Journal of Geographic Information System, 2026, 18(4), 171-195
https://www.scirp.org/journal/jgis
ISSN Online: 2151-1969
ISSN Print: 2151-1950

Towards Scalable and Interoperable Geospatial Vector Data Federation via a Decentralized Canonical Metadata-Driven Model

Landry Engelbert Tongo[1], Eric Fotsing[2], Georges Edouard Kouamou[3]

[1]Laboratory of Image Processing for Stereo-Restitution, National Institute of Cartography, Yaounde, Cameroon
[2]Department of Computer Engineering, Fotso Victor University Institute of Technology, Bandjoun, Cameroon
[3]Department of Computer Engineering, National Polytechnic High School of Yaounde, Yaounde, Cameroon
Email: ltongo@yahoo.fr, efotsing@gmail.com, georges_edouard@yahoo.com

How to cite this paper: Tongo, L.E., Fotsing, E. and Kouamou, G.E. (2026) Towards Scalable and Interoperable Geospatial Vector Data Federation via a Decentralized Canonical Metadata-Driven Model. *Journal of Geographic Information System*, **18**, 171-195.
https://doi.org/10.4236/jgis.2026.184010

Received: May 4, 2026
Accepted: July 25, 2026
Published: July 28, 2026

Abstract

In complex and distributed environments, geospatial vector data is vital for planning, analysis, and decision-making. Yet centralized architectures often limit data producers autonomy, create scalability challenges, and hinder seamless sharing across institutional boundaries—particularly in heterogeneous systems with varying metadata standards. Existing spatial infrastructures are largely optimized for both raster and vector data and rely on centralized repositories or tightly coupled systems. These approaches provide limited support for decentralized governance, flexible metadata management, and federated access to vector datasets, which are crucial for autonomy and interoperability. This study introduces a metadata-driven model to federate access to distributed geospatial vector data while preserving contributor sovereignty. The model consists of three layers: 1) Presentation Layer—offers user interface tools for querying and visualizing geospatial data. 2) Semantic Layer—manages a canonical metadata schema replicated across stakeholders via a federated LDAP directory and distributed Directory Information Tree, ensuring consistent yet autonomous governance. 3) Federation Layer—enables remote data retrieval through an access engine, APIs, and GeoJSON formatting without enforcing global standardization. The system supports flexible mappings for non-core metadata fields, avoiding data loss and allowing local customization. This model advances spatial data infrastructures by enabling decentralized, scalable, and interoperable sharing. It reduces reliance on centralized map servers, mitigates synchronization latency, and facilitates conflict resolution across domains. Most importantly, it fosters collaborative environments where producers retain control of their assets while contributing to unified decision-support systems. The model provides a foundation for open, feder-

ated platforms in environmental monitoring, urban planning, and smart governance.

Keywords

Canonical Metadata Model, Geographical Federation, Vector Data, GeoJSON, Spatial Data Infrastructure, Repository

1. Introduction

The growing importance of geographic information in economic development and territorial planning has prompted public and private organizations to produce and share an ever-increasing volume of spatial data. These datasets, whether raster or vector formats, are used in spatial analysis to represent continuous phenomena such as land use, precipitation patterns, or elevation gradients. Consequently, they contribute directly to the generation of geographic insights in sectors ranging from thematic mapping and urban planning to environmental management and transportation logistics.

However, unlike raster data—which can implicitly contain rich geographic information even when taken in isolation—vector data do not possess this intrinsic informational capacity. As a result, vector datasets must be systematically combined either with raster data or with other vector data to produce actionable geographic information for decision-making purposes. This interdependency drives operational stakeholders to seek access to complementary vector data that they neither produce nor own. For instance, a road infrastructure manager requires vector data about another entity's underground electrical network to safely plan interventions on their own assets.

Due to the high cost of producing vector data and the overlapping interests of multiple professionals operating within the same geographic areas or thematic domains, cooperative acquisitions of vector datasets have become increasingly common. These collaborations are typically formalized via data federation platforms, often led by dominant actors who deploy the underlying infrastructure that other stakeholders subsequently join [1].

This conventional federation model relies on a centralized architecture comprising three functional layers:

- A Data Ingestion Layer that handles the import, conversion, and integration of geospatial data from multiple formats and sources into a central repository.
- A Metadata Management Layer, considered the backbone of the system, that ensures the storage, indexing, and accessibility of metadata.
- A User Access and Visualization Layer that provides an intuitive interface and services for interacting with data and metadata.

Despite its structured nature, this centralized model presents inherent limitations. Once a dataset is integrated into the system, it becomes difficult for the original producer to retain control over its use and updates. In today's landscape, con-

tributors increasingly express the desire to maintain authority over the datasets they produce and manage—especially since they possess the domain expertise required to ensure proper versioning and maintenance. The lack of an automated update mechanism in centralized repositories often leads to degraded data quality and diminished traceability. Furthermore, data sovereignty is compromised.

In the case of raster tile sharing platforms, major providers such as Google, Bing, and Esri illustrate alternative models of control by employing access tokens—not only to regulate usage rights but also to monitor how their data is consumed [2] [3].

This paper proposes a metadata-driven federation model for geospatial vector datasets that preserves the independence and control of each contributor. The goal is to design a model that enables stakeholders to share access to their vector geographic data while retaining oversight of how those datasets are utilized.

The rest of this paper is organized as follows. Section 2 reviews existing literature and provides a comparative analysis of metadata-driven cataloging architectural frameworks. Section 3 presents in detail the proposed metadata-driven model. Its first sub-section includes a description of the model via a conceptual diagram and the second subsection deals with technical implementation aspects. In section 4, the governance and management aspects of the federation platform model will be addressed. Section 5 describes the model implementation, highlighting the main results. Section 6 is devoted to discussions on the relevance of the model. Finally, section 7 draws conclusion and outlines perspectives of our work.

2. Comparative Analysis of Metadata Cataloging Models

2.1. Reviews Existing Literature and Limit

In a context where public and private organizations manage large volumes of geographic data—stemming from cadastral records, urban infrastructure mapping, or environmental studies—the high acquisition costs have encouraged the development of collaborative resource strategies. Among these, data federation has emerged as an innovative approach for ensuring interoperability and enabling real-time information access [4].

Federation of vector geospatial data aims to provide unified access to distributed datasets while maintaining the heterogeneity and autonomy of the individual sources. In contrast to centralized architectures, federated models preserve the independence of contributing systems while facilitating efficient querying and transparent data integration [5] [6].

The core objective of integration within a federated architecture is to abstract the origin of the data sources. In practice, this involves retrieving datasets through a metadata-driven search engine [7], where the metadata descriptors supply the semantic and structural information necessary for rendering and utilizing spatial content.

Certain ingestion engines further enhance the integration process by transparently ingesting data into a central repository—without compromising the auton-

omy of contributing sources. Building on this foundation, according to [8], a federation model is aligned with the federated database paradigm, which enables tight coupling of data access mechanisms while preserving the operational independence of the source databases. Figure 1 illustrates the structural components of this architecture.

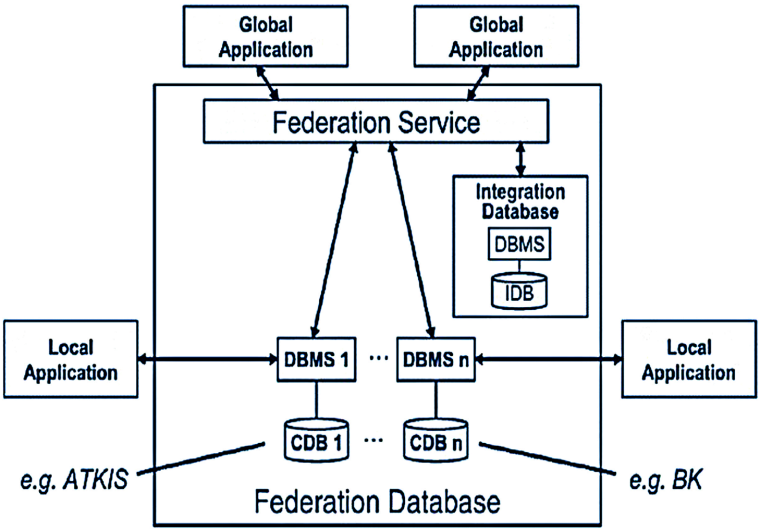

Figure 1. Geographical data federation model using data ingestion [8].

In this configuration, a service federation engine is tasked with integrating data into the unified schema of the central database. To achieve this, it establishes connections to remote databases, enabling transparent integration of pre-identified datasets into the central repository. Within such an environment, the targeted data are already known and explicitly defined, which explains the absence of metadata in the retrieval process.

Conversely, in mainstream geospatial data federation solutions—such as GeOrchestra, GeoNetwork, GeoNode, EasySDI, among others—metadata play a critical role in supporting searchability and facilitating data integration. These platforms leverage metadata descriptors to enable semantic interoperability, improve discovery mechanisms, and streamline access to heterogeneous geospatial resources (Figure 2).

In this type of architecture, users access metadata through a public interface that allows them to refine queries and retrieve relevant information. Leveraging metadata descriptors, the system performs ingestion-based retrieval from a central database. Visualization of the resulting datasets is made possible via map servers and standardized Open Geospatial Consortium (OGC) services (e.g. Web Map Service (WMS) and Web Feature Service (WFS)) [9].

In the federation architectures illustrated in Figure 2, the metadata cataloging models employed are those defined by the OGC, namely Catalogue Services for the Web (CSW) or SpatioTemporal Asset Catalogs (STAC). CSW is a widely adopted standard that enables the publication and retrieval of metadata describing

geospatial data and services [10]. These metadata records can be configured according to various existing standards (ISO 19115, Dublin Core, etc.). CSW provides strong interoperability and facilitates resource discovery, but it is sometimes affected by complexity and limited performance [11]. CSW catalogues can be integrated into distributed architectures, particularly to federate multiple catalogues (**Figure 3**). However, natively, CSW does not provide modules for replication across nodes within a geospatial data federation. Implementing a catalogue federation base on CSW, is highly complex, requiring advanced knowledge of XML standards, and the metadata models employed must be identical, which restricts the autonomy of stakeholders within a federation. In cases where metadata models are not identical, an additional software layer must be constructed to ensure federation, which results in degraded performance and increased complexity in the implementation of a GeoFederation based on this catalogue.

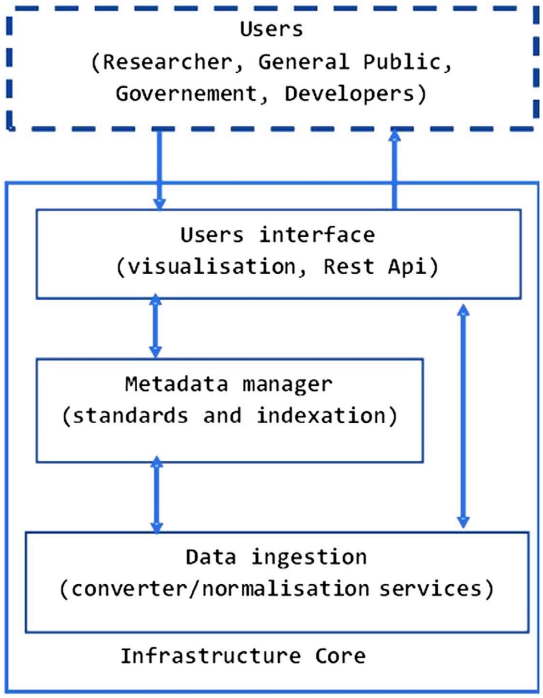

Figure 2. Architecture of the geospatial data federation.

CSW services are not optimized for highly heterogeneous environments; they require the creation of distributed catalogues linked to a central catalogue, which through a dedicated software layer, manages synchronization [11]. The content of the slave catalogues differs from one another, with homogeneity ensured by the central catalogue [12]. The master catalogue does not store the data itself. When a user initiates a query, the CSW server redistributes the request in real time to multiple nodes (slave catalogues) and aggregates the results. This centralized catalogue mode weakens the federation. Indeed, in the event of a crash of the central catalogue, restoration techniques for CSW catalogues are extremely complex to implement, since no version of the directory is available on any node.

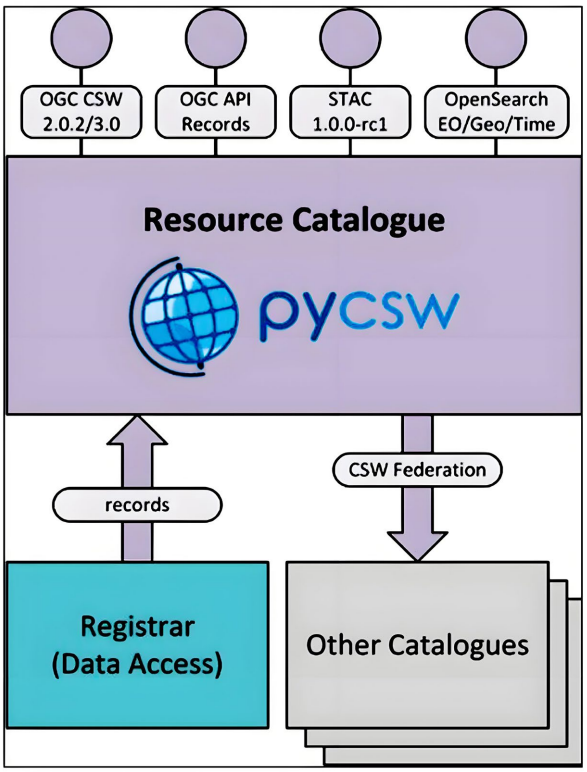

Figure 3. CSW catalogues federation example using pycsw.

STAC are an open specification, initiated in 2018 by Radiant Earth, that standardizes the description and organization of geospatial data (satellite imagery, observations, etc.) using JSON/GeoJSON. They are widely employed to facilitate the discovery and access of satellite data within cloud environments. Their primary advantage lies in interoperability and simplicity; however, challenges may arise due to the maturity of available tools and the management of very large data volumes [13]. STAC is primarily used for cataloging satellite datasets such as Landsat, Sentinel, MODIS, and others [14].

However, within this framework, data producers are required to upload both their metadata to a central metadata catalog and their datasets to a central database. This approach enables matching and extraction processes to operate on an integrated view of multiple databases through a unified schema. Consequently, such systems function as black boxes from the user's perspective—offering limited control to contributors over the management of their own data. This configuration is particularly suited to environments where institutions wish to make internally produced data publicly accessible, while also allowing third parties to integrate additional datasets under read-only conditions. In the specific case of the platform base on OGC standards, they are protected by a security proxy and a single sign-on authentication system. They provide access to a suite of independent, interoperable modules that users can selectively activate to assemble a customized Spatial Data Infrastructure (SDI) "à la carte" [15].

The geospatial data federation to be established is designed to leverage the ca-

pabilities of open-access tools, without requiring any additional code extensions. It does not fall within the scope of satellite imagery processing as defined in ISO 19115 and related standards. The federation model does not involve rebuilding software layers; rather, it exploits the inherent flexibility of the tools employed. While the OGC CSW specification provides recognized advantages in metadata discovery and catalog interoperability, its limited flexibility constrains its applicability to the proposed federation architecture. The approach adopted herein emphasizes compliance with OGC and ISO standards while ensuring operational simplicity and architectural resilience.

The previously described geospatial data federation architecture (**Figure 2**) is tailored for contexts in which institutions aim to disseminate their data through Geoportals. These models are typically implemented by a central entity that builds the entire infrastructure around its own datasets to ensure their accessibility and visibility. In such configurations, the central operator maintains full control—there is no cooperation, no collaboration, and particularly no exchange between domain actors in the production of geographic information.

To enable professionals to share data while retaining ownership and control, it is essential to federate their resources through a platform that facilitates mutual benefit from each other's datasets. This model allows contributors to remain autonomous while managing access rights and usage policies, thereby preserving independence within a collaborative framework.

2.2. Steps Underlying the Federation of Vector Geospatial Data

The main contribution of this paper is a decentralized alternative to centralized SDI architectures, with emphasis on interoperability, scalability, and data sovereignty.

The guiding premise of this work stems from the difficulty encountered by certain organizations in producing decision-support geographic information, from their own vector geospatial datasets due to the lack of complementary data provided by third-party producers. To address this challenge, it became necessary to establish a framework for data exchange and sharing among stakeholders.

In this context, metadata, which constitute a fundamental component of geospatial data sharing and interoperability, played a crucial role. Furthermore, to accommodate the possibility that participating organizations may adopt heterogeneous metadata standards and models, a set of canonical metadata was developed. These canonical metadata represent the common semantic core shared across the metadata schemas of all stakeholders involved in the data-sharing process [16].

Following the development of the canonical metadata model, and in order to facilitate data discovery and access, the metadata were integrated into an Lightweight Directory Access Protocol (LDAP)-based metadata directory. The use of a LDAP directory offers a significant advantage in decentralized environments through its native replication capabilities, allowing the directory to be distributed

and synchronized across all participating organizations involved in data exchange and sharing. This directory stores the essential information required for the discovery, access, and retrieval of vector geospatial datasets without data ingestion [17].

The canonical metadata model and the associated directory constitute the foundation of the federation model proposed in the following sections. Together, they provide a distributed mechanism for metadata management, resource discovery, and sharing.

3. Design of the Metadata-Driven Model for Geographic Vector Data Federation

The model of the geospatial data federation will be introduced in the first subsection, followed by the technical components of the geofederation in the second sub-section and at the last sub-section, a formal technical architectural specification of all components required for its implementation.

3.1. Geofederation Model

Vector geospatial data federation refers to the ability to dynamically query and integrate vector datasets originating from disparate sources without requiring their physical replication. By allowing access to data distributed across multiple servers and systems, this method enhances operational flexibility and enables near-instantaneous updates—an essential feature for high-precision geospatial analyses.

The GeoFederation model (**Figure 4**), comprises three cores layers: the presentation layer, the semantic layer, and the federation layer.

- Presentation Layer, interfaces directly with users and data producers. It is the access point through which stakeholders connect to the platform and execute their operations. It consists of two components:
  - a query engine, responsible for executing all search-related tasks;
  - a cartographic visualisator, that enables the rendering of geospatial layers resulting from queries.

  The presentation layer serves a dual function:
  - it allows users/producers to search for metadata associated with geographic data via the query engine, which communicates with the semantic layer to process the requests;
  - it displays the results of those queries using the cartographic visualisator, offering a spatial representation of retrieved data layers.
- Semantic Layer, is the pivotal layer. It stores metadata, enabling users/producers to publish detailed descriptive information about their datasets while retaining full control over access and availability. All stakeholders register metadata to facilitate both semantic description and structured access to their resources. Metadata submission is user-driven—contributors retain control and may withdraw access at any time by deleting the related metadata. The

semantic layer interacts with the presentation layer through the query engine and relies on a dedicated component, the metadata catalogue, to handle the creation, registration, update, and deletion of metadata records within a centralized directory.

- Federation Layer, acting as the engine of the model, handles the retrieval of geospatial data identified during search queries. It comprises the federation engine—a component capable of interfacing with various types of remote geographic data sources, including servers and databases. Upon establishing access, the federation engine retrieves datasets requested by the query engine and transmits the necessary parameters to the presentation layer for rendering via the cartographic visualisator. This modular data flow ensures seamless, federated integration without compromising system autonomy or real-time responsiveness.

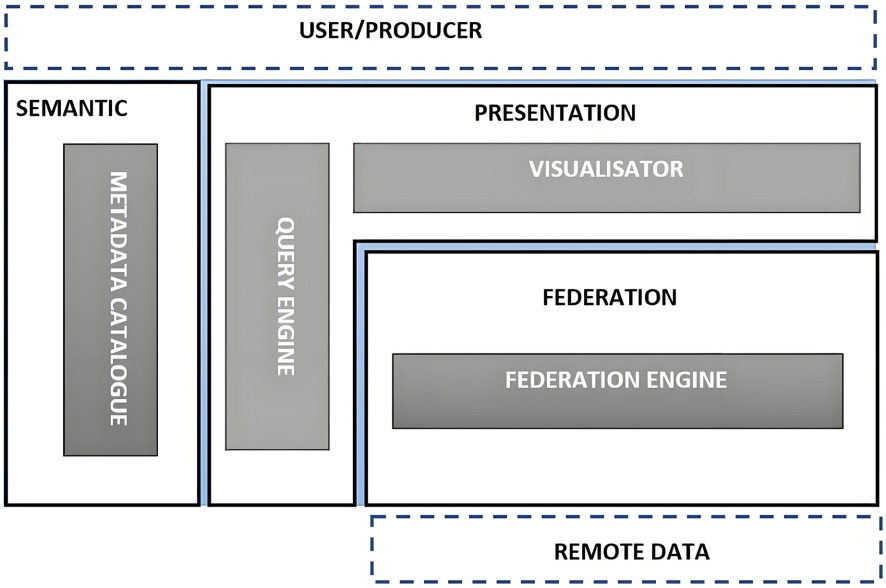

Figure 4. GeoFederation model.

3.2. GeoFederation Components

The federation approach is based on the use of open standards for accessing vector data and the GeoJSON format for their transmission. The use of standardized protocols, combined with the use of metadata compliant with ISO 19115 specifications, facilitates the indexing and discovery of spatial resources. According to [18], the structuring of metadata is essential to ensure effective interoperability between geospatial information systems. Thus, to maintain the independence of our platform model, the different stakeholders are not required to standardize the type of metadata standard they use. A key innovation of the semantic layer is its use of a canonical model, which abstracts away the heterogeneity of contributor metadata standards. This design lowers the barrier to entry, as participants are not forced into a costly and time-consuming migration to a single, rigid standard.

Although the ISO 19115 standard is becoming established and is increasingly

used as a standard, some professionals in the field still widely use standards such as: CSDG metadata, ENV 12657, Dublin Core, etc…. This situation is not an obstacle for the model.

In fact, at the level of the metadata catalog, a canonical standard may be implemented. It is a standard that brings together the core elements shared across various metadata schemas and serves as an integrative reference point. According to [16], a canonical standard can be built around several standards to make a unique standard and be used for an appropriate federation architectures.

Concerning storage's structure, a LDAP directory is used. According to [19], LDAP-based directories are well-suited for storing metadata in the context of geospatial data federation, offering both scalability and interoperability across distributed systems (**Figure 5**).

A LDAP directory service enforces catalog consistency across all Directory Information Tree (DIT) nodes through local replication of a central directory instance, as specified in RFC 4511 [20]. This replication mechanism enables query operations to be executed against the local replica, thereby minimizing network latency and optimizing search performance. In accordance with the replication and fault-tolerance properties defined in RFC 4512 [21], if a directory node or the central directory server experiences a failure, the directory service can be reconstructed from any available replica node. This intrinsic behavior of LDAP directory services ensures architectural stability, service continuity, and resilience within the federation.

```
dn: cn=Topographic_Map_NIC,ou=GeospatialData,dc=example,dc=com
objectClass: top
objectClass: geospatialMetadata
cn: Topographic_Map_NIC
title: Topographic Map 1:25 000 - NIC
identifier: INC-25K-2025
abstract: Detailed map of Cameroon with contour lines and infrastructure.
dateStamp: 2025-03-06T10:00:00Z
metadataStandardName: ISO 19115
metadataStandardVersion: 2003/Cor.1:2006
language: french
characterSet: utf8
topicCategory: topography
spatialResolution: 25
extentBoundingBox: 2.0, 41.0, 9.6, 51.0
verticalExtentMinimum: 0
verticalExtentMaximum: 4800
referenceSystemInfo: EPSG:4326
contactName: Institut National de Cartographie (INC)
contactEmail: contact@inc.cm
contactRole: distributor
dataFormat: GeoTIFF
distributionUrl: https://data.inc.cm/map/25k.tiff
lineage: Data collected through field surveys and satellite imagery.
useConstraints: Accès restreint aux institutions publiques
accessConstraints: Licence NIC
```

Figure 5. Example of inserting a metadata set based on ISO 19115 into a LDAP directory.

In this context, [17] implemented a canonical metadata catalog on a LDAP di-

rectory based on the approach of [16] (**Figure 6**). Our metadata catalog, manager of the central directory built in accordance with the specifications developed by [17].

```
The following language's grammar is established
starting from the syntactic tree:
Let ω1 be the set of non final symbols of ISO
Let ω2 be the set of non final symbols of CSDG
Let ω3 be the set of non final symbols of Dublin Core
Let be the set of non final symbols of Θ, Ω= ω1 ∩ω2 ∩ω3
Γ is the set of finals symbols of Θ.
Γ= {real, integer, text, date, time}
It rises directly from what this precedes; that the total
alphabet of the language is: Φ= Ω u Γ
Let R be the set of grammatical rules, or productions;
each production will be form: l → r where:
l ϵ Φ* is called head or left member and
r ϵ Φ* is called body or right member.
Φ* is the set of all the formed sequences of vocabulary
symbols of Φ, including the null string ε.
The head α contains at least a non final symbol.
α ϵ Ω constitutes the axiom or starting symbol, here one
has α = metadata. One will have:
Metadata := Idinfo | dataqual | spdoinfo | spref | distinfo
| ε
dataqual:= attracc | logic | complete | posacc | lineage
attracc:= attraccr | qattracc
attraccr:=text
qattracc := attraccv | attracce
attraccv:= text
Finally the grammar is Θ = { Ω, Γ, R, metadata}. The
grammar Θ, is the grammar of the canonical language of
handling, interrogation and storage of metadata.
```

Figure 6. An example of construction of a canonical metadata language model [16].

Although ISO 19115 tends to establish itself as the dominant reference, there remains a wide array of metadata standards currently in use. Standards such as the Content Standard for Digital Geospatial Metadata (CSDGM), Dublin Core, and others continue to be extensively adopted in contemporary practice. From these three standards, a canonical standard is produced according to [16]. The objective of the canonical metadata is to establish a common interoperable core between ISO 19115 (geospatial), CSDGM/FGDC (geospatial), and Dublin Core, thereby enabling stakeholders to achieve mutual understanding and semantic alignment. On the basis of [16], the canonical standard items are specified below (**Table 1**):

Table 1. Structuring elements of canonical metadata.

ISO 19115	CSDGM (FGDC)	Dublin Core	Canonical Field
MD_Metadata.fileIdentifier	Identification_Information.Citation.Online_Linkage	dc:identifier	Identifier
CI_Citation.title	Identification_Information.Citation.Title	dc:title	Title
MD_DataIdentification.abstract	Identification_Information.Description.Abstract	dc:description	Description
MD_Keywords.keyword	Identification_Information.Keywords	dc:subject	Keywords
CI_ResponsibleParty.role=originator	Data_Set_Credit/Originator	dc:creator	Creator
CI_ResponsibleParty.role=publisher	Publisher	dc:publisher	Publisher
CI_Date.dateType=creation	Time_Period_of_Content.Begin_Date	dcterms:created	Date created
CI_Date.dateType=publication	Publication_Date	dcterms:issued	Date issued
MD_Metadata.dateStamp	Metadata_Reference_Information.Metadata_Date	dcterms:modified	Date modified
MD_ScopeCode/hierarchyLevel	Data_Set_Type	dc:type	Resource type
MD_LegalConstraints	Use_Constraints/Access_Constraints	dc:rights	Rights
EX_GeographicBoundingBox	Spatial_Domain.Bounding_Coordinates	dcterms:spatial	Spatial coverage
MD_DigitalTransferOptions.onLine	Distribution_Information.Online_Option	dcterms:accessRights/dcterms:identifier	Distribution link

3.3. Technical Architecture of GeoFederation

The semantic layer of our model is instantiated as a LDAP directory, where metadata inputs are provided directly by platform stakeholders. The latter are not forced to modify or update their metadata standard, since the system allows them to provide the essential elements for semantic and structural harmonization to make the data directly usable by different users.

We take advantage of the benefits of using the LDAP DIT which can be replicated between the local servers of the stakeholders. While not traditionally used for high-velocity data, LDAP's mature replication capabilities and hierarchical structure are uniquely suited for the 'read-mostly, write-infrequently' nature of foundational geospatial metadata. This choice prioritizes directory consistency and contributor autonomy over the raw write-throughput of a conventional database.

In the GeoFederation model, all stakeholders participate in building the infrastructure by configuring the central directory and replicating it across their respective nodes, followed by the deployment of the platform for shared access. All participants operate at the same decision-making level, and each retains control over their own data, since only metadata are recorded in the directory. In the event of a local directory crash, it can be easily reconstructed from the central directory; the same holds true for the central directory itself. Unlike centralized models, datasets are not ingested into a central database, nor is a map server employed for visualization.

Within the GeoFederation platform, search operations are executed locally, which significantly reduces execution time (Figure 7).

This reinforces the independence of the stakeholders in the platform because each of them can have a copy of the metadata catalog locally, and thereby exercise granular control over their data federation responsibilities.

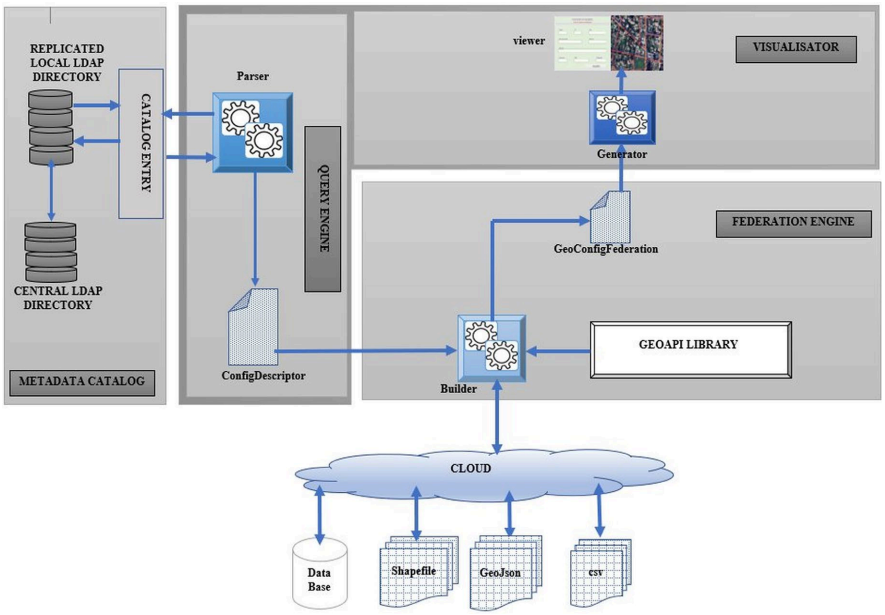

Figure 7. GeoFederation technical architecture.

The query engine serves as a parsing module that enables targeted searches within the metadata directory based on the input provided via the search form. Leveraging its parser, it analyzes user-submitted data and executes queries to retrieve records that match specified search criteria. Connectivity is handled through a Restful API interfacing with a LDAP directory backend. This setup offers seamless integration and enhanced security, particularly via HTTPS protocols. The interaction sequence is as follows:

- The search form transmits credentials, query strings, and other relevant inputs through an HTTPS request (POST or GET);
- The Rest API processes the request and communicates with the replicated LDAP directory (Figure 8), returning the appropriate response to the client interface.

The parser component facilitates querying operations within the replicated metadata directory and retrieves the result corresponding to the submitted request. Based on this output, the parser generates a configuration variable in Json format—designated as ConfigDescriptor (Figure 9)—which includes entries specific to the types of geographic datasets identified during the search process. This configuration variable encapsulates the parameters required to access the data whose metadata records are stored in the central directory. The configuration file provides connection attributes for the referenced datasets, which may include:

```
fetch("https://api.cnxldap.com/ldap/search", {
    method: "POST",
    headers: {
        "Content-Type": "application/json"
    },
    body: JSON.stringify({
        username: "usr",
        password: "pwd"
    })
})
.then(response => response.json())
.then(data => console.log(data))
.catch(error => console.error("Erreur:", error));
```

Figure 8. Rest API connection to the directory.

- A spatial database, denoted by the database entry;
- A GeoJSON file, represented by the GeoJSON entry;
- A Shapefile, mapped via the shapefile entry;
- A CSV file, associated with the csv entry.

```
{
  "database": {
    "type": "PostgreSQL",
    "host": "localhost",
    "port": 5432,
    "database_name": "data_base_name",
    "username": "user",
    "password": "pwd",
    "schema": "public",
    "table": "table_name",
    "ssl": false
  },
  "geojson": {
    "url": "https://example.com/data.geojson",
    "auth": {
      "type": "token",
      "token": "your_token"
    },
    "refresh_interval": 3600
  },
  "shapefile": {
    "url": "https://example.com/data_shapefile.zip",
    "files": {
      "shp": "data_shapefile.shp",
      "shx": "data_shapefile.shx",
      "dbf": "data_shapefile.dbf",
      "prj": "data_shapefile.prj"
    },
    "auth": {
      "type": "basic",
      "username": "user",
      "password": "pwd"
    }
  },
  "csv": {
    "url": "https://example.com/data.csv",
    "delimiter": ";",
    "encoding": "utf-8",
    "auth": {
      "type": "none"
    },
    "has_header": true
  }
}
```

Figure 9. Structure of a ConfigDescriptor variable.

The Federation Engine is the system component tasked, with retrieving geospatial data, whose access parameters are defined within the ConfigDescriptor. It acts as the orchestrator of data federation, ensuring stakeholder transparency and operational independence across distributed environments.

Authentication and authorization mechanisms are handled by the ConfigDescriptor and the federation engine. Sensitive connection credentials, such as secrets, passwords, access tokens, and other authentication parameters, are maintained within the metadata structure. This design facilitates secure and interoperable access to distributed geospatial resources across the federated nodes of the decentralized Spatial Data Infrastructure while preserving the autonomy of participating data providers. The Federation Engine comprises two main elements:

- A Rest GeoAPI library, enabling access to diverse remote data sources;
- A Builder, which functions as the core data-reading mechanism.

3.3.1. GeoAPI Library

The GeoAPI Library is responsible for establishing connections to remote data resulting from search operations. The technical specifications of the library that enable its implementation are:

- The data format being processed (Table 2)

Table 2. GeoAPI Library format structure.

Format	Functions	Answer
Postgres/Postgis	Connection parameter (user, pwd, bd, table, geometry column, attributs)	GeoDataSet
GeoJSON	Path file (.geojson, .json, auth)	
Shapefile	Path file [.shp (+ .dbf, .shx, .prj), auth]	
CSV	Path file (.csv, auth)	

- Main functions:
  - read_geojson(file_path, auth)
  - read_shapefile(file_path, auth)
  - read_cvs(file_path, auth)
  - read_postgres(usr, pwd, db, table, geometry column, attributs)
- The resulting GeoDataset returned by the GeoAPI Library in GeoJSON format is structured as follows:

GeoDataset:{
- features:{
- {list[Feature]}}
- crs:{ str}
- geometry_type:{str}
- attributes: {dict}
- bbox:{tuple}

},

- Error handling is carried out according to the following variables:

- FileNotFoundError: indicates that a file could not be located, most likely due to relocation or absence;
- UnsupportedFormatError: the stored data format is not recognized or supported by the GeoAPI;
- InvalidGeometryError: the recorded geometry does not exist within the database;
- MissingFieldError: the selected fields are not available in the dataset;
- ConnectionError: unable to establish a connection to the database.

3.3.2. The Builder Function

The Federation Engine does not rely on a map server or utilize conventional geospatial protocols such as WMS, WFS, or WCS. Its lightweight architecture favors decoupled, protocol-agnostic interoperability.

The ConfigDescriptor variable is exchanged between the Parser and the Builder through a RESTful interface using HTTP(S), with support for the POST and GET methods. At this level:

- The Parser generates the ConfigDescriptor;
- The Builder consumes it to instantiate the data federation orchestration process.

While iterating through each entry in the ConfigDescriptor, the Builder dynamically invokes the corresponding Rest GeoAPI—referenced in its internal library—to construct the GeoConfigFederation variable (Figure 10). This federated configuration includes individual entries representing the retrieved geographic datasets, all formatted in GeoJson.

```
def generate_geojson(ConfigDescriptor):
    """
    Generate a GeoJSON file .
    """
    GeoConfigFederation = {
        "type": "FeatureCollection",
        "features": []
    }
    # datat base
    if "database" in ConfigDescriptor:

GeoConfigFederation["features"].extend(fetch_from_database(ConfigDescri
ptor ["database"]))
    # remote GeoJSON
    if "geojson" in ConfigDescriptor  and "url" in GeoConfigFederation
["geojson"]:

GeoConfigFederation["features"].extend(fetch_from_geojson(ConfigDescrip
tor ["geojson"]["url"]))
    # remote Shapefile
    if "shapefile" in ConfigDescriptor  and "url" in ConfigDescriptor
["shapefile"]:

GeoConfigFederation["features"].extend(fetch_from_shapefile(ConfigDescr
iptor  ["shapefile"]["url"]))
    # remote CSV
    if "csv" in ConfigDescriptor  and "url" in ConfigDescriptor
["csv"]:

GeoConfigFederation["features"].extend(fetch_from_csv(ConfigDescriptor
["csv"]["url"]))
    return geojson_data
else:
        raise UnsupportedFormatError()
elseif
        raise FileNotFoundError()
```

Figure 10. Pseudocode for generating GeoConfigFederation files extracted from the ConfigDescriptor.

The Builder remains adaptable, it can be reconfigured to accommodate new APIs function added to the GeoAPI Library, ensuring scalability and long-term extensibility. The Federation Engine is designed for extensibility. Its reliance on a configurable API Library allows the federation to evolve organically, incorporating new data source types (e.g., cloud-native formats, streaming APIs) without requiring a redesign of the core architecture (Figure 11).

```
{
  "type": "FeatureCollection",
  "features": [
……..
    {
      "type": "Feature",
      "geometry": {
        "type": "Polygon",
        "coordinates": [[[long1, lat1], [long2, lat2], [long3, lat3], [long1, lat1]]]
      },
      "properties": {
        "source": "geojson",
        "url": "https://example.com/data.geojson",
        "auth_type": "token"
      }
    },
……..
    {
      "type": "Feature",
      "geometry": {
        "type": "Point",
        "coordinates": [csv_longitude, csv_latitude]
      },
      "properties": {
        "source": "csv",
        "url": "https://example.com/data.csv",
        "delimiter": ",",
        "encoding": "utf-8"
      }
    }
  ]
}
```

Figure 11. Structure of a GeoConfigFederation variable.

The Visualisator within the presentation layer is responsible for rendering geographic datasets derived from metadata-driven search operations. It integrates essential formatting and interaction capabilities, including spatial data manipulation functions such as zooming, panning, and layer toggling. This component comprises two cores modules:

- A Generator, which receives the GeoConfigFederation variable as input and iterates through its entries;
- A Viewer, tasked with displaying the resultant geographic layers.

For each entry in the GeoConfigFederation variable, the Generator dynamically produces a corresponding GeoJSON object with appropriate geometric descriptors. These objects represent individual geographic information layers sub-

sequently rendered by the Viewer. Consequently, if GeoConfigFederation contains n entries, the Viewer will display n distinct geographic layers. Communication between the Builder and the Generator is conducted via a Restful API over the HTTP(S) protocol, using POST or GET methods for data exchange. The Viewer module may be developed using JavaScript or through established cartographic libraries such as Leaflet, OpenLayers, or any equivalent GIS visualization tool compatible with the JavaScript ecosystem.

4. Governance and GeoFedaration Management

The establishment of a vector geographical data federation cannot be achieved without a clear definition of management rules. These rules enable effective administration of the platform without imposing a significant increase in workload on the stakeholders. They are articulated by addressing the following aspects: synchronization latency, conflict resolution, and data sovereignty versus operational overhead.

- Synchronization latency: In a geographically distributed federation, replication is rarely instantaneous. It is therefore possible that a user queries a local replica and receives a descriptor for data that has already been moved or deleted on the source node. This situation may lead to failed data retrieval and a degraded user experience. To mitigate this, the Builder who integrate GeoAPI library, incorporate error handling mechanisms that manage such cases during resource access. The system can simply indicate that the requested resource is unavailable, thereby preserving consistency in user interaction;
- Conflict resolution: To safeguard the integrity of the centralized catalog, update conflicts must be properly managed. The federation leverages the LDAP conflict resolution mechanism. For example, if two stakeholders concurrently modify the same metadata record on their respective local replicas, the LDAP directory applies version metadata, unique identifiers, and automatic priority rules, with the possibility of administrative intervention in cases of ambiguity. It should be noted that conflicts can only occur during the insertion of metadata into the directory, not within the underlying geospatial data itself;
- Data sovereignty vs. operational overhead: While replication of the central directory ensures control and sovereignty over metadata, it also introduces additional workload. If not properly managed, this overhead may lead to deterioration of the platform's performance. To address this, the builder is responsible for generating alerts related to directory maintenance. The implementation of such alert operations can be carried out in a consensual manner among the different stakeholders, ensuring both operational efficiency and respect for data governance principles.

5. Implementation and Results

5.1. Comparative Analysis of the GeoJSON Format and the WFS Protocol

This subsection discusses the benefits of the adopted technical architecture and

presents the experimental results obtained from the performance evaluation. The analysis focuses on two main aspects: i) metadata query latency within the directory service and ii) the efficiency of GeoJSON data dissemination compared with the WFS protocol. To this end, a performance evaluation is conducted by comparing the latency of geospatial data stored in a GeoJSON file with that of the same data retrieved from a PostgreSQL/PostGIS database through the Web Feature Service (WFS) protocol.

A dedicated algorithm (Figure 12) was designed and executed on a set of vector dataset composed of 1798 geographic features and a total of 3,465,451 vertices each. The dataset was imported into a PostgreSQL/PostGIS spatial database, while an equivalent copy was maintained on the server in GeoJSON format for comparative testing.

```
<script>
        async function testPerformance(url, label) {
        const start = performance.now();
        const response = await fetch(url);
        const data = await response.text(); // we read raw to measure the transfer
        const end = performance.now();
        const duration = (end - start).toFixed(2);
        return ` Latency time ${label} : ${duration} ms (size ${data.length} characters)`;
          }

        async function runTests() {
        const results = [];
        // URL GeoJSON (yourgeojson's file )
        const geojsonUrl = " https://192.168.10.102/mapserv/dba/mapfile/limite.geojson";
        // URL WFS (mapserver or other OGC service)
        const wfsUrl = "https://192.168.10.102/cgi-
        bin/mapserv.exe?&SERVICE=WFS&VERSION=1.1.0&REQUEST=GetFeature&TYPENAME
        =limite &OUTPUTFORMAT=geojson"
        results.push(await testPerformance(geojsonUrl, "GeoJSON"));
        results.push(await testPerformance(wfsUrl, "WFS"));
        document.getElementById("result").textContent = results.join("\n");
          }
        runTests();

 </script>
```

Figure 12. Performance test script.

Regarding the metadata directory, metadata registration and search operations were simulated using a local replica of the Directory Information Tree (DIT). This experiment aimed to assess the responsiveness and scalability of metadata management processes within the directory infrastructure.

The outcomes of these experiments are presented and discussed in the following sections.

5.1.1. Metadata Query Latency

This represents the time elapsed from a user submitting a search query through the Presentation Layer to receiving a complete list of matching metadata results from the replicated LDAP directory. The advantage of using a DIT together with directory replication is that the search is executed locally on the replicated version of the directory within the node. This ensures a constant query response time and eliminates network latency. The only potential limitation lies in the network la-

tency associated with updating the replicated directories. Table 3 presents test results obtained using JMeter on a LDAP directory containing approximately one hundred thousand entries.

Table 3. Metadata query mode latency.

Query mode	Typical time	Limiting factors
Local replication	5 - 50ms	CPU, Indexation
Remote (via WAN)	100 - 500 ms	Network latency
Remote without replication	>500 ms	Depends on traffic and availability of master server

5.1.2. GeoJSON Payload Efficiency

The primary advantage of GeoFederation lies in minimizing server side operations, thereby significantly reducing server load. The platform enables client side processing, allowing end users to configure the symbology (e.g., color, size, and other visual parameters) of received vector geospatial data independently, without requiring server intervention. To achieve such flexibility, the most suitable data format is GeoJSON.

GeoJSON is an open format (standardized by the IETF in RFC 7946) designed to encode geographic structures in Json. It has become the reference format for exchanging and visualizing geospatial data on the web due to its simplicity and widespread adoption [22]. On the client side, GeoJSON offers several key characteristics:

- Web compatibility: directly supported by JavaScript mapping libraries such as Leaflet, Mapbox, and OpenLayers;
- Readability: plain text format, easy to understand and manipulate;
- Interoperability: widely adopted in REST APIs and web services;
- Performance: more compact than GML/XML, enabling faster transfer and parsing on the client side;
- Standardization: officially recognized by the IETF (RFC 7946), ensuring stability and broad adoption [23].

In contrast, platforms based on CSW catalogues generally rely on the WFS protocol. WFS, is a standard of the OGC, provides access to vector geospatial data on the web not as rendered images (as in WMS), but as manipulable features (points, lines, polygons with attributes). It is commonly used to query, download, and update geospatial datasets in GIS environments and web applications. However, WFS is verbose: each geometry is encapsulated within numerous XML tags, which increases data size. Unlike GeoJSON, WFS is executed server side, making it less suitable for lightweight applications. For this reason, WFS outputs are often converted to GeoJSON server side to simplify client usage. Moreover, WFS does not support client side symbology configuration.

To evaluate the performance of the proposed federation framework, response times were measured under varying payload sizes. Experimental datasets ranging from 100 to 2000 geographic features and geometric complexities ranging to

3,000,000 vertices per feature were used (Figure 12). The results indicate that response times remained below 300 ms for payloads up to 2 MB and increased linearly with payload size thereafter (Table 4). These findings demonstrate that the federation mechanism maintains acceptable performance despite variations in GeoJSON complexity and volume.

Table 4. Comparative analysis between GeoJSON and WFS.

Criterion	GeoJSON	WFS
Average file size	More compact (simple Json)	Heavier (XML/GML tags)
Requests per second (req/s)	Higher (better performance)	Lower (costly XML parsing)
Bandwidth impact	Low, suitable for REST APIs	Higher, increased network load
Client-side parsing	Fast (native JavaScript support)	Slower (requires XML parser)

Table 4 highlights that GeoJSON benefits from its simple and compact structure, which reduces the size of transferred data and accelerates client-side parsing. In contrast, the WFS protocol is more verbose, each geometry is encapsulated within numerous tags, increasing file size and slowing down transmission and processing. On a medium-bandwidth network (e.g., 10 - 20 Mbps), this difference results in an additional latency of approximately 200 - 500 ms for WFS compared to GeoJSON. In practice, for interactive web applications, GeoJSON is preferred as it provides a superior user experience. WFS, however, remains valuable in GIS environments where complex operations (e.g., transactions, OGC filters) are required. Independent tests conducted elsewhere have reported similar findings [24].

This is why, for the platform dedicated to the federation of vector geospatial data, the decision was made to adopt the GeoJSON format.

5.2. Implementation

The developed platform is grounded in the aforementioned model and designed to support institutions seeking to establish a decentralized community of data producers within a metadata-driven data federation framework (Figure 13).

The platform's interface is composed of three distinct modules:

- Metadata Registration Form enables individual data producers to input and submit metadata records for integration into the federation system;
- Interactive Map Viewer presents the visual output generated by the Viewer component, displaying geographic layers derived from metadata search operations;
- Geospatial Search Form facilitates user access to geographic datasets through targeted metadata queries and filtering criteria.

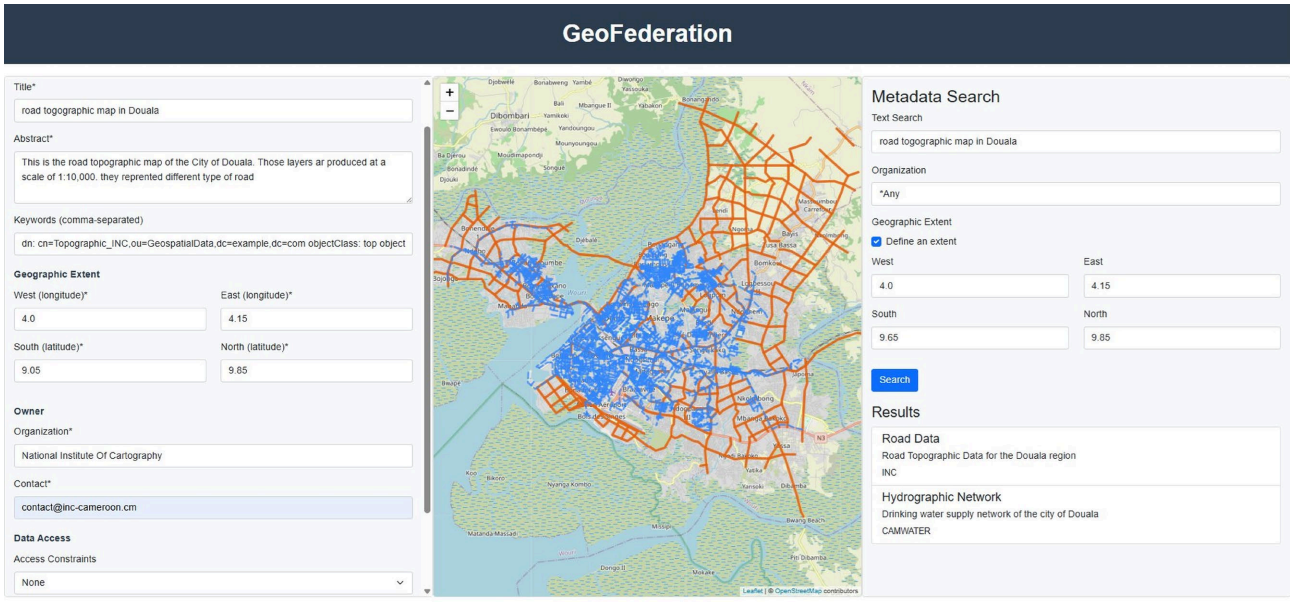

Figure 13. Geographical vector data federation platform user interface.

6. Discussions on the Relevance of the Model

The replication capabilities provided by LDPA directories at the local level have proven beneficial, while the use of Restful APIs guarantees reliable communication among the model's components. By standardizing component communication on a stateless restful architecture, the model ensures decoupled interoperability and minimizes the integration burden for stakeholders, allowing them to connect their data sources without adopting heavyweight geospatial protocols. We favor data formats such as Json and GeoJSON for their efficiency in directly exchanging geographic objects, semantics, and geometry. In contrast, protocols such as WMS and WFS typically require multiple connections to handle both aspects. Consequently, our architecture is particularly well adapted to settings where limited financial resources and a strong desire for independence drive stakeholders to build cohabitative frameworks for data production and sharing.

The implementation of the model in the development of collaborative geographical platforms contributes significantly to reducing investment costs associated with infrastructure deployment and ongoing operational oversight. The decentralized topology significantly lowers the total cost of ownership. By eliminating the need for a central entity to procure, manage, and scale a monolithic data warehouse and map server infrastructure, the model distributes operational costs in proportion to participation. Moreover, the model enhances principles of good governance by promoting transparency, traceability, and autonomy among participating stakeholders. The architecture operationalizes good governance. By keeping data at its source and providing contributors with direct control over their metadata entries in a replicated directory, the model ensures a transparent and auditable chain of custody, reinforcing accountability. It fosters an inclusive

framework in which control over data and decision-making processes remains distributed—encouraging accountability and sustainable collaboration across institutional boundaries.

7. Conclusion and Perspectives

The objective of our work is to propose a platform model based on metadata, which guarantees independence and control over the data by the actors.

An analysis of existing platforms has shown that their architecture is based on approaches that favor a model based on a central database. This model is not well-suited to environments where producers do not want to relinquish control of their data. Conventional centralized models create a custodial risk, where producers must relinquish direct control over their assets. Our federated approach, by contrast, establishes a non-custodial framework where data sovereignty is architecturally guaranteed, not merely promised.

Faced with the high costs of data acquisition, pooling efforts can be a suitable solution. The model has three data layers: the presentation layer, which is in contact with users, the semantic layer, which manages metadata, and the federation layer, which federates the data. The advantage offered by LDPA directories in terms of local data replication was useful. Then, the use of Rest APIs guarantees communication between the components of the model.

This is why the model is better suited to contexts where the scarcity of financial resources combined with the desire for independence pushes the different actors to develop solutions that allow them to coexist at the level of production and sharing of data. GeoFederation is strategically positioned for collaborative ecosystems, such as academic consortia or regional governments, where a mandate for data sovereignty and constraints on central infrastructure investment preclude the adoption of traditional, centralized SDI models. This coexistence can evolve into a community for sharing and exchanging geographic data around a federation that guarantees the independence of each party. The federation model we propose gives these stakeholders this opportunity.

The advantage of using data in JSON and GeoJSON format is their ability to exchange directly geographic objects, semantics and geometry. The model's reliance on GeoJSON as the primary data exchange format is a strategic decision to maximize web compatibility and processing efficiency. Unlike OGC services that can separate geometry from attributes, GeoJSON provides a complete, self-contained feature representation in a single payload, simplifying client-side rendering. We look forward to proposing this model in context where producers are jealous of their data and always want to keep control. In this perspective, implementation will target development projects operating in regions where similar conditions prevail as introduced above.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Janowicz, K., Scheider, S., Pehle, T. and Hart, G. (2012) Geospatial Semantics and Linked Spatiotemporal Data—Past, Present, and Future. *Semantic Web*, **3**, 321-332. https://doi.org/10.3233/sw-2012-0077

[2] Fernandes, E., Jung, J. and Prakash, A. (2016) Security Analysis of Emerging Smart Home Applications. 2016 *IEEE Symposium on Security and Privacy* (*SP*), San Jose, 22-26 May 2016, 636-654. https://doi.org/10.1109/sp.2016.44

[3] Tyler, M. (2005) Web Mapping Illustrated: Using Open Source GIS Toolkits. O'Reilly.

[4] Stefan, C. (1997) Föderierte datenbanksysteme. Springer.

[5] Paul, M., Ghosh, S. and Acharya, P. (2006) Enterprise Geographic Information System (E-GIS): A Service-based Architecture for Geo-spatial Data Interoperability. 2006 *IEEE International Symposium on Geoscience and Remote Sensing*, Denver, 31 July-4 August 2006, 229-232. https://doi.org/10.1109/IGARSS.2006.63

[6] Goodchild, M.F. (2007) Citizens as Sensors: The World of Volunteered Geography. *GeoJournal*, **69**, 211-221. https://doi.org/10.1007/s10708-007-9111-y

[7] Foerster, T., Stoter, J. and Van Oosterom, P. (2012) On-Demand Base Maps on the Web Generalized according to User Profiles. *International Journal of Geographical Information Science*, **26**, 99-121. https://doi.org/10.1080/13658816.2011.574292

[8] Butenuth, M., Gösseln, G.v., Tiedge, M., Heipke, C., Lipeck, U. and Sester, M. (2007) Integration of Heterogeneous Geospatial Data in a Federated Database. *ISPRS Journal of Photogrammetry and Remote Sensing*, **62**, 328-346. https://doi.org/10.1016/j.isprsjprs.2007.04.003

[9] Open Geospatial Consortium (OGC) (2011) OGC Web Services Standard.

[10] Nebert, D., Whiteside, A. and Voges, U. (2007) OGC Catalogue Services Specification 2.0.2. Open Geospatial Consortium.

[11] Florczyk, A.J., Lopez-Pellicer, F.J., Nogueras-Iso, J. and Javier Zara-Zaga-Soria, F. (2012) Automatic Generation of Geospatial Metadata for Web Resources. *International Journal of Spatial Data Infrastructures Research*, **7**, 151-172.

[12] Granell, C., Díaz, L. and Gould, M. (2010) Service-Oriented Applications for Environmental Models: Reusable Geospatial Services. *Environmental Modelling & Software*, **25**, 182-198. https://doi.org/10.1016/j.envsoft.2009.08.005

[13] le Moigne, J., Smith, B.D., Rogers, L.J., Little, M.M., Ranson, K.J., Morris, R.A. and Oza, N.C. (2023) "What Now/What Next/What If" or NASA Earth System Digital Twins. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=11215672

[14] Konrad, E., Piechl, T., Paulus, G. and Wallner, A. (2025) Application of the Spatiotemporal Asset Catalog Specification for Open Government Data. *AGIT Conference*, *Issue* 1. *Shaping Geospatial Futures*, **1**, 104-109.

[15] geOrchestra. (2025) geOrchestra Architecture Read the Docs. https://georchestra-main-documentation.readthedocs.io/admin_guide/architecture/

[16] Tongo, L.E. and Kouamou, G.E. (2009) Building a Canonical Language for Distributed System Repository. 2009 *Fourth International Conference on Software Engineering Advances*, Porto, 20-25 September 2009, 173-178. https://doi.org/10.1109/icsea.2009.35

[17] Tongo, L., Kouamou, G. and Tchudjo, G.A. (2014) Une approche d'implémentation des dictionnaires de métadonnées pour la fédération de données géographiques multisource. *Revue Africaine de Recherche en Informatique et Mathématiques Appli-*

quées, **18**, 53-65. https://doi.org/10.46298/arima.1975

[18] Treguer, M. (2006) Techniques d'interopérabilité au service de l'intégration des données géographiques. La lettre du Sismer, IFREMER de Brest Technopole Brest Iroise.

[19] Ramroop, S. and Pascoe, R. (2001) Use of LDAP to Partially Implement the OGIS Discovery Service. *International Journal of Geographical Information Science*, **15**, 391-413. https://doi.org/10.1080/13658810110047230

[20] Lightweight Directory Access Protocol (LDAP): The Protocol. https://www.rfc-editor.org/rfc/rfc4511

[21] Lightweight Directory Access Protocol (LDAP): Directory Information Models. https://www.rfc-editor.org/rfc/rfc4512

[22] The GeoJSON Format. https://datatracker.ietf.org/doc/html/rfc7946

[23] GeoJSON Format—Explanations, Examples. https://www.infobelpro.com/fr/blog/format-geojson

[24] GeoServer WFS Performance Comparison. https://geops.com/en/blog/geoserver-wfs-performance-comparison