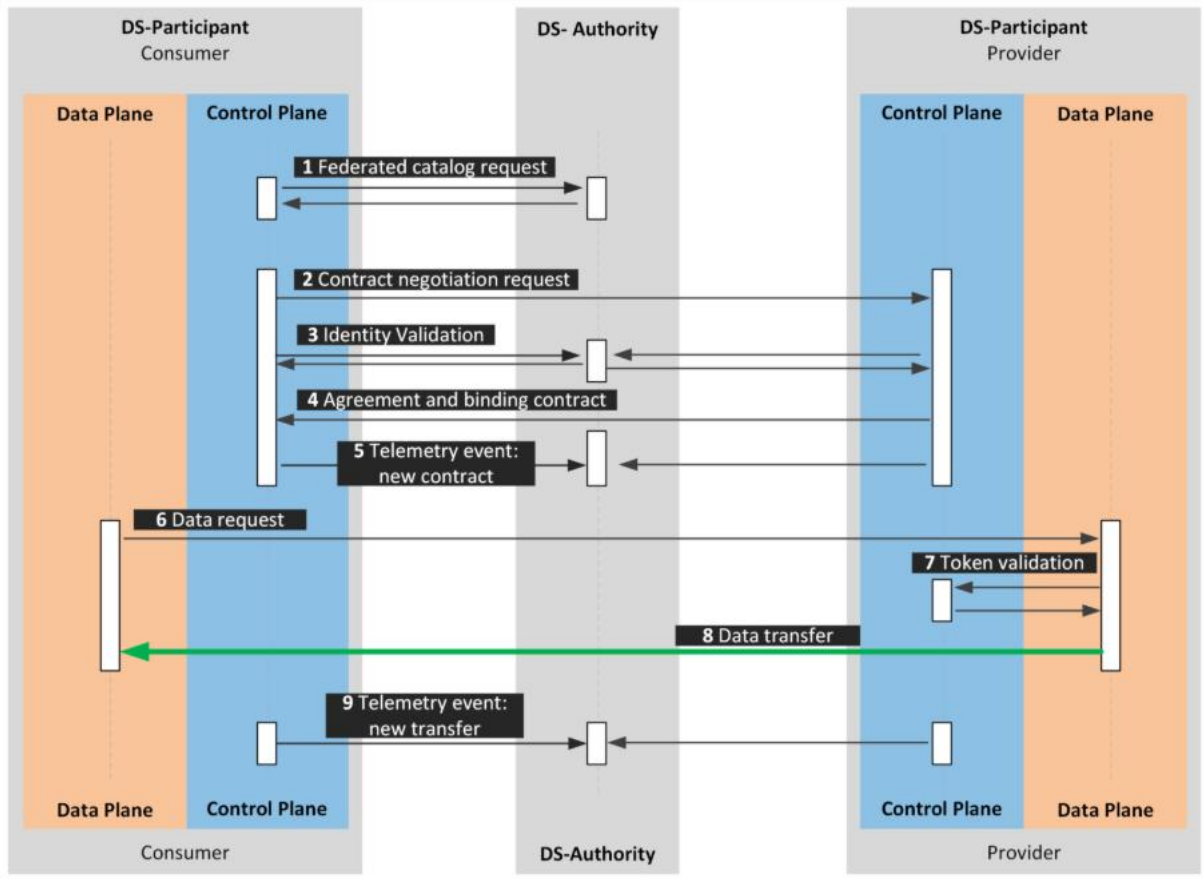

Figure 10 Sharing process between DSs

4 Development Process and Tools

Technical environment:

The MVDS prototype is designed to support functional validation and integration testing, rather than high availability or large-scale production workloads.

The selection of technologies, however, is driven by the need for this model to be scalable, reproducible, and interoperable within a controlled pilot environment.

- Kubernetes is chosen for container orchestration to standardise deployment, simplify service networking, and ensure consistency across different infrastructures. This approach facilitates future scalability and aligns with modern cloud-native practices.
- Helm charts are employed to centralise configuration and streamline the installation, upgrade, and removal of components, ensuring a reproducible and maintainable deployment process.
- For data persistence and event-driven communication, PostgreSQL and Kafka are utilised, respectively. PostgreSQL provides reliable storage, while Kafka enables real-time telemetry and event-based interactions between services.
- Vault is integrated for secure management of sensitive configurations, addressing security requirements without compromising operational flexibility.
- The use of DNS, load balancers, and ingress controllers ensures controlled external access, supporting testing, validation, and observability during the pilot phase.

These technologies collectively create a robust, modular foundation that supports the MVD's functional validation and integration testing objectives, while remaining adaptable for future expansion and fine-tuning.

Frameworks:

The MVDS's technical design aims to align with frameworks such as the DSSC Blueprint, GAIA-X, and IDSA Data Space Protocol by prioritising interoperability. Moreover, it will use the European Interoperability Test Bed (ITB) -based conformance testing for CP, CNP, and TPP, ensuring compliance with standardised JSON Schemas and state transitions. It's Kubernetes and Helm-based deployment support modularity and scalability, meeting DSSC and GAIA-X requirements for flexible, cloud-agnostic architectures. Security is addressed via Vault for secrets management and controlled access, aligning with GAIA-X's federated trust principles. PostgreSQL and Kafka ensure observability, fulfilling DSSC and IDSA transparency mandates. The reusable ITB test framework and Helm-based reproducibility support DSSC's and GAIA-X's emphasis on standardisation.

Use of open-source code:

The MVDS will leverage a robust foundation of open-source components, namely EDC and FDC.

EDC includes core repositories such as the Minimum Viable Data Space, Connector, and Identity Hub, which provide standardised, modular, and interoperable building blocks for data sharing. These components are designed to comply with established data space frameworks, including the IDSA Data Space Protocol, ensuring alignment with EU-wide interoperability and governance standards. More information can be found at:

- https://github.com/eclipse-edc

In addition, the project integrates recently released open-source contributions developed by Amadeus for the EONA-X data space, specifically the dataspace-ecosystem and Dataspace_UI repositories.

- https://github.com/AmadeusITGroup/dataspace-ecosystem
- https://github.com/AmadeusITGroup/Dataspace_UI

These resources enhance the MVDS's functionality by providing operational tools and a user-friendly interface. These elements have proven capable of functioning in operational and production contexts. This should also facilitate interoperability between DEPLOYTOUR and the EONA-X transport and tourism data space. The Dataspace_UI is particularly valuable for facilitating adoption among diverse stakeholders, including those with less technical skills, such as Destination Management Organisations (DMOs) and other tourism data providers, by simplifying the publication of data products, contract negotiation, and catalog management.

On the other hand, the FIWARE Data Space Connector combines EDC and FDC components supporting compatibility with other EDC-based connectors while bringing support to data exchange scenarios involving providers offering services beyond access to data (processing, visualisation) and users within consumer organisations (natural persons using their digital wallets, devices such as sensors or robots, or software agents, including AI agents).

- https://github.com/FIWARE/data-space-connector

All referenced repositories are released under permissive Apache-2.0 and CC-BY-4.0 licenses, enabling broad reuse and adaptation. The DEPLOYTOUR project will further contribute to this by publishing its own developed code and documentation in a dedicated public repository:

- https://github.com/DEPLOYTOUR/Technical-documentation

4.1 Integration strategy

The European Interoperability Test Bed (ITB) is a testing platform developed by the European Commission that supports the validation of interoperability and conformance for systems that need to communicate with each other. ITB allows defining test suites, executing message-exchange scenarios between components, validating data structures, and generating result reports, thus providing a controlled and reproducible environment for evaluating interoperability-oriented solutions.

In the context of DEPLOYTOUR, ITB will be used to define and execute a set of conformance tests for the DSP associated with the MVDS. This approach serves a dual purpose:

- To validate the implementation of the Data Space Protocol in the MVDS, ensuring its correct behaviour and interoperability within data spaces.
- To act as a reusable reference for third parties (tourism data providers and consumers) that wish to verify the compatibility of their connectors with DEPLOYTOUR, by executing the same test set and acceptance criteria.

Scope of verification and validation activities

The verification and validation activities foreseen in DEPLOYTOUR on the MVDS connector, in relation to the Data Space Protocol, aim to ensure its correct implementation, operation and compliance with requirements within a data space environment.

The scope includes DP conformance tests executed in ITB, structured around the three main subprotocols:

- Catalog Protocol (CP)
- Contract Negotiation Protocol (CNP)
- Transfer Process Protocol (TPP)

The following are explicitly out of scope for these activities: performance tests, advanced security, integration with external systems, and any functionality not expressly covered in the tests described.

4.1.1 Characteristics to be tested

Within DEPLOYTOUR, for the conformance testing of the MVDS connector against the Data Space Protocol, the following characteristics are considered necessary:

- Expected HTTP status codes: It is verified that each protocol operation returns the HTTP status codes defined in the specification, for both successful responses and error cases. This makes it possible to validate the correct interpretation of requests and the error handling in accordance with the protocol.
- Validation against JSON Schema: All exchanged messages (requests and responses) are validated against the JSON Schemas defined in the Data Space Protocol, thus ensuring that structure, data types and mandatory fields comply with the specification.
- Correct state transitions: It is verified that the connector correctly manages state transitions in the CP, CNP and TPP flows. Each state change is checked to guarantee coherence and the correct execution of the processes defined in the protocol.

The connector can be considered to operate in accordance with the definition of the Data Space Protocol when, for all defined scenarios, the expected HTTP status codes are obtained, the messages pass validation against the corresponding JSON Schemas, and the state transitions match those established by the protocol.

4.1.2 Test approach in ITB

Conformance tests of the MVDS connector will be executed through the graphical interface of the Interoperability Test Bed, using independent test scenarios that simulate end-to-end flows for each Data Space Protocol subprotocol. Each scenario is defined as an isolated test, specifying in its preconditions the required initial context.

The subprotocols and their main scenarios are as follows:

- Catalog Protocol (CP)
    - Successful request for the complete catalog without filter
    - Successful request for the complete catalog with filter
    - Request for catalog with invalid filter
    - Successful request for a complete dataset
    - Request for a non-existent dataset
    - Request for an empty catalog
- Contract Negotiation Protocol (CNP)

The scenarios focus on nominal flows initiated by the Consumer, as well as negotiation cancellation cases, including:

    - Complete and successful negotiation initiated by the Consumer
    - Negotiation cancellation transitions from different states, both on the Provider side (REQUESTED, OFFERED, ACCEPTED, VERIFIED) and on the Consumer side (REQUESTED, OFFERED, AGREED)
- Transfer Process Protocol (TPP)

The following scenarios are covered for pull and push transfers, as well as termination paths:

    - Successful pull transfer, including start, suspension, resumption and completion of the process

- Successful push transfer, including start, suspension, resumption and completion of the process
- Transfer cancellation transitions from the REQUESTED, STARTED and SUSPENDED states, both for the Provider and the Consumer

4.1.3 Resources and execution environment

The following resources will be used to carry out the interoperability tests in DEPLOYTOUR:

- Interoperability Test Bed (ITB): Test platform used for the design and execution of the Data Space Protocol conformance scenarios.
- State initialisation mechanism: Auxiliary component (for example, a service exposed via API or a set of data preparation scripts) responsible for configuring the database with the data required for each test and cleaning this state afterwards, ensuring independence between test cases. The concrete implementation of this mechanism will be defined during the development phase, according to the final MVDS architecture.
- MVDS connector: Connector under test in ITB, which implements the Data Space Protocol in the DEPLOYTOUR context.
- Execution environment: Server or infrastructure where ITB, the MVDS connector and the state-initialisation mechanism are deployed, including operating system and Java version, or other required runtimes, for their execution.

4.1.4 Evidence, traceability and reference for third parties

The test evidence will consist of the reports generated by the Interoperability Test Bed after executing all described scenarios. These reports record, for each test case, the exchanged requests and responses, the results of JSON Schema validations, the returned HTTP status codes and the final state of the processes.

The project will maintain a repository of reports and results associated with the MVDS, which will make it possible to:

- Demonstrate the conformance of the MVDS connector with the Data Space Protocol, in terms of HTTP status codes, message structures and state transitions.
- Facilitate the periodic re-execution of the tests, including the possibility of automating their execution via the Interoperability Test Bed API on preconfigured environments (projects, communities and test suites), which would allow integrating them, where appropriate, into the project's continuous integration (CI/CD) pipeline.

In addition, the test set described in this section is conceived as a reusable interoperability artefact:

- Any external organisation deploying a connector and wishing to ensure its compatibility with DEPLOYTOUR may execute on its own implementation the same CP/CNP/TPP scenarios described here.
- Successfully passing all tests with the expected results (HTTP status codes, JSON Schema validation and correct state transitions) provides an objective guarantee of technical compatibility with the MVDS.

4.2 Technical environment and deployment status

This section describes the technical environment and deployment setup of the Minimum Viable Dataspace (MVDS) prototype developed.

The objective of this deployment is to support a pilot installation focused on functional validation and integration testing, rather than on high availability or large-scale production workloads.

Figure 11 MVP technical environment

4.2.1 Runtime environment and orchestration

The MVDS is deployed as a containerised solution orchestrated with Kubernetes.

Given the nature of the MVDS:

- All components are deployed within a single Kubernetes cluster
- All services run inside the same Kubernetes namespace
- The Kubernetes cluster is intentionally small and lightweight, as no high-load or production-grade scalability is required at this stage

Kubernetes is used mainly to:

- Standardise deployment and configuration
- Simplify service networking
- Enable reproducible installations across different infrastructures
- Facilitate the transition towards more advanced deployments in future phases

4.2.2 Deployment Model

All components of the MVDS are deployed using Helm charts, which:

- Define Kubernetes resources in a consistent and reproducible way
- Centralise configuration values
- Simplify installation, upgrade and removal of the prototype

A single Helm-based deployment installs all components in the same namespace, ensuring simplicity and ease of operation for the pilot.

Only one environment is foreseen for this MVDS (pilot environment). There is no separation between development, staging or production environments at this stage.

4.2.3 Infrastructure and supporting services

The Kubernetes namespace contains the following groups of components:

- Authority services
  - authority-identityhub

  - authority-issuerservice
  - authority-federatedcatalog
  - authority-telemetryservice
  - authority-telemetrycsvmanager
- Provider services
  - provider-controlplane
  - provider-dataplane
  - provider-backend
- Consumer services
  - consumer-controlplane
  - consumer-dataplane

Additionally, the following supporting infrastructure components are deployed:

- PostgreSQL for persistent storage
- Event Hub / Kafka for event-based communication and telemetry
- Vault for secrets and sensitive configuration management

All these components coexist within the same Kubernetes namespace and are deployed as Kubernetes resources (Deployments, Services, StatefulSets, where applicable)

4.2.4 Networking, external access and observability

To support testing and validation activities during the pilot, the deployment requires:

- A DNS configuration to expose selected services externally
- A load balancer or ingress controller to provide access from outside the cluster

External access is required to:

- Invoke and test public APIs exposed by control plane and backend services
- Access dashboards (e.g. monitoring or telemetry visualisations)
- Access PostgreSQL and other services for validation and troubleshooting purposes

Ingress rules and service exposure are configured via Helm, and external endpoints are environment-specific.

4.2.5 Infrastructure Requirements

As this deployment targets a Minimum Viable Dataspace pilot, the infrastructure requirements are intentionally modest.

A typical deployment can run on a small Kubernetes cluster backed by virtual machines with the following minimum specifications:

Resource	Recommended specification
Compute	2–3 virtual machines
CPU	4 vCPUs per VM
Memory	16 GB RAM per VM
Storage	~70 GB persistent storage
Networking	External IP, DNS support, load balancer or ingress

Table 2 Infrastructure requirements

These requirements are compatible with any cloud provider (AWS, GCP, Azure, others) as well as on-premises virtualised environments.

Conclusions

This deliverable represents a key step in developing the ETDS Prototype, translating the conceptual architecture into its first functional implementation: the Minimum Viable Data Space (MVDS). This initial version focuses on the essential components and interaction flows needed to support secure, decentralised, and interoperable data sharing within the tourism sector.

Rather than aiming for completeness, the MVDS is designed to demonstrate the technical feasibility of the ETDS architecture. It establishes a foundation for integration and testing using open-source technologies and aligns with European frameworks such as IDSA, GAIA-X, and DSSC. The implementation prioritises modularity, reproducibility, and adherence to interoperability standards to ensure long-term scalability.

The MVDS is conceived as the starting point of an iterative development process that will evolve throughout the project. Future iterations will progressively enrich the system by integrating new components, refining the existing modules, and advancing semantic, functional, and governance capabilities.

In parallel, upcoming phases will also address the challenge of interoperability between different connector implementations, contributing to a more flexible and inclusive technical landscape. The potential alignment with complementary frameworks such as SIMPL will be assessed to strengthen convergence with emerging European interoperability infrastructures.

In this context, Deliverable D2.2 not only lays down the initial technical foundations of the ETDS but also sets the direction for its continued evolution across subsequent phases of DEPLOYTOUR.

Annex 1 – UI Navigation Structure

Home (Landing page)

Route: /

Authentication: not required.

Functionalities:

- Landing page with login button to initiate authentication.
- If already authenticated, automatically redirects to */catalog*

Federated catalog

Route: */catalog*

Authentication: Required (Login and route protection using an authentication guard).

Functionalities:

- Display all datasets available in the federated catalog.
- Search bar for filtering by name/description.
- Filter by participant (data provider) and status.
- Two view modes:
  - Table view: Sortable columns (Name, Participant, Content Type, Created, Updated, Contracts).
  - Card view: Visual grid with dataset cards showing name, description, provider, and dates.
- View dataset details (opens modal with full metadata).
  - Display comprehensive dataset metadata-
  - Title, description, and publisher information
  - Temporal coverage (creation date, last updated).
  - Keywords and categories.
  - Data format and content type.
  - License information.
  - List all associated contract offers with policy details.
  - Display provider connector information.
  - Action button to initiate contract negotiation.
  - Copy dataset/connector identifiers to clipboard.
- Navigate to existing contracts (if dataset already negotiated).
- Link to initiate contract negotiation from catalog.

Dataset Management Page

Route: */dataset-management/dataset*

Functionalities:

- List all assets/datasets published by the participant.
- Create new asset (opens modal form).
- Edit existing asset (opens modal form in edit mode).
- Delete asset (with confirmation dialog).
- View asset details (read-only mode in modal).
- Quick actions per asset (edit, delete, view contracts).
- Search and filter assets by name/properties.
- Pagination for large asset lists.

Asset details modal:

- Data Address Configuration:
  - Define where the actual data resides:
    - REST API endpoints
    - Database connections
    - S3/Object storage
    - File systems
  - Configure authentication for data access.
  - Set HTTP methods (GET, POST, etc.) and headers.
  - Define base URLs and paths.
  - Proxy configuration (if needed).
- Asset Properties:
  - DCAT metadata fields:
    - Title and description.
    - Keywords and categories.
    - Publisher information.
    - License and rights statements.
    - Temporal coverage (valid from/until).
    - Spatial coverage (geographic scope).
    - Data format and schema information
  - Custom properties: Key-value pairs for domain-specific metadata.
  - Access information: Content type, size, language.
- View Mode:
  - Read-only display of all asset information.
  - Copy identifiers and metadata to the clipboard.
  - JSON/JSON-LD representation view.

Policy Management Page

Route: */dataset-management/policy*

Authentication: Required (via AuthGuard).

Functionalities:

- List all ODRL policies defined by the participant.
- Create new policy (opens modal editor).
- Edit existing policy (opens modal in edit mode).
- Delete policy (with usage verification - warns if policy is in use).
- View policy details (read-only mode).
- Preview policy in ODRL JSON-LD format.
- Search and filter policies by name/type.
- Policy validation and syntax checking.

Policy details modal:

- Visual ODRL Policy Editor:
  - Define permissions:
    - Allow read/display.
    - Allow use/execute.
    - Allow distribution/transfer.
    - Allow modification/derivation.
  - Define prohibitions:
    - Prohibit commercial use.

        - Prohibit redistribution.
        - Prohibit modification.
    - Define obligations:
        - Attribution requirements.
        - Delete after X days.
        - Anonymisation requirements.
        - Notification requirements.
    - Set constraints:
        - Temporal: Valid from/until dates, usage duration limits.
        - Spatial: Geographic restrictions (e.g., EU only, specific countries).
        - Purpose: Allowed use cases (research, commercial, analytics, etc.).
        - Count: Maximum number of uses/accesses.
        - Attribute-based: Verify consumer credentials/attributes.
- View mode:
    - Display policy in human-readable format.
    - Show ODRL JSON-LD representation.
    - Syntax validation status.
    - Policy usage information (which contracts use this policy).

Contract Definition Management Page

Route: */dataset-management/contract-definition*

Authentication: Required (via AuthGuard).

Functionalities:

- List all contract definitions (Asset + Policy combinations).
- Create new contract definition (opens modal form).
- Edit existing contract definition.
- Delete contract definition (with confirmation).
- View contract definition details (read-only).
- See which assets and policies are linked to each contract.
- Search and filter contract definitions.
- Display eligibility criteria for consumers.

Contract Definition Form Modal:

- Select asset from dropdown (from published assets list).
- Select access policy from dropdown (defines who can access).
- Select contract policy from dropdown (defines usage terms).
- Set contract validity period (start/end dates).
- Define additional constraints.
- Preview resulting contract offer.
- Save contract definition to publish to catalog.

Contract Negotiation Page

Route: */contract-management/contract-negotiation*

Authentication: Required (via AuthGuard).

Functionalities:

- Display all contract negotiations (past and active)
- Show negotiation states with visual indicators:
    - REQUESTED - Initial request sent to provider.
    - OFFERED - Provider made a counteroffer.

  - ACCEPTED - Consumer accepted offer.
  - AGREED - Both parties agreed (contract valid).
  - VERIFIED - Contract verified.
  - FINALIZED - Contract finalised and ready for transfer.
  - TERMINATED - Negotiation cancelled/failed.
- Filter negotiations by state
- Search by provider name or asset name
- View detailed negotiation history (opens modal)
- Cancel ongoing negotiation
- Retry failed negotiations
- Initiate data transfer after agreement (button appears when state = AGREED/FINALIZED)

Contract Negotiation Details Modal:

- Display complete negotiated contract details.
- Show DSP message exchange history (timeline view).
- Display agreed ODRL policies with a human-readable explanation.
- Show timestamps for each state transition.
- Display cryptographic signatures and verification status.
- Show provider and consumer participant information.
- Display contract validity period.
- Copy contract agreement ID to clipboard.
- Export contract as JSON or PDF.

Contract Transfer Management Page

Route: /contract-management/contract-transfer

Authentication: Required (via AuthGuard).

Functionalities:

- List all data transfer processes (past and active)
- Show transfer states with visual indicators:
  - REQUESTED - Transfer requested.
  - STARTED - Transfer in progress.
  - SUSPENDED - Transfer paused.
  - COMPLETED - Transfer finished successfully.
  - TERMINATED - Transfer cancelled/failed.
- Display transfer progress:
  - Progress percentage.
  - Data size transferred / total size.
  - Transfer speed (MB/s).
  - Estimated time remaining.
- Filter transfers by state.
- Search by dataset name or provider.
- Actions per transfer:
  - Suspend ongoing transfer.
  - Resume suspended transfer.
  - Terminate/cancel transfer.
  - View transfer details (opens modal).
  - Download transferred data (if applicable, when COMPLETED).
- Monitor real-time transfer metrics.

Annex 2 – ETDS Prototype generic design principles summary table

Having outlined each principle in detail, it is helpful to recap how these generic design principles map to the concrete standards implemented through EDC and FDC components in the ETDS. The table below summarises this mapping:

Design Principle	Description / Rationale	EDC Support (Standards & Components)
Interoperability & Open Standards at the control plane	Use common protocols and vocabularies so all participants' systems interconnect seamlessly at the control plane. Ensures cross-sector compatibility.	*Data Sharing:* IDSA Dataspace Protocol (DSP) implemented by EDC; *Metadata:* W3C DCAT for catalogs; *Policy:* W3C ODRL for usage rules.
Data Sovereignty & Trust	Participants retain control over data sharing; trust is established via verified identities and agreements.	*Identity:* W3C Decentralised Identifiers (DID) and Verifiable Credentials (VC) for self-sovereign identity (EDC Identity Hub, DID Registry); *Trust:* Gaia-X Trust Framework compliance (Self-Descriptions as VCs); *Auth:* OID4VC and DCP for issuance and exchange of VCs. Data sovereignty: enforcement of ODRL-based access and usage control policies.
Federated & Decentralized Architecture	No central data repository – connectors at each participant enable direct peer-to-peer data exchange; minimal central facilitation.	*Connectors:* EDC-based Control Plane at each connector; *P2P Negotiation:* DSP contract negotiation protocol; *Optional Federation Services:* EDC Federated Catalog, Registration Service for onboarding. Follows IDS' "decentralisation by default" ethos.
Modularity & Extensibility	Architecture is composed of distinct building blocks that can be upgraded or replaced independently. Easy to add new capabilities.	*Modular Components:* Connector (control plane), Data Plane, Catalog, Identity services, Policy Engine, Audit, etc., each as separate modules; *Extensions:* EDC's plug-in architecture allows adding new protocols, data types, backends without core changes.
Security & Policy Enforcement	Secure data sharing by design and enforce strict usage control as per contracts. Protect data integrity and confidentiality.	*Policy Enforcement:* EDC and FDC Policy Engines evaluating ODRL rules and participant attributes for every request; *Secure Comm:* TLS encryption, mutual authentication via DIDs/VCs; *Audit Logs:* Immutable logging of all actions (EDC Audit extension planned).

Scalability & Resilience	Handle growth and component failures gracefully while ensuring high availability and performance.	*Cloud-agnostic Deployment:* EDC and FDC components run on any cloud or on-prem, enabling horizontal scaling. Resilience Patterns: Retry/circuit breaker logic in EDC and FDC components. Observability: Telemetry via OpenTelemetry for monitoring health. "Design for failure": decoupled services and failover strategies.

Table 5. Design principles

Each of these principles has been incorporated into the ETDS prototype's design to ensure that the resulting system is not only aligned with European best practices for data spaces but also robust, adaptable, and trustworthy for all stakeholders. By leveraging open-standard components (primarily EDC, with the option to integrate FDC), the ETDS team accelerates development while upholding all these design principles. In practice, the EDC-based stack serves as the initial technical backbone supporting interoperability, data sovereignty, and modularity (in line with IDS-RAM and Gaia-X guidelines), with FDC components considered as complementary enhancements to reinforce these qualities without introducing technology lock-in.