



Project title: All Data 4 Green Deal - An Integrated, FAIR Approach for the Common European Data Space

Project number: 101061001

Project Acronym: AD4GD

Type: HORIZON-AG - HORIZON Action Grant Budget-Based

Work program topics addressed: HORIZON-CL6-2021-GOVERNANCE-01

DELIVERABLE NO: D1.3
GD DATA SPACE CONCEPT

Due date of deliverable: 31/08/2023

Actual submission date: 15/10/2023

Version: 1.0

Main Authors: Raul Palma (PSNC), Karolina Dostania (PSNC), Bogusz Janiak (PSNC), Rob Atkinson (OGC), Piotr Zaborowski (OGC), Alejandro Villar (OGC), Francesca Noardo (OGC), Alba Brobia Ansoleaga (CREAF), Joan Maso Pau (CREAF)

**DOCUMENT METADATA**

Project number	101061001
Project title	All Data 4 Green Deal - An Integrated, FAIR Approach for the Common European Data Space

Deliverable title	GD Data Space Concept
Deliverable number	D1.3
Deliverable version	1
Contractual date of delivery	31/08/2023
Actual date of delivery	15/10/2023
Document status	Final
Document version	1.0
Online access	https://drive.google.com/drive/folders/1WWIBQRCss4U5LYk0jVrqCNqNhnDyWdBg?usp=drive_link
Dissemination	Public
Work package	WP1: Semantic Interoperability Space
Partner responsible	Poznan Supercomputing and Networking CENTER
Author(s)	Raul Palma (PSNC), Karolina Dostania (PSNC), Bogusz Janiak (PSNC), Rob Atkinson (OGC), Piotr Zaborowski (OGC), Alejandro Villar (OGC), Francesca Noardo (OGC), Alba Brobia Ansoleaga (CREAF), Joan Maso Pau (CREAF)
Editor(s)	Raul Palma
Reviewer(s)	Cross-collaborators (each section was reviewed by contributors of other sections)
EC Project Officer	Lara Congiu

Abstract	The AD4GD project will provide interoperability support for the heterogeneous and increasingly growing set of data and services available in the various action areas addressed by the GD (e.g., climate, energy, industry, agriculture and biodiversity), contribute towards the creation of an EU GD data space. This document will focus on the semantic interoperability building blocks enabling different systems to exchange data with unambiguous meaning, and to enable an integrated data access for the execution of advanced data analytics, to support the pilots. These building blocks include the semantic data model (AD4GD information model), the method
-----------------	---



	and tools for extracting, harmonising, and sharing data, complying with data privacy and security requirements, and the set of open and standardised APIs necessary to provide the core services of the AD4GD ecosystem, which are based on the AD4GD information model.
Keywords	Semantic interoperability, data management, data harmonisation, data integration, semantic data models, ontologies, APIs, linked data
Disclaimer	Views and opinions expressed in this deliverable are those of the author(s) only and do not necessarily reflect those of the European Union, the United Kingdom or Switzerland. Neither the European Union nor United Kingdom nor Switzerland can be held responsible for them



DOCUMENT VERSION HISTORY

Version history			
Version	Date	Modification reason	Modified by
0.1	05/08/2023	Initial version of the document & ToC	Raul Palma (PSNC)
0.2	20/08/2023	Inputs Section 1, 2	Raul Palma (PSNC)
0.3	10/09/2023	Inputs Section 3,4	Raul Palma (PSNC), Bogusz Janiak (PSNC)
0.4	20/09/2023	Inputs Section 4	Alba Brobia Ansoleaga (CREAF), Joan Maso Pau (CREAF)
0.5	02/10/2023	Inputs Section 1, 4, 5	Rob Atkinson (OGC), Piotr Zaborowski (OGC), Alejandro Villar (OGC), Francesca Noardo (OGC),
1.0	10/10/2023	Final inputs Section 1,2,3,4,5,6 and formatting	Raul Palma (PSNC), Karolina Dostania (PSNC)



ABBREVIATIONS

Abbreviation	Definition
AD4GD	AllData4GreenDeal, the current project
EU	European Union
EC	European Commission
IPR	Intellectual Property Rights
IoT	Internet of Things
AI	Artificial Intelligence
FAIR	Findable, Accessible, Interoperable and Re-usable
UN	United Nations
GEOSS	Global Earth Observation System of Systems
EOSC	European Open Science Cloud
API	Application Programming Interface
CitSci	Citizen Science
WP	Work Package
GDPR	General Data Protection Regulation
DPO	Data Protection Officer
DPIA	Data Protection Impact Assessment
DPA	Data Protection Authority
DCA	Data Controllorship Agreement
NIS	Network and Information Security Directive
DGA	Data Governance Act
WP29	Article 29 Working Party
EDPB	European Data Protection Board

**TABLE OF CONTENTS**

1	Executive summary.....	9
2	Introduction.....	9
2.1	Ad4Gd gd semantic interoperability data space concept.....	9
2.2	Purpose of the document.....	14
2.3	Deliverable structure.....	14
3	AD4GD information model.....	15
3.1	Overview.....	15
3.2	Model layers.....	15
3.2.1	General Implementation Approach.....	17
3.2.2	Core Meta-Model Layer.....	18
3.2.3	Cross-Domain Layer.....	19
3.2.4	Domain-Specific Layer.....	20
3.2.4.1	Vocabularies/models aligned.....	20
3.2.5	Pilot-Specific Layer.....	21
3.2.6	Metadata Layer.....	22
3.3	Semantic interoperability with other vocabularies.....	24
3.3.1	Essential variables.....	24
3.3.2	Climate & Forecast variables.....	28
3.3.3	Agrovoc.....	28
3.4	Application in Pilots.....	29
4	Methods and tools for data harmonisation and integration.....	31
4.1	Overview.....	31
4.2	Data harmonisation and integration Tools.....	33
4.2.1	Data Preparation and Integration Pipelines.....	33
4.2.1.1	Client Web Application.....	36
4.2.1.2	Extended GRLC service.....	39
4.2.1.3	SensorThings API Generation service.....	39
4.2.2	OGC Data Exchange Toolkit.....	40
4.2.3	Semantic Annotation of Tables and Its Relation with STA.....	42
4.3	Application in pilots.....	53
5	Green Deal Data Space APIs.....	57
5.1	Overview of spatial standards with APIs.....	59
5.2	APIs and semantics.....	64
5.3	Relation to other data spaces' Building Blocks.....	70



5.4	Application in pilots	71
6	Conclusion.....	74

TABLE OF FIGURES

Figure 1. Data spaces building blocks from OPEN DEI project. source: Design Principles for Data Spaces – Position Paper - https://design-principles-for-data-spaces.org/	10
Figure 2. Pay-as-you-go Data Management tiers adapted from [2]	12
Figure 3. High-level view of the AD4GD data space ecosystem	14
Figure 4. Overview of the layers of the first release of the AD4GD Information Model	17
Figure 5. Overview of the first release of the GDIM Cross-Domain ontology.....	20
Figure 6. Core classes of the GDIM Metadata Schema	22
Figure 7. Digital Content properties in the OIM Metadata Schema	23
Figure 8. Artefact and Endpoint properties in the OIM Metadata Schema	23
Figure 9. EV status in 2021 in the GEO SBAs in support of SDGs and in relationship with the DPSIR policy framework. Percentages represent level of development of EV classes from EO to indicators	24
Figure 10. Essential Agriculture Variables (EAVs) entry page in OGC Rainbow server (part 1)	25
Figure 11. Essential Agriculture Variables (EAVs) entry page in OGC Rainbow server (part 2)	26
Figure 12. Essential Biodiversity Variables (EBVs) entry page in OGC Rainbow server (part 1)	26
Figure 13. Essential Biodiversity Variables (EBVs) entry page in OGC Rainbow server (part 2)	27
Figure 14. Ecological Quality Ratio (EQR) of freshwater fish in OGC Rainbow server (part 1)	27
Figure 15. Ecological Quality Ratio (EQR) of freshwater fish in OGC Rainbow server (part 2)	28
Figure 16. Representation of global trends in biodiversity (BES-SIM AIM) dataset in GDIM	30
Figure 17. Representation of dataset of monthly mean water levels from 21.12.2013 (from the Berlin Senate "Wasserportal") in GDIM. The terms in orange are not yet in GDIM, and will become part of a pilot extension module.....	31
Figure 18. High level workflow for multisource data integration	31
Figure 19. Linked Data pipelines for data harmonisation and integration	32
Figure 20. DPI Web Service architecture.....	36
Figure 21. DPI GUI client application - configuration section	37
Figure 22. DPI GUI client application - settings section.....	38
Figure 23. DPI STA generation service entry point.....	40
Figure 24. Example pilot in STA generation service	40
Figure 25. High-level view of the data workflows that can be created with the OGC Data Exchange Toolkit	42
Figure 26. Air temperature definition in EnvThes Thesaurus. URL http://vocabs.lter-europe.net/EnvThes/22035	44
Figure 27. Celsius unit definition in QUDT units vocabulary. URL https://qudt.org/vocab/unit/DEG_C	45



Figure 28. TAPIS STA+ reader - exporting a dataset into a GeoJSON file	47
Figure 29. MiraMon Map Browser showing observations with linked units of measures definitions opened in different tabs	48
Figure 30. Example tabular data of complex observations	48
Figure 31. TAPIS STA+ reader - showing a table with links to propertyURL and unitMeasureUrl	52
Figure 32. Upload table as STA observations from TAPIS STA+ reader (submit)	52
Figure 33. Upload table as STA observations from TAPIS STA+ reader (complete)	53
Figure 34. Global trends in biodiversity (BES-SIM AIM) collection of observations (simulations)	54
Figure 35. Single observation (simulation) of the Global trends in biodiversity (BES-SIM AIM)	54
Figure 36. SPARQL constructs defining the transformation of GEO BON dataset into GDIM	55
Figure 37. Observation collection of monthly mean levels from 21.12.2013 from a particular lake and station .	56
Figure 38. Single observation of the monthly mean water for particular lake/station for April 2014	56
Figure 39. Result of observation of the monthly mean water for particular lake/station for April 2014	57
Figure 40. API definition roles and assets	58
Figure 41. OGC API Resources navigation tree	60
Figure 42. Multipart standard of OGC API with advanced functionalities	62
Figure 43. Illustration of different mechanisms to attach a JSON schema and/or JSON-LD context to a service response	68
Figure 44. Example of a schema + semantics specialisation hierarchy using OGC Building Blocks	69
Figure 45. STA for the monthly mean water observations for lake: M.-H.-Grenzgr. station: 5865900.....	72
Figure 46. STA observations for the monthly mean water observations for lake: M.-H.-Grenzgr. station: 5865900	73
Figure 47. STA locations for the global trends in biodiversity (BES-SIM AIM)	74

TABLE OF LISTINGS

Listing 1. CoverageDescription in OGC WCS 1.0.....	63
Listing 2. CoverageDescription in OGC WCS 2.1.....	64
Listing 3. Rangeset in OGC API EDR example	64
Listing 4. OGC API Common landing page JSON representation.....	65
Listing 5. Example of the context definition for the API landing page with title, description and all the next level endpoints.....	66
Listing 6. Example of the proposed service description part declaring authentication technology support. JWT is a de-facto standard	70



TABLE OF TABLES

Table 1. Example tabular data of simple observations	43
Table 2. JSON schema fields for the semantic annotation of tables.....	45
Table 3. Proposed attributes extending CSVW to describe units of measures.....	50
Table 4. JSON schema semantic annotations and example of a fully annotated JSON schema definition	67

1 EXECUTIVE SUMMARY

This document presents the results of the first iteration of AD4GD building blocks addressing the semantic interoperability aspects in a GD Data Space. These include the design and implementation of the semantic data model (AD4GD information model) that provides the basis to enable different services to interoperate, i.e., exchanging data with unambiguous meaning, as well as the integration of data collected from various heterogeneous sources in order to provide an integrated view on top of them. Additionally, these building blocks include the methods and tools for data harmonisation and integration, which will support service and data providers in the generation of data that is aligned with the AG4GD information model, complying with data privacy and security requirements to ensure proper data usage and exploitation. Finally, the building blocks include open and standardised APIs necessary to provide the core services of the AD4GD ecosystem, and which are also aligned with the AD4GD information model. For each of the components realising the building blocks, the document includes references to the corresponding repositories, and examples of how they have been used and applied in the AD4GD pilots.

2 INTRODUCTION

2.1 AD4GD GD SEMANTIC INTEROPERABILITY DATA SPACE CONCEPT

One of main goals of AD4GD is to design and implement the building blocks to support a GD Data Space that will provide interoperability support for the heterogeneous and increasingly growing set of data and services available in the various action areas addressed by the GD (e.g., climate, energy, industry, agriculture and biodiversity) through definition, sharing and assembly of standardised building blocks and that will directly contribute towards a European GD Data Space. Indeed, interoperability is one of the main categories of building blocks identified by the OPEN DEI project, as depicted in Figure 1. The three building blocks related to interoperability include:

- Data Models and Formats, which establish a common format for data model specifications and representation of data in data exchange payloads. Combined with the Data Exchange APIs building block, this ensures full interoperability among participants.
- Data Exchange APIs, which facilitate the sharing and exchange of data (i.e., data provision and data consumption/use) between data space participants.
- Data Provenance and Traceability, which provide the means for tracing and tracking in the process of data provision and data consumption/use. It provides the basis for a number of important functions, from identification of the lineage of data to audit-proof logging of transactions. Provenance information will enable the participants to maintain data provenance as part of the metadata during the process of data exchange.

In line with this framework, AD4GD is dealing with the building blocks addressing common interoperability challenges, defining a modular data model for green deal related data, including the provenance metadata, and

APIs, leveraging and reusing existing standards as much as possible, and providing the mechanisms to enable different systems to exchange data with unambiguous meaning, and to enable an integrated data access for the execution of advanced data analytics (WP5), to support the project' pilots (WP6).



Figure 1. Data spaces building blocks from OPEN DEI project. source: Design Principles for Data Spaces – Position Paper - <https://design-principles-for-data-spaces.org/>

Interoperability challenges can be addressed at different levels. The International Data Space Association (IDSA)¹, for example, relies on the European Interoperability Framework (EIF) [1], which defines four layers:

- technical, which covers the interface specifications, interconnection services, data integration services, data presentation and exchange, and secure communication protocols. Hence, this layer includes all the hardware and software components enabling controlled, sovereign and secure data sharing, and ensuring that two different parties are able to technically communicate with each other.
- semantic, which ensures the precise format and meaning of exchanged data and information is preserved and understood between parties.
- organisational, which refers to the alignment of responsibilities, expectations and business processes between stakeholders
- legal, which ensures that organisations operating under different legal frameworks, policies and strategies are able to work together, e.g., they can share data with common legally binding conditions.

The ISO 19941 - Cloud Computing Interoperability and Portability standard² provides another classification of interoperability layers, which is quite similar. This ISO standard makes a separation between transport and syntactic layers, which are part of the technical and semantic layers, respectively, in EIF, and defines a behavioural and a policy layer that correspond to the organisational and legal layers in EIF.

Although AD4GD is interested in all the interoperability layers, in this document we are focusing on the semantic interoperability aspects and the relationships to the technical interoperability layer which in practical terms constrains the common data structures. In particular, semantic Interoperability requires describing parts of

¹ https://docs.internationaldataspaces.org/ids-knowledgebase/v/ids-ram-4/perspectives-of-the-reference-architecture-model/4_perspectives/4_3_governance_perspective/4_3_9_data_spaces_instances

² <https://www.iso.org/standard/66639.html>



information systems that need to be exploited by other components in a system. The available mechanisms for describing the semantics of data are varied and evolving and it's not possible to define a single solution that can be used for all cases, however the technical challenges and best practice options to address these can be identified. The level of expertise required and potential variability of semantic description mechanisms means that reusable patterns need to be identified and applied consistently. This is too complex to achieve for complex and complete application data models, and requires standardisation of specific parts as easy to use “building blocks” for system implementation.

In this sense, our data space concept also gets inspiration from the same approach as the AgriDataSpace project³, which focuses on an incremental model towards a full semantic integration based on the pay-as-you-go (PAYG) data management approach [2]. Inspired by the 5 star rating scheme defined by Tim Berners-Lee⁴ that helps data publishers to evaluate how much their datasets conform to the linked data principles, this model provides flexibility by reducing the initial cost and barriers to joining the dataspace.

According to the PAYG model, at the minimum level, a data source needs to be made available within a dataspace, and over time, the level of integration with the dataspace support services can be improved in an incremental manner on an as-needed basis. These support services include not only data services (catalogue, search/query, data service discovery, etc.), but also stream and event services (event processing, indexing, etc.). A key element not discussed in the PAYG model but relevant to spatio-temporal data is the semantic aspects of data acquisition methodologies, in particular how sampling relates to a phenomenon being observed, and how analytical models treat spatio-temporal granularity and dimensional operations. Consequently, it is critical to understand that the necessary information to support evaluation and integration may be distributed across multiple services and complex processing chains, leading to a requirement for significant levels of semantic interoperability in multiple places in a dataspace.

The more the investment made to integrate with the support services, and the standards used to describe aspects of semantics in different services, the better (faster, cheaper, more robust, repeatable and transparent) the integration achievable in the dataspace.

The five levels (and stars) of the model are (see Figure 2):

1. Basic (minimum): data source is published in the data space with limited or no integration with support services.
2. Machine-readable: the data source is publishing data in a machine-readable format. This enables services to provide a minimal level of support with basic functionality (e.g., browsing the data) where available basic interfaces are exposed.
3. Basic integration: the use of a non-proprietary (data) format, and machine-readable metadata, enable support services to provide essential services at the data-item/entity level with support for simple functionality (e.g., keyword search).
4. Advanced integration: the data is integrated with most support service features (e.g., structured queries) with an awareness of its relationships to other data sources participants with basic support for federation.
5. Full semantic integration: the data is fully integrated into the support services (e.g., question answering) and linked to relevant participants. It plays its full role in the global view of the data space.

³ <https://agridataspace-csa.eu/>

⁴ <https://5stardata.info/en/>

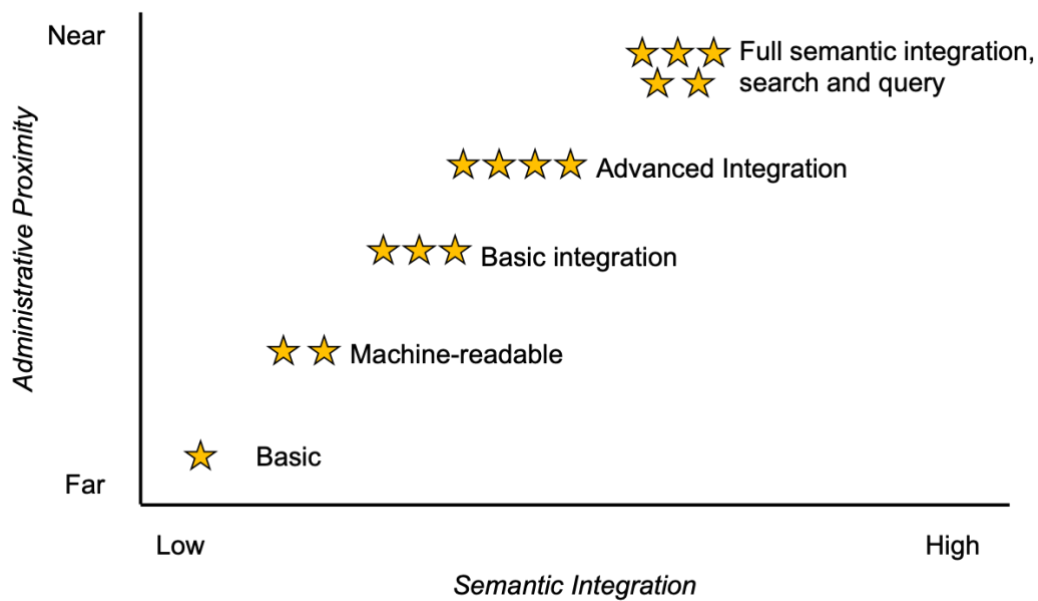


Figure 2. Pay-as-you-go Data Management tiers adapted from [2]

These levels can be also loosely mapped to the ESI/ISO frameworks, although the PAYG model is more concerned about the (meta-)data itself. For example, the basic (minimum) layer is covered by the technical interoperability aspects enabling a basic connection between systems and to transport data from A to B, relying on communication protocols and platform-independent standards for data exchange such as MQTT, HTTP/REST, Kafka, CoAP, DDS, OPC-UA, etc. The machine-readable layer and basic interoperability layers concerned with machine readable data and metadata formats, can be mapped into the syntactic interoperability aspects, and from a building blocks perspective, it translates into foreseeing the implementation of a metadata store which can be used by data consumers to find relevant data, or by data producers to share their own data. Agreeing on a common metadata, though, would be at a higher level of interoperability. The advanced integration and full semantics layers, are both covered by the semantic interoperability aspects, where there is an agreement on the use of open standard formats, data models and APIs.

Hence, in order to enable different systems to exchange data with unambiguous meaning, and the provision of an integrated view over data from different (and heterogeneous) data sources, there is a need for an agreed common information model, or lingua franca, identifying and defining the data elements relevant to the application domain (i.e., Green deal in case of AD4GD) along with their associated semantics/meaning for information exchange, and the delivery of the necessary interoperability mechanisms enabling data transformation/lifting and standard access.

Such an information model, which typically takes the form of ontologies, taxonomies, or controlled vocabularies, would provide the basis of a common green deal data space and enable the interoperability of different systems, potentially from different vendors. In order to maximise the potential interoperability with other data and systems, it is envisioned that such a model will leverage and reuse, whenever possible, existing standards and/or well established ontologies/vocabularies and code lists. Thus, the information model will be a highly modular ecosystem of well-known ontologies, profiled to simplify interpretation and consistent use, and extended with domain specific concepts. Standardisation of many aspects will be an ongoing process, and the modular ontology approach can support this evolution by providing alignment ontologies relating pragmatic and legacy choices with alternatives and emerging standards. The use of standardised ontology languages such as OWL facilitates reasoning over such alignments and robustness of the system as it evolves.



The adoption of a common semantic information model can provide benefits to different stakeholders, including end-users (e.g., pilot stakeholders) and technology providers, in a transparent way. For pilot stakeholders, for example, it would enable them to use the best suited solution for their needs, including systems and components from different technology providers that will be able to seamlessly interoperate and exchange data. The transparent use of these different components would allow them to use the best and most cost-effective combination to carry out their activities efficiently and economically, avoiding vendor lock-in. Additionally, the underlying tools/analytic services will be able to have an integrated access to exploit the full value of available data.

For technology providers, adopting a common semantic information model, would allow their systems and components to interoperate with other existing solutions. This will allow them to focus their efforts on developing specialised components reflecting their main expertise, and/or reduce costs, time and efforts needed to develop components that are already available. More importantly, the possibility to interoperate with components from different providers will allow some providers, especially smaller ones (e.g., SMEs, start-ups), to enter in otherwise monopolised solutions and scale-up. Additionally, technology providers will be able to ensure the future interoperation with other components, as long as they will be able also to produce/consume data compliant with the common model.

Once we have an agreed information model, it is necessary to provide the mechanisms to transform/lift data into this common model and format, and to integrate it with other related datasets, providing a harmonised data layer that can be exploited by the different data analytics tools and decision support systems. As mentioned in the EIF, Linked Data (combined with a data-driven design) can substantially improve semantic interoperability. Indeed, Linked Data is nowadays one of the most popular methods for publishing data on the Web due to the benefits it can provide (e.g., improved data accessibility, support for data integration and interoperability, knowledge discovery and linking). Such an approach has been demonstrated (and it is being demonstrated) in different projects (e.g., DEMETER, SIEUSOIL, ILIAD, OPEN IACS, DataBio, etc.), where Linked Data has been used or is being used as a federated layer to support large scale harmonisation and integration of a large variety of data collected from various heterogeneous sources. Following a similar approach, AD4GD will rely on the implementation of Linked Data pipelines, which in turn rely on the agreed information model, for the representation of data.

Finally, the AD4GD approach considers that different components or systems will implement common open APIs to expose and consume the data, leveraging existing standards particularly from OGC, in order to boost the interoperability potential with existing and future components. In particular, the combined use of a common information model and open APIs will facilitate the integration of heterogeneous data sources, such as satellite-based earth observation-, climate model-, IoT- and citizen science (CitSci) data and measurements provided by scientists as well as other data created by administrations such as INSPIRE data, into a common European GD data space.

Having all these elements in place, it will be possible to exchange harmonised and integrated data between different components with unambiguous meaning and via standard APIs.

These elements can be visualised in a high-level view of the AD4GD data space foundation illustrated in the Figure 3 below. As it can be seen in the figure, the Information Model is one of the main building blocks of the architecture, which is used as the common model to harmonise and represent the datasets (both open and private). The OGC APIs are the interfaces providing the access to the harmonised data, and the data harmonisation/transformation services are part of the software components on the right.

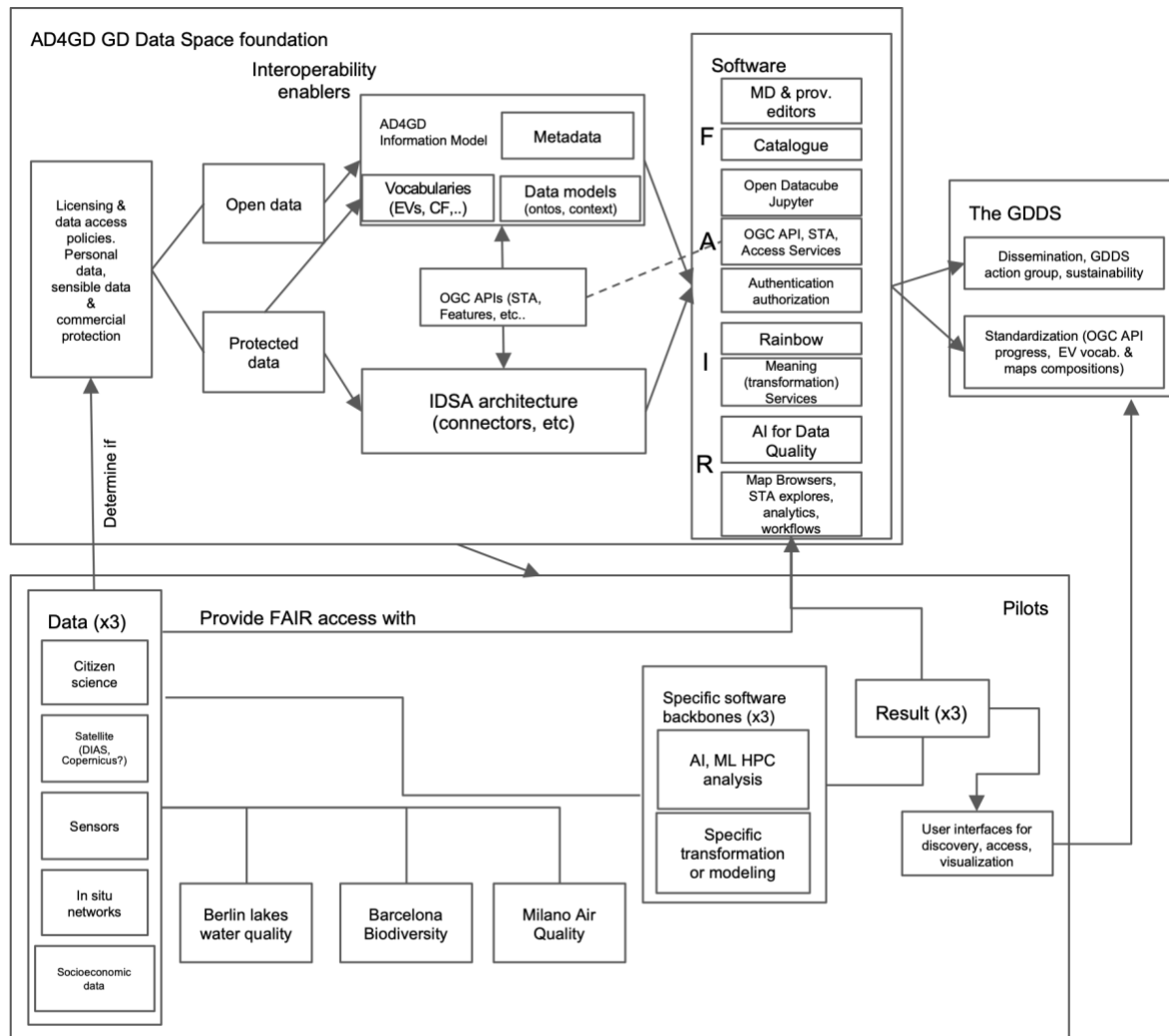


Figure 3. High-level view of the AD4GD data space ecosystem

2.2 PURPOSE OF THE DOCUMENT

This deliverable is intended for both internal and external audiences, interested in learning about the approach taken in AD4GD for building dataspace in the green deal related domains, and particularly regarding how the interoperability aspects are being addressed in order to ensure an unambiguous exchange of data between different systems and the provision of an integrated view of data from different and heterogeneous sources. The document provides the reader with detailed information about the semantic data information model used to represent the data, the tools and technological support to enable the integration of data for the execution of advanced data analytics (WP5), to support the project's pilots (WP6), as well as the set of APIs proposed to expose and consume the data, based on existing standards.

2.3 DELIVERABLE STRUCTURE

The rest of the document is structured as follows:

- Section 3 describes AD4GD information model, including the layered and modular approach, the specific ontologies and vocabularies that were reused, the alignments between them, and the connection with Essential Variables (EV)



- Section 4 describes the methods and tools proposed in AD4GD for data harmonisation and integration, which rely on the use of Linked Data as a federated layer to access multiple sources in an integrated way, and using the previous information model as the underlying model to represent the data
- Section 5 describes the approach to interact with services exposing and consuming data, leveraging existing standards to maximise the interoperability with existing and future systems and applications.
- Section 6 provides the general conclusions

3 AD4GD INFORMATION MODEL

3.1 OVERVIEW

The AD4GD Green Deal Information Model (GDIM) is a common vocabulary aiming at providing the basis of a common green deal data space and enabling the interoperability of different systems, potentially from different vendors. This will in turn enable the analysis of data produced by those systems in an integrated manner to make economically and environmentally sound decisions. The model defines the data elements, including concepts, properties and relationships relevant to green deal applications, as well as their associated semantics/meaning for information exchange.

Building on the review of the state of the art and based on the initial analysis of the modelling requirements (T1.1), the AD4GD information model is built following best practices, and reusing and extending existing standard ontologies and vocabularies available for the relevant domains, including in particular OGC and W3C standards whenever possible. The model, though, is treated as an agile artefact responding to the changing requirements of the pilots and the wider GD ecosystem. The model includes semantic mappings (“alignments”) between standard and/or dominant data models/ontologies, which will be exploited in WP6 for the (semantic) integration of pilot data.

Based on the GDIM, i) data producers/integrators will be able to adapt and apply existing tools to pre-process, integrate and harmonise data from different sources (see Section 4); ii) service providers will be able to develop lightweight service wrappers and translators, also known as data providers and consumers, which will enable the different tools/platforms to expose and consume data in an interoperable form.

The GDIM will be accessible via the OGC Registry for Accessible Identifiers of Names and Basic Ontologies for the Web (RAINBOW) server (formerly OGC Definitions Server), which supports the profiling of complex models to provide a pathway to multiple implementation patterns. The server is a Web accessible source of information about things (“Concepts”) that are defined by the broader community, such as AD4GD, or result from external but related standardisation and policy bodies. OGC standards use stable web addresses (URIs) to unambiguously identify concepts in its specifications. The OGC RAINBOW makes those URIs “work” - i.e., makes them dereference to a definition that can be used, so that vocabularies with uniquely defined terms can be created.

The other service provided by the OGC RAINBOW is a Continuous Integration/Continuous Testing/Continuous Deployment (CI/CT/CD) implementation of best practices in the informatics domain. This means that each component of the information model is supported by examples and validation testing, including core components and alignments. CI/CT/CD means that development can be accelerated by providing stakeholders confidence in proposed model components, and regression testing automated to ensure that any corrections and refinements do not introduce unforeseen inconsistencies.

3.2 MODEL LAYERS



In line with best practices and recommendations, the specification of GDIM follows a modular approach in a layered architecture, enabling among others:

- straight-forward interoperability with existing models by reusing available (well-scoped) models in the modules, instead of defining new terms, whenever possible,
- easy mapping/alignment with other models, by module instead of the entire model,
- easy extension of the domain/areas covered in OIM with additional modules,
- easy extension of the domain model, by modifying only specific modules,
- easy mapping to top-level/cross-domain ontologies.

Based on the analysis of the state of the art, GDIM is being implemented by reusing and building partially over the Agriculture Information Model (AIM). The same approach has been taken to adapt AIM in the Ocean domain in the ILIAD project to create the Ocean Information Model (OIM), currently under development.

AIM provides the basis to enable a semantic interoperability data space in agriculture. AIM was designed following a layered and modular approach, and was realised as a suite of ontologies implemented in line with best practices, reusing existing standards and well-scoped dominant models as much as possible and establishing alignments between them to enable their interoperability and the integration of existing data. AIM is currently under the process of becoming an OGC standard. Accordingly, the Agriculture Information Model Standards Working Group (AIM SWG) was recently established under the auspices of the Agriculture Domain Working Group (<https://github.com/opengeospatial/aim-swg>), following the OGC procedures for establishing new standards.

A key value provided by AIM is that it harmonises and aligns relevant cross-domain standards such as Time Ontology, SOSA/SSN, GeoSparql, QUDT, Data Cubes, DCAT, The Profiles Vocabulary and PROV with domain-specific models, bridging various views on the agriculture data and providing a formal representation enabling unambiguous translations between them. In that line, AD4GD will build on those cross-domain standards and specialise the model for the green deal domain.

Hence, analogously to AIM, the GDIM defines the following layers:

- the meta-model layer, defining the model building blocks and enabling the back-and-forth conversion between datasets that are based on the property graph model and linked data datasets
- the cross-domain layer, defining relevant concepts and properties that are common across multiple domains, and enabling the interoperability with existing standard models and vocabularies
- the domain layer defining green deal related concepts and properties covering different aspects of interest of GD applications, and enabling the integration of relevant vocabularies in the area.
- the pilot-specific layer defining additional concepts and properties that are of specific use for particular applications (if needed)
- a metadata model layer that can be used to describe datasets, services or applications in AD4GD.

GDIM is scalable and can easily be extended in order to address additional needs and incorporate new concepts, maintaining its consistency and compliance. An overview of the initial release of the GDIM is provided in Figure

4. In the Figure, the dark blue boxes are implemented, while the grey ones are not (yet). Hence, the pilot-specific modules (if needed) will be specified in the next iteration after pilots have matured.

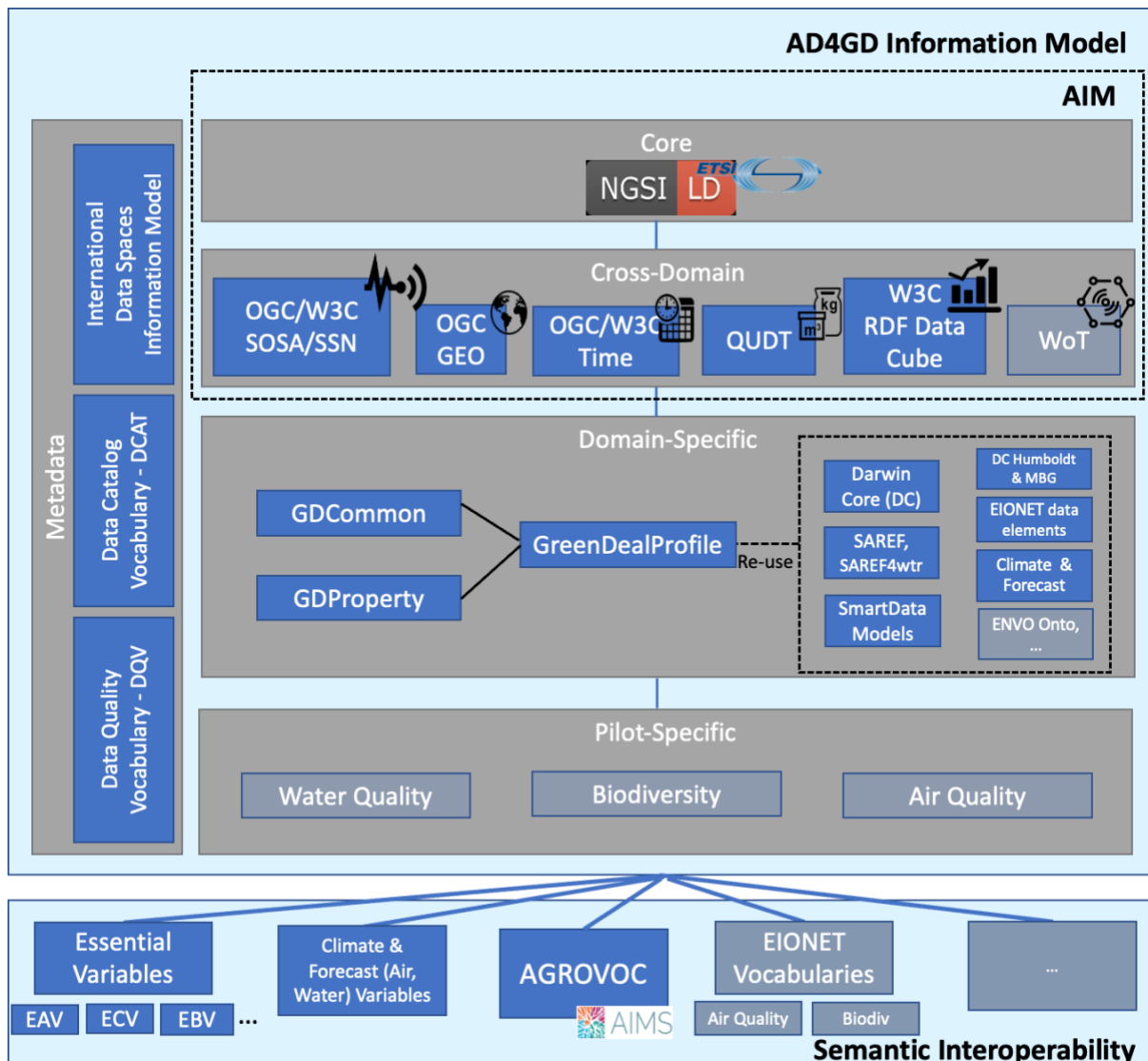


Figure 4. Overview of the layers of the first release of the AD4GD Information Model

3.2.1 GENERAL IMPLEMENTATION APPROACH

In general, AD4GD modules have been implemented as OWL ontologies, and serialised as Turtle. From these ontologies we generate several related semantic artefacts, including JSON-LD contexts and SHACL shapes (to be carried out during the next period). The former allows the encoding of linked data in JSON, one of the most commonly used formats to exchange data between services; it also helps JSON data to interoperate at Web-scale. The context in JSON-LD is used to map terms, i.e., properties with associated values in a JSON document, to URIs (Uniform Resource Identifiers), such as OWL entities. JSON-LD contexts allow disambiguating keys shared among different JSON documents by mapping them to URIs which describe their meaning: two applications can use shortcut terms to communicate together more efficiently, without losing accuracy. The latter allows the validation of RDF data against GDIM at the semantic level, which can enable providers and consumers to make sure the data produced/consumed is compliant with the model.



A key challenge is the development and use of specific JSON data structures, specified in JSON-schema and used to define APIs in modern frameworks such as OpenAPI. The mapping of JSON schemas to the GDIM model via JSON-LD is done on a modular basis, allowing this complex task to be reused and leveraged by applications. This process may require **structural translation of schema** to an intermediate form, and is a pattern required for use of other common data structures, such as CVS, NestCDF and other compact forms. Section 4 describes a range of alternatives that can be extended as requirements emerge. Note that requiring transformations to support semantic annotation can be used during design and testing, and generally speaking does not need to be performed at “run-time” - but can be performed when evaluating data, configuring processing steps and exploring the data used to produce results to understand it.

All the AD4GD modules and related resources are publicly available in the AD4GD GitHub repository: <https://github.com/AD4GD/GDIM>.

For the implementation of the ontologies and related contexts, we used different tools. The ontologies were implemented mostly manually using a simple text editor, updating and extending the reused modules from AIM, and using these modules as skeletons for AD4GD-specific modules. The ontology editor Protégé was also used mostly to visualise and navigate the ontologies, and to validate them using the reasoners available in the tool. In particular, we used the Pellet reasoner to verify the logical consistency of the ontologies.

In order to transform the ontologies generated into JSON-LD contexts (to enable services to exchange JSON data about the different entities), we used the tool owl2jsonld. This tool generates a JSON-LD @context for concepts (classes and properties) found in the specified OWL or RDFS ontology. The script to generate the contexts for the AD4GD modules is available at: <https://github.com/AD4GD/GDIM/tree/main/jsonld/utills>

All the ontologies generated, as well as the corresponding JSON-LD contexts, use persistent identifiers that are resolvable. This facilitates both the sharing and usage within different applications, through time. In particular, we use w3id service for permanent identifiers on the Web. The service, run by the W3C permanent identifier community group, provides secure, permanent URL re-directions for Web applications. As a result, AD4GD resources will always be accessible and resolvable, even if the physical locations of the resources change. The base namespace for OIM is: <https://w3id.org/ad4gd/model> (which resolves to the AD4GD green deal profile module, the main entry point to the IM).

Regarding mappings between different vocabularies/ontologies, GDIM defines them in each module by including appropriate ontology axioms, such as equivalent classes (owl:equivalentClass), equivalent properties (owl:equivalentProperty), subclasses (rdfs:subClassOf) and subproperties (rdfs:subPropertyOf). We reused the mappings already available from AIM and created new ones when possible. This process is being carried out mostly manually.

3.2.2 CORE META-MODEL LAYER

A meta-model, as its name implies, is a model of a model. Meta-models are typically used for different purposes. For instance, they can be used for the specification of modelling language constructs in a standardized, platform independent manner [HaPa09], to specify and restrict a domain in a data model and systems specification [lvVo11], or to provide an explicit model of the constructs and rules needed to build specific models within a domain of interest [Wel]. In fact, as noted in [Wel], meta-models can be viewed from three different perspectives: i) as a set of building blocks and rules used to build models; ii) as a model of a domain of interest; iii) as an instance of another model. In the context of the ILIAD meta-model, we are considering it as the first perspective.



In AD4GD, the core-model layer is based on the AIM corresponding layer, which follows the NGSI-LD meta-modelling approach [NGS1]. NGSI-LD is based on a 3-layer architecture, including a property graph meta-model layer grounded in RDF/RDFS, a cross-domain ontologies layer, and the domain/application ontologies. However, as opposed to NGSI-LD, AIM, and thus GDIM, implement the cross-domain and domain/application layers by reusing existing standards and/or well-known ontologies/vocabularies as much as possible from the outset, thereby implementing semantic referencing. Moreover, the information models (GDIM, AIM, OIM) have extended the architecture to include the pilot specific extensions, the metadata layer, and the explicit alignments to other vocabularies.

It is important to note, though, that the meta-model layer can be based and/or aligned with other meta-models in the future. Hence, the current releases of the information models (GDIM, AIM, OIM) extracted the alignments from the cross-domain layer to NGSI-LD into a separate module, enabling the creation of other modules with alignments to other meta-models. For example, in the ILIAD project, the OIM is also using OGC Features and OGC EDR as meta-models.

The GDIM core meta-model ontology is available at:
https://github.com/AD4GD/GDIM/blob/main/alignments/ngsi-ld_ad4gd.ttl

3.2.3 CROSS-DOMAIN LAYER

Cross-domain ontologies are defined as a set of generic models which are aimed at avoiding conflicting or redundant definitions of the same classes in the domain-specific layer. Selecting such ontologies is the basis for interoperability with other information systems and tooling that already use these. Hence, in general “canonical” ontologies managed by standardisation bodies are preferred, although “de facto standards” in widespread use may have advantages.

Hence, the main objectives of the GDIM cross-domain layer are, to:

- Capture concepts and terms that are generic and applicable to various domains.
- Avoid conflicting or redundant definitions of the same concept in different domain specific models.
- Provide a basis for interoperability with other information systems and tooling.

Figure 5 presents an overview on the cross-domain layer. As previously mentioned, this layer has been specified by reusing (parts of) and aligning various relevant standard ontologies and vocabularies, resulting in a generic model that bridges between the different ontologies. The layer reuses:

- W3C OWL Time ontology for concepts of temporal properties and time values
- OGC GeoSPARQL ontology and associated definitions for geographical and geometrical properties
- The W3C/OGC recommendation SOSA/SSN ontologies regarding sensor and actuator data, including observations, observation collections, observed properties, systems and platforms
- QUDT ontologies regarding units of measurement, and concepts to represent quantities and quantity kinds, with a partial QU aligned classification of quantity kinds
- Concepts from the RDF data cube vocabulary to represent statistical data, including datasets, data structures, slices, measure properties, dimension properties, etc.
- Concepts from ISO geographic technology standards, including features (domain and sampling feature), and observations
- Basic terms from general purpose standards or widely used vocabularies like skos, foaf, schema.org

As part of AD4GD, the cross-domain ontology of AIM has been extended to include additional quantity kinds that are relevant to the project pilots and applications.

The AD4GD cross-domain ontology is available at: <https://github.com/AD4GD/GDIM/blob/main/cross-domain.ttl>

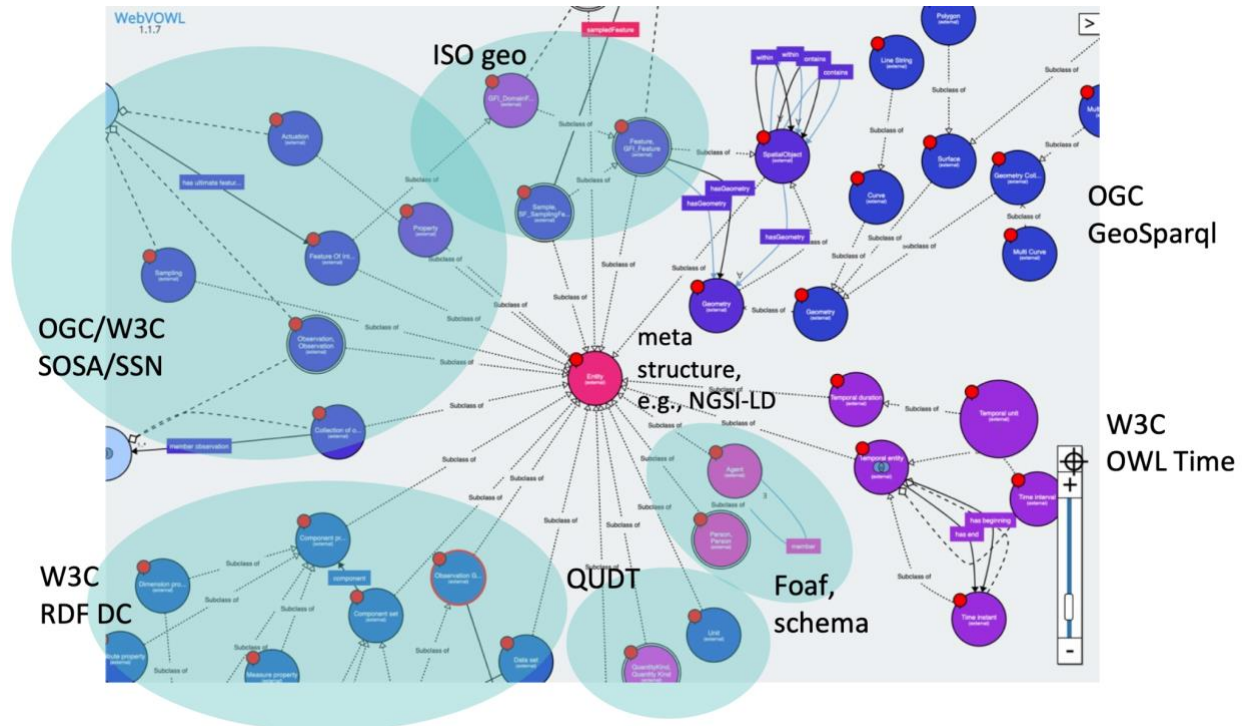


Figure 5. Overview of the first release of the GDIM Cross-Domain ontology

The module that contains the alignments of the cross-domain terms with core meta-model terms (NGSI-LD) is available at: <https://github.com/AD4GD/GDIM/blob/main/alignments/gdim-ngsi-ld.ttl>

3.2.4 DOMAIN-SPECIFIC LAYER

In addition to the cross-domain ontology that covers the data model that spans different application domains, as part of AD4GD we also need to define the ontology/ies that will cover the data needed specifically in the Green Deal domain. Following an analysis of requirements from different pilots and applications providers, we have identified a number of areas that the GDIM needs to cover, which are relevant for the development and support of different green deal based platforms and solutions. As our goal is to ensure interoperability with existing ontologies and systems, we base this model on a number of such existing ontologies with their terms aligned (as several have overlapping terms) and enriched (where appropriate). In particular, the ontologies and models reused are described in next subsection

3.2.4.1 VOCABULARIES/MODELS ALIGNED

- Darwin Core (DC) (<https://dwc.tdwg.org/>): Darwin Core is a standard maintained by the Darwin Core Maintenance Interest Group. It includes a glossary of terms, such as properties and concepts, with identifiers, labels, and definitions, which are intended to facilitate the sharing of information about biological diversity. Darwin Core is primarily based on taxa, their occurrence in nature as documented by observations, specimens, samples, and related information.
- Darwin Core Humboldt extension (<https://www.tdwg.org/community/osr/humboldt-extension/>): this DC extension includes terms to enable inventory data to be shared, re-used, compared to one another,



and further integrated with other sources of biodiversity data to significantly expand biodiversity dataset discovery, interoperability, and modelling.

- Darwin Core Marine Biogeography (https://mmisw.org/ont/ioos/marine_biogeography) : this DC extension includes terms related to marine biogeography to meet specific requirements identified by OBIS-USA (Ocean Biodiversity Information System USA) and US IOOS (U.S. Integrated Ocean Observing System Program).
- SAREF (Smart Applications REFerence Ontology) (<https://saref.etsi.org/>): SAREF explicitly specifies recurring core concepts in the smart applications domain, the main relationships between these concepts, and axioms to constrain the usage of these concepts and relationships. In particular, GDIM domain layer reuses mostly terms from SAREF extension for water (saref4watr) and SAREF core.
- SmartDataModels (<https://smartdatamodels.org/>): a collaborative program (led by FIWARE) to provide data models for digital twins and data spaces to allow data interchange between organisations by providing open licensed shared data models according to the principles of agile standardisation. In particular, GDIM domain layer reuses mostly elements from the models for WaterQualityObserved, WaterQualityPredicted, WaterObserved, WeatherObserved, and the common model.
- Climate & Forecast (features) (<https://www.w3.org/2005/Incubator/ssn/ssnx/cf/cf-feature>): This is an ontology of the generic features defined by Climate and Forecast (CF) standard names vocabulary (<https://cfconventions.org/Data/cf-standard-names/82/build/cf-standard-name-table.html>), maintained by the Program for Climate Model Diagnosis and Intercomparison (<http://cf-pcmdi.llnl.gov/>) which is intended for use with climate and forecast data, in the atmosphere, surface and ocean domains. In particular, GDIM domain layer reuses features terms related to air and water, and the corresponding (observed) properties related to those features.

For the next period we will evaluate other relevant ontologies identified, such as the ENVI Ontology.

In this initial version of the GDIM, the domain layer includes three modules, namely gdCommon, gdProperty, and greenDealProfile. For their implementation

- The gdCommon module (<https://github.com/AD4GD/GDIM/blob/main/gdCommon.ttl>) includes common properties and concepts used across all other domain modules. This module includes mostly elements from Darwin Core (and extensions), and some general terms from SmartDataModels.
- The gdProperty module (<https://github.com/AD4GD/GDIM/blob/main/gdProperty.ttl>) is focused on the different properties measured/observed in green deal related applications, and particularly related to the pilots domains (e.g., air temperature, direction, etc.) and their connection to the systems used to collect them and features of interest (e.g., air, water, etc.). This module includes mostly elements from SAREF, SmartDataModels and Climate & Surface (features).
- The green deal profile ontology (<https://github.com/AD4GD/GDIM/blob/main/greenDealProfile.ttl>) imports all the GDIM ontology modules and is the main entry to model, i.e., it's the module used to load the whole model in an application or an editor (e.g., Protégé). It is the module resolved by: <https://w3id.org/ad4gd/model>.

3.2.5 PILOT-SPECIFIC LAYER

Following the same approach as AIM, GDIM may also include also a pilot-specific layer which includes ontology modules consisting of terms and properties that are needed by the project (or other) pilots to integrate their systems to AD4GD and comply with GDIM, and which are not defined in any well-known ontology and are not generic enough to become part of the domain layer.

This layer will be developed during the next iteration, if needed, once the pilots have matured and the data requirements have been stabilised.

3.2.6 METADATA LAYER

Metadata, “a set of data that describes and gives information about other data”, is a pervasive concept that is relevant to all components of an information system. The GDIM metadata layer is at the moment entirely based on the AIM corresponding layer. As the other layers, this layer is based on existing standard metadata vocabularies and models. In particular, it builds up on DCAT 2.0 and uses a subset of the IDS Information Model to increase the expressivity of certain aspects of the GDIM Metadata Schema. Furthermore, the GDIM Metadata Schema makes references to the W3C DQV to allow capturing data quality information.

The metadata schema is available via: <https://github.com/AD4GD/GDIM/blob/main/metadata.ttl>

Figure 6 shows the core classes of DCAT, IDS and DQV and how they were integrated within the GDIM Metadata Schema. The `idsa:Artifact` represents a physical instance of a data set. It can represent, a physical file as well as the output of an Endpoint (e.g., a Webservice). Properties of `idsa:Artifact` refer to the physical nature of the data (timestamps, filesize, checksum) as shown in Figure 7. The `idsa:Artifact` is more specific than the `dcat:Dataset` or its subclass `idsa:DigitalContent`. A `dcat:Dataset` (or its subclass `idsa:DigitalContent`) represent the properties of a data set which tend to be static across multiple batches from the same data source. These are properties such as the type of content (file format, schema), the frequency of updates, the temporal resolution and the spatial coverage. In general, the endpoint properties are shown in Figure 8.

A key activity moving forward will be addition of standardised profiles of the PROV (provenance) ontology to support semantic descriptions of data sets. Basic properties such as `creationDate` in DCAT are not sufficient to understand data characteristics that arise from original data acquisition and subsequent processing activities.

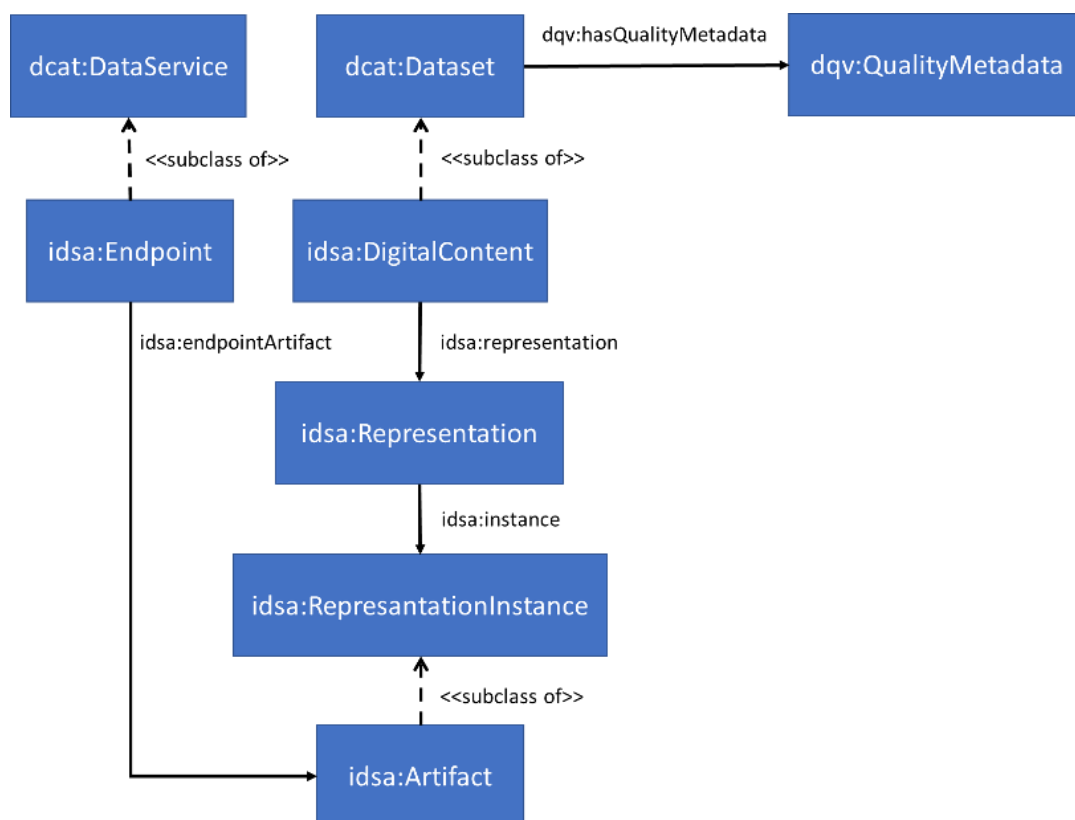


Figure 6. Core classes of the GDIM Metadata Schema

The GDIM metadata schema is proposed to AD4GD solution providers who provide catalogues or other similar repositories. As part of the uptake process, we expect feedback regarding missing properties or terms, and that will be addressed during the next period of the project.

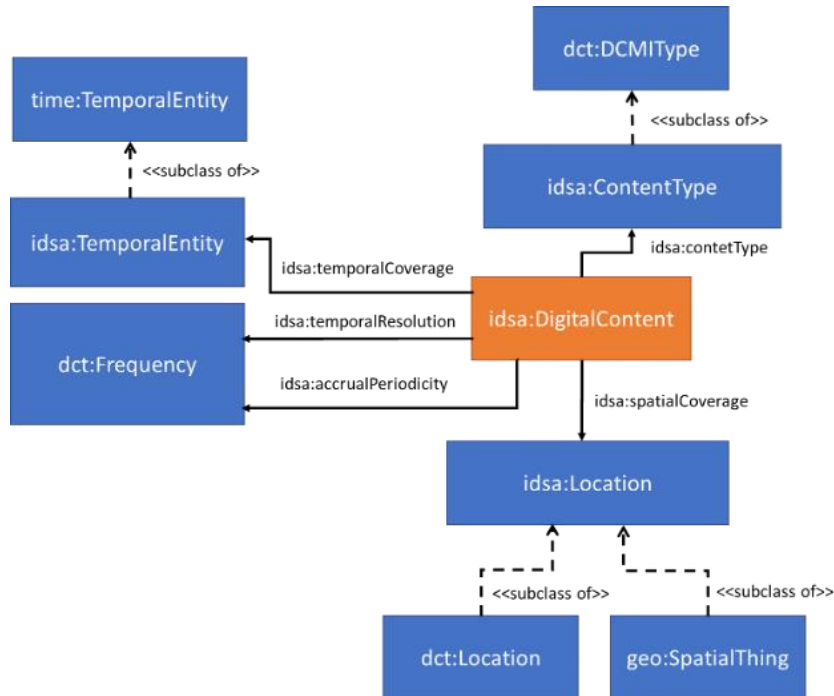


Figure 7. Digital Content properties in the OIM Metadata Schema

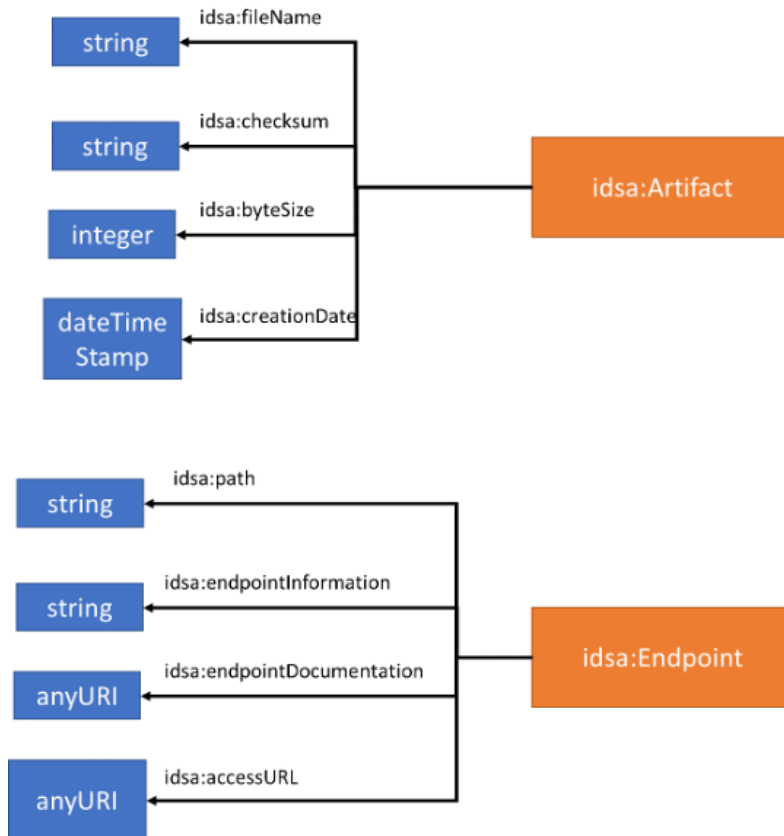


Figure 8. Artefact and Endpoint properties in the OIM Metadata Schema

3.3 SEMANTIC INTEROPERABILITY WITH OTHER VOCABULARIES

In addition to including alignments between the reused ontologies and vocabularies across the different layers, GDIM establishes connections with various well-known vocabularies and thesauri to enable interoperability with data based on them. Note that these connections are not formal alignments, like the ones included across the GDIM layers, but there are mechanisms to connect terms from these vocabularies to GDIM data.

GDIM currently includes connections to Essential Variables, AGROVOC and the Climate & Forecast (air and water) variables. In the next period we will evaluate other vocabularies like the EIONET (air quality and biodiversity) vocabularies.

3.3.1 ESSENTIAL VARIABLES

The Essential Variables (EVs), defined as part of expert groups within the GEO community, are [3] sets of variables that are crucial for characterising and monitoring particular systems across space and time, providing insight into underlying processes and their changes, and/or feeding indicators that inform environmental policies at multiple scales. Each EV can be implemented as one or more EV products that are measurable parameters needed to characterise the EV. An EV product defines a unit of measure and a set of requirements specifying spatial and temporal resolution among others. EV products may already exist as datasets; being produced by remote sensing, in-situ observations, modelling, etc or may not exist as a dataset yet.

So, at the core of the EV products are the observations, which condition the requirements (e.g. temporal and spatial scales) of the environmental data collected. There are over 10 different EV being defined by the different expert groups. Together they describe the socio-ecological Earth system for its monitoring and modelling in order to track progress towards sustainable development. Figure 9 below proposed by Lehmann et al. [4] provides an integrated vision of EVs for the global socio-ecological system and their development status in 2021 in the different GEO SBAs (Societal Benefit Areas).

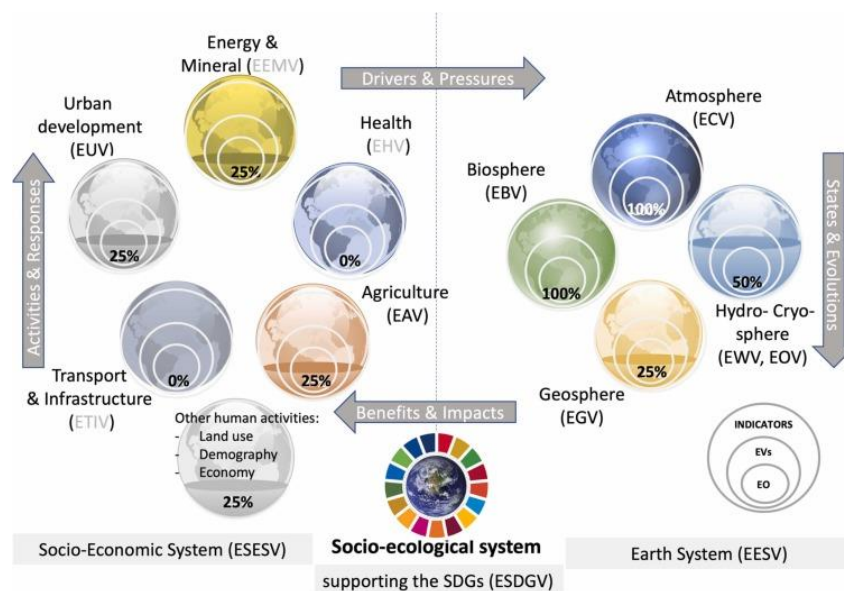


Figure 9. EV status in 2021 in the GEO SBAs in support of SDGs and in relationship with the DPSIR policy framework. Percentages represent level of development of EV classes from EO to indicators

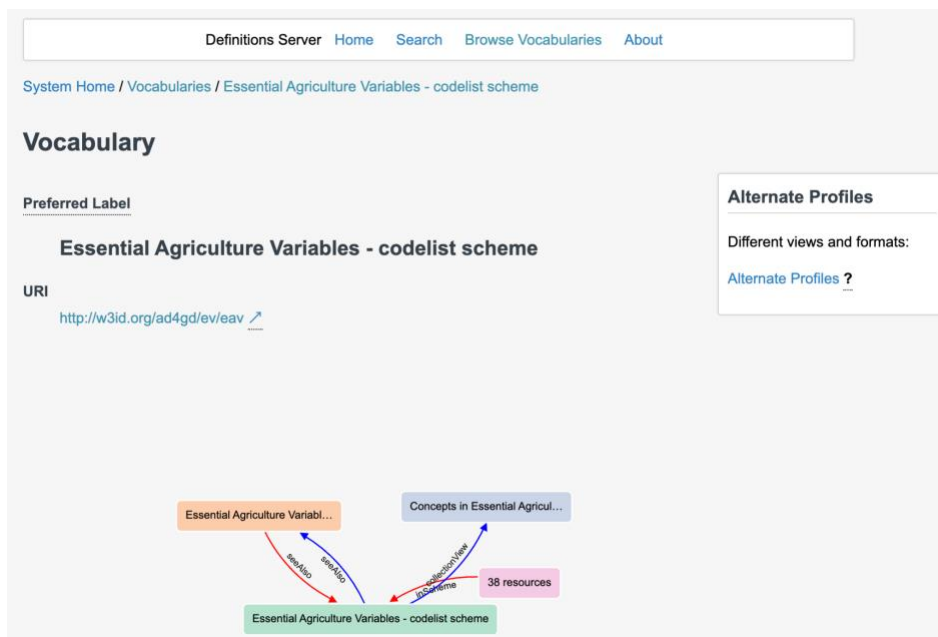
As described in the GEO report [3], EVs are semantic concepts requiring semantic interoperability. That means that product or dataset providers must inform users that the dataset content is an EV as defined by the community and generated according to a well-recognized workflow. Moreover, the report recommends having

user and service interfaces for discovering and accessing EV products. To enable this, an ontology/vocabulary of EVs and EV product requirements should be defined/maintained by an international body capable of managing it, allowing annotating data and models and enhancing their discovery and usage.

AD4GD has addressed that vision by starting the definition of ontologies for the different EVs. In particular, we created controlled vocabularies expressed using SKOS (Simple Knowledge Organisation System)⁵ and OWL, where each EV is defined as a concept Scheme with a corresponding OWL class that enumerates all the different products. If EV defines classes and products they are organised in a taxonomy of concepts, and each product is defined as both a concept of the EV concept scheme, and an instance of the corresponding EV class. Finally, each product definition includes all the detailed information identifying it (e.g., spatial and temporal coverage).

Furthermore, the EVs concepts use PID (permanent identifiers), based on the W3ID service that provides secure, permanent URL redirects for Web applications. Finally, the EVs are published in the OGC rainbow server maintained by OGC (international standardisation organisation).

At the moment, we have two EVs vocabularies: i) the Essential Agriculture Variables (EAVs) - and the ii) Essential Biodiversity Variable (EBVs). The PIDs of EAVs (<https://w3id.org/ad4gd/ev/eav>) and EBVs (<http://w3id.org/ad4gd/ev/ebv>) redirect to the entry page of the EV in the OGC rainbow server (see Figure 10, Figure 11 for EAV and Figure 12, Figure 13 for EBVs). Additionally, each term has its own PID which is also resolvable and redirects to the OGC rainbow server, e.g., Ecological Quality Ratio (EQR) of freshwater fish: [http://w3id.org/ad4gd/ev/ebv/EcologicalQualityRatio\(EQR\)offreshwaterfish](http://w3id.org/ad4gd/ev/ebv/EcologicalQualityRatio(EQR)offreshwaterfish) (see Figure 14 and Figure 15)



The screenshot shows the OGC Rainbow server interface for the 'Essential Agriculture Variables - codelist scheme'. The page includes a navigation bar with 'Definitions Server', 'Home', 'Search', 'Browse Vocabularies', and 'About'. The breadcrumb trail is 'System Home / Vocabularies / Essential Agriculture Variables - codelist scheme'. The main heading is 'Vocabulary'. Under 'Preferred Label', it reads 'Essential Agriculture Variables - codelist scheme'. The 'URI' is 'http://w3id.org/ad4gd/ev/eav'. A sidebar on the right offers 'Alternate Profiles' and 'Different views and formats: Alternate Profiles?'. A diagram at the bottom shows three interconnected boxes: 'Essential Agriculture Variabl...' (orange), 'Concepts in Essential Agricul...' (blue), and 'Essential Agriculture Variables - codelist scheme' (green). Arrows indicate relationships: 'isScheme' from the green box to the orange box, 'isConcept' from the orange box to the blue box, and 'isResource' from the blue box to the green box. A pink box labeled '38 resources' is also connected to the green box.

Figure 10. Essential Agriculture Variables (EAVs) entry page in OGC Rainbow server (part 1)

⁵ <https://www.w3.org/TR/swbp-skos-core-guide/>



Definition Essential Agricultural Variables for GEOGLAM are Earth observation-based building blocks that in combination with one another or with other non-EO information provide insight into the GEOGLAM Agricultural Indicators which themselves provide actionable information on the state, change, and forecast of agricultural land use and productivity. This list cover Land Use and Productivity EAVs.

Concept Hierarchy expand all, click "*" to expand individually

- [Above Ground Agricultural Biomass](#)
- [Actual Evapotranspiration](#)
- [Agriculture Mask](#)
- [Air Temperature](#)
- [Crop Condition Assessment](#)
- [Crop Residue Cover Percentage](#)
- [Crop Rotation Sequence](#)
- [Crop Type Area Estimate](#)
- [Crop Type Masks](#)
- [Crop Yield Estimation](#)
- [Crop Yield Forecast](#)
- [Cropland Mask](#)
- [Current Crop Stage](#)
- [Degree Growing Days](#)
- [Field Boundaries](#)
- [Fractional Cover](#)
- [Incoming Radiation](#)
- [Irrigated Cropland Map](#)
- [Land Surface Temperature](#)
- [Leaf Area Index](#)

Search

Search

Figure 11. Essential Agriculture Variables (EAVs) entry page in OGC Rainbow server (part 2)

Definitions Server [Home](#) [Search](#) [Browse Vocabularies](#) [About](#)

[System Home](#) / [Vocabularies](#) / Essential Biodiversity Variables - codelist scheme

Vocabulary

Preferred Label

Essential Biodiversity Variables - codelist scheme

URI
<http://w3id.org/ad4gd/ev/ebv>

Alternate Profiles

Different views and formats:

[Alternate Profiles ?](#)

```

    graph TD
      A[Essential Biodiversity Variab...] -- "Also" --> B[Essential Biodiversity Variables - codelist scheme]
      B -- "Also" --> A
      B -- "77 resources" --> C[Essential Biodiversity Variab...]
      B -- "6 resources" --> D[Concepts in Essential Biodive...]
      B -- "collezione" --> D
  
```

Figure 12. Essential Biodiversity Variables (EBVs) entry page in OGC Rainbow server (part 1)



Definition
 Essential Biodiversity Variables (EBVs) are defined as a minimum set of measurements, complementary to one another, that can capture major dimensions of biodiversity change. EBVs are organized in six classes (Genetic composition, Species populations, Species traits, Community composition, Ecosystem functioning, Ecosystem structure) and cover the three realms (Marine/coastal, Terrestrial and Freshwater)

Search
 Your search term

Concept Hierarchy expand all, click '+' to expand individually

- Community composition
 - Aerial biomass of migrating birds, bats and insects
 - Community abundance and taxonomic diversity of pollinator insects
 - Community biomass of selected functional groups of terrestrial arthropods (e.g. predator, decomposer)
 - Community biomass of soil microbes
 - Ecological Quality Ratio (EQR) of benthic freshwater invertebrates
 - Ecological Quality Ratio (EQR) of freshwater fish
 - Ecological Quality Ratio (EQR) of freshwater macrophytes
 - Ecological Quality Ratio (EQR) of freshwater phytobenthos
 - Ecological Quality Ratio (EQR) of freshwater zooplankton
 - Ecological Quality Ratio (EQR) of phytoplankton in lakes
 - Functional composition of marine phyto/zooplankton (based on traits)
 - Functional composition of soil biota
- + Ecosystem function
- + Ecosystem structure
- + Genetic composition
- + Species populations
- + Species traits

Figure 13. Essential Biodiversity Variables (EBVs) entry page in OGC Rainbow server (part 2)

Definitions Server Home Search Browse Vocabularies About

System Home / Vocabularies / Essential Biodiversity Variables - codelist scheme / Ecological Quality Ratio (EQR) of freshwater fish

Concept

Preferred Label
Ecological Quality Ratio (EQR) of freshwater fish

URI
[http://w3id.org/ad4gd/ev/ebv/EcologicalQualityRatio\(EQR\)offreshwaterfish](http://w3id.org/ad4gd/ev/ebv/EcologicalQualityRatio(EQR)offreshwaterfish)

Within Vocab
 Essential Biodiversity Variables - codelist scheme

Alternate Profiles
 Different views and formats:
[Alternate Profiles ?](#)

```

    graph TD
      CC[Community composition] -- narrower --> EQR[Ecological Quality Ratio (EQR) of freshwater fish]
      EQR -- broader --> CC
      EQR -- isobroader --> EBV1[Essential Biodiversity Variab...]
      EQR -- type --> EBV2[Essential Biodiversity Variab...]
    
```

Figure 14. Ecological Quality Ratio (EQR) of freshwater fish in OGC Rainbow server (part 1)



Definition	None
Broader	Community composition
Object Type	Essential Biodiversity Variables - codelist class
Status	valid
Identifier	13
Is Defined By	https://github.com/EuropaBON/EBV-Descriptions/wiki/Master-EBV-List
http://w3id.org/ad4gd/ev/spatialResolutionUnit	Lakes and river catchments as delineated in ECRINS (European catchments and rivers network system)
http://w3id.org/ad4gd/ev/TaxonomicEcosystemFocusGroup	Freshwater fish species with indicator values defined in the Water Framework Directive Intercalibration Technical Reports (Part 1, Rivers; Part 2, Lakes)
http://w3id.org/ad4gd/ev/realm	Freshwater
http://w3id.org/ad4gd/ev/stepInIdentificationProcess	User & Policy Needs Assessment
http://w3id.org/ad4gd/ev/temporalResolutionUnit	3-6 years

Figure 15. Ecological Quality Ratio (EQR) of freshwater fish in OGC Rainbow server (part 2)

The EVs are used in GDIM as observable properties to describe the observations/measurements. That is, following the sosa/ssn pattern, an observation (or observation collection) is linked to the property that was observed, whereas this property is an observable quality (property, characteristic) of the FeatureOfInterest linked to the observation (observation collection).

3.3.2 CLIMATE & FORECAST VARIABLES

Climate & Forecast (CF) standard name vocabulary (<https://www.w3.org/2005/incubator/ssn/ssnx/cf/cf-property>) is an ontology representing the climatic data variables defined by the Climate and Forecast (CF) standard names vocabulary (<https://cfconventions.org/Data/cf-standard-names/82/build/cf-standard-name-table.html>), maintained by the Program for Climate Model Diagnosis and Intercomparison (<http://cf-pcmdi.llnl.gov/>) which is intended for use with climate and forecast data, in the atmosphere, surface and ocean domains.

The vocabulary includes hundreds of variables (properties), but for AD4GD we are leveraging in particular those related to air and water. These variables have been imported in the gdProperty module described in Section 3.2.4.1.

Similar to EVs, these CF variables are used in GDIM as observable properties to describe the observations/measurements. That is, following the sosa/ssn pattern, an observation (or observation collection) is linked to the property that was observed, whereas this property is an observable quality (property, characteristic) of the FeatureOfInterest linked to the observation (observation collection).

3.3.3 AGROVOC

The Agrovoc vocabulary⁶ is a collection of component vocabularies related to the Agrifood, developed to support semantic representations and data modelling. The use of such standard vocabulary can ensure both interoperability and absence of ambiguity in the data interpretation process. Agrovoc is today the most complete multilingual controlled vocabulary for agriculture. One of the predominant aspects, which highlights it more than

⁶ <https://agrovoc.fao.org/browse/agrovoc/en/>



other vocabularies/thesauruses, is its multilingual character, useful especially for those applications that involve interactions with multiple users (e.g. Web Platform). Even if today it is possible to encode metadata from several languages into English (which still represents the predominant language for ontologies and vocabularies), in the Agrifood sector it is essential to be able to have access to different countries language labels (e.g., Czech, Danish, German, Italian, Polish, Portuguese, Slovak and Thai.) for different concepts.

Agrovoc was originally designed for indexing literature, but it is also used to facilitate the sharing and exchange of knowledge through electronic media and data formats. It contains over 40,000 concepts in up to 21 languages, where each concept is identifiable by its own PID, e.g., http://aims.fao.org/aos/agrovoc/c_12332 (maize as one plant product).

As Agrovoc concepts can be associated with different types of entities (e.g., crop type, production type, measure types, etc.), in GDIM we leverage the SmartDataModels property “agroVocConcept”, which can be used to annotate an entity with the corresponding Agrovoc term.

3.4 APPLICATION IN PILOTS

The GDIM has been applied already in two of the project pilots, namely the biodiversity and water quality pilots.

Regarding the former, we started the process of harmonisation of datasets from the Group of Earth Observations Biodiversity Observation Network (GEO BON). GEO BON is building up for the pathway to link biodiversity data and metadata to GEOSS, the Global Earth Observation System of Systems. And in particular, GEO BON is focusing its efforts on the implementation and adoption of the Essential Biodiversity Variables (EBVs) and related monitoring guidelines and interoperable data management systems and through targeted capacity building efforts at the national and regional level. GEO BON provides access to the EBV data portal which provides access to 35 EBV datasets⁷. As an example, we started with the Global trends in biodiversity (BES-SIM AIM)⁸ that contains projections from the AIM model from 1900-2050 using LUH2 and SSPs-RCPs, done in the BES-SIM inter-model comparison for IPBES. The dataset, available as NetCDF file, consists of 4 relevant fields: lon, lat, time and entity. In order to represent this dataset in GDIM compliant format, the dataset is mapped into an observation collection, and each simulation becomes a single observation. Additionally, to be compliant with the sosa/ssn pattern, we defined the related feature of interest, observed property, sensor and used procedure. Figure 16 below depicts two observations from the dataset. In the figure, all the classes, properties and relations are coming from the sosa/ssn ontology (in the cross-domain layer of GDIM), except from the lat/long properties which are part of the WGS84 vocabulary (also in the cross-domain layer of GDIM).

⁷ <https://portal.geobon.org/home>

⁸ <https://portal.geobon.org/ebv-detail?id=31>

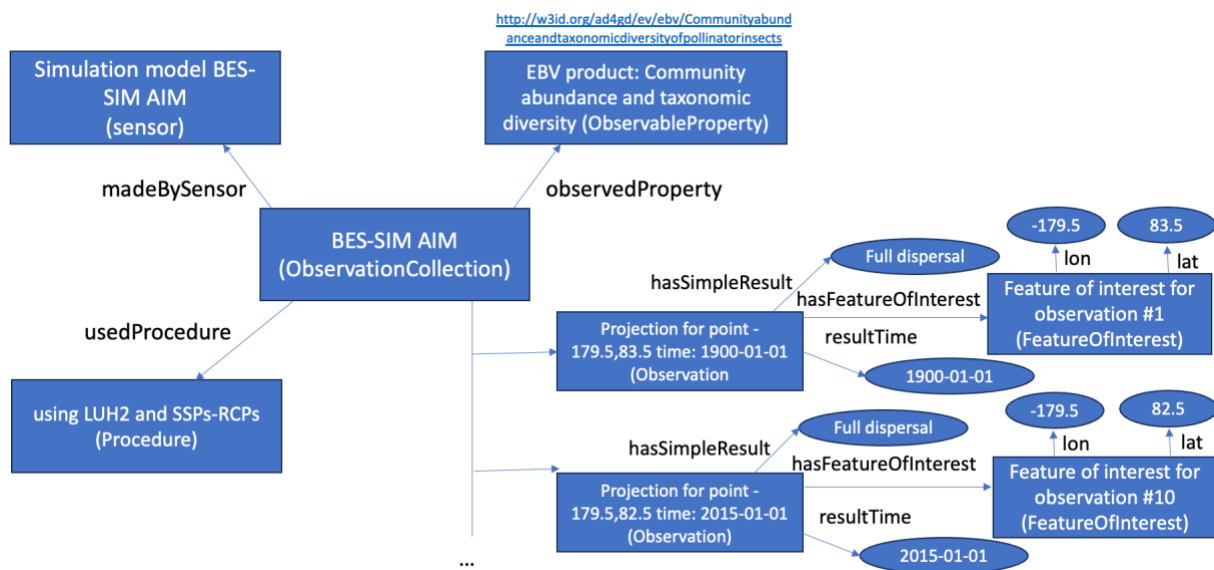


Figure 16. Representation of global trends in biodiversity (BES-SIM AIM) dataset in GDIM

Regarding the water pilot, we took datasets of water quality observations from the Berlin Senate "Wasserportal"⁹. As an example, we took the dataset of monthly mean levels from 21.12.2013 (including monthly minimum and maximum values). The dataset is available as CSV files and includes the fields: Jahr (year), Monat (month), Monatsminimum (minimum value for the month), ZeitpunktMinimum (date of minimum value), Monatsmittelwert (monthly average), Monatsmaximum (maximum value for the month), ZeitpunktMaximum (date of maximum value), Stationsnummer (station number), Stationsname (station name), Gewässer (river), lon, lat.

Similar to the previous example, the whole dataset is represented as an observation collection, and each monthly average row is represented as an observation. Based on the input data, we can also represent the feature of interest, sensor and observed property. Figure 17 below depicts two observations from the dataset. In the figure, all the classes, properties and relations (not in orange) are coming from the *sosa/ssn* ontology (in the cross-domain layer of GDIM), except from the lat/long properties that are part of the *WGS84* vocabulary (also in the cross-domain layer of GDIM), *numericValue* and *unit* that are part of the *QUDT* ontology (also in the cross-domain layer of GDIM), and *Lake* that is part of the *Saref4Water* ontology (in the domain layer of GDIM). The terms in orange are currently not part of GDIM, and thus, they will be defined in a pilot extension module in GDIM or potentially as part of the domain layer. Some of these terms are already identified in different ontologies/vocabularies. For example, the class *WaterLevel* is defined by the *m3* (machine to machine measurement) lite ontology, while *minValue* and *maxValue* are standard *schema.org* properties. However, the observed property *waterLevel* and *maxValueDate* and *minValueDate* have not been identified in any existing resource, and thus they should be defined in a pilot extension module.

⁹ <https://wasserportal.berlin.de/start.php>

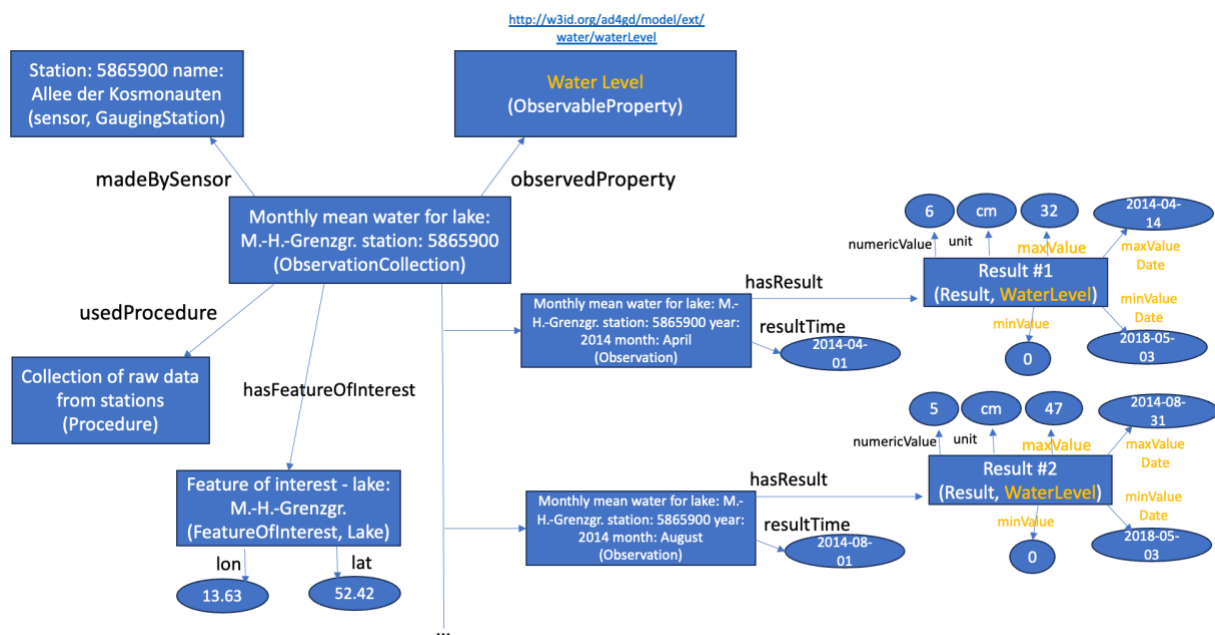


Figure 17. Representation of dataset of monthly mean water levels from 21.12.2013 (from the Berlin Senate "Wasserportal") in GDIM. The terms in orange are not yet in GDIM, and will become part of a pilot extension module.

4 METHODS AND TOOLS FOR DATA HARMONISATION AND INTEGRATION

4.1 OVERVIEW

A workflow for successful multisource data integration (Figure 18) usually starts with the definition of data requirements, including legal constraints and accessibility, according to which data are retrieved. Retrieved data should be then assessed against data requirements and feasibility of proper data processing to make them compliant. Whether this results as effective, the necessary processing should be applied to the data to make them harmonised, first, and to merge them through proper data fusion a second time, whether necessary. Final validation against data requirements and update of metadata, including the documentation of the new lineage is the final step.

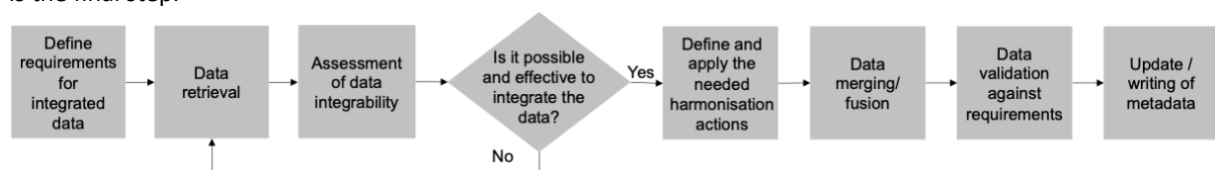


Figure 18. High level workflow for multisource data integration

In the following part of the section, various technologies supporting the different phases of the workflow are described, mainly consisting with Linked Data methods and implementations, to cover data requirements definition and assessment of data against requirements, as well as the application of semantic data processing and mapping (with the support of the Data Preparation and Integration pipelines and the OGC Data Exchange Toolkit), including any needed harmonisation action and final data fusion if needed. Finally, APIs support effective and consistent data retrieval and processing, including geometric processing, while data provenance standards allow storing the new metadata as well as keeping track of the applied processes, to safeguard data transparency and re-usability.

In this line, AD4GD is implementing a data harmonisation and integration approach based on the adoption of Linked Data as a federated layer, combined with the use of knowledge graph technologies, where data is made available and combined according to common ontologies/vocabularies. This approach has been showcased and

demonstrated in different projects, including DEMETER for the agriculture domain, SIEUSOIL for the soil domain, ILIAD for the Ocean domain, OPEN IACS for the CAP framework, and others. As showcased in those projects, this approach supports large scale harmonisation and integration of various data collected from various heterogeneous sources in order to provide an integrated view on top of them.

The approach is based on the implementation of Linked Data pipelines, depicted in Figure 19 which carry out some general tasks, following the best practices and guidelines of Linked Data publication. These tasks include: i) take as input selected datasets that are collected from heterogeneous sources (shapefiles, GeoJSON, CSV, relational databases, RESTful APIs), ii) curate and/or pre-process the datasets when needed, iii) select and/or create/extend the vocabularies (e.g., ontologies) for the representation of data in semantic format, iv) process and transform the datasets into RDF triples according to underlying ontologies, v) perform any necessary post-processing operations on the RDF data, vi) identify links with other datasets, and vii) publish the generated datasets as Linked Data and applying required access control mechanisms.

Regarding the step iii), the selected ontologies/vocabularies will generally depend on the application domain. In the case of AD4GD this will be the Green Deal Information Model (GDIM) introduced in Section 3, which harmonises and aligns relevant cross-domain standards such as Time Ontology, SOSA/SSN, GeoSparql, QUDT, Data Cubes, with domain-specific models such as Darwin Core, Climate and Forecast model, and relevant models from SAREF SmartDataModels, bridging various views on the green deal data and providing a formal representation enabling unambiguous translations between them.

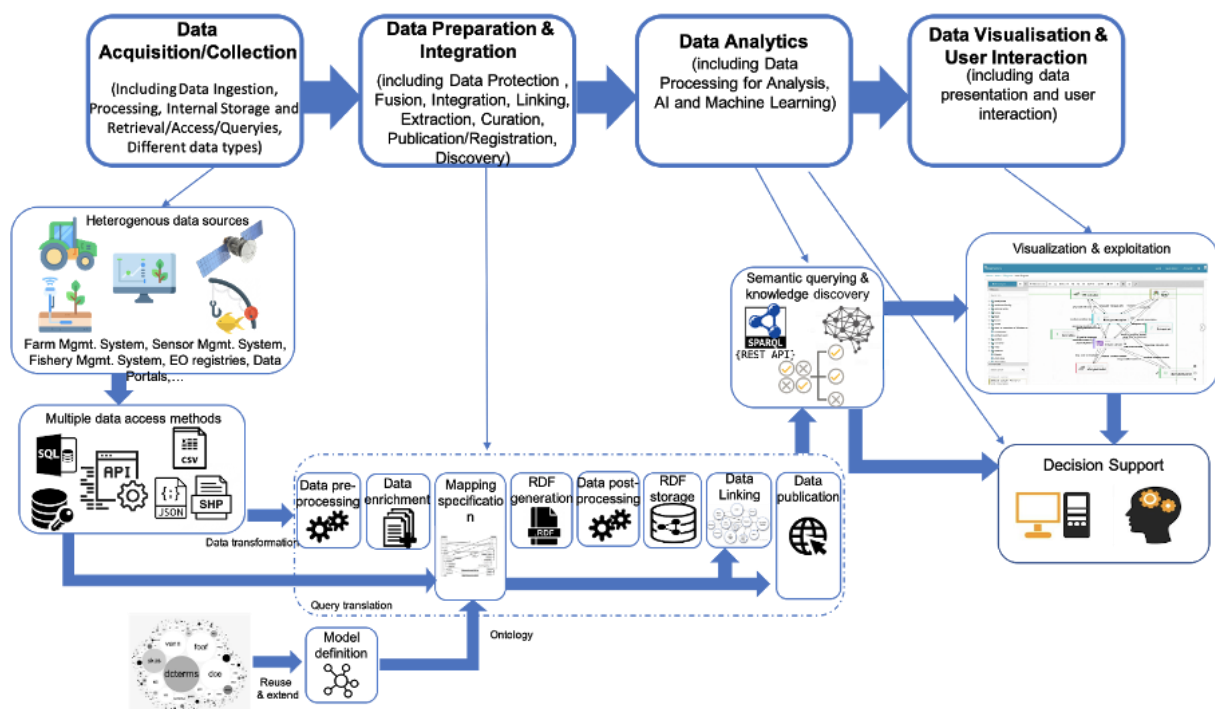


Figure 19. Linked Data pipelines for data harmonisation and integration

The transformation process depends on different aspects of the data like format of the available input data, the purpose (target use case) of the transformation and the volatility of the data (how dynamic is the data). There are broadly two main approaches for making the transformation for a dataset: i) Data upgrade or semantic lifting, which consists of generating RDF data from the source dataset according to mapping descriptions and then storing it in semantic triple store (e.g., Virtuoso); ii) On-the-fly query transformation, which allows evaluating semantic queries over a virtual RDF dataset, by re-writing those queries into source query language according to the mapping descriptions. In this scenario, data physically stays at their source and a new layer is provided to



enable access to it over the virtual RDF dataset. This applies mainly to highly dynamic relational datasets (e.g. sensor data) or RESTful APIs.

In order to enable the access to the harmonised and integrated data, the AD4GD approach considers different possible interfaces. Data can be accessed directly via semantic queries (i.e., (Geo-)SPARQL), but in order to facilitate its usage by service providers, data providers and other users, the pipelines consider the possibility to generate Restful API that can be created on-the-fly from SPARQL queries, and the (semi-)automatic generation of standard APIs, mainly from OGC, such as the SensorThings API or Features API.

4.2 DATA HARMONISATION AND INTEGRATION TOOLS

4.2.1 DATA PREPARATION AND INTEGRATION PIPELINES

The Data Preparation and Integration (DPI) software implements the approach described in Section 4.1. The DPI software leverages and connects different tools, abstracting the different interfaces and implementation details of the underlying tools and applications through simple to use interfaces. Accordingly, the DPI facilitates the exploitation of the pipelines' underlying components via a homogenous layer, enabling users, or other components to launch the whole pipeline, or individual steps. Additionally, the DPI software facilitates access to the integrated data in the following ways:

- via SPARQL queries directly over the semantic data database (triplestore)
- via OGC APIs, which may be generated automatically or semi-automatically, enabling to expose generated data via standard APIs that will be more convenient and require less effort to use by developers and client applications
- via custom APIs that expose pre-defined queries as API access methods. This would not only allow to execute pre-defined methods, but also allow users and developers to define their own queries that are converted on the fly to API methods.

The DPI pipelines are available as a CLI tool, as a Web Service, and include a GUI client application, which will be leveraged, and extended as necessary, in the AD4GD project.

The CLI tool is an ETL software written in Python. It takes care of fetching, extracting, preprocessing, transforming, post-processing, and loading linked data into the triplestore. The interaction with the user is performed through CLI (Command Line Interface) and configuration files. Users can choose from a set of specific pipelines, as well as the generic pipeline. The tool re-uses other existing tools for particular tasks, providing a unified interface over them and connecting them transparently in sequences to implement full pipelines. The project is distributed with a Dockerfile which facilitates setting up the whole environment in a stable and reproducible way. The project distribution also includes a singularity definition file, which enables the deployment and usage of the pipelines in HPC environments. This is particularly useful to process very large datasets, as demonstrated as part of the OPEN IACS project. The docker file, singularity definition file and installation instructions, as well as the usage instructions are clearly documented in the README file. The project is distributed using MIT licence, and is available via: <https://git.man.poznan.pl/stash/projects/DEM/repos/pipelines/browse>.

There are two predefined pipelines, one for processing Farm Accountancy Data Network (FADN) dataset which covers the farms' income and business activities at the EU level. This dataset is particularly relevant for the agriculture use case in DATAMITE. The second is for Land Parcel Identification System (LPIS) datasets, which includes sample mappings for some countries, but users can easily provide a custom mapping, in the form of a simple YAML file, for these or other countries. For these pipelines, the user only provides as input the location of the source dataset (URL or local directory), and run either the whole pipeline from the data collection, to its pre-



processing, mapping, transformation, post-processing and finally to load the transformed data into the configured triplestore, or execute just part of the pipeline (e.g., until the transformation).

The generic pipeline, on the other hand, enables single and more flexible operations on different types of data sources. The pipeline currently supports data sources in the form of Shapefiles, JSON files, CSV files, NetCDF files (ongoing), and relational databases. Similar to FADN and LPIS Pipeline users can choose to provide data through a URL (`url_input`) or just point to the directory (`dir_input`). However, unlike other pipelines, there is no full pipeline method, as every process can be treated as an autonomous step. With that being said, one can still use the generic pipeline to stack multiple processes. The tool can handle transformation for the number of mappings equal to the number of input files (in this scenario, mapping files should have the same base name as data files). In the case of single mapping with multiple input files or single mapping with a single data file, the tool will adjust mapping appropriately to align it with input file/files. The pipeline supports the following processes: pre-process, automatic mapping, transform, post-process, load, and link discovery. A key feature of the general pipeline is that it supports a simple mapping generator, which can create mappings for the pipeline from scratch based on a simple configuration file (in the form of a YAML file) provided by the user. At the moment of writing this generator is being extended to support the use of JSON-LD contexts.

The Web Service wraps the CLI tool and exposes its functionalities through a RESTful API. The architecture of the service is depicted in Figure 20 below. The service is deployed in a PaaS environment, and is connected with other containers to provide storage, security, queue management and other functionalities. Additionally, as shown in the figure, the service relies on a semantic triplestore where the generated data is stored, and there are several other components on top providing simpler user interfaces as well as standard programmatic access to the harmonised data generated by the pipelines. In detail, the DPI service architecture includes the following components:

1. DPI main API: this container is based on the Django and Django-REST framework, and it's the component exposing the full service API that includes the following endpoints:
 - a. `"/users"`: enables to retrieve information about current user (or all users for admin),
 - b. `"/service-description"`: provides short description of the service
 - c. `"/lpis"`: enables the creation, modification and execution of LPIS pipelines to process and transform LPIS data from different countries
 - d. `"/fadn"`: enables the creation, modification and execution of FADN pipelines to process and transform FADN datasets
 - e. `"/generic"`: enables the creation, modification and execution of full generic pipelines, or combinations of individual steps, for the harmonisation of datasets in different formats, including pre-processing, mapping, transformation, post-processing and linking.
 - f. `"/access/queries"`: enables loading a file with query definition into the server to create a new custom API endpoint to access data, and to list all the available ones.
 - g. `"/access/querysets"`: enables to load the input confirmation file for the GRLC component with the list of queries for the custom API, and to list all the available ones.
 - h. `"/access/files"`: enables users to upload files (e.g., input datasets, mappings) to the server and get a shareable link to use in the pipelines.
 - i. `"/jobs"`: enables to retrieve status of pipelines execution
2. Keycloak: This is the identity and access management (IAM) service integrated with the DPI, which is responsible for the authentication and authorization of users. The service is based on the Keycloak technology,
3. Storage: This is a container providing the file storage capabilities in the DPI, that is based on the Minio MinIO, an S3 compatible performant and scalable object store. The storage is used by the service to



store results of executions and also to allow users to upload their own resources (e.g., input datasets, mappings)

4. PostgreSQL: This is the DPI database container, based on PostgreSQL, which is used for the internal execution of the DPI.
5. Pipelines API: This container wraps the CLI tool providing a micro API in a separate container from the main service API
6. Virtuoso: This is a general and separate Virtuoso triplestore service that is used by default by the DPI to store the RDF data generated as part of the pipelines's executions.
7. Pipelines worker: This container is based on Celery, an asynchronous task queue that is based on distributed message passing, which is responsible for running the pipelines's API image as well as the jobs of pipelines's executions.
8. Rabbit queue: This container is the message broker for the pipelines worker, which is based on RabbitMQ.
9. Web app (see Section 4.2.1.1): This Web application is a simple yet complete client graphical user interface (GUI) enabling users to interact directly with the DPI service, e.g., to create, modify and execute pipelines and visualise the results.
10. GRLC (see Section 4.2.1.2): This container deploys a customised and extended version of GRLC service, which enables access to Linked Data (in Virtuoso) via RestFul APIs created on the fly from SPARQL queries.
11. SensorThings API (see Section 4.2.1.3): This container deploys a service providing access to Linked Data (in Virtuoso) via a standard SensorThings API, which is created automatically from the specified datasets (identified as graphs in Virtuoso).

For more information please refer to the service documentation <https://docs.psn.pl/display/DEM/The+architecture+of+dpi-enabler-v2>.

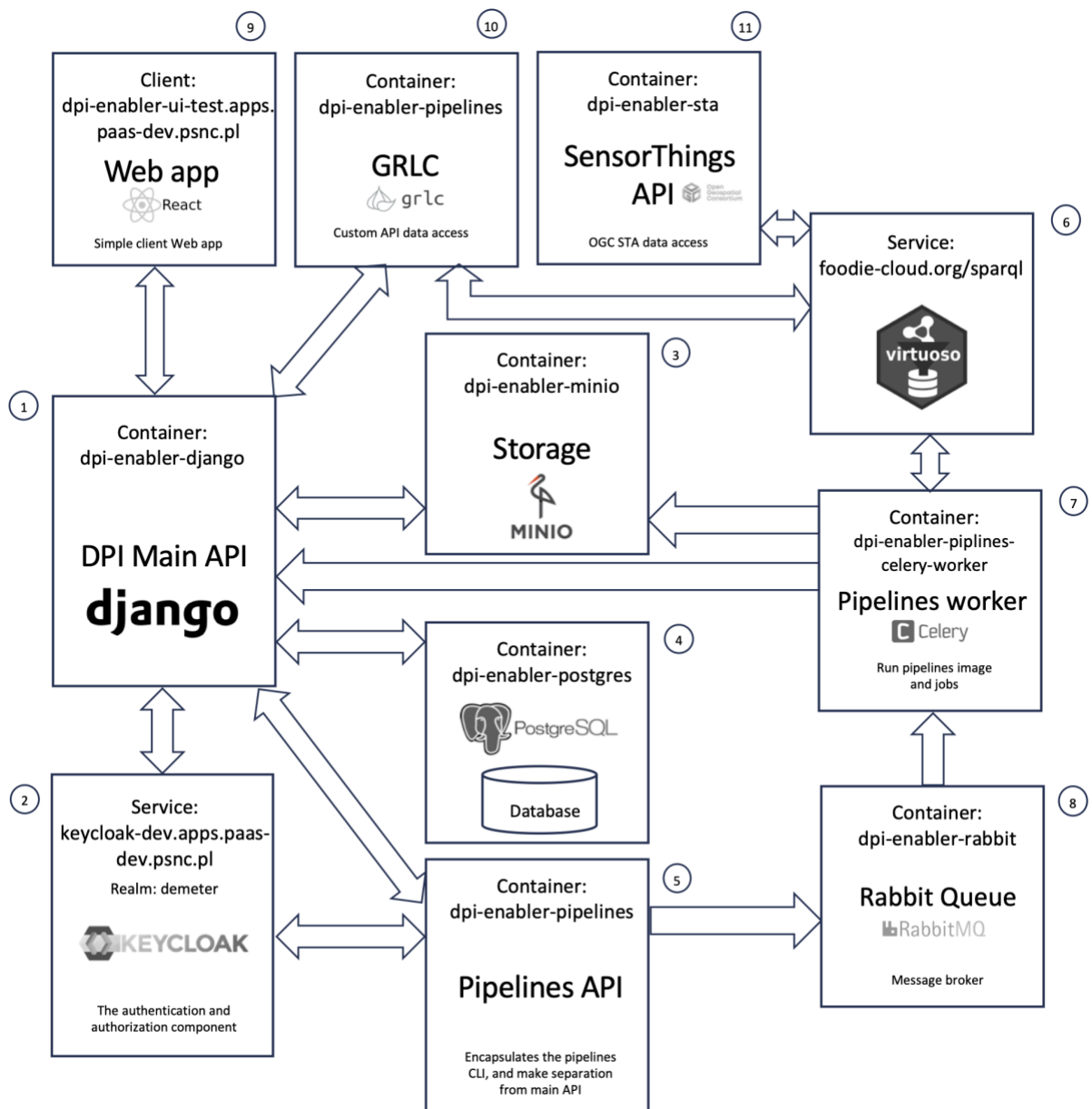


Figure 20. DPI Web Service architecture

The Web Service is publicly available via: <https://dpi-enabler-demeter.apps.paas-dev.psn.pl/api/>, which opens by default the Django REST framework Webpage, and the service also includes a Swagger interface for simpler interaction at <https://dpi-enabler-demeter.apps.paas-dev.psn.pl/api/swagger/>.

4.2.1.1 CLIENT WEB APPLICATION

The GUI client application provides a simple user interface to use the DPI functionalities, allowing users, such as data and service providers, to create and configure their pipelines, execute them and access the results. The application is available via: <https://dpi-enabler-ui-test.apps.paas-dev.psn.pl/>. As depicted in Figure 21 and Figure 22, there are two main sections. The first section allows the user to choose the pipeline type, and to choose whether to create a new pipeline or select an existing one.



Data preparation and Integration Pipelines

Choose pipeline

Generic ▾

Create new or select existing

You need to be logged in to create a new pipeline or modify existing one.

✓ Create new pipeline
Select existing

Figure 21. DPI GUI client application - configuration section

The second section provides the details of the pipeline, which can be updated as needed. As can be seen in the Figure 22, the user can easily upload or point to updated datasets, or mappings to apply the pipeline with different inputs.



JellyFish pipeline full

JellyFish pipeline full - final

Input URL or upload files

<https://dpi-enabler-demeter.apps.paas-dev.psnc.pl/api/access/files/79913a05-c683-4b24-9bac-e1acdb37213a/download>

 Drag and drop or [browse](#) files here...

Settings

Input data type

CSV

Process:

- Pre-process input data
- Generate mapping
- Transform data

Provide mapping file (RDF or SPARQL) as:

- i) URL to ZIP archive containing mapping file(s). The ZIP archive may contain one or multiple RDF files (e.g., ttl) or a SPARQL file with multiple constructs
- ii) Upload RDF mapping file(s)
- iii) Upload SPARQL mapping file

<https://dpi-enabler-demeter.apps.paas-dev.psnc.pl/api/access>

 Drag and drop or [browse](#) files here...

- Mapping is sparql query
- Post-process data
 - Transform to RDF turtle (ttl)
- Load data into triplestore

<http://w3id.org/iliad/dataset/jf-pilot-sample>

- Reload graph
- Graph per dump
- Link discovery

Update pipeline

Execute pipeline

Figure 22. DPI GUI client application - settings section



4.2.1.2 EXTENDED GRLC SERVICE

GRLC¹⁰ is a service that takes SPARQL queries (stored in a GitHub repository, in your local filesystem, or listed in a URL), and translates them to Linked Data Web APIs. This allows service and application developers to easily access Linked Data (in Virtuoso) without any knowledge of SPARQL. The GRLC service deployed on top of DPI is a customised and extended version¹¹ of the open source project, which supports additional custom tags in JSON generation, generation of JSON-LD, complex values of properties, etc. In order to specify the queries, the users can point the grlc service to a GitHub repository or provide the URL of the configuration file. For the latter, the the DPI main API provides the “/access/queries” endpoint to upload the SPARQL queries and the “/access/querysets” endpoint to upload a query set configuration file (including a short description and the link to SPARQL queries of the target Restful API), which returns the new swagger API entry point URL.

Two examples:

- Swagger API generated from queries in a GitHub repository: <https://grlc-dpi-enabler-demeter.apps.paas-dev.psncl.pl/api-git/ILIAD-ocean-twin/JF-API/>
- Swagger API generated from a configuration file URL: <https://grlc-dpi-enabler-demeter.apps.paas-dev.psncl.pl/api-url/?specUrl=https://dpi-enabler-demeter.apps.paas-dev.psncl.pl/api/access/querysets/c84582e9-7402-4b33-9d56-4e3ef10d2926/raw>

4.2.1.3 SENSORTHINGS API GENERATION SERVICE

The SensorThings API (STA) generation is the latest service deployed on top of the DPI pipelines. It enables access to the Linked Data (in Virtuoso) generated by the pipelines via a standard STA, which is created automatically from the specified dataset(s) (identified as graph(s) in Virtuoso). The dataset(s) should represent observations/measurements that are represented according to the GDIM, i.e., following the SOSA/SSN standard approach. The observations can be grouped in observations collections, and should include (at the observation or collection level) the observed property(ies), feature(s) of interest, sensor(s) making the observation. Also, each observation should include its result (with result and phenomenon time).

To specify the source graph(s) to generate a new STA, the service require the creation of a “pilot”. One pilot can have associated one or more graphs, where at least one contains observations. Other graphs can contain for example codelists of observed properties, or other entities used in the observations.

The entry point to this service is <https://sta-demeter.apps.paas-dev.psncl.pl/>

The service provides only 5 endpoints to create or retrieve a new pilot, and to retrieve, modify or delete a specific pilot, as depicted in Figure 23. In detail a pilot is created by specifying a name, a description, and the graph(s). As a result, the service returns the root STA URL and the Swagger STA URL. The other methods allow to retrieve all or individual pilots, modify them or delete them. Figure 24 lists the configuration of one example pilot. As it can be seen the STA URL for this pilot is: <https://sta-demeter.apps.paas-dev.psncl.pl/AD4GD-water-5865900/api/v1.0/> and the Swagger STA URL is: <https://sta-demeter.apps.paas-dev.psncl.pl/AD4GD-water-5865900/api/v1.0/docs> (See Section 5.4 for further details of this and other examples).

¹⁰ <https://github.com/CLARIAH/grlc>

¹¹ <https://git.man.poznan.pl/stash/projects/DEM/repos/grlc/browse>



OGC SensorThings API 1.0.0 OAS 3.1

/openapi.json

Pilots

GET	/pilots	Read Pilots
POST	/pilots	Create Pilot
GET	/pilots/{name}	Read Pilot
PUT	/pilots/{name}	Update Pilot
DELETE	/pilots/{name}	Delete Pilot

Figure 23. DPI STA generation service entry point

```
{
  "name": "AD4GD-water-5865900",
  "description": null,
  "graphs": [
    "http://w3id.org/ad4gd/dataset/pilot/water/5865900_wasserstand_mw_21_12_2013"
  ],
  "rootURL": "https://sta-demeter.apps.paas-dev.psn.pl/AD4GD-water-5865900/api/v1.0/",
  "swaggerURL": "https://sta-demeter.apps.paas-dev.psn.pl/AD4GD-water-5865900/api/v1.0/docs"
},
```

Figure 24. Example pilot in STA generation service

4.2.2 OGC DATA EXCHANGE TOOLKIT

In order to properly achieve interoperability, full compliance with standards and data exchange formats is essential. This means that not only the provider and the consumer (be they human or machine) need to implement the protocols needed for successful communication, but the data itself being transferred has to follow a set of specifications, as described in Section 3. When dealing with a sizable (and variable) number of potential providers and/or consumers, the need for tools that can automatically validate the syntax (is the data formatted correctly?), value format (do the values or fields have the right data types?) and constraints (are all the values within their expected ranges? Are geographical/geometric/geospatial representations valid?), completeness (is all the required information present?), and coherence (do the data and the relationships in it make sense?) throughout the lifecycle of the data arises. Additionally, consumers may desire to access datasets that are not readily available in any of the data formats they support, prompting the need for including one or more transformation steps to their data retrieval process, but resulting in increased interoperability across the whole platform.

Therefore, an implementation of a GDDS should make the tooling for building such data pipelines available to its data consumers, through design of interoperability standards that conform to and exploit the underlying data



models. These designs can be implemented with the OGC Data Exchange Toolkit, a set of software tools, templates, and pipeline definitions for documenting and validating modular data models and implementations.

The OGC Data Exchange Toolkit uses Continuous Integration (CI; frequently merging contributions to a data or code repository) / Continuous Deployment (CD; frequent, automatic delivery of results or end products) / Continuous Testing (CD, execution of automated testing workbenches for error detection when new code, data or metadata are available) data workflows. This toolkit, which has emerged from experiences implementing modular specifications and linking specifications for APIs, schemas and ontologies, has been further tested by developing CT pipelines in the context of the European Union's Horizon2020 CHEK project (<https://chekdbp.eu>). It will be enhanced with AD4GD-specific functionality for testing integration with Dataspace environments. It can process a number of input data formats, apply a combination of simple, atomic transformations on the data, derive linked data representations of it, perform different types of validations (schema, semantic relationships, etc.) on it, entail new metadata and semantic links for it, and store the results in semantic databases. A collection of such ready-to-use data workflows will be made available to the GDDS data providers and consumers to guarantee compliance both with the GDIM and with the specific data and API formats used.

Semantic interoperability can also be enhanced by performing entailment operations where new data or metadata is inferred for a dataset. This enables consumers that can process only a limited set of data profiles (models or ontologies) to ingest additional types of sources or datasets, via the creation of new metadata and relationships in the latter. Entailment regimes between commonly used profiles should be applied to any linked data generated in the context of the GDDS, which, as mentioned before, can be effectively implemented by using the aforementioned OGC Data Exchange Toolkit. Once the data is enriched by inferring information through linked data mechanisms, clients can use different strategies, to detect the profiles available for a dataset or piece of data, and request the one that is of interest to them. For example, clients can employ content negotiation by profile (<https://www.w3.org/TR/dx-prof-conneg/>, PROF-CONNEX), where, by using different headers in HTTP requests and replies, a server can provide a list of the profiles that an entity is compliant with. The client can then choose the one that best suits its needs, similar to traditional content negotiation by MIME type. These mechanisms can be applied both on the data itself (when it is in semantic format) and on the catalogue metadata.

A high-level view of the data workflows that can be created with the OGC Data Exchange Toolkit can be found in Figure 25 in which a number of different types of data formats can be consumed at the beginning, resulting in its linked data representation according to additional profiles at the end.

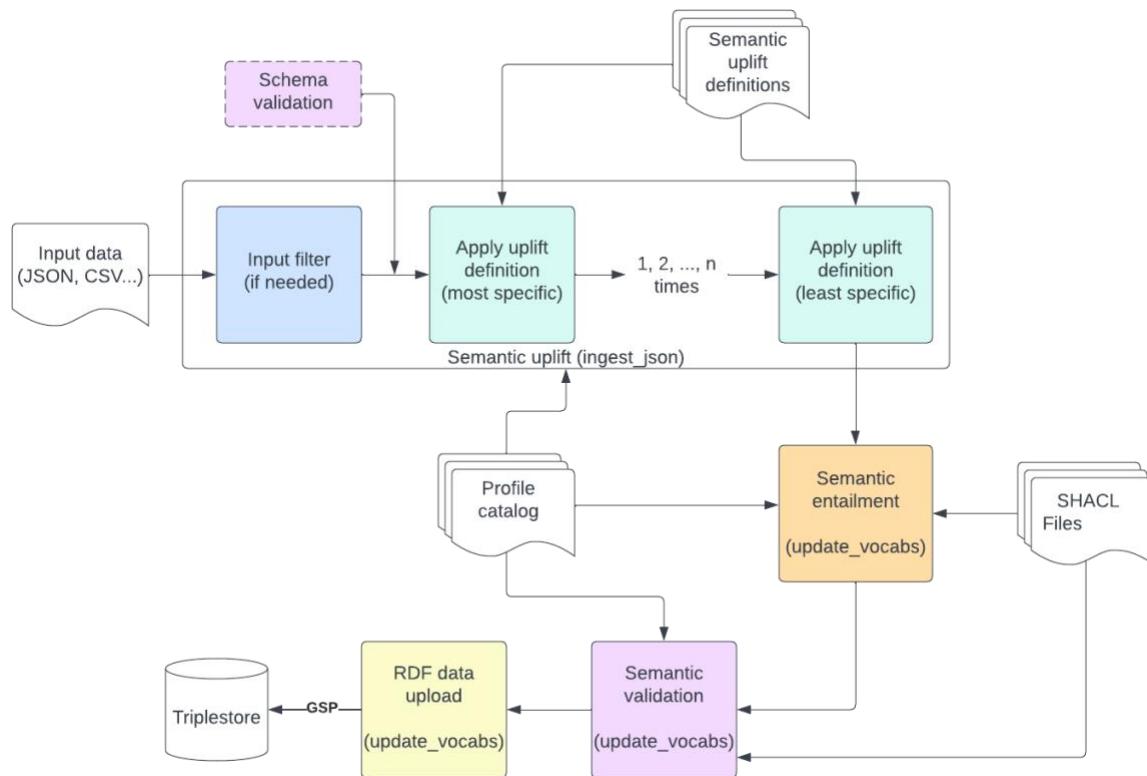


Figure 25. High-level view of the data workflows that can be created with the OGC Data Exchange Toolkit

Another issue inherent to the use, reuse and/or combination of different data sources is traceability. A user ingesting a dataset obtained in the scope of a GDDS will most certainly be interested in the specific data sources employed for its creation, as well as in the potential transformations that may have been used along the way. The PROV-O ontology (<https://www.w3.org/TR/prov-o/>), which defines an OWL model for provenance description in RDF graphs, including information about the entities (e.g. documents), agents (persons, software tools...) and activities involved in producing data, should be used in every step of the data processing pipelines to attach provenance paradata to the semantic metadata outputs. Similarly, for JSON payloads, the OGC is in the process of defining a Location Building Block (documentation, JSON Schema, JSON-LD context, SHACL shapes...) aligned with PROV-O (<https://github.com/ogcincubator/bblock-prov-schema>).

4.2.3 SEMANTIC ANNOTATION OF TABLES AND ITS RELATION WITH STA

This section introduces the two ways to semantically annotate data tables that have been tested in AD4GD. While the examples focus on environmental data, the method can be applied to any use case in AD4GD. The objective of this section is not to describe a full data model for a particular data type but to describe how some field (columns) in a table can be described as “variables” that have been measured (what has been measured and its units of measure). The objective is to facilitate a mapping (and import/export) from tables to the STA data model. The first method describes how to annotate a table using an extension of JSON schema and the second method describes how to annotate a CSV table using CSVW. The last method included a tool that is capable of importing a CSV table into STApplus automatically due to this annotation.

Data are generally sequences of numbers and text in a particular structure. A very common structure is a table where rows represent objects and columns represent attributes characterising those objects. Commonly columns are provided with a name and rows are provided with an identifier. One typical example of this is data



distributed in CSVs but other common formats such as shapefiles use directly that and more recently GeoJSON can be mapped into the same data structure (each feature can be expressed as an object with several attributes called properties). To make data reusable the structure of the data needs to be better defined. In particular the meaning of the attributes of each object should be clearly specified.

This project proposes to do that by mapping the tabular data structure into a JSON file and using an extended version of JSON Schema language as a method to describe the attributes of each object. The idea of using JSON Schema is not new and has been proposed in <https://www.synvert-tcm.com/blog/json-schema-vocabulary/>.

Let assume that a tabular dataset such as this one:

Table 1. Example tabular data of simple observations

id	temp	date
1	23.1	2023-08-03 10:30:00Z
2	31.5	2023-08-03 10:30:00Z

If we assume that the first row is a header that describes the columns and the other rows are objects, we can easily transform it into a JSON file:

```
{
  "objects": [
    {
      "id": 1,
      "temp": 23.1,
      "date": "2023-08-03T10:30:00Z"
    },
    {
      "id": 2,
      "temp": 31.5,
      "date": "2023-08-04T10:30:00Z"
    }
  ]
}
```

This JSON file can be now described in a JSON Schema:

```
{
  "$schema": "http://json-schema.org/draft-06/schema#",
  "type": "object",
  "properties": {
    "objects": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "id": {
            "description": "Identifier",
            "type": "integer"
          },
          "temp": {
            "description": "Temperature",
            "type": "number"
          },
          "date": {
            "description": "Date and time of measurement",
            "type": "string",
            "format": "date-time"
          }
        }
      }
    }
  }
}
```



```

    }
}

```

JSON Schema is helping to provide some extra information. Now we know that “id” means “identifier” and “temp” means “Temperature” but JSON Schema alone cannot provide enough information by itself. To completely describe the temperature we need to extend the JSON schema into that:

```

{
  "$schema": "http://json-schema.org/draft-06/schema#",
  ...
  "temp": {
    "description": "Temperature",
    "type": "number",
    "definition": "http://vocabs.lter-
europe.net/EnvThes/22035",
    "UoM": "Celsius",
    "UoMSymbol": "C",
    "UoMDefinition":
"http://qudt.org/vocab/unit/DEG_C"
  },
  ...
}

```

Now we are semantically annotating the “temp” attribute to communicate the exact meaning of it thanks to the link to a “definition” that point to a vocabulary:

PREFERRED TERM

air temperature

DEFINITION

[GEMET] The temperature of the atmosphere which represents the average kinetic energy of the molecular motion in a small region and is defined in terms of a standard or calibrated thermometer in thermal equilibrium with the air.

BROADER CONCEPT

variable

ENTRY TERMS

temperature of air

SCOPE NOTE

US LTER controlled vocabulary

CREATOR

herbert.schentz@umweltbundesamt.at

IN OTHER LANGUAGES

درجة حرارة الهواء

Arabic

Температура на въздуха

Bulgarian

temperatura zraka

Croatian

teplota vzduchu

Czech

Figure 26. Air temperature definition in EnvThes Thesaurus. URL <http://vocabs.lter-europe.net/EnvThes/22035>

It also provides us with the units of measurement of the variable and its definition:



QUDT

unit:DEG_C

URI: http://qudt.org/vocab/unit/DEG_C

Type

qudt:DerivedUnit
qudt:Unit

Description

Celsius, also known as centigrade, is a scale and unit of measurement for temperature. It can refer to a specific temperature on the Celsius scale as well as a unit to indicate a temperature interval, a difference between two temperatures or an uncertainty. This definition fixes the magnitude of both the degree Celsius and the kelvin as precisely 1 part in 273.16 (approximately 0.00366) of the difference between absolute zero and the triple point of water. Thus, it sets the magnitude of one degree Celsius and that of one kelvin as exactly the same. Additionally, it establishes the difference between the two scales' null points as being precisely 273.15 °C.

Properties

qudt:applicableSystem
sou:CGS
sou:SI
qudt:conversionMultiplier
1.0
qudt:conversionOffset
273.15
qudt:dbpediaMatch

Figure 27. Celsius unit definition in QUDT units vocabulary. URL https://qudt.org/vocab/unit/DEG_C

The proposed extension can be formally defined as proposed in the Appendix D of the 2019-09 version of the JSON schema <https://json-schema.org/draft/2019-09/json-schema-core.html#rfc.appendix.D> and clarified here: <https://stackoverflow.com/questions/64138556/how-do-i-define-my-own-json-schema-keyword-and-vocabulary>. Four new fields need to be added to the JSON schema.

Table 2. JSON schema fields for the semantic annotation of tables

Name	Description	Type
<i>definition</i>	A URI that defines the observedProperty or variable. You may find the right definitions in https://qudt.org/2.1/vocab/quantitykind , http://vocabs.lter-europe.net/EnvThes , https://www.eea.europa.eu/help/glossary/eea-glossary or http://vocab.nerc.ac.uk/standard_name	URI
<i>UoM</i>	Units of measurement of the attribute.	String
<i>UoMSymbol</i>	Symbol of the units of measurement of the attribute	String
<i>UoMDefinition</i>	A URI that defines the units of measurement of the observedProperty or variable. You may find the right definitions in https://qudt.org/2.1/vocab/unit or http://vocab.nerc.ac.uk/collection/P06/current/	URI

This can be specified in a metaschema like this:



```

{
  "title": "GeoJSON properties meaning schema",
  "$schema": "http://json-schema.org/draft/2019-09/schema#",
  "$id": "https://meaning.ad4gd.eu/json-meta/meaning",
  "$vocabulary": {
    "https://json-schema.org/draft/2019-09/vocab/core": true,
    "https://json-schema.org/draft/2019-09/vocab/applicator": true,
    "https://json-schema.org/draft/2019-09/vocab/validation": true,
    "https://json-schema.org/draft/2019-09/vocab/meta-data": true,
    "https://json-schema.org/draft/2019-09/vocab/format": false,
    "https://json-schema.org/draft/2019-09/vocab/content": true,
    "https://meaning.ad4gd.eu/json-meta/meaning": false
  },
  "$recursiveAnchor": true,
  "allOf": [
    {
      "$ref": "https://json-schema.org/draft/2019-09/schema"
    },
    {
      "$ref": "#/definitions/AttributeDescription"
    }
  ],
  "definitions": {
    "AttributeDescription": {
      "title": "GeoJSON meaning vocabulary meta-schema",
      "type": "object",
      "$comment": "The properties that define each attribute
can be de ones defined below or properties from JSON schema itself if indicated in this
comment. Common properties such as 'description' can be used. Others from 'string'
(https://json-schema.org/understanding-json-schema/reference/string.html) or number can be
useful (https://json-schema.org/understanding-json-schema/reference/numeric.html).",
      "properties": {
        "definition": {
          "description": "A URI that defines the
observedProperty or the variable. You may find the right definitions in
https://qudt.org/2.1/vocab/quantitykind, http://vocabs.lter-europe.net/EnvThes or
https://www.eea.europa.eu/help/glossary/eea-glossary.",
          "type": [ "string", "null" ],
          "format": "uri"
        },
        "UoM": {
          "description": "Units of measurement of
the attribute.",
          "type": [ "string", "null" ]
        },
        "UoMSymbol": {
          "description": "Symbol of the units of
measurement of the attribute.",
          "type": [ "string", "null" ]
        },
        "UoMDefinition": {
          "description": "A URI that defines the
units of measurement of the observedProperty or variable. You may find the right definitions
in https://qudt.org/2.1/vocab/unit",
          "type": [ "string", "null" ],
          "format": "uri"
        }
      }
    }
  }
}

```

Then, the original JSON schema is modified like this.

```

{
  "$schema": "https://meaning.ad4gd.eu/json-meta/meaning",
  "type": "object",
  "properties": {
    "objects": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "id": {

```



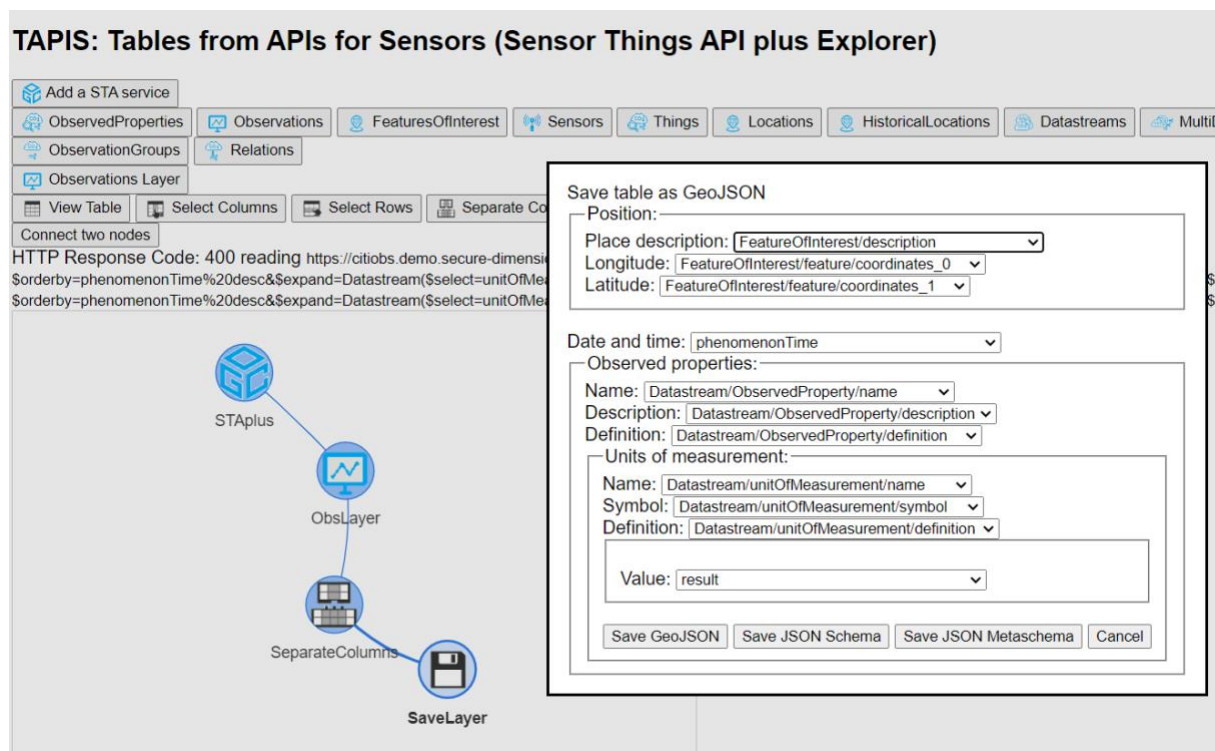
```

    "description": "Identifier",
    "type": "integer"
  },
  "temp": {
    "description": "Temperature",
    "type": "number",
    "definition": "http://vocabs.lter-
europe.net/EnvThes/22035",
    "UoM": "Celsius",
    "UoMSymbol": "C",
    "UoMDefinition":
"http://qudt.org/vocab/unit/DEG_C"
  },
  "date": {
    "description": "Date and time of
measurement",
    "type": "string",
    "format": "date-time"
  }
}
}
}
}
}

```

A JSON schema and metaschema like this one can be validated modern validators such as: <https://www.jsonschemavalidator.net/>

This approach has been implemented in the SensorThings API plus reader that we call “TAPIS: Tables from APIs for Sensors” (<https://github.com/joanma747/TAPIS>). TAPIS is capable to generate the extended JSON schema and the metaschema when exporting a dataset into a GeoJSON file:



This concept is also used in the MiraMon Map Browser. In the queries by location, the representation of the name and units of measure of the attributes are now links that can be clicked to know more about the definitions of the observedProperties and the units of measurement of each attribute:

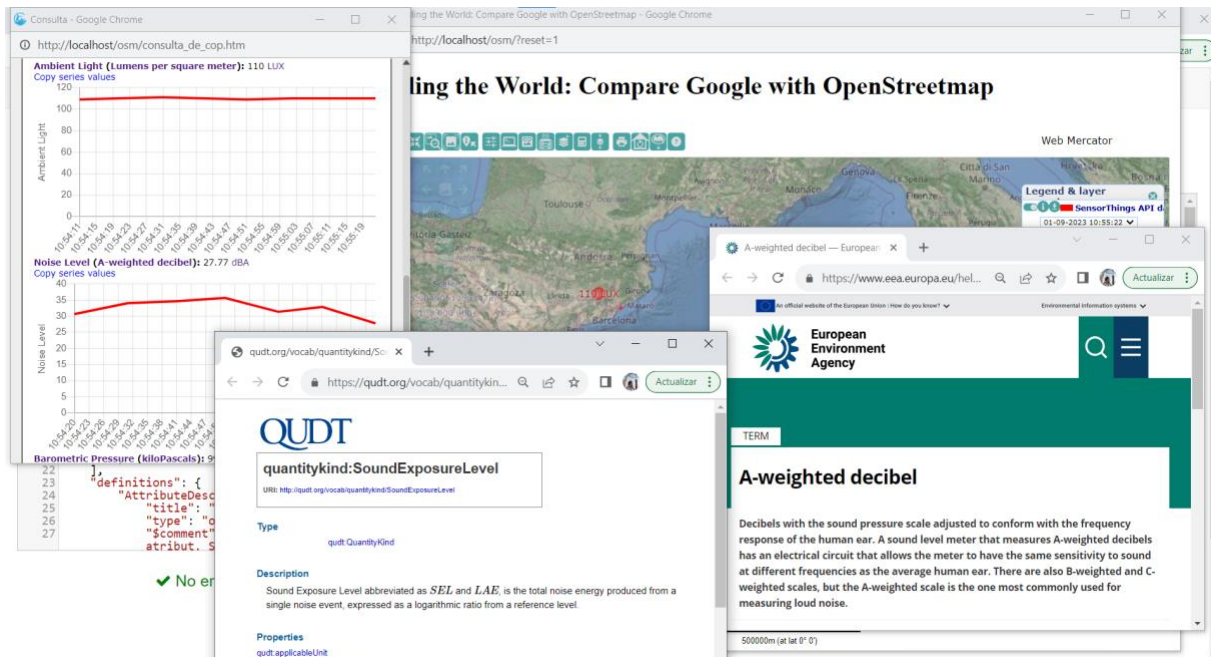


Figure 29. MiraMon Map Browser showing observations with linked units of measures definitions opened in different tabs

For the CSV files, there is a relatively new proposed format to describe its columns. The proposal comes from a company called Swirrl (now part of TPXImpact) called CSVW (<https://github.com/Swirrl/csvw.org> and <https://csvw.org/>). The proposal consists in adding a companion file to the CSV files (in JSON format) that describes how to read the CSV in terms of delimiters etc (in a section called dialect) and each column of the CSV. Let's imagine the following CSV (presented in MS Excel):

	A	B	C	D	E	F	G	H	I	J	K	L
1	phenomenonTime	place	long	lat	temp	RH	light	pres	noise	PM1	PM25	PM10
2	2023-09-06T18:02:03Z	Next to Carde	1.78108	41.78143	29.47	52.22	82	99.11	31.64	1	5	5
3	2023-09-00T00:00:01Z	Next to Carde	1.78108	41.78143	31.47	62.22	92	99.22	41.64	2	6	7

Figure 30. Example tabular data of complex observations

The current proposal allows for defining each column in terms of datatype, description and definition URL. The CSVW file will look like this:

```
{
  "tableSchema": {
    "columns": [
      {
        "name": "phenomenonTime",
        "datatype": "number",
        "propertyUrl":
"http://www.opengis.net/def/ogc/PhenomenonTime"
      },
      {
        "name": "place",
        "datatype": "string",
        "propertyUrl":
"http://www.opengis.net/def/CaLALThe/4.0/PlaceName"
      },
      {
        "name": "long",
        "datatype": "number",
        "propertyUrl":
"http://www.w3.org/2003/01/geo/wgs84_pos#long"
      },
      {
        "name": "lat",
```




```

        "datatype": "number",
        "propertyUrl":
"http://www.w3.org/2003/01/geo/wgs84_pos#lat"
    },
    {
        "name": "temp",
        "datatype": "number",
        "titles": "Air Temperature",
        "propertyUrl": "http://vocabs.lter-
europe.net/EnvThes/22035",
    },
    {
        "name": "RH",
        "datatype": "number",
        "titles": "Relative Humidity",
        "propertyUrl": "http://vocabs.lter-
europe.net/EnvThes/21579",
    },
    {
        "name": "light",
        "datatype": "number",
        "titles": "Ambient Light",
        "propertyUrl":
"http://qudt.org/vocab/quantitykind/LuminousExposure",
    },
    {
        "name": "pres",
        "datatype": "number",
        "titles": "Barometric Pressure",
        "propertyUrl": "https://vocabs.lter-
europe.net/EnvThes/22060",
    },
    ...
]
},
"dialect": {
    "header": true,
    "delimiter": ";"
}
}

```

The complete format is described here <https://w3c.github.io/csvw/syntax/#bib-tabular-metadata>. The properties of the array “columns” is described here: <https://www.w3.org/TR/2015/REC-tabular-metadata-20151217/>

In it we can find the “propertyUrl” that is the equivalent to “definition” in OGC SensorThings API¹². The CSVW specification acknowledges the need for a unit of measurement but it does not make any concrete proposal on how to do it. Instead it suggests using the “unitMeasure” proposed by <https://www.w3.org/TR/vocab-data-cube/>. Since it was necessary for the GDIM to represent more details on the UoM, we finally opted by extending the CSVW format by including 3 new elements based on “unitMeasure using a similar style (see them in italics in the table):

¹² <https://www.ogc.org/standard/sensorthings/>



Table 3. Proposed attributes extending CSVW to describe units of measures

Name	Description	Type	Equivalent to
propertyUrl	A URI that defines the observedProperty or variable. You may find the right definitions in https://qudt.org/2.1/vocab/quantitykind , http://vocabs.lter-europe.net/EnvThes , https://www.eea.europa.eu/help/glossary/eea-glossary , or http://vocab.nerc.ac.uk/standard_name	URI	definition
unitMeasureTitles	Units of measurement of the attribute.	String	UoM
unitMeasureSymbol	Symbol of the units of measurement of the attribute	String	UoMSymbol
unitMeasureUrl	A URI that defines the units of measurement of the observedProperty or variable. You may find the right definitions in https://qudt.org/2.1/vocab/unit or http://vocab.nerc.ac.uk/collection/P06/current/	URI	UoMDefinition

This is the final CSVW example.

```
{
  "tableSchema": {
    "columns": [
      {
        "name": "phenomenonTime",
        "datatype": "number",
        "propertyUrl":
"http://www.opengis.net/def/ogc/PhenomenonTime"
      },
      {
        "name": "place",
        "datatype": "string",
        "propertyUrl":
"http://www.opengis.net/def/CaLALThe/4.0/PlaceName"
      },
      {
        "name": "long",
        "datatype": "number",
        "propertyUrl":
"http://www.w3.org/2003/01/geo/wgs84_pos#long"
      },
      {
        "name": "lat",
        "datatype": "number",
        "propertyUrl":
"http://www.w3.org/2003/01/geo/wgs84_pos#lat"
      },
      {
        "name": "temp",
        "datatype": "number",
        "titles": "Air Temperature",
        "propertyUrl": "http://vocabs.lter-
europe.net/EnvThes/22035",
        "unitMeasureTitles": "Celsius",
        "unitMeasureSymbol": "C",
        "unitMeasureUrl": "https://qudt.org/vocab/unit/DEG_C"
      },
      {
        "name": "RH",
        "datatype": "number",
        "titles": "Relative Humidity",
        "propertyUrl": "http://vocabs.lter-
europe.net/EnvThes/21579",
        "unitMeasureTitles": "Percentage",
        "unitMeasureSymbol": "%",
        "unitMeasureUrl": "https://qudt.org/vocab/unit/PERCENT"
      }
    ]
  }
}
```



```

    },
    {
      "name": "light",
      "datatype": "number",
      "titles": "Ambient Light",
      "propertyUrl":
"https://qudt.org/vocab/quantitykind/LuminousExposure",
      "unitMeasureTitles": "Lumens per square meter",
      "unitMeasureSymbol": "LUX",
      "unitMeasureUrl": "https://qudt.org/vocab/unit/LUX"
    },
    {
      "name": "pres",
      "datatype": "number",
      "titles": "Barometric Pressure",
      "propertyUrl": "https://vocabs.lter-
europe.net/EnvThes/22060",
      "unitMeasureTitles": "kiloPascals",
      "unitMeasureSymbol": "kPa",
      "unitMeasureUrl": "https://qudt.org/vocab/unit/KiloPA"
    },
    {
      "name": "noise",
      "datatype": "number",
      "titles": "Noise Level",
      "propertyUrl":
"https://qudt.org/vocab/quantitykind/SoundExposureLevel",
      "unitMeasureTitles": "A-weighted decibel",
      "unitMeasureSymbol": "dBA",
      "unitMeasureUrl":
"https://www.eea.europa.eu/help/glossary/eea-glossary/a-weighted-decibel"
    },
    {
      "name": "PM1",
      "datatype": "number",
      "titles": "Particulate matter with particulate matter
with an average aerodynamic diameter of up to 1 micrometers",
      "propertyUrl": "https://www.iqair.com/us/newsroom/pm1",
      "unitMeasureTitles": "Microgram per cubic meter",
      "unitMeasureSymbol": "ug/m3",
      "unitMeasureUrl": "https://qudt.org/vocab/unit/MicroGM-
PER-M3"
    },
    {
      "name": "PM25",
      "datatype": "number",
      "titles": "Particulate matter with particulate matter
with an average aerodynamic diameter of up to 2.5 micrometers",
      "propertyUrl":
"https://www.eea.europa.eu/help/glossary/eea-glossary/pm2.5",
      "unitMeasureTitles": "Microgram per cubic meter",
      "unitMeasureSymbol": "ug/m3",
      "unitMeasureUrl": "https://qudt.org/vocab/unit/MicroGM-
PER-M3"
    },
    {
      "name": "PM10",
      "datatype": "number",
      "titles": "Particulate matter with particulate matter
with an average aerodynamic diameter of up to 10 micrometers",
      "propertyUrl":
"https://www.eea.europa.eu/help/glossary/eea-glossary/pm10",
      "unitMeasureTitles": "Microgram per cubic meter",
      "unitMeasureSymbol": "ug/m3",
      "unitMeasureUrl": "https://qudt.org/vocab/unit/MicroGM-
PER-M3"
    }
  ],
  "dialect": {
    "header": true,
    "delimiter": ";"
  }
}

```



This approach has been implemented in the SensorThings API plus reader that we call “TAPIS: Tables from APIs for Sensors” (<https://github.com/joanma747/TAPIS>). TAPIS is capable to read a CSV file and present it as a table where the titles of the columns use the links to the propertyUrl and the show the unitMeasureSymbol linked to the unitMeasureUrl:

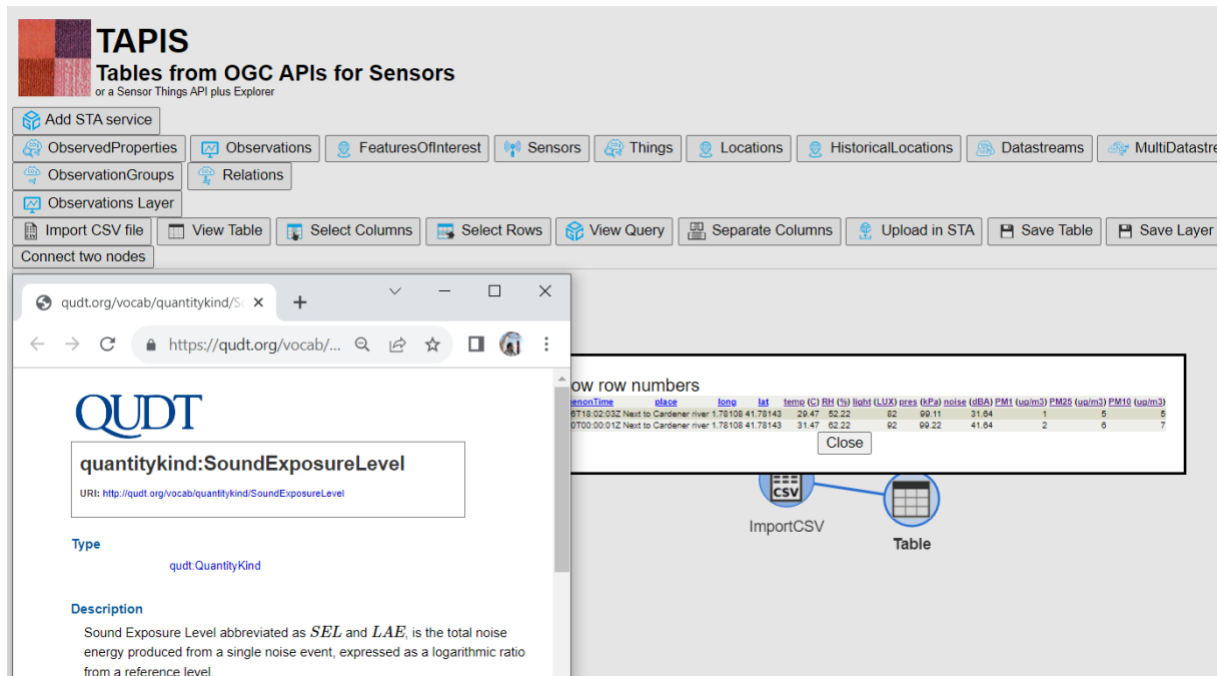


Figure 31. TAPIS STA+ reader - showing a table with links to propertyURL and unitMeasureUrl

Thanks to this annotation TAPIS is able to send this observations to a STApplus instance automatically:

Upload table as STA observations

- Each row represents a feature of interest with several observed properties
- Several observed properties of the same feature in a single column and in multiple rows

Position:

Place description:

Longitude:

Latitude:

Date and time:

STA service URL:

Figure 32. Upload table as STA observations from TAPIS STA+ reader (submit)

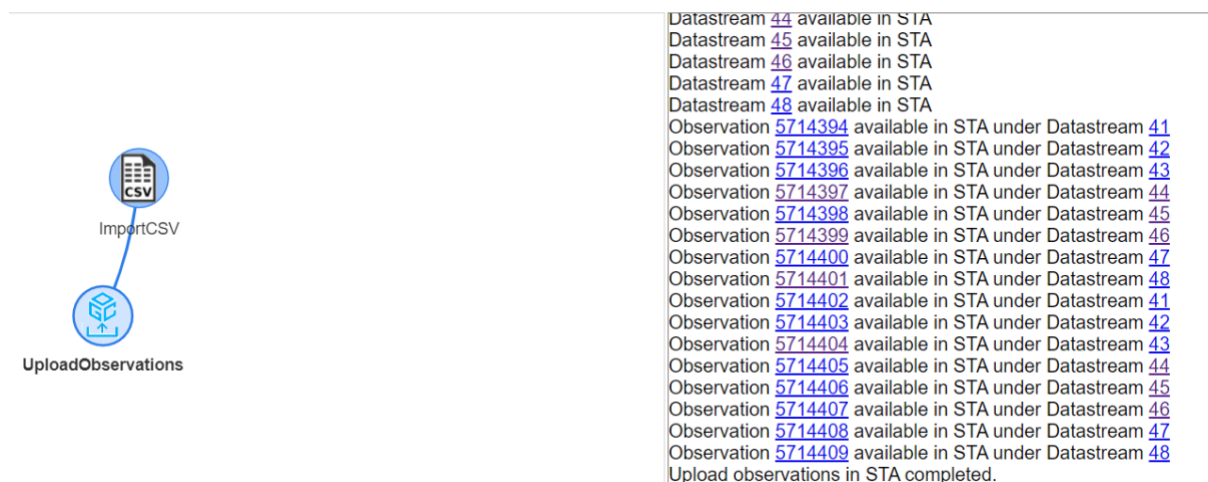


Figure 33. Upload table as STA observations from TAPIS STA+ reader (complete)

4.3 APPLICATION IN PILOTS

The Data Preparation and Integration pipelines have been applied in two of the project pilots, namely the biodiversity and water quality pilots, already introduced in Section 3.4.

As described in Section 3.4, regarding the former, we started the process of harmonisation of datasets from the Group of Earth Observations Biodiversity Observation Network (GEO BON). We used as an example the Global trends in biodiversity (BES-SIM AIM)¹³ that contains projections from the AIM model from 1900-2050 using LUH2 and SSPs-RCPs, done in the BES-SIM inter-model comparison for IPBES. The data, available as NetCDF file, was harmonised and transformed into GDIM compliant format, and stored in the Virtuoso triplestore. The transformation into GDIM was done following the mapping described in Section 3.4. Figure 34 below shows the observation collection¹⁴ representing the dataset visualised in the Virtuoso faceted search endpoint, while Figure 35 depicts a single observation.

¹³ <https://portal.geobon.org/ebv-detail?id=31>

¹⁴ <https://tinyurl.com/2aj8wxam>



About: [Projections from the AIM model from 1900-2050 using for IPBES](#)

[Goto](#) [Sponge](#) [NotDistinct](#) [Permalink](#)

An Entity of Type : <http://www.w3.org/ns/sosa/ObservationCollection>, within Data Space : www.foodie-c

Type: Command:

Attributes	Values
type	Collection of observations
label	Projections from the AIM model from 1900-2050 using LUH2 and SSPs-RCPs, done i
made by sensor	Simulation model BES-SIM AIM
observed property	Community abundance and taxonomic diversity of pollinator insects
member observation	Projection for point -179.5,83.5 time: 1900-01-01 Projection for point -179.5,82.5 time: 2015-01-01 Projection for point -179.5,67.5 time: 2015-01-01 Projection for point -178.5,61.5 time: 2015-01-01 Projection for point -168.5,1.5 time: 2015-01-01 »more»

Figure 34. Global trends in biodiversity (BES-SIM AIM) collection of observations (simulations)

About: [Projection for point -179.5,83.5 time: 1900-01-01](#)

An Entity of Type : <http://www.w3.org/ns/sosa/Observation>, within Data Space : www.foodie-clou

Type: Command:

Attributes	Values
type	Observation
label	Projection for point -179.5,83.5 time: 1900-01-01 (en)
has simple result	b'Full dispersal'
has feature of interest	Feature of interest for observation #1
phenomenon time	1900-01-01 00:00:00 (xsd:dateTime)
result time	2022-02-16 00:00:00 (xsd:dateTime)

Figure 35. Single observation (simulation) of the Global trends in biodiversity (BES-SIM AIM)

The pipeline input was the original dataset (in NetCDF) and the mapping specification. The latter was provided as a set of SPARQL constructs, but it could have also been provided as a YAML file. Figure 36 below depicts the SPARQL constructs for the observation collection and the observations. A key benefit of the DPI is that the same pipeline specification can then be used for all the different input datasets having the same structure (10 in the portal), or for new versions in the future, just by providing a new input file.

```

### observation collection
CONSTRUCT {
  ?obs_col_URI a sosa:ObservationCollection ;
  rdfs:label ?obs_col_label ;
  sosa:observedProperty ?property_URI ;
  sosa:madeBySensor ?sensor_URI ;
  sosa:usedProcedure ?procedure_URI ;
  sosa:hasMember ?obs_URI ;
}
FROM <file:pereira_comcom_id31_20220321_v2.csv>
WHERE {
  BIND (URI('http://w3id.org/ad4gd/geobon/biodiversityprojection/observationCol/BES-SIM_AIM') AS ?obs_col_URI)
  BIND (STRLANG('Projections from the AIM model from 1900-2050 using LUH2 and SSPs-RCPs, done in the BES-SIM inter-model comparison') AS ?property_URI)
  BIND (URI('http://w3id.org/ad4gd/ev/ebv/Communityabundanceandtaxonomicdiversityofpollinatorinsects') AS ?property_URI)
  BIND (URI('http://w3id.org/ad4gd/geobon/biodiversityprojection/model/BES-SIM_AIM') AS ?sensor_URI)
  BIND (URI('http://w3id.org/ad4gd/geobon/biodiversityprojection/procedure/BES-SIM_AIM') AS ?procedure_URI)
  BIND (URI(CONCAT('http://w3id.org/ad4gd/geobon/biodiversityprojection/observation/BES-SIM_AIM/', STR(?ROWNUM))) AS ?obs_URI)
}

### observation
CONSTRUCT {
  ?obs_URI a sosa:Observation ;
  rdfs:label ?obs_label ;
  sosa:phenomenonTime ?proptert_date ;
  sosa:resultTime ?result_date ;
  sosa:hasSimpleResult ?entity ;
  sosa:hasFeatureOfInterest ?feature_URI ;
}
FROM <file:pereira_comcom_id31_20220321_v2.csv>
WHERE {
  BIND (strdt(concat(?time, 'T00:00:00'), xsd:dateTime) AS ?proptert_date)
  BIND (strdt('2022-02-16T00:00:00', xsd:dateTime) AS ?result_date)
  BIND (URI(CONCAT('http://w3id.org/ad4gd/geobon/biodiversityprojection/observation/BES-SIM_AIM/', STR(?ROWNUM))) AS ?obs_URI)
  BIND (STRLANG(CONCAT('Projection for point ', ?lon, ', ', ?lat, ' time: ', ?time), "en") AS ?obs_label)
  BIND (URI(CONCAT('http://w3id.org/ad4gd/geobon/biodiversityprojection/feature', STR(?ROWNUM))) AS ?feature_URI)
}

```

Figure 36. SPARQL constructs defining the transformation of GEO BON dataset into GDIM

Regarding the water quality pilot, we took the dataset of monthly mean levels from 21.12.2013 (including monthly minimum and maximum values), available as a CSV, and harmonised and transformed it into GDIM compliant format using the DPI pipelines. So, for this case, the input to the pipeline is the CSV and the mapping specification was provided in the form of SPARQL construct like in the previous example. Figure 37 shows the observation collection¹⁵ representing the dataset visualised in the Virtuoso faceted search endpoint, while Figure 38 and Figure 39 show an individual example observation and the corresponding observation result.

¹⁵ <https://tinyurl.com/4z3a7cbb>



About: [Monthly mean water for lake: M.-H.-Grenzgr. station: 5865900](http://www.w3id.org/ad4gd/pilot/water/level/observation/5865900/)

An Entity of Type : <http://www.w3.org/ns/sosa/ObservationCollection>, within Data Space : www.foodie-cloud.org, [www](http://www.foodie-cloud.org)

Type: Command:

Attributes	Values
type	Collection of observations
label	Monthly mean water for lake: M.-H.-Grenzgr. station: 5865900 (en)
made by sensor	Station: 5865900 name: Allee der Kosmonauten
has feature of interest	Feature of interest - waters: M.-H.-Grenzgr.
observed property	Water level
member observation	http://w3id.org/ad4gd/pilot/water/level/observation/5865900/2014/April http://w3id.org/ad4gd/pilot/water/level/observation/5865900/2014/August http://w3id.org/ad4gd/pilot/water/level/observation/5865900/2014/Dezember http://w3id.org/ad4gd/pilot/water/level/observation/5865900/2014/Februar http://w3id.org/ad4gd/pilot/water/level/observation/5865900/2014/Januar »more»

Figure 37. Observation collection of monthly mean levels from 21.12.2013 from a particular lake and station

About: <http://w3id.org/ad4gd/pilot/water/level/observation/5865900/>

An Entity of Type : <http://www.w3.org/ns/sosa/Observation>, within Data Space : www.foodie-cloud.org, [www.foodie-cloud.o](http://www.foodie-cloud.org)

Type: Command:

Attributes	Values
type	Observation
has result	Result of monthly mean water for lake: M.-H.-Grenzgr. station: 5865900 year: 2014 month: April
phenomenon time	2014-04-01 00:00:00 (xsd:dateTime)
result time	2014-04-01 00:00:00 (xsd:dateTime)

Figure 38. Single observation of the monthly mean water for particular lake/station for April 2014

**About: Result of monthly mean water for lake: M.-H.-Grenzgr. station: 5865900 year:**An Entity of Type : <http://purl.org/iot/vocab/m3-lite#WaterLevel>, within Data Space : www.foodie-cloud.org, www.foodie-cloud.org associatedType: Command:

Attributes	Values
type	Result http://purl.org/iot/vocab/m3-lite#WaterLevel
label	Result of monthly mean water for lake: M.-H.-Grenzgr. station: 5865900 year: 2014 month: April (en)
numeric value	6
unit	cm
http://w3id.org/ad...aterpmaxValueDate	14.04.2014 07:00
http://w3id.org/ad...aterpminValueDate	05.03.2018 03:30
https://schema.org/maxValue	32
https://schema.org/minValue	0

Figure 39. Result of observation of the monthly mean water for particular lake/station for April 2014

5 GREEN DEAL DATA SPACE APIS

Application Programming Interfaces (APIs), as the software components access services, are a critical element of the data space ecosystem that enables exchange of information. In regulations^{16, 17} shaping the European Data Spaces: ‘application programming interface (API)’ means a set of functions, procedures, definitions and protocols for machine-to-machine communication and the seamless exchange of data;

APIs can be characterised by the set of:

- data models
- operations like read, write, update, delete, subscribe
- query operators like
 - filters based on simple properties like Ids, strings and numbers with
 - advanced query languages like SQL, GraphQL, CQL, SPARQL
- transport protocol related to its mode like HTTP for request-response, MQTT or JMS for messaging, while combinations are possible like streaming of message pull via HTTP

For High Value Datasets (HVD), these regulations elaborate on the requirements for APIs:

- Machine-readability (art3p1)
- bulk -download option for selected (art3p1)
- Documented art3p3
- Accompanied with terms of service and quality of service (including performance, capacity, availability, also in human and machine readable form
- Metadata compliant with INSPIRE
- Relations

¹⁶ Commission Implementing Regulation (EU) 2023/138 of 21 December 2022 laying down a list of specific high-value datasets and the arrangements for their publication and re-use (Text with EEA relevance) https://eur-lex.europa.eu/eli/reg_impl/2023/138/oj

¹⁷ DIRECTIVE (EU) 2019/1024 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 20 June 2019 on open data and the re-use of public sector information <https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32019L1024&qid=1696244943811>

Other requirements that may influence the API:

- Minimum resolution for given information
- Minimum metadata properties (HDV Annex 1.1) which means set of properties defined in the directive plus: Boundary status, National identification code, Identification code of the upper administrative level; multi-language support for multi-lingual countries
- Plus, some potentially renamed: Official name

Standards define the set of required and optional elements from the above with various levels. Based on these, service providers implement suitable elements (e.g functional elements and formats) and deploy the service bound to the particular endpoints. Altogether is the definition of the final API.

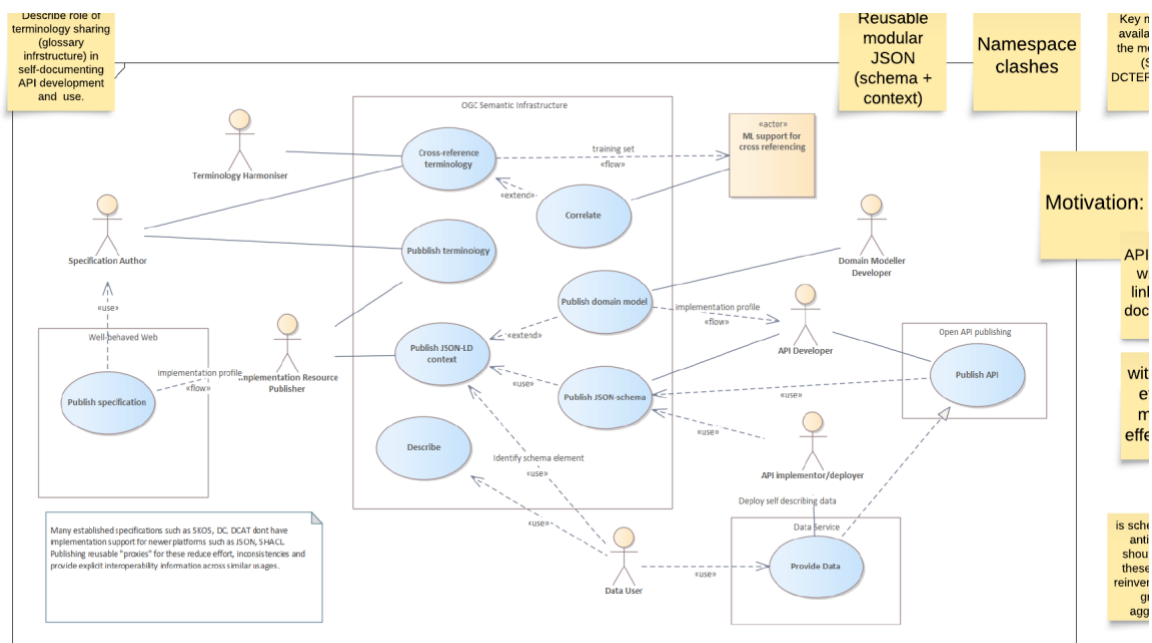


Figure 40. API definition roles and assets

API consumers' applications often combine multiple services integrations and federation can rise the complexity of the integration equipotentially. For example, according to IDSA System Layer architecture¹⁸ client applications communicate with Data Broker, Data Provider, Clearing House, Application Store with the support of Vocabulary service. Logical diagram¹⁹ suggests simplified connections through the IDS connector. In case data is encapsulated, data space connectors can limit the complexity with standardised API. In practice, catalogue navigation (Data Broker in IDSA) uses both generic and domain specific metadata and operators (like area of interest and information definition). Then, Brokers can delegate the queries online in full federation or act as the central hub for the metadata. In both cases translations between specific data interfaces of domain specific models can be required. Similarly, depending on the Data Provider profile various data and access models can be supported. For example, some APIs are optimised for batch access (like GeoPackage) and some for interactive chunking (like Tiles services) or streaming. Then the role of the Connector can still be a mediator, interface to the IdP, proxy

¹⁸ <https://docs.internationaldataspaces.org/ids-knowledgebase/v/ids-ram-4/layers-of-the-reference-architecture-model/3-layers-of-the-reference-architecture-model/3.5.0-system-layer>

¹⁹ <https://docs.internationaldataspaces.org/ids-knowledgebase/v/ids-ram-4/layers-of-the-reference-architecture-model/3-layers-of-the-reference-architecture-model/3.5.0-system-layer>



managing and clearing (logging) the access (Api gateway), and being transparent to the transfer protocol. Then, the Connector shall also use and support well defined (in Vocabularies) of the lower level APIs.

To satisfy FAIR principles, APIs shall:

- provide discoverability - to determine which endpoint is providing data needed on the relevant conditions and in useful manner
- implement access - to get to the right data in the way that is efficient for use case (like streaming, bulk, trimmed)
- use standards for interoperability with other components
- provide metadata describing sufficiently information including: data content like models and definitions, relations with other data entities, lineage, legal use conditions, ownership

Considering natural data spaces platform is the Web, among good practices of the data sharing, it is worth to mention W3C work on the 'Data on the Web Best practices'²⁰ and given most of the Green Deal data is localised, also 'Spatial Data on the Web Best Practices'²¹.

As the focus of this document is spatial data, chapter explains good practices implementation for the data space building blocks on the proposed suite of OGC standards as well as outline potential extensions and tools enabling support for semantics and data integration.

5.1 OVERVIEW OF SPATIAL STANDARDS WITH APIS

OGC standards define models, formats and exchange mechanisms for localised data defined by the community of practitioners with the long-standing heritage. OGC APIs is the sub-suite of standards modernising widely adopted OGC Web Services (OWS) like WMS, WCS, WFS, WPS, CSW. They remain compliant with higher level ISO standards and some were adopted as ISO ones. OGC API and Sensor Things API are proposed as the JRC good practices^{22, 23}.

Structure and self-descriptiveness

In the space of services and data, the discovery phase includes finding the right resources and recognising how it can be used. The step requires both understanding of data content and all the interoperability aspects from the EIF (see introduction). Following good practices, it shall be part of the metadata service description provided as part of the interface standard. Then metadata includes information about: data access protocol, formats, representations (like spatial projection), ontologies referred, licences, data model and content with its boundaries like spatial area and time period. In the OGC API suite, the process starts with the HTTP endpoint that declares relevant metadata partly inline and partly via links.

OGC (and ISO) standards are a set of conformance(compliance) requirements grouped in classes identifiable by URIs. Implementations can select and declare which classes they satisfy while the minimum core requirements are the obligatory ones. Standards can be also multipart extending optional core with complementary optional functionalities. Then, standards can be extended with custom functionalities and data as the standard profile.

²⁰ <https://www.w3.org/TR/dwbp/>

²¹ <https://www.w3.org/TR/sdw-bp/>

²² <https://inspire.ec.europa.eu/portfolio/good-practice-library>

²³ <https://inspire.ec.europa.eu/good-practice/ogc-sensorthings-api-inspire-download-service>

Example

Service provider wants to deploy API for car charging stations localisation and share it with partners that would provide current occupancy. She uses OGC API Features API Part 1 - Core that defines GeoJSON Features collection, Part 4 for CRUD operations and Part 2 for local CRS support. In addition, she needs to define schema for car charging stations including landing station slots, occupancy, queue but also technical details like socket type that needs unambiguous ID and circuit parameters. These are defined in the payload schema with references to proper vocabularies.

OGC API is a family sharing some principles including OpenApi compliance, structure of endpoints, default human and machine-readable encodings. All of them are HTTP request-response services with JSON and HTML encodings on default with various extensions possible. They may be combined in one endpoint with separate relative URLs but also referred from one Landing Page in the fully distributed manner preserving consistency of implementations.

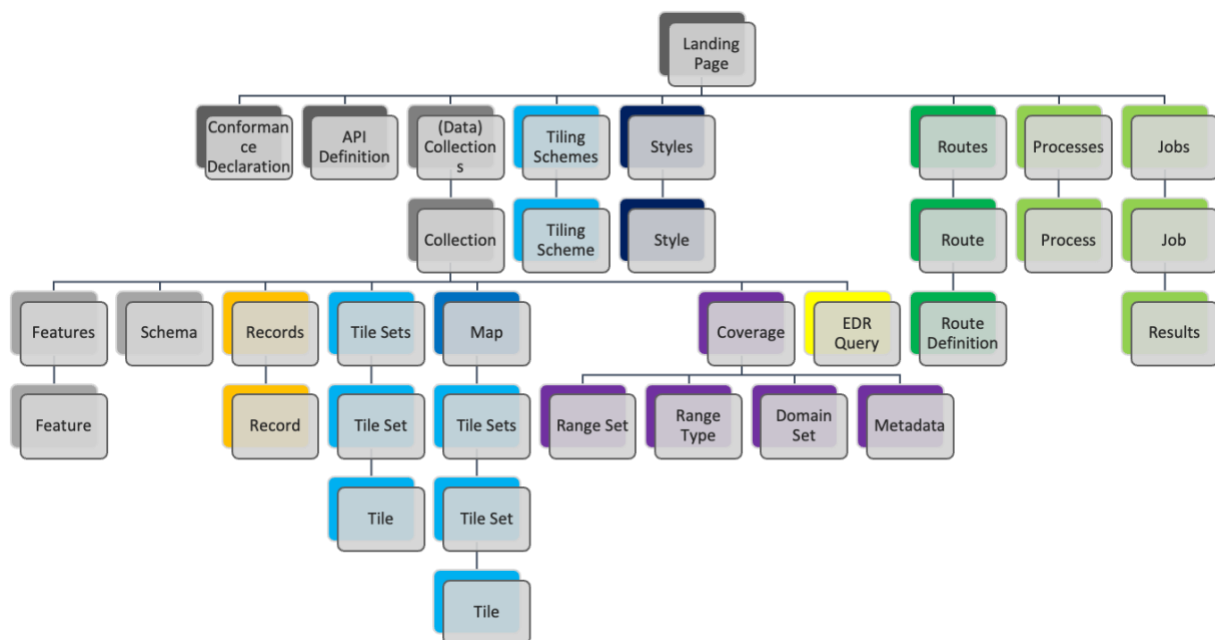


Figure 41. OGC API Resources navigation tree

Leveraging the OpenApi capabilities of the self-descriptiveness, OGC APIs standards are accompanied with schema (in JSON schema) and api definition (in YAML).

An example is common part now included in most of the APIs:

<https://app.swaggerhub.com/apis/OGC/ogcapi-common-1-example-1/1.0.0#/server/getLandingPage>

Core elements embrace:

- api definition compliant with OpenApi exposed by the endpoint with all functions, parameters, responses
- human and machine readable encodings support
- landing page with API endpoint title, description, all the next level links in the navigation tree (see Figure 41)
- Conformance declarations of implemented: standard, version, conformance classes



Catalogues and brokers

Geospatial data cataloguing is continuing long standing efforts of the community around TC211 metadata models (ISO19115), items governance (ISO19135), their encodings (e.g. ISO19136) and services implementation specifications like OGC CSW and emerging OGC API Records²⁴.

They implement functionalities analogues to the IDSA brokering functionality responsible for publication and query of self-Descriptions, while the information scope is significantly different between these. OGC CSW and Records API metadata and respective search capabilities are based on the requirements of the spatial data management, in IDSA it is more service ownership and contract focused.

IDSA foresee the need to integrate additional attributes and their query enablement²⁵. Thus, the potential approach is to integrate the GIS catalogue and IDSA broker. On the encoding level OGC API Records and IDSA Broker catalogue supports JSON and can be extended to JSON-LD. If the catalogue contains data objects it might be extended with the data metadata description and additional querying operations from OGC API Records and similar. If the catalogue resource can be OGC Records service, catalogue metadata could be used in a similar way.

OGC API Records is already a good example of the service's other standard extension. OGC API Features/Common functionalities are included directly with the spatial data represented with its vector footprint, query and filter options. "The core catalogue model is based on an extension of Dublin Core (CSW Record). Application profiles can be developed to target specific metadata information models (such as ISO 19115/19139, etc.)"²⁶ which alone is not sufficient to satisfy HVD or INSPIRE requirements²⁷. It is, however, together with OGC API Records core queryables and extensions²⁸ (keyword, URI, licence, rights,), API declarations (conformance) and data space Broker contractual descriptors. Proposed approach for the composition definition is based on the smallest relevant elements definitions and the definition of the composition. The assumption is that a catalogue of spatial and non-spatial data shall also have a canonical reference so that alignment can be defined to this central one. This way, the [Building Blocks for OGC API Records](#) catalogue has been initiated to define these elements with alignments.

Example

STAC v1.0²⁹ is the example of the API for specific data type (Earth Observation raster assets) that was growing next to the OGC API Records and became fully aligned to the OGC API suite recently. The community is maintaining now the repository of extensions³⁰ with both new functionalities like [Aggregation](#) and extensions to existing functionalities like [search with EO specific parameters](#). Their further identification within data space shall allow clients to recognise compatible and partly compatible interfaces but the interface shall declare OGC API Features compliance and any additional extension together to support minimum compliance model (see [/rec/record-core/extensions of OGC API Records](#)).

²⁴ <https://docs.ogc.org/DRAFTS/20-004.html>

²⁵ [https://docs.internationaldataspaces.org/ids-knowledgebase/v/ids-ram-4/layers-of-the-reference-architecture-model/3_5_0_system_layer/3_5_4_metadata_broker](https://docs.internationaldataspaces.org/ids-knowledgebase/v/ids-ram-4/layers-of-the-reference-architecture-model/3-layers-of-the-reference-architecture-model/3_5_0_system_layer/3_5_4_metadata_broker)

²⁶ <https://docs.ogc.org/DRAFTS/20-004.html#api-behaviour-model-overview>

²⁷ https://inspire.ec.europa.eu/reports/ImplementingRules/metadata/MD_IR_and_ISO_20090218.pdf

²⁸ <https://docs.ogc.org/DRAFTS/20-004.html#record-model>

²⁹ <https://github.com/radiantearth/stac-api-spec/tree/v1.0.0/ogcapi-features>

³⁰ <https://github.com/stac-api-extensions/>

Data access services

Vector data APIs - Features API and Sensor Things

Two first JSON based OGC APIs were *Sensor Things API* and *OGC OGC API Features*. Both are designed for access to spatial vector data with given localisation. First standard is designed to handle observation data following models of ISO Observations & Measurements³¹ and W3C SOSA/SSN³². It is implementing OData interfaces combining flat collections of key entities like Observations, DataStream, ObservedProperties and Locations with more advanced query operators tailoring the service answers. Second one is the successor of the OGC Web Map Feature service that is specified according to the OGC APIs principles. It is more generic than STA, defining only a few entities directly related to collection of vector features: Collection, Item (FeatureType ISO 19109), Geometry with optional properties' set. Both were positively evaluated against suitability for INSPIRE services implementation³³ with the relation exemplified. Further, OGC API payload profiles can be defined as standard extensions. Example one for features representation of observation is defined in the OGC Building Blocks incubator: <https://opengeospatial.github.io/ogcapi-sosa/build/generateddocs/slate-build/unstable/sosa/features/observation/>. OGC Features API is also a multipart standard where the implementer can declare conformance to classes from one or several parts. Given all that, the service description contains of:

- /conformance declaration with conformance classes (from multiple standards/parts) and custom ones like encoding extensions
- /api declaration with endpoints and schemas including extensions:
 - /Collections - is default schema from OGC API Features or OGC API Common
 - /Items - as observations features collection profile like [ogc.unstable.sosa.features.observation - SOSA Observation Feature](https://opengeospatial.github.io/ogcapi-sosa/build/generateddocs/slate-build/unstable/sosa/features/observation/) extending [Feature Collection](#)
 - /Items/{itemid} - as observation feature <https://opengeospatial.github.io/ogcapi-sosa/build/generateddocs/slate-build/unstable/sosa/properties/observation/> extending [GeoJSON Feature](#)

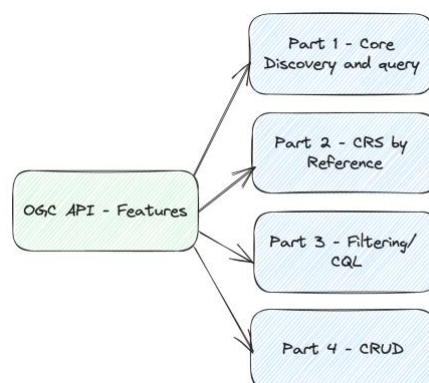


Figure 42. Multipart standard of OGC API with advanced functionalities

Coverage and raster data

Coverage are specific data representing phenomena for particular spatial area and time. In practice they are sometimes defined as a function of space and time (domain) to the physical world representation (range set of

³¹ <https://www.iso.org/standard/32574.html>

³² <https://www.w3.org/TR/vocab-ssn/>

³³ <https://www.ogc.org/blog-article/inspire-and-ogc-apis-modernizing-inspire/>



properties) (ISO19123). Data can represent sensor data like satellite imagery, aggregates, compositions and model outputs. Various formats are popular in different specialisations; e.g., GeoTiff in Earth Observation and NetCDF in meteorology and oceanography. Commonly, data size is significant (up to GBs in EO scenes), for which reason access APIs needs to encounter several techniques for performance optimisation that related to the metadata description

- Separation of metadata form data payload or aggregation - some formats mix metadata and data (like CovergaJSON, NetCDF) and other separate (like GeoTiff in WCS), but metadata can be retrieved fully and read before data payload request is issued
- Aggregation of metadata on grid level (not on pixel/feature level), description of the domain set and range set definitions (like DescribeCoverage function in WCS and optionally in the OGC EDR collection definition))
- Multi-resolution support - can be supported on the server side like in the WMS, WMTS or aggregated on the client side line in OGC API EDR
- trimming in data access to query data for selected area and time span, optionally also with resolutions
- Various encodings for various needs (like JSON and Jpeg for web portals and map applications, NetCDF for research applications)
- Compression of transferred payload

Metadata and related semantics need to support these various queries to accompany data payload per request or be defined sufficiently for the whole grid, or both. Most of the standards propose how the definitions can be referred from the metadata content. In some, like WCS1.0 variables representing phenomena are defined as RangeSet without explicitly references of properties required.

```
DescribeCoverage WCS 1.0
...
<rangeSet>
  <RangeSet>
    <name>RangeSet_1</name>
    <label>nhm_dom_topo_25833 RangeSet</label>
    <axisDescription>
      <AxisDescription>
        <name>Band</name>
        <label>Band Numbers</label>
        <values>
          <singleValue>1</singleValue>
        </values>
      </AxisDescription>
    </axisDescription>
  </RangeSet>
</rangeSet>
```

Listing 1. CoverageDescription in OGC WCS 1.0

WCS2.1 proposes Coverage description proposes rangeType definition with SWE DataRecords compliance and definition reference to property definition.



```
DescribeCoverage WCS 2.1
<cis11:rangeType>
  <swe:DataRecord>
    <swe:field name="singleBand">
      <swe:Quantity
        definition="http://www.opengis.net/def/property/OGC/0/Radiance">
        <swe:description>Panchromatic Channel</swe:description>
        <swe:uom code="W/cm2"/>
      </swe:Quantity>
    </swe:field>
  </swe:DataRecord>
</cis11:rangeType>
```

Listing 2. CoverageDescription in OGC WCS 2.1

In addition, specific profiles like 'WCS2.1: Part 0 MetOcean Metadata' require more specifically identification of the range set with unambiguous, Climate and Forecast convention references. EDR coverages also proposes explicitly reference though observedProperty values.

```
type: Parameter
  id: sea_ice
  description: Sea Ice concentration (ice=1;no ice=0)
  unit:
    label: Ratio
    symbol:
      value: "1"
      type: http://www.opengis.net/def/uom/UCUM/
  observedProperty:
    id: http://vocab.nerc.ac.uk/standard_name/sea_ice_area_fraction/
    label: Sea Ice Concentration
```

Listing 3. Rangeset in OGC API EDR example

5.2 APIS AND SEMANTICS

API semantics can potentially include all the properties of the service description. Given OpenApi de-facto standard enable definition of the API functions, parameters, schemas, following considerations tackle levels of the semantic interoperability available in these.

Semantics of technical service description

As described, OGC API endpoint provides several information about the service and links to further information and data. An example defined for the OGC API Commons is like following:

<https://app.swaggerhub.com/apis/OGC/ogcapi-common-1-example-1/1.0.0#/server/getLandingPage>



```
{
  "title": "Buildings in Bonn",
  "description": "string",
  "links": [
    {
      "href": "http://data.example.com/buildings/123",
      "rel": "alternate",
      "type": "application/json",
      "hreflang": "en",
      "title": "Trierer Strasse 70, 53115 Bonn",
      "length": 0
    }
  ]
}
```

Listing 4. OGC API Common landing page JSON representation

In addition to the specification of properties in the standard³⁴. Definition is partly self-explanatory and partly human readable. Properties like 'rel', 'href', 'title' are commonly used, while in general case the non-OGC API can use a different set of properties. Reader can guess 'type' is the MIME and lang is the country code and not language/country coded once she sees the data only. meaning of length could be document size under the link or the length of collection or something else. Properties defined can be found in the specification or alternatively specified in the extended metadata. Following example outlines how JSON-LD extension can be used to define the namespaces and properties mapping to abstract ontologies and vocabularies.

```
{@context{
  "title": "rdfs:label",
  "description": "dct:description",
  "links": {
    "@context": {
      "href": "oa:hasTarget",
      "rel": {
        "@id": "http://www.iana.org/assignments/relation",
        "@type": "@id",
        "@context": {
          "@base": "http://www.iana.org/assignments/relation/"
        }
      },
      "type": "dct:type",
      "hreflang": "dct:language",
      "title": "rdfs:label",
      "length": "dct:extent"
    }
  }

  "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
  "oa": "http://www.w3.org/ns/oa#",
  "dct": "http://purl.org/dc/terms/",
  "@version": 1.1
}

{
  "title": "Buildings in Bonn",
  "description": "string",
```

³⁴ https://docs.ogc.org/is/19-072/19-072.html#_c3f80621-bf44-44d4-8c6a-41def97110ff



```
"links": [  
  {  
    "href": "http://data.example.com/buildings/123",  
    "rel": "alternate",  
    "type": "application/json",  
    "hreflang": "en",  
    "title": "Trierer Strasse 70, 53115 Bonn",  
    "length": 0  
  }  
]
```

Listing 5. Example of the context definition for the API landing page with title, description and all the next level endpoints

Now we see explicitly that e.g., 'rel' type is defined in IANA vocabulary and the length is the size of the object. However, as the standard OGC API Common defines the subset of IANA as base relations, it could be even more specific.

The context can be defined as part of the JSON-LD data payload, as a reference from the payload. One more option is the reference from the service definition in the link 'rel' property of the referring link <http://www.opengis.net/def/rel/ogc/1.0/data-meta>

Semantics of data models

The widespread use of JSON as the data exchange format for most modern APIs provides several benefits: it is easy to read and write (both by computers and humans), has a lightweight syntax (unlike other, more verbose formats such as XML), and can represent almost any type of non-binary data natively. The flexibility that JSON offers to API implementers, however, comes with a lack of in-band information for consumers regarding structure and semantics; indeed, client implementers are usually required to manually read service-specific documentation in order to effectively communicate using APIs. This issue is present even when using standard API specifications: an OGC Features API-compatible service could offer data about collections of observations gathered by a sensor, about points of interest in a specific location, or about the number of available spots in each parking lot in a city; while all of them are geospatially enabled features that may share a set of common characteristics, the data models necessary to describe each collection will most certainly differ, as will the semantics of their specific properties and values.

Therefore, achieving effective data interoperability requires both defining those structural and semantic models and signalling their location to clients. JSON Schema can be used to describe constraints for the shape of a piece of JSON data, and, as described in section 4.2.3, there are several techniques that can be applied on top of it to attach semantic information / context to the different elements contained in the schema. While implementing these types of mechanisms in APIs and services is a step in the right direction, the need to standardise common patterns depending on characteristics such as the nature (observation, point of interest) or domain (agriculture, cartography) of the described data arises as a new interoperability issue.

The methodologies and practices developed in the scope of the OGC Location Building Blocks initiative will be employed in AD4GD for this purpose. The Location Building Blocks are a collection of commonly recurring patterns in software development, such as data types, schemas, parts of (and even full) APIs / specifications, in the form of, among others, documentation, machine-readable metadata, JSON schemas, JSON-LD contexts, and SHACL shapes for validation. New building blocks can be created, either from scratch or, preferably, by combining (composition) or specialising (inheritance) others. The resulting definitions are then run through a Continuous Delivery / Continuous Testing pipeline where they are validated (including the correctness and the semantic coherence of documentation examples), augmented and published.



Table 4 shows a list of the property names employed when annotating building blocks JSON schemas and OpenAPI specifications.

Table 4. JSON schema semantic annotations and example of a fully annotated JSON schema definition

Schema semantic annotation properties	Example
<p>All properties are prefixed with "x-jsonld-":</p> <ul style="list-style-type: none"> • context: Reference to location of JSON-LD context definition for the schema. • id: equivalent to JSON-LD @id term. • prefixes: prefix mapping for JSON-LD context generation. • extra-terms: other declared terms that are not bound to any properties. • base: equivalent to JSON-LD @base term inside a scoped context. 	<pre>{ "description": "SOSA Observation", "type": "object", "properties": { "resultTime": { "type": "string", "format": "date-time", "x-jsonld-id": "http://www.w3.org/ns/sosa/resultTime" }, ... "x-jsonld-extra-terms": { "Observation": "http://www.w3.org/ns/sosa/Observation", "Sample": "http://www.w3.org/ns/sosa/Sample", "observes": { "x-jsonld-id": "http://www.w3.org/ns/sosa/observes", "x-jsonld-type": "@id" } } ... "features": "http://www.w3.org/ns/sosa/hasMember", "properties": "@nest", "featureType": "@type" }, ... "x-jsonld-prefixes": { "sosa": "http://www.w3.org/ns/sosa/", "ssn": "http://www.w3.org/ns/ssn/", "ssn-system": "http://www.w3.org/ns/ssn/systems/" } }</pre>

Service implementers can point to these meta-resources when they serve their data and in their API descriptions (e.g., OpenAPI), for example using mechanisms such as adding a 'Link' HTTP header with a reference to the JSON-LD context of the employed building block or including OGC Features and Geometries JSON (JSON-FG) 'link' elements inside their data items, offering an additional interoperability tier to machine consumers.

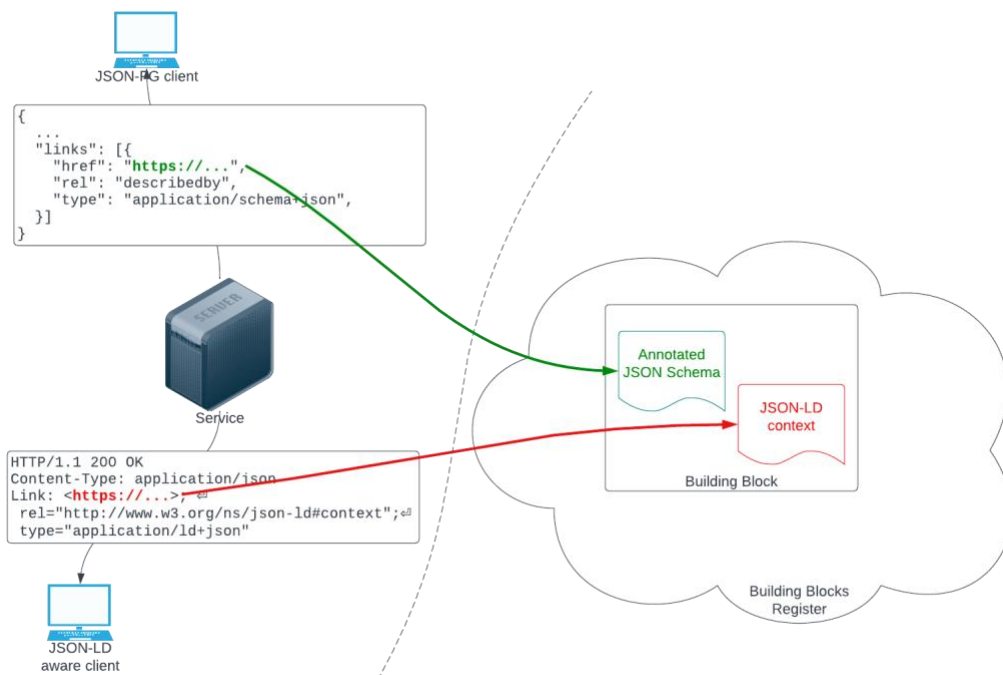


Figure 43. Illustration of different mechanisms to attach a JSON schema and/or JSON-LD context to a service response

Apart from its own metadata, the central component of a building block is its JSON schema, that can be linked to a JSON-LD context with a set of annotations. References to other schemas belonging to already existing building blocks can be leveraged for several purposes, such as reducing code duplication (along with the problems that it involves), detecting dependencies between building blocks, and guaranteeing compliance with base standards, thus increasing semantic interoperability. Additionally, full, properly scoped JSON-LD contexts are built for specialised building blocks by traversing the whole hierarchy, reducing the burden on implementers, who can focus on the particular characteristics of their data or use case.

Thus, building block specialisation hierarchies can be created, with simpler schemas at the bottom and guaranteed compliance with a series of specifications all the way to the top (Figure 44).

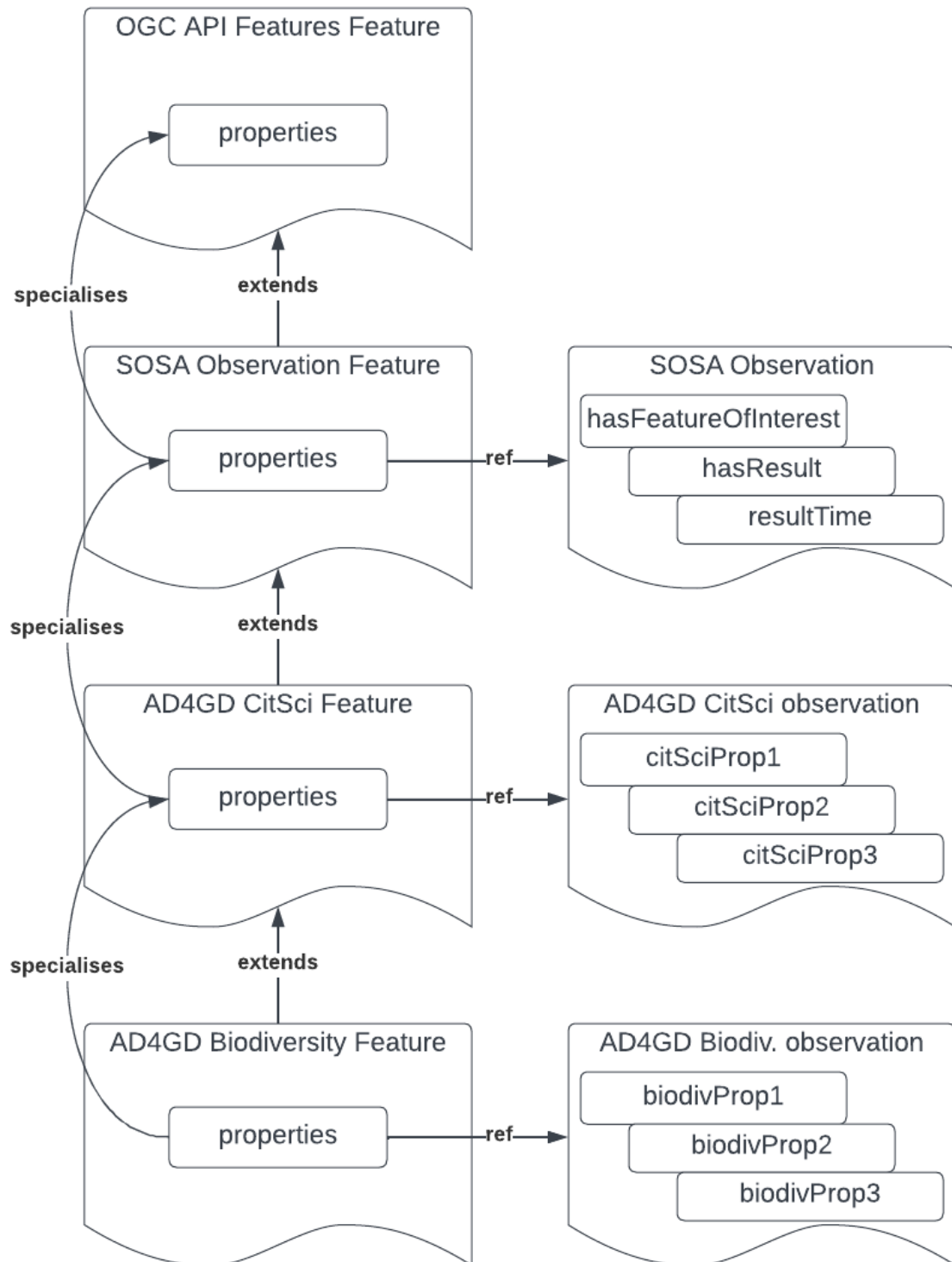


Figure 44. Example of a schema + semantics specialisation hierarchy using OGC Building Blocks

Therefore, a specific, individual Building Blocks Register will be created for AD4GD, where all the building blocks created during the development of the project will be published and validated. AD4GD building blocks will specialise already existing ones, such as the SOSA Building Block collection for observations and measurements,



adding the necessary constraints and mappings to make data generated within the project compatible with the GDIM.

5.3 RELATION TO OTHER DATA SPACES' BUILDING BLOCKS

Data Models

Data models can be complicated and hard to standardise part of the API definition. In particular, due to its dynamics and numerous applications they must cover a variety of domain specific properties that are constantly changing. Models can be defined in the standard on the very abstract level (like OpenStreetMap, NGSI-LD), data specific for core (like in SensorThings API) or a mix (OGC API Features has geospatial constant core and open list of properties). Decisions can be made based on the domain specificity, implementation efficiency and human readability. It is expected that the more open the key-value catalogue, the more challenging is the effort of Vocabularising the entities definition.

Identity and Access Management

Service description may need to be extended with the capabilities declarations like proposed in OGC API Features part 4: https://docs.ogc.org/DRAFTS/20-002.html#_security_considerations

It is one of the aspects the Vocabularies between standards should be cross-referenced.

```
{
  "openapi" : "3.0.3",
  "info" : {
    "title" : "My API",
    "description" : "This API ...",
    "version" : "1.0.0"
  },
  "servers" : [ {
    "url" : "https://example.com/api/v1"
  } ],
  "security" : [ {
    "JWT" : [ ],
    "api_key": [ ]
  } ],
}
```

Listing 6. Example of the proposed service description part declaring authentication technology support. JWT is a de-facto standard

Grant type reference access needs to be agreed between the Authentication party and the application, but the access management may be too specific to establish any reference model.

Security and Service Levels

Being web (mostly HTTP) interfaces the APIs fall under the natural security constraints elaborated in standards like https://docs.ogc.org/is/17-069r4/17-069r4.html#_security_considerations or https://docs.ogc.org/DRAFTS/20-002.html#_security_considerations. OGC proposes there some recommendations on how to handle risks related to invalid or not allowed requests that can influence responses code lists and require detailed validation/sanitisation of the requests. For services with access granulation based on the spatial areas GeoXACML proposes the language of the constraints definition.

At the same time, OGC APIs do not specify how the particular deployment shall implement security likewise continuity of the service declarations like performance, . It is recommended that standard Web API measures



are applied like OWASP recommendations, ISO 27001 and similar according to the sensitivity of data and system criticality.

Service Levels

According to the regulations concerning HDV, service description embracing performance, capacity and availability shall be machine readable (HVD art3p2). As OGC APIs are generic, similar measures shall be linked to the service description metadata according to the selected data space definition (IDSA, Gaia-X or other). It is therefore recommended that some of the relevant definitions (like <https://lov.linkeddata.es/dataset/lov/vocabs/ends>) are endorsed in the Data Space Vocabulary accordingly.

Metadata and Lineage

On all levels of description, metadata schemas and vocabularies can vary among the platforms and standards. The integration within data space may include combining metadata inline or through linkages. First one may be required for consistency and query functionality, while the other is more lightweight but can contain inconsistencies. To enable interoperability within the data space, standards and their profiles shall be listed in the Data Space Vocabularies to enable unambiguous client-service matching. Further model matching requires ontologies and codelists definitions so the models could be aligned or the crosswalks and translations formalised. Lineage Building Block from the OGC API Records³⁵ could be the common reference for provenance based on the DCAT provenance, but details need to be assessed with detailed cases.

5.4 APPLICATION IN PILOTS

The harmonised pilots' data, described in Section 4.3, has been made accessible via a standard SensorThings API, which is automatically generated by the DPI STA service, described in Section 4.2.1.3. The automatic generation of the STA API showcases the ultimate value of the DPI pipelines, which covers all the tasks from the data collection to data harmonisation up to exposing them via standard APIs, with minimum user intervention, and which can be easily re-executed with other datasets or new versions of the original datasets.

The STA for the monthly mean water for the lake: M.-H.-Grenzgr. station: 5865900 is accessible via: <https://sta-demeter.apps.paas-dev.psnec.pl/AD4GD-water-5865900/api/v1.0/> while the Swagger interface is accessible via <https://sta-demeter.apps.paas-dev.psnec.pl/AD4GD-water-5865900/api/v1.0/docs> (depicted in Figure 45). Figure 46 depicts the retrieval of the STA observations.

³⁵ <https://github.com/ogcincubator/geodcat-ogcapi-records>



OGC SensorThings API 1.0.0 OAS 3.1

</AD4GD-water-5865900/api/v1.0/openapi.json>

Servers

▾

Base

Observations

Datastreams

FeaturesOfInterest

HistoricalLocations

Locations

ObservedProperties

Sensors

Things

Figure 45. STA for the monthly mean water observations for lake: M.-H.-Grenzgr. station: 5865900



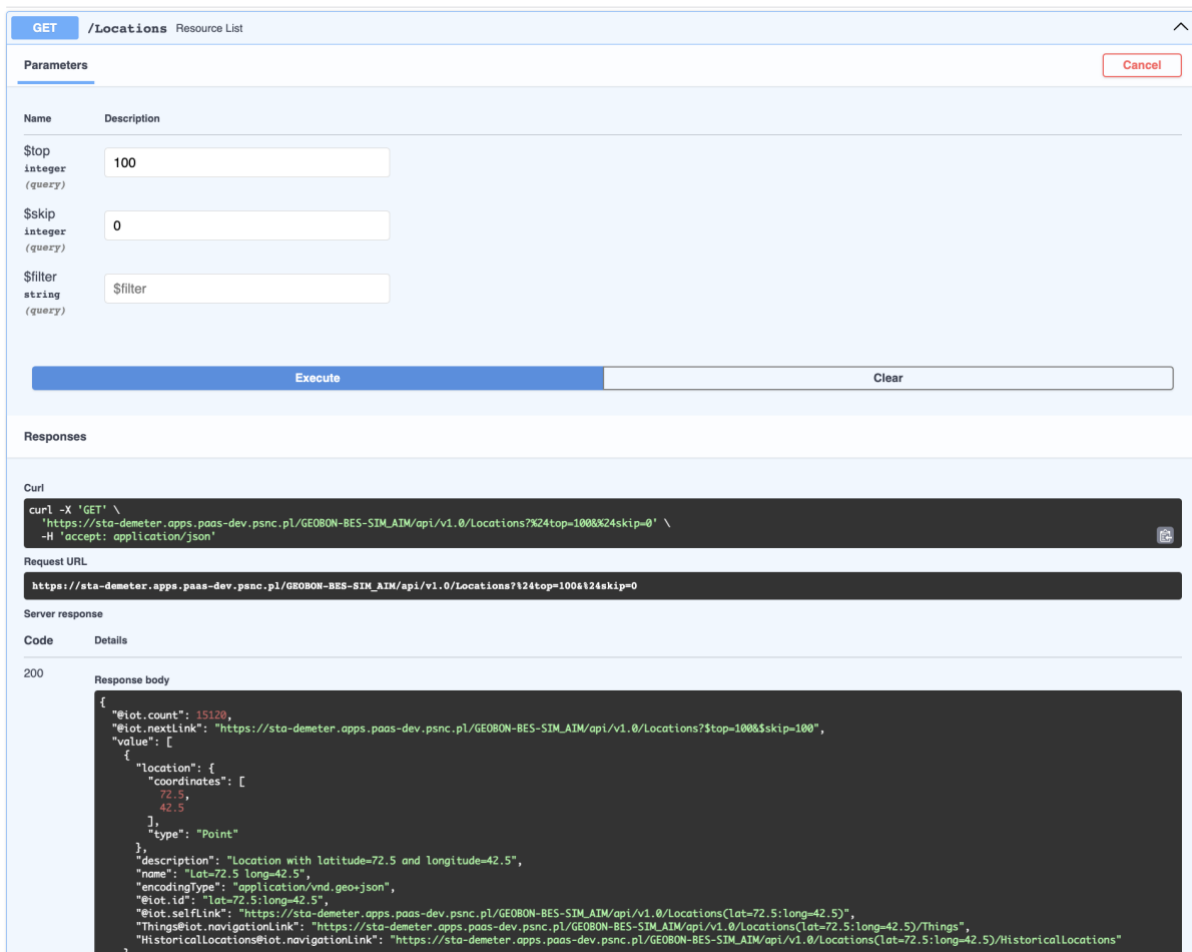
The screenshot shows a REST client interface with the following details:

- Method:** GET
- Path:** /Observations
- Parameters:**
 - \$top:** integer (query), Default value: 100, value: 100
 - \$skip:** integer (query), Default value: 0, value: 0
- Request URL:** `https://sta-demeter.apps.paas-dev.psnec.pl/AD4GD-water-5865900/api/v1.0/Observations?$top=100&$skip=0`
- Server response:** 200
- Response body (JSON):**

```
{
  "@iot.count": 117,
  "@iot.nextLink": "https://sta-demeter.apps.paas-dev.psnec.pl/AD4GD-water-5865900/api/v1.0/Observations?$top=100&$skip=100",
  "value": [
    {
      "result": {
        "type": "http://purl.org/iot/vocab/m3-lite#WaterLevel",
        "label": "Result of monthly mean water for lake: M.-H.-Grenzgr. station: 5865900 year: 2014 month: April",
        "numericValue": "6",
        "unit": "cm",
        "waterpmaxValueDate": "14.04.2014 07:00",
        "waterpminValueDate": "05.03.2018 03:30",
        "maxValue": "32",
        "minValue": "0"
      },
      "phenomenonTime": "2014-04-01T00:00:00",
      "resultTime": "2014-04-01T00:00:00",
      "@iot.id": "5865900-2014-April",
      "@iot.selfLink": "https://sta-demeter.apps.paas-dev.psnec.pl/AD4GD-water-5865900/api/v1.0/Observations(5865900-2014-April)",
      "Datastream@iot.navigationLink": "https://sta-demeter.apps.paas-dev.psnec.pl/AD4GD-water-5865900/api/v1.0/Observations(5865900-2014-April)/Datastream",
      "FeatureOfInterest@iot.navigationLink": "https://sta-demeter.apps.paas-dev.psnec.pl/AD4GD-water-5865900/api/v1.0/Observations(5865900-2014-April)/FeatureOfInterest"
    }
  ]
}
```

Figure 46. STA observations for the monthly mean water observations for lake: M.-H.-Grenzgr. station: 5865900

The STA for the global trends in biodiversity (BES-SIM AIM) is accessible via: https://sta-demeter.apps.paas-dev.psnec.pl/GEOBON-BES-SIM_AIM/api/v1.0/ while the Swagger interface is accessible via https://sta-demeter.apps.paas-dev.psnec.pl/GEOBON-BES-SIM_AIM/api/v1.0/docs. Figure 47 depicts the retrieval of the STA locations.



The screenshot shows a REST client interface for a GET request to the endpoint `/Locations`. The parameters section includes `$top` (integer, value 100), `$skip` (integer, value 0), and `$filter` (string, value \$filter). The response is a 200 status code with a JSON body containing location data.

```

curl -X 'GET' \
  "https://sta-demeter.apps.paas-dev.pssc.pl/GE080N-BES-SIM_AIM/api/v1.0/Locations?$top=100&$skip=0" \
  -H 'accept: application/json'

Request URL
https://sta-demeter.apps.paas-dev.pssc.pl/GE080N-BES-SIM_AIM/api/v1.0/Locations?$top=100&$skip=0

Server response
Code    Details
200

Response body
{
  "@iot.count": 15120,
  "@iot.nextlink": "https://sta-demeter.apps.paas-dev.pssc.pl/GE080N-BES-SIM_AIM/api/v1.0/Locations?$top=100&$skip=100",
  "value": [
    {
      "location": {
        "coordinates": [
          72.5,
          42.5
        ],
        "type": "Point"
      },
      "description": "Location with latitude=72.5 and longitude=42.5",
      "name": "lat=72.5 long=42.5",
      "encodingType": "application/vnd.geo+json",
      "@iot.id": "lat=72.5;long=42.5",
      "@iot.selflink": "https://sta-demeter.apps.paas-dev.pssc.pl/GE080N-BES-SIM_AIM/api/v1.0/Locations(lat=72.5;long=42.5)",
      "Things@iot.navigationLink": "https://sta-demeter.apps.paas-dev.pssc.pl/GE080N-BES-SIM_AIM/api/v1.0/Locations(lat=72.5;long=42.5)/Things",
      "HistoricalLocations@iot.navigationLink": "https://sta-demeter.apps.paas-dev.pssc.pl/GE080N-BES-SIM_AIM/api/v1.0/Locations(lat=72.5;long=42.5)/HistoricalLocations"
    }
  ]
}

```

Figure 47. STA locations for the global trends in biodiversity (BES-SIM AIM)

6 CONCLUSION

This document presented the results of the first iteration regarding the AD4GD building blocks addressing the semantic interoperability aspects in a GD Data Space. This include:

- Green Deal Information Model (GDIM): a common vocabulary aiming at providing the basis of a common green deal data space and enabling the interoperability of different systems, potentially from different vendors, as well as the integration of data collected from various heterogeneous sources in order to provide an integrated view on top of them. Based on previous results, the model reuses and aligns relevant standard ontologies and vocabularies, and is implemented in a layered and modular architecture facilitating its extension and reuse.
- Methods and tools for data harmonisation and integration, which will support service and data providers in the generation of data that is aligned with the GDIM. These include i) the Data Preparation and Integration(DPI) Pipelines that relies on the adoption of Linked Data as a federated layer, combined with the use of knowledge graph technologies, where data is made available and combined according to common ontologies/vocabularies (e.g., GDIM); ii) the OGC Data Exchange Toolkit which includes a set of software tools, templates, and pipeline definitions for documenting and validating modular data models and implementations; iii) the semantic annotation of Tables and its relation with STA supported by TAPIS, a SensorThings API plus reader capable to generate extended JSON schema and the metaschema when exporting a dataset into a GeoJSON file.



- Green Deal Data Space APIs, which include an overview of spatial standards with OGC APIs, the relation between OGC APIs and semantics and the relation to other data spaces' building blocks.

For each of the components realising the building blocks, the document includes references to the corresponding repositories, and examples of how they have been used and applied in the AD4GD pilots.

As part of the future work, we will continue refining and extending the GDIM based on the requirements of the project pilots, which will provide during the next period more concrete results and datasets. Based on these we will be able to determine what additional terms should be included in GDIM, including concepts, properties and relations. Additionally, we will continue with the representation of Essential Variables as controlled vocabularies available via the OGC rainbow server, updating the existing ones (particularly as part of our collaboration with the GEO BON team), and implementing others (e.g., Essential Climate Variables, Essential Water Variables). Similarly, we plan to leverage and align other relevant vocabularies like EIONET, and the M3-lite ontology. Regarding the tools and methods for data harmonisation and integration, the main goal is to make them easily accessible and usable by data and service providers, so that they can be widely applied by the pilots and beyond, as well as enhancing functionalities offered. Finally, regarding the standard OGC APIs, the goal for the second period is to support the pilots and data/service providers to expose their data via the proposed APIs, either by leveraging the tools offered, or by implementing them in their services.

References

[1] European Union (2017). "New European Interoperability Framework (EIF) – Promoting seamless services and data flows for European public administrations". URL: https://ec.europa.eu/isa2/sites/isa/files/eif_brochure_final.pdf.

[2] Curry E. (2020) Fundamentals of Real-time Linked Dataspaces. In: Real-time Linked Dataspaces. Springer, Cham.

[3] LEHMANN, Anthony et al. GEO community activity report : Mainstreaming EVs across GEO. 2023 <https://doi.org/10.13097/archive-ouverte/unige:166309>

[4] Lehmann, A., Mazzetti, P., Santoro, M., Nativi, S., Maso, J., Serral, I., Spengler, D., Niamir, A., Lacroix, P., Ambrosone, M., McCallum, I., Kussul, N., Patias, P., Rodila, D., Ray, N., Giuliani, G., 2022. Essential Earth Observation Variables for high-level multi-scale indicators and policies. Environmental Science and Policy. <https://doi.org/10.1016/j.envsci.2021.12.024>